universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Initiation of Filopodia“

verfasst von / submitted by

Yolanda Einfalt, BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2019 / Vienna 2019

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 066 821
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Mathematik
Betreut von / Supervisor:	Univ.-Prof. Dr. Christian Schmeiser

Contents

1 Acknowledgement

I would like to thank my advisor Christian Schmeiser, who introduced me to the fascinating word of cell biology and proposed such an exciting topic for my master thesis. I want to thank him for the patience with which he answered all my questions and for his continuous support while we were working on this model.

2 Introduction

2.1 Abstract

My master thesis is about initiation of filopodia. So the first question that comes up is, what are filopodia. To answer this question we have to take a look at the domain of cell biology. So in the beginning of my paper I will present some biological informations about filopodia. First we will find out that these are some kind of finger like protrusions, which mainly consist of actin filaments, that grow out of procaryotic cells. These protrusions do not have to stay at their growing position on the cell membrane, but they can move. I will present some real pictures in which we can see their movement. Further I will give some examples of their functions, to answer the question, why do cells produce such protrusions. And in the end of the following section I want to mention some proteins, that are involved in the initiation of filopodia.

The second part of this paper is about a mathematical model that describes the process of initiation. That filopodia can occur, first some actin filaments have to be bundled together. These bundles move along the cell membrane where they catch more filaments or fuse with another bundle. That‘s how the bundles get thicker until they are strong enough to bend the membrane and grow further.

In Section 4 I present a mathematical model which was formulated by my advisor Christian Schmeiser and me. This model describes where on the membrane new bundles arise and how they will move. The model I will present will be continuous, but we wanted to implement it on Matlab to get some visualizations. For this we needed a discrete model, so in Subsection 4.1 the discretization of the model is done. Further I will state in this part the steps how an algorithm would look like, that describes our model.

Next I will present some dimensional analysis, where the equations in our model are scaled to reduce the number of parameters, that can be chosen freely. After this we get some simpler equations, which I used to implement the model on Matlab.

In the end, in Subsection 4.3, I present some pictures, created on Matlab, that visualize our model. In these pictures we can see how existing bundles move along the interval, and how new bundles behave. The Matlab code I used to produce these pictures can be found in the appendix.

2.2 Zusammenfassung

In meiner Arbeit geht es um das Entstehen von sogenannten Filopodien. Im ersten Abschnitt wird der biologische Hintergrund erklärt. Hierbei erfährt man zuerst, dass Filopodien dünne Fortsätze sind, die aus einer Zelle, genauer gesagt einer prokariotischen Zelle, heraus wachsen können. Der wichtigste Baustein dieser Fortsätze sind Actin Filamente, die man in manchen Aufnahmen von Filopodien auch sehr gut erkennen kann, wie man in einem Bild später sehen wird können. Diese dünnen Zellfortsätze sind nicht an einen Ort gebunden, sondern können sich entlang der Zellmembran bewegen. Im Abschnitt 3 zeige ich einige Bilder, wobei in einem davon sehr gut die Bewegung der Filopodien erkennbar ist. Danach widme ich mich kurz der Frage, aus welchem Grund diese Fortsätze überhaupt entstehen und gebe ein paar unterschiedliche Beispiele ihrer Funktionen. Im letzten Abschnitt des folgenden Kapitles gehe ich noch auf einige Proteine ein, die im Entstehungsprozess der Filopodien beteiligt sind.

Nach diesem Abschnitt hat man ausreichend biologisches Wissen gesammelt, um zu Kapitel 4 überzugehen. Hier wird ein mathematisches Model zur Entstehung von Filopodien präsentiert. Damit diese Fortsätze wachsen können, müssen zuerst Actin Filamente zu Bündeln zusammengefasst werden. Diese Bündel können sich entlang der Membran bewegen, wodurch sie auf ihrem Weg weitere Filamente einfangen bzw. falls sie auf ein anderes Bündel treffen, mit diesem verschmelzen. Auf diese Art werden die Bündel immer dicker, bis sie stark genug sind, die Membran zu verbiegen und darüber hinaus zu wachsen, wodurch Filopodien entstehen.

Das mathematische Model, das mein Betreuer Christian Schmeiser und ich aufgestellt haben, beschreibt das Entstehen neuer Bündel und wie sie sich entlang der Membran bewegen. Um eine Visualisierung von diesem Prozess zu erhalten, wollten wir dieses Model auch in Matlab implementieren. Da

allerdings unser ursprüngliches Model stetig ist, musste es erst diskretisiert werden, was ich im Abschnitt 4.1 beschreibe. Hier gehe ich auch auf die einzelnen Schritte ein, wie ein Algorithmus für unser Model auszusehen hat.

Danach gehe ich zu einem Abschnitt über, in dem ich mich der Dimensions Analyse widme. Dort werden die Gleichungen aus unserem Model durch richtiges Skalieren vereinfacht und durch Zusammenhänge zwischen den Parametern, wird die Anzahl der frei zu wählenden Parameter reduziert. Diese vereinfachten Gleichungen habe ich im weiteren in meinem Matlab Programm verwendet.

Im letzten Abschnitt 4.3 präsentiere ich die erlangten Resultate. Hier zeige ich die mit Matlab entstandenen Bilder, in denen man erkennen kann, wie sich die Bündel entlang der Membran bewegen und wie sich neu entstandene Bündel unterschiedlich verhalten, je nachdem an welcher Membran Position sie aufgetaucht sind. Den Code, den ich dafür in Matlab verwendet habe, kann man im Anhang dieser Arbeit nachlesen.

3 Biological Background

3.1 What are filopodia?

To find out what filopodia are, we have to enter the world of cell biology. The biological informations in the following are mostly taken from [1], if not stated differently.

We differ between two classes of cells: the prokaryotes and the eukaryotes. The mayor difference between these two is that the second have a nucleus while the first don‘t have one. Prokaryotes are for example found in bacteria and eukaryotes are for example cells in animals or plants.

Let‘s take a closer look at this second class of cells. The basic structure of such cells is that the nucleus inside the cell is surrounded by the cytoplasm which is covered by the membrane. Inside the cell one can find different proteins which all play different roles. A protein I want to focus on is the structure protein actin, because it is very important in the theory of filopoida. Actin can be aligned to form the so called actin filaments. These filaments are polar that means one can differ between the two ends of the filaments. The fast growing end is always pointing towards the membrane of the cell. This end is called barbed end, while the end which points inside is called pointed end.

If for example a cell wants to move, it cannot spare actin filaments. At the front part of the cell - meaning that side it wants to move - a branched network of actin filaments exist. This network is a sheet like protrusion which is called lamellipodium. With the help of some proteins, some of the actin filaments in the lamellipodium can be bundled together. By elongation of these bundles thin finger-like protrusions can grow out of the lamellipodium. These protrusions are called filopodia.

In the following picture one can see the initiation of filopodia.[2] The bar in the corner of the first box is $2\mu m$ long. One can see pictures of the lamellipodium taken in time steps of 20 seconds. In the first box are already three young filopodia, which are marked by the tall arrows. After some time, two of them moved closer together and fused to one, that‘s why we see in the second box just two filopodia. Separately at other positions some actin filaments are bundled together, which is marked by the small arrows. Right now these bundles are too small to call them filopodia, but they could become one if they bundle more filaments and get strong enough to resist the

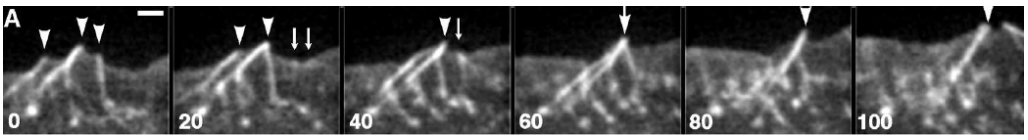

Figure 1: Lamellipodium and Filopodia, [2]

tension of the cell membrane to elongate further. After another 20 seconds the two remaining filopodia also fused to one, exactly like the two bundles. In the following box the bundle is already caught by the filopodium, which moves along the membrane for the last 40 seconds.

In Figure 2 one can see a close up view of filopodia.[2] The bars in the upper corner of box A and B are $0.2\mu m$ long and the grey sticks are actin filaments. In box A one can see that a filopodium consists of parallel bundles of actin. At the bottom is the root of the filopodium where one can see the filaments of the bundle before they were tightly bundled together. In the small box in A there is a close up view of the lamellipodium in the background of the filopodium. In comparison with the filaments of the root of the filopodium one can recognize that the filaments in the lamellipodium are much shorter than in the root. That‘s because there are some proteins, that support the filaments in the filopodium such that they can‘t be capped. While the filaments outside can be capped, that means a protein binds to them and stop their growth.

In box B of Figure 2 another filopodium is shown. But this one consists of two sub-bundles, that fused to one. So first at two different places some of the actin filaments were bundled together. These bundles moved along the membrane - one to the right, the other one to the left, until they met some time ago and fused to one.

The box C is a close up view of the root of the second filopodium. Here one can see the network of the actin filaments. The small circles mark the positions where new filaments arose. Actin filaments are not like fix components, but they can arise from each other with the help of some proteins. So that‘s how they build a whole network of actin.

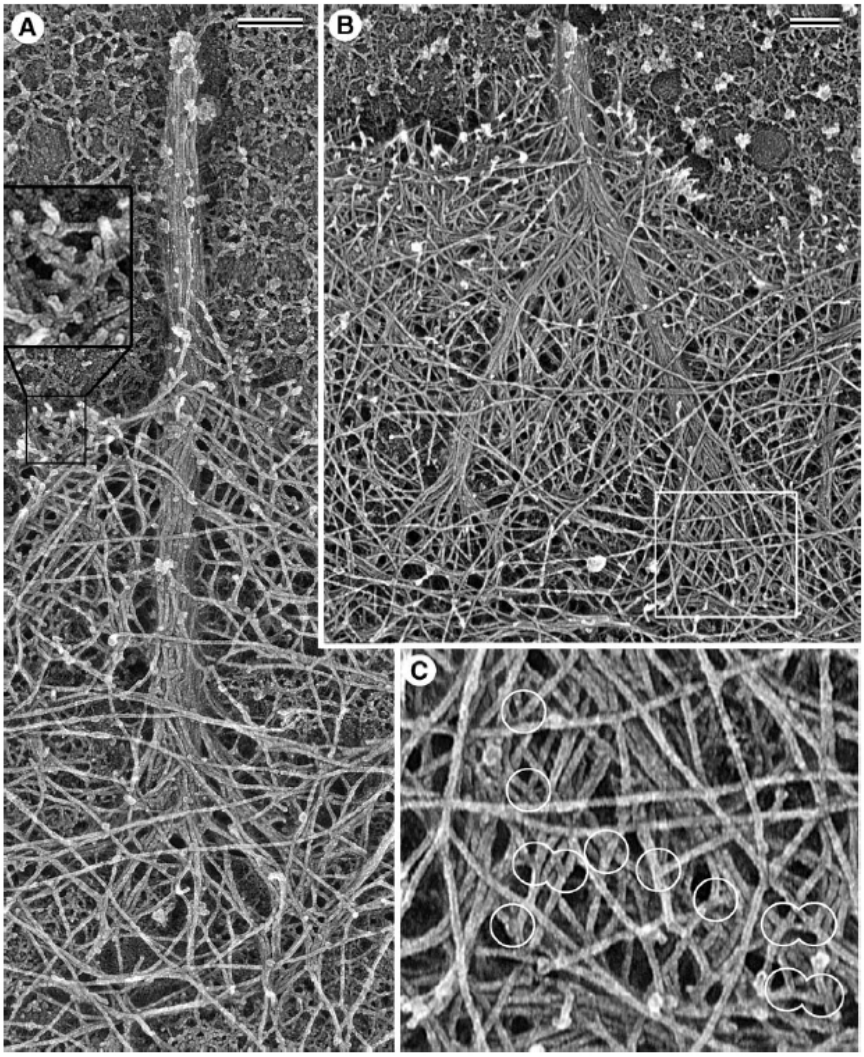

Figure 2: Filopodia, [2]

3.2 Functions of filopodia

So now is clear what are fiopodia, but why do cells produce such protrusions? One can find out that they have different functions and are important for cells in many ways.[1]

If for example a cell wants to change its position, it cannot spare filopodia. The cell is fixed by adhesion to the substrate in it‘s environment. At the leading edge of the cell exists the lamellipodium out of which some filopodia can grow. These can analyse their environment and if they find a proper place, they can bind to it. After that the filopodia retract themselves such that the cell body is stretched and the front part is moved forward. Now the cell breaks its adhesion at the back and the whole body is moved forward. That‘s how a cell is able to move.

So filopodia are very important for the movement of a cell but they also have other functions. If for example a cell wants to adhere to another cell it also uses filopodia. Both cells generate these protrusions at that part of the cell, where they two meet, and fix the contact with help of these filopodia. This adhesion is much stronger than it would be if just the smooth membranes would adhere together.

Another process where filopodia are needed is wound healing. Macrophages produce these protrusions which are used like antennae that explore the environment of the cell. If one finds a pathogen it binds to it and starts to retract itself. Because of this it pulls the pathogen into the cell, where it can be transformed.

There are many more functions of filopodia but there are still open questions and not all of the biological functions are completely understood.

3.3 Proteins involved in initiation process

Now let‘s take a closer look at the initiation of filopodia. What else is needed, beside actin, that these finger like protrusions can occur? The process of the formation is the result of an interaction of many proteins which all support the initiation in different ways. I want to mention some of this proteins and give a short summary of their functions.

• *ARP2/3:* This protein induces branching of the actin filaments. It binds to a side of the barbed end of an actin filament from where a new filament can grow. The fast growing end of this new filament is pointing away from the first one. So if ARP2/3 binds to an actin filament which grows to the left, the generated one grows to the right.

• *ENA/VASP:* The ENA/VASP proteins are a whole family which affect filopodia formation. First of all they protect filaments from capping. So if they are bound to an actin filament the capping protein has no chance of binding and capping them.

On the other hand they also support the elongation of filopodia, so it‘s possible to get long protrusions.

Further they have an anti-branching activity, that means they protect the filaments from ARP2/3. This is important, because otherwise it wouldn‘t be possible to get parallel bundles of actin filaments. If they would be able to branch, many new filaments could arise inside the filopodium, which grow in the completely wrong direction.

And at last these proteins have the ability to bundle actin filaments. This is important because one filament alone would never be strong enough to build filopodia. The force of the polymerization of one filament is too weak to bend the membrane. Only if more filaments are bundled together they can deform the membrane. ENA/VASP proteins are good bundler of single actin filaments but the linkage isn‘t strong enough to resist the tension of the cell membrane. So another protein is needed that filopodia can occur:

• *Fascin:* This protein is the major cross linker of actin filaments. It‘s not efficient for bundling single filaments together, but if they are loosely linked together, fascin increases the connection. So with the help of this protein stiff and parallel bundles of actin are created.

• *IRSp53:* The next protein I want to mention is IRSp53. This has different effects on filopodia formation especially in connection with other proteins. But one important function is its ability to induce the membrane deformation. The shape of this protein looks like a Zeppelin and with this form it can bend the membrane. This helps the filament bundles to grow further and build filopodia.

• *Dia2:* Dia2 has also different roles in filopodia formation. If one compares Dia2 with ENA/VASP proteins, one can find out that they have partially overlapping functions in filopodia formation. Like ENA/VASP it supports for example filament elongation. And further it competes with the capping protein and has therefore an uncapping activity. [3] That means it can remove the capping proteins from filaments, in contrast to ENA/VASP which can just protect filaments from being capped.

• *Myosin-X:* And at last I want to mention Myosin-X. This protein consists of two parts which have different properties - the motor domain and the tail. Because of its motor domain this protein is able to walk along the filaments. But only in one direction - from the pointed end inside the cell to the barbed end at the cell membrane. On the tail of Myosin-X other proteins can bind to, so they can be transported elsewhere.

For filopodia they are useful, especially to get long filopodia. Because of the two domains of Myosin-X, important proteins for the growth of filopodia can bind to the tail of Myosin-X, before it walks, because of its motor domain, to the tip of the filopodium. So that‘s how the components for filopodia elongation are transported to the growing end on the tip. To get back inside the cell, the motor domain of Myosin-X is useless, because it can only walk in one direction. But these proteins don‘t have to stay at the tip of the filopodium. They just have to stick to a filament. Because of polymerization at the barbed end, that part of the filament where the protein is fixed, is pushed backward inside the cell. There it can again search for important proteins.

In Figure 3 one can see how these proteins work together to form a filopodium. Because of ARP2/3 new actin filaments arise from old one and a whole network of actin is build. Some filaments come together and are bundled by ENA/VASP proteins. The membrane deformation is induced by IRSp53 and therefore the filaments can elongate further. With the help of fascin the connection of the filaments is increased such that a stiff bundle is created. So the filaments in the bundle can grow further and build a filopodium. And because of Myosin-X all the needed components for elongation of the filopodium are transported to the tip.

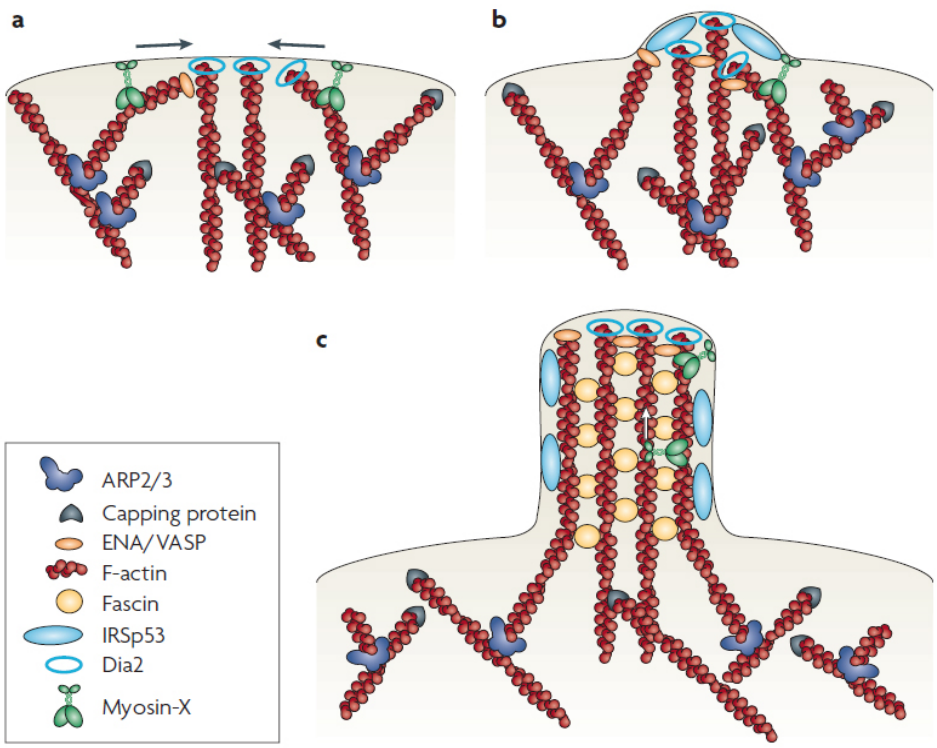

Figure 3: Initiation of a filopodium, [1]

Of course there are more proteins involved in this process. There are consistently new studies which discover the influence of another protein to filopodia formation or show how the proteins influence each other. So right now not all parts of the process are known and there are still open questions.

4 Mathematical Model

Now let's take a look at the mathematical part of this work. The task is to describe a model for the initiation of filopodia. That means we want a model that tells us where on the membrane new bundles will arise and how they will move. For simplicity let's just take ARP2/3 and the capping protein into our model, such that we get branching and capping. But I'm sure one can extend the model to get step by step an influence of other proteins as well.

Let's consider that part of the cell membrane, we want to look at, as an interval $[0, L]$. On this interval there are ends of filaments - some of them point to the right and some of them point to the left. Let's denote the density of the right, respectively the left pointing filaments by

$$\rho_r(x,t)\ ,\ \rho_l(x,t) \quad for \quad x \in [0,L]\ ,\ t \in [0,T]$$

where x is a position on the membrane and t gives us the time. That means

$$\int_a^b \rho_r(x,t)dx$$

gives us the number of right pointing filaments at time t in the interval $[a, b]$.

The positions on the membrane of the bundle ends are denoted by

$$X_1(t), ..., X_{N(t)}(t)$$

where $N(t)$ is the total number of bundles on the interval. This number is time dependent because any time new bundles can arise or two bundles can fuse to one.

Further we need the number of right and left pointing filaments in a bundle, which is denoted by

$$r_1(t), ..., r_{N(t)}(t) \quad and \quad l_1(t), ..., l_{N(t)}(t).$$

These are important because they tell us how the bundle will look like. If the same number of right and left pointing filaments are bundled together, the bundle will grow vertically against the membrane. But if for example more right pointig filaments are caught, the bundle itself will also point to the right.

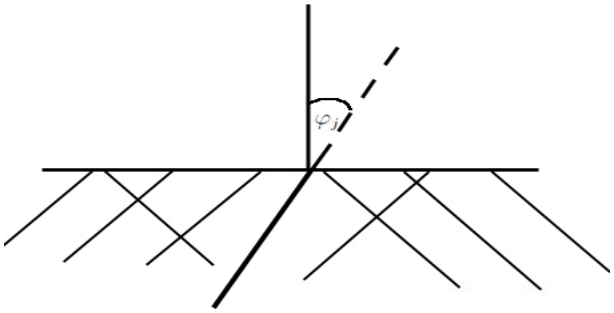

Figure 4: Angle of a bundle

The angles at which the bundles grow against the membrane are denoted by

$$\varphi_1(t), ..., \varphi_{N(t)}(t)$$

while we assume the angle of the filaments that are not caught in a bundle is equal for every filament, namely

$$\varphi_0 = 35°.$$

Like I said before, the direction in which the bundle grows is dependent on the filaments, that are caught. That means the filaments that are bundled together, fix the angle of the bundle. So if the same number of right and left pointing filaments are bundled, the angle is 0. The other extreme case is, that only one kind of filaments are bundled, for example only right pointing filaments. In this case the angle is 35°. So the angle of a bundle is always between 0 and 35°, respectively $-35°$ if only left pointing filaments are bundled.

In Figure 4 we see a visualization of the angle φ_j of a bundle that grows to the right.

Further we need some parameters to describe our model:

- v_p ... the polymerization speed of the filaments. That‘s the speed at which the filaments grow.

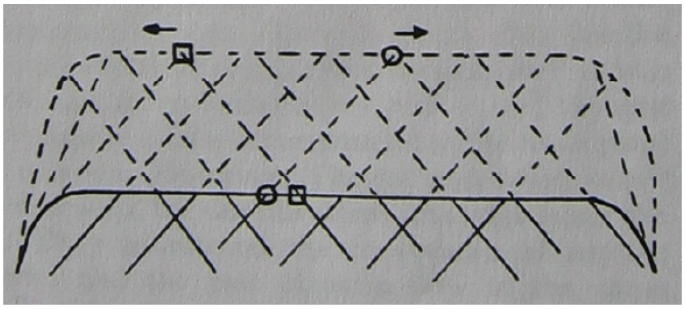

Figure 5: Lateral Flow, [4]

- v_{lat} ... the lateral-flow speed of the filaments. That is the speed at which the filaments move along the membrane because of polymerization. In Figure 5 one can see how the filaments move. It's a drawing of a cell membrane with some right and left pointing filaments inside. Two filament ends are marked at the membrane. With some time these filaments grow further, which is visualized by the dashed lines, but they are too weak to bend the membrane, so the pointed end of the filament is pushed backward inside the cell. After some time the new filament ends are again marked at the membrane, where one can see that the filament end changed it's position. That's the lateral flow. So the filaments move along the membrane because of polymerization. That means the lateral flow depends on the polymerization speed and we obtain: $v_{lat} = v_p sin(\varphi_0)$.

- $v_b(\varphi_j)$ ... the polymerization speed of the bundles. This depends on the angle at which the bundle grows against the membrane. The greater the angle, the faster it can grow.

- κ_{cap} ... a capping rate. The rate at which the filaments are capped.

- κ_{br} ... a branching rate. The rate at which new filaments arise through branching.

- κ_b ... a bundling rate. The rate at which the free filaments are caught by a bundle.

Let's assume $\kappa_{cap} < \kappa_{br}$ because if more filaments would be capped than new one arise through branching, we would get no filopodia.

With these we can now start to develop a model. Let‘s first describe the density of the filaments on the interval $[0, L]$. If we don‘t have any capping or branching and no bundles are around, we get this equations:

$$\begin{aligned} \partial_t \rho_r + \partial_x(\rho_r v_{lat}) &= 0 \\ \partial_t \rho_l - \partial_x(\rho_l v_{lat}) &= 0. \end{aligned} \tag{1}$$

The right pointing filaments walk along the membrane to the right, so the lateral flow speed is positive, while the left pointing filaments move to the left, so the lateral flow is negative.

If the number of filaments change because of branching and capping, the right hand side of the equation changes and we get these equations:

$$\begin{aligned} \partial_t \rho_r + \partial_x(\rho_r v_{lat}) &= -\kappa_{cap}\rho_r + \frac{\kappa_{br} c_{rec} \rho_l}{c_{rec} + \kappa_{br}(\rho_r + \rho_l)} \\ \partial_t \rho_l - \partial_x(\rho_l v_{lat}) &= -\kappa_{cap}\rho_l + \frac{\kappa_{br} c_{rec} \rho_r}{c_{rec} + \kappa_{br}(\rho_r + \rho_l)} \end{aligned} \tag{2}$$

which can be compared to the model in [4]. The first term on the right hand side describes the capping part, that means the number of filaments is reduced by the capping rate. Further we get more filaments because of branching, which is captured in the second part. This part clearly has to depend on the branching rate and for the right pointing filaments it depends on the number of left pointing filaments and vice versa. That‘s because new right pointing filaments for example arise through an ARP2/3 protein that binds to a side of a left pointing filament. The constant c_{rec} gives us the influence of ARP2/3.

If there are already bundles, they also influence the density of filaments, because some filaments are caught by the bundles, so we need a jump condition at these positions. To get these let‘s take a look at the density of the filaments while we assume we have a bundle at position X_1. For simplicity let‘s for a moment assume $\kappa_{br} = \kappa_{cap} = 0$. Let‘s first consider the right pointing filaments, for which we get:

$$\partial_t \rho_r + v_{lat}\partial_x \rho_r = -\kappa_b(v_{lat} - \dot{X}_1)\delta(x - X_1)\rho_r(x-). \tag{3}$$

The right hand side of the equation which describes the bundling is clearly depending on the bundling rate, on the speed of the filaments and the bundle

and on the number of filaments that reach the bundle. The right pointing filaments will reach the bundle from it‘s left, so we need the limit from the left.

For simplicity let‘s consider the position of the bundle $X_1 = 0$. For this we get:

$$\partial_t \rho_r + v_{lat} \partial_x \rho_r = -\kappa_b v_{lat} \delta(x) \rho_r(x-).$$

Let‘s take a closer look at the area of the position of the bundle. So let‘s integrate over the interval $(-\epsilon, \epsilon)$ for $\epsilon > 0$:

$$\frac{d}{dt} \int_{-\epsilon}^{\epsilon} \rho_r dx + v_{lat}(\rho_r(\epsilon) - \rho_r(-\epsilon)) = -\kappa_b v_{lat} \rho_r(0-).$$

If we reduce the size of the interval by $\epsilon \to 0$ we get:

$$\begin{aligned} & v_{lat}(\rho_r(0+) - \rho_r(0-)) = -\kappa_b v_{lat} \rho_r(0-) \\ & \Leftrightarrow v_{lat} \rho_r(0+) = (v_{lat} - \kappa_b v_{lat}) \rho_r(0-) \\ & \Leftrightarrow v_{lat} \rho_r(0+) = v_{lat}(1 - \kappa_b) \rho_r(0-) \\ & \Leftrightarrow \rho_r(0+) = (1 - \kappa_b) \tilde{\rho} \end{aligned}$$

where $\tilde{\rho}$ gives the initial rate of ρ_r. So we have found the first jump condition for the density of the filaments at a bundle position.

Next let‘s take a look at the left pointing filaments, while there exists a bundle at position X_1:

$$\partial_t \rho_l - v_{lat} \partial_x \rho_l = -\kappa_b \cdot (-v_{lat} - \dot{X}_1) \delta(x - X_1) \rho_l(x+). \tag{4}$$

Here we need the limit from the right for the density of the filaments that reach the bundle, because the left pointing filaments move to the left, so they will reach the bundle from its right.

If the bundle is again at position $X_1 = 0$:

$$\partial_t \rho_l - v_{lat} \partial_x \rho_l = -\kappa_b \cdot (-v_{lat}) \delta(x) \rho_l(x+).$$

By integrating we get:

$$\frac{d}{dt} \int_{\epsilon}^{-\epsilon} \rho_l dx - v_{lat}(\rho_l(-\epsilon) - \rho_l(\epsilon)) = \kappa_b v_{lat} \rho_l(0+)$$

which leads to the following if $\epsilon \to 0$:

$$\begin{aligned}
&- v_{lat}(\rho_l(0-) - \rho_l(0+)) = \kappa_b v_{lat}\rho_l(0+) \\
&\Leftrightarrow -v_{lat}\rho_l(0-) = (-v_{lat} + \kappa_b v_{lat})\rho_l(0+) \\
&\Leftrightarrow -v_{lat}\rho_l(0-) = v_{lat}(-1 + \kappa_b)\rho_l(0+) \\
&\Leftrightarrow \rho_l(0-) = (1 - \kappa_b)\tilde{\rho}
\end{aligned}$$

if we assume that the initial rate for the left pointing filaments is also $\tilde{\rho}$.

Now the question is, do these jump conditions change if there are more than one bundle on the interval? To answer this let‘s consider a second bundle at position X_2. If both bundles arise roughly at the same time and if the space between them is big enough, they do not influence each other in the beginning. Both get the initial rate of filaments from right and left. After some time the situation is changed. Let‘s say the second bundle position is to the right of the first one, then this bundle will get at some point less filaments that point right, because some of them are already bundled by the first one.

So let‘s take a look at the neighbourhood of the second bundle. First we have:

$$\begin{aligned}
&\partial_t \rho_r + v_{lat}\partial_x \rho_r = \\
&= -\kappa_b(v_{lat} - \dot{X}_1)\delta(x - X_1)\rho_r(x-) - \kappa_b(v_{lat} - \dot{X}_2)\delta(x - X_2)\rho_r(x-).
\end{aligned} \tag{5}$$

If we integrate over (a, b), where $a < X_2 < b$, we get:

$$\frac{d}{dt}\int_a^b \rho_r dx + v_{lat}(\rho_r(a) - \rho_r(b)) = -\kappa_b(v_{lat} - \dot{X}_2)\rho_r(X_2-).$$

If we look at the limit $a \nearrow X_2$ and $b \searrow X_2$ the first term does not vanish like before, because:

$$\frac{d}{dt}\int_a^b \rho_r dx = \frac{d}{dt}\left(\int_{X_2(t)}^b \rho_r dx + \int_a^{X_2(t)} \rho_r dx\right) =$$
$$= \int_{X_2(t)}^b \partial_t \rho_r dx - \dot{X}_2 \rho_r(X_2+) + \int_a^{X_2(t)} \partial_t \rho_r dx + \dot{X}_2 \rho_r(X_2-) =$$
$$= \int_a^b \partial_t \rho_r dx - \dot{X}_2 \rho_r(X_2+) + \dot{X}_2 \rho(X_2-)$$

and for $a \nearrow X_2$ and $b \searrow X_2$:

$$\frac{d}{dt}\int_a^b \rho_r dx = 0 - \dot{X}_2 \rho_r(X_2+) + \dot{X}_2 \rho_r(X_2-) =$$
$$= \dot{X}_2(\rho_r(X_2-) - \rho_r(X_2+)).$$

So the limit of the interval of the whole equation will lead to:

$$\begin{aligned}
&\dot{X}_2(\rho_r(X_2-) - \rho_r(X_2+)) + v_{lat}(\rho_r(X_2+) - \rho_r(X_2-)) = \\
&= -\kappa_b(v_{lat} - \dot{X}_2)\rho_r(X_2-) \\
&\Leftrightarrow (\rho_r(X_2+) - \rho_r(X_2-))(v_{lat} - \dot{X}_2) = -\kappa_b(v_{lat} - \dot{X}_2))\rho_r(X_2-) \\
&\Leftrightarrow \rho_r(X_2+) - \rho_r(X_2-) = -\kappa_b \rho_r(X_2-) \\
&\Leftrightarrow \rho_r(X_2+) = (1-\kappa_b)\rho_r(X_2-)
\end{aligned}$$

where $\rho_r(X_2-) = \rho_r(0+)$.

That means if there are more bundles on the interval, the jump condition for the density of the filaments stays the same.

With analogous computations we will get the jump conditions for the left pointing filaments at the second bundle:

$$\rho_l(X_2-) = (1-\kappa_b)\rho_l(X_2+).$$

So no matter how many bundles there are, at each bundle position X on the interval, we get these jump conditions:

$$\begin{aligned}
\rho_r(X+) &= (1-\kappa_b)\rho_r(X-) \\
\rho_l(X-) &= (1-\kappa_b)\rho_l(X+).
\end{aligned} \tag{6}$$

The number of filaments that pass by is reduced by the bundling rate. So like I mentioned before, on the right side of a bundle will be less right pointing filaments than on its left side. And vice versa on the left side of a bundle will be less filaments that point left.

Furthermore the filaments always catch up with the bundle, so the bundle can never run away of the filaments. That‘s because the angle of the filaments is always greater than the angle of the bundle, so they move faster. The angle of the filaments and the bundle cannot be equal for a long time because the bundle will get filaments from the other side, which push it in the other direction, such that the angle gets smaller again.

Now we know the density of the filaments on the whole interval, so let‘s next take a look at the bundles. The position of a bundle changes over time because it moves along the membrane because of polymerization. So just like for the lateral flow of the filaments, we get the change of the position of a bundle depending on the polymerization speed:

$$\dot{X}_j = v_b(\varphi_j) sin\varphi_j. \tag{7}$$

Further also the number of filaments in the bundle is changing because it bundles always some more. So we get:

$$\begin{aligned} \dot{r}_j &= \kappa_b(v_{lat} - \dot{X}_j)\rho_r(X_j-) \\ \dot{l}_j &= \kappa_b(-v_{lat} - \dot{X}_j)\rho_l(X_j+). \end{aligned} \tag{8}$$

These equations clearly depend on the amount of filaments that reach the bundle and on the bundling rate, but also on the speed - so how far the bundle gets.

Next we also need an equation which gives us the angle of a bundle, because depending on which filaments are caught the angle gets greater or smaller. We choose:

$$\varphi_j = \varphi_0 \frac{(r_j - l_j)}{r_j + l_j} \tag{9}$$

where φ_0 is the angle of the filaments. And we take $(r_j - l_j)$ because the right pointing filaments will push the bundle to the right, so the angle gets greater, while the left pointing filaments push it to the left, so the angle gets

smaller.

Last but not least we have to describe how new bundles arise. A Poisson process with parameter λ gives us the random arising of new bundles, where

$$\lambda = v_n \int_0^L \rho_r \rho_l dx. \tag{10}$$

The parameter v_n should not be too great, otherwise we would get too many bundles, but it also shouldn‘t be too less, because if there would arise too less bundles, nothing interesting could be seen.

With this process we know now when a new bundle is arising, so what is left to show is where on the membrane this new bundle will be. This is given by the probability density f:

$$f(x) = \frac{\rho_r(x)\rho_l(x)}{\int_0^L \rho_r(y)\rho_l(y)dy}. \tag{11}$$

If at some part of the interval the value $\rho_r \rho_l$ is greater than at the rest, it‘s more likely that a new bundle will arise at this place, because if there are more filaments it‘s more likely that some of them will be bundled together. If at this position a new bundle arises, some of the filaments are bundled, so the total number in this area is decreased. That means it‘s more likely that the next bundle will arise elsewhere.

4.1 Algorithm

Until now we described a continuous process, but to implement it, we have to discretize it. Let‘s consider a coordinate system with the position on the membrane on the x-axis and the time on the y-axis. In the end we want to see in this coordinate system at each time where on the membrane the bundle ends are.

We fix N positions in the interval $[0, L]$ at which the density of the filaments are known. So N positions are set on the x-axis with fixed distance Δx. After one time step Δt the positions are shifted by $\frac{\Delta x}{2}$ and in the following time step the positions are chosen like in the beginning. So with this procedure a network of positions is fixed. The filaments move along the membrane because of their lateral-flow and the distance of the steps are chosen such that the filaments will move in one time step from one position point

to a position point one time step further. So we choose $\Delta x = \Delta t$. The right pointing filaments move in the first time step from position i to position i and in the second step from position i to i+1. The left pointing filaments move in the first step from position i to i-1 and in the second step from position i to i. So the algorithm is divided into two parts: an "odd" time step and an "even" time step. The odd time step denote the steps that go from time $t = 2n - 1$ to $t = 2n$, so for example the first time step, which goes from time $t = 1$ to $t = 2$. While the even time step denote the steps that go from time $t = 2n$ to $t = 2n + 1$, so for example the second step. But except the indices both parts of the algorithm have the same structure.

1.) New bundles arise

We are at time t_n and test if at this time a new bundle arises. We choose $y \in [0, 1]$ randomly. If

$$y > e^{-\lambda^n \Delta t} \quad \text{for} \quad \lambda^n = v_n \int_0^L \rho_l^n \rho_r^n dx$$

a new bundle arises. For simplicity let's write ρ_r^n for the number of right pointing filaments at time t_n, and $\rho_{r,j}^n = \rho_r(x_j, t_n)$ where x_j is the j-th position point.

The integral can be easily approximated by the following sum

$$\int_0^L \rho_l \rho_r \sim \Delta x \sum_{j=2}^{N} \rho_{l,j} \rho_{r,j} + \frac{\Delta x}{2} \rho_{l,1} \rho_{r,1}$$

if we are in an odd time step. And we get

$$\int_0^L \rho_l \rho_r \sim \Delta x \sum_{j=1}^{N-1} \rho_{l,j} \rho_{r,j} + \frac{\Delta x}{2} \rho_{l,N} \rho_{r,N}$$

for an even time step. So for the approximation of the integral, we just sum over all inner points on the interval, plus the half of the boundary point.

Next we have to fix the position of the new bundle, which is given by the distribution function F_n:

$$F_n(x) = \frac{\int_0^x \rho_l(y,t_n)\rho_r(y,t_n)dy}{\int_0^L \rho_l(y,t_n)\rho_r(y,t_n)dy}$$

where the integrals can be discretized like before.

Now choose a random F_i in the interval $[0,1]$ and with this the position of the new bundle is:

$$X_i = F_n^{-1}(F_i).$$

At this time the new bundle has no filaments caught, so in the beginning we choose $r_i^n = l_i^n = 0$. The first filaments that are added to this bundle, are bundled in the first time step after its arising.

Notice, that the arising of a new bundle will change the number of the bundles. So one has to find out the position of the new one in comparison with the others. The bundles that lie left stay the same, while the bundles that are right will get a new number. If for example the new bundle arise between bundle k-1 and k, it will be in following the k-th bundle, while the old k-th bundle get the number k+1, just like the following that all get a number switched by one.

2.) Change of bundles

To derive the bundle informations after one time step (like position and number of filaments that are caught) we have to find out what will happen in this time step. If two bundles cross, they will fuse to one and grow further as a thicker bundle. So let‘s differ between two cases: if two bundles cross and if they don‘t cross the way of an other bundle. So look at one bundle after the other - starting with the first one. Let‘s say we are at the k-th bundle. Then we check if it‘s possible that this bundle will cross with its right neighbour. This is the case if the distance between these two is small enough, so if $X_{k+1}^n - X_k^n < \Delta x$, where $X_k^n = X_k(t_n)$. The angle of a bundle cannot be greater than 35°, that‘s why two bundles cannot cross in one time step, if their distance is greater than Δx. But still it‘s possible, that these two will not cross in this step, even if they are close enough together. Depending on their angle they will cross in this time step, or not. So we have to compute their cutting point at time t^* and let‘s denote the time until they meet by $\Delta t^* = t^* - t_n$. Only if $\Delta t^* < \Delta t$ these bundles will meet in the following

time step.

2.1) *If the bundle don't cross another bundle*

Let's consider the k-th bundle at time t_n. And assume we have already checked if it crosses the way of its neighbour and found out that they don't meet in the next time step. Consider the position X_k^n of the bundle and find out which position points are to its left and right - so search for a j such that $x_j \leq X_k^n \leq x_{j+1}$, where x_j denotes the j-th position point on the interval $[0, L]$.

After one time step we get the new position of the bundle:

$$X_k^{n+1} = X_k^n + \Delta t g(\varphi_k^n)$$

where $g(\varphi_k^n) := \dot{X}_k(t_n) = v_b(\varphi_k^n) sin(\varphi_k^n)$. So the new position of the bundle is just the old position plus the length of one time step times the change of the position. Analogously we also get the number of filaments after one time step:

$$\begin{aligned} r_k^{n+1} &= r_k^n + \Delta t \kappa_b (v_{lat} - g(\varphi_k^n)) \rho_{r,j}^n \\ l_k^{n+1} &= l_k^n + \Delta t \kappa_b (-v_{lat} - g(\varphi_k^n)) \rho_{l,j+1}^n. \end{aligned}$$

The right pointing filaments that reach the bundle in this time step will come from position x_j so the density of the right pointing filaments is taken from this position, while the left pointing filaments that reach it will come from position x_{j+1}.

With these we can easily compute the new angle of the bundle:

$$\varphi_k^{n+1} = \varphi_0 \frac{r_k^{n+1} - l_k^{n+1}}{r_k^{n+1} + l_k^{n+1}}.$$

2.2) *If two bundles cross*

Let's again consider the k-th bundle at time t_n, but this time assume that it is close enough to its right neighbour to cross its way. Again we write $g(\varphi_k^n) := \dot{X}_k(t_n) = v_b(\varphi_k^n) sin(\varphi_k^n)$ and denote their cutting time by t^*.

Then we obtain:

$$X_k^n + (t^* - t_n)g(\varphi_k^n) = X_{k+1}^n + (t^* - t_n)g(\varphi_{k+1}^n)$$

and therefore:

$$\Delta t^* = (t^* - t_n) = \frac{X_{k+1}^n - X_k^n}{g(\varphi_k^n) - g(\varphi_{k+1}^n)}.$$

If $\Delta t^* < \Delta t$ they cross in the following time step. So let's compute the position of the union at time t^*. This is computed just like before, but this time only a smaller time step Δt^* is made:

$$X^* = X_k^n + \Delta t^* g(\varphi_k^n).$$

To get the number of left and right pointing filaments in the union we need to know from which positions the filaments reach the bundles until t^*. For this we have to find the next position points in the neighbourhood of the bundles. So let's search for j_k and j_{k+1} such that $x_{j_k} \leq X_k^n \leq x_{j_k+1}$ and $x_{j_{k+1}} \leq X_{k+1}^n \leq x_{j_{k+1}+1}$. With these we can now compute the number of filaments in the union at time t^*:

$$\begin{aligned} r_k^* = r_k^n + \Delta t^* \kappa_b (v_{lat} - g(\varphi_k^n))\rho_{r,j_k}^n + \\ + r_{k+1}^n + \Delta t^* \kappa_b (v_{lat} - g(\varphi_{k+1}^n))\rho_{r,j_{k+1}}^n \end{aligned}$$

$$\begin{aligned} l_k^* = l_k^n + \Delta t^* \kappa_b (-v_{lat} - g(\varphi_k^n))\rho_{l,j_k+1}^n + \\ + l_{k+1}^n + \Delta t^* \kappa_b (-v_{lat} - g(\varphi_{k+1}^n))\rho_{l,j_{k+1}+1}^n. \end{aligned}$$

So the number of filaments in the union is just the number of filaments in the first bundle at time t^* plus the number of filaments of the second bundle at time t^*. With these we can compute the angle of this bundle with the equation from before:

$$\varphi_k^* = \varphi_0 \frac{r_k^* - l_k^*}{r_k^* + l_k^*}.$$

At this point we are still not in the end, because we only made a smaller time step Δt^*, so we also have to compute the rest of the time step. The

length of the step which is left is $\Delta t - \Delta t^*$, so we get the position of the union at time t_{n+1}:

$$X_k^{n+1} = X^* + (\Delta t - \Delta t^*) g(\varphi_k^*).$$

The union will be the k-th bundle and because the old bundle with number k+1 has disappeared, the number of the following bundles is reduced by one.

If we denote the position point to the left of the bundle position X^* by x_{j^*}, we derive the number of filaments at time t_{n+1}:

$$\begin{aligned} r_k^{n+1} &= r_k^* + (\Delta t - \Delta t^*)\kappa_b(v_{lat} - g(\varphi_k^*))\rho_{r,j^*}^n \\ l_k^{n+1} &= l_k^* + (\Delta t - \Delta t^*)\kappa_b(-v_{lat} - g(\varphi_k^*))\rho_{l,j^*+1}^n \end{aligned}$$

and with these we can again compute the angle:

$$\varphi_k^{n+1} = \varphi_0 \frac{r_k^{n+1} - l_k^{n+1}}{r_k^{n+1} + l_k^{n+1}}.$$

To make our life easier we will tread this bundle in the following as if it would have always been one bundle and not two. Because of this assumption it will be easier to compute the density of the filaments. So we have to take a look back in time to find out how the bundle would have looked like one time step ago, if two bundles had not crossed. So instead of making one time step further, we make one step back, so the position of the bundle at time t_n would be:

$$\bar{X}_k^n = X^* - \Delta t^* g(\varphi_k^*).$$

If a bundle didn‘t cross with an other bundle in the last step, we get just

$$\bar{X}_k^n = X_k^n$$

3.) Change of the density of the right pointing filaments

To get the change of the filament density we again have to differ between two cases. First, if the filaments have no bundle in their neighbourhood, the

bundles don‘t influence their density. But if they cross the way of a bundle, some of them are caught and added to the bundle, so in this case we also have to consider the jump conditions.

3.1) *If there are no bundles around*

Let‘s consider the equation from before:

$$\partial_t \rho_r + \partial_x(\rho_r v_{lat}) = -\kappa_{cap}\rho_r + \frac{\kappa_{br} c_{rec} \rho_l}{c_{rec} + \kappa_{br}(\rho_r + \rho_l)}.$$

Depending on in which step we are, the indices in the discretization are different. Like I already mentioned before, in every odd time step the right pointing filaments will move from position i to i and in every even step, they will move from i to i+1. Assume we are at time t_n then we get for an odd time step a discretization of the following form:

$$\frac{\rho_{r,j}^{n+1} - \rho_{r,j}^n}{\Delta t} + v_{lat}\frac{\rho_{r,j}^n - \rho_{r,j-1}^n}{\Delta x} = -\kappa_{cap}\rho_{r,j}^{n+1} + \frac{\kappa_{br} c_{rec} \rho_{l,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)}$$

where we used a semi-implicit Ansatz, so the capping part is assumed implicit such that the solution will not be negative. That‘s a good choice because if one would take an explicit Ansatz for the capping part, the equation would be correct only for small κ_{cap}, while with the implicit Ansatz κ_{cap} can be chosen arbitrary.

The second part on the left hand side, that describes the change of the position on the interval, can be neglected. That‘s because the filaments move from one position point to the next, so we look at the solutions along their characteristics. Then the upper equation can be transformed to:

$$\rho_{r,j}^{n+1} = \rho_{r,j}^n + \Delta t\left(\frac{\kappa_{br} c_{rec} \rho_{l,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)} - \kappa_{cap}\rho_{r,j}^{n+1}\right).$$

In an even time step we derive the following equation:

$$\rho_{r,j+1}^{n+1} = \rho_{r,j}^n + \Delta t\left(\frac{\kappa_{br} c_{rec} \rho_{l,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)} - \kappa_{cap}\rho_{r,j+1}^{n+1}\right).$$

So after further transformation we get the density of the right pointing filaments at time t_{n+1} after an odd respectively an even time step:

$$\rho_{r,j}^{n+1} = \frac{1}{1+\Delta t\kappa_{cap}}(\rho_{r,j}^{n} + \Delta t\frac{\kappa_{br}c_{rec}\rho_{l,j}^{n}}{c_{rec}+\kappa_{br}(\rho_{r,j}^{n}+\rho_{l,j}^{n})})$$
$$\rho_{r,j+1}^{n+1} = \frac{1}{1+\Delta t\kappa_{cap}}(\rho_{r,j}^{n} + \Delta t\frac{\kappa_{br}c_{rec}\rho_{l,j}^{n}}{c_{rec}+\kappa_{br}(\rho_{r,j}^{n}+\rho_{l,j}^{n})}).$$

3.2) *If they cross the way of a bundle*

If a bundle, assume the k-th bundle, is in the neighbourhood, it‘s possible that the filaments cross its way. The right pointing filaments move in an odd step from position i to i. So for the filaments that move from (x_j, t_n) to (x_j, t_{n+1}) the following conditions have to be satisfied, that the bundle cross their way:

$$x_j \leq \bar{X}_k^n \leq x_{j+1}$$
$$x_{j-1} \leq X_k^{n+1} \leq x_j.$$

If the first line is not satisfied the bundle starts in an area where its impossible to catch the right pointing filaments, because the angle of the bundle is smaller than the angle of the free filaments, so it will never move faster than they can. If the second condition is not satisfied the bundle moves elsewhere, such that these filaments will not cross its way.

If we are in an even step, the second condition changes to

$$x_j \leq X_k^{n+1} \leq x_{j+1}$$

because the right pointing filaments will move from position (x_j, t_n) to (x_{j+1}, t_{n+1}).

If these conditions are satisfied, the filaments will meet the bundle in this time step. So let‘s first compute the density of the filaments at the cutting point t^*.

The smaller time step Δt^* is computed like in the last section where two bundles were crossing:

$$\Delta t* = \frac{\bar{X}_k^n - x_j}{v_{lat} - g(\varphi_k^n)}.$$

With this we can now derive the density of the filaments at time t^*:

$$\rho_r^* = \frac{1}{1 + \Delta t^* \kappa_{cap}} (\rho_{r,j}^n + \Delta t^* \frac{\kappa_{br} c_{rec} \rho_{l,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)}).$$

At this point some of the filaments are bundled and some move on. So the number of filaments at time t^* is reduced by the bundling rate and the rest of the time step $\Delta t - \Delta t^*$ is made. So we get

$$\begin{aligned}\rho_{r,j}^{n+1} =& \frac{1}{1 + (\Delta t - \Delta t^*)\kappa_{cap}} (\rho_r^*(1 - \kappa_b) + \\ & + (\Delta t - \Delta t^*) \frac{\kappa_{br} c_{rec} \rho_{l,j+1}^n}{c_{rec} + \kappa_{br}(\rho_r^*(1 - \kappa_b) + \rho_{l,j+1}^n)})\end{aligned}$$

for an odd time step. And

$$\begin{aligned}\rho_{r,j+1}^{n+1} =& \frac{1}{1 + (\Delta t - \Delta t^*)\kappa_{cap}} (\rho_r^*(1 - \kappa_b) + \\ & + (\Delta t - \Delta t^*) \frac{\kappa_{br} c_{rec} \rho_{l,j+1}^n}{c_{rec} + \kappa_{br}(\rho_r^*(1 - \kappa_b) + \rho_{l,j+1}^n)})\end{aligned}$$

for an even time step.

4.) Change of the density of the left pointing filaments

The density of the left pointing filaments after one time step we get by following the same ideas like for the right pointing filaments. So we again differ between the case in which there are no bundles around and the case in which some filaments are bundled. But different to the right pointing filaments, the left pointing filaments move to the left. So in an odd time step they move from position i to i-1 and in an even step they move from i to i.

4.1) *If there are no bundles around*

Let's assume we already are at time t_n and want to go to time t_{n+1}. Especially we want to compute the density of the left pointing filaments at position j after one time step. By analogous computations like before we derive the density of the filaments after an odd time step:

$$\rho_{l,j-1}^{n+1} = \frac{1}{1+\Delta t \kappa_{cap}}(\rho_{l,j}^n + \Delta t \frac{\kappa_{br} c_{rec} \rho_{r,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)})$$

and the density after an even time step:

$$\rho_{l,j}^{n+1} = \frac{1}{1+\Delta t \kappa_{cap}}(\rho_{l,j}^n + \Delta t \frac{\kappa_{br} c_{rec} \rho_{r,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)}).$$

4.2) *If they cross the way of a bundle*

To find out if there is a bundle that crosses the way of the filaments in that time step, one has to look at its positions at time t_n and t_{n+1}. If

$$x_{j-1} \leq \bar{X}_k^n \leq x_j$$
$$x_{j-1} \leq X_k^{n+1} \leq x_j$$

the filaments at position j cross the way of the k-th bundle in the following odd time step. If the next time step is even, the second condition changes to:

$$x_j \leq X_k^{n+1} \leq x_{j+1}.$$

If the conditions are satisfied, we found out that they cross the k-th bundle, so like before we have to compute their cutting point at time t^* first. The time step until we reach the cutting point is:

$$\Delta t* = \frac{x_j - \bar{X}_k^n}{v_{lat} + g(\varphi_k^n)}$$

and the density of the filaments at that time is:

$$\rho_l^* = \frac{1}{1+\Delta t^* \kappa_{cap}}(\rho_{l,j}^n + \Delta t^* \frac{\kappa_{br} c_{rec} \rho_{r,j}^n}{c_{rec} + \kappa_{br}(\rho_{r,j}^n + \rho_{l,j}^n)}).$$

Now some of the filaments are added to the bundle, so the number of filaments at this point is reduced by the bundling rate. The rest moves on, so after the smaller time step that is left, we get the density of the filaments at time t_{n+1}:

$$\begin{aligned}\rho_{l,j-1}^{n+1} =& \frac{1}{1+(\Delta t-\Delta t^*)\kappa_{cap}}(\rho_l^*(1-\kappa_b)+ \\ &+(\Delta t-\Delta t^*)\frac{\kappa_{br}c_{rec}\rho_{r,j-1}^n}{c_{rec}+\kappa_{br}(\rho_l^*(1-\kappa_b)+\rho_{r,j-1}^n)})\end{aligned}$$

after an odd time step and

$$\begin{aligned}\rho_{l,j}^{n+1} =& \frac{1}{1+(\Delta t-\Delta t^*)\kappa_{cap}}(\rho_l^*(1-\kappa_b)+ \\ &+(\Delta t-\Delta t^*)\frac{\kappa_{br}c_{rec}\rho_{r,j-1}^n}{c_{rec}+\kappa_{br}(\rho_l^*(1-\kappa_b)+\rho_{r,j-1}^n)})\end{aligned}$$

after an even time step.

4.2 Scaling

Now we know the exact steps how to implement our model. But we have quite a lot of parameters that can be chosen arbitrarily, so it would be great to find some connections between them to reduce the number of free parameters. A good way to do so is to scale the equations to bring them in a dimensionless form.

Some connections of parameters I already mentioned before. For example that the lateral-flow speed depends on the polymerization speed of the filaments, so we get the equation

$$v_{lat} = v_p sin(\varphi_0).$$

Further the polymerization speed of a bundle is always smaller than the speed of the filaments. So

$$v_b(\varphi) \leq v_p$$

because the angle of the bundle is always smaller than the angle of the free filaments, so $\mid \varphi \mid \leq \varphi_0$. That‘s why a bundle can never grow faster than

the free filaments and because of this the filaments always move faster along the membrane.

To rescale our equations we have to define some reference numbers for the different units. So let's define a reference number for the length, denoted by $\bar{x}$, a reference number for the time, denoted by $\bar{t}$ and a reference number for the density, denoted by $\bar{\rho}$. With these we can now express some parameters, depending on their unit. For example the lateral-flow speed v_{lat} has the dimension $[v_{lat}] = \frac{length}{time} = \frac{cm}{s}$ because it's a speed. So it makes sense to choose $v_{lat} = \frac{\bar{x}}{\bar{t}}$. By choosing the reference numbers in the right way, it will be possible to obtain simpler equations with less free parameters.

Some of the parameters in our model are dimensionless which we can see if we take a look at the dimension of some equations. So let's next focus on the bundling rate κ_b. This parameter can be found in different equations in our model, like the change of the number of filaments that are bundled together:

$$\dot{r}_j = \kappa_b(v_{lat} - \dot{X}_j)\rho_r(x_j-).$$

Let's take a look at the dimension of each part of the equation. First of all, the number of right pointing filaments is dimensionless because it's just a number, so the derivative of r has dimension $[\dot{r}] = \frac{1}{s}$. The lateral-flow speed, like I already mentioned, has the dimension $[v_{lat}] = \frac{cm}{s}$, just like $[\dot{X}_j] = \frac{cm}{s}$ because X has dimension cm and so it's derivative with respect to time has dimension $\frac{cm}{s}$. The density of the right pointing filaments ρ_r is the number of filaments per area, so it has the dimension $[\rho_r] = \frac{1}{cm}$. So with these we can easily find out the dimension of the bundling rate:

$$[\kappa_b] = \frac{1}{s}\frac{s}{cm}cm = 1$$

so we see that the bundling rate is dimensionless.

Let's next take a look at the equation for the change of the density of the filaments:

$$\partial_t\rho \pm \partial_x(\rho v_{lat}) = -\kappa_{cap}\rho + \frac{\kappa_{br}c_{rec}\rho_*}{c_{rec} + \kappa_{br}(\rho + \rho_*)}$$

where ρ_* denotes the other filaments (so if we look at the right pointing filaments, $\rho_* = \rho_l$). Now we want to scale the equation to get a dimensionless form. For this let's choose the reference numbers as the following:

$$\bar{t} = \frac{1}{k_{cap}} \quad , \quad \bar{x} = v_{lat}\bar{t} = \frac{v_{lat}}{\kappa_{cap}} \quad , \quad \bar{\rho} = \frac{c_{rec}}{\kappa_{br}}.$$

With these we can now rescale our equation by choosing $\rho \to \rho\bar{\rho}$, $x \to x\bar{x}$ and $t \to t\bar{t}$:

$$\begin{aligned}
&\frac{1}{\bar{t}}\partial_t \rho\bar{\rho} \pm v_{lat}\frac{1}{\bar{x}}\partial_x \rho\bar{\rho} = -\kappa_{cap}\rho\bar{\rho} + \frac{\kappa_{br}c_{rec}\rho_*\bar{\rho}}{c_{rec} + \kappa_{br}(\rho+\rho_*)\bar{\rho}} \\
\Rightarrow \\
&\kappa_{cap}\partial_t\rho \pm v_{lat}\frac{\kappa_{cap}}{v_{lat}}\partial_x\rho = -\kappa_{cap}\rho + \frac{\kappa_{br}c_{rec}\rho_*\frac{1}{c_{rec}}}{\frac{c_{rec}}{c_{rec}} + \frac{\kappa_{br}}{c_{rec}}(\rho+\rho_*)\frac{c_{rec}}{\kappa_{br}}} \\
\Rightarrow \\
&\partial_t\rho \pm \partial_x\rho = -\rho + \frac{\alpha\rho_*}{1+\rho+\rho_*}
\end{aligned}$$

for $\alpha = \frac{\kappa_{br}}{\kappa_{cap}}$.

We will assume $\alpha > 1$, just like we assumed above $\kappa_{br} > \kappa_{cap}$, such that more filaments arise through branching, than filaments are capped.

In the equilibrium $\rho = \rho_* = \tilde{\rho}$ we get

$$0 = -\tilde{\rho} + \frac{\alpha\tilde{\rho}}{1+2\tilde{\rho}} \quad \Rightarrow \quad \tilde{\rho} = \frac{\alpha-1}{2}$$

which will be our starting value for a homogeneous system where the same number of filaments move right and left.

The jump conditions

$$\rho(x\pm) = (1-\kappa_b)\rho(x\mp)$$

for the filaments that cross the way of a bundle are already dimensionless. The density of the filaments ρ on both sides of the equation have the same dimension, 1 is just a number and therefore dimensionless. And we discovered already above that κ_b must be dimensionless, but we could have also concluded it from here, because it has to be dimensionless such that the

equation is correct.

The next equation I want to focus on is the change of the bundle positions:

$$\dot{X}_j = v_b(\varphi_j) sin(\varphi_j).$$

If we choose v_{lat} as the reference speed and rescale the equation by taking $v_b(\varphi) \to v_{lat} v_b(\varphi)$ we get

$$\begin{aligned} & \frac{\bar{x}}{\bar{t}} \dot{X}_j = v_{lat} v_b sin(\varphi_j) \\ \Rightarrow \quad & \frac{v_{lat}}{\kappa_{cap}} \kappa_{cap} \dot{X}_j = v_{lat} v_b sin(\varphi_j) \\ \Rightarrow \quad & \dot{X}_j = v_b sin(\varphi_j). \end{aligned}$$

For the number of right pointing filaments in a bundle, we get

$$\begin{aligned} & \frac{1}{\bar{t}} \dot{r}_j = \kappa_b (v_{lat} - \frac{\bar{x}}{\bar{t}} \dot{X}_j) \rho_r(X_j -) \bar{\rho} \\ \Rightarrow \quad & \frac{1}{\bar{t}} \dot{r}_j = \kappa_b v_{lat} (1 - \dot{X}_j) \rho_r(X_j -) \bar{\rho}. \end{aligned}$$

If we combine all these free parameter to one, we can define $\beta := \bar{t} v_{lat} \bar{\rho} = \frac{v_{lat} c_{rec}}{\kappa_{br} \kappa_{cap}}$ and get

$$\dot{r}_j = \kappa_b \beta (1 - \dot{X}_j) \rho_r(X_j -).$$

For the number of left pointing filaments that are bundled we get

$$\begin{aligned} & \frac{1}{\bar{t}} \dot{l}_j = \kappa_b (-v_{lat} - \frac{\bar{x}}{\bar{t}} \dot{X}_j) \rho_l(X_j +) \bar{\rho} \\ \Rightarrow \quad & \frac{1}{\bar{t}} \dot{l}_j = \kappa_b v_{lat} (-1 - \dot{X}_j) \rho_l(X_j +) \bar{\rho} \\ \Rightarrow \quad & \dot{l}_j = \kappa_b \beta (-1 - \dot{X}_j) \rho_l(X_j +) \end{aligned}$$

for the same β.

The last equations that are left now are that one, that describe the arising of new bundles. So let‘s first take a look at λ, which has the dimension $[\lambda] = \frac{1}{s}$.

With the same scaling as before and $L \to L\bar{x}$ we derive

$$\begin{aligned} \frac{1}{\bar{t}}\lambda &= v_n \int_0^{L\bar{x}} \bar{\rho}^2 \rho_r \rho_l d(x\bar{x}) \\ \Rightarrow \quad \lambda &= \bar{t} v_n \bar{\rho}^2 \bar{x} \int_0^L \rho_r \rho_l dx \end{aligned}$$

and if we choose $\gamma := \bar{t} v_n \bar{\rho}^2 \bar{x}$ we get

$$\lambda = \gamma \int_0^L \rho_r \rho_l dx.$$

And at last let's take a look at the equation that gives us the position of the new bundle, which is given by the density

$$f = \frac{\rho_r \rho_l}{\int_0^L \rho_r \rho_l dy}.$$

By scaling like before and with $f \to \frac{f}{\bar{x}}$ we get

$$\begin{aligned} \frac{1}{\bar{x}} f &= \frac{\bar{\rho}^2 \rho_r \rho_l}{\int_0^{L\bar{x}} \bar{\rho}^2 \rho_r \rho_l dy} \\ \Rightarrow \quad \frac{1}{\bar{x}} f &= \frac{\rho_r \rho_l}{\bar{x} \int_0^L \rho_r \rho_l dy} \\ \Rightarrow \quad f &= \frac{\rho_r \rho_l}{\int_0^L \rho_r \rho_l dy} \end{aligned}$$

which is dimensionless.

4.3 Result

Now we have finished our model. All parts that are needed are described. We know how new bundles arise and how they change over time. But right now it's very theoretically, so let's implement this model on Matlab to get a visualization, that shows how the bundle ends move along the membrane. To get a better understanding of the movement let's assume a homogeneous system, where the same density of filaments come from right and left. This

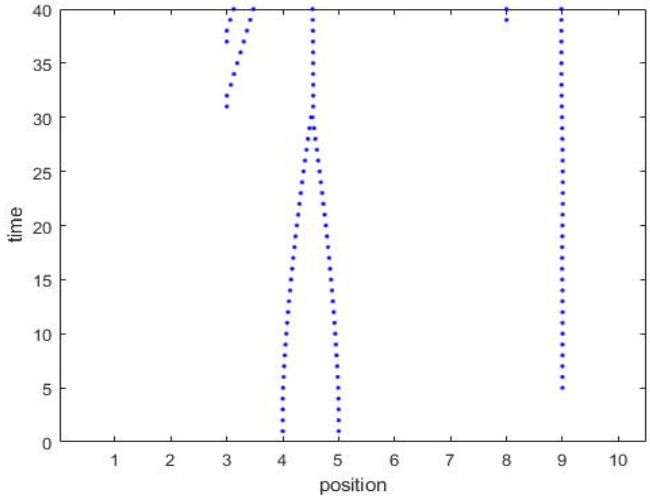

Figure 6: Position-Time diagram with two bundles in the beginning

initial density should be the density $\tilde{\rho}$ chosen like in the last section about scaling.

In Figure 6 we see such a visualization. On the x-axis are the positions on the membrane, so in this case we considered the membrane as the interval $[0, 10]$. On the y-axis runs the time, so here we are looking at 40 time steps and in this case we chose $\Delta t = \Delta x = 1$. We have a homogeneous system, so at each position point we start with the same density. In the beginning there are two bundles: one at position 4 and the other one at position 5. Both have the same number of right and left pointing filaments bundled, and we already know, that because of this they will grow normal against the membrane. So they don‘t change their position in the first view time steps. But both start to bundle more filaments, so the number of filaments in their neighbourhood starts to change. On the right side of a bundle there will be less filaments that point right and on its left side there will be less filaments that point left. So the second bundle at position 5 gets less right pointing filaments, because some of them are already bundled by the first one. After some time the difference of the right and left pointing filaments that are bundled together will be great enough that we see the difference in the new angle of the bundle. It starts to bend to the left, because more left pointing filaments are bundled. For the same reason the first bundle at position 4 starts to bend to the right. So after some time these two bundles

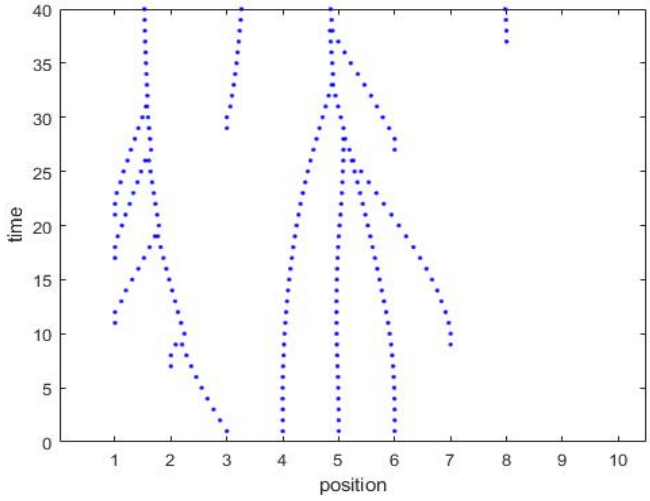

Figure 7: Position-Time diagram with four bundles in the beginning

will meet and fuse to one. The union has again the same number of right and left pointing filaments bundled, so it grows vertically against the membrane like in the beginning.

In this figure we also see some new bundles that arise at some point. The first new bundle arises after 5 time steps at position 9. Because of the homogeneous system it gets the same number of right and left pointing filaments, so it grows normal against the membrane. This bundle is not effected by the first two, because it‘s too far away. In the area between them new filaments can arise through branching, so this new bundle does not notice that there are some other bundles to its left. In contrast to the other bundles that arise. These arise after some time at the left side of the union of our starting bundles. These new bundles are close enough to notice that there is an other bundle. They get from the beginning on less filaments that point left, so the grow to the right and will meet at some future point the union bundle.

In Figure 7 we see the same position-time diagram as in the last picture, but with different starting bundles. This time we start with four bundles in the beginning. The first one grows to the left, because it has more left pointing filaments bundled while the next three all have the same number of right and left pointing filaments bundled and grow therefore vertically against the membrane. But just like we would expect it, these three start

to change their position after some time. The one at the right is the first one that bends because it has three bundles to its left. While the bundle at position 4 has one bundle to its left and two bundles to its right (respectively three, after some time). So this one also starts to bend, but slower than that at position 6. Because of this it‘s not a surprise that these two don‘t meet the bundle in their middle at the same time but after each other.

The new bundles that arise are all close enough to feel the difference of the filament density because of the other bundles. The bundles that arise to the right of our starting bundles quickly grow to the left, while the bundles that arise in the beginning of the interval, all move to the right. One bundle arises between our first starting bundle and the next three. This one grows to the right, but not as quick as the other new bundles in the beginning of the interval, because this one has also bundles to its left in contrast to the others.

So we see that the movement of a bundle is influenced by the other bundles. They will always turn to each other. They "want" to fuse with other bundles to get thicker. The more bundles are at some position, the greater is the attraction for the others. So they rather try to build a very thick bundle than just fuse with a smaller one. So bundles that have already fused with other bundles will very likely catch other bundles, if there exist some.

5 Appendix

Here I want to present my Matlab code, with which I produced the last pictures. The first code is to implement the distribution function $F_n(x)$ that gives us the position of a new bundle. This function will be used in the second code, which is the implementation of the whole algorithm described in section 4.1. There we define first all needed starting values and fix the free parameters, before we iterate the odd and the even step to calculate the change of the bundle positions on the membrane. In the end a figure is produced where the position points at the different time steps are plotted.

5.1 Distribution function

```
function [fkt] = F(s_1,s_2,N,P_l,P_r,delta_x)

%s_1,s_2 ... gibt an in welchem Schritt wir sind (s_2=0 ...
    erster (ungerader) Schritt)
%N ... Anzahl der Punkte an denen die Filamentdichte bekannt ist
%P_l,P_r ... Dichte der links-/rechts-geneigten Filamente an den
     Gitterpunkten (Vektor zum jeweiligen Zeitpunkt)
%delta_x ... raeumliche Schrittweite

fkt = zeros(1,N); %wird Funktionswerte enthalten

if s_2 == 0 %dann sind wir in einem ungeraden Schritt

    integral = delta_x * sum(P_l(2:N).*P_r(2:N)) + delta_x/2 *
       P_l(1)*P_r(1); %Integral ueber gesammtes Intervall

    for i = 1:N
        fkt(i) = ( delta_x * sum(P_l(2:i).*P_r(2:i)) + delta_x/2
            * P_l(1)*P_r(1) ) / integral;
    end

end


if s_1 == 0 %dann sind wir in einem geraden Schritt

    integral = delta_x * sum(P_l(1:N-1).*P_r(1:N-1)) + delta_x/2
        * P_l(N)*P_r(N);
```

```
    for i = 1:N-1
        fkt(i) = ( delta_x * sum(P_l(1:i).*P_r(1:i)) + delta_x/2
            * P_l(N)*P_r(N) ) / integral;
    end

    fkt(N) = 1;

end
```

5.2 Movement of bundle ends

```
%modelliert die Veraenderung der Filamente und Buendel in einem
%bestimmten Bereich ueber einen gewissen Zeitraum hinweg

L = input('Gib die Laenge des zu betrachtenden Intervalls ein: L
    = ');
delta_x = input('Gib die Schrittweite zwischen den Punkten ein,
    an denen die Filamentdichte betrachtet werden soll: delta_x =
    ');

N = floor(L/delta_x+1); %betrachten die Dichte der Filamente an
    N Punkten
x_1 = 0:delta_x:L; %Positionsvektor zu ungeraden Zeitpunkten t=2
    m+1 (an diesen Punkten betrachten wir die Filamentdichte)
L = (N - 1/2)*delta_x; % = (N-1)*delta_x + delta_x/2 ... da wir
    in rautenfoermigen Schritten vorgehen, ist einmal links,
    einmal rechts bei dem betrachteten Intervall noch eine halbe
    Schittweite Platz
x_2 = delta_x/2:delta_x:L; %Positionsvektor zu geraden
    Zeitpunkten t=2m

T = input('Gib an wie viele Zeitpunkte es geben soll: T = ');
delta_t = delta_x; %anstonsten stimmen die Winkel nicht mehr,
    bzw in einem Zeitschritt koennte sich sonst die Position des
    Buendels um mehr als einen Punkt aendern

%Fixieren der freien Parameter
a = 1.5;
b = 1;
c = 1;
p_start = (a-1)/2; %Startdichte der Filamente
k_b = 0.5; %Buendelungsrate
v_lat = 1;
```

```
%Dichte der linksgeneigten Filamente:
P_l = zeros(T,N);  %in der n-ten Zeile stehen die Werte zum n-
    ten Zeitpunkt
P_l(1,:) = p_start; %Startwert der linksgeneigten Filamente

%Dichte der rechtsgeneigten Filamente:
P_r = P_l;

X = zeros(T,1); %wird spaeter Buendelpositionen enthalten
r = zeros(T,1); %Anzahl der rechtgeneigten Filamente im Buendel
l = zeros(T,1); %Anzahl der linksgeneigten Filamente im Buendel
winkel = zeros(T,1); %Winkel der Buendel
winkel_0 = pi*35/180; %Winkel der Filamente ausserhalb des
    Buendels

disp(' ')

k = input('Soll bereits am Anfang ein Buendel existieren? Gib 1
    fuer "ja" ein und 0 fuer "nein": ');

if k == 1
    k = input('Gib die Anzahl der Buendel ein: k = ');
    X = zeros(T,k); %Position der Buendel - in ersten Spalte
        steht die Position des ersten Buendels zum jeweiligen
        Zeitpunkt
    r = zeros(T,k);
    l = zeros(T,k);
    winkel = zeros(T,k);

    disp(' ')
    disp('Gib die Informationen der Buendel in aufsteigender
        Reihenfolge an.')
    for index = 1:k
        X(1,index) = input('Position des Buendels: X = ');
        r(1,index) = input('Anzahl der rechtsgeneigten Filamente
            im Buendel: r = ');
        l(1,index) = input('Anzahl der linksgeneigten Filamente
            im Buendel: l = ');
        winkel(1,index) = winkel_0 * (r(1,index) - l(1,index))/(
            r(1,index) + l(1,index));
        disp(' ')
    end
end

X_position = X; %soll den jeweils links benachbarten Punkt (an
```

```
   dem wir die Filamentdichte wissen) des Buendels enthalten
if k > 0
    for u = 1:k
        jk = 1;
        v = X(1,u);
        while v >= delta_x
            v = X(1,u)-jk*delta_x;
            jk = jk+1;
        end
        X_position(1,u) = jk; %liefert den Index des
            benachbarten Punkt
    end
end

t = 1; %wir starten zum Zeitpunkt 1

for A = 1:T/2
    %geht alle Zeitschritte durch
    %nur bis T/2, da im Folgenden in einem Schleifendurchlauf
        jeweils zwei Schritte programmiert werden


%zuerst der "ungerade Schritt" von t nach t+1


%ENTSTEHEN NEUER BUENDEL:

    lambda = delta_x * sum(P_l(t,2:end).*P_r(t,2:end)) + delta_x
        /2 * P_l(t,1)*P_r(t,1);
    y = rand(1); %eine zufaellige Zahl aus (0,1)

    if y > exp(-lambda * delta_t) %dann entsteht ein neues
        Buendel
        z = rand(1);
        funktion = F(1,0,N,P_l(t,:),P_r(t,:),delta_x);
        diff = abs(funktion - z);
        m = min(diff);

        for u = 1:N
            if diff(u) == m
                pos = u; %an der Stelle pos entsteht ein neues
                    Buendel
            end
        end
```

```
if pos < L-delta_x/2 && pos > delta_x/2 %damit es
    innerhalb des Intervalls liegt

%neues Buendel in Buendelliste hinzufuegen:
d = 0;
%der Wert d enthaelt die Spalte der Matrix X an der das
%linksbenachbarte Buendel des neue entstandenen Buendels
     liegt
%(dh das "groesste" Buendel, das noch kleiner als pos
    ist)
for u = 1:size(X,2)
     if X(t,u) > 0 %falls an der Stelle u ein
         Buendeleintrag steht
          if pos > X(t,u) %kontrolliere ob das neue
              Buendel weiter rechts als das Buendel an
              Stelle u liegt
               d = u;
          end
     end
end
%das neue Buendel kommt in die Spalte d+1 der Matrix X

%definiere Hilfsmatrizen, um die Matrizen X,r,l,winkel,
    X_position um eine Spalte zu vergroessern
H = zeros(T,size(X,2)+1);
RE = zeros(T,size(X,2)+1);
LI = zeros(T,size(X,2)+1);
W = zeros(T,size(X,2)+1);
PO = zeros(T,size(X,2)+1);

%die Werte der vergangenen Zeitpunkte werden uebernommen
    , wobei die d+1-te Position fuer das neue Buendel
    freigehalten wird
H(1:t-1,1:d) = X(1:t-1,1:d);
RE(1:t-1,1:d) = r(1:t-1,1:d);
LI(1:t-1,1:d) = l(1:t-1,1:d);
W(1:t-1,1:d) = winkel(1:t-1,1:d);
PO(1:t-1,1:d) = X_position(1:t-1,1:d);

H(1:t-1,d+2:size(X,2)+1) = X(1:t-1,d+1:size(X,2));
RE(1:t-1,d+2:size(X,2)+1) = r(1:t-1,d+1:size(X,2));
LI(1:t-1,d+2:size(X,2)+1) = l(1:t-1,d+1:size(X,2));
W(1:t-1,d+2:size(X,2)+1) = winkel(1:t-1,d+1:size(X,2));
PO(1:t-1,d+2:size(X,2)+1) = X_position(1:t-1,d+1:size(X
    ,2));
```

```
        %Zum Zeitpunkt t:
        %die ersten d Buendel bleiben an der selben Stelle
        H(t,1:d) = X(t,1:d);
        RE(t,1:d) = r(t,1:d);
        LI(t,1:d) = l(t,1:d);
        W(t,1:d) = winkel(t,1:d);
        PO(t,1:d) = X_position(t,1:d);

        %an die Stelle d+1 kommt das neue Buendel (mit r=l=0 und
             winkel=0)
        H(t,d+1) = pos;
        jk = 1;
        v = pos;
        while v >= delta_x
             v = pos-jk*delta_x;
             jk = jk+1;
        end
        PO(t,d+1) = jk;

        %die nachfolgenden Buendel verschieben sich in der
            Nummerierung um eins
        H(t,d+2:size(X,2)+1) = X(t,d+1:size(X,2));
        RE(t,d+2:size(X,2)+1) = r(t,d+1:size(X,2));
        LI(t,d+2:size(X,2)+1) = l(t,d+1:size(X,2));
        W(t,d+2:size(X,2)+1) = winkel(t,d+1:size(X,2));
        PO(t,d+2:size(X,2)+1) = X_position(t,d+1:size(X,2));

        %aendere die Bezeichnung der Matrizen um wieder die
            vorherige Notation zu erhalten
        X = H;
        r = RE;
        l = LI;
        winkel = W;
        X_position = PO;

        k = k+1; %da es nun ein Buendel mehr gibt
        end

    end


%VERAENDERUNG DER BUENDEL:
```

```
if k >= 1

    %hier werden die Buendelpositionen nochmal extra
        abgespeichert
    %falls zwei sich schneiden wird das entstandene Buendel
        hier so behandelt
    %als haette es immer schon so existiert (wichtig zur
        Bestimmung der Filamente spaeter)

    X_bar = zeros(2,size(X,2));
    X_bar_position = zeros(2,size(X,2));
    w_bar = zeros(2,size(X,2)); %der entsprechende Winkel
        des Buendels

    geschnitten = 0; %bestimmt ob sich gerade zwei Buendel
        geschnitten haben

    %gehe jeden einzelnen Eintrag in betreffender Zeile in X
         durch -
    %teste ob sich Buendel mit anderem schneidet oder nicht
        - je nachdem wird Folgewert berechnet

for j = 1:size(X,2)
        test = 0;
        if X(t,j)>0 %habe in j-ter Spalte ein Buendel
            %falls "=0" waere, gehe gleich weiter zu
                Iterationsschritt j+1, da sich nichts aendert
                 in diesem Punkt - sonst mache weiter wie
                folgt

            %teste ob es sich mit rechtem Nachbar kreuzt

            %suche rechten Nachbar:
            e = 1;
            for i = 1:size(X,2)
                if j+e <= size(X,2) %dann sind wir noch
                    innerhalb der Matrix X - haben also noch
                    die Chance mit e Schritten zum naechsten
                    Buendel zu kommen
                     if X(t,j+e)>0 %dann liegt in der Spalte
                         j+e das rechts benachbarte Buendel
                          break
                     else
                          e=e+1;
                     end
```

```
        elseif j+e > size(X,2) %dann sind wir bei
            der Suche nach einem rechts benachbarten
            Buendel ueber die Dimension von X
            hinausgewandert - dh das j-te Buendel hat
             keinen rechten Nachbar
            e = 0;
        end
    end

%teste ob die Buendel an den Stellen j und j+e sich
    schneiden:
    if e>0 && X(t,j+e)-X(t,j) < delta_x %dann
        koennten sich die Buendel kreuzen
        delta_tt = (X(t,j+e)-X(t,j)) / (sin(winkel(t
            ,j))-sin(winkel(t,j+e)));  %wenn wir v_b(
            phi)=1 annehmen
        if delta_tt >= 0 && delta_tt < delta_t %dann
             schneiden sie sich wirklich in diesem
            Schritt
            test = 1;
        end
    else
        test = 0; %wenn sich keine der Buendel
            schneiden
    end

if geschnitten == 1 %dann haben sich im vorherigen
    Schritt zwei Buendel geschnitten - dh der
    naechste positive Wert in X braucht nicht mehr
    beachtet zu werden, da er sich bereits mit dem
    vorherigen vereinigt hat
    geschnitten = 0;

elseif test == 1 %falls sich zwei Buendel schneiden
    (das an Stelle j mit dem an Stelle j+e)
    pl1 = X_position(t,j)+1;
    pl2 = X_position(t,j+e)+1;
    pr1 = X_position(t,j);
    pr2 = X_position(t,j+e);

    if pl1<1 || pl2>N
        X(t+1,j)=0;
        l(t+1,j)=0;
        r(t+1,j)=0;
        winkel(t+1,j)=0;
```

```
    k=k-1; %es gibt ein Buendel weniger, da
        dieses Buendel ueber den Rand hinaus
        gewachsen ist
    continue
end
if pr1<1 || pr2>N
    X(t+1,j)=0;
    l(t+1,j)=0;
    r(t+1,j)=0;
    winkel(t+1,j)=0;
    k=k-1;
    continue
end

XX = X(t,j+e) + delta_tt*sin(winkel(t,j)); %
    Position des vereinigten Buendels zum
    Zeitpunkt des Schnittpunkts
ll = l(t,j) + delta_tt * k_b * b * (1-sin(winkel
    (t,j))) * P_l(t,pl1) + l(t,j+e) + delta_tt
    * k_b * b *(1-sin(winkel(t,j+e))) * P_l(t,pl2
    );
rr = r(t,j) + delta_tt * k_b * b * (1-sin(winkel
    (t,j))) * P_r(t,pr1) + r(t,j+e) + delta_tt
    * k_b * b *(1-sin(winkel(t,j+e))) * P_r(t,pr2
    );
ww = winkel_0 * (rr - ll)/(rr + ll);
%Position:
jk = 1;
v = XX;
while v >= delta_x
    v = XX-jk*delta_x;
    jk = jk+1;
end
XX_position = jk;

%nach einem vollen Zeitschritt bekommen wir dann
    das vereinigte Buendel an der Stelle j:
X(t+1,j) = XX + (delta_t - delta_tt)*sin(ww); %
    Vereinigung von Buendel j und j+e wird zu j-
    tem Buendel
l(t+1,j) = ll + (delta_t - delta_tt)* k_b * b *
    (1-sin(ww)) * P_l(t,XX_position+1);
r(t+1,j) = rr + (delta_t - delta_tt)* k_b * b *
    (1-sin(ww)) * P_r(t,XX_position);
winkel(t+1,j) = winkel_0 * (r(t+1,j) - l(t+1,j))
```

```
        /(r(t+1,j)+l(t+1,j));

    ju = X_position(t,j);
    if  X(t+1,j)-x_2(ju)<delta_x && X(t+1,j)-x_2(ju)
        >=0
        X_position(t+1,j) = ju;
    elseif X(t+1,j)-x_2(ju-1)<delta_x && X(t+1,j)-
        x_2(ju-1)>=0
        X_position(t+1,j) = ju-1;
    end

    %die Anzahl der Buendel ist um ein kleiner
        geworden:
    k = k-1;

    %an Stelle j kommt der Wert des vereinigten
        Buendels
    %(als haette es zu dem Zeitpunkt schon existiert
        )
    X_bar(1,j) = XX - delta_tt*sin(ww);
    X_bar(2,j) = X(t+1,j);

    dk = 1;
    v = X_bar(1,j);
    while v > delta_x
        v = X_bar(1,j)-dk*delta_x;
        dk = dk+1;
    end
    X_bar_position(1,j) = dk;
    X_bar_position(2,j) = X_position(t+1,j);

    w_bar(1,j) = winkel_0 * ((r(t,j)+r(t,j+e)) - (l(
        t,j)+l(t,j+e))) / (r(t,j)+r(t,j+e) + l(t,j)+l
        (t,j+e));
    w_bar(2,j) = winkel(t+1,j);


    geschnitten = 1; %dh es haben sich gerade zwei
        Buendel geschnitten wodurch der naechste Wert
         in der Matrix nicht mehr behandelt werden
        muss


else %falls sich dieses Buendel nicht mit Nachbar
    schneidet
```

```
pl = X_position(t,j)+1;
pr = X_position(t,j);

if pl<1 || pl>N
    X(t+1,j)=0;
    l(t+1,j)=0;
    r(t+1,j)=0;
    winkel(t+1,j)=0;
    k=k-1; %es gibt ein Buendel weniger, da
        dieses Buendel ueber den Rand hinaus
        gewachsen ist
    continue
end
if pr<1 || pr>N
    X(t+1,j)=0;
    l(t+1,j)=0;
    r(t+1,j)=0;
    winkel(t+1,j)=0;
    k=k-1;
    continue
end

X(t+1,j) = X(t,j) + delta_t*sin(winkel(t,j));
l(t+1,j) = l(t,j) + delta_t * k_b * b * (1-sin(
    winkel(t,j))) * P_l(t,pl);
r(t+1,j) = r(t,j) + delta_t * k_b * b * (1-sin(
    winkel(t,j))) * P_r(t,pr);
winkel(t+1,j) = winkel_0 * (r(t+1,j)-l(t+1,j))/(
    r(t+1,j)+l(t+1,j));


if X(t+1,j)>0 && X(t+1,j)<L
    ju = X_position(t,j);
    if ju>1 && ju<N
        if  X(t+1,j)-x_2(ju)<delta_x && X(t+1,j)
            -x_2(ju)>=0
            X_position(t+1,j) = ju;
        elseif X(t+1,j)-x_2(ju-1)<delta_x && X(t
            +1,j)-x_2(ju-1)>=0
            X_position(t+1,j) = ju-1;
        end
    elseif ju==1
        if  X(t+1,j)-x_2(ju)<delta_x && X(t+1,j)
            -x_2(ju)>=0
            X_position(t+1,j) = ju;
```

```
                    else
                        X_position(t+1,j) = 0;
                    end
                elseif ju==N
                    if X(t+1,j)-x_2(ju-1)<delta_x && X(t+1,j
                        )-x_2(ju-1)>=0
                        X_position(t+1,j) = ju-1;
                    else
                        X_position(t+1,j) = 0;
                    end
                end
            end

            %bestimme X_bar, dessen Position und Winkel:
            %in der ersten Zeile von X_bar stehen die
                Buendelpositionen zum
            %Zeitpunkt t (haben sich zwei Buendel in diesem
                Schritt
            %vereinigt, werden sie hier so behandelt, als
                waeren sie schon
            %immer ein einziges Buendel gewesen)
            %in der zweiten Zeile von X_bar stehen die
                Buendelpositionen zum
            %Zeitpunkt t+1

            X_bar(1,j) = X(t,j);
            X_bar(2,j) = X(t+1,j);

            X_bar_position(1,j) = X_position(t,j);
            X_bar_position(2,j) = X_position(t+1,j);

            w_bar(1,j) = winkel(t,j);
            w_bar(2,j) = winkel(t+1,j);

        end
        end

    end

    end
```

%VERAENDERUNG DER DICHTE DER RECHTSGENEIGTEN FILAMENTE:

```
    for j = 2:N-1
        %geht von Position j nach j
```

```
    %teste ob es sich mit Buendel kreuzt:
        s = 0;
        for i = 1:size(X_bar,2)
            if X_bar(1,i)>0 %fall ein Buendel in i-ter
                Spalte der Matrix X ist
                if X_bar_position(1,i)==j && X_bar_position
                    (2,i)==j-1
                    s = 1;
                    d = i; %merke mit welchem Buendel es
                        sich kreuzt
                end
            end
        end

    if s == 1 %es kreuzt sich mit einem Buendel
        delta_tt = (X_bar(1,d) - x_1(j)) / (v_lat - sin(
            w_bar(1,d)));
        pp = 1/(1+delta_tt) * (P_r(t,j) + delta_tt * (a*P_l(
            t,j))/(1+P_r(t,j)+P_l(t,j)));
        P_r(t+1,j) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b) +
            (delta_t-delta_tt)*(a*P_l(t,j+1))/(1+pp*(1-k_b)+
            P_l(t,j+1)));

    else %es kreuzt sich mit keinem Buendel
        P_r(t+1,j) = 1/(1+delta_t) * (P_r(t,j) + delta_t * (
            a*P_l(t,j))/(1+P_r(t,j)+P_l(t,j)));
    end

end

%Wert fuer j=1:
    P_r(t+1,1) = 1/(1+delta_t) * (P_r(t,1) + delta_t * (a*
        P_l(t,1))/(1+P_r(t,1)+P_l(t,1)));

%Wert fuer j=N:
    s = 0;
    for i = 1:size(X_bar,2)
        if X_bar(1,i)>0
            if X_bar_position(1,i)==N && X_bar_position(2,i)
                ==N-1
                s = 1;
                d = i; %merke mit welchem Buendel es sich
                    kreuz
            end
```

```
        end
    end
    if s == 1 %es kreuzt sich mit einem Buendel
        delta_tt = (X_bar(1,d) - x_1(N)) / (v_lat - sin(
            w_bar(1,d)));
        pp = 1/(1+delta_tt) * (P_r(t,N) + delta_tt * (a*P_l(
            t,N))/(1+P_r(t,N)+P_l(t,N)));
        P_r(t+1,N) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b) +
            (delta_t-delta_tt)*(a*p_start)/(1+pp*(1-k_b)+
            p_start));

    else %es kreuzt sich mit keinem Buendel
        P_r(t+1,N) = 1/(1+delta_t) * (P_r(t,N) + delta_t * (
            a*P_l(t,N))/(1+P_r(t,N)+P_l(t,N)));
    end
```

%VERAENDERUNG DER LINKSGENEIGTEN FILAMENTE:

```
for j = 2:N-1
    %geht von Position j nach j-1

    %teste ob es sich mit Buendel kreuzt:
        s = 0;
        for i = 1:size(X_bar,2)
            if X_bar(1,i)>0
                if X_bar_position(1,i)==j-1 &&
                    X_bar_position(2,i)==j-1
                    s = 1;
                    d = i; %merke mit welchem Buendel es
                        sich kreuzt
                end
            end
        end

    if s == 1 %es kreuzt sich mit einem Buendel
        delta_tt = (x_1(j) - X_bar(1,d)) / (v_lat + sin(
            w_bar(1,d)));
        pp = 1/(1+delta_tt) * (P_l(t,j) + delta_tt * (a*P_r(
            t,j))/(1+P_r(t,j)+P_l(t,j)));
        P_l(t+1,j-1) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b)
            + (delta_t-delta_tt)*(a*P_r(t,j-1))/(1+pp*(1-k_b)
            +P_r(t,j-1)));

    else %es kreuzt sich mit keinem Buendel
```

```
            P_l(t+1,j-1) = 1/(1+delta_t) * (P_l(t,j) + delta_t *
                (a*P_r(t,j))/(1+P_r(t,j)+P_l(t,j)));
        end

    end

    %P_l(t+1,N) enthaelt als Randpunkt den unveraenderten
        Startwert:
        P_l(t+1,N) = p_start;

    %P_l(t+1,N-1):
        s = 0;
        for i = 1:size(X_bar,2)
            if X_bar(1,i)>0
                if X_bar_position(1,i)==N-1 && X_bar_position(2,
                    i)==N-1
                    s = 1;
                    d = i; %merke mit welchem Buendel es sich
                        kreuz
                end
            end
        end
        if s == 1 %es kreuzt sich mit einem Buendel
            delta_tt = (x_1(N) - X_bar(1,d)) / (v_lat + sin(
                w_bar(1,d)));
            pp = 1/(1+delta_tt) * (P_l(t,N) + delta_tt * (a*P_r(
                t,N))/(1+P_r(t,N)+P_l(t,N)));
            P_l(t+1,N-1) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b)
                + (delta_t-delta_tt)*(a*P_r(t,N-1))/(1+pp*(1-k_b)
                +P_r(t,N-1)));

        else %es kreuzt sich mit keinem Buendel
            P_l(t+1,N-1) = 1/(1+delta_t) * (P_l(t,N) + delta_t *
                (a*P_r(t,N))/(1+P_r(t,N)+P_l(t,N)));
        end



%dann der "gerade Schritt" von t+1 nach t+2 (ausser t+2 > T)

if t+2 <= T

    %ENTSTEHEN NEUER BUENDEL:
```

```
lambda = delta_x * sum(P_l(t+1,1:end-1).*P_r(t+1,1:end-1)) +
    delta_x/2 * P_l(t+1,end)*P_r(t+1,end);
y = rand(1); %eine zufaellige Zahl aus (0,1)

if y > exp(-lambda * delta_t) %dann entsteht ein neues
   Buendel
    z = rand(1);
    funktion = F(0,1,N,P_l(t+1,:),P_r(t+1,:),delta_x);
    diff = abs(funktion - z);
    m = min(diff);

    for u = 1:N
        if diff(u) == m
            pos = u; %an der Stelle pos entsteht ein neues
               Buendel
        end
    end

    if pos < L-delta_x/2 && pos > delta_x/2 %damit es
       innerhalb des Intervalls liegt

    %neues Buendel in Buendelliste hinzufuegen:
    d = 0; %der Wert d enthaelt die Spalte der Matrix X an
       der das linksbenachbarte Buendel des neue
       entstandenen Buendels liegt (dh das "groesste"
       Buendel, das noch kleiner als pos ist)
    for u = 1:size(X,2)
        if X(t+1,u) > 0 %falls an der Stelle u ein
           Buendeleintrag steht
            if pos > X(t+1,u) %kontrolliere ob das neue
               Buendel weiter rechts als das Buendel an
               Stelle u liegt
                d = u;
            end
        end
    end
    %das neue Buendel kommt in die Spalte d+1 der Matrix X

    %definiere Hilfsmatrizen, um die Matrizen X,r,l,winkel,
       X_position um eine Spalte zu vergroessern
    H = zeros(T,size(X,2)+1);
    RE = zeros(T,size(X,2)+1);
    LI = zeros(T,size(X,2)+1);
    W = zeros(T,size(X,2)+1);
    PO = zeros(T,size(X,2)+1);
```

```
%die Werte der vergangenen Zeitpunkte werden uebernommen
    , wobei die d+1-te Position fuer das neue Buendel
    freigehalten wird
H(1:t,1:d) = X(1:t,1:d);
RE(1:t,1:d) = r(1:t,1:d);
LI(1:t,1:d) = l(1:t,1:d);
W(1:t,1:d) = winkel(1:t,1:d);
PO(1:t,1:d) = X_position(1:t,1:d);

H(1:t,d+2:size(X,2)+1) = X(1:t,d+1:size(X,2));
RE(1:t,d+2:size(X,2)+1) = r(1:t,d+1:size(X,2));
LI(1:t,d+2:size(X,2)+1) = l(1:t,d+1:size(X,2));
W(1:t,d+2:size(X,2)+1) = winkel(1:t,d+1:size(X,2));
PO(1:t,d+2:size(X,2)+1) = X_position(1:t,d+1:size(X,2));

%Zum Zeitpunkt t+1:
%die ersten d Buendel bleiben an der selben Stelle
H(t+1,1:d) = X(t+1,1:d);
RE(t+1,1:d) = r(t+1,1:d);
LI(t+1,1:d) = l(t+1,1:d);
W(t+1,1:d) = winkel(t+1,1:d);
PO(t+1,1:d) = X_position(t+1,1:d);

%an die Stelle d+1 kommt das neue Buendel (mit r=l=0 und
    winkel=0)
H(t+1,d+1) = pos;
jk = 1;
v = pos;
while v >= delta_x
    v = pos-jk*delta_x;
    jk = jk+1;
end
PO(t+1,d+1) = jk;

%die nachfolgenden Buendel verschieben sich in der
    Nummerierung um eins
H(t+1,d+2:size(X,2)+1) = X(t+1,d+1:size(X,2));
RE(t+1,d+2:size(X,2)+1) = r(t+1,d+1:size(X,2));
LI(t+1,d+2:size(X,2)+1) = l(t+1,d+1:size(X,2));
W(t+1,d+2:size(X,2)+1) = winkel(t+1,d+1:size(X,2));
PO(t+1,d+2:size(X,2)+1) = X_position(t+1,d+1:size(X,2));

%aendere die Bezeichnung der Matrizen um wieder die
    vorherige Notation zu erhalten
```

```
        X = H;
        r = RE;
        l = LI;
        winkel = W;
        X_position = PO;

        k = k+1; %da es nun ein Buendel mehr gibt
        end

    end


%VERAENDERUNG DER BUENDEL:


    if k >= 1

        X_bar = zeros(2,size(X,2));
        X_bar_position = zeros(2,size(X,2));
        w_bar = zeros(2,size(X,2));

        geschnitten = 0; %bestimmt ob sich gerade zwei Buendel
            geschnitten haben

        %gehe jeden einzelnen Eintrag in betreffender Zeile in X
             durch -
        %teste ob sich Buendel mit anderem schneidet oder nicht
            - je nachdem wird Folgewert berechnet

    for j = 1:size(X,2)
            test = 0;
            if X(t+1,j)>0 %habe in j-ter Spalte ein Buendel

                %teste ob es sich mit rechtem Nachbar kreuzt
                e = 1;
                for i = 1:size(X,2)
                    if j+e <= size(X,2) %dann sind wir noch
                        innerhalb der Matrix X - haben also noch
                        die Chance mit e Schritten zum naechsten
                        Buendel zu kommen
                         if X(t+1,j+e)>0 %dann liegt in der
                            Spalte j+e das rechts benachbarte
                            Buendel
                             break
```

```
            else
                e=e+1;
            end
        elseif j+e > size(X,2) %dann sind wir bei
            der Suche nach einem rechts benachbarten
            Buendel ueber die Dimension von X
            hinausgewandert
            e = 0;
        end
    end

%teste ob die Buendel an den Stellen j und j+e sich
    schneiden:

    if e>0 && X(t+1,j+e)-X(t+1,j) < delta_x %dann
        koennten sich die Buendel kreuzen
        delta_tt = (X(t+1,j+e)-X(t+1,j)) / (sin(
            winkel(t+1,j))-sin(winkel(t+1,j+e)));  %
            wenn wir v_b(phi)=1 annehmen
        if delta_tt >= 0 && delta_tt < delta_t %dann
            schneiden sie sich wirklich in diesem
            Schritt
            test = 1;
        end
    else
        test = 0; %wenn sich keine der Buendel
            schneiden
    end

if geschnitten == 1 %dann haben sich im vorherigen
    Schritt zwei Buendel geschnitten - dh der
    naechste positive Wert in X braucht nicht mehr
    beachtet zu werden, da er sich bereits mit dem
    vorherigen vereinigt hat
    geschnitten = 0;

elseif test == 1 %falls sich zwei Buendel schneiden
    (das an Stelle j mit dem an Stelle j+e)
    pl1 = X_position(t+1,j)+1;
    pl2 = X_position(t+1,j+e)+1;
    pr1 = X_position(t+1,j);
    pr2 = X_position(t+1,j+e);

    if pl1<1 || pl2>N
        X(t+2,j)=0;
```

```
    l(t+2,j)=0;
    r(t+2,j)=0;
    winkel(t+2,j)=0;
    k=k-1;
    continue
end
if pr1<1 || pr2>N
    X(t+2,j)=0;
    l(t+2,j)=0;
    r(t+2,j)=0;
    winkel(t+2,j)=0;
    k=k-1;
    continue
end

XX = X(t+1,j+e) + delta_tt*sin(winkel(t+1,j)); %
    Position des vereinigten Buendels zum
    Zeitpunkt des Schnittpunkts
ll = l(t+1,j) + delta_tt * k_b * b * (1-sin(
    winkel(t+1,j))) * P_l(t+1,pl1) + l(t+1,j+e)
     + delta_tt * k_b * b *(1-sin(winkel(t+1,j+e)
    )) * P_l(t+1,pl2);
rr = r(t+1,j) + delta_tt * k_b * b * (1-sin(
    winkel(t+1,j))) * P_r(t+1,pr1) + r(t+1,j+e)
     + delta_tt * k_b * b *(1-sin(winkel(t+1,j+e)
    )) * P_r(t+1,pr2);
ww = winkel_0 * (rr - ll)/(rr + ll);
%Position:
jk = 1;
v = XX;
while v >= delta_x
    v = XX-jk*delta_x;
    jk = jk+1;
end
XX_position = jk;

%nach einem vollen Zeitschritt bekommen wir dann
     das vereinigte Buendel an der Stelle j:
X(t+2,j) = XX + (delta_t - delta_tt)*sin(ww); %
    Vereinigung von Buendel j und j+e wird zu j-
    tem Buendel
l(t+2,j) = ll + (delta_t - delta_tt)* k_b * b *
    (1-sin(ww)) * P_l(t+1,XX_position+1);
r(t+2,j) = rr + (delta_t - delta_tt)* k_b * b *
    (1-sin(ww)) * P_r(t+1,XX_position);
```

```
winkel(t+2,j) = winkel_0 * (r(t+2,j) - l(t+2,j))
    /(r(t+2,j)+l(t+2,j));

ju = X_position(t+1,j);
if  X(t+2,j)-x_1(ju)<delta_x && X(t+2,j)-x_1(ju)
    >=0
     X_position(t+2,j) = ju;
elseif X(t+2,j)-x_1(ju+1)<delta_x && X(t+2,j)-
    x_1(ju+1)>=0
     X_position(t+2,j) = ju+1;
end

%die Anzahl der Buendel ist um ein kleiner
    geworden:
k = k-1;

%an Stelle j kommt der Wert des vereinigten
    Buendels
%(als haette es zu dem Zeitpunkt schon existiert
    )
X_bar(1,j) = XX - delta_tt*sin(ww);
X_bar(2,j) = X(t+2,j);

dk = 1;
v = X_bar(1,j);
while v > delta_x
     v = X_bar(1,j)-dk*delta_x;
     dk = dk+1;
end
X_bar_position(1,j) = dk;
X_bar_position(2,j) = X_position(t+2,j);

w_bar(1,j) = winkel_0 * ((r(t+1,j)+r(t+1,j+e)) -
     (l(t+1,j)+l(t+1,j+e))) / (r(t+1,j)+r(t+1,j+e
    ) + l(t+1,j)+l(t+1,j+e));
w_bar(2,j) = winkel(t+2,j);


geschnitten = 1; %dh es haben sich gerade zwei
    Buendel geschnitten wodurch der naechste Wert
     in der Matrix nicht mehr behandelt werden
    muss


else %falls sich dieses Buendel nicht mit Nachbar
```

```
schneidet
 pl = X_position(t+1,j)+1;
 pr = X_position(t+1,j);

 if pl<1 || pl>N
     X(t+2,j)=0;
     l(t+2,j)=0;
     r(t+2,j)=0;
     winkel(t+2,j)=0;
     k=k-1;
     continue
 end
 if pr<1 || pr>N
     X(t+2,j)=0;
     l(t+2,j)=0;
     r(t+2,j)=0;
     winkel(t+2,j)=0;
     k=k-1;
     continue
 end

 X(t+2,j) = X(t+1,j) + delta_t*sin(winkel(t+1,j))
     ;
 l(t+2,j) = l(t+1,j) + delta_t * k_b * b * (1-sin
     (winkel(t+1,j))) * P_l(t+1,pl);
 r(t+2,j) = r(t+1,j) + delta_t * k_b * b * (1-sin
     (winkel(t+1,j))) * P_r(t+1,pr);
 winkel(t+2,j) = winkel_0 * (r(t+2,j)-l(t+2,j))/(
     r(t+2,j)+l(t+2,j));


 if X(t+2,j)>0 && X(t+2,j)<L
     ju = X_position(t+1,j);
     if ju>1 && ju<N-1
         if  X(t+2,j)-x_1(ju)<delta_x && X(t+2,j)
             -x_1(ju)>=0
             X_position(t+2,j) = ju;
         elseif X(t+2,j)-x_1(ju+1)<delta_x && X(t
             +2,j)-x_1(ju+1)>=0
             X_position(t+2,j) = ju+1;
         end
     elseif ju==1
         if  X(t+2,j)-x_1(ju)<delta_x && X(t+2,j)
             -x_1(ju)>=0
             X_position(t+2,j) = ju;
```

```
                    elseif X(t+2,j)-x_1(ju+1)<delta_x && X(t
                        +2,j)-x_1(ju+1)>=0
                        X_position(t+2,j) = ju+1;
                    end
                elseif ju==N-1
                    if  X(t+2,j)-x_1(ju)<delta_x && X(t+2,j)
                        -x_1(ju)>=0
                        X_position(t+2,j) = ju;
                    elseif X(t+2,j)-x_1(ju+1)<delta_x/2 && X
                        (t+2,j)-x_1(ju+1)>=0 % < delta_x/2 da
                         ab dann die Intervallgrenze erreicht
                         ist
                        X_position(t+2,j) = ju+1;
                    else
                        X_position(t+2,j) = 0;
                    end
                end
            end

            %bestimme X_bar, dessen Position und Winkel:
            X_bar(1,j) = X(t+1,j);
            X_bar(2,j) = X(t+2,j);

            X_bar_position(1,j) = X_position(t+1,j);
            X_bar_position(2,j) = X_position(t+2,j);

            w_bar(1,j) = winkel(t+1,j);
            w_bar(2,j) = winkel(t+2,j);

        end
        end

    end

    end
```

%VERAENDERUNG DER DICHTE DER RECHTSGENEIGTEN FILAMENTE:

```
    for j = 2:N-1
        %geht von Position j nach j+1

        %teste ob es sich mit Buendel kreuzt:
            s = 0;
```

```
        for i = 1:size(X_bar,2)
            if X_bar(1,i)>0 %dann steht in der i-ten Spalte
                von X ein Buendeleintrag
                 if X_bar_position(1,i)==j && X_bar_position
                    (2,i)==j
                     s = 1;
                     d = i; %merke mit welchem Buendel es
                         sich kreuzt
                 end
            end
        end

    if s == 1 %es kreuzt sich mit einem Buendel
        delta_tt = (X_bar(1,d) - x_2(j)) / (v_lat - sin(
            w_bar(1,d)));
        pp = 1/(1+delta_tt) * (P_r(t+1,j) + delta_tt * (a*
            P_l(t+1,j))/(1+P_r(t+1,j)+P_l(t+1,j)));
        P_r(t+2,j+1) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b)
            + (delta_t-delta_tt)*(a*P_l(t+1,j+1))/(1+pp*(1-
            k_b)+P_l(t+1,j+1)));

    else %es kreuzt sich mit keinem Buendel
        P_r(t+2,j+1) = 1/(1+delta_t) * (P_r(t+1,j) + delta_t
             * (a*P_l(t+1,j))/(1+P_r(t+1,j)+P_l(t+1,j)));
    end

end

%P_r(t+2,1) enthaelt als Randpunkt den Startwert:
    P_r(t+2,1) = p_start;

%P_r(t+2,2):
    s = 0;
    for i = 1:size(X_bar,2)
        if X_bar(1,i)>0
            if X_bar_position(1,i)==1 && X_bar_position(2,i)
               ==1
                s = 1;
                d = i; %merke mit welchem Buendel es sich
                    kreuzt
            end
        end
    end

    if s == 1 %es kreuzt sich mit einem Buendel
```

```
            delta_tt = (X_bar(1,d) - x_2(1)) / (v_lat - sin(
                w_bar(1,d)));
            pp = 1/(1+delta_tt) * (P_r(t+1,1) + delta_tt * (a*
                P_l(t+1,1))/(1+P_r(t+1,1)+P_l(t+1,1)));
            P_r(t+2,2) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b) +
                (delta_t-delta_tt)*(a*P_l(t+1,2))/(1+pp*(1-k_b)+
                P_l(t+1,2)));

        else %es kreuzt sich mit keinem Buendel
            P_r(t+2,2) = 1/(1+delta_t) * (P_r(t+1,1) + delta_t *
                 (a*P_l(t+1,1))/(1+P_r(t+1,1)+P_l(t+1,1)));
        end
```

%VERAENDERUNG DER LINKSGENEIGTEN FILAMENTE:

```
    for j = 2:N-1
        %geht von Position j nach j

        %teste ob es sich mit Buendel kreuzt:
            s = 0;
            for i = 1:size(X_bar,2)
                if X_bar(1,i)>0
                    if X_bar_position(1,i)==j-1 &&
                        X_bar_position(2,i)==j
                        s = 1;
                        d = i; %merke mit welchem Buendel es
                            sich kreuzt
                    end
                end
            end

        if s == 1 %es kreuzt sich mit einem Buendel
            delta_tt = (x_2(j) - X_bar(1,d)) / (v_lat + sin(
                w_bar(1,d)));
            pp = 1/(1+delta_tt) * (P_l(t+1,j) + delta_tt * (a*
                P_r(t+1,j))/(1+P_r(t+1,j)+P_l(t+1,j)));
            P_l(t+2,j) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b) +
                (delta_t-delta_tt)*(a*P_r(t+1,j-1))/(1+pp*(1-k_b)
                +P_r(t+1,j-1)));

        else %es kreuzt sich mit keinem Buendel
            P_l(t+2,j) = 1/(1+delta_t) * (P_l(t+1,j) + delta_t *
                 (a*P_r(t+1,j))/(1+P_r(t+1,j)+P_l(t+1,j)));
```

```
        end

    end

    %es fehlt noch P_l(t+2,1):
        P_l(t+2,1) = 1/(1+delta_t) * (P_l(t+1,1) + delta_t * (a*
            P_r(t+1,1))/(1+P_r(t+1,1)+P_l(t+1,1)));

    %und fuer j=N, P_l(t+2,N):
        s = 0;
        for i = 1:size(X_bar,2)
            if X_bar(1,i)>0
                if X_bar_position(1,i)==N-1 && X_bar_position(2,
                    i)==N
                    s = 1;
                    d = i; %merke mit welchem Buendel es sich
                        kreuzt
                end
            end
        end

        if s == 1 %es kreuzt sich mit einem Buendel
            delta_tt = (x_2(N) - X_bar(1,d)) / (v_lat + sin(
                w_bar(1,d)));
            pp = 1/(1+delta_tt) * (P_l(t+1,N) + delta_tt * (a*
                P_r(t+1,N))/(1+P_r(t+1,N)+P_l(t+1,N)));
            P_l(t+2,N) = 1/(1+delta_t-delta_tt) * (pp*(1-k_b) +
                (delta_t-delta_tt)*(a*P_r(t+1,N-1))/(1+pp*(1-k_b)
                +P_r(t+1,N-1)));

        else %es kreuzt sich mit keinem Buendel
            P_l(t+2,N) = 1/(1+delta_t) * (P_l(t+1,N) + delta_t *
                (a*P_r(t+1,N))/(1+P_r(t+1,N)+P_l(t+1,N)));
        end

end


    t=t+2; %beginne wieder von vorne mit einem "ungeraden
        Schritt"
end


xMax=L;
yMax=size(X,1);
```

```
YA = 1:size(X,1);

for i = 1:size(X,1)
    figure(1)
      plot(X(1:i,:),YA(1:i),'b.')
      axis([0.01 xMax 0 yMax])
      xlabel('position')
      ylabel('time')
      drawnow
      pause(0.6)
end
```

List of Figures

References

[1] P.K. Mattila, P. Lappalainen: Filopodia: molecular architecture and cellular functions, *Nat. Rev. Mol. Cell Biology*, **9** (2008), 446-454

[2] T.M. Svitkina, E.A. Bulanova, O.Y. Chaga, D.M. Vignjevic, S. Kojima, J.M. Vasiliev, G.G. Borisy: Mechanism of filopodia initiation by reorganization of a dendritic network, *The Journal of Cell Biology*, **160** (2003), 409-421

[3] A. Schirenbeck, R. Arasada, T. Bretschneider, T.E.B. Stradal, M. Schleicher, J. Faix: The bundling activity of vasodilator-stimulated phosphoprotein is required for filopodium formation, *Proc. Natl. Acad. Sci. USA*, **103** (2006), 7694–7699

[4] A. Manhart, C. Schmeiser: Decay to equilibrium of the filament end density along the leading edge of the lamellipodium, *J. Math. Biol.*, **74** (2017), 169-193