Modelling transcription factors diffusion in 3D using Hi-C data

A.M. Dumitrana

A thesis submitted for the degree of Master of Science (by Dissertation)

Department of Life Sciences

University of Essex

Date of submission (October 2019)

Acknowledgements

I would like to give huge thanks to my supervisor Dr. Radu Zabet for the opportunity to work on this project and for all the help and feedback during this year, to the Zabet group for the endless help and support, and to the genomics group and PGR community.

I would also like to thank University of Essex for providing its facilities and staff to develop my skills and to complete this degree.

List of contents

Abstract

The influence of the transcription factors (TFs) as a result of their interaction with the genetic material made it subject to a lot of studies. From roles of TFs in shaping the DNA landscape, to their functionality and purpose in cell cycle and identity, these DNA binding molecules can upregulate or downregulate the rate at which transcription occurs. Their main mechanism of functioning is described by their ability to interact to DNA, namely, to find their target site and bind to it. In the search mechanism also known as facilitated diffusion, TFs can float freely (Brownian motion) in the nucleoplasm and can bind to non-specific sites performing one-dimensional walks along the DNA strand. Once bound, TFs can slide, hop, or jump across. Recent technological advancements enabled modelling of facilitated diffusion while accounting for the 3D architectural structure of the DNA and parameterization with actual biological data using high-resolution (sub-kilobase) measurements of 3D contacts. In this research, the influence of such environment in the search mechanism performed by the DNA binding molecule is outlined and the model takes in consideration the probability of a TF to rebind on DNA fragment that might be hundreds of base pairs apart but comes in close proximity as a consequence of the DNA's 3-dimensional structure. DNA fragments that come in contact with each other has been hypothesised to influence the search speed. While other effects like crowding (presence of other non-cognate species of DNA binding proteins) have been shown to influence the speed of the facilitated diffusion mechanisms by covering non-specific binding sites, tests ran on the model with nucleosomes being bound to DNA showed that the intersegmental jumps, being performed by the TFs, are affected by the number of nucleosomes as well as a certain probability of the protein to stay in the microenvironment and to not completely dissociate in the nucleoplasm.

Introduction

A cell's identity is modelled by processes protein interaction that happen inside. One of these shaping mechanisms that has been studied broadly is transcription. Activation or silencing are results of the interaction between transcription factors (TFs) and DNA (Woringer and Darzacq, 2018). The interaction also known as facilitated diffusion, is the process of searching and binding to a specific site for a given TF. Building models that mimic the search process led to the investigation of different factors that come into play and impact the facilitated diffusion mechanism, the environment in which the process occurs governing the TFs' velocity and frequency with which it binds to a given site (Riggs et al., 1970; Berg et al., 1981; Elf et al., 2007; Hammar et al., 2012; Zabet and Adryan, 2012a; Zabet and Adryan 2012b; Woringer and Darzacq, 2018). The cellular landscape controls the process and the process in turn dictates the rate of gene expression, more specifically the rate of TFs binding to their specific site (Zabet and Adryan 2012a).

The free diffusion (3D) process through which the DNA-binding molecules go, can be described as a free motion in the fluid (nucleoplasm or cytoplasm for prokaryotes). While the free diffusion is a well-established process through which the molecules go through to find the binding sites, the rate at which TFs find the sites in vivo is much greater that the rate of free diffusion alone (Riggs et al., 1970; Berg et al., 1981; Elf et al., 2007; Hammar et al., 2012; Zabet and Adryan, 2012a; Zabet and Adryan 2012b). Furthermore, another mechanism thorough which TFs perform the search process is described as a 1-dimensional (1D) random walk. The biophysical process of 1D diffusion shelters the diffusion of DNA, by attaching to the string of nucleotides and performing either a sliding event or a hopping event. The combination between the 1D random walk and the free diffusion facilitates the TFs to find their site in a shorter period amongst the numerus non-specific DNA regions and different protein species that occupy the neighbouring volume of space.

The pioneering work (Riggs et al., 1970, Berg et al., 1981) that describes the combination of 1D diffusion and 3D diffusion shows how the reduced dimensionality from free diffusion to 1D random walk has increased the speed rate. The 1D random walk is a slower process that free diffusion, but the rate of finding the binding site is 10-100 times greater than free diffusion alone because of the reduced dimensionality of the search process (Woringer and Darzacq, 2018).

The facilitated diffusion model proposed by (Mirny et al., 2009) describes the TF molecule as having two states: the search state or S state and recognition state or R state. In the 1D random walk a DNA binding protein would adopt the S state when interacting with non-specific site through the electrostatic interactions with the DNA. Coming into close proximity with a target site, the protein would undergo a conformational change (Leven and Levy, 2019) to recognise the target site and bind to it. The recognition mode is characterised as the process in which a TF adopts a conformation that resembles the DNA specific site's conformation (Piatt et al., 2019). In their work (Zabet & Adryan, 2012), model presented adopts only one mode due to the research being focused on combining 1D diffusion with 3D diffusion and other properties like crowding.

Recent technological advancements enabled modelling of facilitated diffusion while accounting for the 3D architectural structure of the DNA and parameterization with actual biological data using high-resolution (sub-kilobase) measurements of 3D contacts (Rao et al., 2014; Cubenas-Potts et al., 2017; Zabet and Adryan 2012c).

While other effects like crowding (presence of other non-cognate species of DNA binding proteins) (Ghosh et al., 2018) have been shown to influence the speed of the facilitated diffusion mechanisms by covering non-specific binding sites, tests ran on the model with nucleosomes being bound to DNA resulted in a low number of intersegmental jumps being performed by the TFs.

Literature Review

Broad picture

The activation of a gene requires transcription factors (TFs) to perform the search in the nucleoplasm (or cytoplasm in the case of prokaryotes) for the binding site on the DNA. This leads to the formation of an entire transcriptional apparatus. Other similar processes of protein interaction that are present shape the cellular landscape (Woringer and Darzacq, 2018). The rate of transcribed sequences in this environment is dictated by the rate of site-specific binding of transcription factors (Zabet and Adryan, 2012a). Thus, it can be stipulated that the environment governs the frequency and velocity of TF site binding (Woringer and Darzacq, 2018). Mechanisms that regulate the transcription process have been closely studied and highly regarded by many disciplines including developmental biology, drug screening and cancer biology.

General aspects of TF dynamics

Studies like (Riggs et al., 1970; Berg et al., 1981; Elf et al., 2007; Hammar et al., 2012; Zabet and Adryan, 2012a; Zabet and Adryan 2012b) present the process TFs go through to reach their target site only by free diffusion (thermal agitation of a fluid causing the motion of particles). It has also been discovered the occurrence of a facilitated diffusion mechanism, a combination of free diffusion and 1-dimensional diffusion. The facilitated diffusion is characterized as a biophysical process that covers the diffusion on DNA (surface of reduced dimensionality) and 3D diffusion. The diffusion on DNA is slower than 3D diffusion, but the finding of the binding site is 10-100 times faster than the 3D free diffusion alone due to the reduction in dimensionality where the TF performs the search process (Woringer and Darzacq, 2018). Following a search event, the transcription factors bind to the DNA and perform the so called 1-dimensional diffusion. This includes either of the possibilities to perform a random walk along the DNA strand (without detaching itself from the strand) or a hopping action (the transcription factor will unbind from the DNA loosely but not completely). Another movement that can be performed by the small molecules can be a complete detachment from the DNA and disassociation into the cytoplasm referred to as 3-dimensional diffusion. The specified events combined give the facilitated diffusion mechanism. The published work of (Woringer and Darzacq, 2018) also discuss the exploration properties of proteins that adhere on nuclear substructures as having two parameters, one for the random walk and one for the measurement of the available space of diffusion. They also suggest that due to the short-lived interactions between the TF and the DNA entity, the diffusion phenomenon could not be described by general biochemical techniques. Thus, there is a need for computational involvement into the TF mechanistic functionality (Woringer and Darzacq, 2018). Even as the pioneers (Riggs et al., 1970, Berg et al., 1981) have brought the conclusion of the reduced dimensionality from a 3-dimension diffusion to 1-dimension diffusion as a catalyst for speeding the process substantially, this would apply to a DNA of linear value (considering it a string), rather than seeing the DNA as a 3-dimensional structure in space. The probability of the transcription factors to dissociate into cytoplasm (the mass of the TF playing a role in the speed of the diffusion) is influenced by factors like the affinity for the binding motifs present in the vicinity that have to be accounted for (Cortini and Filion, 2018). Each time the TF is unbound from the DNA, the proteins are in search for their target site. Factors like crowding (Ghosh et al., 2018), where other molecules are bound to a possible target site, influence the search process and the model presented in (Zabet and Adryan 2012c) considers this. The large-scale simulations model developed in (Zabet and Adryan 2012a; Zabet and Adryan 2012b; Zabet 2012) is focusing of the one-dimensional walk of the transcription factor, as a bound TF, the protein will execute a random walk along the DNA strand. Other models focus on 3D aspects and assume random/uniform affinity profile for the TF, which is not true for real biological systems (Cortini and Filion, 2018; Brackley et al., 2012). When it unbinds completely, the TF disassociates wholly from the DNA and diffuse into cytoplasm. One assumption of the model presented by (Zabet and Adryan, 2012a) is that DNA is a linear string, but this is clearly not the case. Distal loci can come into 3D proximity and one possibility is that TF molecules performing hopping can be relocated on the 3D proximal DNA fragment instead of the vicinity of the dissociation sites. First, we look closer at what 1-dimensional diffusion refers to.

Additional aspects that impact TF binding mechanism

TFs have the capacity to communicate quantitative information regarding different signals presented by the environment. Transcription factors activation can present different dynamical patterns that are triggered by various stresses (Hao and O'Shea, 2011; Ezer et al., 2014). The dynamical patterns that are communicated to the promoters are further interpreted, which would in turn produce a variety of gene expression patterns (Hao and O'Shea, 2011). It is clear that specific stress signals trigger different responses, but the oscillations in gene expression can be influenced by the duration modulation or frequency of a transcription factor (Hao and O'Shea, 2011). The TF activation of a specific dynamical pattern would affect the expression as a response of a gene to a specific stress condition that might vary based on alterations in specific parts of the gene's promoter (Hao and O'Shea, 2011). Alterations at the promoter are influenced by the TF binding sites number, their position and the presence of nucleosomes with respect to the binding site order (Ezer et al., 2014). Thus, the influence of a specific binding pattern on the gene expression can be seen as a consequence of a stress stimulus applied and its duration (Hao and O'Shea, 2011). The time in which a TF finds a target site sets in turn the response time of a gene. The search time of the TF for the specific target gene is dependent on the free diffusion (3D search) in cytoplasm and by the nonspecific search alongside the DNA strand (1D search) as observed in prokaryotes (Hettich 2018). In other similar work, the DNA residence times of TF were measured by single-molecule imaging (Gebhardt et al., 2013). The findings presented in (Gebhardt et al., 2013) suggest three modes by which DNA can bind with TFs, that is, monomeric, dimeric and indirect DNA binding. They have used a time-lapse approach and tracked the residence times of TF binding to DNA with values between 50 ms to several seconds. Even with the advancements in monitoring TF dynamics by different biochemical methods (Cha and Zhou, 2014, Gebhardt et al., 2013), the photobleaching would cause the fluorescing dyes to become sparse, or the cellular movements to avert reliable observation of a long-term bound molecule (time-lapse illumination). The majority of studies have a high impact on the understanding of target-search mechanisms as respected tools for evaluating the movement of proteins on DNA.

Some TFs have the ability to bind to DNA during mitosis, such proteins are known as mitotic bookmarking TFs or simply BFs. These molecules are known to maintain the rapid functional regulatory complexes re-establishment in post-mitosis (Festuccia et al., 2019). The general rule when it comes to TFs is that during mitosis their concentration level is highly reduced. This is due the fact that the chromatin becomes highly condensed and a high percentage of DNA becomes inaccessible. Likewise, the decrease in TF concentration is linked to the phosphorylation of most of the regulators in mitosis thus the TF ability to bind to DNA is greatly reduced (Rizkallah et al., 2011). The BFs are believed to attach to their binding sites during cell division, through this mechanism these factors would deliver information regarding the gene regulation to the daughter cells, as these regulatory elements are left accessible (Caravaca et al., 2013).

It can be summarised that the transcription factors influence cell fate and/or many other mechanisms that are located in this cell. The cell fate, in turn, depends of the DNA state, the DNA state would influence the actions of transcription factors. Ultimately, any factor that has direct or indirect effect on the diffusion will affect the reaction rate of the transcription factors.

1D diffusion

Supposing a TF molecule came in contact with the DNA, it could perform one of three possible actions that is chosen stochastically (Zabet and Adryan, 2012a; Mirny et al., 2009). The (Mirny et al., 2009) model assumes the two binding modes (search mode and recognition mode), while the (Zabet & Adryan, 2012) model does not distinguish between them. The facilitated diffusion model (Mirny et al., 2009) also describes the recognition and binding of the TF to the target site in a pool of non-specific sequences. The recognition performed by the DNA-binding protein is executed rapidly while being in an environment filled with different proteins (non-cognate species). The facilitated diffusion model contains a two-state mode. This implies a search mode and a recognition mode in which the TF can be found while in 1D random walk. In the search mode or the S-state, the TF is bound to the DNA in a non-specific manner. The TF-DNA bond is a result of the electrostatic interactions between the residues of the TF that are positively charged, and the DNA phosphate groups that are negatively charged (Viadiu and Aggarwal, 2000). Binding to the target site requires the TF molecule to recognise it, thus the protein enters its recognition mode or R state. In the R state, the TF forms a hydrogen bond between its residues and the DNA bases (von Hippel, 2007). The two states approach sustain the idea that the DNA-binding protein is in a flexible state where it can switch between S-state and R-state. (Leven and Levy, 2019) show the relation between the states as being negatively correlated. In doing so, they have introduced the term of frustration. Frustration can be measured as the degree of overlap between the TF positively residues that bind in the S state and the residues that bind in the R state (as calculated from the protein's X-ray structure). Coarse-grained simulations (Leven and Levy, 2019) showed how at high frustration (the similarity or overlap between the two states is negatively correlated: high frustration means low similarity between the two states) the energy of sliding is high while the recognition of the target site probability is poor. Likewise, for low frustration where the similarity between the two states is high, the sliding energy is low and the probability for recognising the specific site is high (Leven and Levy, 2019). The (Mirny et al., 2009) model assumes that the TF molecule exists in two conformations, the search mode conformation with low sequence specificity and the recognition mode in which the molecule conformation resembles the target site's conformation (Piatt et al., 2019). (Zabet & Adryan, 2012) model assumes only one mode. This is mainly because in the research the focus is on other aspects such as crowding and combining 1D diffusion with 3D diffusion.

The action in which the attached TF performs a random walk on the DNA strand without detaching itself from the DNA is called sliding (Wunderlich and Mirny, 2008). The sliding would occur either right or left, but the TF would maintain its orientation on the DNA. This assumes an unbiased random walk, as the probability to slide left or right is equal (Zabet and Adryan, 2012a). Whereas in a biased event, the probability to slide left or right is influenced by the energy landscape with respect to the efficiency, speed and direction, in other words it is sequence-dependent (funnel effect) (Weindl et al., 2009; Slutsky and Mirny, 2004). (Cencini and Pigolotti, 2018) showed that the biased random walk is connected to the AT gradients being present in the DNA region base structure in the vicinity of the binding sites. They further identified an enhancement in the probability of TFs to localise the target sites while sliding due to the funnel effect. The TF would be able to change direction when it would unbind partially (hopping action) and rebind or detach completely and rebind later on.

The sliding event involves the TF molecule moving to a proximal non-cognate site without using ion recondensation against the DNA backbone therefore the rate of diffusion on the strand would be of little dependence to the salt concentration (Piatt et al., 2019). While the ion recondensation is of little reliance for sliding, the hopping mechanism means the partial de-attachment from the DNA would enable the recondensation. This would mean that the ratio between sliding and hopping is given by their coefficients which are salt-dependent (Piatt et al., 2019).

Single-molecule dynamics show a hopping action performed by transcription factors flanked by clustered binding sites in subnuclear regions that are restricted spatially (Gowers et al., 2005, Brackley et al., 2012), and points at the possibility of topological chromosome domains that are in the nucleus to implement a physical sequestering mechanism that would shape the local gene activities. This concept was extrapolated from Hi-C experiments (chromosome conformation capture coupled with high-throughput sequencing) with a need to be followed up and thus confirmed (Liu et al., 2017). Experimental data of individual trajectories of Sox2 that were provided and analysed by (Liu et al., 2017) clearly states the hopping action and local interaction in the nucleus that is performed by the transcription factor.

Non-specific binding of transcription factors can influence the speed of site-specific localisation and binding. Further, the influence of nucleosomes on the TF search time for 1- and 3-dimensional diffusion has been observed (Murugan, 2018). It was suggested that TF behave a certain way when the nucleosome roadblocks are formed on DNA that present target sites. Essentially, when the nucleosomes are present between TF and their cognate site, the nucleosomes would apply a maximum steric hindrance (Zabet and Adryan, 2015) amount that would coerce the TF present between nucleosomes or close to one to enforce a sub-diffusive dynamic, thus, enhancing the search dynamics of the TF (1D diffusion would increase in this sense) (Murugan, 2018). This suggests that the DNA region should be nucleosome free for the TF to find the targeted site. In their research (Murugan, 2018) have described the reduction in speed of a TF when coming in close proximity to a cognate site in order for it to form a close junction and a complex formation. This is not always the case, when it comes to pioneer TFs, these transcription factors have the ability to bind to inaccessible nucleosomal DNA Stadhouders et al., 2019).

(Raccaud et al., 2019) showed in their research how the DNA state influences the TF affinity for the DNA; they have focused on the properties of the transcription factors in interphase. It showed the impact of non-specific TF- DNA binding, and how it increases the speed of the search event for the specific sites, thus the effect on accessible chromatin. Looking at both ChIP-seq datasets and fluorescence microscopy as well as extensive literature search, they suggest that the TFs co-localization on mitotic chromosomes seen by the microscopy is caused by the non-specific DNA interactions. The specific binding observed in the mitotic ChIP-seq data is caused by a minority of TFs thus the observed interactions on the chromosomes by fluorescence microscopy can only be due to the non-specific binding of TFs with a few TF exceptions that displayed reduced mitotic chromosome association (Caravaca et al., 2013;

Raccaud et al., 2019). Their work also points out the possible existence of a TF property, more specifically, that the binding event would open regions of chromatin that are condensed, thus their hypothesis is that TFs can control gene reactivation at the pre-stage of mitosis, mediating cell fate (Raccaud et al., 2019). The researchers also highlight the need for further study into post-translational modifications and 3D structures of TF- DNA contact interface.

Likewise, non-specific binding of TFs to DNA segments was found to impact the dynamics of transient DNA loop formation (Shin and Kolomeisky, 2019). The formation of these loops would take place when a protein would bind simultaneously at two DNA sites that are spatially distant. So far, the formation of such loops was shown to happen in a crowded environment (Stiehl et al., 2016).

3D DNA architecture

The genome conformation in different stages in which the cell can be found, has a direct impact on the transcriptional regulation- that is the architecture of the DNA would play a major role in the gene expression dynamics (Pal, Hoinka and Przytycka, 2019; Delaneau et al., 2019; Stadhouders et al., 2019; Brackley et al., 2016; Tao Hu, Grosberg A. Y. and Shklovskii B. I, 2006.). More specifically, fragments of DNA that come in close proximity sustained transgene activation in Drosophila cells that is required for promoter-enhancer interactions. As a result, the spatial organisation (e.g. Topological associated domains-TADs and loops) facilitates specific pairing between enhancers and promoters, leading to various gene expression outcomes. However, the topological organisation is in turn driven by TF-DNA interactions (Stadhouders et al., 2018). This interplay would directly influence the cell's identity by modulating the gene activity, each cell having its specific expression. The conformation of the DNA would lead to silencing or activating the TF-DNA binding by respectively restricting or making available target sites (Stadhouders et al., 2019).

The transcription process can be considered a stochastic one. (Zabet and Adryan, 2012a; Brackley et al., 2012; Ancona et al., 2019) describes this process as being a 'bursty' one, meaning that by observing a cell's gene and its transcription events, there can be seen clusters of events in close proximity. Following this cascade of events in each other's vicinity a refractory period is observed. These time intervals of dormancy vary from eukaryotic genome (hours) to bacteria (minutes) (Ancona et al., 2019). This study also points out that the transcriptional bursts are formed/associated with the supercoiling of the DNA (over-twisting), as a result of the low rate of transcription overall.

Furthermore, TFs are known to have complex binding patterns. (Cha and Zhou, 2014) have followed on this idea and investigated the combinatorial binding patterns and cis-regulatory modules (CRMs- A short DNA region that presents co-localized multiple TF binding sites clusters). They suggest the possibility of an ordering preference between binding sites of various TFs in a regulatory region. In this regard, the researchers developed new statistical methods for the detection of combinatorial binding patterns using ChIP-seq data (Cha and Zhou, 2014). They have tested procedures using the Ripley's K-function (function used widely for analysis of spatial data for detecting point patterns) in order to identify binding patterns associated with large clusters (between two given TFs) of binding sites assuming that the TFs follow an inhomogeneous Poisson point process. They have used ChIP-seq data from mouse embryonic stem cells, due to the uniqueness of these cells or being pluripotent and their ability to replicate indeterminately and their wide use in clinical research.

GRiP computational tool performs simulations of facilitated diffusion

Based on the research that was presented so far on transcription factor DNA binding alongside the factors that influence the mechanism in a specific or unspecific manner, researchers (Zabet and Adryan, 2012c) developed a comprehensive computational model with estimated parameters. The GRiP model allows simulation of the search process in a versatile and efficient way that is highly customizable. The program would show representations of TFs, the facilitated diffusion mechanism, as well as a cooperative behaviour that points at interactions between TFs (Zabet and Adryan, 2012c). These simulations of the mechanism are achieved by recording information of the spatial coordinates of the molecules, collision hotspots, occupancy-bias and affinity landscape (steady-state results). The program would implement a stochastic simulation (Erban and Chapman, 2007; Gillespie, 1976) to overcome the issues that arise from dynamic crowding on DNA and consider real DNA sequences. By using these types of simulations to look at the TF search process one could observe weather there exists a preference between the two possible actions (sliding and hopping) in 1D random walk and weather this would be of significance if the TF process of finding the target site. Other models

(Das and Kolomeisky 2010) that were insufficient in providing a detailed and fast enough simulation focused on 3D diffusion alone without taking into consideration the 1D diffusion.

The model developed by (Zabet and Adryan, 2012c) showed that by mainly focusing on the rate of TFs binding non-specifically on the DNA would be sufficient as long as the molecule bound would lose contact and rebind in the vicinity of the detachment site. Thus, the program adds the probability of fast rebinding in the unbinding rate of the TFs dynamics. The events of TFs are simulated using stochastic algorithms (Gillespie, 1977). The model assumes the cytoplasm as being a perfectly mixed reservoir. The TF molecule size on the DNA is calculated as the number of base pairs of the binding motif and a number of base pairs that are obstructed by the presence of the TF on the DNA, left and right. At the end of the simulation, the model would show the computed affinity landscapes of the TF species (the program sustains multiple species of TFs), information regarding the occupancy bias that was measured for the TFs; sliding distances captured for each TF species, statistical representation of the collisions and the position of TF on the DNA strand (Zabet and Adryan, 2012c). The program calculates the affinity of the TFs for the DNA and the probability of binding a given site or entering an event of 1D diffusion is attributed equally to each TF molecule. The time spent at the biding site by the TF is dictated by the energy of binding between TF and the DNA site. After each movement of a TF on the DNA, the available unbound regions are updated for the other TFs to be able to have an entire view of the possible moves. Once attached to a site on the DNA, the TF has two possible actions: to rebind fast in the vicinity or to release in the cytoplasm (Zabet and Adryan, 2012a). A cooperative behaviour has been implemented between TFs, where the binding of one TF to a site would influence the waiting time of another TF to bind on a proximal region.

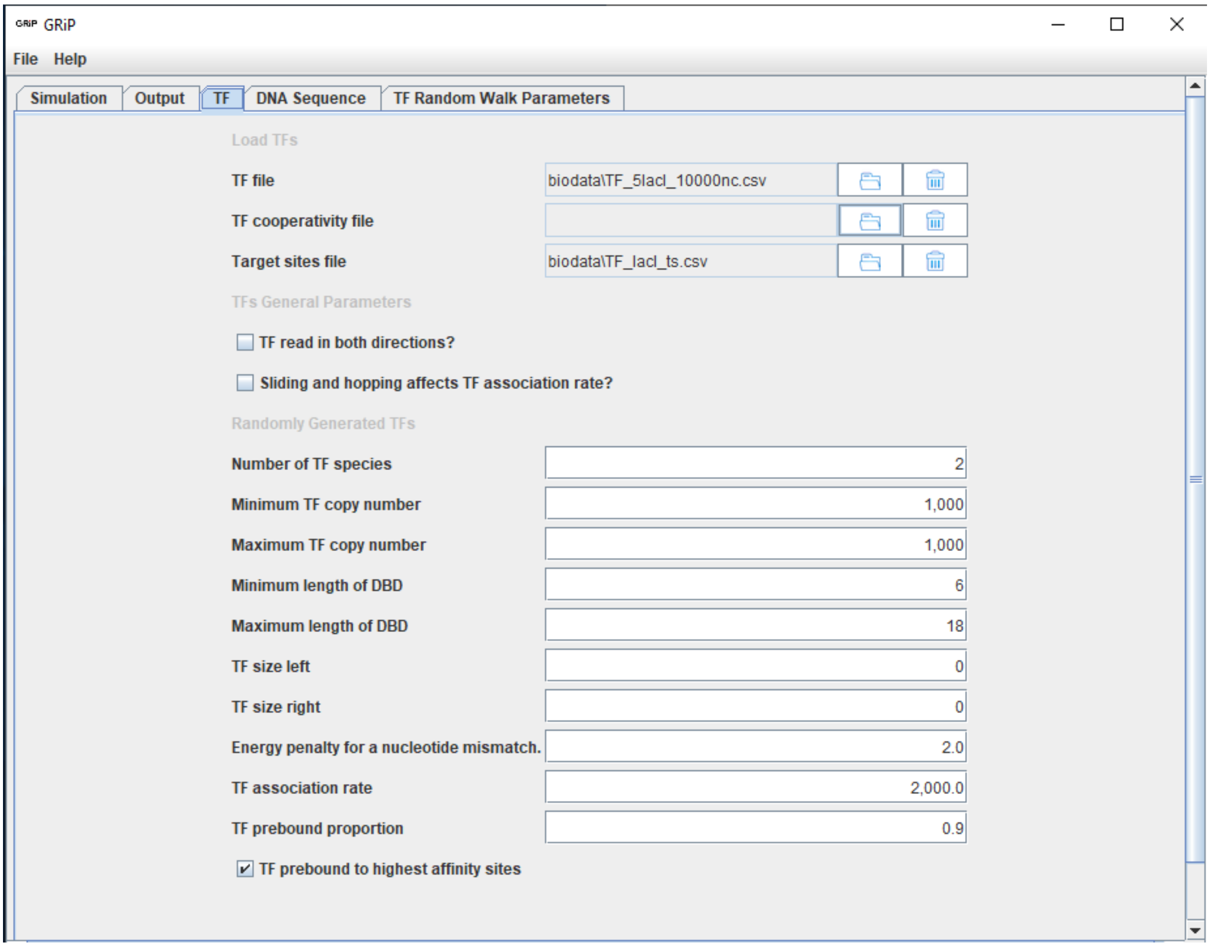
Other important aspect that is considered is the molecular crowding. The main effect that this factor would add, is the presence of other molecules on the DNA that would completely/partially cover a possible target site and considering the steric hindrance the number of bound TFs could be reduced and the search time decreases (Zabet and Adryan 2012b, Woringer and Darzacq, 2018). The coverage of DNA segments would exhaust the search space and the TF would find a target site in a shorter search time, but the crowding would shorten the search time only if there are a large enough number of TFs. Biomolecules like TFs are known to perform in their 3D search the facilitated dissociation (slip bond) performed under stress (applied force). This phenomenon happens as a result of concentration-dependence, meaning other proteins from the microenvironment being present would 'encourage' the dissociation (Dahlke et al., 2019). In their research (Dahlke et al., 2019) present a model that show how TFs in the dissociation process actually enter in a molecular bond. They refer to this specific type of bond as 'catch bond'. In this interaction the protein's dissociation is suppressed by applied forces. This bonding model would differ from others by its dependency on the micro-environmental protein concentration, alongside the dependency on the molecular sensitivity to binding geometry and force (Dahlke et al., 2019). By comparison the molecular complexes in a slip bond would disassociate under an applied force more rapidly than the molecules that form catch bonds. This difference is considered to be related to their internal structure (Dahlke et al., 2019).

Theoretical approaches lacked in considering the obstacles as moving objects and the computational approaches lacked in taking in consideration the whole DNA. An improved model of the simulation was presented in (Zabet, 2012). As the simulation time on a large-scale model would be quite increased, it would become less efficient, and for this reason they have considered adopting a sub-system approach. Molecular crowding can be caused by obstacles that are static on the DNA (such as nucleosomes) or mobile (such as other TFs) and their presence can influence the binding affinity of TFs. Thus, the occupancy of a target site may be enriched by increased crowding. However, if the crowding (Krepel et al., 2016) is produced by the static obstacles the search time is decreased, but in the presence of mobile obstacles the search time can be increased as they could form a barrier on a possible target site (Zabet and Adryan, 2013, Zabet et al., 2013).

There are still missing pieces to the research presented so far and further study to be taken. One aspect that is gaining more support for having an impact on gene regulation is 3D architecture of the DNA in the TF dynamic search mechanisms (Cortini, and Filion, 2018; Avcu and Molina, 2016; Fosado et al. 2016; Brackley et al. 2016). By comparison the real DNA sequence presents sequence heterogeneity as opposed to the theoretical models so far presented. Moreover, the sequence would have the free energy, on which the protein executes the random walk, nonuniform (Brackley et al., 2012). Here, I plan to improve GRiP tool by adding the 3D structure of DNA as a component for the simulation and test the presumed model with actual biological data.

Methods

Simulation: The implementation of the 3D structure of DNA was done using Java programming language. The GRiP program has a user interface (Figure 1), this part was not edited, it is specified as additional information about the program that is being edited. The project is structured by connecting several classes. This classes allow the user to input and change parameters of the facilitated diffusion (e.g. TFs copy number, size etc.) (Figure 1).



Load TFs

TF file: biodata\TF_5lacl_10000nc.csv

TF cooperativity file:

Target sites file: biodata\TF_lacl_ts.csv

TFs General Parameters

☐ TF read in both directions?

☐ Sliding and hopping affects TF association rate?

Randomly Generated TFs

Number of TF species: 2

Minimum TF copy number: 1,000

Maximum TF copy number: 1,000

Minimum length of DBD: 6

Maximum length of DBD: 18

TF size left: 0

TF size right: 0

Energy penalty for a nucleotide mismatch.: 2.0

TF association rate: 2,000.0

TF prebound proportion: 0.9

☑ TF prebound to highest affinity sites

Figure 1. GUI (graphical user interface) of GRiP contains the parameters that can be changed for simulation, output, TFs, DNA sequence, TF random walk parameters, for data analysis of TFs rates of facilitated diffusion.

The program can be ran from command line using the parameters file and number of steps. The minimum system requirements for running the program are as follows: Intel Core i3 Processor; 8 GB of RAM; and 64-bit operating system. Additionally, for the minimum requirements, the Hi-C interactions file should not have the sequence higher than 131000 bp and 10^8 seconds of simulation. The Eclipse version used was Eclipse Java 2018-12.

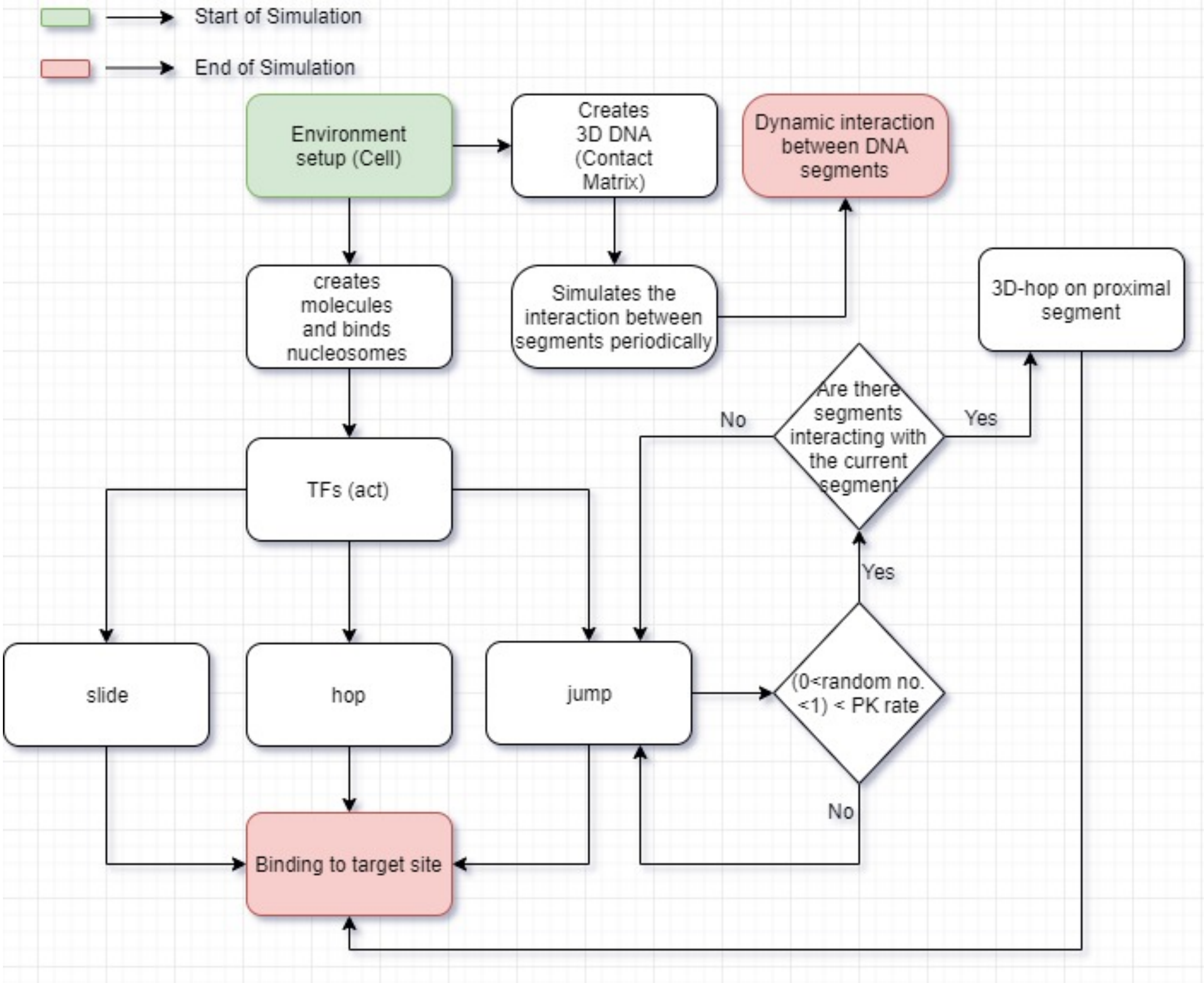

Start of Simulation
End of Simulation
Environment setup (Cell)
Creates 3D DNA (Contact Matrix)
Dynamic interaction between DNA segments
creates molecules and binds nucleosomes
Simulates the interaction between segments periodically
3D-hop on proximal segment
No
Are there segments interacting with the current segment
Yes
TFs (act)
Yes
slide
hop
jump
(0<random no. <1) < PK rate
No
Binding to target site

Figure 2. Flow diagram of the steps of the 3D implementation; the implementation of 3D-DNA architecture and the periodically simulated interactions. (random no. – a random number chosen by the program between 0 and 1; PK rate- the probability of the protein to be kept in the microenvironment)

To simulate the 3D structure of the DNA we implemented the Contact Matrix class to read the Hi-C data file provided in the parameters file. One other value that is used from this file is the size of the bins to be created. The Cell class initialises the Contact Matrix by creating a Contact Matrix object. The Contact Matrix class reads the Hi-C data file, extracts the size of the bins it has to create and adds the value from the Hi-C data to its specific bin. The data is normalized to the highest value. In order to create a dynamic environment we used an exponentially distributed random variable for a given mean (also taken from the parameters file). The time at which the Contact Matrix would simulate the matrix again is as follows: $S_t=C_t+r_v$; Where S_t

is the next simulation time, C_t is the cell time and r_v is the random variable. This implementation gave the dynamic interaction between the DNA segments (Figure 2).

A 3D-hop (intersegmental jumps) is performed using the jump action of the TF molecule (Mirny et al., 2009). The program checks whether the action is a jump, in which case, it further checks any bins that might be interacting with the current bin on which the molecule is found choosing randomly one of them (each interacting bin has the same probability to be chosen), then if a random number generated. If this random number is lower than the probability of the protein to stay in the microenvironment (PKrate)(Figure 2) then the molecule is set to perform the 3D hop.

Description of the model:

The framework for the new proposed program improvement has been outlined in (Figure 2) and shows a representation of the interaction between classes and the main objective of this research.

The system is composed of the TFs, DNA and nucleosomes. The parameters used for SuH molecules and the code for generating the affinity profiles for nucleosomes were taken from (Clark, S., 2017) and adjusted accordingly. First, we consider TFs that can bind to DNA at specific positions that are accessible with an association rate defined as the rate with which the molecules bind to DNA (Zabet, N. R., and Adryan, B. 2012; Zabet, N. R., and Adryan, B. 2015) to be $20s^{-1}$ in computing this value Clark, S., 2017 used the residence time (t_R) equal to 1 s which is the time the protein stays bound to DNA. In addition, the used specific waiting time for the molecule was 1.5 s. This would be the amount of time the TF interacts with the DNA when it is located at the strongest sites. Each action that the TF can perform has an associated probability to it. For the sliding action regardless of it being either right or left the probability is equal for sliding left or sliding right which is 0.4992 (Zabet and Adryan 2012a). As the probabilities to slide left or right are equal, the model assumes an unbiased random walk (Blainey et al., 2006). Next, after the protein performed a binding event to the DNA, the TF can relocate or change directionality by first unbinding from the local site. The associated probability for this action is P_{unbind}=1.47*e*-3 (Zabet and Adryan 2012a) and is inputted in the TF file alongside the other parameters. But the unbinding probability here would refer to or better said would offer the protein the possibility to rebind fast. This probability thus controls the micro-dissociations from the DNA in relation to the complete unbinding. To completely unbind/jump the probability estimated by Wunderlich and Mirny (2008) and used here is P_{jump}=0.1675. With regards to the nucleosomes, and the main parameters associated with them, the residence time of the nucleosomes was left at 3600s making them highly stable (Deal and Henikoff, 2010) and they were set as immobile roadblocks. However, the amount of nucleosomes used varied when performing the simulations.

Running the simulation: We used initially 3570 sets of different parameters to test the impact of the PKrate and the impact of the number of nucleosomes. We tested 10 different simulation times, 21 PKrates and 17 different numbers of nucleosomes. Following results from the 357 tested we adjusted the set of the number of nucleosomes and used 18 different values.

Data analysis: The region used for the simulations was 25951000-26082000 on chromosome 3R in D. *melanogaster.* The maximum number of nucleosomes for this region was calculated according to Sian Clark's work (Clark, S., 2017) and R script that created a nucleosome affinity vector. The affinity profile was generated using the DNA hypersensitivity sites of BG3 cells (Kharchenko et al., 2011). According to her work and (Zabet, 2012) the number of accessible regions from the given loci (here 25951000-26082000) were subtracted from the length of the region and divided by the size of the molecule (147bp) such that for the 131000 bp region, we used 828 of maximum no of nucleosomes. The accessible regions were used to generate the nucleosome affinity file using FDR1 DNaseI from modencode dataset (Boley et al., 2014). The pre-processing and the import function were written by Patrick Martin (PhD student in the lab). In the last step we used the simulation time that would require the matrix to be simulated the least and the entire sets of nucleosome numbers and PKrates to generate the AUC, and correlation between the occupancy file for the given parameters and the ChIP profile for the chromosome 3R: 25951000-26082000 in D. *melanogaster* (Martin and Zabet, 2019). The ChIP-seq profile used for BG3 and Kc167 were taken from (Martin and Zabet, 2019) in the dm6 genome version.

To understand the role the number of nucleosomes in the simulation of the intersegmental jumps, simulations were performed for 0 to 828 (with an increment of 50) no of nucleosomes and a probability of the protein to stay in the microenvironment (PKrate) between 0 and 1. The values for the observed sliding length and residence time of the TF were analysed to see how they are influenced by different number of nucleosomes and PKrates. To understand the influence of the intersegmental jumps on the transcription the simulation occupancy was normalised and converted to ChIP signal. The ChIP like profile was compared with the ChIP profiles of SuH in BG3 cells (Skalska et al., 2015).

The performance of the simulations was analysed using the Pearson correlation (Figure 10), the AUC (Figure 11) and MSE (Figure 12) scores. I have used the AUC of the ROC curve, the values for this were calculated using the method from ChIPanalyser package (Martin and Zabet, 2019). The metrics were calculated between the silico ChIP generated profile based on the occupancy profile of the simulation for the given parameters and the ChIP-seq signal for the region.

Results

Simulation: The current work extended the model presented in (Zabet, N. R., and Adryan, B. 2012c). The strategy of the model was to simulate stochastically the facilitated diffusion performed by TFs. The model considers the DNA as a string of nucleotides. We added a new feature to include 3D chromatin organisation from Hi-C data and the program has been adapted for the molecules to interact in such an environment by performing intersegmental jumps here 3Dhops. This does not however include the intersegmental jump across chromosomes.

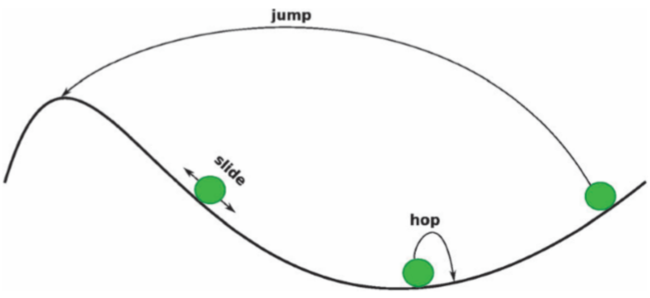


Figure 3. The actions that a TF molecule can perform while bound to DNA. Figure presented in (Zabet and Adryan 2012b). Green dot represents the molecule while the black line the string of DNA.

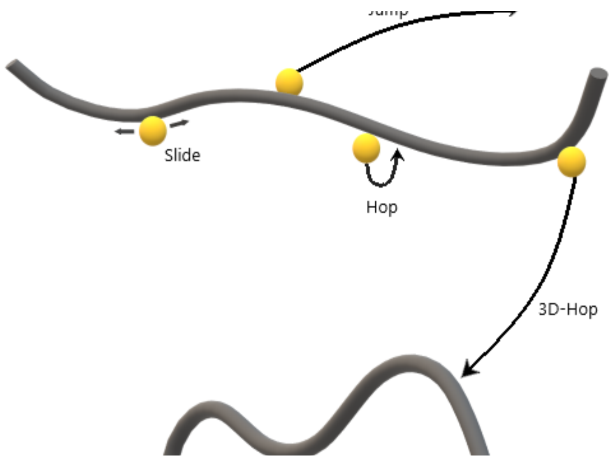


Figure 4. The actions that a TF molecule can perform while bound to DNA. Yellow dot represents the molecule and the two black lines two strings of DNA that come in close proximity.

The facilitated diffusion assumes that the DNA-binding proteins can perform three actions while bound to DNA, namely: sliding-moving across the DNA for a short BP distance; hopping-partially detach from the DNA for a short distance and reattach back; jump- completely detach from DNA into the microenvironment; The actions have been graphically represented in Figure 3. The main scope of introducing the 3-dimensional structure of DNA brings on a new type of interaction that the molecules could perform while bound to the genetic material. The new action that can now be performed in the presence of two or more spatially proximal segments was introduced in the model as '3D-hop' or '3-dimensional hop' (Figure 4). Furthermore, the mechanisms behind each concept are outlined as follows:

1. Simulating the 3D space from Hi-C dataset (testing was done using the Hi-C dataset from BG3 cell line); 2. Dynamic simulation implementation; 3. Implementation of 3D-hop. Detailed content of the methods created in each step and described here accordingly can be found in appendix Table 1 as well as additional methods used.

1 Simulating the 3D space from Hi-C dataset (testing was done using the Hi-C dataset from BG3 cell line (Chathoth, KT. and Zabet, NR., 2019)): The class constructed in Java created a 2D matrix that stored the interaction scores between bins (default 500 bp length bins) using the Hi-C data from BG3 cell lines. A method was created for reading the file, extracting the data and adding the score to a local contact matrix. The DNA region of interest to be simulated can be specified if not the whole length of the DNA region would be simulated. Furthermore, the class contains a method that normalizes the contact matrix score values with a range between 0 and 1; the normalized values would be then simulated, meaning, a method would draw a random number between 0 and 1, if the normalized value between two bins is higher that the random number the interaction between two bins would be true and the data would be stored in the simulation matrix (1). Where X is the generated random number and Y is the normalised Hi-C value between two bins.

(1) $For\ X, Y \epsilon [0,1]\ if\ X > Y\ \rightarrow simulate\ interaction = TRUE$.

The method that creates the simulation matrix is public meaning that the matrix can be simulated from other classes (further reading: 2. Dynamic simulation implementation). Other methods that are used to access data in this class are as follows:

a) Random bin position generator: generates a position in a specified bin (this method is used in creating a new position on the specified region for the TF molecule to hop to-used for 3D hop only);

b) Cumulative result: returns the cumulative simulation time between two bins (The cumulative time two bins were in interaction);

c) interacting bins: for a given bin, the method would return an array of bins that are found be interacting the given bin (when 3D hopping, a molecule would choose one of the interacting bins to jump to);

d) get current bin index: returns the index of the current bin in which the molecule is found. The method checks in which bin of the array of bins the given position of the TF is;

2. Dynamic simulation implementation: A more realistic approach when it comes to the simulation of the 3D structure of the DNA would be to have a dynamic simulation. A dynamic simulation implies that two segments that are now interacting might move away from each other. In this regard, a simulation event class was constructed to keep information regarding the time at which the matrix will be simulated again (calling the method in the Contact Matrix class as explained above). The cell stores each event (e.g. jump, hop, simulation of the Hi-C matrix) in the form of a queue of events. The queue of events mainly contains actions that will

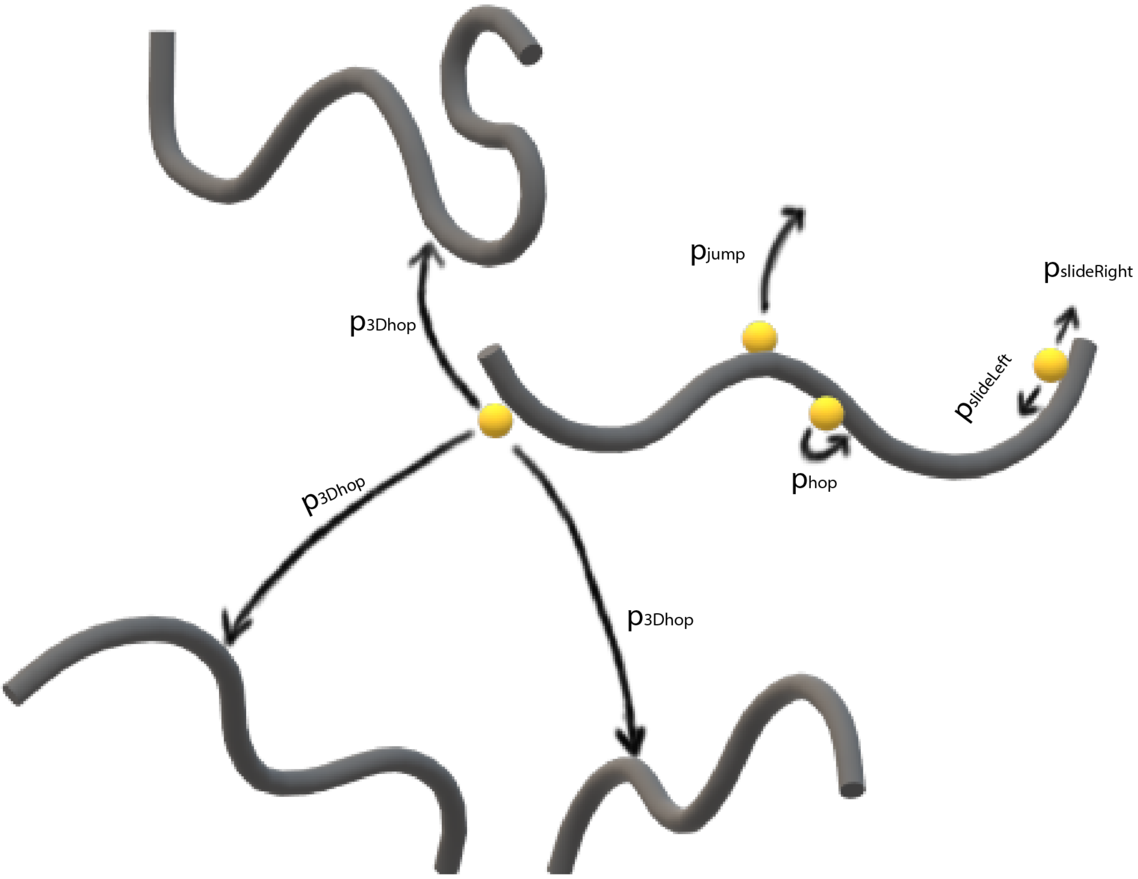


Figure 5. The actions that a TF molecule can perform while bound to DNA. Yellow dots represent the molecule and the black lines represent strings of DNA that come in close proximity. Each probability is assigned to the possible action the protein can perform.

be performed next by the protein and the dynamic simulation has been added as an event in this queue. The first matrix simulation will generate a value for the next time the matrix should be simulated again using the exponential distribution (2) $exponential\ sample = -\frac{1}{\lambda}\log e\,(1 - X)$ where λ is the average time simulation and X is a random number between 0 and 1 generated by the program. This value plus to current time of the cell will give the time for the next time the 3D map will be updated and stored in an array. When the cell time will be equal to the next appointed time of simulation, the matrix is simulated again giving a dynamic environment in which the proteins act. The approach is based on the Monte Carlo simulation for cellular processes by using the general method, the Gillespie algorithm (Gillespie, 1977).

3. Implementation of 3D-hop: Each molecule has the ability to act, meaning it would choose stochastically an action from sliding to hopping to jumping; Additional to this we have introduced a new action: When the next action for the TF molecule to perform is chosen, if the action is a jump action (as jumps can include intersegmental transfer of TFs from one DNA segment to another), again a random number between 0 and 1 is chosen. If the random number is lower than the value of the protein's probability to be kept in microenvironment (PKrate), the program generates the new position on which the TF will jump to on one of the bins that interacts with the current bin. The new position is added, and the action performed is the 3D hop (Figure 4). The probability to keep in the microenvironment is used to determine whether the action performed will be a 3D hop or a jump. In other words, if the protein situates in the value of the probability it will stay in the microenvironment by attaching to a close fragment. In the long-range excursion each DNA site that comes in close proximity with the fragment of DNA on which the protein is located has an equal likeliness to have the re-association point (Figure 5). The number of 3D-hops is outputted in the file containing the number of all the other actions and the time that each two segments stay in interactions is added in a cumulative matrix and outputted in a cumulative matrix file.

Data analysis:

Evaluation of the simulation of 3D contacts from Hi-C dataset: The first step in the analysis was to see how well the simulated matrix represents the actual Hi-C data (Chathoth, KT. and Zabet, NR., 2019). The dynamic state of the 3D map is performed by the program by updating the matrix periodically during the simulation. One parameter that needs to be set to draft such an environment for the molecule, is the average interval when the 3D map needs to be updated. We used 10 different averages: 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10; drawing a sample from the exponential distribution of the average, the next time the map is updated is set as the current time of the cell plus the sample value. We analysed each simulation time by producing histograms of log base 2 of the times at which the matrix was simulated against the Hi-C dataset. One example can be seen in Figure 6. The model was designed, more specifically the addition of the 3D space in a dynamic state is controlled for each bin. Let's say we have the Hi-C score between two interacting bins and the score is normalised so that it will not be higher than 1; if the program draws a number between 0 and 1 the chances of it being lower than the Hi-C score depends on the score, the higher the score the higher the probability that those bins will be interacting each time the map is updated. Following this, we took the cumulative matrix file which stores for each bin the amount of time it was interacting with other bins and put it against the Hi-C score for the specific interaction. In Figure 6 the most frequent values for the log base 2 of the difference between the simulated interaction time and HiC scores were 0 or very close to it (right-BG3 cell histogram). We took KC167 Hi-C dataset and plotted

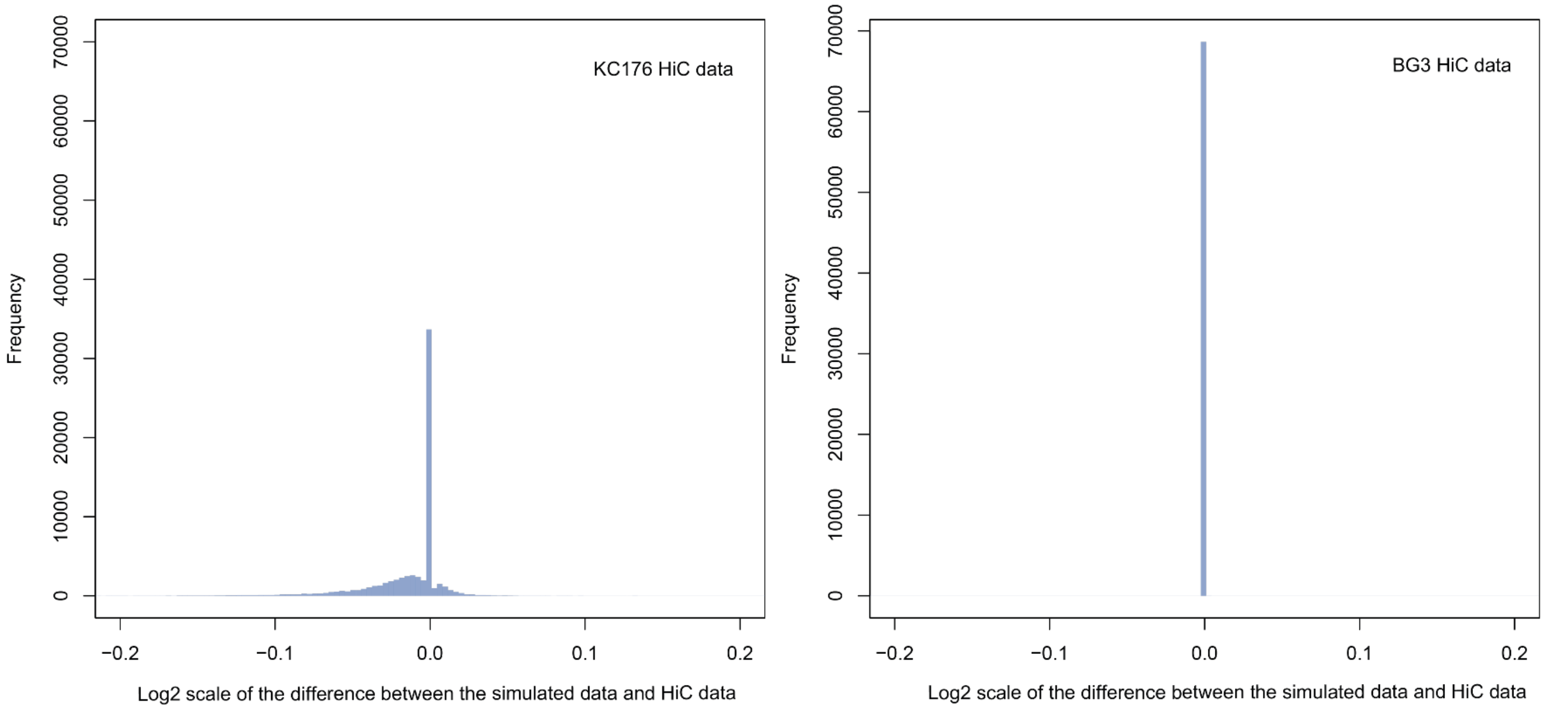


Figure 6. histograms of BG3 cell (right) and KC176 cell (left) logarithmic values of the difference between the time segments are interacting and HiC score for that interaction. The logarithm value measured the time points of the matrix simulation against Hi-C data collected for the BG3 cells (right) and for time points of the matrix simulation against Hi-C data collected for the Hi-C data from KC167 cell line (left).

the times of interaction between bins (from the simulation performed for BG3 cell line) against the Hi-C from KC167. Here the frequency values of the log base 2 spread across the x axis.

The score of the simulated matrix was similar for all the different values of the average simulation time. While the higher the parameter's value, the higher the number of times the 3D map is updated, it also showed increased time of the computational simulation. As a result, we used the average simulation time that required the matrix to be simulated at the lowest rate. Using the lowest value that simulated the matrix less frequently, Figure 6. shows the high similarity between the simulated matrix and the real Hi-C score recorded for this locus (right).

Presence of nucleosomes affects the number of observed 3D hops:

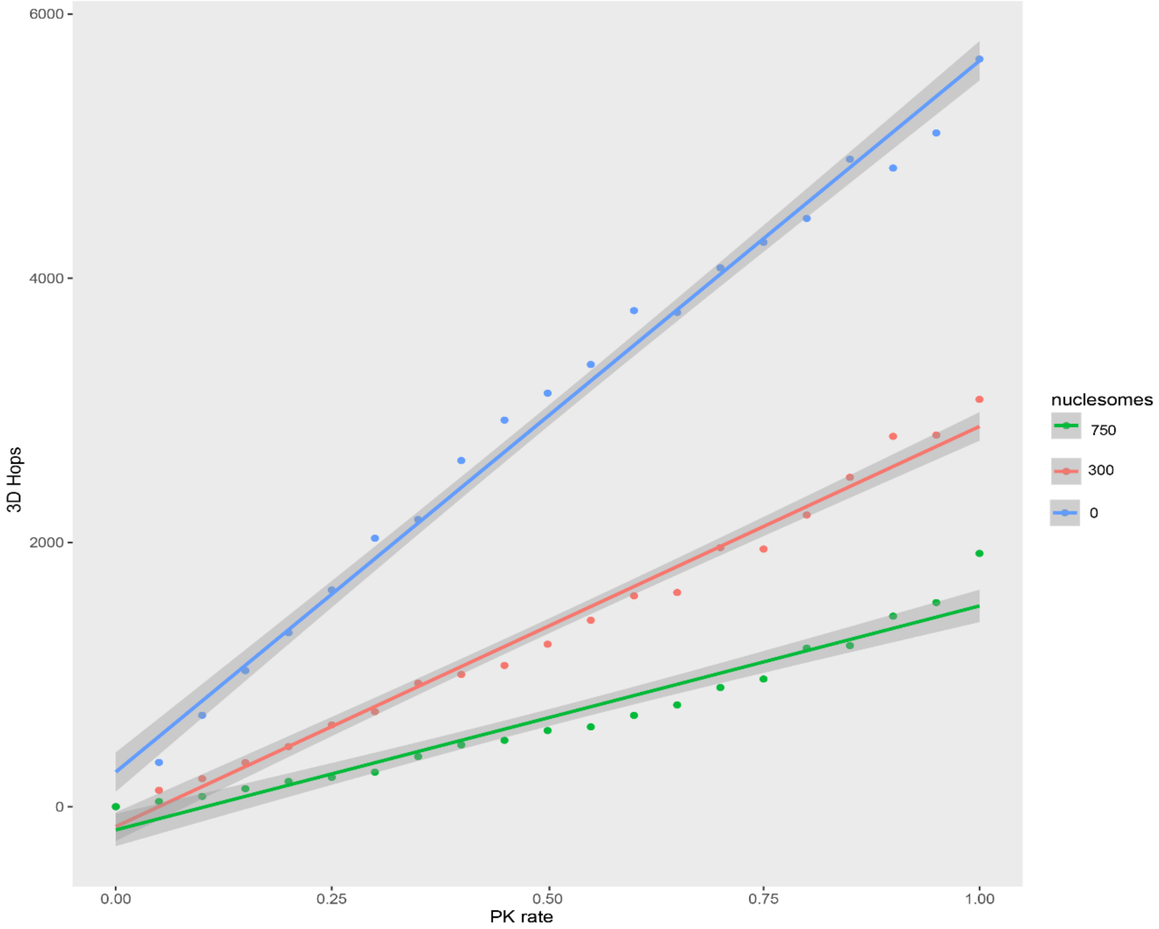


Figure 7. Data analysis of the number of 3D hops recorded when 750, 300 and 0 nucleosomes were present and (PKrate) probability to keep in microenvironment varying between 0 and 1. When the PKrate is increased and the number of nucleosomes is low, the number of intersegmental jumps increases; The fitted lines for 0, 300 and 700 nucleosomes present show an upward trend where the number of 3D hops increases with increasing the PKrate and decreasing the number of nucleosomes. The confidence interval is narrow for all fitted lines showing little variation from the trend in the number of 3Dhops performed. The sampling size for each trend line is equal to 20 samples (20 varying PKrates) each sample is an average of a simulation with a 1000 Ensemble size. The ensemble size of 1000 was used to simulate a wider cell population.

The enhanced GRiP model, now including the 3D spatial structure of the genetic material, was tested with all possible combinations of the following parameters: average time interval at which the simulation of the matrix to take place: 0.01, 0.02, 0.05, 1, 2, 5, 10; the probability of the molecule to be kept in microenvironment if jumping: from 0 to 1 with an increment of 0.05; the number of nucleosomes introduced: from 0 to 828 with an increment of 50. The data collected from simulations at 10 different matrix simulation times and 21 different PKrates (probability of the protein to be kept in the microenvironment) were processed and they show the impact of the 3D DNA architecture on the number of intersegmental jumps (3D hops) that a protein is performing in various conditions (Figure 7). The number of nucleosomes and the PKrate influence the number of 3D hops performed. The number of nucleosomes shows a high

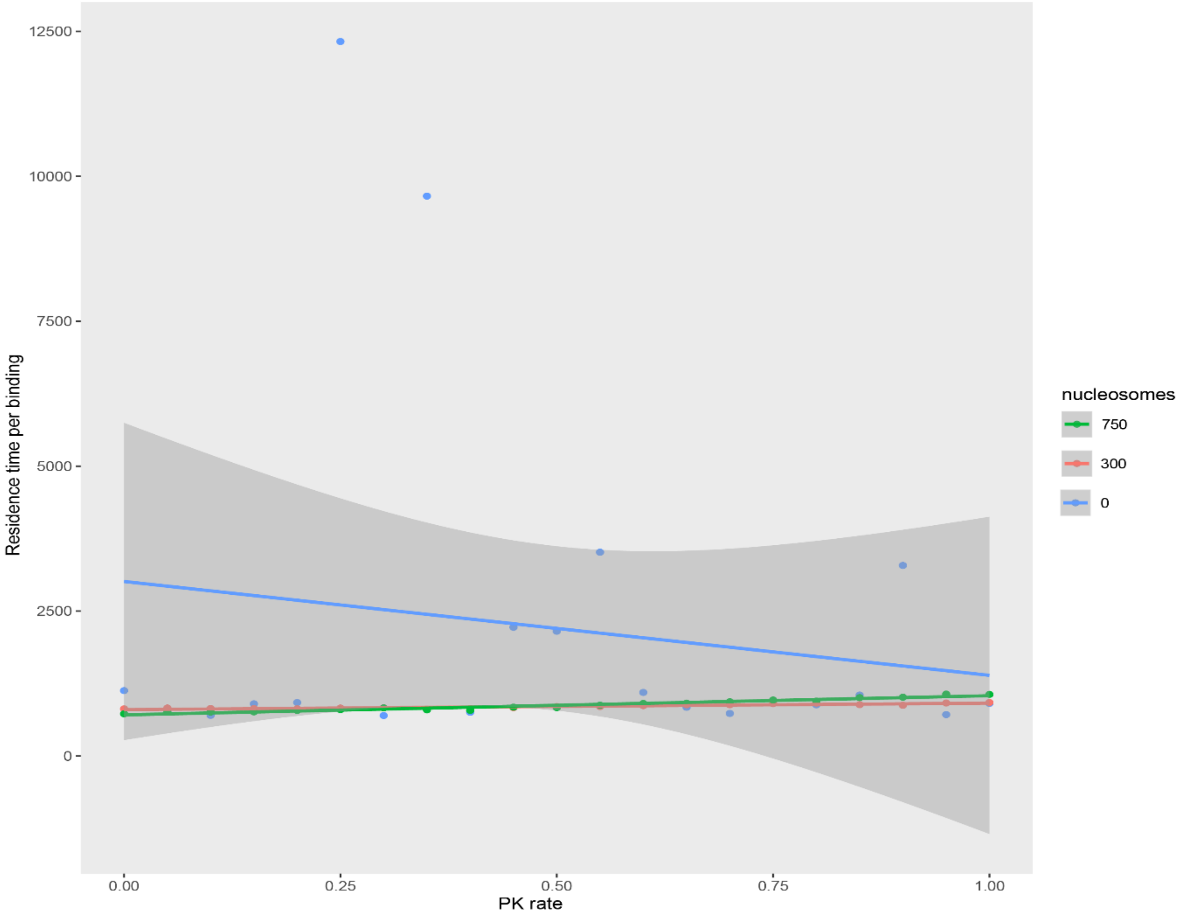


Figure 8. Data analysis of the residence time per binding for the BG3 molecule species recorded when 750, 300, and 0 nucleosomes were present and (PKrate) probability to keep in microenvironment varying between 0 and 1. At 0 nucleosomes present (red) the residence time has the highest decreasing value in residence time; while increasing the number of nucleosomes results in higher residence time. The fitted line for 0 nucleosomes present shows a downward trend where the residence time per binding decreases when increasing the PKrate. The fitted lines for 300 and 750 nucleosomes being present (red and green) show a horizontal trend. The residence time per binding stays the same for all PKrates. The confidence interval is wider for 0 nucleosomes fitted line showing increased variability in the residence time when there are no nucleosomes present. The sampling size for each trend line is equal to 20 samples (20 varying PKrates) each sample is an average of a simulation with a 1000 Ensemble size. The ensemble size of 1000 was used to simulate a wider cell population.

influence on the number of 3D hops and if one decreases the PKrate, it would also result in decreased number of intersegmental jumps. Meaning, the more nucleosomes present, the less unoccupied DNA, thus lower chances for the protein to find available site on the adjacent segments for the protein to jump to, while lowering the number of nucleosomes would result in more available DNA space to diffuse on. In the case of the PKrate, increasing it would mean that the molecule would have greater chances to stay in the microenvironment by binding to a spatially proximal DNA segment. This means that during a detachment from the genetic material, the protein has a higher chance to bind to a neighbouring site.

Molecules spend a certain amount of time bound to the DNA and have a wide range of parameters influencing this (see methods). In the file output, the time the protein stays bound is recorded for each simulation under a 'residence time per binding' column. In Figure 8 the

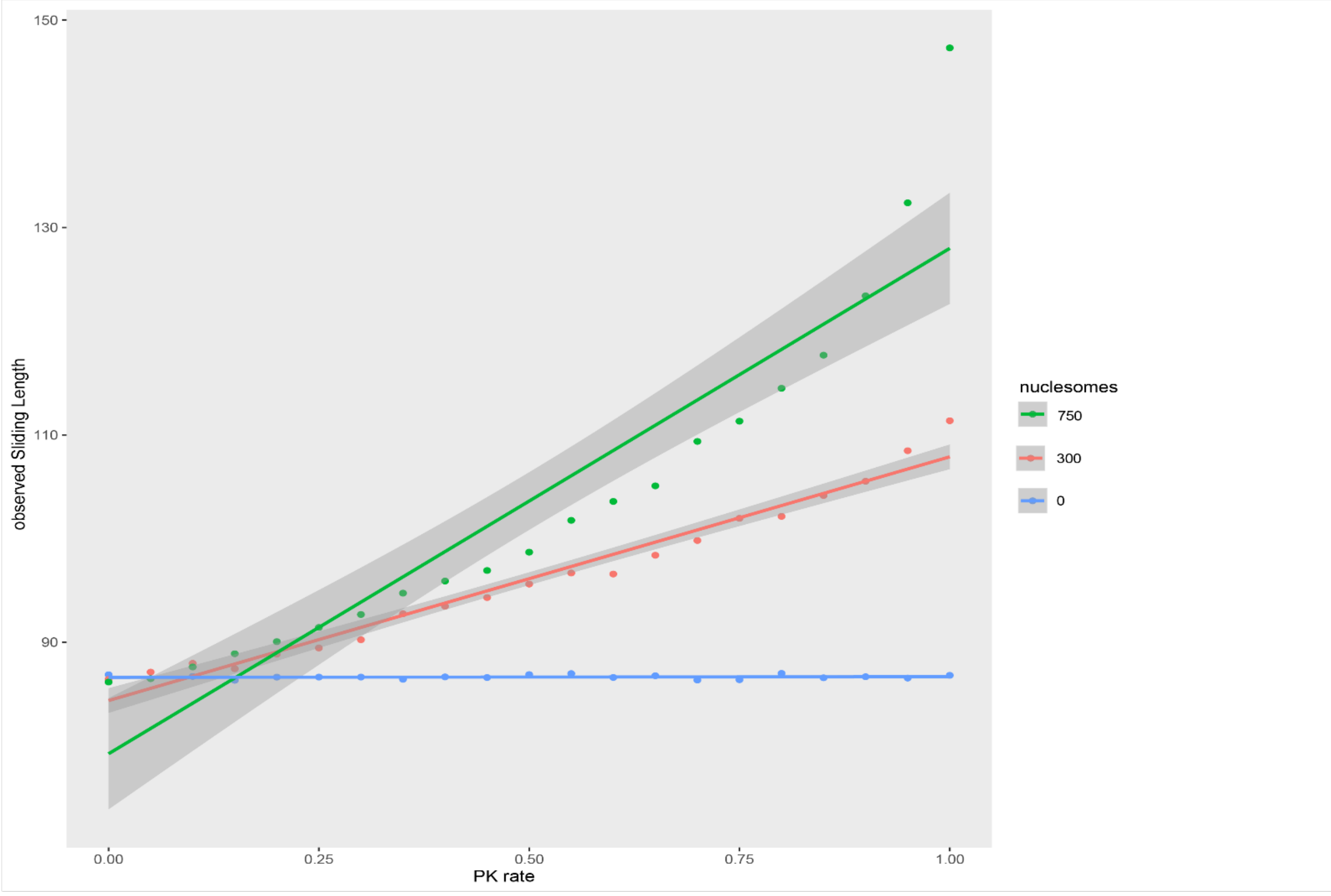


Figure 9. Data analysis of the observed sliding length for the BG3 molecule species recorded when 750, 300 and 0 nucleosomes were present and (PKrate) probability to keep in microenvironment varying between 0 and 1. Increasing the number of nucleosomes and PKrate resulted in increased sliding length as a consequence of resampling of the same site. The fitted line for 0 nucleosomes present shows a horizontal trend where the observed sliding length stays the same when there are 0 nucleosomes present. The fitted lines for 300 and 750 nucleosomes being present (red and green) show an upward trend where the observed sliding length increases with increasing the PKrate. The confidence interval is wider for 750 nucleosomes fitted line showing increased variability in the observed sliding length. The sampling size for each trend line is equal to 20 samples (20 varying PKrates) each sample is an average of a simulation with a 1000 Ensemble size. The ensemble size of 1000 was used to simulate a wider cell population.

residence time per binding is plotted against the PKrate and for 3 sets of nucleosomes; At the 750 nucleosomes present the residence time is higher than when 300 or 0 nucleosomes are present, this would be the case for a high PKrate. This suggests that the molecules are detaching less from the DNA when more nucleosomes are present. When the PKrate is decreased however, the residence time increases when there are no nucleosomes present and Figure 8 shows lower a lower amount of time is spent at the site when there are more nucleosomes for a low PKrate. A plausible explanation for a high residence time when using a high PKrate would be that the molecule has less space to perform actions that would make the protein semi-/completely detach from the DNA and binding in a different region as the chances of finding a free space to bind would decrease. In the absence of other possible moves, the TF would stay attached and perform more actions that would require it to stay bound to DNA such as sliding.

Data analysis of the observed sliding length (Figure 9) shows that the sliding length of the TF reaches higher values the more nucleosomes are added. As stated above, increasing the number of nucleosomes would indicate that the protein cannot perform actions that require moving a longer distance such as 3Dhops (or the number of those movements decreases e.g. Figure 7;) as the 3D view/space is obstructed by the roadblocks, in this case the nucleosomes. In Figure 9 we plotted the observed sliding lengths against The PKrate. The trend of the slope would increase for PKrate and the nucleosomes addition would in its turn increase the sliding length. For a protein that is performing its actions in the presence of 750 nucleosomes at a PKrate of 1, the amount of possible actions decreases. In the absence of jumping, hopping and 3D hopping as well as massive roadblocks that the protein would have to pass, figure 9 suggests that the protein in this situation would resample the same site multiple times.

Metrics: The performance of the simulations was analysed using the Pearson correlation (Figure 10), the AUC(Figure 11) and MSE(Figure 12) scores. The correlation score was higher for the simulation profile where higher number of nucleosomes and higher pk rate was used. Similar for the AUC (Figure 11) the score was higher between profiles the more nucleosomes the simulation used and when the protein had a higher PK rate.

The MSE (Figure 11) showed increased values for simulations where fewer nucleosomes were present and where the PK rate was low. Indicating that the similarity between the real ChIP profile and the simulated profile is increasing when the no. of nucleosomes and PKrate value is increased.

In figures 13 and 14 can be seen the occupancy profiles of the simulations and ChIP-seq data in the presence (Figure 13) and in the absence (Figure 14) of nucleosomes. The occupancy profile of the nucleosomes was used in figure 14 to generate the grey lines. The profile was generated as an output file of the simulation. In the absence of nucleosomes (figure 13), the occupancy profile did not show peaks at the known target sites. In this case, the ChIP seq and of the occupancy the profiles do not show any similarity. In figure 14 however, when we added the nucleosomes the peaks of the occupancy profile are more 'organised', and the occupancy

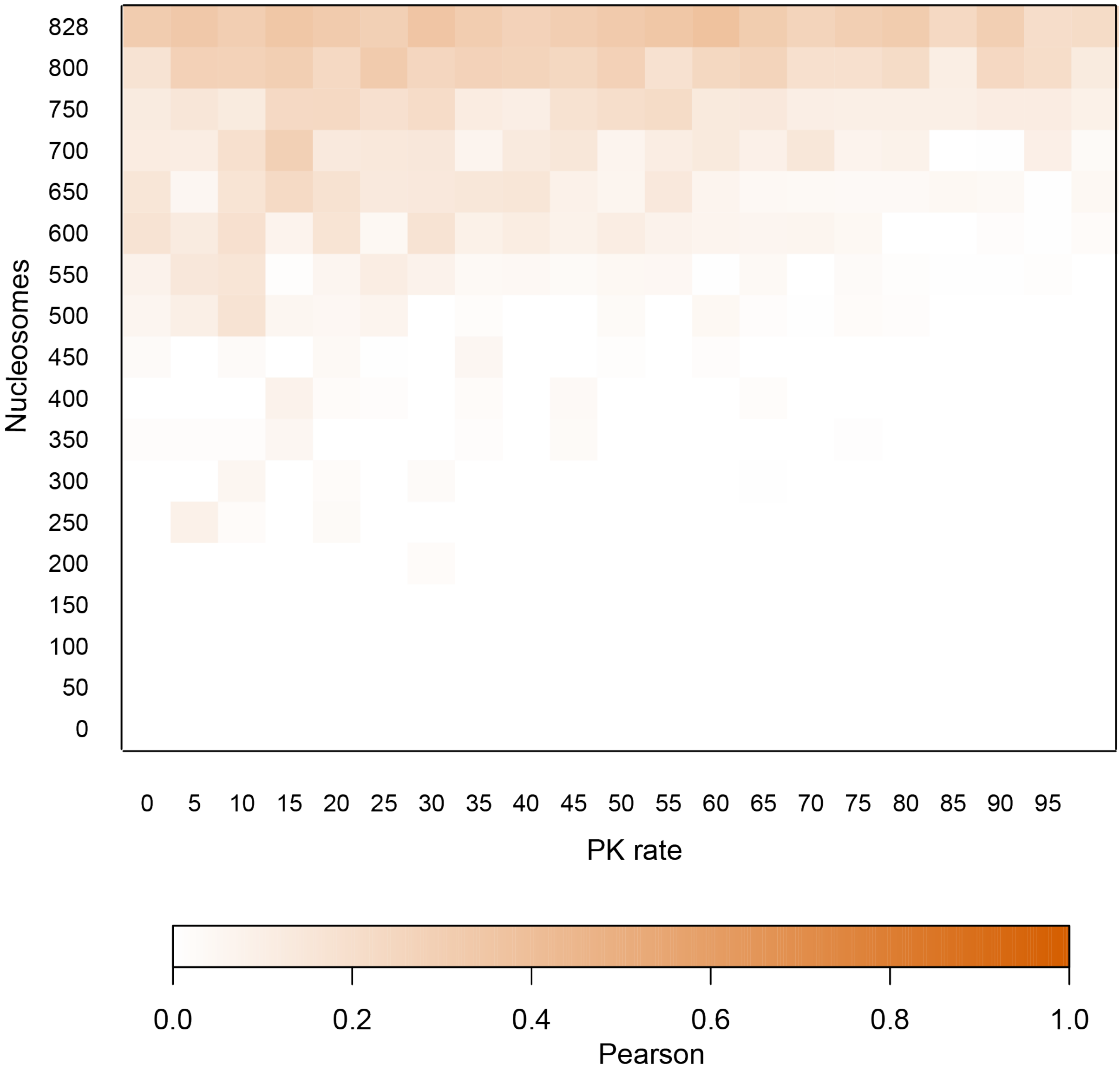


Figure 10. Heat map of the Pearson correlation measured for the predicted occupancy profile by the simulation for BG3 molecule species and actual ChIP-Seq profile. The simulations were run with 17 different no.of nucleosomes and (PKrate) probability to keep in microenvironment varying between 0 and 1. Increasing the number of nucleosomes and PKrate resulted in increased correlation between the profiles.

profile seems to follow the pattern of the ChIP-seq profile in BG3 cell line. The peaks in Figure 14 are present where there are no nucleosomes occupying the sites. The program simulated the positions at which the nucleosomes would bind upon adding them using the affinity profile of the nucleosomes (see methods: Data Analysis) and the number of nucleosomes to be present.

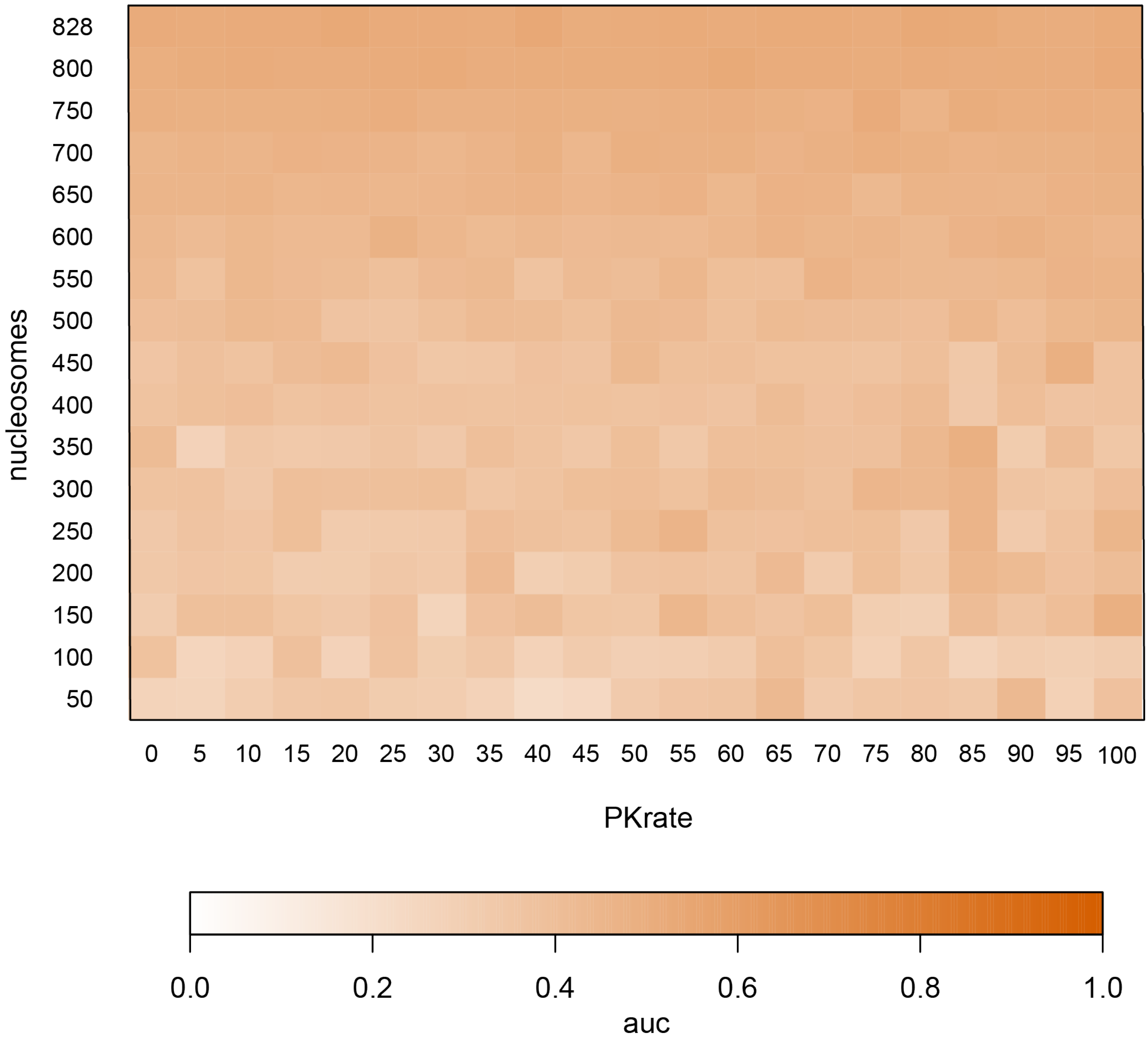


Figure 11. Heat map of the AUC measured for the predicted occupancy profile by the simulation for BG3 molecule species and actual ChIP-Seq profile. The simulations were run with 17 different no.of nucleosomes and (PKrate) probability to keep in microenvironment varying between 0 and 1. Increasing the number of nucleosomes and PKrate resulted in increased AUC values.

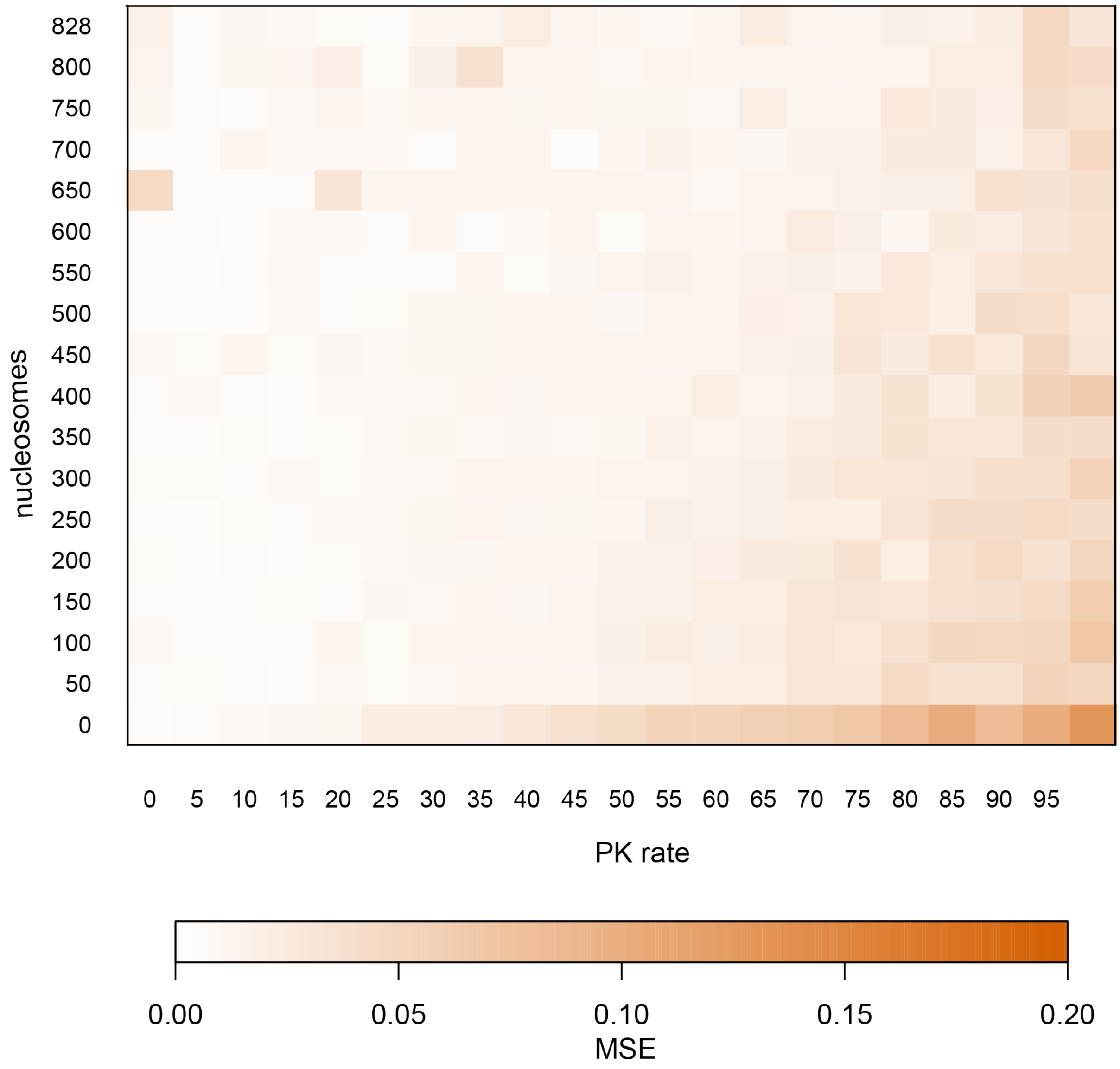


Figure 12. Heat map of the MSE measured for the predicted occupancy profile by the simulation for BG3 molecule species and actual ChIP-Seq profile. The simulations were run with 17 different no.of nucleosomes and (PKrate) probability to keep in microenvironment varying between 0 and 1. Increasing the number of nucleosomes and reducing the PKrate resulted in reduced mean squared error between the profiles.

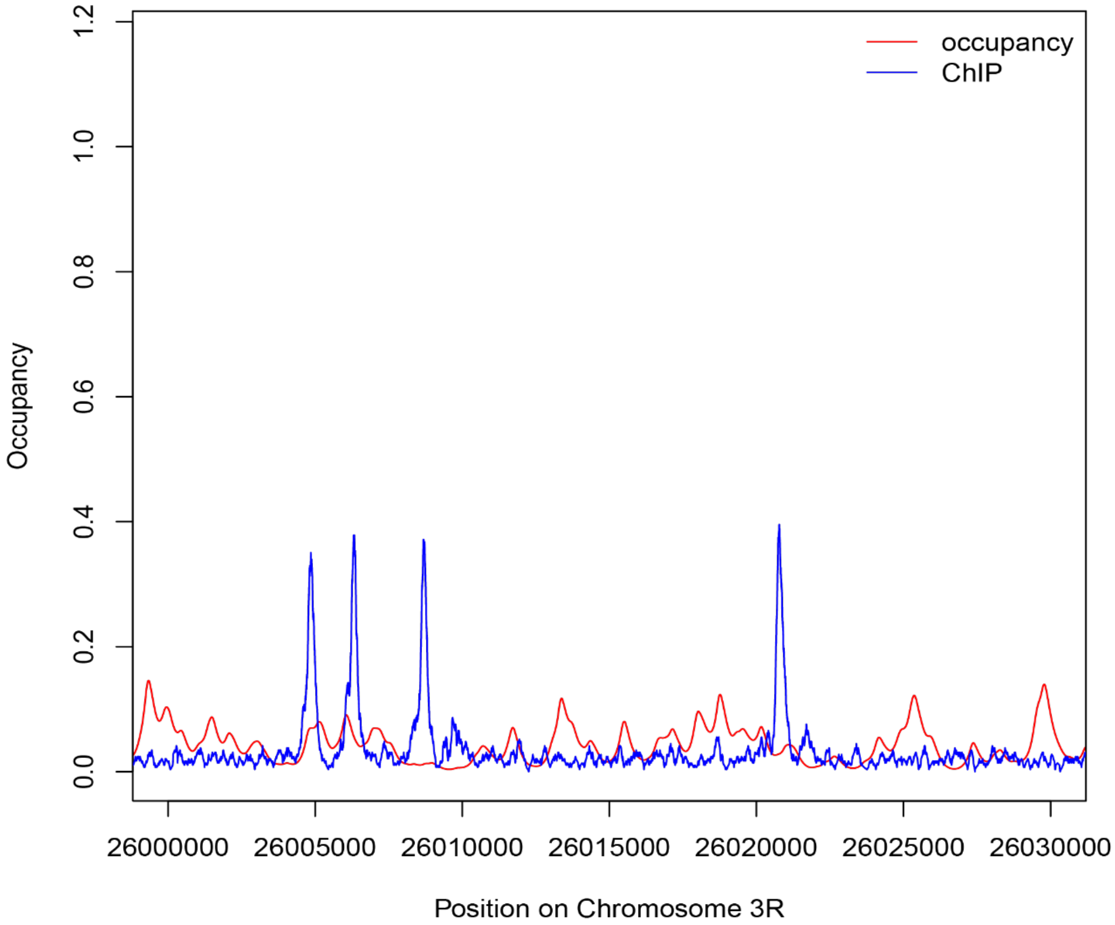


Figure 13. Occupancy of SuH cell line on chromosome 3R D. Melanogaster. The Blue line is the experimental ChIP-seq profile, and the red line is the simulated occupancy profile. The region for the Chromosome positions is reduced from the original 25951000-26082000 to 26000000-26030000 of chr3R to make it easier to compare the difference at the peak regions between the ChIP occupancy and the simulated occupancy for the given parameters. Here there were no nucleosomes added to highlight the variance across the simulated occupancy when compared to the the ChIP profile.

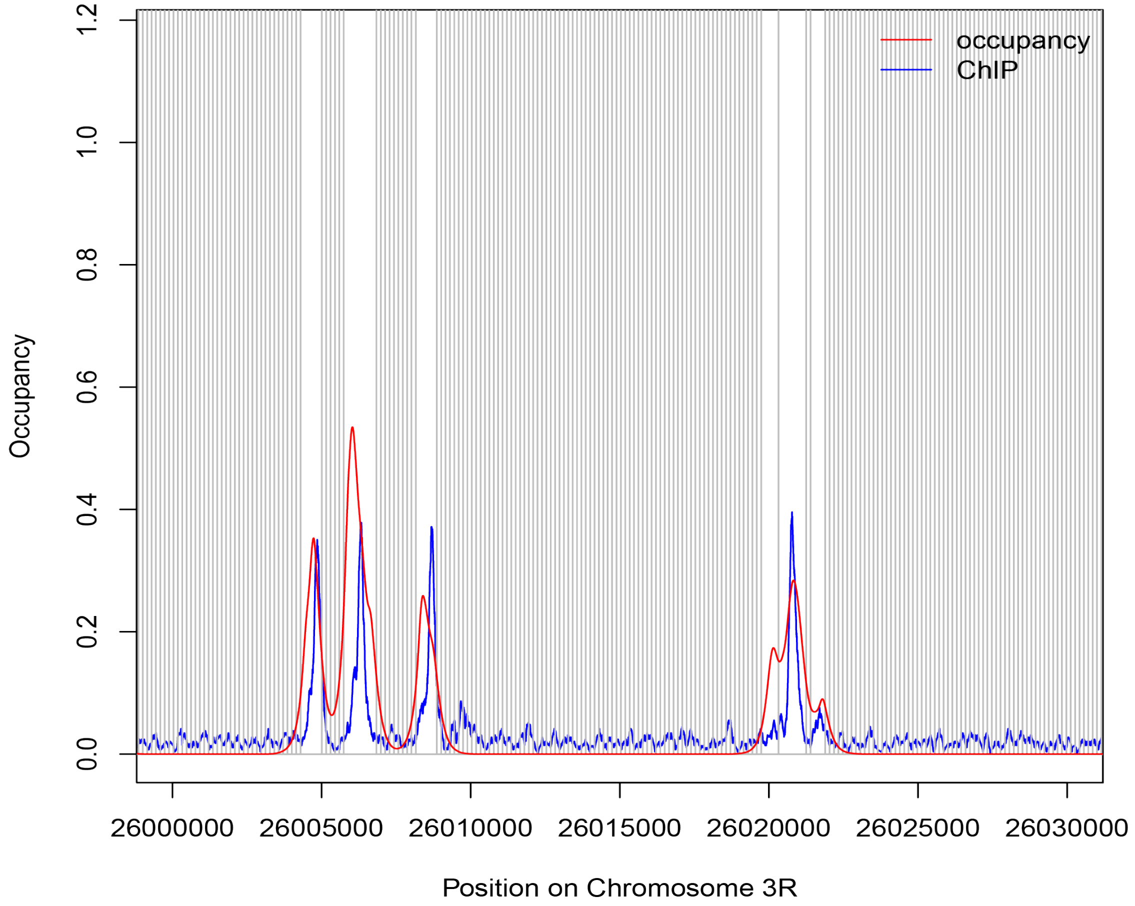


Figure 14. Occupancy of SuH cell line on chromosome 3R D. Melanogaster. The vertical grey lines represent the nucleosomes attached at the specific position from nucleosome affinity profile that was taken from the output file generated by the simulation together with the occupancy profile of SuH. The Blue line is the experimental ChIP-seq profile, and the red line is the simulated occupancy profile.

Discussion

The previous model presented in (Zabet, N. R., and Adryan, B. 2012c) was subject to changes in this work introducing a new approach on the facilitated diffusion using an implementation of the 3D architectural structure of the DNA with Hi-C data. When two DNA fragments come in contact, the TF molecules can perform an intersegmental jump from the segment they are found on, to the one the current segment is interacting with. This action can be performed even if the site is hundreds of base pairs apart due to the two segments being spatially close.

In theory, not performing intersegmental hops would slow the search time of the molecules. Thus, the number of bound nucleosomes would influence the rate at which the TFs find the target sites. For instance, when there is no space on the adjacent segments, the molecule will not perform a 3D hop. There are number of scenarios in which the protein can be when attempting to perform a 3D hop. First, it would try to distinguish between a jump and 3D hop, if the protein has a high probability to stay into the micro space, it will be more likely that the TF will 3D hop rather than dissociate into the cytoplasm. In the next step the TF will look for all the segments/bins that are interacting with the region it is found on, if there aren't any available, it will simply detach from the site of the DNA and perform 3D search. If, however, there are segments, the molecule will randomly choose one of them with equal likeliness. When it selects the interacting 3D bin, the TF will choose a position to attach to and it will check the availability of that site. In the case of the position is not accessible, the TF will stop performing a 3D hop and will dissociate. The 3D hop action thus has the protein choose whether to jump or 3Dhop, search for the interacting bins and the selected position accessibility (looking at the accessibility before selecting a bin and position could be an idea for further study and to see whether the protein would check the accessibility or if this could introduce bias). Upon successfully passing all the steps the protein will finally attach to the selected site by performing the intersegmental jump.

When the TF performs a 3D hop, the molecule has to detach completely from the position it is on the DNA fragment. With this is mind, the model has the protein detach by performing a jump. The jump would further start a cascade of events that will have the protein search for nearby segments and find a place to attach on one of them.

A number of factors (here discussed and analysed) can affect the number of 3D hops the protein performs. Among them would be the probability that the protein will stay in its microsystem and the number of nucleosomes present that could, if in great numbers, obstruct sections of possible DNA sites that the protein could jump to. Increasing or decreasing the rate at which two segments come in contact (the dynamic environment) does not however affect the performance of 3D hops. When we look at how the latter two factors that affect the rate of 3D hopping, decreasing the number of nucleosomes and increasing the PKrate resulted in increased intersegmental jumps. The lack of hops across DNA segments could be due to nucleosomes covering possible binding sites for the TFs. Increasing the no. of nucleosomes would decrease the available DNA space and increasing the PKrate would give a higher probability to the protein to rebind in the close proximity (aka perform the 3D hop). Thus, in figure 7, the highest number of recorded intersegmental jumps was in the absence (nucleosomes no.=0) of nucleosomes (as any selected position on the interacting region was empty) and at the highest probability that the protein stays in the microenvironment (PKrate=1).

We also attempted to look at how the 3D hops are influencing other characteristics of the facilitated diffusion such as the time the protein spends attached to DNA and how it affects the observed sliding length across the simulation. The protein would spend more time bound to DNA when the number of nucleosomes is 750 and the PK rate is closer to 1, this may be the result of the protein being unable to perform hops and 3D hops as the regions are covered by roadblocks. The time spent on the DNA is also high when there are 0 nucleosomes and a PK rate closer to 0 due to the protein being unable to perform 3D hops and being forced to perform only sliding across the DNA.

The analysed performance of the simulations against the ChIP seq profile of the sequence was higher when the number of nucleosomes and the PK rate was increased which can be seen in figure 10 and 11 for the correlation and AUC and the MSE in figure 12 was higher for lower number of nucleosomes and pk rate. This indicates that the program simulates the profile better when there are more nucleosomes and when the protein has a higher probability to stay in the microenvironment. The occupancy profiles for the simulations where 0 nucleosomes and 828 nucleosomes were used are added in the Appendix.

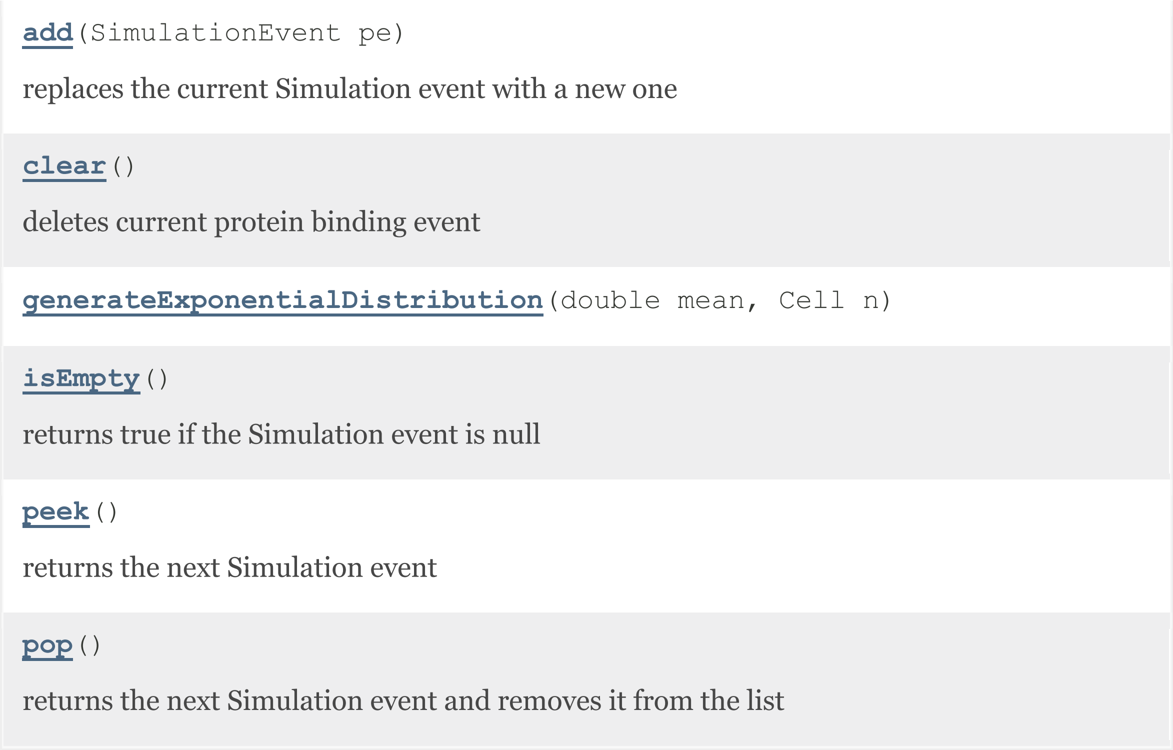
Furthermore, the simulation of the matrix was tested against its real data collected from BG3 Hi-C to evaluate the implementation of the simulation and how similar the two cell lines are, the closer the log value was to 0 the higher the similarity and suggests that the program simulates the matrix for BG3 well (Figure 6). In order to check if the simulation is true to this specific cell, we tested the simulation matrix against real data from KC167 Hi-C. To evaluate if the simulated data is specific to the Hi-C dataset we used as input, we also evaluated how different it is against the simulated matrix to Hi-C dataset in a different Drosophila cell line (Kc167) (Chathoth, KT. and Zabet, NR., 2019) Kc167 cells have been shown to have a different 3D organisation to BG3 cells and our simulation reflects that. The results showed increased variability across the x-scale, showing that the matrix is specifically simulated for the provided data and varies when compared to other cell lines HiC data.

Further study: Here the simulation was a time average performed with a stop time at 10^8 simulation time for one cell. As the ChIP-seq profile of the sequence used is an ensemble average time for hundreds of cells (Robertson, G; et al., 2007), the correlation between the two profiles (predicted and the actual) can be affected. One other factor that could be further explored would be the presence of other TF species. Other molecules could be performing their facilitated diffusion either by random walk or 3D diffusion in the 3D space where the target sites (here the ESPL locus for SuH) of the studied TF species are found (Zabet & Adryan, 2012; Zabet et al., 2013). The presence of other species could affect search of the TF by reducing the search at redundant, non-specific sites.

Appendix

Table 1. Methods created and used with the purpose of integrating the 3D structure of the DNA. Each method used has the name specified alongside the input parameters and the action/result of using the specific method.

Method and Description
addScoreToMatrix(DNAregion regionOfInterest, java.lang.String ContactMatrixLocation) reads the file of interest and adds the values from it to a local file the local file is read and the values for local arraylists for bins and score are added
addValueToCumulativeSimulatedMatrix(double newTime) Iterates through the simulated matrix and if the bins are interacting the cumulative matrix will give the amount of time the bins were interacting
createMatrix(DNAregion regionOfInterest, int binWidth) creates the empty matrix using the DNA region of interest
CumulativeResult(int t, int b) method that returns the cumulative simulation time between two bins
find(int BinX, int BinY) prints out details regarding 2 bins and the score between them as well as the bins ID
getBin(int index) gets the bin's details based on its index
getBinSize() gives the size of the bins list
getBinsList() Accessor method for the arraylist 'bins';
getCurrentBinIndex(int position, java.lang.String chromosome) Method that searches the bin in which a certain position is in
getInteractingBins(int bin) takes the input parameter and searches against the existing arraylist of bins to get the interacting bins
radomBinPosition(Random generator, int randomBinID) method that generates a random number between the values of the start and end of a bin
simulateMatrix(Random generator, double time) Simulates the Interaction Matrix Can be accessed and simulated from other classes to have a dynamic simulation matrix



add(SimulationEvent pe)

replaces the current Simulation event with a new one

clear()

deletes current protein binding event

generateExponentialDistribution(double mean, Cell n)

isEmpty()

returns true if the Simulation event is null

peek()

returns the next Simulation event

pop()

returns the next Simulation event and removes it from the list

Table 2. Molecule parameters

name	SuH	nucleosome
COPYNUMBER	1	0-828 (with an increment of 50)
SIZELEFT	1	0
SIZERIGHT	1	0
ASSOCRATE	20	4.06
UNBINDINGPROBABILITY	0.001474	1
SLIDELEFTPROBABILITY	0.499263	0
SLIDERIGHTPROBABILITY	0.499263	0
JUMPINGPROBABILITY	0.1675	1
HOPSTDDISPLACEMENT	1	1
SPECIFICWAITINGTIME	1.5	3600
UNCORRELATEDDISPLACEMENTSIZE	5	5
STALLSIFBLOCKED	true	true
COLLISIONUNBINDPROBABILITY	0	0
AFFINITYLANDSCAPEROUGHNESS	0	0
PREBOUNDPROPORTION	0	1
PREBOUNDTOHIGHESTAFFINITY	true	true
TFISIMMOBILE	false	true
ISBIASEDRANDOMWALK	false	false
ISTWOSTATERANDOMWALK	false	false
PKMICROENV	0-1 (with an increment of 0.05)	0.75

Table 3. Default parameters

#SIMULATION PARAMATERS
#The length of the simulation (in seconds). If lower or equal to zero and there are target sites to be reached by TFs then the simulation will stop when all target sites are reached.
STOP_TIME = 10000.0;
#The number of independent replicate simulations to be performed.
ENSAMBLE_SIZE = 1000;
#The seed of the random number generator. Use 0 to get a different behaviour each time or different number to get the same behaviour.
RANDOM_SEED = 0;
#The number of decimals when computing the TF affinity.
COMPUTED_AFFINITY_PRECISION = 2;
#The size of the DNA sector. Breaking the DNA into sectors increases the speed at which empty spots on the DNA are located. Put 0 for autoselect.
DNA_SECTOR_SIZE = 0;
#The event list is broken into sub-lists of the specified size. This is highly recommended for Direct Method and should not be used for First reaction Method. Put 0 for autoselect.
EVENT_LIST_SUBGROUP_SIZE = -1;
#This is true if the 1D event list is implemented using the First Reaction method or false if the Dirtect Method is used (Gillespie 1977).
EVENT_LIST_USES_FR = true;
#The folder where the result files will be saved.
OUTPUT_FOLDER = "results/trial";
#The filename where the output results will be saved. Extension will be automatically added at the end. If this is blank then a random unique name will be generated
OUTPUT_FILENAME = "results500_0.01simulation_time_10rate_828nucleosomes";
#The time interval in seconds after which intermediary results will be printed. If zero is used then no intermediary results will be produced.
PRINT_INTERMEDIARY_RESULTS_AFTER = "0.0";
#This is true if the occupancy at the end of the simulation is printed and false otherwise.
PRINT_FINAL_OCCUPANCY = "false";
#This is true if the simulation is in debug mode (prints all actions to the status file) and false otherwise
DEBUG_MODE = false;
#TF species of which dynamic behaviour is followed
OUTPUT_TF = "";
#The number of intermediary points at which the TF species dynamic behaviour is recorded.
OUTPUT_TF_POINTS = 1;
#This is true if the simulator will output the dynamic behaviour of the target site occupancy.
FOLLOW_TS = true;
#If this is true the simulator will output the affinity landscape at the end of the simulation
OUTPUT_AFFINITY_LANDSCAPE = true;
#If this is true the simulator will output the DNA binding energy instead of affinity at the end of the simulation
OUTPUT_BINDING_ENERGY = true;
#If this is true the simulator will output the DNA occupancy at the end of the simulation
OUTPUT_DNA_OCCUPANCY = true;

```
#If this is true a bound molecule will affect the DNA occupancy of the entire cover area of the DNA, while when is false only the first binding position of the molecule is considered when computing the DNA occupancy
DNA_OCCUPANCY_FULL_MOLECULE_SIZE = false;
#If this is true, the simulator will print all recorded sliding lengths.
OUTPUT_SLIDING_LENGTHS = false;
#The value of the step in a fixed step wig file, used for the occupancy output.
WIG_STEP = 1;
#This represents the threshold (as procentage of the highest peak) for discarding peaks in wig files. Use -1 for auto-select and 0 for no threshold.
WIG_THRESHOLD = 0.0;

#TF PARAMATERS
#The csv file which stores the TF data.
TF_FILE = "biodata/2019biodata/trial/TF_parameters_10rate_828nucleosomes.csv";
#The csv file which stores the TF cooperativity table data.
TF_COOPERATIVITY_FILE = "";
#A file containing the target sites.
TS_FILE = "biodata/2019biodata/ESPL/TF_SuH_peak1_ts.csv";
#TF_RANDOM PARAMATERS
#The minimum length of the DNA Binding Domain of TFs (bp).
TF_DBD_LENGTH_MIN = 6;
#The maximum length of the DNA Binding Domain of TFs (bp).
TF_DBD_LENGTH_MAX = 18;
#The number of TF species.
TF_SPECIES_COUNT = 2;
#The minimum TF copy number.
TF_COPY_NUMBER_MIN = 1000;
#The maximum TF copy number.
TF_COPY_NUMBER_MAX = 1000;
#The energy penalty for a nucleotide mismatch.
TF_ES = 2.0;
#The  number of base pairs covered to the left of the DBD by a TF molecule bound to the DNA.
TF_SIZE_LEFT = 0;
#The  number of base pairs covered to the right of the DBD by a TF molecule bound to the DNA.
TF_SIZE_RIGHT = 0;
#The association rate between TF molecules and DNA.
TF_ASSOC_RATE = 20.0;
#This is true if TFs read in both directions and false otherwise.
TF_READ_IN_BOTH_DIRECTIONS = false;
#The proportion of TF molecules that are already bound when the simulation starts.
TF_PREBOUND_PROPORTION = 0.9;
#This is true if the TF is already bound to the highest affinity sites when the simulation starts.
TF_PREBOUND_TO_HIGHEST_AFFINITY = true;
#This is true if  sliding and hopping affects the association rate between TF molecules and DNA.
SLIDING_AND_HOPPING_AFFECTS_TF_ASSOC_RATE = false;

#DNA PARAMETERS
#The fasta file which stores the DNA sequence.
DNA_SEQUENCE_FILE = "biodata/2019biodata/dm6_3R_ESPL.fasta";
```

#DNA_RANDOM PARAMATERS
#The length of the DNA (bp).
DNA_LENGTH = 131000;
#The proportion of adenine (A) in the randomly generated DNA.
DNA_PROPORTION_OF_A = 0.246;
#The proportion of thymine (T) in the randomly generated DNA.
DNA_PROPORTION_OF_T = 0.246;
#The proportion of (cytosine) C in the randomly generated DNA.
DNA_PROPORTION_OF_C = 0.254;
#The proportion of (guanine) G in the randomly generated DNA.
DNA_PROPORTION_OF_G = 0.254;
#boundary condition of the DNA (absorbing/reflexive/periodic).
DNA_BOUNDARY_CONDITION = reflexive;
#This is true if the TF is immobile on DNA.
TF_IS_IMMOBILE = false;
#TF_RANDOM_WALK PARAMATERS
#The probability that a TF unbinds. 0.0001
TF_UNBINDING_PROBABILITY = 0.001474111;
#The probability that a TF slides left.0.49995
TF_SLIDE_LEFT_PROBABILITY = 0.4992629;
#The probability that a TF slides right.0.49995
TF_SLIDE_RIGHT_PROBABILITY = 0.4992629;
#The probability that a TF performs a jump when unbound instead of returning to the DNA. (0.1675 Wunderlich and Mirny 2008)
TF_JUMPING_PROBABILITY = 0.1675;
#The standard displacement of a TF that unbinds and attempts to rebind correlated. The displacement distribution is Gaussian.
TF_HOP_STD_DISPLACEMENT = 1.0;
#Waiting time of a TF to a specific site (s).
TF_SPECIFIC_WAITING_TIME = 1.5;
#The size of the step to the left when the TF performs a left slide.
TF_STEP_LEFT_SIZE = 1;
#The size of the step to the right when the TF performs a left slide.
TF_STEP_RIGHT_SIZE = 1;
#The size of the uncorrelated displacement (bp).
TF_UNCORRELATED_DISPLACEMENT_SIZE = 5;
#This parameter is true if the TF stays at current position if cannot relocate and false if it unbinds.
TF_STALLS_IF_BLOCKED = true;
#This probability that if a TF collides with another molecule on the DNA it will unbind.
TF_COLLISION_UNBIND_PROBABILITY = 0.0;
#The roughness of the affinity landscape. This is usually specified for non-cognate species.
TF_AFFINITY_LANDSCAPE_ROUGHNESS = 1.0;
#This is true if the simulator will check the DNA occupancy before binding and false otherwise
CHECK_OCCUPANCY_ON_BINDING = true;
#This is true if the simulator will check the DNA occupancy before sliding and false otherwise
CHECK_OCCUPANCY_ON_SLIDING = true;
#This is true if the simulator will check the DNA occupancy before re-binding and false otherwise
CHECK_OCCUPANCY_ON_REBINDING = true;
#This is true if the biased and false if the random walk is unbiased

IS_BIASED_RANDOM_WALK = false;
#This is true if the random walk is performed in a two-state model (search/recognition) and false otherwise
IS_TWO_STATE_RANDOM_WALK = false;
#The tsv file that contains the HiC data
HIC_CONTACT_MATRIX_FILE = "biodata/2019biodata/ESPL/BG3_Keerthi_merged_hic_matrix_500bp_corrected_3R_25951000_26082000.GInteractions.tsv";
#The width of the bins that create the HiC matrix
BIN_WIDTH= 500;
#The probability of the molecule to stay in the microenvironment
PK_MICROENV = 0.15;
#The time interval at which the matrix is simulated again.
IM_SIMULATION_TIME = 0.01; (used values: 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10)

SimulationEvent Class

```
package event;

/**
 *class that describes a simulation event. it is an instantiation of Event
 * @author adumita@essex.ac.uk
 *
 */
public class SimulationEvent extends Event {

        /**
         * 
         */
        private static final long serialVersionUID = 1L;
        public boolean isSimulationEvent;

        public SimulationEvent(double time, int nextAction, boolean 
isSimulationEvent) {
                super(time,nextAction);
                this.isSimulationEvent = isSimulationEvent;
        }


        /**
         * generates the description string of current event
         */
        public String toString(){
                String stateStr=""+time+": ";
                stateStr+=" through an event of type "+nextAction;
                return stateStr;
        }
        /**
         * compares whether this event equals the one supplied as an argument
         * @param pe
         * @return
         */
        public boolean isEqualTo(SimulationEvent pe){
                return this.time == pe.time;
        }
}
```

SimulationEventQueue Class

```
package event;

import java.io.Serializable;

import environment.Cell;

/**
 * class that holds all the simulation events
 * @author adumita@essex.ac.uk
 *
 */
public class SimulationEventQueue implements Serializable {
        /**
```

```
     *
     */
    private static final long serialVersionUID = 1L;

    protected double simulationPropensity;// the propensity that the matrix
will be simulated again;

    private SimulationEvent simulationEvent;

    public SimulationEventQueue(Cell n){

    this.simulationPropensity=n.ip.IM_SIMULATION_TIME.value;
                    this.simulationEvent = null;
                    //simulationEvent.time=
generateExponentialDistribution(simulationPropensity,n);
    }

    public double generateExponentialDistribution(double mean,Cell n) {
        return Math.log(1-n.randomGenerator.nextDouble())/(-mean);
    }

    /**
     * returns the next Simulation event
     * @return
     */
    public SimulationEvent peek(){
        return simulationEvent;
    }

    /**
     * returns the next Simulation event and removes it from the list
     * @return
     */
    public SimulationEvent pop(){
        SimulationEvent pe=simulationEvent;
        simulationEvent = null;
        return pe;
    }

    /**
     * replaces the current Simulation event with a new one
     * @param re the new event
     */
    public void add(SimulationEvent pe){
        simulationEvent=pe;
        //System.out.println(re);
    }


    /**
     * returns true if the Simulation event is null
     * @return
     */
    public boolean isEmpty(){
        return simulationEvent==null;
    }
```

```
        /**
         * deletes current protein binding event
         */
        public void clear(){
                this.simulationEvent = null;
        }

}
```

InteractionMatrix Class

```
package objects;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Random;
import java.util.Scanner;
import utils.Utils;

/**
 * class that constructs the 3D contact matrix based on the file inputted
 * @author adumita@essex.ac.uk
 *
 */
public class InteractionMatrix implements Serializable{

        /**
         *
         */
        private static final long serialVersionUID = 1L;
        private boolean[][] SimulatedMatrix;
        private double[][] CumulativeSimulatedMatrix;
        private double[][] ContactMatrix;
     private ArrayList<DNAregion> bins;
     private double[][] NormalisedMatrix;
     private double LastTimeSimulated;

        /**
         * class constructor
         *
         * @param ContactMatrixLocation the path to the file containing the HiC
interactions
         * @param regionOfInterest object of type DNAregion that contains the start
and end of the sequence based on which the bins are created
         * @param binWidth the size of the bins to be created
         * @param generator
         * @throws Exception
         */
        public InteractionMatrix(String ContactMatrixLocation,DNAregion
regionOfInterest, int binWidth,Random generator) throws Exception {

                        LastTimeSimulated=0;
                        bins = new ArrayList<DNAregion>();
                        createMatrix(regionOfInterest,binWidth);
```

```
                 addScoreToMatrix(regionOfInterest, ContactMatrixLocation);
                 normaliseMatrix();
                 simulateMatrix(generator,0);
       }

       /**
        * class constructor
        *
        * @param regionOfInterest object of type DNAregion that contains the start
and end of the sequence based on which the bins are created
        * @param binWidth the size of the bins to be created
        * @param generator
        */
       public InteractionMatrix(DNAregion regionOfInterest,int binWidth,Random
generator) {
                    bins = new ArrayList<DNAregion>();
                    createMatrix(regionOfInterest,binWidth);
                    normaliseMatrix();
                    simulateMatrix(generator,0);
       }

             /**
              * creates the empty matrix using the DNA region of interest
              * @param regionOfInterest the subsequence of dna of interest
              * @param binWidth the width of the bins to be created
              */
             public void createMatrix(DNAregion regionOfInterest,int binWidth) {

                          //reads the start value of the sequence and end value
of it
                          long start=regionOfInterest.start;
                          long end=regionOfInterest.end;
                          //calculates the number of bins for the sequence
                          int nrOfBins=(int) (end-start)/binWidth;
                          //creates the bins and adding them to an array of type
bin

                          for(int i= (int)start;i<end; i=i+binWidth) {
                            DNAregion bin= new
DNAregion(regionOfInterest.chromosome,i,i+binWidth);
                            bins.add(bin);
                          }

                          // creates the empty matrix with the values of 1 on
diagonal
                          int n=nrOfBins;
                          CumulativeSimulatedMatrix = new double[n][n];
                          ContactMatrix = new double[n][n];
                          SimulatedMatrix= new boolean[n][n];
                          NormalisedMatrix= new double[n][n];
                          for(int i=0;i<nrOfBins;i++) {
                            ContactMatrix[i][i]=1;
                            SimulatedMatrix[i][i]=true;
                            NormalisedMatrix[i][i]=0;
                            CumulativeSimulatedMatrix[i][i]=0;
                          }

              }
             /**
              * Iterates through the simulated matrix and if the bins are
interacting
              * the cumulative matrix will give the amount of time the bins were
interacting
              * @param newTime the time at which the simulation happened
              */
             public void addValueToCumulativeSimulatedMatrix(double newTime) {
                    double addTime= newTime-LastTimeSimulated;
                    for(int i=0;i<bins.size();i++) {
                          for(int j=0;j<bins.size();j++) {
                                 if(SimulatedMatrix[i][j]) {
                                        CumulativeSimulatedMatrix[i][j]+=addTime;
                                 }
                          }
                    }
             }

             /**
              * reads the file of interest and adds the values from it to a local
file
              * the local file is read and the values for local arraylists for
bins and score are added
              * @param ContactMatrixLocation the path to the file containing the
HiC interactions
              * @throws Exception
              */
             public void addScoreToMatrix(DNAregion regionOfInterest, String
ContactMatrixLocation) {
                          //initialises every list
                          ArrayList<String> localFile= new ArrayList<String>();
                          ArrayList<DNAregion> binsX= new ArrayList<DNAregion>();
                          ArrayList<DNAregion> binsY= new ArrayList<DNAregion>();

                          ArrayList<Double> score = new ArrayList<Double>();

                          //reads the interaction matrix file
                          String fileName= ContactMatrixLocation;
                          File file = new File(fileName);

                          try {
                            Scanner inputStream= new Scanner(file);

                            while (inputStream.hasNextLine()) {
                                   String data = inputStream.nextLine();
                                   String[] values = data.split("\t");
                                   for(String val:values) {
                                      localFile.add(val);

                                   }
                          }
                            inputStream.close();
                          }
                          catch(FileNotFoundException e) {
                           e.printStackTrace();
                          }
                          //adds the values of the score from the local file to
the arraylist score
                          for(int i=6; i<localFile.size(); i=i+7) {
                               double value = 
Double.parseDouble(localFile.get(i));
                                  score.add(value);
                          }
                          //local bins X and Y from file
                          for(int i=1; i<localFile.size(); i=i+7) {
                            DNAregion bin= new 
DNAregion(localFile.get(0),Integer.parseInt(localFile.get(i)), 
Integer.parseInt(localFile.get(i+1)));
                                 binsX.add(bin);
                          }
                          for(int i=4; i<localFile.size(); i=i+7) {
                             DNAregion  bin = new 
DNAregion(localFile.get(3),Integer.parseInt(localFile.get(i)), 
Integer.parseInt(localFile.get(i+1)));
                                 binsY.add(bin);
                          }
                          boolean anyValue = false;
                          
                          //adds the values from the score for the specified 
region of interest into the matrix
                          for(int g=0;g<score.size();g++) {
                                   int indexBinX=getBinIndex(bins,binsX.get(g));
                                   int indexBinY=getBinIndex(bins,binsY.get(g));
                                   if(indexBinX>=0 && indexBinY>=0) {
                                 ContactMatrix[indexBinX][indexBinY]=score.get(g);
                                 
                                 anyValue = true;
                                 
                                   }
                          }
                          //throws an error if there is a chromosome mismatch 
between the interaction matrix and the dna region
                          if(!anyValue) {
                                 throw new IllegalArgumentException("Chromosome in 
the InteractionMatrix does not match the chromosome in the DNA sequence.");
                          }
              }

         /**
          * the method looks into the array and compares each object in the array 
with the object to be found
          * returns the index in the array at which the bin has the same value
          * @param bins array of type Bin
          * @param subject object of class Bin
          * @return the bin position in the list based on its details
          */
              private int getBinIndex(ArrayList<DNAregion> bins,DNAregion 
subject) {
                          for(int i=0; i<bins.size(); i++) {
                         if(bins.get(i).equals(subject)) {
                               return i;
                            }
                          }
                          return -1;
              }

              /**
               * Method that searches the bin in which a certain position is in
               * @param position the position of the object of interest
               * @param chromosome the input param has to match the chromosome of
the bins from the list
               * @return the index of the current bin
               */
              public int getCurrentBinIndex(int position, String chromosome) {
                          int local=-1;
                          for(int i = 0; i<bins.size(); i++) {
                                 if(chromosome.equals(bins.get(i).chromosome) &&
position>=bins.get(i).start && position<=bins.get(i).end) {
                                  local= i;

                                 }
                          }
                          return local;
              }

              /**
               * gets the bin's details based on its index
               * @param index the index of the bin
               * @return the details of the bin
               */
              public DNAregion getBin(int index) {
                          return bins.get(index);
              }

              /**
               * takes the input parameter and searches against the existing
arraylist of bins to get the interacting bins
               * @param bin the no of the bin of interest
               * @return a list of indexes of bins
               */
              public ArrayList<Integer> getInteractingBins(int bin){

                          ArrayList<Integer> local = new ArrayList<Integer>();
                          for(int i =0; i<bins.size(); i++) {
                                  if(SimulatedMatrix[bin][i]) {
                                       local.add(i);
                                  }
                          }
                          return local;
              }

              /**
               * gives the size of the bins list
               * @return integer = the size of bins list
               */
              public int getBinSize() {
                          return bins.size();
              }

              /**
               * prints out details regarding 2 bins and the score between them
as well as the bins ID
               */
              public void find(int BinX,int BinY) {
                          System.out.println("BinX: "+BinX+ "BinY: "+BinY
+"Score: " +ContactMatrix[BinX][BinY] +"Bins ID: " +bins.get(BinX) +" "+
bins.get(BinY) );
```

```
              }

              /**
               * Accessor method for the arraylist 'bins';
               * @return a list of type ArrayList<DNAregion>
               */
              public ArrayList<DNAregion> getBinsList() {
                          return bins;
              }

              /**
               * Method that takes the highest score from the score list
               * and divides every value from the interaction matrix by the
highest score to get the normalised values between 0 and 1
               */
              private void normaliseMatrix() {
                          double highestScore = 0;
                          //get the highest score
                          for(int i=0;i<bins.size();i++) {
                           for(int j=0;j<bins.size();j++) {
                                 if(ContactMatrix[i][j]>highestScore) {
                                       highestScore=ContactMatrix[i][j];
                                 }
                           }
                          }
                          //generate normalised matrix
                          for(int i=0;i<bins.size();i++) {
                           for(int j=0;j<bins.size();j++) {

NormalisedMatrix[i][j]=ContactMatrix[i][j]/highestScore;
                           }
                          }
              }

              /**
               * Simulates the Interaction Matrix
               * Can be accessed and simulated from other classes to have a
dynamic simulation matrix
               * @param generator
               */
              public void simulateMatrix(Random generator, double time) {

                     LastTimeSimulated=time;
                          for(int i=0;i<bins.size();i++) {
                                for(int j=0; j<bins.size();j++) {
                                      double
probability=Utils.generateNextDouble(generator, 0, 1);

      SimulatedMatrix[i][j]=(NormalisedMatrix[i][j]>probability);
                                }
                          }
              }

              /**
               * method that returns the cumulative simulation time between two
bins
               * @param t bin X
               * @param b bin Y
               * @return
               */
              public double CumulativeResult(int t, int b) {
                    return CumulativeSimulatedMatrix[t][b];
              }

              /**
               * method that generates a random number between the values of the
start and end of a bin
               */
              public int radomBinPosition(Random generator,int randomBinID) {
               int newPosition =
(generator.nextInt((int)(bins.get(randomBinID).end-
bins.get(randomBinID).start)+1)+(int)bins.get(randomBinID).start)-
(int)bins.get(0).start;
              return newPosition;
              }

              /**
               *
               * @return
               */
              public String headerToString() {
                    String str = "\"BIN_X\", \"BIN_Y\", \"HIC_SCORE\",
\"CUMULATIVE_SIMULATION\"";
                          return str;
              }

              /**
               * a string with the details about the bins interacting, their
interaction score and the cumulative time they are interacting
               * @param x bin X
               * @param y bin Y
               * @return
               */
              public String toString(int x, int y){
                          //"\"BIN_X\", \"BIN_Y\",
                          String str="\""+ bins.get(x);
                          //\"BIN_Y\", \"HIC_SCORE\", \"CUMULATIVE_SIMULATION\"
                          str+="\","+bins.get(y)+", "+ContactMatrix[x][y]+",
"+CumulativeSimulatedMatrix[x][y];
                          return str;
                    }
}
```

Methods in Cell Class

```
      /**
       * prints details regarding the bins the interacting score between them and
the cumulative simulation time
       * @param filename The name of output file
       */
      private void saveCumulativeMatrix(String filename) {
             // checks if there is a HIC interacting matrix file
             if(!this.ip.HIC_CONTACT_MATRIX_FILE.value.isEmpty()){
                   BufferedWriter bufferFile =  null;
                   try {
```

```
           //Construct the BufferedWriter object
                if(this.outputPath.isEmpty()){
                    bufferFile = new BufferedWriter(new
FileWriter(filename));
                } else{
                    bufferFile = new BufferedWriter(new FileWriter(new
File(this.outputPath,filename)));
                }
                bufferFile.write(this.HIC_CONTACT_MATRIX.headerToString());
                bufferFile.newLine();
                for(int i=0; i<this.HIC_CONTACT_MATRIX.getBinSize();i++) {
                    for(int j=0;
j<this.HIC_CONTACT_MATRIX.getBinSize();j++) {

    bufferFile.write(this.HIC_CONTACT_MATRIX.toString(i, j));
                        bufferFile.newLine();
                    }
                }
                bufferFile.flush();
                bufferFile.close();
                 } catch (FileNotFoundException ex) {
                       ex.printStackTrace();
                  } catch (IOException ex) {
                       ex.printStackTrace();
                  }
            }

    }



    /**
     * either load the Contact matrix from a file or generate an empty one
     * @throws Exception
     */
    private void createInteractionMatrix() throws Exception{
        if(ip.HIC_CONTACT_MATRIX_FILE.value!=null) {
            this.HIC_CONTACT_MATRIX = new
InteractionMatrix(ip.HIC_CONTACT_MATRIX_FILE.value,new
DNAregion(dna.subsequence.chromosome,dna.subsequence.start,dna.subsequence.end),bi
nWidth,randomGenerator);

        }
        else {
            this.HIC_CONTACT_MATRIX= new
InteractionMatrix(dna.subsequence,binWidth,randomGenerator);
        }
    }
```

In addition to this, other changes to existing methods and classes were done to implement the code from above.

References

Ancona, M. et al. (2019). Transcriptional Bursts in a Nonequilibrium Model for Gene Regulation by Supercoiling. Biophys J. 117(2), pp.369-376.

Avcu, N. and Molina, N. (2016). Chromatin structure shapes the search process of transcription factors. Preprint at https://www.biorxiv.org/content/early/2016/04/25/050146 (2016).

Berg,O.G. et al. (1981) Diffusion-driven mechanisms of protein translocation on nucleic acids. 1. models and theory. Biochemistry, 20, 6929–6948.

Blainey,P.C. et al. (2006) A base-excision DNA-repair protein finds intrahelical lesion bases by fast sliding in contact with DNA. PNAS, 103, 5752–5757.

Boley, N., Wan, K., Bickel, P. and Celniker, S. (2014). Navigating and mining modENCODE data. Methods, 68(1), pp.38-47.

Brackley, C. A. et al. (2012). Facilitated diffusion on mobile DNA: configurational traps and sequence heterogeneity. Phys Rev Lett, 109, 168103.

Brackley, C. et al., (2016). Stochastic Model of Supercoiling-Dependent Transcription. Physical Review Letters, 117(1).

Caravaca, J. M. et al. (2013). Bookmarking by specific and nonspecific binding of FoxA1 pioneer factor to mitotic chromosomes. Genes Dev, 27, 251-60.

Cencini, M. & Pigolotti, S. (2018). Energetic funnel facilitates facilitated diffusion. Nucleic Acids Res, 46, 558-567.

Cha, M. and Zhou, Q. (2014). Detecting clustering and ordering binding patterns among transcription factors via point process models. Bioinformatics, 30(16), pp.2263-2271.

Chathoth, KT. and Zabet, NR., (2019). Chromatin architecture reorganisation during neuronal cell differentiation in Drosophila genome. Genome Research. 29 (4), 613-625

Clark, S., (2017). ‘A bioinformatics approach to model binding of suppressor of hairless to DNA’. BSc thesis, University of Essex, Colchester.

Cortini, R. and Filion, G. (2018). Theoretical principles of transcription factor traffic on folded chromatin. Nature Communications, 9(1).

Cubeñas-Potts, C.; et al., (2017). 'Different enhancer classes in Drosophila bind distinct architectural proteins and mediate unique chromatin interactions and 3D architecture', Nucleic Acids Research 45(4), 1714.

Dahlke, K., Zhao, J., Sing, C. and Banigan, E. (2019). Force-dependent facilitated dissociation can generate protein-DNA catch bonds. Biophysical Journal, 117(6), pp.1085-1110.

Das,R.K. and Kolomeisky,A.B. (2010). Facilitated search of proteins on DNA: correlations are important. Phys. Chem. Chem. Phys., 12, 2999–3004.

Deal, R. and Henikoff, S. (2010). Capturing the dynamic epigenome. Genome Biology, 11(10), p.218.

Delaneau, O. et al. (2019). Chromatin three-dimensional interactions mediate genetic effects on gene expression. Science, 364.

Elf, J., Li, G. and Xie, X. (2007). Probing Transcription Factor Dynamics at the Single-Molecule Level in a Living Cell. Science, 316(5828), pp.1191-1194.

Erban, R. and Chapman, S. (2007). Reactive boundary conditions for stochastic simulations of reaction–diffusion processes. Physical Biology, 4(1), pp.16-28.

Ezer, D., Zabet, N. R. & Adryan, B. (2014). Physical constraints determine the logic of bacterial promoter architectures. Nucleic Acids Res, 42, 4196-207.

Festuccia, N. et al. (2019). Transcription factor activity and nucleosome organization in mitosis. Genome Res, 29, 250-260.

Fosado, Y. et al. (2016). A single nucleotide resolution model for large-scale simulations of double stranded DNA. Soft Matter, 12(47), pp.9458-9470.

Gebhardt, J. et al. (2013). Single-molecule imaging of transcription factor binding to DNA in live mammalian cells. Nature Methods, 10(5), pp.421-426.

Gillespie, D.T. (1976). A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of Computational Physics, 22(4), pp.403-434.

Gillespie, D.T. (1977). Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem., 81, 2340–2361.

Ghosh, S., Mishra, B., Kolomeisky, A. and Chowdhury, D. (2018). First-passage processes on a filamentous track in a dense traffic: optimizing diffusive search for a target in crowding conditions. Journal of Statistical Mechanics: Theory and Experiment, 2018(12), p.123209.

Gowers, D. M., Wilson, G. G. & Halford, S. E. (2005). Measurement of the contributions of 1D and 3D pathways to the translocation of a protein along DNA. Proc Natl Acad Sci U S A, 102, 15883-8.

Hammar, P. et al., (2012). The lac Repressor Displays Facilitated Diffusion in Living Cells. Science, 336(6088), pp.1595-1598.

Hao, N. and O'Shea, E. (2011). Signal-dependent dynamics of transcription factor translocation controls gene expression. Nature Structural & Molecular Biology, 19(1), pp.31-39.

Kharchenko, P.V., Alekseyenko, A.A., Schwartz, Y.B., Minoda, A., Riddle, N.C., Ernst, J., Sabo, P.J., Larschan, E., Gorchakov, A.A., Gu, T. and Linder-Basso, D., 2011. Comprehensive analysis of the chromatin landscape in Drosophila melanogaster. Nature, 471(7339), pp.480-485.

Krepel, D., Gomez, D., Klumpp, S. and Levy, Y. (2016). Mechanism of Facilitated Diffusion during a DNA Search in Crowded Environments. *The Journal of Physical Chemistry B*, 120(43), pp.11113-11122.

Leven, I. and Levy, Y. (2019). Quantifying the two-state facilitated diffusion model of protein-DNA interactions. Nucleic Acids Res, 47(11), pp.5530-5538.

Liu, H. et al. (2017). Visualizing long-term single-molecule dynamics in vivo by stochastic protein labeling. Proceedings of the National Academy of Sciences, 115(2), pp.343-348.

Martin, P.C.N., Zabet, N.R., (2019) Dissecting the binding mechanisms of transcription factors to DNA using a statistical thermodynamics framework. bioRxiv. 2019;666446.

Mirny,L. et al. (2009). How a protein searches for its site on DNA: the mechanism of facilitated diffusion. J. Phys. A, Math. Theor., 42, 434013.

Murugan, R. (2018). Theory of Site-Specific DNA-Protein Interactions in the Presence of Nucleosome Roadblocks. Biophysical Journal, 114(11), pp.2516-2529.

Pal, S., Hoinka, J. and Przytycka, T. (2019). Co-SELECT reveals sequence non-specific contribution of DNA shape to transcription factor binding in vitro. Nucleic Acids Research, 47(13), pp.6632-6641.

Piatt, S. C., Loparo, J. J. & Price, A. C. (2019). The Role of Noncognate Sites in the 1D Search Mechanism of EcoRI. Biophys J. 116(12), pp.2367-2377.

Raccaud, M., Friman, E., Alber, A., Agarwal, H., Deluz, C., Kuhn, T., Gebhardt, J. and Suter, D. (2019). Mitotic chromosome binding predicts transcription factor properties in interphase. Nature Communications, 10(1).

Rao, S. P.; et al., (2014). 'A 3D Map of the Human Genome at Kilobase Resolution Reveals Principles of Chromatin Looping', Cell 159(7), 1665 - 1680.

Riggs,A.D. et al. (1970). The lac represser-operator interaction: iii. kinetic studies. J. Mol. Biol., 53, 401–417.

Rizkallah, R., Alexander, K. E. & Hurt, M. M. (2011). Global mitotic phosphorylation of C2H2 zinc finger protein linker peptides. Cell Cycle, 10, 3327-36.

Robertson, G; et al. (2007). "Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing". Nature Methods. 4 (8): 651–657.

Sheinman M. and Kafri Y., (2009). The effects of intersegmental transfers on target location by proteins Phys. Biol. 6(1), p.016003.

Shin, J. and Kolomeisky, A. B. (2019). Facilitation of DNA loop formation by protein-DNA non-specific interactions. Soft Matter, 15(26), pp.5255-5263.

Skalska, L., Stojnic, R., Li, J., Fischer, B., Cerda-Moya, G., Sakai, H., Tajbakhsh, S., Russell, S., Adryan, B., and Bray, S.J., (2015). Chromatin signatures at Notch-regulated enhancers reveal large-scale changes in H3K56ac upon activation. The EMBO journal, p.e201489923.

Slutsky, M. & Mirny, L. A. (2004). Kinetics of protein-DNA interaction: facilitated target location in sequence-dependent potential. Biophys J, 87, 4021-35.

Stadhouders, R. et al. (2018). Transcription factors orchestrate dynamic interplay between genome topology and gene regulation during cell reprogramming. Nature Genetics, 50, 238-+.

Stadhouders, R., Filion, G. J. & Graf, T. (2019). Transcription factors and 3D genome conformation in cell-fate decisions. Nature, 569, 345-354.

Stiehl, O., Weidner-Hertrampf, K. & Weiss, M. (2016). Corrigendum: Kinetics of conformational fluctuations in DNA hairpin-loops in crowded fluids (2013New J. Phys.15113010). New Journal of Physics, 18(9), p.099501.

Tao Hu, Grosberg A. Y. and Shklovskii B. I., (2006). How proteins search for their specific sites on DNA: the role of DNA conformation Biophys. J. 90 2731–44

Viadiu,H. and Aggarwal,A.K. (2000). Structure of BamHI bound to nonspecific DNA: a model for DNA sliding. Mol. Cell, 5, 889–895.

von Hippel,P.H. (2007). From ‘simple’ DNA-protein interactions to the macromolecular machines of gene expression. Annu. Rev. Biophys. Biomol. Struct., 36, 79–105.

Weindl, J., Dawy, Z., Hanus, P., Zech, J. & Mueller, J. C. (2009). Modeling promoter search by E. coli RNA polymerase: one-dimensional diffusion in a sequence-dependent energy landscape. J Theor Biol, 259, 628-34.

Woringer, M. and Darzacq, X. (2018). Protein motion in the nucleus: from anomalous diffusion to weak interactions. Biochemical Society Transactions, 46(4), pp.945-956

Wunderlich,Z. and Mirny,L.A. (2008). Spatial effects on the speed and reliability of protein-DNAsearch. Nucleic Acids Res., 36, 3570–3578.

Zabet,N.R. (2012). System size reduction in stochastic simulations of the facilitated diffusion mechanism. BMC Syst.Biol. 6:121.

Zabet,N.R.,and Adryan,B.(2012a). A comprehensive computational model of facilitated diffusion in prokaryotes. Bioinformatics 28, 1517–1524.

Zabet,N.R.,and Adryan,B.(2012b). Computational models for large-scale simulations of facilitated diffusion. Mol. Biosyst. 8, 2815–2827.

Zabet,N.R.,and Adryan,B.(2012c). GRiP: a computational tool to simulate transcription factor binding in prokaryotes. Bioinformatics 28, 1287–1289.

Zabet, N., Foy, R. and Adryan, B. (2013). The Influence of Transcription Factor Competition on the Relationship between Occupancy and Affinity. PLoS ONE, 8(9), p.e73714.

Zabet, N. and Adryan, B. (2013). The effects of transcription factor competition on gene regulation. Frontiers in Genetics, 4, 197–206.

Zabet, N. R. & Adryan, B. (2015). Estimating binding properties of transcription factors from genome-wide binding profiles. Nucleic Acids Res, 43, 84-94.