Complexity Reduction for the TASEP Master Equation

Universität Bayreuth

Kilian Pioch

February 17, 2023

Contents

Chapter 1

Introduction

The following work is a continuation of my Bachelors Thesis and serves as a preliminary study for a subsequent doctoral dissertation. As such it is concerned with providing a solid foundation for further research and a generalization of the models studied during my bachelors degree. It is my belief that the results of this work will also aid a number of biological and physical research projects, which could benefit from the numerical and computational improvements that will be presented. My aim for this thesis is to introduce a general model for arbitrary TASEP networks as well as an array of solution methods for these. These methods will range from an exact solution, except for errors made by numerical solvers, to very simple approximations. Another generalization that was achieved is the calculation of solutions to arbitrary networks.

This work is part of a research grant by the DFG and under the supervision of Prof. Thomas Kriecherbauer and Prof. Lars Grüne. I want to express my gratitude towards them for their continued support and encouragement during my time working on this problem.

Chapter 2

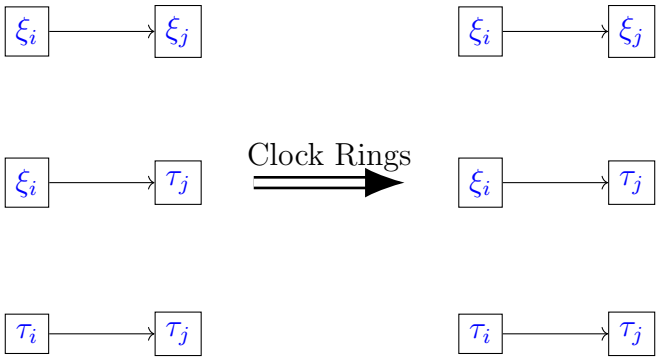
Models and Notation

2.1 The TASEP Model

Definition

The TASEP (totally asymmetric simple exclusion process) and ASEP (asymmetric simple exclusion process) are used to model the stochastic movement of particles on a directed graph consisting of N vertices M and edges. It was originally introduced to model the movement of ribosomes along a one dimensional chain of mRNA base pairs by MacDonald and Gibbs [1]. It has since found usage in simulations of pedestrian and vehicular traffic [2] [3] as well as simulation of cellular automata. The term TASEP does not describe one specific mathematical model and as such a handful of different interpretations have been developed over the years. In our model each node can only be occupied by a single particle or be empty. This property is fundamental to all TASEP and ASEP systems and is found across basically all works regarding this topic. The dynamics of the system we are modeling in this work arise from assigning an independent random clock with a Poisson distribution to each edge. In general the laws of movement in the chain can be understood as follows. If the node i is occupied and the connected j node is empty then a jump occurs once the corresponding clock rings. If we assume τ_i to denote if the node with index i is occupied and ξ_i if it is empty, we can schematically describe situations in which jumps can and cannot occur as follows.

In this case a jump can occur. The three remaining cases prohibit a jump, these are pictured below.



In these cases, a jump cannot happen. In the upper two cases it is of course because there exist no particles that could jump into the adjacent node. In the lower case it is because the adjacent node is already occupied and thus the particle in node i cannot move into node j. This is different from the model used in [2] for example, where the movement of all particles is applied simultaneously. This leads to different dynamics of the systems, one main difference being the formation of particle chains that move through the lattice as one unbreakable unit in the case of global updates. In our case such chains cannot occur. This is because of two reasons. Firstly if the node to which the connection leads is occupied when the clock rings, a jump cannot happen, thus only the leading particle of a chain can jump, while the rest of the chain stays in its original position. Secondly while simultaneous jumps can occur, these events represent a null set. This entails that the solution is not influenced by them.

To allow for particles to flow through the lattice we have to make an addition to the model. For this imagine two infinite reservoirs. One of which is always filled completely and particles can enter into a node i if it is empty and a corresponding clock rings, the other one is an empty one, into which particles can leave the lattice under the same constraints. In the figure below a simple one dimensional chain is portrayed. This lattice configuration could be used to model mRNA translation for example. The arrows not connected to a node on one side symbolize the connections from the infinitely filled as well as into the infinitely empty reservoirs.

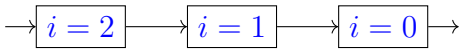
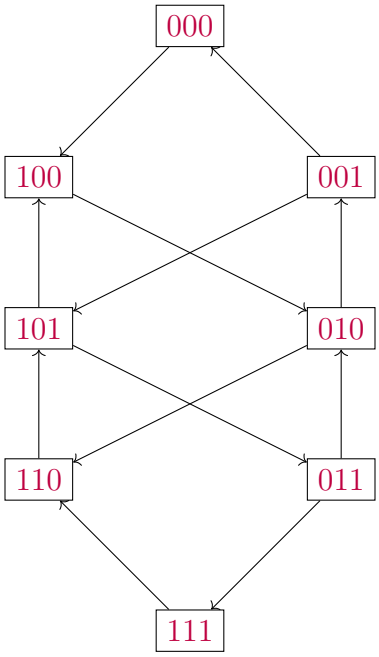


Figure 2.1: A representation of a lattice with 3 nodes, which particles can enter in node 2 and leave in node 0

Our interest now lies in the movement of particles along this chain. However, all movement along the lattice is stochastic in nature. This means that we can analyze either the behavior of the paths of the markov-chain, that means the movement of individual particles, or the probability distribution for occupation of the vertices of the graph. In this work we are concerned with the expected values of occupation as well as the probability distribution of the network states. As such the paths of the markov-chain will not be discussed further, additional research into this topic was done by Grüne, Kriecherbauer and Margaliot

in [4]. Going further we will denote the node with index i being in the occupied state at time t by $\tau_i(t)$. Analog to this the empty state will be written as $\xi_i(t)$. We will write the expected value of node i being occupied as $\langle \tau_i \rangle(t)$. This notation will be expanded later to accommodate more complex lattices and expected values for combinations of nodes. The question that now arises is how the expected value of occupancy can be calculated. For this we have to introduce a way to describe movement in probabilistic terms as well as a way to encode every possible lattice configuration. A lattice configuration or lattice state is a specific arrangement of particles on the nodes of the lattice. For this we employ binary numbers with a 1 as the digit at position i representing τ_i and a 0 at position j representing ξ_j. For example in the lattice above, the state in which node 0 and node 1 are occupied with node 2 being empty, would be enumerated by $(011)_2 = (3)_{10}$. This way we can enumerate all possible 2^N states into binary numbers with N bits. We can then find every possible state transition and the corresponding particle exchange. Graphically these can be expressed in the following way.



In the above figure, every possible state transition is marked between the two corresponding state numbers in base 2.

2.1.1 Preliminary Definitions

It is our aim to rigorously describe the dynamics of TASEP and ASEP. As such a few basic definitions are needed.

Definition 2.1.1. *A Directed Graph is a pair $G = (V, E)$ where $V \subset \mathbb{N}$ is a set of vertices and E is a set of edges, which are ordered pairs of distinct vertices, such that:*

$$E \subseteq \{(i,j)|(i,j) \in V^2, i \neq j\}.$$

In our case we enumerate the nodes starting from 0 and ending at $N-1$.

Definition 2.1.2. *A Weighted Directed Graph or a Directed Network is a graph $G = (V, E)$ where a weight $g_{i,j} \in \mathbb{R}^{\geq 0}$ is assigned to each edge $(i,j) \in E$ of the graph. The weight of the edge (i,j) is denoted as $g_{i,j}$.*

Note: In this work we will use the terms "node" and "vertex" interchangeably. In general we are interested in adjacent vertices, as movement can only occur if two nodes are connected. For this we define the so called adjacency Matrix in the following way.

Definition 2.1.3. *A matrix $A \in \mathbb{R}^{N \times N}$ is called adjacency matrix for a directed network $G = (V, E)$ if the entries $a_{i,j}$ of the Matrix are of the form*

$$a_{i,j} = \begin{cases} g_{i,j} & (i,j) \in E \\ 0 & otherwise \end{cases}$$

It follows immediately that the number of nonzero entries of this adjacency matrix is equal to the number M of edges in the Network. Furthermore we define an *induced subgraph* as follows.

Definition 2.1.4. *An induced subgraph $g_k = (\tilde{V}, \tilde{E})$ with the index $k \in \mathbb{N}$ with $0 \leq k \leq 2^N$ is a subset of a graph $G = (V, E)$ with N vertices if it satisfies the following conditions.*

1. *$V \ni i \in \tilde{V}$, if the $i-th$ digit of the binary representation of k is equal to* 1
2. *$(i,j) \in \tilde{E}$, if $\{i,j\} \subset \tilde{V}$.*

As was alluded to earlier, the movement of particles is governed by poisson-processes attached to each edge. These are all independent in our case. This simply means, that every edge is equipped with a unique process, which is not dependent on any other attributes of the system.

Definition 2.1.5. *A discrete random variable X has a poisson distribution, with a parameter or rate $\lambda \in \mathbb{R}^{\geq 0}$ if it has a probability mass function given by:*

$$f(k;\lambda) = \mathbb{P}(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

where k denotes the number of occurrences.

Equipped with these definitions we can move on to discussing the different approaches of describing the dynamics of a TASEP or ASEP system.

2.1.2 The Master Equation

TASEP can be understood as a stochastical process, which describes the movement of particles subject to Poisson processes attached to connections between nodes. Since these movements change the state of the system, different sequences of these "clocks" ringing will result in different paths of the system through the state space. This allows us to formulate a probability measure, which describes the distribution of paths on the state space. In other words, the measure describes how many of the paths are currently located in some state. Since the exact description of such systems is not directly related to the results presented in this work we will not go into further detail here. A more rigorous examination of this approach to TASEP systems can be found in [4], while stochastic processes are more thoroughly discussed in [**?**](hier noch Lehrbuch verweis). However, the Kolmogorov Equations for continuous time Markov Processes, which TASEP is an example of, allow us to formulate a system of differential equations, that describes the evolution of the probability measure of the paths over time. In particular, the Kolmogorov backward equation is also known as the Master Equation for these systems. The master equation is dependent on the transition rate matrix of the markov process. In our case, this matrix is comprised of the Poisson rates of the connections of the system. We will motivate the derivation of the master equation in the following section. The enumeration of the states is carried over from earlier. The rates of the Poisson processes are written as g_{ij} for the internal rates from node i to node j as well as α and β for the entry and exit rate respectively. For example, we can now assume, that the system described in figure 2.1 is in the state (010) with the binary representation equal to $(2)_{10}$ with some probability $\mathbb{P}$. We write this as $\mathbb{P}(X(t) = s_2)$. It is now possible to formulate the expression for this probability in the following manner.

$$\begin{aligned}
\mathbb{P}(X(t+\Delta t) = s_2) = &\ \mathbb{P}(X(t) = s_0)\alpha g_{2,1}(\Delta t)^2 \\
&+ \mathbb{P}(X(t) = s_1)\beta\alpha g_{2,1}(\Delta t)^3 \\
&- \mathbb{P}(X(t) = s_2)(1-\alpha-g_{1,0})\Delta t \\
&+ \mathbb{P}(X(t) = s_3)\beta\Delta t \\
&+ \mathbb{P}(X(t) = s_4)g_{2,1}\Delta t \\
&+ \mathbb{P}(X(t) = s_5)\beta g_{2,1}(\Delta t)^2 \\
&+ \mathbb{P}(X(t) = s_6)g_{1,0}\beta g_{2,1}(\Delta t)^3 \\
&+ \mathbb{P}(X(t) = s_7)\beta^2 g_{1,0}g_{2,1}(\Delta t)^4
\end{aligned} \tag{2.1}$$

We can rewrite this equation by summarizing all terms in which Δt has a higher exponent than 1. This then yields the following.

$$\begin{aligned}
\mathbb{P}(X(t+\Delta t) = s_2) = &- \mathbb{P}(X(t) = s_2)(1-\alpha-g_{1,0})\Delta t \\
&+ \mathbb{P}(X(t) = s_3)\beta\Delta t \\
&+ \mathbb{P}(X(t) = s_4)g_{2,1}\Delta t \\
&+ O(\Delta t^2)
\end{aligned} \tag{2.2}$$

This allows us to now rearrange the equation into a differential quotient, which yields the next expression.

$$\frac{1}{\Delta t}[\mathbb{P}(X(t+\Delta t)=s_2)-\mathbb{P}(X(t)=s_2)] = \\ -(g_{1,0}-\alpha)\mathbb{P}(X(t)=2)+g_{2,1}\mathbb{P}(X(t)=s_4)+\beta\mathbb{P}(X(t)=s_3)+O(\Delta t), \tag{2.3}$$

which, when $\Delta t \to 0$, yields an ordinary differential equation for the probability of the system being in state 2. Note that all of the expressions we lumped together in $O(\Delta t)$ have also gone to 0 in the limit, which means that those expressions have no impact on the change of the probability distribution, as they constitute a null-set. The differential equation for $\mathbb{P}(X(t)=s_2)$ then reads as:

$$\frac{d}{dt}\mathbb{P}(X(t)=s_2) = -(g_{1,0}-\alpha)\mathbb{P}(X(t)=2)+g_{2,1}\mathbb{P}(X(t)=s_4)+\beta\mathbb{P}(X(t)=s_3) \tag{2.4}$$

It is useful to simplify this notation. To achieve this, we define a vector $x \in \mathbb{R}^{2^N}$ where $x_i(t) = \mathbb{P}(X(t)=s_i)$, as well as a matrix $\mathbf{A}$ for the rates of the Poisson processes. This matrix is also known as the transition rate matrix for the Markov process. We can then describe the entire system of ODEs for every possible state of the lattice as:

$$\dot{x}(t) = \mathbf{A}x(t) \tag{2.5}$$

This expression is known as the *Master Equation* for the TASEP. For the lattice in figure 2.1 the master equation would have the following matrix $\mathbf{A}$:

$$\begin{bmatrix} -\alpha & \beta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\alpha-\beta & g_{1,0} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\alpha-g_{1,0} & \beta & g_{2,1} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\alpha-\beta & 0 & g_{2,1} & 0 & 0 \\ \alpha & 0 & 0 & 0 & -g_{2,1} & \beta & 0 & 0 \\ 0 & \alpha & 0 & 0 & 0 & -g_{2,1}-\beta & g_{1,0} & 0 \\ 0 & 0 & \alpha & 0 & 0 & 0 & -g_{1,0} & \beta \\ 0 & 0 & 0 & \alpha & 0 & 0 & 0 & -\beta \end{bmatrix} \tag{2.6}$$

This kind of approach is also known as a *top down* model, since every possible state of the system is calculated. To calculate the expected value $\langle\tau_i\rangle(t)$ for some node i, the sum of the probability of every state in which node i is occupied needs to be evaluated. In this case the three equations for the expected values read:

$$\begin{aligned} \langle\tau_0\rangle(t) &= \sum_{i=1}^{4} x_{2i-1}(t) \\ \langle\tau_1\rangle(t) &= \sum_{i\in\{2,3,6,7\}} x_i(t) \\ \langle\tau_2\rangle(t) &= \sum_{i=4}^{7} x_i(t) \end{aligned} \tag{2.7}$$

We can also expand this derivation to Poisson processes that have a time dependent mean rate, in which case all rates become dependent on some abstract value of time denoted by t. Previous publications were able to prove that the master equation will reach a steady state, if the rates are constant [**?**]. In the case of periodic jump rates it is possible to prove entrainment of the expected values to the periodicity of the corresponding rates. [5] One major drawback of the *top down* approaches are their sheer size. As is shown above, the ODE-system grows exponentially with the amound of nodes inside the lattice. This has many serious ramifications, foremost of which is the amount of memory and processing time needed to calculate solutions. Quick, back of the napkin math suggests that to calculate the solution of a system with a few hundred nodes with this model, more storage capacity is needed, than the entire internet provides to date. We undertook efforts to reduce the memory and computational footprint of our programs. These improvements allowed us to calculate systems with up to a few dozen nodes. This is still not enough nodes to simulate anything meaningful, as mRNA chains can contain several hundred nodes for ribosomes to attach to [**?**]. In the case of traffic simulations many thousands of nodes might have to be included to achieve reliable and meaningful results. These requirements necessitate an approximation that behaves more favorably with growing lattices.

2.1.3 The Mean Field Approximation

As was shown before, the master equation is not suitable for calculating solutions to large systems. Because of this, other solution methods have been developed over the years. One of these is the so called mean field approximation. The general idea stems from an inversion of the approach of the master equation. Instead of a *top down* approach, in which the probability for every possible state is calculated, the mean field approximations utilize so called *bottom up* approaches. To motivate this approach we first look at another way of describing the expected values for the occupancy of nodes. Because the values of the nodes can either be 0 or 1, calculating the expected value boils down to summing all probabilities for the nodes being occupied. To ease notation, we define another probability measure on our system. Instead of a measure for all the system states we define $\mathbb{P}(\tau_i)$ as the probability of node i being occupied. Because of the binary node states, it holds that $\langle \tau_i \rangle = \mathbb{P}(\tau_i)$. Note that this is the same expression as is found in equation (2.7). However we can now expand this expression to multiple nodes as well. We write the probability of the system being in a state in which nodes i and j are occupied as $\mathbb{P}(\tau_i \tau_j)$. This can be understood as the sum of all probabilities for states in which node i and j are occupied. Rewriting the probabilities in this manner allows us to formulate an expression for the change of the probability of occupation of arbitrary collections of nodes, as opposed to the formulation of the master equation, which was describing system states. In a manner similar to before we can now express the change in probability of some node i being occupied in terms of these expressions. We can write the probability for node i being occupied when some amount of time Δt has passed as $\mathbb{P}(\tau_i(t + \Delta t))$. Describing the correlation of occupied and unoccupied

nodes then yields the following expression for the one dimensional TASEP.

$$\begin{aligned}\mathbb{P}(\tau_i(t+\Delta t)) = \mathbb{P}(\tau_i(t)) \\ - \mathbb{P}(\tau_i(t), \xi_{i+1}(t)) g_{i,i+1} \Delta t \\ + \mathbb{P}(\tau_{i-1}(t), \xi_i(t)) g_{i-1,i} \Delta t\end{aligned} \tag{2.8}$$

This expression is still exact. The reason for this is that until now we have just rewritten sums of specific lattice state probabilities. Because of linearity, we can factor out the Poisson rates $g_{i,j}$ for the relevant state transitions from the sums. Rearranging the equation yields the following expression for the differential quotient.

$$\begin{aligned}\frac{\mathbb{P}(\tau_i(t+\Delta t)) - \mathbb{P}(\tau_i(t))}{\Delta t} = - \mathbb{P}(\tau_{i+1}(t), \xi_i(t)) g_{i+1,i} \\ + \mathbb{P}(\tau_i(t), \xi_{i-1}(t)) g_{i,i-1},\end{aligned} \tag{2.9}$$

which when $\Delta t \to 0$ and rewriting the occupancy in terms of expected value yields

$$\frac{d}{dt}\langle \tau_i \rangle(t) = g_{i,i-1} \langle \tau_i \xi_{i-1} \rangle(t) - g_{i,i-1} \langle \tau_i \xi_{i-1} \rangle(t). \tag{2.10}$$

This expression is also exact, however the two terms on the right hand side contain expected values for the occupancy of two nodes. Since these values are not known, we cannot solve this ODE. It is of course possible to run trough the same procedure for the new expressions. This would yield the following ODEs.

$$\begin{aligned}\frac{d}{dt}\langle \tau_{i+1} \xi_i \rangle(t) = - \langle \tau_{i+1} \xi_i \rangle(t) g_{i+1,i} \\ + \langle \tau_{i+1} \tau_i \xi_{i-1} \rangle(t) g_{i,i-1} \\ + \langle \tau_{i+2} \xi_{i+1} \xi_i \rangle(t) g_{i+2,i+1}\end{aligned} \tag{2.11}$$

$$\begin{aligned}\frac{d}{dt}\langle \tau_i \xi_{i-1} \rangle(t) = - \langle \tau_i \xi_{i-1} \rangle(t) g_{i,i-1} \\ + \langle \tau_{i+1} \xi_i \xi_{i-1} \rangle(t) g_{i+1,i} \\ + \langle \tau_i \tau_{i-1} \xi_{i-2} \rangle(t) g_{i-1,i-2}\end{aligned} \tag{2.12}$$

Again the same construction of sums of probabilities for specific states was used here. However, the same issue as before persists here as well. In general, every ODE describing some moment of order k is dependent on moments of order $k+1$. We can describe these dependencies in a more general manner. To achieve this we denote with $s_n^d = s_{n-1}, \cdots, s_0$ the binary number with n digits which has a decimal representation of d. In mathematical terms this means that $(s_{n-1} \cdots s_1 s_0)_2 = (d)_{10}$. Additionally the expression $s_n^d(i)$ is used to describe the value of s_i We generalize the expected values $\langle \tau_i \rangle$ to higher order moments in the following way.

$$T(s_n^d, k) = \langle \tau_{k+n} \xi_{k+n-1} \cdots \tau_k \rangle \tag{2.13}$$

Furthermore we denote the entry flow rate into node i with α_i as well as the exit flow rate out of node i with β_i. Lattice internal flow rates are described the same way as earlier. The general ODE for arbitrary moments on one dimensional lattices is constructed as follows. First of all we sum over all state changes which result in the state $T(s_n^d, k)$ and are facilitated by particles leaving or entering the lattice. It follows that these sums are over states in which there is one particle missing at the node with index $k+i$, compared to the original moment, ie. $T(s_n^{d-2^i}, k)$ as well as states in which there is an additional particle at position $k+i$, ie. $T(s_n^{d+2^i}, k)$. We will be utilizing the notation "+=" to denote assigning the sum of the right and left hand side to the left hand side, similar to how this symbol is used in programming languages.

$$\frac{d}{dt}T(s_n^d, k)+ = \sum_{i=0}^{n-1}\left[\alpha_{k+i}T(s_n^{d-2^i}, k)\big|_{s_n^d(i)=1} + \beta_{k+i}T(s_n^{d+2^i}, k)\big|_{s_n^d(i)=0}\right] \tag{2.14}$$

Similarly we need to subtract the sum over all state transitions away from $T(s_n^d, k)$ through particles leaving or entering the lattice. This part of the ODE then reads as follows.

$$\frac{d}{dt}T(s_n^d, k)- = \sum_{i=0}^{n-1}\left[\alpha_{k+i}T(s_n^{d}, k)\big|_{s_n^d(i)=0} + \beta_{k+i}T(s_n^{d}, k)\big|_{s_n^d(i)=1}\right] \tag{2.15}$$

Note that in the case of the 1d-TASEP where the lattice only has one entry and exit point, these expressions can be simplified by omitting the sum. They would then read as

$$\begin{aligned}\frac{d}{dt}T(s_n^d, k)+ = &\left[\alpha T(s_n^{d-2^{n-1}}, k)\big|_{s_n^d(n)=1} + \beta T(s_n^{d+1}, k)\big|_{s_n^d(1)=0}\right] \\ &- \left[\alpha T(s_n^{d}, k)\big|_{s_n^d(n)=0} + \beta T(s_n^{d}, k)\big|_{s_n^d(1)=1}\right].\end{aligned} \tag{2.16}$$

Next we want to sum over all state changes in which the number of particles in the lattice does not change, only their positions. That means that a state in which node i is occupied and node $i+1$ is empty can be the result of this particle moving from node $i+1$. Subsequently the state that can produce this has the enumeration index $d + 2^{i+1} - 2^i$. First we sum over all transitions that increase the probability for the state $T(s_n^d, k)$ to occur.

$$T(s_n^d, k)+ = \sum_{i=0}^{n-2}\left[g_{k+i+1,k+i}T(s_n^{d-2^i+2^{i+1}}, k)\big|_{s_n^d(i)=1, s_n^d(i+1)=0}\right] \tag{2.17}$$

Like before, we also have to take the inverse process, meaning the reduction of the probability through particles moving inside the lattice, into account.

$$T(s_n^d, k)- = \sum_{i=0}^{n-2}\left[g_{k+i+1,k+i}T(s_n^{d}, k)\big|_{s_n^d(i+1)=1, s_n^d(i)=0}\right] \tag{2.18}$$

Lastly we want to account for state changes in which particles enter from or leave into lattice sites which are not included in $T(s_n^d, k)$. However in contrast to (2.14) and (2.15) the particles

only move to or from outside of the scope of the moment, while remaining inside the lattice. We begin by adding the positive probability change, since the lattice is one dimensional, there can be at most two such such cases. One is out of scope particles entering the moment from the left hand side.

$$T(s_n^d, k) += g_{k+n+1,k+n} T(s_{n+1}^{d+2^{n+k+1}-2^{n+k}}, k+1)\big|_{s_n^d(n+k)=1} \tag{2.19}$$

While the other is the case of a particle leaving from the node on the right most side.

$$T(s_n^d, k) += g_{k,k-1} T(s_{n+1}^{2*(d+1)}, k)\big|_{s_n^d(1)=0} \tag{2.20}$$

The last terms are the respective inversions of the two previous cases.

$$T(s_n^d, k) -= g_{k+n+1,k+n} T(s_{n+1}^{d+2^{n+k+1}}, k+1)\big|_{s_n^d(n+k)=0} \tag{2.21}$$

$$T(s_n^d, k) -= g_{k+n-1,k+n} T(s_{n+1}^{2*d}, k)\big|_{s_n^d(1)=1} \tag{2.22}$$

Note that here every vertical bar is to be understood as an *if* condition. More precisely, the respective summands are only to be used in the ODE if the condition marked on the bottom right is fulfilled, they should be left out otherwise. Another assumption we make in these ODEs is that all transition rates which are not defined, for example $g_{0,-1}$ or $g_{N,N-1}$ are equal to 0. With this we don't have to formulate individual formulae for edge cases. This expression allows us to calculate the mean field approximation of the master equation up to any length of moment.

It should also be noted, that for the case of $T(\cdot, N)$ this expression yields the master equation. This statement is the result of the way we constructed the mean field expressions. As was already discussed earlier, the probabilities shown here are the sums of state probabilities in which the occupation of some subset of nodes is known. It follows that higher order moments are the sums of states in which the state of more nodes is fixed. At $T(\cdot, N)$ we have fixed the occupational state of every node in the lattice, in other words, we describe the sum over only one network state. This means, that the full mean field description of a TASEP lattice is actually more complex than the master equation and would not lead to any tangible benefits during the solution process. To circumvent this issue, some method for closing the mean field ODEs at some order of moments has to be derived. Inspiration for this procedure was found in [6] where a similar issue for epidemiological simulations was discussed. This approach can be qualitatively motivated by looking at the impact of an occupied node to nodes further and further down the lattice. Assuming some node i to be occupied, we know that the expected value of occupancy of the next node $i-1$ is immediately correlated to node i being filled. Going one step further, node $i-2$ is correlated to node i only indirectly through node $i-1$. While correlation still exists, it is weakened by the fact that the middle node acts as a sort of buffer. The larger this buffer gets, the weaker the two nodes should be correlated. The general idea of closures is to assume that at some point, the correlation between nodes is small enough to introduce a small enough error, that the results are still

accurate to same order of consistency as the solver of the ODE. We can describe this with a so called correlation measure C_{XY} between the moments X and Y, given by

$$C_{X,Y} = \frac{\langle XY \rangle}{\langle X \rangle \langle Y \rangle}. \tag{2.23}$$

We know that if $C_{X,Y} = 1$ the two moments are uncorrelated. However, this will never be the case. This can be shown by writing an example for these moments as sums of state probabilities. These expressions are as follows.

$$\begin{aligned} \langle \tau_i \rangle(t) &= \sum_{k \in \{l | s_N^l(i)=1\}} \mathbb{P}(X(t) = s_k) \\ \langle \tau_j \rangle(t) &= \sum_{k \in \{l | s_N^l(j)=1\}} \mathbb{P}(X(t) = s_k) \\ \langle \tau_i \tau_j \rangle(t) &= \sum_{k \in \{l | s_N^l(i)=s_N^l(j)=1\}} \mathbb{P}(X(t) = s_k) \end{aligned} \tag{2.24}$$

It follows, that the expression $\langle \tau_i \rangle \langle \tau_j \rangle$ contains probabilities that don't occur in $\langle \tau_i \tau_j \rangle$. Namely all states which are of the form $\langle \xi_i \tau_j \rangle$ and $\langle \tau_i \xi_j \rangle$. Rearranging equation (2.23) yields an expression for closures of second order moments.

$$\langle XY \rangle = \langle X \rangle \langle Y \rangle C_{X,Y} \tag{2.25}$$

A similar derivation can be done for third order moments, resulting in closures of one of these forms.

$$\begin{aligned} \langle XYZ \rangle &= \langle X \rangle \langle Y \rangle \langle Z \rangle C_{X,Y,Z} \\ &= \langle XY \rangle \langle Z \rangle C_{XY,Z} \\ &= \frac{\langle XY \rangle \langle YZ \rangle}{\langle Y \rangle} C_{XY,YZ} \end{aligned} \tag{2.26}$$

These closures for third order moments are not unique anymore, hence choosing the one which introduces the smallest error is necessary. In general, it is not known which closure produces the most accurate results, however. If we now apply closures to the mean field approximations outlined in the equations (2.14) - (2.22), we can formulate reduced ODEs for TASEP systems. The simplest of which is also known as RFM or ribosome flow model. Applying it to the same 3-node chain for which the master equation was formulated previously, we get the following expression.

$$\begin{aligned} \frac{d}{dt}\langle \tau_0 \rangle &= g_{1,0} \langle \tau_1 \rangle \langle \xi_0 \rangle - \beta \langle \tau_0 \rangle \\ \frac{d}{dt}\langle \tau_1 \rangle &= g_{2,1} \langle \tau_1 \rangle \langle \xi_0 \rangle - g_{1,0} \langle \tau_1 \rangle \langle xi_0 \rangle \\ \frac{d}{dt}\langle \tau_2 \rangle &= \alpha \langle \xi_2 \rangle - g_{2,1} \langle \tau_2 \rangle \langle \xi_1 \rangle \end{aligned} \tag{2.27}$$

The simplification achieved by this should be immediately obvious. In general the mean field approach grows with $O(n)$ instead of $O(2^n)$ for the master equation. The consequence of this is that much larger networks can be simulated. In our case networks with around 10^5 nodes could be calculated. This approach can of course be expanded to closures of any moment. Though it will later be shown that the benefits of larger closures get outweighed by their drawbacks, foremost of which is numerical instability. Another issue are numerical errors caused by floating point arithmetic. These can even become so severe that the solution process fails. Last but not least, the ODE solver introduces an error.

2.1.4 Exact Solutions

To be able to verify solutions which exceed the size constraints of the master equation, we have also implemented another exact solution method. First of all it should be noted, that exact solutions exist only for a small subset of TASEP networks. The following section is largely based on work done by B. Derrida and M.R. Evans [7] [8]. In their work an exact solution was found for one dimensional lattices which have unit Poisson rates for all internal connections and time independent input and output rates α and β. The main idea of their approach is to first define the steady state of the system as the configuration where the change of the probabilities of every state of the system is 0.

$$\frac{d}{dt}\mathbb{P}(X(t) = s_i) = 0 \tag{2.28}$$

This probability can also be expressed with unnormalized weights f_N where the following holds.

$$\mathbb{P}(X(t) = s_i) = f_N(s_i)/Z_N \tag{2.29}$$

with

$$Z_N = \sum_{\tau_i=0,1} \cdots \sum_{\tau_i=0,1} f_N(\tau_0, \tau_1, \cdots, \tau_{N-1}). \tag{2.30}$$

It is then possible to find the value of f_N with the following expression.

$$f_N(\tau_0, \tau_1, \cdots, \tau_{N-1}) = \langle W| \prod_{i=0}^{N-1} (\tau_i D + (1-\tau_i)E)|V\rangle, \tag{2.31}$$

with D and E square matrices and $|V\rangle$ and $\langle W|$ vectors which satisfy these conditions:

$$\begin{aligned} DE &= D + E \\ D|V\rangle &= \frac{1}{\beta}|V\rangle \\ \langle W|E &= \frac{1}{\alpha}\langle W|. \end{aligned} \tag{2.32}$$

In short, the expression (2.31) tells us that any weight $f_N(s_i)$ is given by a product of matrices D and E with the first of these at position i if site i is occupied and the latter at

that position if it is empty. Furthermore, an additional matrix C can be defined by

$$C = D + E, \tag{2.33}$$

which allows us to define an expression for $\langle \tau_i \rangle$:

$$\langle \tau_i \rangle = \frac{\langle W | C^{i-1} D C^{N-i} | V \rangle}{\langle W | C^N | V \rangle}. \tag{2.34}$$

Note that these expressions can also be formulated for higher order moments. It can then be shown that the matrices D and E are one dimensional in the case that $\alpha + \beta = 1$. In this case the entire chain is uncorrelated and the matrices can be defined as $D = \frac{1}{\beta}$ and $E = \frac{1}{\alpha}$. In the case that $\alpha + \beta \neq 1$ the matrices are infinite dimensional. There are a number of matrices which could be used here [8]. Since infinite dimensional matrices are unwieldy to implement in numerical algorithms, we chose another approach, which is also outlined in the before mentioned sources. We use the following notation to make the expressions more legible.

$$\begin{aligned} Z_N &= \langle W | C^N | V \rangle \\ &= \sum_{p=1}^{N} \frac{p(2N-1-p)!}{N!(N-p)!} \frac{(\frac{1}{\beta})^{p+1} - (\frac{1}{\alpha})^{p+1}}{(\frac{1}{\beta}) - (\frac{1}{\alpha})} \\ B_{N,p} &= \begin{cases} \dfrac{p(2N-p-1)!}{N!(N-p)!} & 0 < p \leq N \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{2.35}$$

Note that the second formula for Z_N can be obtained by applying the commutation rules defined in (2.32). This allows us to formulate the expression for $\langle \tau_i \rangle$ as follows:

$$\langle \tau_i \rangle = \sum_{n=1}^{N-i} B_{n,1} \frac{Z_{N-n}}{Z_N} + \frac{Z_{i-1}}{Z_N} \sum_{p=1}^{N-i} B_{N-i,p} \frac{1}{\beta^{p+1}} \tag{2.36}$$

Note that now the enumeration of the nodes starts at 1. The main challenge in computing these values lies in the size of the values themselves. Already for chain sizes of 15 the factorials exceed the value range of standardized integer data types. The solution to this problem will be discussed in the Programs section.

2.1.5 Expansion to arbitrary Networks

In this section a generalization of the one dimensional TASEP will be derived. To motivate this we can assume that up until now every network had an adjacency matrix $adj(\mathbf{G})$ of the

following form:

$$adj(\mathbf{G}) = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \ddots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{pmatrix}, \tag{2.37}$$

which just describes a network in which every node $i+1$ leads either into the next node i or nowhere. If we also introduce entry and exit vectors A and B which contain values α_i and β_i for the rates with which particles enter or exit node i we can describe every possible network. To describe a 1-dimensional TASEP lattice, these vectors would read:

$$A = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \alpha \end{pmatrix}, \quad B = \begin{pmatrix} \beta \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \tag{2.38}$$

Note that the rules of movement outlined earlier still apply here. As such we can still describe state transitions using state probabilities of the lattice. The reasoning for this analogously follows the reasoning of the 1-dimensional case outlined in (2.1) and (2.3). Numerically the only major difference is that the matrix $\mathbf{A}$ of the master equation gets denser if the lattice is more highly connected. Aside from that the algorithmic derivation as well as the solving process are identical.

The mean field approximation on the other hand is much more cumbersome to evaluate and derive. We begin by redefining a few expressions from the 1-dimensional case. First of all we want to generalize the expression for the moments to arbitrary subnetworks of the lattice. For this we enumerate these partial lattices similarly to the enumeration of the lattice states, where a binary number is used to describe every possible sub lattice. We write $g(k)$ as the set of nodes of the lattice in which every node with index i is included if the ith digit of the binary representation of k is equal to 1. Furthermore we describe with $adj(g(k))$ the adjacency matrix describing the lattice given by $g(k)$. Note that these will be the same dimension as the original adjacency matrix $adj(\mathbf{G})$ as the numeration of the nodes is not changed and it can be trivially expanded to the original dimensionality. We can then analogously to (2.13) describe a moment of a subnetwork as follows

$$\tilde{T}(s_n^d, g(k)) = \langle \tau_{i_0} \xi_{i_1} \cdots \tau_{i_{n-1}} \rangle \tag{2.39}$$

Where again τ denotes some node being full and ξ denoting it being empty. The indexes have to be subindexed as the nodes don't have to be sequentially ordered for them to be connected. In this case n is the amount of nodes in the subnet and the definition of s_n^d as

well as $s_n^d(i)$ has not changed. The indices i_k are the elements of the Vertex set $\tilde{V}_{g(k)}$ of $g(k)$. We can now formulate the general ODE for a moment $\tilde{T}(s_n^d, g(k))$. This is again similar to the equations (2.14) - (2.22). We begin by formulating an expression for the increase in probability through particles entering or leaving the lattice from or into the infinite reservoirs.

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))+ = \sum_{i=0}^{N-1}\left[\alpha_i \tilde{T}(s_n^{d-2^i}, g(k))\big|_{i\in\tilde{V}_{g(k)}, s_n^d(i)=1} + \beta_i \tilde{T}(s_n^{d+2^i}, g(k))\big|_{i\subset\tilde{V}_{g(k)}, s_n^d(i)=0}\right] \tag{2.40}$$

Analogously we can sum all state transitions for particles leaving the lattice, thus deteriorating the state. The corresponding expression is

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))- = \sum_{i=0}^{N-1}\left[\alpha_i \tilde{T}(s_n^d, g(k))\big|_{i\subset\tilde{V}_{g(k)}, s_n^d(i)=0} + \beta_i \tilde{T}(s_n^d, g(k))\big|_{i\subset\tilde{V}_{g(k)}, s_n^d(i)=1}\right]. \tag{2.41}$$

Next we want to sum all positive probability changes which are facilitated by particles moving inside of the lattice defined by $g(k)$ and s_n^d. We know that every time two adjacent nodes are in the configuration (01) the state can be the result of another state in which has the same configuration except the occupation of these two nodes is inverted. We have to take the sum over all possible combinations of nodes of the sub network, to make sure that all possible connections are checked.

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))+ = \sum_{i=0}^{n-1}\left[\sum_{j=0}^{n-1} adj(g(k))_{ij}\tilde{T}(s_n^{d-2^j+2^i}, g(k))\big|_{\{i,j\}\ \subset\tilde{V}_{g(k)}, s_n^d(i)=0, s_n^d(j)=1}\right] \tag{2.42}$$

It follows, that the same process can also reduce the probability for the lattice configuration. As such we also have to check all possibilities for the deterioration of this state. The corresponding equation reads as

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))- = \sum_{i=0}^{n-1}\left[\sum_{j=0}^{n-1} adj(g(k))_{ij}\tilde{T}(s_n^d, g(k))\big|_{\{i,j\}\ \subset\tilde{V}_{g(k)}, s_n^d(i)=1, s_n^d(j)=0}\right] \tag{2.43}$$

Now we have to account for all probability increases which are dependent on particles leaving the sub network, but remaining inside of the full network, as well as increases which are due to particles entering the sub lattice from nodes which are not an element of $\tilde{V}_{g(k)}$. This can be done by constructing all possible sub networks which include $g(k)$ as well as one additional node out of the full lattice. First we formulate the expression for an increase in probability due to a particle from outside the sub net entering.

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))+ = \sum_{i=0}^{N-1}\left[\sum_{j=0}^{n-1} adj(g(k+2^i))_{ij}\tilde{T}(s_{n+1}^{d+2^i-2^j}, g(k+2^i))\big|_{\{i,j\}\ \subset\tilde{V}_{g(k+2^i)}\subset\mathbf{G}, i\notin\tilde{V}_{g(k)}, s_n^d(j)=1}\right] \tag{2.44}$$

Next we write the formula for the increase in probability due to a particle leaving into a lattice node, which is not in the sub network. Here we assume, that the occupation of the

added node is always determined by the most significant bit in $(d)_{10}$. This is mainly because it simplifies notation.

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))+ = \sum_{i=0}^{N-1}\left[\sum_{j=0}^{n-1} adj(g(k+2^i))_{ji}\tilde{T}(s_{n+1}^{d+2^n}, g(k+2^i))\Big|_{\{i,j\}\ \subset g(k+2^i)\subset \mathbf{G}, i\not\subset \tilde{V}_{g(k)}, s_n^d(j)=0}\right] \tag{2.45}$$

At last we have to take the analogous inverse cases of the two previous cases into account. That means the decrease of probability due to particles leaving or entering from or into sub network nodes into or from the full network. First the expression for particles leaving, thus decreasing probability. We make the same assumption for the node being determined by the most significant bit.

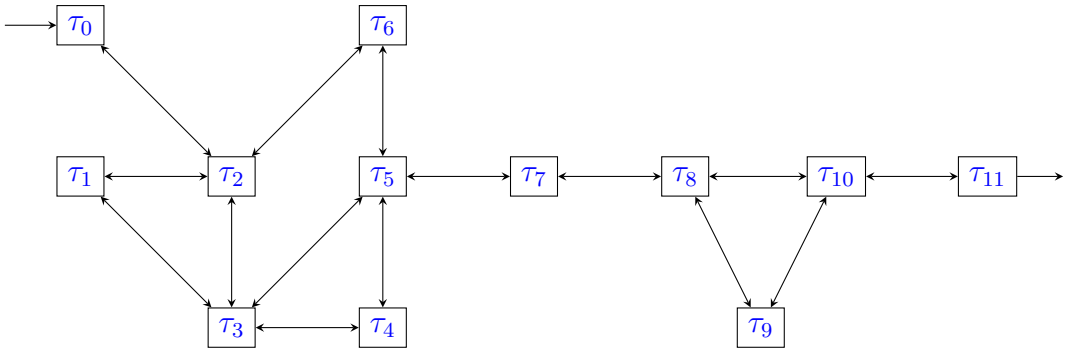
$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))- = \sum_{i=0}^{N-1}\left[\sum_{j=0}^{n-1} adj(g(k+2^i))_{ji}\tilde{T}(s_{n+1}^{d}, g(k+2^i))\Big|_{\{i,j\}\ \subset g(k+2^i)\subset \mathbf{G}, i\not\subset \tilde{V}_{g(k)}, s_n^d(j)=1}\right] \tag{2.46}$$

And lastly the decrease due to particles entering an empty sub network node.

$$\frac{d}{dt}\tilde{T}(s_n^d, g(k))- = \sum_{i=0}^{N-1}\left[\sum_{j=0}^{n-1} adj(g(k+2^i))_{ij}\tilde{T}(s_{n+1}^{d+2^n}, g(k+2^i))\Big|_{\{i,j\}\ \subset g(k+2^i)\subset \mathbf{G}, i\not\subset \tilde{V}_{g(k)}, s_n^d(j)=0}\right] \tag{2.47}$$

This equation is also exact, if the moments of every order of moments on every subnetwork would be calculated. In addition to the 2^N different lattice configurations, the size of the ODE is increased by another factor of 2^N because the moments of every partial lattice have to be evaluated. In Practice most of these equations will be discarded, however, as non connected networks can be handled individually and we are only interested in the equations needed to compute closures of some size $n < N$. These closures present an additional challenge as there are now many more possible closure configurations to take into account.

AB HIER ALLES IGNORIEREN



TASEP in traffic simulation

The *TASEP* model is also applied in traffic simulation, where it is used to simulate traffic jams and their behaviour. Slowly moving traffic can be described as a lattice of either

occupied or empty sites which vehicles can enter and leave in only one direction. Each site can then be attributed with a flow rate, describing the average time an object occupies this site until it moves on to the next one, given that it is empty. One example for this would be a one lane road with heavy traffic. If there is a point of congestion along the way, it could be described by the flow rate of one site inside the lattice being lower than that of the proceeding sites. Computing the trajectories of the corresponding differential equation yields the probabilities of occupancy of the sites over time. This allows for the examination of traffic behaviour.

2.2 Comparison of Mean-Field and Master Equation Methods

The master equation

Consider a stochastic system which can exist in one of N possible states. For each state there exists an expression $x_i(t)$, denoting the probability that a given system is in the configuration i at time t. The *master equation* of that system then describes the changes of these probabilities over time. It finds use in various disciplines including physics [9], chemistry [10], and traffic simulation [11]. To describe it accurately it is necessary to know the transition rates p_{ij} between the states i and j. The *totally asymmetric simple exclusion principle (TASEP)* describes a lattice of n consecutive sites, each of which can either be free or occupied by a particle. This structure allows every possible state to be encoded into a binary number in which empty sites correspond to zeroes and occupied sites to ones. The number of possible states N the system can be in grows exponentially with the number of sites of the lattice. This leads to $N = 2^n$. The same also applies for the dimension of the *master equation*, because it takes every possible lattice configuration into account. As such N also denotes the dimension of its system matrix. This leads to any solving attempt being very hard to compute even for relatively small values of n. For every site in such a system there exists a non-zero translation rate which describes the flow of particles from that site to the next. This approach describes the behaviour of a *TASEP* bound system more accurately than the *mean field* approach which will be shown later.

The correlation functions

To motivate the expressions used in the *mean field* approach a short derivation of the correlation functions will be given. These functions are the basis for the approximations that are used later on. For a more in depth derivation consider [12]. Consider the following expression, which accurately describes the change in probability of occupancy of any non-boundary lattice site.

$$\tau(t+\Delta t) = \begin{cases} \tau_i(t), & \text{with probability } 1-(h_{i-1}+h_i)\ \Delta t \\ \tau_i(t) + [1-t_i(t)]\tau_{i-1}(t), & \text{with probability } h_{i-1}\ \Delta t \\ \tau_i \tau i + 1, & \text{with probability } h_i\ \Delta t \end{cases} \tag{2.48}$$

The first line describes the change in probability if neither site $i-1$ nor site i is updated. Therefore τ_i does not change. The second and third equation describe the a change in occupancy of sites $i-1$ or i respectively. The expected value for τ_i can then be calculated as follows.

$$\langle \tau_i(t+\Delta t)\rangle = \langle \tau_i(t)\rangle + \langle (1-\tau_i(t))\tau_{i-1}(t)\rangle h_{i-1}\Delta t + \langle \tau_i(t)(\tau_{i+1}(t)-1)\rangle h_i \Delta t \tag{2.49}$$

The rate of change of the expected values can the be obtained by taking the limit over the above equation, which leads to the following expression.

$$\frac{d}{dt}\langle\tau_i(t)\rangle = \lim_{\Delta t\to 0}\frac{\langle\tau_i(t+\Delta t)\rangle - \langle\tau_i(t)\rangle}{\Delta t} = \langle(1-\tau_i(t))\tau_{i-1}(t)\rangle h_{i-1} + \langle\tau_i(t)(\tau_{i+1}(t)-1)\rangle h_i \quad (2.50)$$

The above can also be expanded to

$$\frac{d}{dt}\langle\tau_i(t)\rangle = h_{i-1}(\langle\tau_{i-1}(t)\rangle - \langle\tau_{i-1}(t)\tau_i(t)\rangle) + h_i(\langle\tau_i(t)\tau_{i+1}(t)\rangle - \tau_i(t)) \quad (2.51)$$

The rate of change of the resulting product terms can also be described. In a similar fashion to the derivation above, the change of $\langle\tau_i\tau_{i+1}\rangle$ can be expressed as the following. Note: The dependency of τ_i on t was omitted to improve readability.

$$\frac{d}{dt}\langle\tau_i\tau_{i+1}\rangle = \langle\tau_i\tau_{i+1}(\tau_{i+2}-1)\rangle h_{i+1} + \langle\tau_{i-1}(1-\tau_i)\tau_{i+1}\rangle h_{i-1} \quad (2.52)$$

Which, when expanded gives

$$\frac{d}{dt}\langle\tau_i\tau_{i+1}\rangle = (\langle\tau_i\tau_{i+1}\tau_{i+2}\rangle - \langle\tau_i\tau_{i+1}\rangle)h_{i+1} + (\langle\tau_{i-1}\tau_{i+1}\rangle - \langle\tau_{i-1}\tau_i\tau_{i+1}\rangle)h_{i-1} \quad (2.53)$$

These equations are exact and describe the change of the expected values. However, to calculate for example (2.51), one needs to know the solution to (2.53). This would require the knowledge of $\langle\tau_{i-1}\tau_{i+1}\rangle$ and $\langle\tau_{i-1}\tau_i\tau_{i+1}\rangle$. As such the problem is a N-body problem. The calculation of any correlation function requires the knowledge of all other correlation functions. The goal is then to approximate the correlation functions, such that no additional knowledge of other functions is required. This leads to the next section.

The mean field approach

In *mean field theory* the behaviour of complex high-dimensional stochastic models is approximated by simplifying the effects of every other particle on any given particle inside a system down to the effect of only one particle on that given particle. This can be achieved by averaging the effects of the other particles or ignoring them [7]. This effectively narrows a many body problem down to a single body problem, reducing its complexity and easing the computation of its solutions. In the case of *TASEP* one could reasonably assume, that the probability $x_i(t)$ for any site i to be occupied only depends on the probability of the preceding site being occupied ($x_{i-1}(t)$) and the probability of the following site being empty ($1 - x_{i+1}(t)$).

To simplify the complex n-body problem of *TASEP*, one can approximate the unknown correlation functions in (2.50) and (2.52). In the case of (2.50) this leads to the following approximations:

$$\langle(1-\tau_i(t))\tau_{i-1}(t)\rangle h_{i-1} + \langle\tau_i(t)(\tau_{i+1}(t)-1)\rangle h_i \approx h_{i-1}\langle\tau_{i-1}\rangle\langle 1-\tau_i\rangle + h_i\langle\tau_i\rangle\langle\tau_{i+1}-1\rangle \quad (2.54)$$

Because of $E[XY] \neq E[X] * E[Y]$ if X and Y are stochastically dependent, an error is introduced by this approximation. The approximation above leads to the following system of n first-order nonlinear ordinary differential equations. It can also be understood as the continuity equation for this process. As the change of the expected values of occupancy of the sites are equal to the ingoing flux minus the outgoing flux.

$$\begin{aligned}
\dot{x}_1(t) &= \alpha(t)(1 - x_1) - h_1(t)x_1(1 - x_2) \\
\dot{x}_2(t) &= h_1(t)x_1(1 - x_2) - h_2(t)x_2(1 - x_3) \\
\dot{x}_3(t) &= h_2(t)x_2(1 - x_3) - h_3(t)x_3(1 - x_4) \\
&\vdots \\
\dot{x}_{n-1}(t) &= h_{n-2}(t)x_{n-2}(1 - x_{n-1}) - h_{n-1}(t)x_{n-1}(1 - x_n) \\
\dot{x}_n(t) &= h_{n-1}(t)x_{n-1}(1 - x_n) - \beta(t)x_n
\end{aligned} \tag{2.55}$$

This however ignores indirect dependencies of sites, which may lead to inaccuracies of the solutions obtained by this method.

One way to mitigate the error might be to approximate (2.51) instead of (2.50). The idea being, that in (2.51) the multiplication terms can be calculated with the help of (2.52) and (2.53). As such (2.51) remains unchanged. The terms $\langle \tau_{i-1}\tau_i \rangle$ and $\langle \tau_i \tau_{i+1} \rangle$ are calculated using an approximation of (2.52) and (2.53). Possible approximations are:

$$\begin{aligned}
\langle \tau_i \tau_{i+1}(\tau_{i+2} - 1)\rangle h_{i+1} &\approx h_{i+1}\langle \tau_i \rangle \langle \tau_{i+1} \rangle \langle \tau_{i+2} - 1 \rangle \\
&\approx h_{i+1}\langle \tau_i \tau_{i+1} \rangle \langle \tau_{i+2} - 1 \rangle \\
&\approx h_{i+1}\langle \tau_i \rangle \langle \tau_{i+1}(\tau_{i+2} - 1) \rangle \\
\text{for the first term.} & \\
\langle \tau_{i-1}(1 - \tau_i)\tau_{i+1}\rangle h_{i-1} &\approx h_{i-1}\langle \tau_{i-1} \rangle \langle (1 - \tau_i) \rangle \langle \tau_{i+1} \rangle \\
&\approx h_{i-1}\langle \tau_{i-1}(1 - \tau_i) \rangle \langle \tau_{i+1} \rangle \\
&\approx h_{i-1}\langle \tau_{i-1} \rangle \langle (1 - \tau_i)\tau_{i+1} \rangle \\
\text{for the second term.} &
\end{aligned} \tag{2.56}$$

The combination which resulted in the least error compared to the *master equation* during preliminary testing turned out to be

$$h_{i+1}\langle \tau_i \tau_{i+1} \rangle \langle \tau_{i+2} - 1 \rangle + h_{i-1}\langle \tau_{i-1} \rangle \langle (1 - \tau_i)\tau_{i+1} \rangle \tag{2.57}$$

The reason for this might be that only a single multiplicative expected value term is required for any time-step. One upside of this approach in contrast to (2.54) is that in addition to $\langle \tau_{i-1} \rangle$, $\langle \tau_i \rangle$and $\langle \tau_{i+1} \rangle$, the term $\langle \tau_{i+2} \rangle$ also effects the calculation of $\langle \tau_i \rangle$. It is therefore possible that this approach yields more accurate results.

Regardless of which approach is used, the error depends on the covariance of the state occupation probabilities. These covariances depend on the flow of particles between the

lattice sites as higher flow necessarily leads to higher correlation of the probability of occupancy. One immediate effect of this simplification, however, is the reduction of the number of ordinary differential equations from $N = 2^n$ to $N = n$ if (2.50) is used. And a reduction to $N = 2n$ if (2.52) is used. In either case computation is eased drastically. This describes the *mean field* method of calculating the probabilities $x_i(t)$ of *TASEP* systems. The mean field method is commonly used in statistical physics [13], neuroscience [14] and epidemiology [15]

2.3 Motivation of this Work

Differences in computational complexity

Given the difference in computational cost between the two approaches it becomes obvious why the *mean field model* has found widespread use instead of the *master equation* approach. This imbalance is not limited to the time difference in computing the solutions of these two models. A more difficult challenge to overcome when using the *master equation method* is the number of states that have to be taken into consideration, when solving a *TASEP* problem. Consider a system of a lattice consisting of 65 discrete sites. The *mean field method* then requires a system of 65 ordinary differential equations. Computing a solution of a system of this size is comparatively simple. The system of differential equations of the *master equation method*, however, grows exponentially with the number of states. This leads to the dimension of the *master equation* growing to $N = 2^{65}$. This has a few ramifications when trying to compute a solution. Firstly, the amount of memory needed to store a system of this size would exceed 30 exabytes, even if every value of the state-vector would only need one byte of space. The second problem is, that a 64-bit processor is not natively able to point to any address which takes more than 64 bits to describe. This would be necessary to allocate memory larger than 2^{64} bytes (16 Exabytes). While this is possible, it adds another layer of complexity which slows down the calculation and increases the computation time even further. Adding to this problem is the effort needed for finding the *master equation* to a given *TASEP* system. Generating the matrix which is used in computing a solution is not trivial as will be shown later and adds additional challenges for anyone implementing such a program. In contrast, the differential equations used for computing the mean field approximation, can be generated with less effort, as they, despite being nonlinear, only need to take the previous and following sites into account. This reduces complexity, as all equations will have the same structure with the difference being in the indices of the entries of the state vector.

The following thesis will show a computationally efficient method of computing the solutions to the *master equation* and provide an algorithm which can be used to generate its matrix.

Validation of the mean field method

As the *mean field method* is widely used one might assume the propriety of its solutions to *TASEP* systems. However, there is a lack of evidence that could either confirm or deny the accuracy the mean-field approach. This absence of any kind of mathematical validation is

concerning, as the simplified model is used in various commercial and research endeavours. This is understandable, given the previously mentioned challenges rendering any *master equation* based solution approach unusable in most cases. This thesis aims to examine the accuracy of the *mean field* method up to a lattice size of around 30 in time variant and time invariant conditions to be able to asses the validity of small scale approximations.

Chapter 3

Programs and Algorithms

3.1 Explicit Runge-Kutta Methods

Definition

For a more complete derivation and explanation of Runge-Kutta Methods see [16].

Let an initial value problem in $\mathbb{R}^n$, $n \in \mathbb{N}$ be specified as follows:

$$\frac{d}{dt}x(t) = f(t, x(t)), \quad x(t_0) = x_0 \tag{3.1}$$

Where $f : D \to \mathbb{R}^n$ is a continuous function and D is an open subset of $\mathbb{R} \times \mathbb{R}^n$.
Let the exact solution of this problem be described as $x(t; t_0, x_0)$ for $t \in [t_0, T]$. A set $\mathcal{T} = \{t_0, t_1, \cdots, t_N\}$ of time values with $t_0 < t_1 < \cdots < t_N = T$ is called a lattice of the interval $[t_0, T]$.
A function $\tilde{x} : \mathcal{T} \to \mathbb{R}^n$ is called a lattice function.
An iterative single step method is then given by the continuous mapping

$$\Phi : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n \tag{3.2}$$

with which a lattice function can be recursively defined for every lattice $\mathcal{T}$ and initial condition x_0 as follows:

$$\tilde{x}(t_0) = x_0, \quad \tilde{x}(t_{i+1}) = \Phi(t_i, \tilde{x}(t_i), h_i) \quad \text{for} \quad i = 0, 1, \cdots, N-1 \tag{3.3}$$

This allows for the general definition of the lattice function of a *s-step explicit Runge-Kutta-method* as follows:

$$\Phi(t, x, h) \;=\; x + h \sum_{i=1}^{s} b_i k_i \tag{3.4}$$

where

$$k_i \;=\; f\left(t + c_i h,\; x + h\sum_{j=1}^{i-1} a_{ij} k_j\right) \tag{3.5}$$

The coefficients of this methods can be expressed as

$$b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_s \end{pmatrix} \in \mathbb{R}^s, \quad c = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_s \end{pmatrix} \in \mathbb{R}^s, \quad \mathcal{A} = \begin{pmatrix} 0 & & & & \\ a_{21} & 0 & & & \\ a_{31} & a_{32} & 0 & & \\ \vdots & \vdots & \ddots & \ddots & \\ a_{s1} & \cdots & \cdots & a_{s,s-1} & 0 \end{pmatrix} \in \mathbb{R}^{s\times s} \tag{3.6}$$

which can be written in a so called *Butcher-tableau* as:

$$\begin{array}{c|cccccc} c_1 & & & & & & \\ c_2 & a_{21} & & & & & \\ c_3 & a_{31} & a_{32} & & & & \\ \vdots & \vdots & \vdots & \ddots & & & \\ c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & & \\ \hline & b_1 & b_2 & \cdots & b_{s-1} & b_s \end{array} \tag{3.7}$$

This methodology allows us to express the Euler ($s = 1$), Heun ($s = 2$) and RK(4) ($s = 4$) Methods as follows:

$$\begin{array}{c|c} 0 & \\ \hline & 1 \end{array} \qquad \begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad \begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \tag{3.8}$$

3.2 Accuracy and Speed trade-off in ODE Solvers

Prebuilt solvers

There exists a number of solvers for ODE systems that could be used to solve *master equations* of the *TASEP* model. Matlab for example has a range of prebuilt solvers available for use [17] which suit most smaller systems of differential equations. These mostly implement a form of an adaptive Runge-Kutta method, which allows for variable step sizes and increased accuracy. One disadvantage of using a mathematical software environment like Matlab is their lack of speed compared to a direct implementation of a similar solving method.

Adaptive step size solvers

The size of the matrices describing *TASEP master equations* grows rapidly with lattice length. This leads to the fact that minor time savings in one computational step have a considerable cumulative impact on the total time taken to compute solutions. Because of this the main requirement for any solving method is being able to produce accurate results while keeping computational overhead as low as possible. This leads to the decision not to implement adaptive step size methods, as the simpler, fixed step size, RK(4)-method was able to produce highly accurate solutions. The fixed step size however leads to possible overhead in areas, where trajectories remain stationary.

3.3 Chosen Metrics for Evaluation

This thesis aims to evaluate *mean field* approximations on their accuracy. This and the undeniable difference in speed due to the difference in dimension of *mean field* and *master equation methods* lead to the main focus of evaluation lying on the accuracy of the solvers. The developed algorithms are compared with a known accurate solution, which was computed using Octave. Methods which were deemed accurate enough were then compared against each other in speed and usability. Finally, algorithms that are easily parallelised were preferred, as the main computational action the program performs is matrix vector multiplication of the state vectors x with the system matrix A. Matrix vector multiplications are comparably easy to parallelise, which increases the performance of the program regarding time taken to compute a solution.

3.4 Quality Assessment of Chosen Method

Keeping all of the above in mind, the RK(4)-method was chosen as the main algorithm. The reasons for this decision will be explained below.

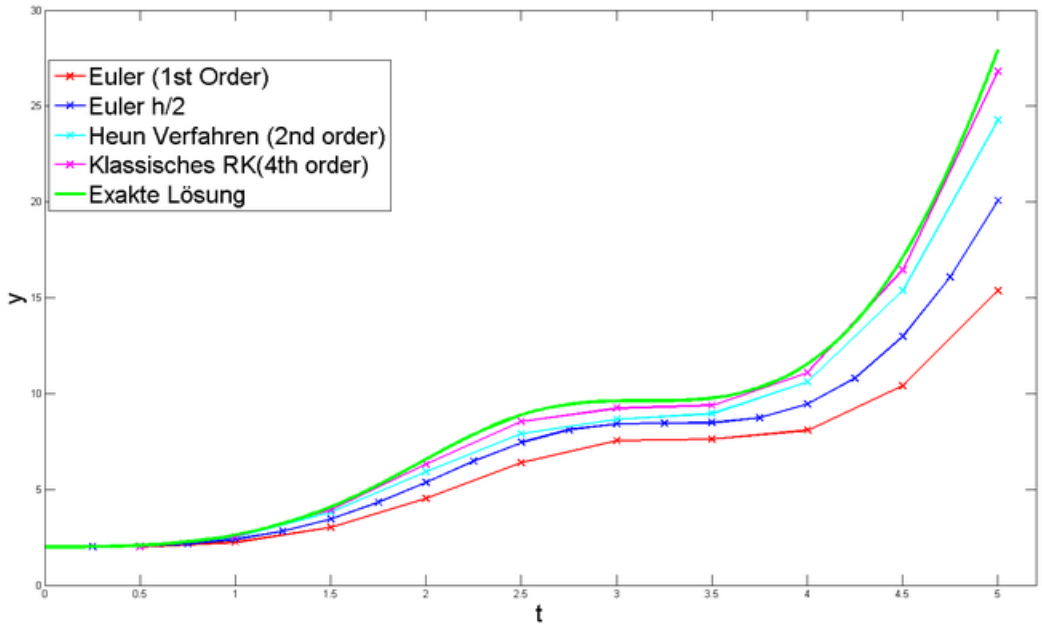
Simple stages

Every Runge-Kutta-method uses a varying number of stages to compute solutions to systems of ordinary differential equations. The RK(4)-method has one coefficient per stage which means that the number of floating point operations per time step is as low as RK methods permit. While this increases the error compared to a more complex method, it enables the algorithm to run at finer time steps without a considerable decrease in performance. This in turn decreases the error as well. Another upside of the simplistic make-up of the Butcher tableau of the RK(4)-method is the ease of implementation in code, especially considering that all programs were written in C which lacks a lot of higher level functionality that newer languages like python provide.

Accuracy

Compared to other methods for solving ODE-systems, mainly the Euler- and Heun-methods, the classic Runge-Kutta algorithm is more accurate, as is demonstrated below.

Figure 3.1: Comparison of ODE solving methods



Picture and corresponding R-code can be found at: https://commons.wikimedia.org/wiki/File:Runge-kutta.svg

This is due to the Runge-Kutta(4)-method having a global error $O(h^4)$, whereas Euler has $O(h^1)$ and Heun $O(h^3)$. As such the RK(4) Method offers a good compromise between speed, accuracy and ease of implementation, which made it the best option for this project.

Chapter 4

Comparison

4.1 Trajectory Behaviour

4.1.1 Time-invariant environment

Overview

In this section the algorithms for solving the master equation and for solving the mean field model, as well as the extension of the mean field model are compared solving time-invariant systems. This means that all flow rates are static and the system will converge to a steady state. The following solutions were obtained using the developed algorithms.

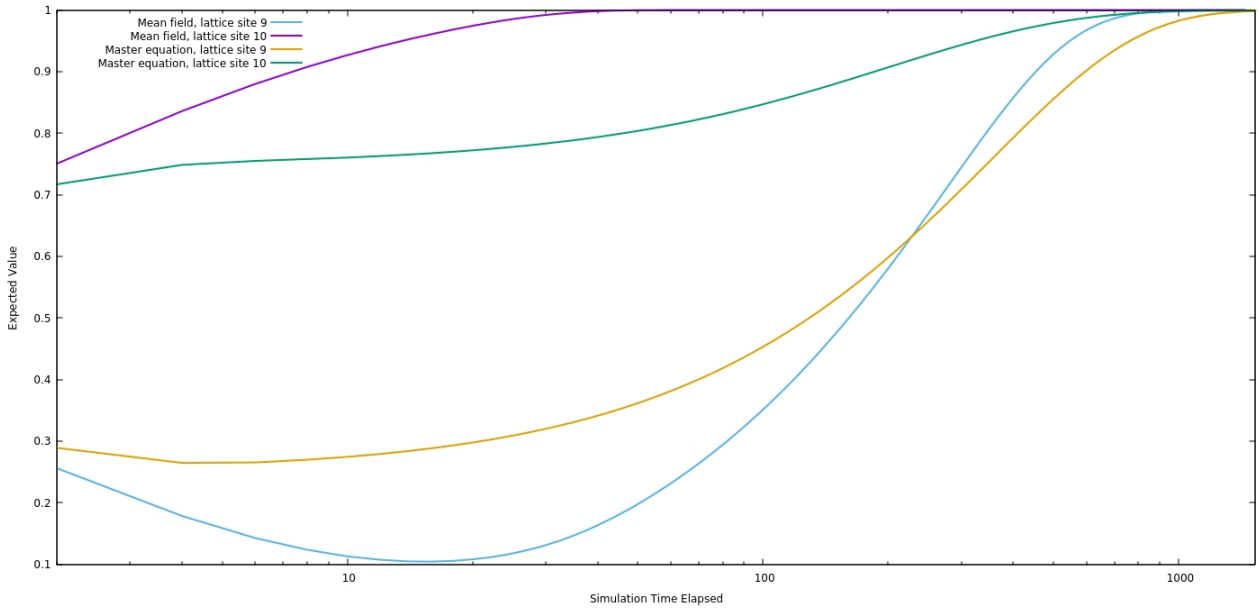
Differences between the non extended mean field approximation and the master equation

We have calculated the trajectories of the expected values of the occupancy of the sites over time for suitable example lattices with given flow rates. These lattices have been chosen to analyse the differences of the models. The y-values represent the expected value of occupancy calculated by the different algorithms. The trajectories calculated by the *non extended mean field model* tend to overshoot the correct solution in cases where the flow rate of one site is significantly higher or lower than the preceding or following site. Take for example a lattice of length 10, where the flow rates are as follows

$$\alpha = h_1 = 0.99 \quad h_2 = h_3 = \cdots = h_8 = 0.01 \quad h_9 = 0,99 \quad \beta = 0.0 \tag{4.1}$$

Let the above model be called Model 1 (M1). This flow leads to a gradual increase in the probability of occupancy of every site over time, as particles cannot leave the last site. The *non extended mean field model* is lacking in accuracy in this case, as is shown in the figure below (Fig 4.1) that compares the result of the non extended model with that of the master equation for the lattice sites 9 and 10. While the mean field model solutions converge to the correct steady state value, the temporal evolution of the trajectories before convergence is incorrect.

Figure 4.1: M1, Sites 9 and 10, Mean Field and Master Equation

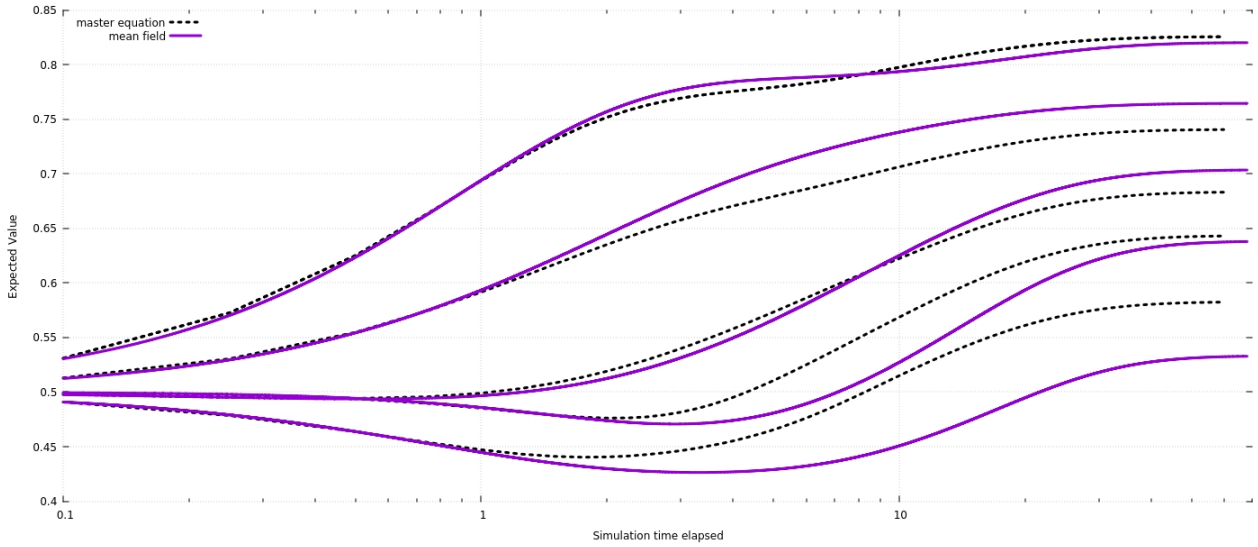


Another, perhaps more serious problem is the trajectory inaccuracy (both in the temporal evolution as well as the convergence values) of the *mean field model* when simulating solutions to lattices with random flow rates:

$$\begin{aligned} \alpha &= 0.840188 \\ h_1 &= 0.394383 \\ h_2 &= 0.783099 \\ h_3 &= 0.798440 \\ h_4 &= 0.911647 \\ \beta &= 0.197551 \end{aligned} \tag{4.2}$$

Let the above model be called Model 2 (M2). These were obtained using the C pseudo-random number generator "random()". This example has been chosen to show differences in lattices where the flow rates are different from one another, but are all the same order of magnitude. The resulting trajectories for all lattice sites can be seen in the figure below (fig. 4.2). As is clearly visible, the *mean field model* fails to accurately model the trajectories. Note that this is not because of low temporal resolution of the algorithms as the timesteps for the *RK(4)* algorithm used to approximate the *master equation* solution are $\Delta t = 0.001$ and those for the same *RK(4)* algorithm used to approximate the solution to the *mean field model* are $\Delta t = 0.00001$. These values were determined during the development of the programs. At these points any further increase of the temporal resolution only increased the time it took to complete a simulation without increasing the accuracy.

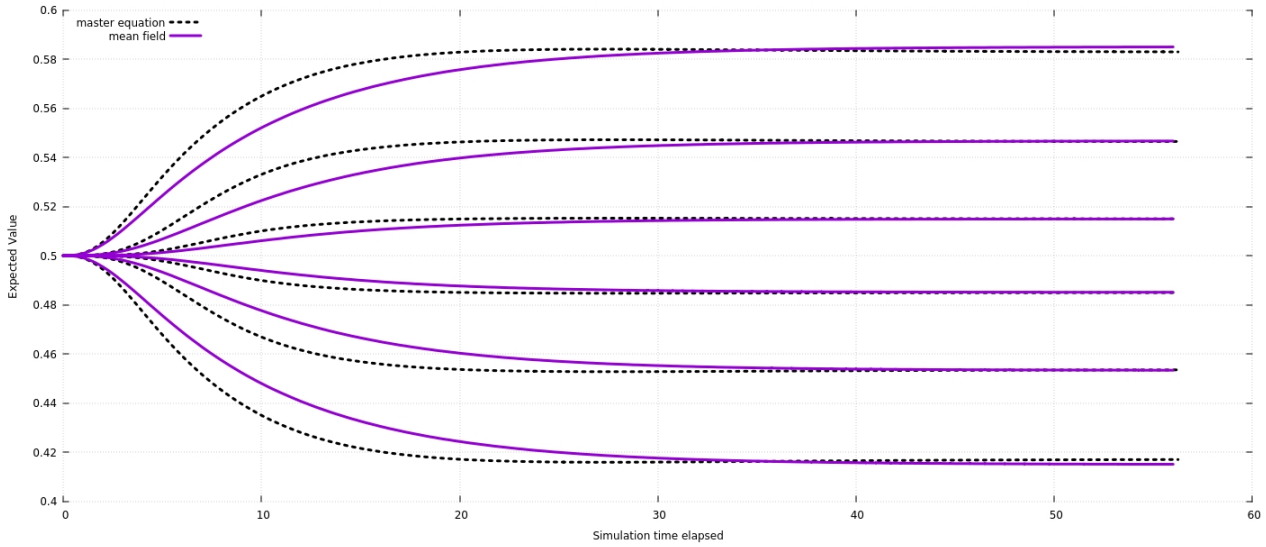
Figure 4.2: M2, Sites 1-5, Mean Field and Master Equation



The *mean field model* produces more suitable results for the expected values of the middle sites of a lattice with equal flow rates. The trajectories diverge from the correct solution at the beginning of the solution. This is because the *non extended mean field model* simulates a response of the trajectories that is slow in comparison to the correct result. This results in a minor error, that corrects itself, when the trajectories converge to their steady states. In the following figure (fig 4.3) a lattice with flow rates of 0.5 was simulated and the values for the sites 3 to 8 are displayed. The following lattice is configuration is called Model 3 (M3):

$$\alpha(t) = h_1(t) = \cdots = h_9 = \beta = 0.5 \tag{4.3}$$

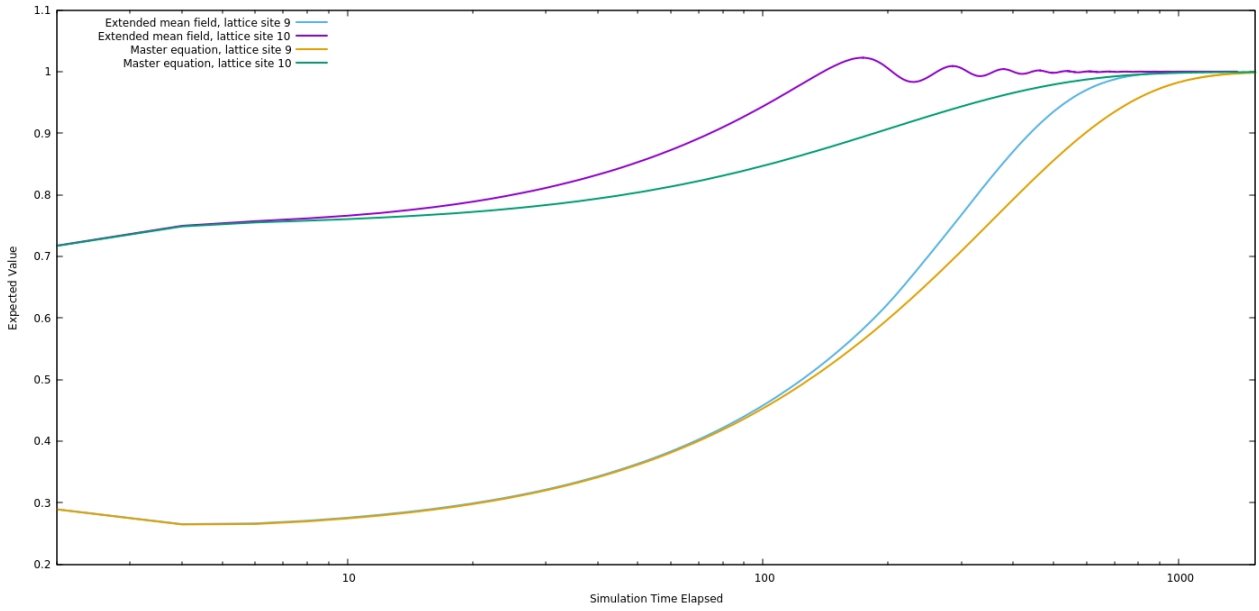
Figure 4.3: M3, Sites 3-8, Mean Field and Master Equation



Differences between the extended mean field approximation and the master equation

Much like the non extended version, the *extended mean field model* has some specific problems that are explored in this section. In the *extended mean field approximation* significantly different results are found for scenario (4.1). The results are more accurate in the first half of the simulation, but, as the trajectories converge, the *extended mean field model* overshoots the steady state convergence value 1.0 and oscillates around it (fig. 4.4). This behaviour and the increased accuracy in the first half of the simulation can be explained in a similar way. When the expected values for the occupation of lattice sites rise in the *master equation* the *non extended model* over values those changes, resulting in a premature rise of the trajectories. This is due to the limited scope of the model, where the expected value of any lattice site is only directly dependent on the expected values of the directly neighbouring sites. The extension increases that scope and the simulation is more responsive to changes in the expected values. This quicker response, however, is contributing to the error in cases where the model reacts too strongly to the high expected values of the following sites. The figure 4.4 compares the result of the extended model with that of the master equation for M1 and the lattice sites 9 and 10.

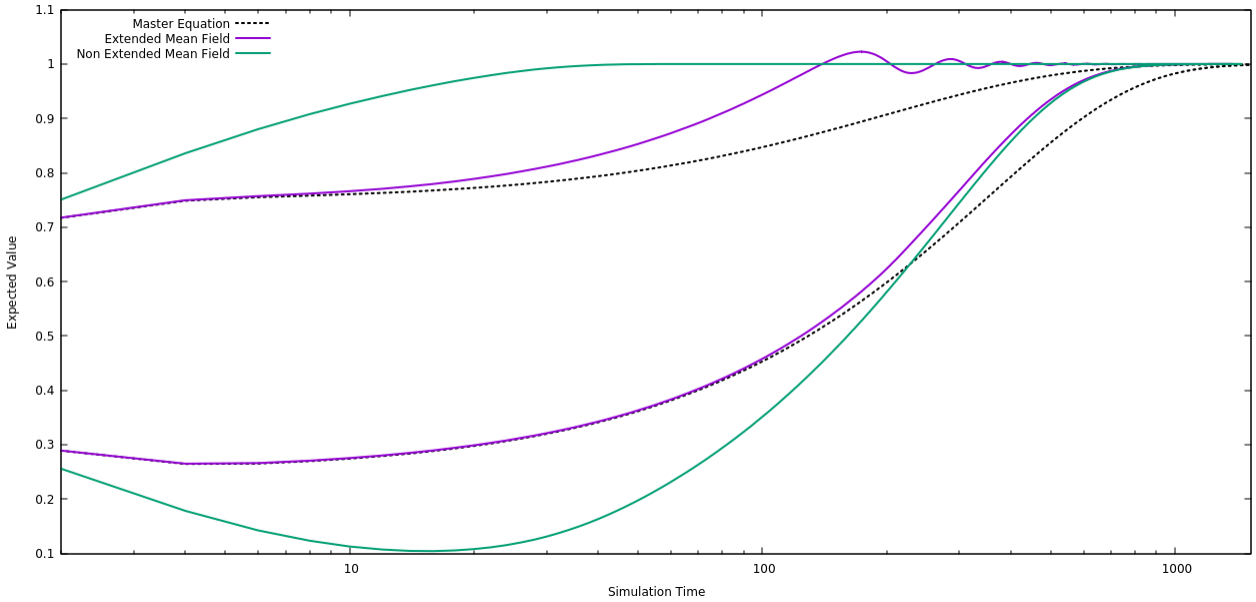
Figure 4.4: M1, Sites 9 and 10, Extended Mean Field and Master Equation



The trajectory for site 9 follows the correct solution very accurately, only slightly differing towards the end. However, when looking at the calculated trajectory for site 10 the shortcomings of the extended model become visible. The *extended mean field model* overshoots the correct solution and the steady state value 1.0 as well.

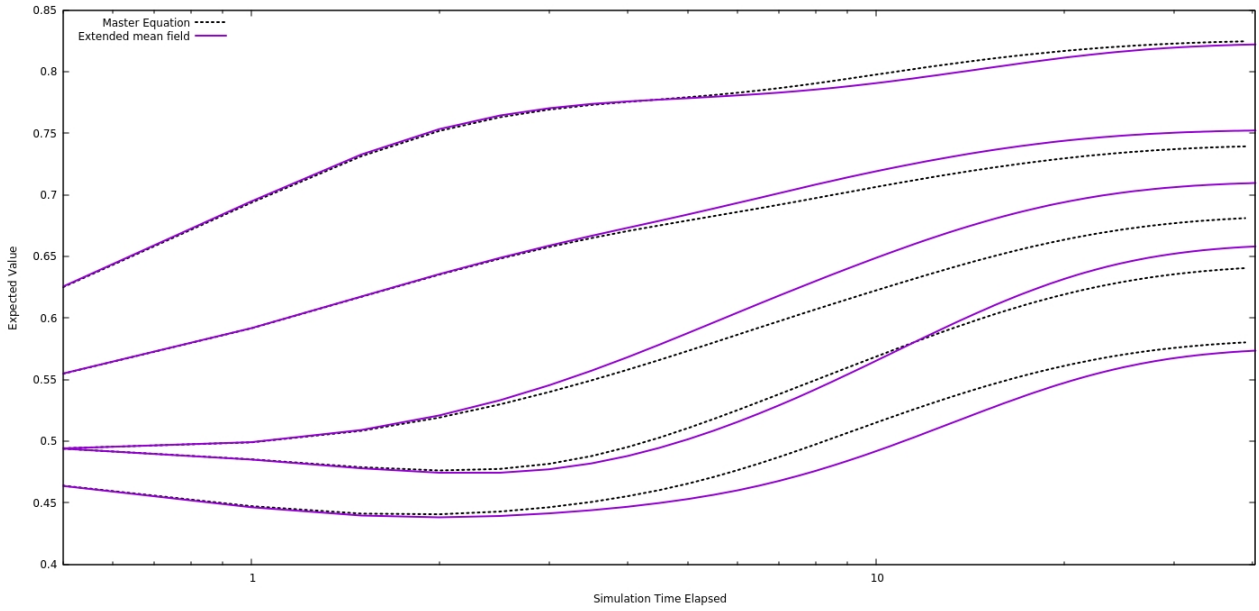
Comparing all methods in figure 4.5 shows the very different trajectories of the three approaches. The *extended mean field model* is much more accurate than the *non extended model* in the beginning. But, the non extended version does not oscillate towards the end of the simulation.

Figure 4.5: M1, Sites 9 and 10, Extended, Non Extended Mean Field and Master Equation



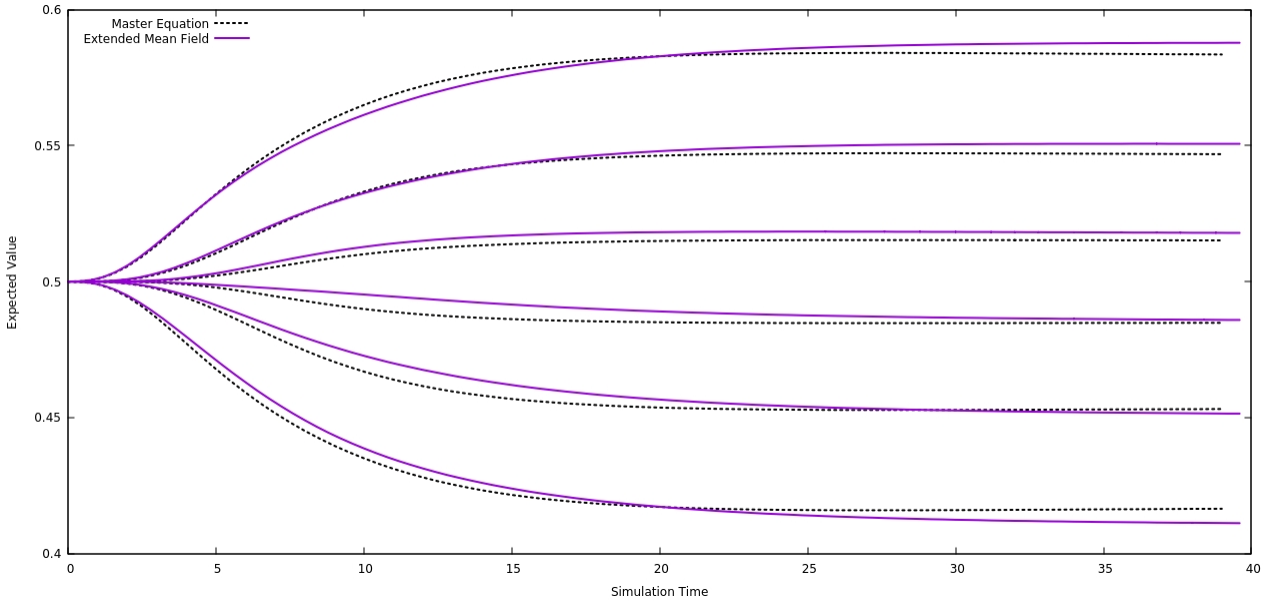
Looking at a lattice with hop rates as described in (4.2), one can see further improvements of the extended model over the non extended model (fig. 4.6). The *extended mean field model* produces trajectories that follow the solutions of the *master equation* more precisely than the *non extended model*. An error still exists in the extended approximation that results in imprecise steady states and a deviation from the correct solutions. The reason for this might again be the limited scope of the extended model compared to the *master equation*. Another reason might be that the results of the *extended mean field model* are more strongly impacted by high values of expected values of occupancy in comparison to the *master equation*. This impact comes from the way the mean field model was expanded. In this case the extended model takes one subsequent lattice site into account, as described in (2.57). This results in an imbalance of the number of preceding and following sites that are considered during the computation of the temporal evolution of the solutions. The obtained solutions are, however, more accurate in comparison to the *non extended mean field model*

Figure 4.6: M2, Sites 1-5, Extended Mean Field and Master Equation



Simulating a lattice with equal flow rates of 0.5 (Model 3), the *extended mean field model* produces similar results to the *non extended mean field model*. The trajectories are more accurate in the first half of the simulation in comparison to the non extended model. The final steady states, however, are less accurate. Below, the trajectories of sites 3 through 8 of a lattice are shown. In the figure below (fig. 4.7) Model 3 was simulated.

Figure 4.7: M3, Sites 3-8, Extended Mean Field and Master Equation



4.1.2 Time-variant environments

Overview

In this section the three algorithms are compared solving time-variant systems. As shown before in [5], under these circumstances, the system will converge to a periodic solution with the same period as the rates.

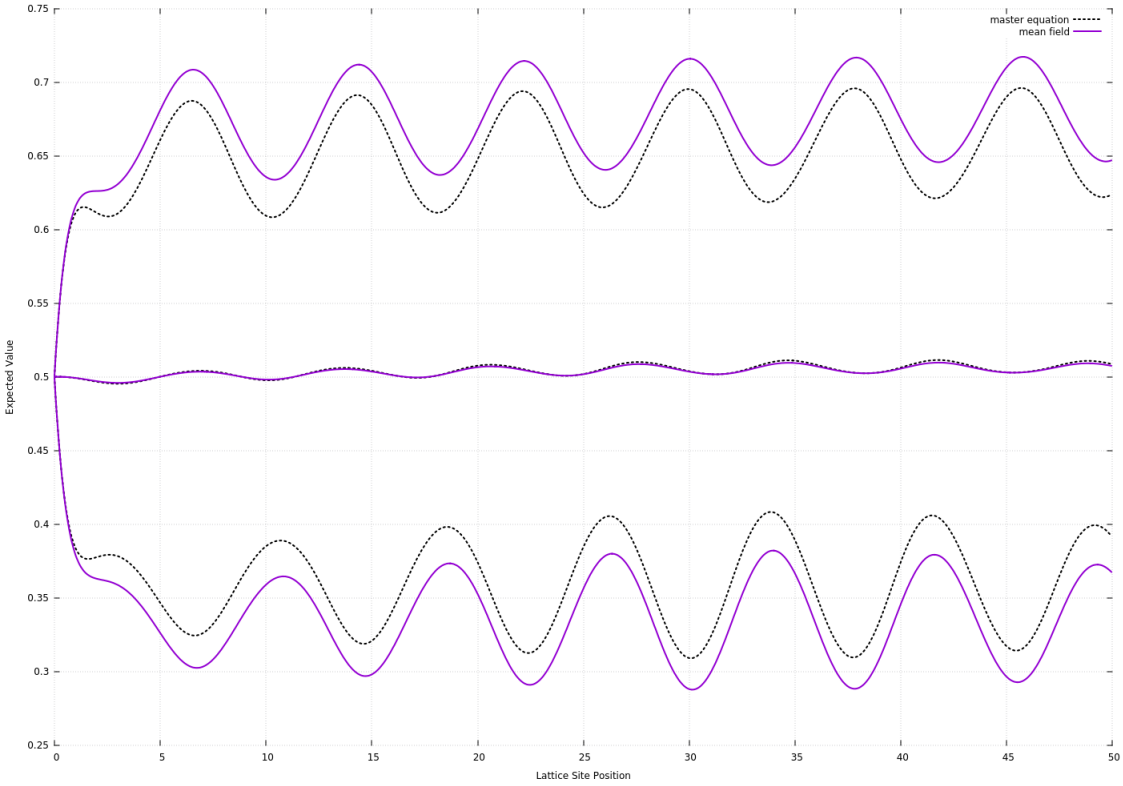
Differences between the non extended mean field approximation and the master equation

Consider the following lattice of length 3 with time dependent jump rates Model 4 (M4):

$$\begin{aligned} \alpha(t) &= 0.05 * \sin(t * 0.03) + 1 \\ h_1(t) = h_2(t) &= 0.19 * \sin(t * 0.8) + 1 \\ \beta(t) &= 0.05 * \sin(t * 0.9) + 1 \end{aligned} \tag{4.4}$$

The next figure (fig. 4.8) compares the results of the *mean field model* with that of the *master equation* for all three lattice sites. The main difference of the solutions is that the *mean field model* tends to offset the expected values of the occupancy of the lattice sites. The trajectory that oscillates around 0.5 is accurate at the beginning of the simulation but becomes less accurate with time.

Figure 4.8: M4, Sites 1-3, non extended mean field and master equation

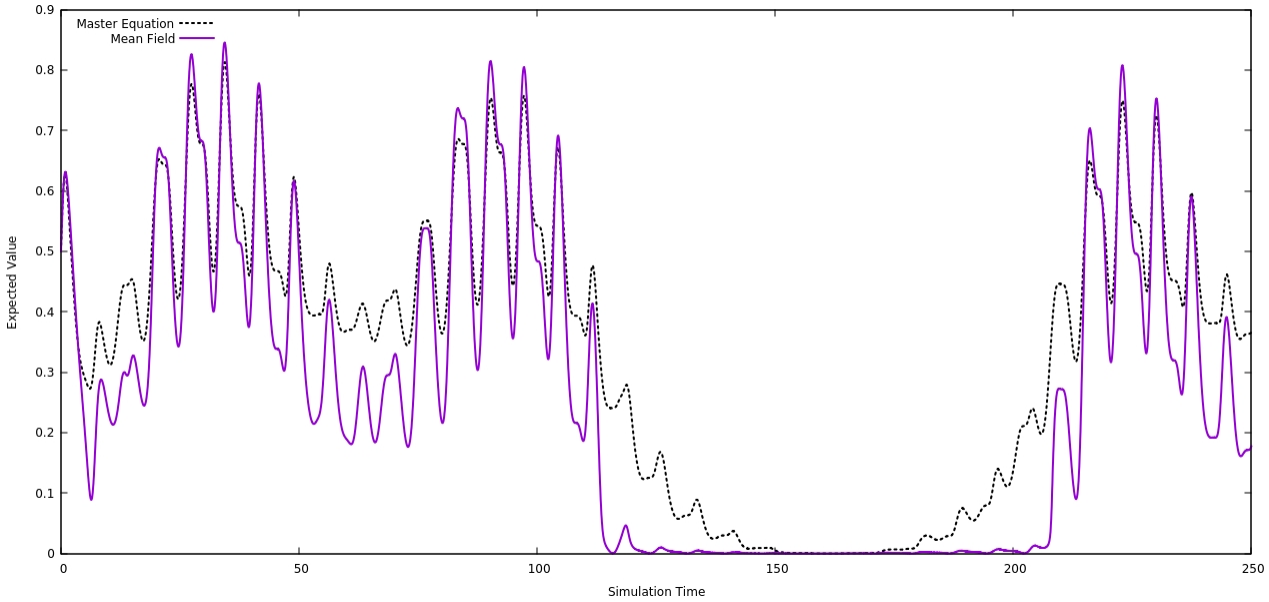


In this case the values obtained by the two methods are roughly similar, the lack in accuracy of the *mean field model* becomes more clear if we simulate a lattice of length 4 with the following flow rate functions:

$$\begin{aligned} \alpha(t) &= \sin(0.3 * t) + 1 \\ h_1(t) &= \sin(0.8 * t) + 1 \\ h_2(t) &= t \\ h_3(t) &= \sin(0.8 * t) + 1 \\ \beta(t) &= \sin(0.9 * t) + 1 \end{aligned} \tag{4.5}$$

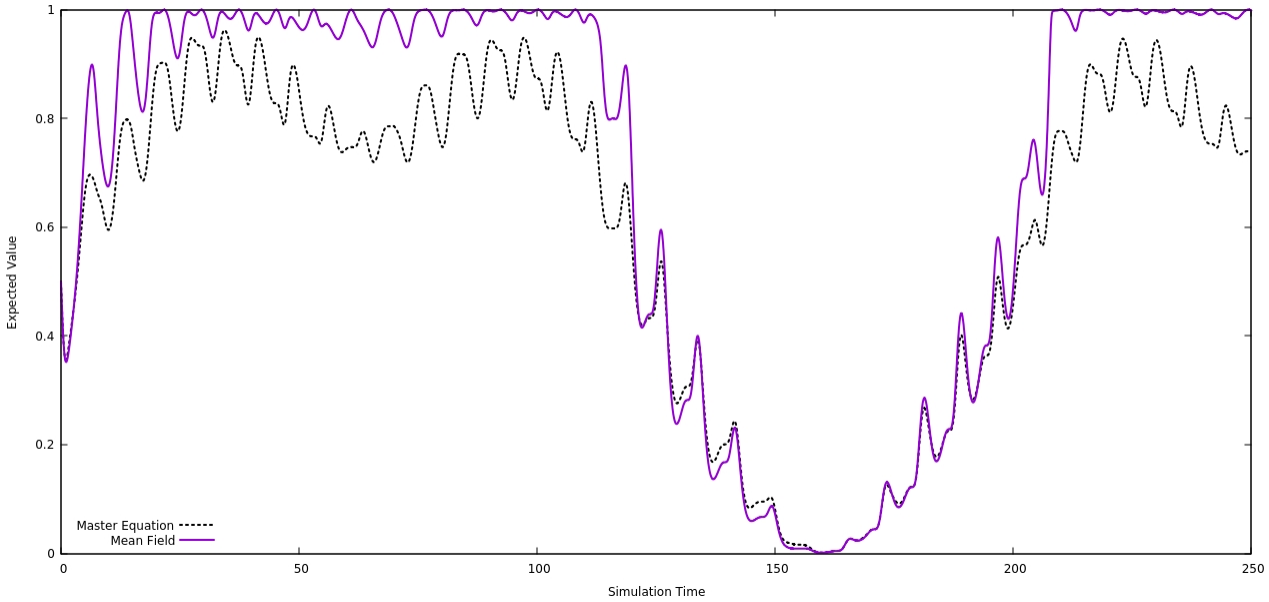
Let the above configuration be called Model 5 (M5). This gives a steadily rising middle flow rate, the immediate effect of which is an increase of the probability that the next site is occupied and the previous site is empty. This leads to the trajectory shown in figure 4.9 for the second site of the lattice:

Figure 4.9: M5, Site 2, Non Extended Mean Field and Master Equation



Here, the solution calculated using the *mean field model* starts out accurately, but, as time moves on, starts to differ by roughly 0.2 at certain points during the simulation and since this solution is periodic and the flow rate $h_2(t)$ is rising with time, the *mean field* trajectory will become more and more inaccurate. This behaviour is due to the fact that a particle which enters site 2 will almost immediately enter site 3, due to the high flow from 2 to 3. The effect of this is that the behaviour of the lattice is almost as if the second site does not exist and particles flow from site 1 directly into site 3. The *mean field model* can not account for such a case. The same error can be seen on the following site. Since the flow rate is high, the expected value of occupancy of the 2nd site is very low and the value for the 3rd site is very high. The following figure shows the trajectories of the 3rd site.

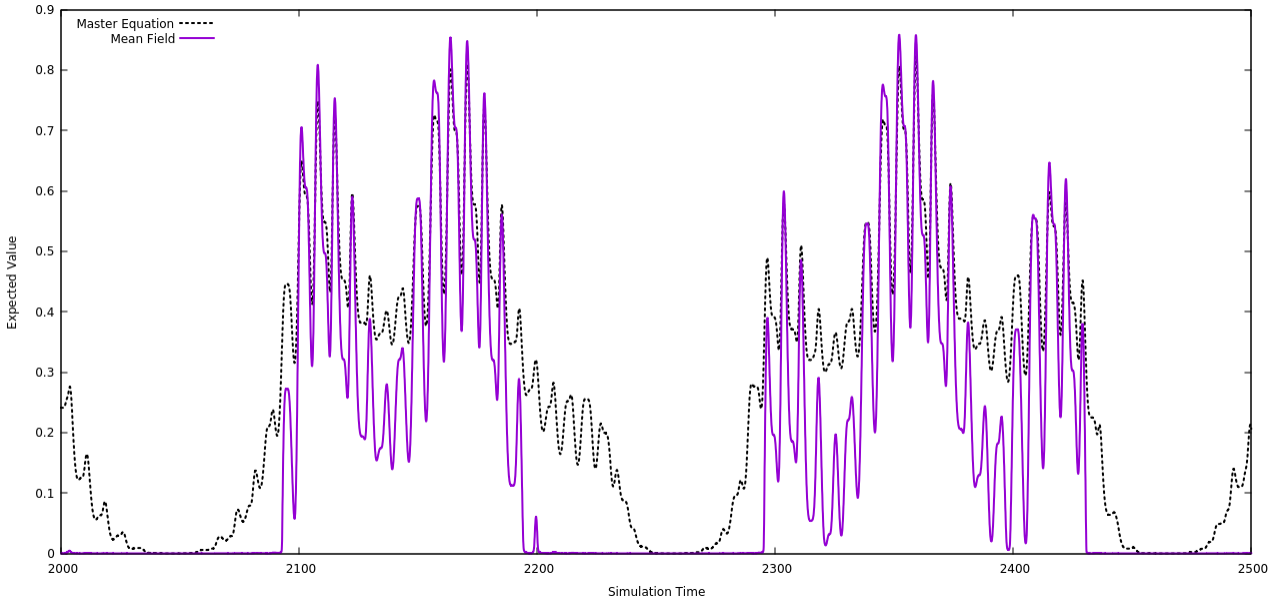
Figure 4.10: M5, Site 3, Non Extended Mean Field and Master Equation



The *mean field* approximation gives increased values for the 3rd site during the periods in which the expected values of occupancy are high.

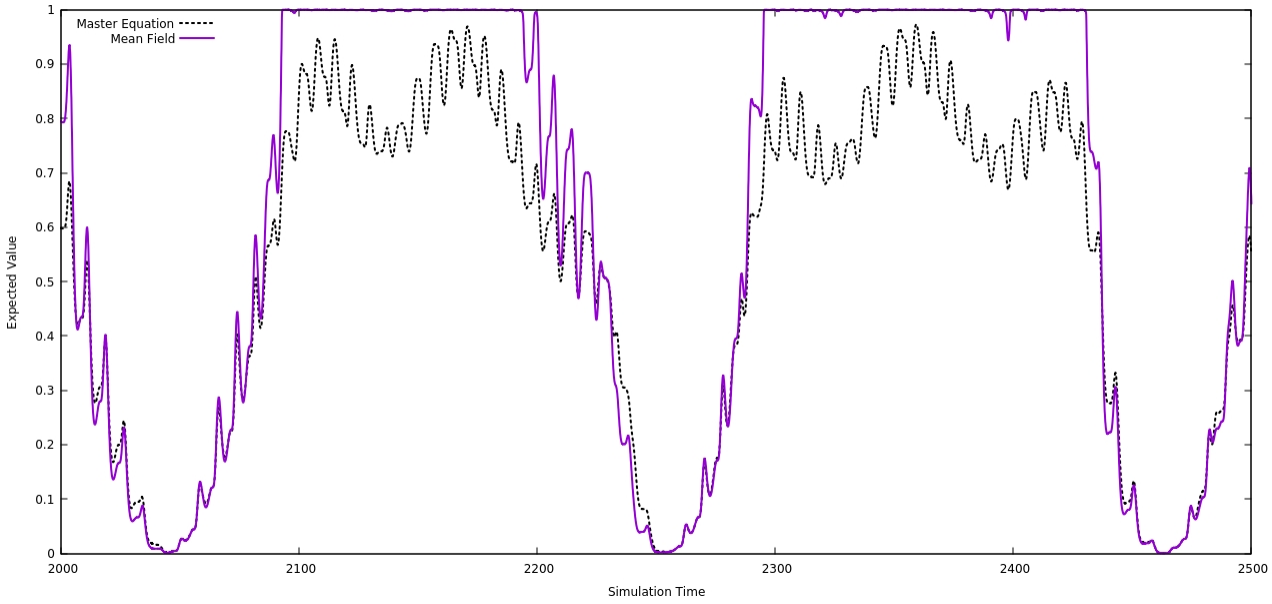
As can be seen in the figure below (fig. 4.11), which displays the trajectories of the second site, similar to the figure above but with $t \in [2000, 2500]$. This shows that, while the *master equation* trajectory remains stable, the *non extended mean field models* remains at 0 for extended periods of time. After which the response is very sudden and extreme. The trajectory of the *non extended mean field model* over- and undershoots the trajectories of the *master equation*. The *non extended mean field models* inability to take the expected values of occupancy of non-neighbouring sites directly into account facilitates these errors.

Figure 4.11: M5, Site 2, Non Extended Mean Field and Master Equation



A similar error appears in the trajectories of the third site. Here, the error lies in increased trajectory-values, because the *non extended mean field model* interprets the high flow rate of the previous site as a steady stream of particles. This error stems from the mean field models limited scope. A consequence of this limitation is the inaccurate trajectory for the third site.

Figure 4.12: M5, Site 3, Non Extended Mean Field and Master Equation

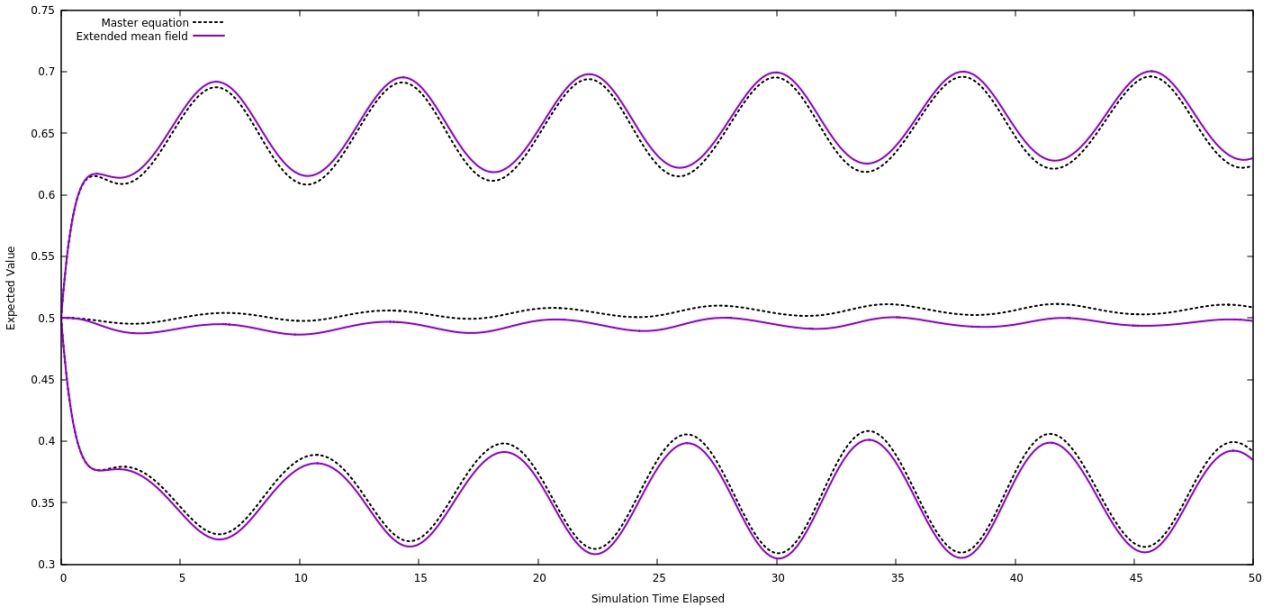


The *non extended mean fields* trajectory is approximately 1.0 for large amounts of time, before drastically shifting to follow the trajectory of the *master equation*. The solution is accurate within a small error, if the expected value of occupancy of the previous site is low.

Differences between the extended mean field approximation and the master equation

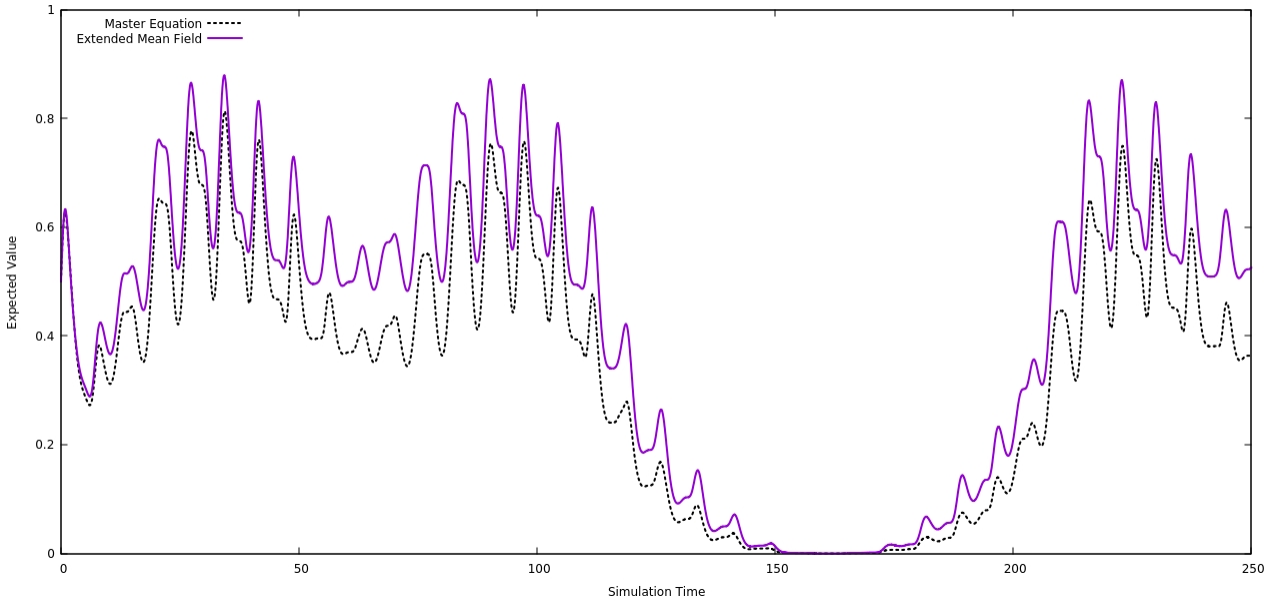
The *extended mean field model* is compared to the *master equation*, simulating the same system as (4.4) and Model 4. The figure below (fig. 4.13) shows the increased accuracy of the extended model with the exception of the middle site. The trajectories of the edge cases, which oscillate more strongly than the middle one, are more accurate in comparison to the non extended model. The trajectory of the expected value for the occupation of the middle site follows the shape of the *master equation* trajectory well. However, the solution is offset by approx 0.01 almost immediately after the beginning of the simulation.

Figure 4.13: M4, Sites 1-3, extended mean field and master equation



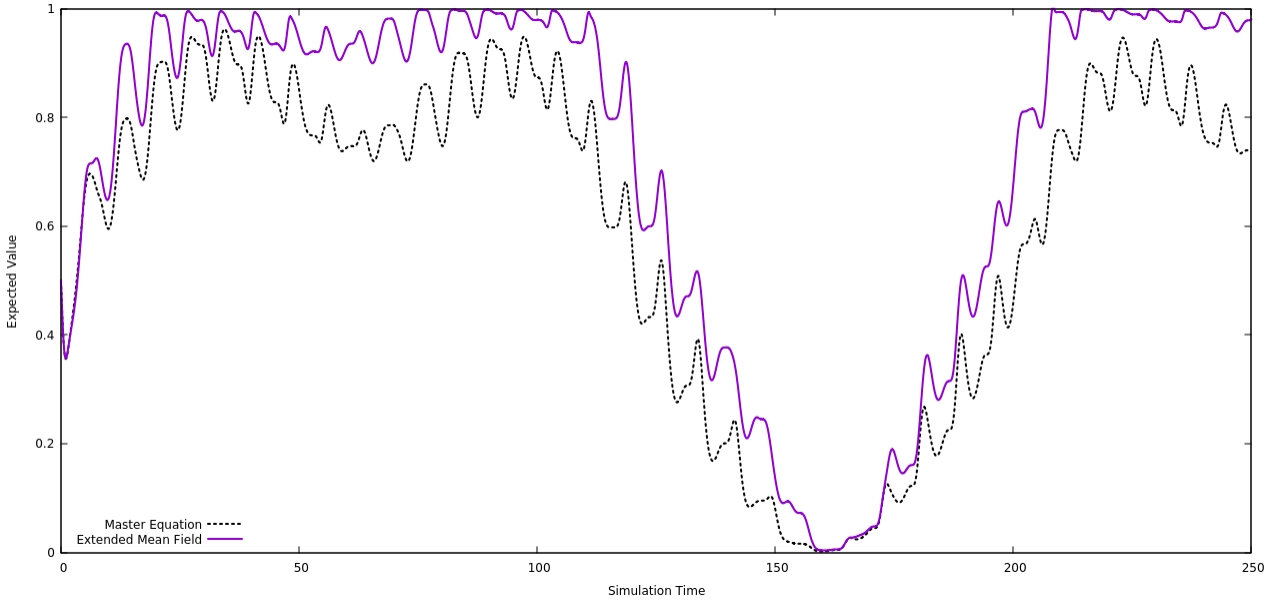
In the following figure (fig. 4.14) the trajectories of the two algorithms are again compared using a steadily rising middle flow rate and periodic flow rates for the other lattice sites as in (4.5) and Model 5. For the second site, the *extended mean field model* shows the same errors as before, where the shape of the trajectories is similar to the trajectories of the *master equation*, but the values are offset by a relatively steady margin. In the case of (4.5) the extended model is off by roughly 0.1 to 0.2 during most of the beginning of the simulation. An immediate effect of this is that the extended model is sometimes less accurate than the non extended version.

Figure 4.14: M5, Site 2, Extended Mean Field and Master Equation



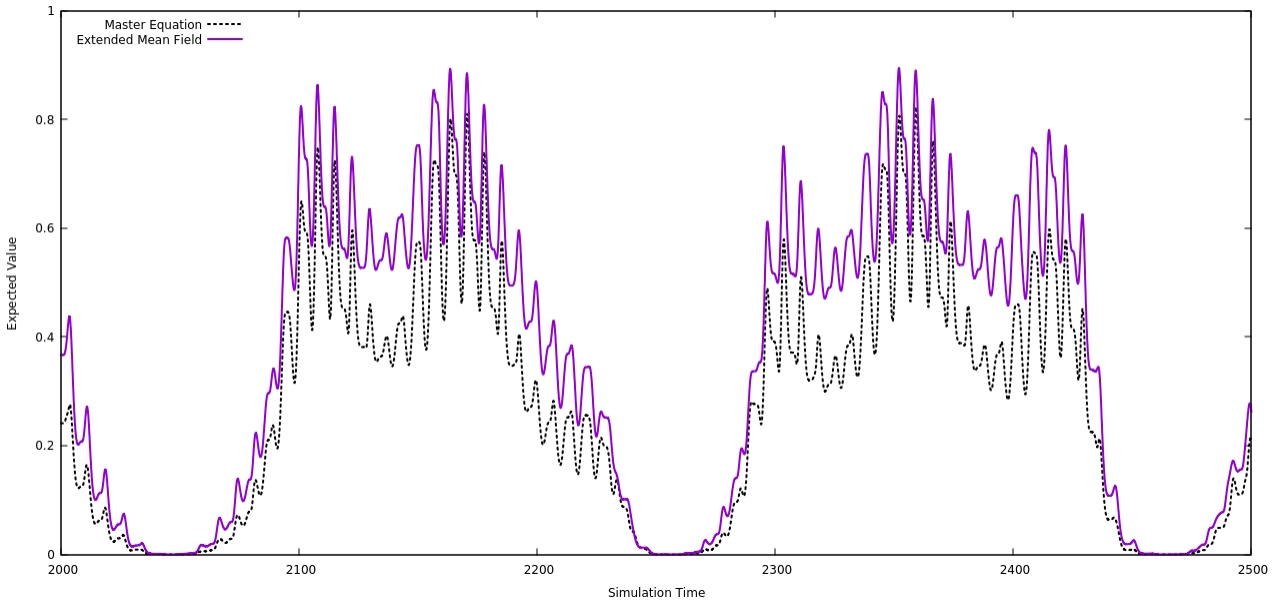
A similar picture emerges for the trajectories of site 3. Again the trajectory of the extension follows the shape of the *master equation* trajectory more accurately, yet the error is often higher than the error of the non extended model, due to the relatively steady offset (fig. 4.15).

Figure 4.15: M5, Site 3, Extended Mean Field and Master Equation



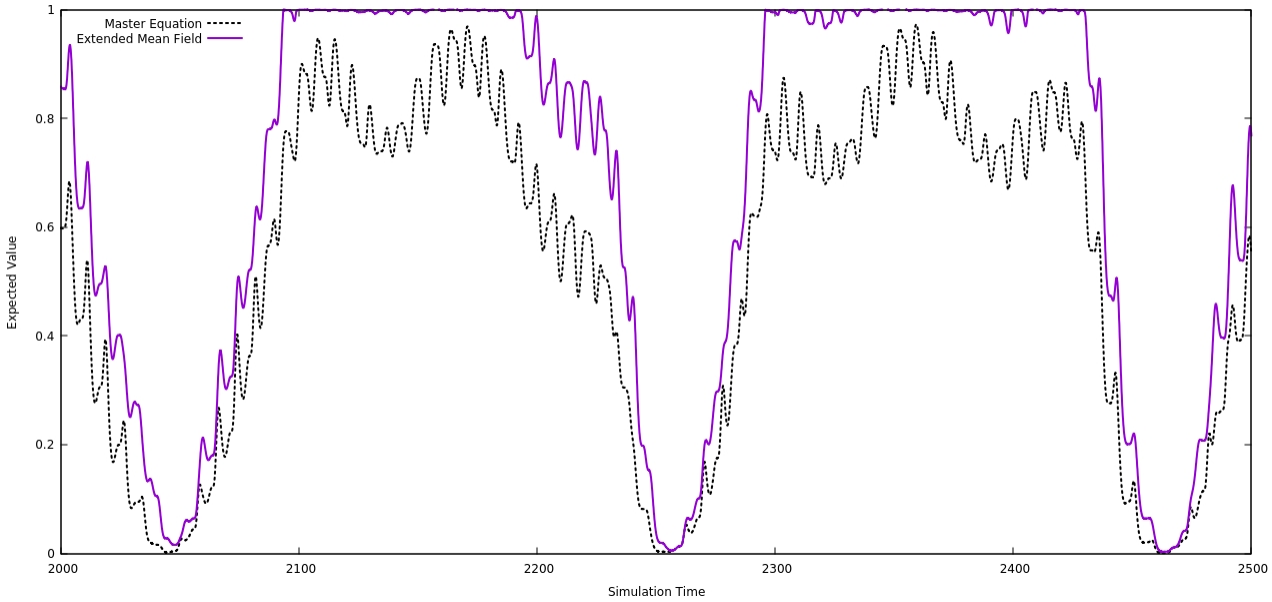
An upside of the *extended mean field model* is that the trajectories remain stable for longer. As such, the error compared to the non extended model is lower, if the simulation is run for longer. The trajectory for the expected value for site 2 as calculated by the extended model does not exhibit the same behaviour as the trajectory of the non extended model (fig. 4.16). Specifically, the extension does not get "stuck" at the values 0 or 1 and follows the *master equations* trajectory more precisely. The reason for the difference of the models is probably the increased "scope" of the extended model.

Figure 4.16: M5, Site 2, Extended Mean Field and Master Equation



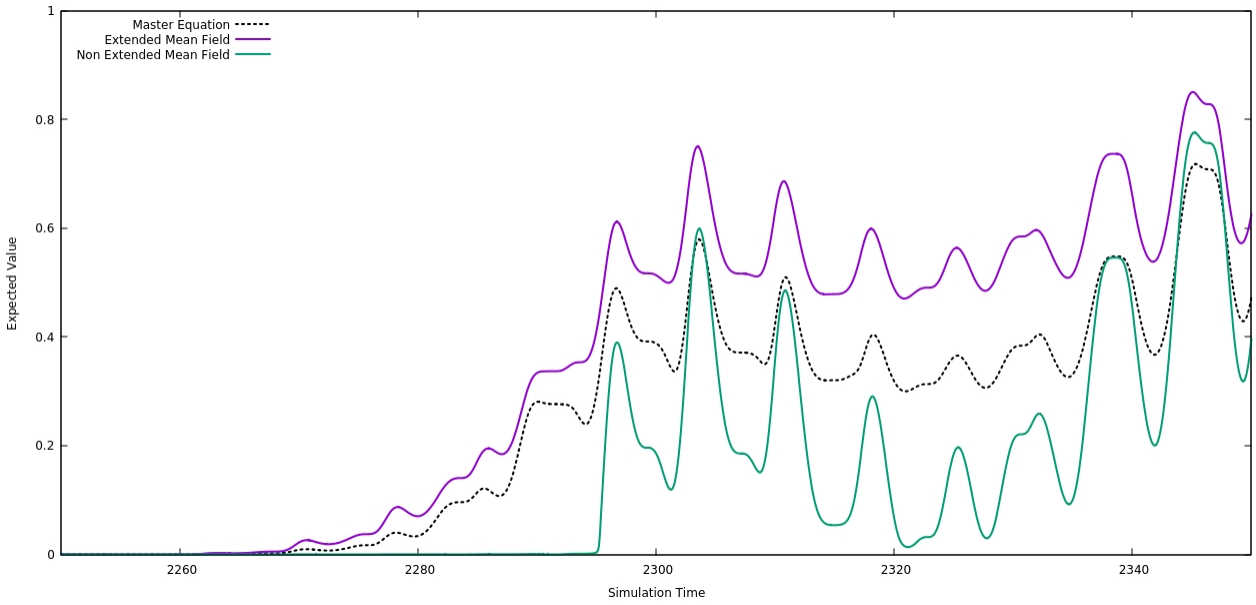
Looking at the trajectories for the 3rd site, however, the extended model again falls short of the accuracy of the non extended model (fig. 4.17). A reason for this might be that, while the *extended mean field model* directly computes the expected values of any site i with the values of sites $i-1$ through $i+2$, the values of site $i-2$ are not directly influencing the calculation. This explains the relatively small error of the trajectories of site 2. The trajectories of site 3, however, suffer from both the inaccurate response to changes in fluxes of the non extended model as well as the steady error margin of the extension. Cumulatively this leads to a larger error than that of the non extended model.

Figure 4.17: M5, Site 3, Extended Mean Field and Master Equation



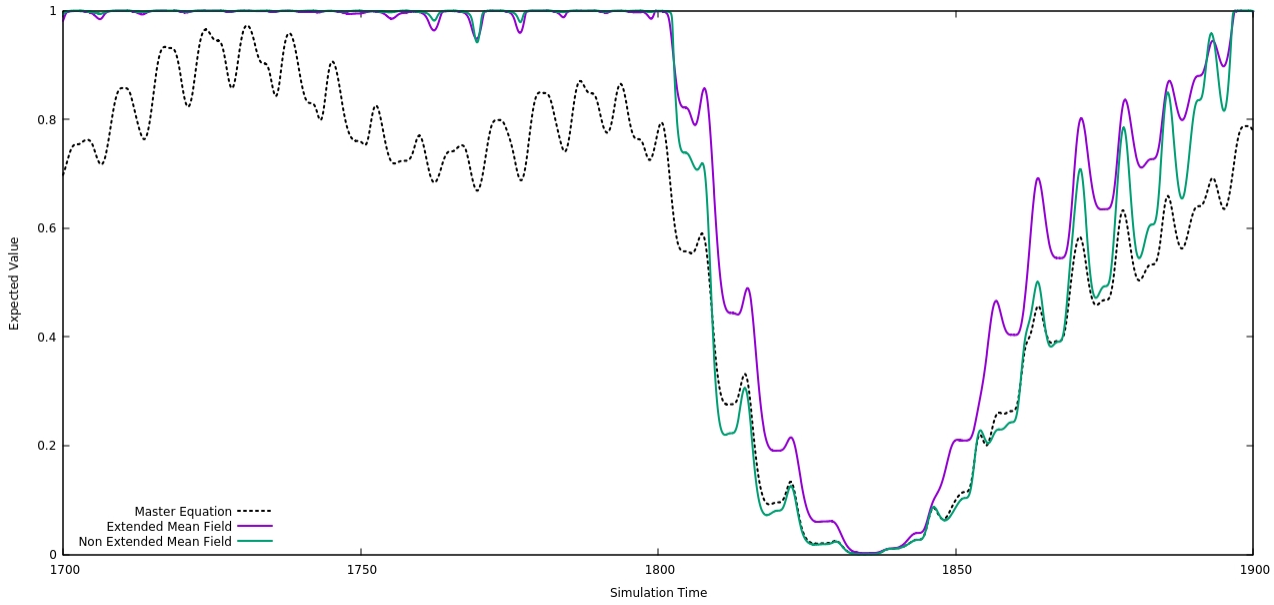
Comparing the extended and the non extended mean field algorithm at selected times in the simulation further illustrates the above mentioned flaws of these approximations. For the second site, the following figure (fig. 4.18) shows the delay in the response of the *non extended mean field model* which leads to a far too drastic change of the trajectory values. Also visible is the "offset" of the trajectories simulated by the *extended mean field model*. This "offset" seems to be typical for the extension of the model. One notable aspect here is that it remains almost constant for large parts of the simulation.

Figure 4.18: M5, Site 2, Extended Mean Field, Non Extended Mean Field and Master Equation



A similar picture emerges for the expected value trajectories for the third site. Here, the extended model displays the same constant error as well as a less accurate shape (fig. 4.19). The decrease in accuracy might be due to the extended model not including more than one previous site in the calculation of the expected values. As such the two algorithms have a similar delay but the *non extended mean field model* changes the gradient of the trajectory more drastically. This quick response leads to an accurate trajectory during the period where the expected value is close to 0.

Figure 4.19: M5, Site 3, Extended Mean Field, Non Extended Mean Field and Master Equation



4.2 Steady States

Overview

In the following section, the steady states as simulated by the three methods, will be compared. Because the system will only ever reach a steady state, if the flow rates are constant, we will only consider the time-invariant case. To identify the time at which a system has reached a steady state, the algorithm will check if the absolute change of all components of the trajectory remain below a given threshold for a set period of time. If this is the case, the algorithm will halt and the approximative solution for the steady states will be returned.

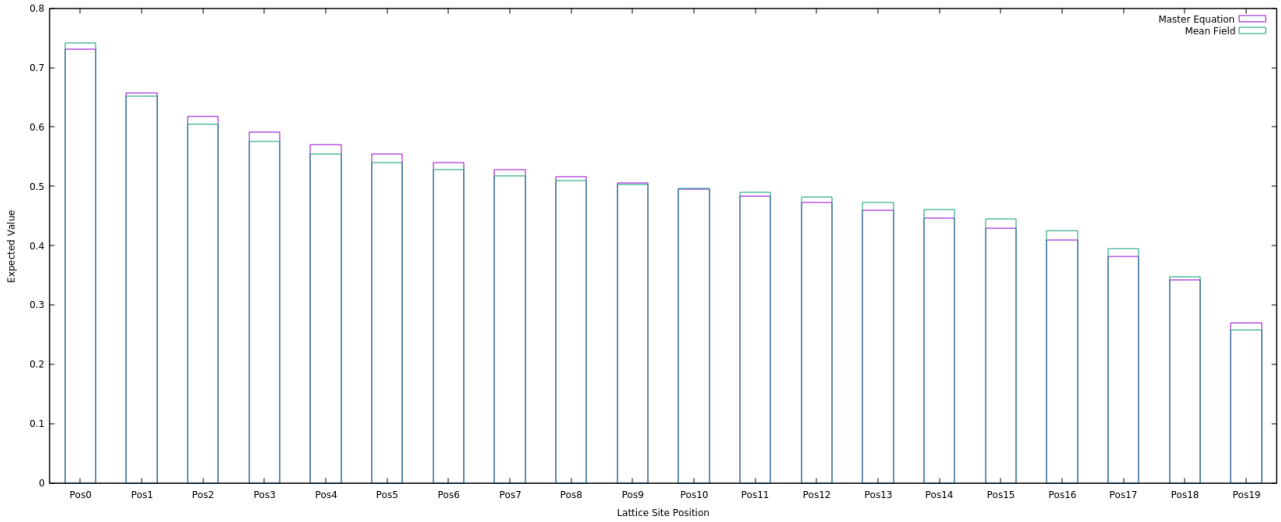
Differences

We look at a lattice with flow rates of the form:

$$\alpha(t) = h_1(t) = \cdots = h_{19}(t) = \beta(t) = 0.1 \tag{4.6}$$

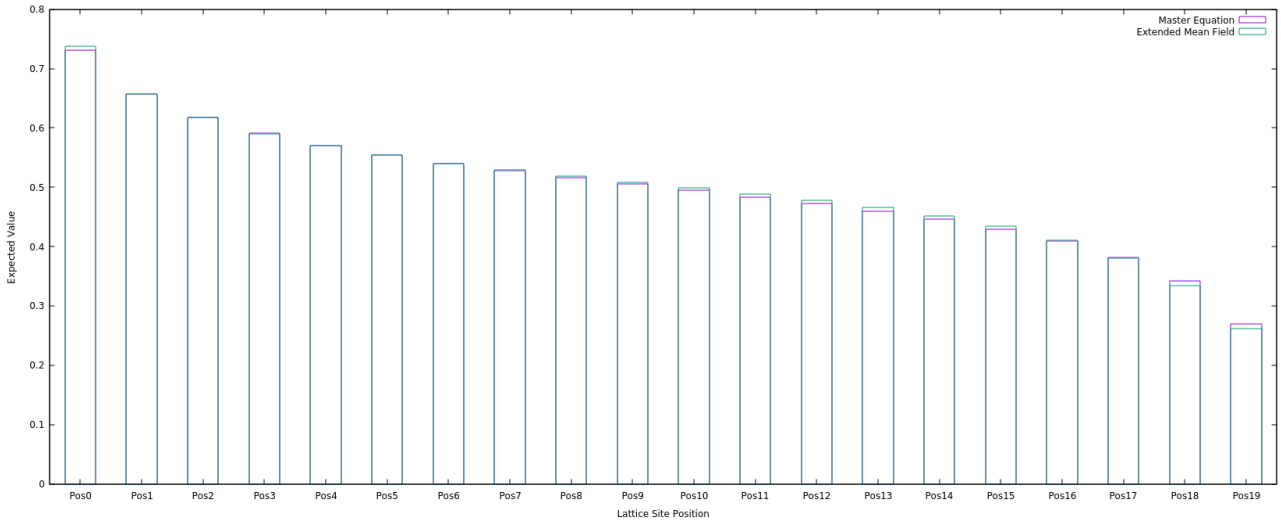
Which we will call Model 6 (M6). The results of which can be seen in the following figure (fig. 4.20). In many cases the expected values as calculated with the *non extended mean field model* are slightly below or above the correct solutions. It seems that this method is slightly less accurate than the *master equation*.

Figure 4.20: M6, Steady States, Non Extended Mean Field and Master Equation



In figure 4.20 the expected values as calculated by the *mean field model* are too high for the first half of the lattice sites and too low for the second half. Simulating the same lattice with the *extended mean field method* yields similar results (fig 4.21). The extension of the model yields more accurate steady states in this case, however.

Figure 4.21: M6, Steady States, Extended Mean Field and Master Equation



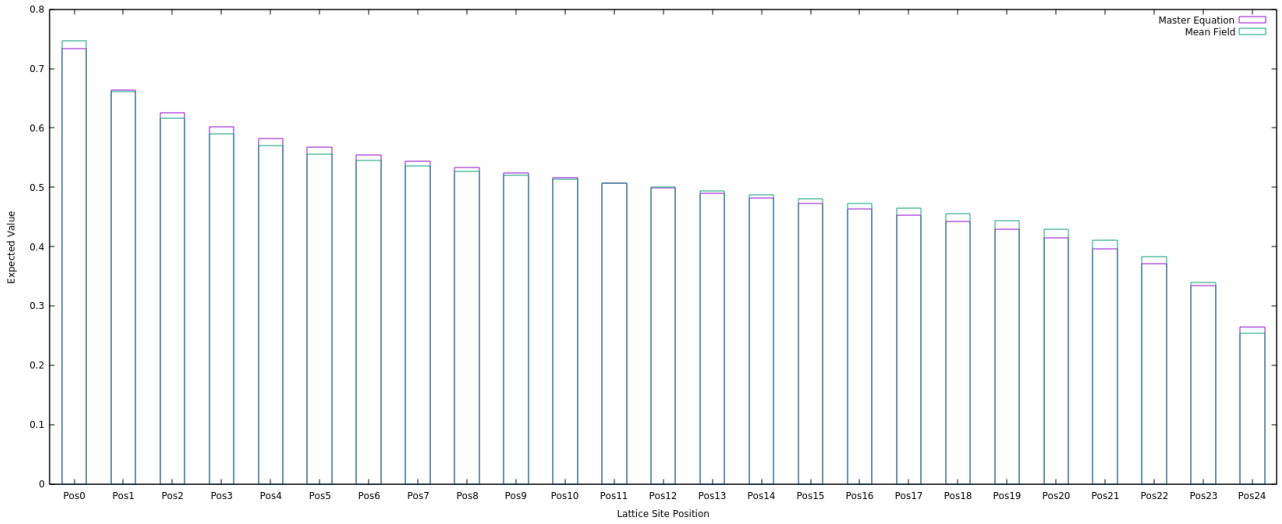
This behaviour remains the same for a lattice with flow rates of 10, with the error roughly equal to that of a lattice with low flow rates.

Consider therefore the next model, Model 7 (M7):

$$\alpha(t) = h_1(t) = \cdots = h_{19}(t) = \beta(t) = 10 \tag{4.7}$$

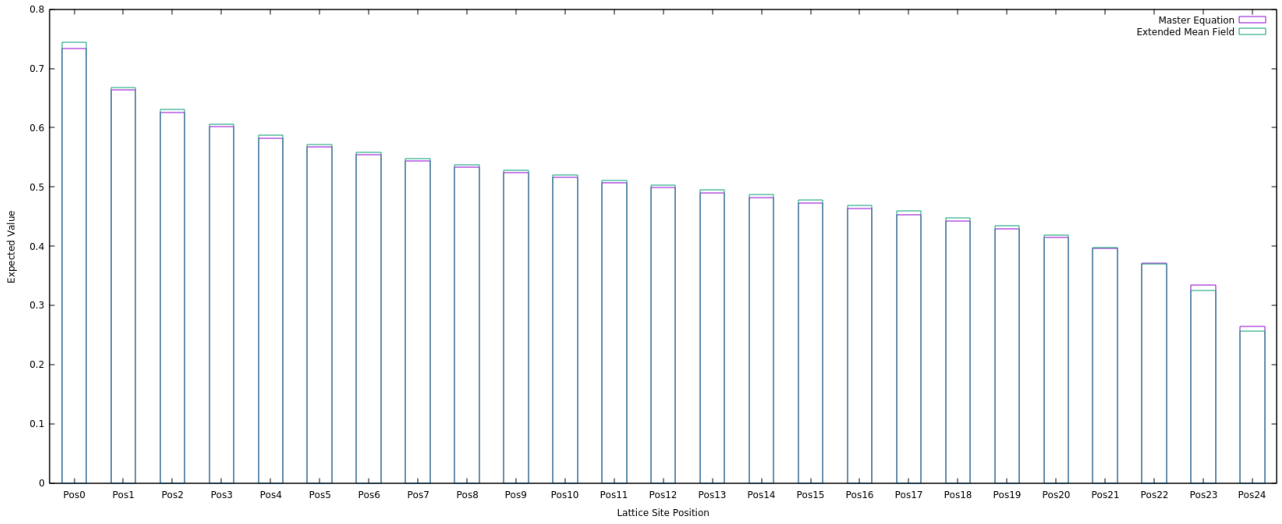
The next figure (fig. 4.22) shows the steady states of such a lattice, calculated using the *non extended mean field model*. We also extended the size of the lattice to 25 to illustrate, that the results will remain similar regardless of lattice size.

Figure 4.22: M7, Steady States, Non Extended Mean Field and Master Equation



The error, however, is relatively small, ranging from around 0.03 to 0.02 at the high end to accurate values for the sites in the middle of the lattice. The *extended mean field model* increases the accuracy even further, producing approximately equal results to the *master equation* (fig. 4.23).

Figure 4.23: M7, Steady States, Extended Mean Field and Master Equation

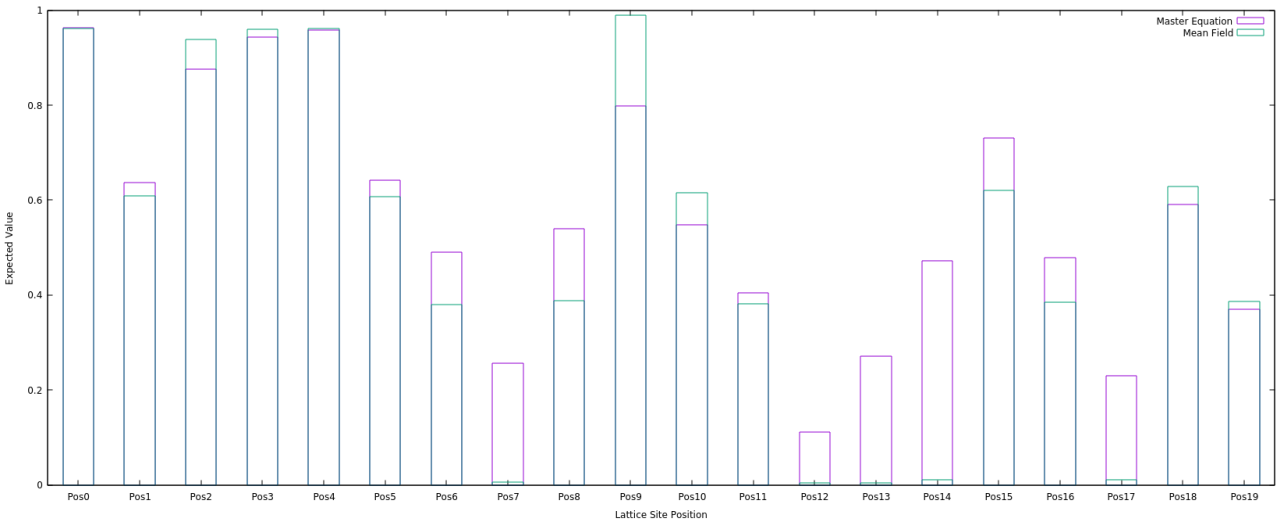


This generally is the case for lattices where the flow rates do not change abruptly. There is, however, one case in which the *mean field model* produces results that are very different from the correct solutions. This happens in lattices containing consecutive sites with very high flow rates, followed immediately by consecutive sites with very low flow rates. For this case we define Model 8 (M8) to be:

$$\begin{aligned} \alpha &= 10 \\ h_1 &= 1 \\ h_2 = h_3 = h_4 &= 10 \\ h_5 = h_6 = h_7 &= 1 \\ h_8 = h_9 &= 100 \\ h_{10} = h_{11} = h_{12} &= 1 \\ h_{13} = h_{14} = h_{15} &= 100 \\ h_{16} = h_{17} &= 1 \\ h_{18} &= 100 \\ h_{19} = \beta &= 1 \end{aligned} \tag{4.8}$$

The results of such a case are shown in figure 4.24 below.

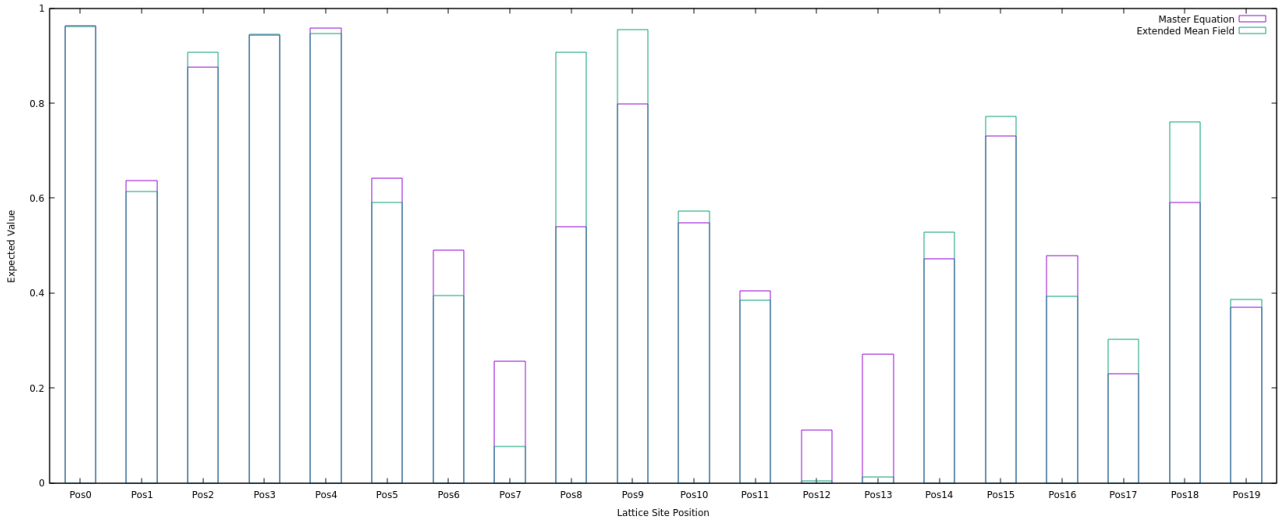
Figure 4.24: M8, Steady States, Non Extended Mean Field and Master Equation



In this example, the *mean field model* fails to accurately calculate the expected values of many of the lattice sites. In the case of position 14 the calculation is off by 0.46.

The extended model is marginally more accurate, yet the error remains high for a lot of lattice sites. While the values for positions 7 and 14 are more accurate, the value for position 13 is almost exactly as inaccurate as calculated by the non extended model. Additionally, the value for site 8 is far less accurate than with the non extended model (fig. 4.25).

Figure 4.25: M8, Steady States, Extended Mean Field and Master Equation



4.3 Advantages of the Master Equation

Additional Information

Because the *master equation* takes the probabilities of every state into account, it allows the program to compute more than just the expected values of a site being occupied. It enables the calculation of the covariances of the states and the probabilities of any single state or number of states can be saved and examined. It is also possible to calculate the multiplicative expected values for any number of states. This ability was detrimental to determine the initial values of the *extended mean field model*. For any additional extension of the *mean field model* this functionality will be useful as well. This is because extending the *mean field model* further will require the multiplicative expected values of three or more states as the initial values of the differential equation. These values can be used to analyse the systems in more detail and allow for a better understanding of them.

Accuracy

As is shown above, the *mean field model* in certain cases fails to accurately calculate the expected values. The *master equation* does not have that problem as the underlying calculations are exact and able to handle, for example large spikes or dips, in flow rate. The Accuracy of the *master equation* method only depends on the accuracy of the solving method used for the differential equation $\dot{x} = Ax$.

4.4 Advantages of the non extended Mean Field Model

Speed and Memory

The *mean field model* is undeniably faster than the *master equation method*. This is due to a significant difference in floating point operations the algorithms have to perform in order to simulate a single time step. This lower number of calculations also means that there is less data needed for any time step, which allows much larger lattices ($6 * 10^8$ on a system with 32 GB of RAM), in comparison to the *master equation* which reaches the memory limit of most computers at a lattice length of around 40.

4.5 Advantages of the extended Mean Field Model

Accuracy compared to the non extended version

Extending the *mean field model* can yield more accurate results in certain situations. However, these situations are mainly limited to the cases in which the non extended version is already accurate to a reasonable degree. Cases in which the extension yields far better results are rare. The greatest improvements can be found in trajectory shape and trajectory accuracy. However the performance impact of the extended version is negligible compared to the non extended version. As such, implementing the extended version is the better choice in many cases.

Chapter 5

Conclusion

Both, the *non extended mean field model* as well as its extended version trade accuracy for a smaller memory footprint and a higher simulation speed. Their main downsides are the simulation of lattices with large differences in flow rate as well as time dependent flow rates. If a system is simulated, in which the flow rates have similar values and if that system additionally has time invariant flow rates, both *mean field methods* are able to produce accurate results for the expected values of *TASEP systems*. This is where the disadvantages of these methods are beginning to show, because if any variability in the flow rates is introduced, the accuracy of these methods begins to decrease. This means that if the lattice has time variant flow rates or large flow rate changes, the *mean field methods* are unable to produce accurate results. This is also true for the trajectories of the expected values, as their behaviour over time until an equilibrium state is reached will differ significantly from the correct solution. This makes it hard to perform accurate observations of the behaviour of the system until it has reached its steady state. Lastly, the *master equation method* supplies much more information about the lattice if implemented, the program can calculate variances, covariances, probabilities of individual lattice states and probabilities of arbitrary groupings of lattice states. These values simply cannot be calculated using the *mean field methods*.

In general, if a simple large system is simulated, the *mean field methods* are able to produce correct results quickly. If the behaviour of a more complicated system is simulated, the *master equation* produces much more accurate results, and it enables a deeper insight into the system.

Bibliography

[1] C. T. MacDonald, J. H. Gibbs, and A. C. Pipkin, "Kinetics of biopolymerization on nucleic acid templates," *Biopolymers*, vol. 6, 1968.

[2] H. J. Hilhorst and C. Appert-Rolland, "A multi-lane TASEP model for crossing pedestrian traffic flows," *J. Stat. Mech.*, vol. 2012, p. P06009, June 2012.

[3] M. E. Foulaadvand and M. Neek-Amal, "Asymmetric simple exclusion process describing conflicting traffic flows," *EPL*, vol. 80, p. 60002, Dec. 2007.

[4] L. Grüne, T. Kriecherbauer, and M. Margaliot, "Random attraction in the tasep model," *SIAM J. Appl. Dyn. Syst.*, vol. 20, pp. 65–93, 2020.

[5] M. Margaliot, L. Grüne, and T. Kriecherbauer, "Entrainment in the master equation," *Royal Society Open Science*, vol. 5, p. 172157, Apr. 2018.

[6] I. Z. Kiss, J. C. Miller, and P. L. Simon, *Mathematics of epidemics on networks*. Interdisciplinary Applied Mathematics, Cham, Switzerland: Springer International Publishing, 1 ed., May 2017.

[7] R. A. Blythe and M. R. Evans, "Nonequilibrium steady states of matrix-product form: a solvers guide," *Journal of Physics A: Mathematical and Theoretical*, vol. 40, pp. R333–R441, Oct. 2007.

[8] B. Derrida, M. R. Evans, V. Hakim, and V. Pasquier, "Exact solution of a 1D asymmetric exclusion model using a matrix formulation," *J. Phys. A Math. Gen.*, vol. 26, pp. 1493–1517, Apr. 1993.

[9] D. Chruściński and S. Pascazio, "A brief history of the GKLS equation," *Open Systems & Information Dynamics*, vol. 24, p. 1740001, Sept. 2017.

[10] D. T. Gillespie, "A rigorous derivation of the chemical master equation," *Physica A: Statistical Mechanics and its Applications*, vol. 188, pp. 404–425, Sept. 1992.

[11] L. Jun-Wei, L. Bo-Liang, and H. Yong-Chang, "An extended master-equation approach applied to aggregation in freeway traffic," *Chinese Physics B*, vol. 17, pp. 473–478, Feb. 2008.

[12] J. Lebowitz, A. Lin, P. Nielaba, V. Privman, Z. Rácz, S. Redner, L. Schulman, R. Sprik, H. Takayasu, T. M., J. Yeomans, and K. Ziegler, *Nonequilibrium Statistical Mechanics in One Dimension*. Cambridge University Press, 1999.

[13] L. P. Kadanoff, "More is the same; phase transitions and mean field theories," *Journal of Statistical Physics*, vol. 137, pp. 777–797, Sept. 2009.

[14] T. Parr, N. Sajid, and K. J. Friston, "Modules or mean-fields?," *Entropy*, vol. 22, p. 552, May 2020.

[15] J.-Y. L. Boudec, D. McDonald, and J. Mundinger, "A generic mean field convergence result for systems of interacting objects," in *Fourth International Conference on the Quantitative Evaluation of Systems (QEST 2007)*, IEEE, Sept. 2007.

[16] L. Grüne, "Numerische Methoden für Differentialgleichungen," 2020. [Online; accessed 29-September-2020].

[17] Matlab, "Choose an ODE Solver," 2020. [Online; accessed 17-September-2020].