Discrete Stochastic Models of Molecular Motor Stepping Cycles

NEVILLE J. BOON

Submitted for the degree of Doctor of Philosophy
Department of Mathematics
University of Surrey

©Neville J. Boon, June, 2013
To my parents
for their lifelong support in all my endeavours
Preface

I would like to thank my advisor Prof. Rebecca Hoyle for her patience, help and guidance, without whom this work would not have been possible. Thanks also to Dr. Anne Skeldon for useful discussions and her feedback.

This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/P505135/1].

This thesis and the work to which it refers are the results of my own efforts. Any ideas, data, images or text resulting from the work of others (whether published or unpublished) are fully identified as such within the work and attributed to their originator in the text, bibliography or in footnotes. This thesis has not been submitted in whole or in part for any other academic degree or professional qualification.

I agree that the University has the right to submit my work to the plagiarism detection service TurnitinUK for originality checks. Whether or not drafts have been so-assessed, the University reserves the right to require an electronic version of the final document (as submitted) for assessment as above.
Abstract

A molecular motor is the nano-scale combustion engine of the cell: it uses a chemical reaction to drive motion. These proteins are fundamental to many cellular processes such as intracellular transport or gene transcription and understanding their behaviour is vital in understanding how we all function. There exist many different types of molecular motor, in this work I am concerned with stepping motors that walk hand-over-hand along a track within a cell.

Experiments imply how molecular motors function but in order to describe this precisely one uses the language of mathematics. As motors are small and difficult to observe there is controversy about their movement and thus many competing descriptions, or models, exist. This work focuses on creating and applying general methods to compare the fit to experimental data of different models of the motor myosin-V and its stepping cycles.

A review of existing theoretical and experimental work on molecular motors is conducted with emphasis on one type: myosin-V (Chapter 1). Extensions of existing theoretical methods are discussed (Chapter 2) and a novel method for calculating experimentally measurable quantities of molecular motors is presented (Chapter 3). A framework to compare competing models of myosin-V is described in Chapter 4 that allows one to identify mechanisms that enable models to reproduce experimentally observed behaviour. In Chapter 5 a set of models for myosin-V is investigated to establish mechanisms compatible with experimental trends for the average velocity and run length against nucleotide concentration. Asymmetric gating, futile cycling (foot stomping) and a loss of chemical coordination within the molecule are shown to be suitable candidates. In Chapter 6 these ideas are extended to include myosin-V under external forcing. Here multiple substeps, the elastic properties of the motor and slippage along the track are demonstrated to be vital in reproducing important experimental trends.
Contents

List of Figures vii

List of Tables ix

Nomenclature xi

1 The Myosin-V Walk 1

1.1 Introduction ............................................. 1

1.2 Experimental Work on Myosin-V 3

1.2.1 Experimental Methods ............................... 4

1.2.2 Measurement of Dynamic Properties ... 7

1.2.3 Kinetic Measurements ............................... 10

1.3 How Does Myosin-V Walk? ............................ 11

1.3.1 Chemical States of the Myosin-V Heads .......... 11

1.3.2 Head Coordination ................................. 12

1.3.3 Molecular Strain ................................... 13

1.3.4 The Telemark State and the Powerstroke Step . 13

1.3.5 Substeps .............................................. 13

1.3.6 The Rate-limiting Step ............................. 15

1.3.7 Disassociation from Actin ......................... 15

1.3.8 Hydrolysis Coupling ................................ 15

1.4 Existing Mathematical Models of Myosin-V ....... 16

1.4.1 Brownian Ratchet Models for Myosin-V .......... 16

1.4.2 Mechanochemical Models for Myosin-V .......... 17

1.4.3 Discrete Stochastic Models for Myosin-V ....... 17

1.4.4 Stepping Cycles ................................... 18
## CONTENTS

1.4.5 Calculating Dynamic Properties ........................................ 20
1.4.6 Combining Theory and Experiment ................................. 22
1.5 Summary ................................................................. 24

2 Dynamic Properties of Molecular Motors .................................. 25

2.1 A Toy Model ............................................................. 27
  2.1.1 Two States, One Branch ........................................ 27
  2.1.2 Two States, Two Branches ..................................... 31
2.2 Existing Work on System Structures .................................... 34
  2.2.1 Single Pathway Model ........................................... 34
  2.2.2 Branching Models ................................................ 36
2.3 Extensions and Original Work .......................................... 38
  2.3.1 Parallel Pathway Models ........................................ 38
  2.3.2 Divided Pathway Models ........................................ 41
2.4 Molecular Detachment .................................................. 44
  2.4.1 Renormalisation ................................................... 44
  2.4.2 Two State, Two Substep System with Unequal Detachment .. 46
2.5 Summary ................................................................. 52

3 Calculating Dynamic Properties Generally ................................ 53

3.1 Introduction ............................................................ 53
3.2 General Steady-State Probabilities ..................................... 55
3.3 Configurational Methods ............................................... 59
3.4 Dynamic Properties .................................................... 61
  3.4.1 Velocity .......................................................... 62
  3.4.2 Dispersion ......................................................... 63
3.5 Calculating the $C_{ij}$ .................................................. 67
  3.5.1 Modifying System Structures ................................... 72
3.6 Generalised System Architectures ...................................... 74
  3.6.1 Single Chain Model .............................................. 75
  3.6.2 Parallel Pathway Model ......................................... 77
  3.6.3 The Divided Pathway ............................................ 78
  3.6.4 Divided Pathway with a Chain .................................. 79
3.7 Four State Model Example ............................................... 80
## CONTENTS

3.7.1 Single Chain ............................................. 81
3.7.2 Divided Pathway with a Jump ................................. 81
3.7.3 Combining Model Structures ................................. 83
3.8 Discussion ................................................. 84

### 4 Myosin-V Stepping Models

4.1 The Skau Optimisation Procedure ................................. 88
4.2 The Metamodel of Myosin-V .................................... 91
  4.2.1 Master Equations ........................................ 93
  4.2.2 Energetics ............................................. 94
  4.2.3 Transition Rates ........................................ 98
4.3 The Cost Function ........................................... 101
4.4 A Modified Optimisation ...................................... 102
4.5 Optimisation Validation ...................................... 104
4.6 Summary .................................................. 105

### 5 Myosin-V Stepping Models: Zero Forcing

5.1 A Comparison of Two Stepping Models .......................... 107
  5.1.1 The Skau Model Versus The Wu Model ...................... 108
5.2 Molecular Detachment Mechanisms ............................... 118
  5.2.1 Chemical Detachment .................................... 118
  5.2.2 Mechanical Detachment .................................. 119
  5.2.3 Cycle with Chemical and Mechanical Detachment .......... 119
5.3 The Skau Asymmetric Gating Model ............................... 120
5.4 The Skau Asymmetric Gating Extra Model ....................... 123
  5.4.1 Critique of the SAGEM .................................. 128
5.5 Conclusions ............................................... 129

### 6 Myosin-V Models Under Force

6.1 The Bierbaum Model and the Skau Model ....................... 134
6.2 SAGEM Under Forcing ....................................... 135
  6.2.1 Two Substeps .......................................... 135
  6.2.2 Three Substeps ......................................... 140
6.3 Discussion ................................................. 145
### 7 Conclusion

### Appendices

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1</td>
<td>Running the Code</td>
<td>155</td>
</tr>
<tr>
<td>A.2</td>
<td>Bespoke Functions</td>
<td>159</td>
</tr>
<tr>
<td>A.3</td>
<td>Printing the Visuals</td>
<td>185</td>
</tr>
<tr>
<td>A</td>
<td>MATLAB Code</td>
<td>155</td>
</tr>
<tr>
<td>B.1</td>
<td>Publications</td>
<td>206</td>
</tr>
<tr>
<td>B.2</td>
<td>Awards</td>
<td>207</td>
</tr>
<tr>
<td>B.3</td>
<td>Conferences Attended</td>
<td>207</td>
</tr>
<tr>
<td>B.4</td>
<td>Courses Attended</td>
<td>207</td>
</tr>
<tr>
<td>B</td>
<td>The Author’s Further Doctoral Activities</td>
<td>206</td>
</tr>
</tbody>
</table>
# List of Figures

1.1 The structure of myosin-V ............................................. 2
1.2 A schematic representation of the procession of myosin-V .......... 3
1.3 Electron microscopy images of myosin-V ............................. 4
1.4 Experimental data: the average velocity of myosin-V molecules against concentrations of ATP and ADP ............................... 7
1.5 Experimental data: the average velocity of myosin-V molecules against external forcing .................................................. 8
1.6 Experimental data: the average run length of myosin-V molecules against concentrations of ATP and ADP .............................. 9
1.7 Experimental data: myosin-V dwell time dependence on external forcing ............................................................... 10
1.8 A comparison between the different suggested sub-steps of myosin-V 14
1.9 Two energy surfaces in a possible Brownian ratchet model of myosin-V 16
1.10 The 5-state model of myosin-V suggested by Rief et al. [41] ........ 19
1.11 The discrete stochastic model developed by Skau et al. [44] to describe myosin-V stepping ................................................. 20
1.12 The discrete stochastic model of myosin-V developed by Wu et al. [61] ................................................................. 21

2.1 A periodic 1D lattice of sites in physical space .......................... 26
2.2 A simple periodic two-state model ....................................... 27
2.3 A simple periodic two-state model with a jump in state space .... 31
2.4 A simple periodic reaction network ....................................... 34
2.5 A simple periodic reaction network with branching from every state 37
2.6 A periodic parallel pathway reaction network .......................... 39
2.7 A periodic divided pathway reaction network .......................... 41
2.8 The periodic divided pathway reaction network in Figure 2.7 mapped onto a parallel pathway model coupled at two states ............ 42
2.9 A toy two-state system with two substeps and two different molecular detachment pathways .................................. 46

3.1 All configuration of state 1 for a 4 state single chain model ............ 57
3.2 The two possible classes of closed rate paths .................................. 60
3.3 A graphical example of how $C_{13}$ is calculated ................................. 69
3.4 A graphical representation of $k_a^b$, $k_{bb}^a$ and $k_{ba}^b$ ....................... 71
3.5 Structural representations of several classes of models ......................... 71
3.6 A graphical representation of the calculation of the $C_{ij}$ ................. 73
3.7 Three potential molecular motor model structures ............................... 76
3.8 A general 4-state system .................................................................. 80
3.9 A toy single chain model with no backwards stepping ....................... 81
3.10 A toy divided pathway with a jump model with no backwards stepping 82
3.11 A divided pathway with a jump and a single chain model with no backwards stepping ........................................ 84

4.1 The energy landscape between states ................................................. 89
4.2 The discrete stochastic model developed by Skau et al.[44] to describe myosin-V stepping ........................................ 90
4.3 The myosin-V metamodel .................................................................. 92

5.1 The discrete stochastic model developed by Skau et al. [44] to describe myosin-V stepping ........................................ 109
5.2 A comparison of the velocity relationships against ATP and ADP nucleotide concentration between the Skau Model and the Original Wu model .................................................. 112
5.3 A comparison of the run length relationships against ATP and ADP nucleotide concentration between the Skau Model and the Original Wu model .................................................. 113
5.4 The discrete stochastic model developed by Wu et al. [61] ................. 114
5.5 A comparison of the velocity relationships against ATP and ADP nucleotide concentration between the Skau Model and the Wu Model 116
5.6 A comparison of the run length relationships against ATP and ADP nucleotide concentration between the Skau Model and the Wu Model 117
5.7 A simple model with both chemical and mechanical detachment. ... 119
5.8 A comparison of the run length relationship against ATP and ADP nucleotide concentration between two models in the metamodel framework: the Skau Model and the Skau Asymmetric Gating Model . . 122
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.9</td>
<td>The proportion of futile flux relative to the total flux against concentration of ADP for three models</td>
<td>123</td>
</tr>
<tr>
<td>5.10</td>
<td>The Skau Asymmetric Gating Extra Model in the metamodel framework</td>
<td>124</td>
</tr>
<tr>
<td>5.11</td>
<td>A comparison of the velocity relationships against ATP and ADP nucleotide concentration between the Skau Model and the Skau Asymmetric Gating Extra Model</td>
<td>126</td>
</tr>
<tr>
<td>5.12</td>
<td>A comparison of the run length relationships against ATP and ADP nucleotide concentration between the Skau Model and the Skau Asymmetric Gating Extra Model</td>
<td>127</td>
</tr>
<tr>
<td>5.13</td>
<td>The proportion of fluxes involving ADP release relative to the total flux against concentration of ADP for the SAGEM</td>
<td>128</td>
</tr>
<tr>
<td>6.1</td>
<td>The run length relationships against ATP and ADP nucleotide concentration for the Bierbaum Model</td>
<td>133</td>
</tr>
<tr>
<td>6.2</td>
<td>A comparison of the relationship of velocity to external forcing between the Original Skau Model and the Bierbaum Model</td>
<td>137</td>
</tr>
<tr>
<td>6.3</td>
<td>The Skau Asymmetric Gating Extra Forcing Model in the metamodel framework</td>
<td>138</td>
</tr>
<tr>
<td>6.4</td>
<td>The run length relationships against ATP and ADP nucleotide concentration for the Skau Asymmetric Gating Extra Forcing Model with two substeps</td>
<td>139</td>
</tr>
<tr>
<td>6.5</td>
<td>The proportion of fluxes involving ADP release relative to the total flux against concentration of ADP for the SAGEM2</td>
<td>140</td>
</tr>
<tr>
<td>6.6</td>
<td>The velocity relationship against external forcing for the Skau Asymmetric Gating Extra Forcing Model with two substeps</td>
<td>141</td>
</tr>
<tr>
<td>6.7</td>
<td>The proportion of fluxes involving ADP release relative to the total flux against concentration of ADP for the SAGEM3</td>
<td>142</td>
</tr>
<tr>
<td>6.8</td>
<td>The velocity relationships against ATP and ADP nucleotide concentration for the Skau Asymmetric Gating Extra Forcing Model with three substeps</td>
<td>143</td>
</tr>
<tr>
<td>6.9</td>
<td>The run length relationships against ATP and ADP nucleotide concentration for the Skau Asymmetric Gating Extra Forcing Model with three substeps</td>
<td>144</td>
</tr>
<tr>
<td>6.10</td>
<td>The relationship of velocity to external forcing for the Skau Asymmetric Gating Extra Forcing Model with three substeps</td>
<td>145</td>
</tr>
</tbody>
</table>
List of Tables

1.1 Model parameters for the model of myosin-V developed by Skau et al. [44] ................................................................. 23

4.1 The optimised energy barriers, differences and the strain and gating values chosen using the optimisation routine developed by Skau et al. [44] ................................................................. 91

4.2 Energy distribution for states in the metamodel of the myosin-V stepping cycle ................................................................. 95

4.3 The experimentally measured data points for the run length and molecular velocity included in the cost function ......................... 101

5.1 Optimised parameters and final cost function values for models of myosin-V ................................................................. 110

5.2 A list of models investigated including their cost function value and whether they reproduce certain experimental trends .................. 111

5.3 The variability $\Delta H_i$ with $\epsilon = 1$ of the final optimised parameter values for models of myosin-V optimised in the metamodel framework115

6.1 Optimised parameters and final cost function values for the Skau Asymmetric Gating Extra Forcing Model with two and three substeps135
### Nomenclature

#### Biological Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actin filament</td>
<td>A helical structure that is part of the cytoskeleton</td>
</tr>
<tr>
<td>ADP</td>
<td>Adenosine diphosphate</td>
</tr>
<tr>
<td>Asymmetric gating</td>
<td>The gating effect where the energy barrier required to open the pocket depends on whether a molecule is contained within it</td>
</tr>
<tr>
<td>ATP</td>
<td>Adenosine triphosphate - the chemical fuel of the inner cell</td>
</tr>
<tr>
<td>Attachment/detachment</td>
<td>The association/disassociation of a myosin head with the actin track</td>
</tr>
<tr>
<td>Average molecular velocity</td>
<td>The averaged speed at which an arbitrarily large group of molecular motors travel towards the plus end of the actin filament</td>
</tr>
<tr>
<td>Binding/release</td>
<td>The association/disassociation of a nucleotide to a myosin head</td>
</tr>
<tr>
<td>Chemical detachment</td>
<td>Detachment from the track caused by a loss of coordination of the chemical reactions at the myosin-V heads</td>
</tr>
<tr>
<td>Cytoskeleton</td>
<td>The structure that maintains the shape of the cell - analogous to the human skeleton</td>
</tr>
<tr>
<td>Dwell time</td>
<td>The average time between stepping cycle completions. For myosin-V this is also the average time between successive powerstoke steps</td>
</tr>
<tr>
<td>Futile cycle</td>
<td>The complete, non-repeating sequence of molecular motor conformational changes in which myosin-V fails to progress down the track</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Gating</td>
<td>A strain dependent energy barrier required to be overcome to release (or bind) a ADP nucleotide into a myosin head. Corresponds to an opening of the pocket within which the nucleotides reside.</td>
</tr>
<tr>
<td>Hydrolysis cycle</td>
<td>A complete, non-repeating sequence of molecular motor conformational changes in which ATP is hydrolysed and the molecule moves along the track.</td>
</tr>
<tr>
<td>Hydrolysis step</td>
<td>The conformation transition in which ATP is transformed into ADP and P&lt;sub&gt;i&lt;/sub&gt;.</td>
</tr>
<tr>
<td>Mechanical detachment</td>
<td>Detachment from the track caused by a mechanical process such as external force or a sudden release of internal molecular strain.</td>
</tr>
<tr>
<td>Molecular detachment</td>
<td>The disassociation of an entire molecular motor molecule from the actin track.</td>
</tr>
<tr>
<td>Molecular motor</td>
<td>The nano-scale combustion engine of the inner cell.</td>
</tr>
<tr>
<td>Molecule conformation</td>
<td>The mechanical and chemical condition the molecular motor molecule exists within.</td>
</tr>
<tr>
<td>Myosin-V</td>
<td>A stepping molecular motor that walks along actin tracks.</td>
</tr>
<tr>
<td>Myosin-V heads</td>
<td>The feet of the myosin-V molecule.</td>
</tr>
<tr>
<td>Nucleotides</td>
<td>Biological molecules that make up the building blocks of DNA and RNA. Those considered here are ATP, ADP and P&lt;sub&gt;i&lt;/sub&gt;.</td>
</tr>
<tr>
<td>P&lt;sub&gt;i&lt;/sub&gt;</td>
<td>Inorganic phosphate.</td>
</tr>
<tr>
<td>Powerstroke</td>
<td>The largest and main step a myosin-V molecule makes.</td>
</tr>
<tr>
<td>Processive</td>
<td>A molecular motor is said to be processive if it remains attached to its track as it moves along it.</td>
</tr>
<tr>
<td>Rate-limiting step</td>
<td>In chemical kinetics this is the slowest reaction rate in a system. For example the slowest transition rate in the hydrolysis of ATP.</td>
</tr>
<tr>
<td>Run length</td>
<td>The average distance and arbitrarily large group of molecular motors travel before molecular detachment.</td>
</tr>
<tr>
<td>Stall force</td>
<td>The external force applied to a stepping molecular motor required to prevent movement.</td>
</tr>
</tbody>
</table>
Telemark state  The fully stained conformation of a myosin-V molecule - bent over like a skier in what is known as the telemark stance

**Modelling Definitions**

Attachment sites  The positions along a motor proteins track at which it attaches

Configuration/configuration*  A configuration of state $b$ is a non-unique rate tree of $b$ containing one rate from every state in the system except $b$ and a configuration* of $b$ is a non-unique rate tree of $b$ containing one rate from every state including $b$. A configuration cannot contain any closed rate paths, however a configuration* of $b$ must contain exactly one closed rate path.

Cost/objective function  A measure as to how close a model with a given set of parameters reproduces experimental data

Coupling state  A state that connects two or more branches

Cycle of states  A sequence on states that in which the last state is also the first

Discrete-stochastic models  A class of models for molecular motor walks that assume the biomechanochemical pathways can be split into discrete states and that transitions between these states occur probabilistically

Divided pathway model  A model that contains exactly one single chain and one parallel chain

Energy barrier  The activation energy required to initiate a chemical or mechanical change

Model branch  A set of states with nearest neighbour transitions

Model parameters  A set of constants that quantify unknown relationships within a model. These are usually chosen to maximise a models agreement with experimental data

Molecular state  A biomechanochemical conformation of a molecular motor

Occupancy probability  The proportion of molecules in a given state normalised to sum to unity in the statistical limit
Optimisation
The process by which model parameters are chosen to maximise their corresponding models agreement with experimental data

Over constrained
The results of an optimisation can be this if the objective function contains too many data points or the model is too specific

Over fitted
The results of an optimisation can be this if the optimisation routine has picked specific parameter values that have a extremised objective function value but the parameter values themselves make little physical sense. Usually the results of a loosely defined objective function

Parallel pathway model
A model with multiple single chains coupled at a single state. Also known as a parallel chain model

Rate path
A *rate path* from states *a* to *b* is a product of transition rates along the directed path from *a* to *b*.

Rate path reversal
A rate path within a rate tree with forwards rates changed into backwards rates so that the rate tree retains its properties.

Rate tree
A *rate tree* of state *b* is a product of state-unique reaction rates (only one from each state in the rate tree) that contains a rate path from each state in the rate tree to *b*.

Renormalisation
A technique to account for molecular motors leaving a discrete stochastic model so that a non-trivial steady state exists

SAGEFMi
The Skau Model with asymmetric gating and an additional pathway under forcing with *i* substeps

SAGEM
The Skau Model with asymmetric gating and an additional pathway

SAGM
The Skau Model with asymmetric gating

Single pathway model
A model with only nearest neighbour transitions. Also known as a single chain model

State transition
A *hopping* of a molecule from one state to another

Stepping mechanism
An idea as to how a stepping molecular motor can walk down its track
Transition rate  The rate at which molecules undergo a state transition

Mathematical Notation

$< x >$  Averaged displacement

$\alpha E_s$  The energy barrier for opening the myosin head pocket that contains an ADP nucleotide. For states with differing amounts of internal molecular strain, $\alpha$ can be replaced by $\beta$, $\gamma$ or $\omega$

$\alpha^* E_s$  The energy barrier for opening a myosin head pocket that does not contain an ADP nucleotide

$\Delta G_{i,j}$  Energy difference between states $i$ and $j$

$\Delta H_i$  The variability of parameter $i$. The larger this value the more that the parameter can be varied whilst keeping the cost function within a certain region

$\Delta_i$  Cost function element $i$. A scalar that represents how far away from experimental results a model is

$\delta_i$  Molecular detachment rate from state $i$

$\dot{x}$  On-track quantity $x$

$\mathcal{M}$  The transition rate matrix defined in equation 3.2.1

$\phi_i$  The $i$th eigenvector of the $\mathcal{M}$

$\rho$  The randomness ratio

$\tau$  Fundamental timescale of a reaction

$\tilde{x}$  Renormalised quantity $x$

$C_{ij}^r$  The sum over all configurations* of state $i \neq 0$ given a rate from 0 to $j$ and dividing through by that rate

$C_{ik}$  The sum over all configurations of state $i \neq 0$ given a rate from 0 to $j$ and dividing through by that rate

$D$  The dispersion of the velocities

$d$  Total distance ‘walked’ in one stepping cycle by a molecular motor

$D'$  The diffusion constant

$d_i$  A smaller substep within a larger walk. Note $\sum_i d_i = d$
**NOMENCLATURE**

$E_H$  
The energy corresponding to pulling on the cargo of the protein assuming that the molecule behaves as a Hookean spring

$E_s$  
The maximum strain energy that the myosin-V motor can hold. The release of this strain causes the powerstroke

$f_{ex}$  
The external forcing on a molecule

$G^t_{i,j}$  
Energy barrier for the transition from state $i$ to state $j$

$G_{Ds-E}$  
The chemical energy barrier (with a $\dagger$) or difference (with a $\Delta$) for a myosin head to release an ADP nucleotide

$G_{Dw-Ds}$  
The chemical energy barrier (with a $\dagger$) or difference (with a $\Delta$) for a myosin head to release an P$_i$ nucleotide

$G_{E-T}$  
The chemical energy barrier (with a $\dagger$) or difference (with a $\Delta$) for a myosin head to bind to an ATP nucleotide

$G_{T-Dw}$  
The chemical energy barrier (with a $\dagger$) or difference (with a $\Delta$) for an ATP-bound myosin head to react and attach to the actin filament. The resulting chemical state of the head now contains an ADP bound and a P$_i$ nucleotide

$J$  
The flux of molecules

$k_B$  
Boltzmann constant

$k_H$  
The Hookean spring constant

$L$  
The average distance a molecule travels before detaching - also known as the run length

$N$  
The normalisation constant

$p_{i,s}$  
The probability of a given molecule being in the $i$th site on the $s$th cycle of the periodic lattice of sites

$P_i$  
The probability of a given molecule being in conformational state $i$

$Q_i$  
The sum over all configurations of state $i$

$T$  
The temperature of the system
NOMENCLATURE

$t$ Variable denoting time

$U^\dagger$ The energy barrier for a myosin-V molecule to slip along the track

$u_{i,j}$ Forwards transition rate from state $i$ to state $j$

$V$ The average molecular velocity of a statistical number of motor proteins

$W_{i,j}$ Transition rate from state $i$ to state $j$

$w_{i,j}$ Backwards transition rate from state $i$ to state $j$

$Z_{ik}$ The sum over all configurations of state $i \neq 0$ given a rate from 0 to $j$

$Z_{ik}^*$ The sum over all configurations* of state $i \neq 0$ given a rate from 0 to $j$

$k\Lambda^b_a$ A rate path between $a$ and $b$ on branch $k$

$k\Pi^b_a$ A reversed rate path between $a$ and $b$ on branch $k$

$k\Xi^b_a$ The sum of all possible rate path reversals between states $a$ and $b$ on branch $k$
Chapter 1

The Myosin-V Walk

“How much easier is it to be critical than to be correct”

Benjamin Disraeli

Molecular motors are proteins that use the chemical energy from the hydrolysis of adenosine triphosphate (ATP) to perform mechanical work. They play important roles within a biological cell, performing functions such as intracellular transport and gene transcription. Myosin-V is a linearly processive molecular motor that moves along tracks of actin that are part of the cytoskeleton. Mathematical modelling of this protein can allow testing of current theoretical ideas against experimental data and provides insight into further avenues of experimental investigation. In this chapter a summary of existing experimental and theoretical work on myosin-V is presented and discussed.

1.1 Introduction

Biological cells are incredibly complex systems involving many biochemical processes and many of these require some form of physical movement. For example intracellular transport (the process of moving material from one place in the cell to another) is often achieved by diffusion; however, there are many cases where diffusion is either not fast or directed enough to be effective. Thus, these processes require a form of powered transport in order to fulfil their function.

A molecular motor is loosely defined to be a micron-to-nano scale protein that converts chemical energy into mechanical work; it uses energy given out from a chemical reaction in order to drive a physical movement. There exist many different types of molecular motor. However, this work focusses upon track-based molecular motors that move along pathways within a cell. These pathways are part of the cell’s cytoskeleton - a semi-solid collection of fibres that maintain a cell’s shape in a manner analogous to the human skeleton.
One example of a track-based molecular motor is myosin-V, which is involved in the intracellular transport of cargo such as vesicles, organelles or other cellular components [37]. The tracks that these cellular components move along are made of actin filaments (a helical structure that is part of the cytoskeleton). Myosin-V moves along the actin towards the barbed or plus end. The manner in which myosin-V moves along actin is the focus of this work, although the analytical techniques can be applied to other track-based motor proteins as well, such as myosin-VI that moves towards the pointed or minus end of actin filaments or the kinesin motor that moves along different tracks known as microtubules.

![Figure 1.1: The structure of myosin-V reproduced from Vale et al. [53]. On the left side there are the two stepping arms that move from one position on the track to another. These are linked at the base of the neck domain, the other end of which is attached to the molecule’s cargo.](image)

Myosin-V has two stepping arms, ending in a head, which move hand over hand (referred to as head over head in the literature) in steps of about 36nm along the actin track, and a third neck region that attaches to the cargo of the molecule i.e. the vesicles organelles or other cellular components that the motor is moving within the cell (see Figure 1.1). The movement is fuelled by a chemical reaction occurring at the end of each head: the hydrolysis of adenosine triphosphate (ATP). This is the reaction cycle in which an ATP nucleotide bound to a head is converted to an adenosine diphosphate (ADP) and an inorganic phosphate ($P_i$) nucleotide through attachment with actin. These reaction products are then released from the myosin-V head to allow another ATP molecule to bind and detachment from the track:

$$
ATP \rightarrow (ADP + P_i)_{\text{attached}} \rightarrow ADP_{\text{attached}} \rightarrow 0_{\text{attached}} \rightarrow ATP,
$$

(1.1.1)

where the 0 represents a absence of a nucleotide and the subscript $\text{attached}$ signifies that the head is attached to the track.

The rates of transition between chemical states are dependent on the concentrations of the relevant chemicals in the bulk. For example the $0_{\text{attached}} \rightarrow ATP$ rate increases as $ATP$ concentration increases because this increases the chance that the
head will come into contact with an ATP molecule. As a result of two coordinated $ATP$ hydrolysis reactions occurring at both heads, the myosin-V arms attach and detach from the track in a coordinated manner to achieve the stepping as shown in Figure 1.2.

![Figure 1.2: A schematic representation of the procession of myosin-V. Each head is the location of an ATP hydrolysis reaction. These are coordinated such that each head of the protein takes it in turns to detach from the actin and move to the next attachment site resulting in stepping down the track. The heads are labelled as $T$, $D$ or $D\cdot P_i$ representing bound ATP, ADP or ADP-$P_i$ nucleotides respectively. The attachment sites are represented with black dots.](image)

The myosin-V molecular motor is vital in many biological processes and therefore both determining how it functions and predicting how the protein will behave under certain conditions is an important part of understanding the inner workings of a biological cell. This has direct relevance for diseases in which the molecular motor is known to be defective e.g. the fatal Griscelli disease [39].

Firstly, in section 1.2 a summary of experimental work - particularly on the myosin-V protein - is presented. This is to aid in the understanding and analysis of experimental data in later chapters. Secondly in section 1.3 the unresolved issues as to how the motor functions are discussed to help identify what this thesis can contribute to the field. Finally existing mathematical modelling techniques and their limitations are explored in section 1.4.

### 1.2 Experimental Work on Myosin-V

Experimental study of myosin-V aims to determine structural and dynamical information about the motor protein. There are many different types of experiment, and each one yields certain data on the molecule under certain conditions. No single experimental method can give a fully comprehensive understanding of how the molecule functions but can give a small fragment of that picture. Combining
many of these experimental fragments together gives an approximate indication as to how aspects of the motor functions. The difficulty with analysing several types of experimental data lies in determining what conditions each experiment imposes. In this section a summary of experimental work that aims to understand how the molecule behaves under varying cell conditions is presented.

### 1.2.1 Experimental Methods

The main experimental methods are outlined in this section. However there are many variants on the methods discussed and some studies combine several different techniques explained here.

**Electron Microscopy**

Walker et al. [58] have used electron microscopy to generate some remarkable images of myosin-V during its procession (Figure 1.3). These images were able to provide some clear insights into both the structure and the behaviour of the molecule. The images confirmed that the actin filaments the motor moves along are helical. They were also able to confirm the deduction by Mehta et al. [37] that at each step myosin-V typically spans the helical repeat of the actin unlike those of some non-processive motor proteins [9].

![Figure 1.3: Electron microscopy images of myosin-V developed by Walker et al. [58]. The image second from bottom far right shows a myosin-V molecule in a high strain telemark stance.](image)

Under low ADP and ATP concentrations it was observed that few molecules were unattached to the actin and that most of those attached were only attached by one head. This implies that in the absence of these nucleotides it is difficult for
1.2. EXPERIMENTAL WORK ON MYOSIN-V

a myosin-V molecule to bond with both heads to the actin but that it is still favourable to remain attached to the actin.

Under high ATP and low ADP concentrations, few molecules were observed to be attached to the actin and of those that were, most were attached with only one head. As the concentration of ADP was increased it was observed that many more molecules became attached with both heads implying that either ADP slows the rate at which the molecule leaves the doubly attached state or ADP increases the rate at which molecules arrive in this state.

The doubly attached myosin-V molecules were separated into two main classes - those that appeared to have relatively little strain and those that were bent over in a similar manner to a skier in the telemark stance (see Figure 1.3). This was postulated to be related to the large mechanical step that the molecule was known to take [37] and so a possible model for the stepping mechanism for myosin-V was suggested.

Whilst electron microscopy may provide high spatial resolution images that give insights such as those discussed in this section, it has many limitations. For example a movement may be observed but the resolution is not sufficient to see the cause (such as a release of an ADP nucleotide etc.).

**Fluorescence**

Fluorescence is a process in which a molecule is excited by photons and re-emits them at a different but characteristic wavelength. Observation of the emitted light gives information about the molecule such as its position.

Mehta et al. [37] have used fluorescent imaging techniques combined with an optical trap to establish direct evidence that myosin-V is a processive molecular motor and takes approximately 36nm steps which span the helical repeat of the actin. Snyder et al. [45] used a similar method and found the average step size to be a little over 37nm.

A related technique is known as quantum dot labelling. A quantum dot is a molecule whose excitable electrons are all confined within all spatial dimensions. Different fluorescent quantum dots with different emission spectra have been used by Warshaw et al. [60] to label different heads of myosin-V molecules. The positions of each quantum dot were determined using a total internal reflectance microscope with an accuracy of ±6nm to demonstrate how the molecule moves in a hand-over-hand manner. In a similar manner Baker et al. [1] used a total internal reflectance fluorescence microscope to track fluorescent myosin-V molecules and determine their velocities and average run length before detachment from the track.
Fluorescent Polarisation

Fluorescent polarisation uses the process of fluorescence with polarised light: when a fluorescent molecule is excited by plane-polarised light then the light emitted will also be polarised and in the same plane, provided that the molecule remains stationary while it is in the excited state. If the molecule rotates out of this plane, the emitted light will be in a different plane. Fluorescent molecules can be attached to myosin-V so that this technique can be used to observe how a myosin-V molecule behaves dynamically.

It is thought that part of the process which generates the stepping force is a structural change in the molecule that involves the *telemark stance* tilt in the lead arm of the molecule. Forkey et al. [16] have used a fluorescent polarisation method to demonstrate directly how the molecule can tilt as it takes its step with time. This method also observed an average step size of 36-37nm for myosin-V.

Optical-Trap Spectrometry

Riev et al. [41] have used a method known as optical-trap spectrometry to study myosin-V. In this technique the cargo of the molecule can be controlled using an external light source. Motor proteins were chemically attached to a micrometer sized bead localised within a beam of light. The electromagnetic field was non-uniform to act as a potential well reproducing the forces experienced by the heads of the motor. Small displacements from the centre of the beam cause a restoring force of the order of pico-Newton. These displacements can be measured.

Riev et al. [41] have used this method to show that not only does a myosin-V molecule drag the bead out of the potential well, it does so in a series of discrete steps rather than in a continuous movement. Lang et al. [33] have developed more advanced methods to localise the bead, reducing the error in the experiments. Purcell et al. [40] used an optical trap to investigate how an external force on a single head changes its kinetics. Veigel et al. [54] have used optical-trap spectrometry to localise either end of an actin filament (rather than the cargo) and measure the degree to which the ends try to leave the potential well and thus the stiffness of the filament. As a class-V myosin molecule - with attached cargo - moves down the track, it pulls on the actin; this changes the strain within the filament and gives clues as to the intermolecular forces.

Kinetic Methods

Kinetic methods essentially determine the rate at which chemical reaction rates occur within a myosin-V molecule. Motor molecules with attached nucleotides (usually ADP or ATP) are mixed with actin filaments and another nucleotide in solution [57]. Relative concentrations of the nucleotides can be measured against time and from these the chemical reaction rates can be determined.

These experiments fall under two categories: single myosin head kinetic studies such
1.2. EXPERIMENTAL WORK ON MYOSIN-V

as those conducted by De La Cruz et al. [11] and double-headed studies such as those conducted by Rosenfeld and Sweeney [42]. Whilst the involvement of strain can be studied in double-headed experiments the results are harder to interpret than in single headed experiments as the chemical changes are a superposition of results from each head.

1.2.2 Measurement of Dynamic Properties

Experimental data for dynamic properties are presented in this section.

Step Size

It was established by Yildiz et al. [64] using florescent polarisation methods that the myosin-V molecule moves in a head-over-head motion along the actin filament. This was confirmed by Warshaw et al. [60] using quantum dot labelling and is distinct from other stepping mechanisms that other motor proteins use such as the inchworm method in which the heads don’t pass each other.

Mehta et al. [37] used fluorescent imaging techniques to show that class-V myosins take at least one large step of about 36nm for each head-over-head step. This approximates the helical repeat of the actin filament. Riev et al. [41] used optical trap spectrometry to show that the step size was 40.2 ± 6.4nm. Forkey et al. [16] used florescence polarisation to show the step size was 36-37nm and Warshaw et al. [60] used quantum dot labelling to confirm that the step size of a myosin-V molecule is about 36nm.

![Figure 1.4: The results of Baker et al. [1] showing the average velocity of myosin-V molecules against concentrations of ATP (at low [ADP]) and ADP (at 10 mM ATP). Reproduced from [1].](image-url)
CHAPTER 1. THE MYOSIN-V WALK

Velocities

The velocity of a myosin-V molecule is the speed at which it moves linearly along the actin filament. Baker et al. [1] tracked fluorescent myosin-V molecules using a total internal reflectance fluorescence microscope to establish velocities at various ATP and ADP concentrations. The results are shown in Figure 1.4. It should be noted that each data point requires measurements of many different molecules in order to get a sufficiently averaged measurement. Therefore there usually are only a few data points from which to infer an experimental trend. Here it appears that average molecular velocity decreases with increasing $[ADP]$ yet increases with increasing $[ATP]$.

Uemura et al. [52] and Gebhardt et al. [17] used optical trap spectrometry to investigate the relationship between the external force on a motor and its velocity for different nucleotide concentrations. These results are shown in Figure 1.5. For large negative forces velocity seems to remain constant, around stall force velocity drops quickly to zero and above stall force velocity appears to decrease inversely proportionally with increasing positive external forcing.

![Figure 1.5: The relationship between force and velocity using results from Uemura et al. [52], Gebhardt et al. [17] and Baker et al. [1]. Negative force represents pulling the molecule forwards along the track increasing velocity. This figure is taken from Vilfan [57].](image)

The force which causes the molecule to stop its procession is called the stall force. Mehta et al. [37] have shown that the stall force is about 3pN whilst Uemura et al. [52] have shown it to be 2.5-3pN. Cappello et al. [5] claim that 2.5pN is the upper bound of the stall force whereas Clemen et al. [7] have shown it to be lower, about 1.7pN. Close to the stall force, it is difficult to measure the velocities as the molecules disassociate from the track relatively quickly.
1.2. EXPERIMENTAL WORK ON MYOSIN-V

Run Lengths

The run length of a processive molecular motor is defined to be the average distance that it travels before dissociating from the actin track.

There have been many experimental studies that measure unloaded molecule run lengths. Mehta et al. [37] first found it to be about 2μm, Veigel et al. [55] showed it was 2.4 μm and Baker et al. [1] found it to be about 1μm. However Clemen et al. [7] suggested that it was closer to 300nm which Vilfan [57] suggests is due to the nature of the experimental set up: the actin being attached to a glass surface caused molecules to disassociate prematurely. Some authors have mathematically investigated the effect of force on unbinding from the track and the effect on the run length [14, 44], however study [7] is the only experimental work to date which investigates the dependence of run length upon external force, measuring it to be constant within the range -5pN to 1.5pN.

Baker et al. [1] tracked fluorescent myosin-V molecules using a total internal reflectance fluorescence microscope to establish run lengths against ATP and ADP concentrations. The results are shown in Figure 1.6: as [ADP] and [ATP] increase run length seems to decrease and attain a constant non zero value at saturating concentrations.

Dwell Times

As the molecular motor processes down a molecular filament it takes discrete steps (see Section 1.3.5). The time between each step is known as the dwell time - the time in which the molecule dwells in a particular configuration. Some models of myosin-V (see Section 1.4.3) state that the molecule goes through many more
conformational states than the pre and post-step states. In this case the time the molecule spends in a particular state is known as the dwell time of that state.

Mehta et al. [37] and Uemura et al. [52] investigated the force-dwell time relation for varying nucleotide concentrations using fluorescence and optical trap spectrometry respectively. The results are shown in Figure 1.7

![Figure 1.7: Dwell time dependence on force found by Mehta et al. [37] (circles, $[ATP]=2\text{mM}$) and Uemura et al. [52] (squares, $[ATP]=10\mu\text{M}$ and triangles, $[ATP]=1\text{mM}$). $[ADP]=200\mu\text{M}$ throughout. This diagram is adapted from Skau et al. [44] (circles, $[ATP]=2\text{mM}$) and Uemura et al. [52] (squares, $[ATP]=10\mu\text{M}$ and triangles, $[ATP]=1\text{mM}$). $[ADP]=200\mu\text{M}$ throughout. This diagram is adapted from Skau et al. [44]](image)

1.2.3 Kinetic Measurements

Single-headed

Kinetic methods which measure the chemical reaction rates between different states of a myosin-V molecule - such as those involved in nucleotide release from a head - have been used to determine the slowest reaction rate in the molecule’s stepping cycle. This is known as the rate-limiting step. De La Cruz et al. [11] suggest that it is the release of ADP from a head attached to the actin. Trybus et al. [49] conducted a similar study but measured a different ADP release rate. Wang et al. [59] suggest that ADP release is not rate limiting and that phosphate release is instead.

De La Cruz et al. [11] and Yengo et al. [62] have measured many other kinetic rates such as attachment to the actin rate, ATP hydrolysis rate, ADP release rate, ADP rebinding rate etc. However, rates associated with short-lived states - such as the rate of disassociation from actin - are not well known.
1.3. HOW DOES MYOSIN-V WALK?

Double-headed

Rosenfeld and Sweeney [42] studied the dependence of the kinetics of nucleotide release with respect to intramolecular strain and found that phosphate release was rapid therefore not dependent on strain and that ADP release depended strongly upon the strain.

Myosin-V with $ADP$ bound to both heads mixed with free actin filaments was found to attach with both heads to the actin and release $ADP$ from the front head at a rate of $0.3-0.4 \text s^{-1}$ and from the rear head at a rate of $28-30 \text s^{-1}$. This difference between the heads was attributed to the difference in the strain at each one, making the results distinct from single molecule studies.

Myosin-V with $ADP$ bound to the rear head and $P_i$ bound to the front mixed with actin was found to release phosphate from the front head at about $200 \text s^{-1}$. However this result was not shown directly and later experiments by Yengo and Sweeney [63] have shown that the value could be slightly lower.

1.3 How Does Myosin-V Walk?

There are many ideas inferred from experimental data as to how the myosin-V motor protein functions, many of which are in conflict with each other. It is important to understand what it is about myosin-V that is in dispute so that the importance of this work can be highlighted. This thesis aims to address some of these issues using theoretical methods.

1.3.1 Chemical States of the Myosin-V Heads

Each head of the myosin-V molecule acts as a site for an $ATP$ hydrolysis cycle. As described previously the coordination of both of the heads and both of the reactions leads to the molecule stepping head-over-head along the actin filament.

The chemical reaction is well known, but there is some dispute as to the nature of the attachment between the head and the actin. Rief et al. [41] postulate that there are three types of head attachment to the actin: without a nucleotide bound, bound to $ADP$ with $P_i$ and $ADP$-bound. Uemura et al. [52] suggest that these nucleotide bound states can be either strongly attached or weakly attached to the actin thus allowing extra transitions between weakly and strongly attached states which have different energetic parameters - for example a weakly attached state could be more vulnerable to the detachment of a head than a strongly attached state. Baker et al. [1] claim that their data suggests that only the $ADP$-$P_i$-bound head and the $ATP$-bound head can be weakly attached to the actin whereas the $ADP$-bound head is always attached to the actin strongly.

Models of myosin-V also differ on the types of attachment. Skau et al. [44] distinguish the types of attachment to the actin chemically - a head with only an $ADP$
bound nucleotide is strongly attached to the actin and a head with $ADP$ and $P_i$ is weakly attached to the actin. Wu et al. [61] however do not distinguish between the $ADP$ and the $ADP-P_i$ head states, instead they simply distinguish between a weakly attached front head and a strongly attached front head. The rear head is always considered to be strongly attached to the actin and thus has always released its phosphate.

1.3.2 Head Coordination

As the motor processes down the actin track it steps head over head [64] and each head passes through the sequence of chemical states described in the previous section: each passes through a phase of attachment to the actin, then detaches to move to the next actin site and attach, allowing the now rear head to detach and move to the next attachment site. This is shown in Figure 1.2. In order to achieve this feat, the heads must somehow coordinate their chemical reactions so that the front head does not detach from the actin before the rear head. This would either cause a futile step - in which the molecule simply dwells in that state until the front head reattaches - or cause the molecule to completely detach from the actin if the rear head detaches before the front head reattaches.

It has been suggested [55] that to prevent the front head from disassociating from the actin before the rear head, or at least allowing it to be less energetically favourable to do so, there must be some process which inhibits disassociation of the front head from the track. A double-headed experiment to determine the differences between the kinetics of ADP release from the front and rear heads was performed by Forgacs et al. [15]. They used a fluorescent $ATP$ analogue which had a much greater emission intensity when bound to a myosin head. They showed that release of ADP from the front head was 250-times slower than from the rear head. Veigel et al. [54] however claim that it is only 50-times slower.

Rosenfeld and Sweeney [42] suggest that strain within the molecule affects the rate of release of ADP from the heads. This is known as lead-head gating. Lower strain in the rear head causes the pocket in which the nucleotide is situated to open, allowing the $ADP$ to be released. As motor cannot detach chemically from the actin until it releases $ADP$, this makes detachment of the front head less energetically favourable.

Purcell et al. [40] have looked at individual heads of a molecule and put forwards and backwards forces (using an optical trap) on these heads when they were attached to actin. Backwards forces were shown to reduce the rate of ADP release and forwards forces were shown to accelerate it. They suggest that the backwards forces close the pocket therefore slowing the rate of $ADP$ release and the forwards forces open it, increasing the rate of $ADP$ release.
1.3. HOW DOES MYOSIN-V WALK?

1.3.3 Molecular Strain

It has been seen [58] that the two myosin-V heads are both attached to the actin at some point during its procession. The molecule must have some elasticity to allow this [57]. Many models assume that in order for the motor to take a step it must store strain in this doubly attached state so that when the rear head detaches the release of the stored strain causes the main step known as the powerstroke (see next section).

Veigel et al. [54] measured the bending stiffness of a myosin head attached to actin and found it to be about 0.2\,pNnm$^{-1}$. They constructed a model in which molecular strain was taken up by heads attaching to the actin and released to give a step. Rosenfeld and Sweeney [42] extended these ideas by showing that the release of phosphate from a head was very fast implying that there was little strain required in this process.

1.3.4 The Telemark State and the Powerstroke Step

Using electron microscopy (EM) Walker et al. [58] have observed a myosin-V molecule in a high strain doubly attached state with the front head bent over the actin (see Figure 1.3). This is known as the telemark state as it is similar to a skier in a telemark stance. Forkey et al. [16] have used a florescence polarisation method to demonstrate directly how the molecule can tilt as it takes its step with time. Snyder et al. [45] have shown that the bend is located within the leading arm (known as the lever arm) of the molecule.

Mehta et al. [37] have shown that a myosin-V molecule takes at least one large step in one processive cycle. This is known as the power stroke step or just power stroke. It has been postulated [4] that the molecule stores elastic strain when it is doubly attached to the actin, moving into the telemark state, and the release of this strain leads to the powerstroke.

There is disagreement as to whether the telemark state occurs before or after the powerstroke or even if it occurs on the main reaction pathway at all. Vilfan [57] suggests that the EM images taken by Burgess et al. [4] - which show that the telemark state is not as common as a non-telemark doubly attached state - provide evidence for the telemark state not being on the main pathway. However, the telemark state could simply be a short-lived state in the main reaction cycle.

1.3.5 Substeps

Due to the noise to which nanoscale measurements are subject, there is difficulty in determining the precise nature of the myosin stepping mechanism. It was first shown that myosin-V takes at least one large step [37]. Electron microscopy images [58] and optical trap techniques [41] suggest that this is when the rear head of the molecule jumps 36nm to the next attachment point on the actin. However, it has
been suggested that this postulated single large step can be split up into up to three substeps as a head moves from one attachment point to the next. A comparison between the different results is shown in Figure 1.8.

Veigel et al. [55] used their results to suggest that the molecule takes two steps to get from one attachment site to another: a 11nm diffusive step and a 25nm working step split into two mechanical phases. The working step involves a 5nm conformational change in which the molecule opens the ‘gate’ to release ADP from the rear head, followed by a 20nm powerstroke step in which the rear head is sprung forwards to then take the 11nm diffusive step and reattach at the next site along the track.

Uemura et al. [52] used an optical trap with advanced spatiotemporal measuring techniques to support the idea that the procession of the motor could involve two separate pathways. One in which two steps are taken - a 12nm diffusive step followed by a 24nm powerstroke - and the other where a single 36nm step is taken.

Cappello et al. [5] claim that they observed three steps: 5nm, 23nm, 8nm in that order using an advanced optical trap technique known as travelling-wave tracking. The first 5nm step was only observed at low ATP concentrations and involved a conformational change in the molecule putting it into the telemark state and
opening the gate. The second 23nm step is the powerstroke and the final 8nm step is the diffusive step.

There is dispute as to whether the telemark state precedes a step, but both sources that claim it does, show it is 5nm. The existence of the diffusive step is agreed upon, but there is controversy as to whether it occurs before or after the powerstroke. It has been shown that the diffusive step size is in the range 8-12nm. Each of the above models places the size of the powerstoke in the range 20-24nm.

1.3.6 The Rate-limiting Step

The slowest reaction rate in the molecule’s stepping cycle (known as the rate-limiting step) has been investigated in many kinetic studies. De La Cruz et al. [11] suggest that the rate limiting step is the release of ADP from an actin-attached head, whilst Wang et al. [59] and Trybus et al. [49] have conducted similar studies and found that it is not ADP release that is rate-limiting. De La Cruz et al. [10] confirm their earlier measurements and argue that if ADP release is rate-limiting then the motor tends to dwell in strongly attached rather than weakly attached states. This decreases the probability of detachment and therefore makes the motor more efficient - an evolutionary advantage.

1.3.7 Disassociation from Actin

There are several ideas as to what causes a myosin-V molecule to disassociate from the actin track. Unfortunately, there are few experimental studies that aim to determine the precise mechanism by which a molecule detaches.

Baker et al. [1] investigated detachment mechanisms by determining the run length as a function of chemical concentration. They produced a model to explain their results which involved constant rates of mechanical detachment of molecules in certain conformations. These are transitions in which the molecule is physically removed or knocked off the track.

An alternative idea to mechanical detachment is chemical detachment: the myosin heads lose coordination as the molecule takes a step and the only attached head detaches [42].

1.3.8 Hydrolysis Coupling

The strength of the coupling between ATP and the number of steps taken is not well known. An early study [46] suggested that one ATP molecule was consumed per step the molecule takes. Existing models also make this assumption [1, 44, 52, 61]. However it has been questioned whether this is true for molecules under large loads. Gebhardt et al. [17] suggested that whilst the backwards steps are not tightly coupled to the reversal of the reaction cycle, the forwards steps are tightly coupled to ATP hydrolysis.
1.4 Existing Mathematical Models of Myosin-V

There are several methods that are used for modelling the movement of myosin-V and other motor proteins: each has its own strengths and weaknesses. These are presented here in order to inform further modelling work presented in the rest of this thesis.

1.4.1 Brownian Ratchet Models for Myosin-V

In a Brownian ratchet model the molecular motor undergoes a form of biased diffusion along an actin filament [22] and through two or more spatially parallel, periodic energy surfaces as in Figure 1.9. This motion is described mathematically by a set of coupled Fokker-Planck equations. Once the molecule becomes stuck in a potential well, the hydrolysis of ATP drives the transition onto another energy surface allowing the molecule to continue its movement.

![Figure 1.9: Two energy surfaces in a possible Brownian ratchet model of myosin-V reproduced from Kolomeisky and Fisher work [29].](image)

These models capture the energetic nature of the protein very well, allowing one to explicitly analyse the energy changes. However, whilst there has been some work on ratchet models (for example Kolomeisky [30]) there are limitations associated with this method [29]. These models do not detail the mechanisms behind the motion, simply the resultant energetics, and so it is difficult to determine precise analytical results from them. Numerical results are computationally expensive to acquire and the nature of the energy surfaces themselves are difficult to determine due a lack of experimental data. Therefore, fitting these models to the data could easily give unreliable results.
Myosin-V data reveals hints as to the underlying mechanochemical changes the molecule undergoes - such as the images produced by Walker et al. [58] - which are difficult to include in ratchet models. Therefore, a class of models that can include these features is desirable.

1.4.2 Mechanochemical Models for Myosin-V

Another class of models are mechanochemical [32]. Using ideas of the mechanical behaviour of myosin-V and the chemical states of an individual head, then deriving results for the two headed myosin-V (reducing the number of parameters in the model by assuming both head domains are identical) the motors can be simulated using molecular dynamics techniques such as Monte Carlo simulation.

Lan and Sun [31] were the first to create such a model. They used Monte Carlo simulations to show that a simple mechanical simulation of myosin-V can reproduce many important experimental observations - such as the molecular velocity - for sensible parameter values. Vilfan [56] extends these results by treating the arms of the molecule as elastic to explain how the heads coordinate their behaviour. Craig and Linke [8] extend these two models to include disassociation from the actin and lead-head gating.

Whilst many mechanochemical models can have experimental data included that ratchet models cannot, calculation of many aspects of the protein and time dependence means results are extracted from numerically expensive molecular simulations. A class of models in which analytical results can be extracted would be beneficial.

1.4.3 Discrete Stochastic Models for Myosin-V

A discrete stochastic model of myosin-V assumes that as the molecule steps from one attachment site on the actin to the next, it passes through a sequence of distinct biomechanochemical states. Such a model encodes the probabilistic behaviour of an ensemble of molecules; any given molecule has a probability of occupying each state. These models have mechanochemical aspects to them however the discretisation assumption and looking at proteins in the statistical limit means that measurable quantities are calculated analytically. This is in contrast to the approach above where quantities must be computed from numerically expensive simulations. The discrete-stochastic method makes results far easier to obtain and allows the possibility of parameter fitting.

The nature of the stepping mechanism (which is disputed - see section 1.3.5) determines the form of the governing equations and the values of its coefficients. Considering a system with $n$ states, the rate of change with respect to time of the molecular state occupancy probability, $P(t)$, is governed by $n$ coupled linear master equations. Such a physical system is assumed to decay to the equilibrium steady state rapidly if molecular detachments are neglected. The steady-state solution
therefore can be used to determine the relevant mean behaviour of the molecules, for example their average velocity. Molecules move between states at time-independent rates; these rates are the coefficients of the governing master equations. Mechanochemical changes can be related to these models using Arrhenius expressions for the rate constants - essentially relating them to the energetics of the system. Two distinct states can have associated with them physical free energy differences, mechanical strain energy differences and chemical energy barriers all of which determine the rate of transition between them.

Discrete stochastic models can have the postulated underlying molecular mechanochemical changes encoded, can be matched to most experimental data and yield analytical results for the motor velocity and dispersion from the steady state solution. For these reasons this class of models is investigated in great depth.

1.4.4 Stepping Cycles

There have been many proposed myosin-V stepping mechanisms that can be described using a discrete stochastic model. Such a mechanism encodes a sequence of static mechanochemical conformations of the molecule each assigned a state in the model. The time evolution is encapsulated in the rate at which the molecule jumps discretely between conformations.

One of the first stepping models was suggested by Rief et al. [41]. This 5-state model simply encapsulated the different chemical states of the myosin-V head coordinated with a single large step. It identified the rate limiting step to be the transition 4\rightarrow 5 (see Figure 1.10).

Veigel et al. [55] extended the simple stepping cycle idea by introducing the idea that build-up of molecular strain causes the powerstroke. Each state has an associated molecular strain which is gained upon attachment with the actin and released when the molecule takes a step. The idea that the molecule can take several substeps was also introduced in the model.

Rosenfeld and Sweeney [42] extended the understanding of how the strain affects the procession of myosin-V. They showed that the release of phosphate from a head was very fast, implying that little strain is required for this process. Rosenfeld and Sweeney also investigated two possible ways a molecule could disassociate from the actin chemically. They determined that the release of ADP from the rear head when the front head is not attached to the track is the most likely pathway.

Rosenfeld and Sweeney also found that the rate of ADP release from the front head is 50 times slower than that from the rear head - a result of lead-head gating. Skau et al. [44] encoded this idea into a discrete-stochastic model with a stepping cycle that included the simple hydrolysis cycle (with strain), a powerstroke step and a diffusive step, chemical detachment and an alternative futile cycle in which the molecule releases ADP from its front head before the rear head and fails to take a step. This is shown in Figure 1.11. This model reproduces some experimental
1.4. EXISTING MATHEMATICAL MODELS OF MYOSIN-V

Figure 1.10: The 5-state model of myosin-V suggested by Rief et al. [41]. In state 1 the rear head is ATP-bound and not attached to the actin whilst the front head is ADP-bound and attached. The molecule takes a 36nm step and the front head becomes the rear head in state 2. In state 3 the ATP-bound unattached front head attaches to the actin and becomes ADP-P, bound. The phosphate is released upon transition to state 4 leaving both heads ADP bound and attached. The ADP from the rear head is released moving the molecule into state 5. The rear head then picks up ATP leaving the molecule in state 1 again but 36nm further down the track.

results (such as the dependence of run length on [ATP]) but does not reproduce others (run length dependence on [ADP]).

Baker et al. [1] and Uemura et al. [52] considered models to explain their experimental findings in which a molecule can undergo procession in several different ways. These different hydrolysis pathways involved the mechanochemical processes occurring in different orders. As previously mentioned Baker et al. [1] also produced experimental results that suggested there was a constant mechanical detachment rate from states with a weakly attached front head. Wu et al. [61] consolidated these ideas and encoded them into a discrete stochastic model. They investigated the pathway occupancy probability dependence on model parameters such as chemical concentration. The model created by Wu et al. is shown in Figure 1.12. It includes three hydrolysis pathways (each state with associated strain), a power-stroke step, a diffusive step and mechanical detachment. This model reproduces some experimental results (such as run length against [ADP] results from Baker et al. [1]) but does not reproduce others (run length against [ATP] from Baker et al. [1]).

Cappello et al. [5] suggested that an additional step exists - the 5nm step in which the molecule moves into the telemark state. This could be included in future models.
CHAPTER 1. THE MYOSIN-V WALK

Figure 1.11: The discrete stochastic model developed by Skau et al. [44] to describe myosin-V stepping. States 1-7, the chemical configuration of each head and the dominant direction in which the molecule moves through state space are labelled. There are two cycles, one main hydrolysis cycle in which the molecule moves forwards and takes a step (passing through states 1, 2, 3, 4 and 5) and one futile cycle in which the molecule fails to take a step (passing through states 2, 3, 4 and 6). The molecule can disassociate from the track from state 7, a result of a loss of coordination between the heads. Each state has an associated amount of molecular strain: states 1, 4, 5 and 6 have the maximum amount of strain (the molecule is in the telemark stance), states 2 and 7 have no associated strain and state 3 has an intermediate amount of strain.

Bierbaum et al. [2] used a discrete-stochastic framework without molecular strain to model an additional force-dependent stepping cycle to reproduce the high forcing results by Gebhardt et al. [17] in addition to reproducing the same experimental results as the Wu model. However this mechanism again failed to reproduce Baker et al’s run length against [ATP] result [1].

1.4.5 Calculating Dynamic Properties

In order to match a theoretical model against experimental data, dynamic properties of the molecular motor need to be derived from it. A discrete stochastic model can give steady-state probabilities, average motor velocities and their dispersions, run lengths and dwell times. There are many existing methods to calculate these quantities.

An approach based on Derrida [13] has proved useful in calculating exact steady states and dynamic properties for specific classes of system architectures of arbi-
1.4. EXISTING MATHEMATICAL MODELS OF MYOSIN-V

Figure 1.12: The discrete stochastic model of myosin-V developed by Wu et al. [61]. There are three pathways a molecule can follow through the states: path A (1, 2, 8, 9, 6, 1), path B (1, 7, 8, 9, 6, 1) and path C (1, 2, 3, 4, 5, 6, 1). Molecules can disassociate from the track mechanically from states 1, 4 and 9.

Flux balance [50] is a general method that allows calculation of quantities such as the velocity or the dwell times, without the need for explicit solutions for the state probabilities. However, it cannot give quantities such as the dispersion or randomness ratio, the reciprocal of which is the number of rate-limiting steps [48]. A method presented by Chemla et al. [6] allows the calculation of velocities, dispersions for any given biochemical pathway but cannot give general formulae. The calculations, particularly for large systems with reversible transitions, can require computationally expensive calculations and are mathematically quite involved. I shall present in Chapter 3 a novel method that permits more straightforward cal-

trary size. The simpler examples of these include single chains [13], parallel chains [26] and divided pathways [12]. Periodic parallel lattices have also been studied [47] in the limit of strong coupling between each branch. Each class can be modified to include branches and molecular detachment [27]. A severe limitation of this method is that the average velocity and its dispersion must be calculated individually for each system architecture. This is discussed further in the next chapter adapting a method for finding the steady-state probabilities of enzymatic networks graphically [18, 24].
calculation of these quantities.

Unfortunately it is not possible to calculate a unique dwell time distribution for a discrete stochastic model with reversible transitions [51]. Studies conducted before the release of this work have failed to account it [44].

Armed with a discrete stochastic model and derived dynamic properties, one can begin to judge the validity of a model against experiment.

1.4.6 Combining Theory and Experiment

Das and Kolomeisky [12] chose the parameters of their model by fitting their velocity curves against a set of experimental results. Whilst simplistic, this method ensures that if possible, the desired results are captured.

Wu et al. [61] used a variety of kinetic studies to choose the kinetic rates (the transition rates between the different states) in their model. Rates such as the mechanical detachment rate that had not been measured directly were taken from the estimates that Baker et al. [1] made so that the run length against $[ADP]$ model results were a best fit to the experimental data. Using these chosen values as well as others quoted from different sources Wu et al. [61] also reproduced these Baker et al. [1] results. This suggests that the majority of the kinetic rates do not affect the run length against $[ADP]$ result. One major disadvantage of simply quoting reaction rates from many different sources is that each experiment will take place under slightly different conditions, making the rates incompatible with each other.

Bierbaum et al. [2] built their model to have several free parameters that were relatively unconstrained and chose the remainder of the reaction rates based on experimental data. This allowed an excellent fit to a limited set of experimentally measured dynamic properties such as the molecular velocity, a poor fit to other dynamic properties such as the run length the underlying energetics as measured in kinetic studies [19]. This is discussed in further detail in Chapter 6.

Skau et al. [44] constructed an optimisation procedure to fit a model to many experimental results and ideas. This was a much more sophisticated way of treating the unknowns in a model as it took into account every piece of experimental data the authors thought was important and allowed for a margin of error in these measurements. This method and variants of it shall be what is used and discussed in the rest of this thesis for determining unknown parameters of a model.

In the study by Skau et al. [44] the kinetic rates were treated energetically using Arrhenius expressions and the number of parameters was minimised by separation into chemical and mechanical energy barriers and differences. The reaction rate to go from the less energetic state $i$ to the more energetic state $j$ (usually backwards) is given by

$$w_{i,j} = \tau^{-1}e^{-\beta(G_{i,j}^{\text{chem}} + \Delta G_{i,j})},$$  \hspace{1cm} (1.4.1)

where $\tau$ is the fundamental timescale of the reaction, $G_{i,j}^{\text{chem}}$ is the chemical energy
Energy Barriers & Other Parameters
\begin{tabular}{|c|c|c|c|c|c|}
\hline
$G_{T-Dw}^i$ & $G_{Dw-Ds}^i$ & $G_{Ds-E}^i$ & $G_{E-T}^i$ & $E_s$ & $\alpha E_s$ \\
\hline
0.3 & 10.4 & 15.7 & 5.8 & 12.8 & 5.4 \\
\hline
\end{tabular}

\begin{tabular}{|c|c|c|c|}
\hline
Energy Differences & \\
$\Delta G_{T-Dw}$ & $\Delta G_{Dw-Ds}$ & $\Delta G_{Ds-E}$ & $\Delta G_{E-T}$ \\
\hline
0.14 & 9.9 & -10 & 13.1 \\
\hline
\end{tabular}

Table 1.1: The optimised energy barriers, differences and the strain and gating values chosen using the optimisation routine developed by Skau et al. [44]. All units are in $k_B T$.

Barrier between the states $i$ and $j$, $\Delta G_{i,j}$ is the energy difference (both chemical and mechanical) between the states and the Boltzmann factor is denoted by $\beta = \frac{1}{k_B T}$. Rates to go from a more energetic state $j$ to the less energetic state $i$ (usually forwards) are given by

$$u_{j,i} = \tau^{-1} e^{-\beta G_{j,i}^i}.$$  \hfill (1.4.2)

Typically, forwards movement through a cycle always corresponds to transitions to lower energy states. If not at least $\Delta G_{i,j} < 0$ (one energy difference must be negative). A Monte Carlo optimisation routine explored energetic parameter space against selected experimental data to give the optimum point. Three distinct chemical reactions can occur during each hydrolysis cycle. The release of a phosphate as the front head attaches strongly to the filament has associated energy difference $\Delta G_{Dw-Ds}$ with energy barrier $G_{Dw-Ds}^i$, the reaction of an empty head with ATP from the bulk has energy difference $\Delta G_{E-T}$ and barrier $G_{E-T}^i$, the release of ADP from a head has energy difference $\Delta G_{Ds-E}$ with barrier $G_{Ds-E}^i$ and the weak attachment of a head to the actin track with conversion of ATP to ADP has energy difference $\Delta G_{T-Dw}$ and barrier $G_{T-Dw}^i$.

In the Skau model there exist three levels of molecular strain that contribute to the energy differences between states: unstrained, partly strained (energy $b E_{\text{strain}}$) and fully strained (energy $E_{\text{strain}}$). $b E_{\text{strain}}$ was determined assuming the assumption that the molecule is a Hookeian spring. There is also an additional energy barrier $\alpha E_s$ relating to the energy required to open the pocket in which the bound nucleotides sit. This is associated with the transition 4 $\to$ 6 in the Skau model (see Figure 1.11) and is assumed to be proportional to the strain of the molecule and gives the lead head gating effect discussed above.

Skau et al. [44] optimised their system to get the numerical results shown in table 1.1. Their method allows a model to be fitted to many experimental results at once and allows investigation of the robustness of a model. However, it is important to select appropriate experimental results to optimise against and identify what conditions each experimental result is subject to as some experimental results may be in conflict with others.
1.5 **Summary**

Myosin-V has been extensively studied. However, there are still many unresolved questions surrounding its stepping cycle. How many substeps does it take in a hydrolysis cycle and why do different experiments measure different numbers of substeps? Why do some models produce results that others cannot (for example the trend of run length against $[ADP]$ and $[ATP]$)? What are the alternative reaction pathways to the main cycle and how can they be identified?

The primary aim of this thesis is to create mathematical tools and apply them in an attempt to answer some of these questions. Extensions of existing theoretical methods of molecular motors are discussed (Chapter 2) and a novel and general method for calculating experimentally measurable quantities is presented (Chapter 3). A framework to compare competing models of myosin-V is described (Chapter 4) and this is used to gain novel insights into its underlying mechanisms (Chapters 5 and 6).
Chapter 2

Dynamic Properties of Molecular Motors

“Know how to solve every problem that has ever been solved”

Richard Feynman

Calculating the dynamic properties, such as state occupancy probabilities or average velocities from a discrete-stochastic model of a molecular motor, is vital to determine whether the model can reproduce experimental data. In this chapter I focus upon methods to do this.

If we consider a molecule stepping along an actin filament in one dimension in a discrete manner, there is a sequence of sites along the track at which the protein can be located. Assuming some sort of periodicity in the stepping the spacing between the sites becomes periodic also. This system can be described as a 1D uneven periodic lattice.

Assuming there are $n$ sites for each period of the physical-space lattice, the $i$th site in the $s$th period can be described as position $(i, s)$ with $i \in \{1, ..., n\}$ with the probability of being in position $(i, s)$ is denoted by $p_{i,s}$. Each site is connected to $n - 1$ sites forwards and $n - 1$ sites backwards, assuming that a molecule cannot jump a cycle length or longer for simplicity (jumping a cycle length is captured by this regardless as it has no effect on the governing equations [27]). For example, site $(0, s)$ is connected to each site $(i, s)$ and $(i, s - 1)$ for all $i \neq 0$. The forwards and backwards transition rates from site $i$ to site $j$ are $u_{ij}$ and $w_{ij}$ respectively. The distance from site $(0, s)$ to site $(i, s)$ is denoted by $d_i$ and $d_0 = 0$ and the total physical distance over the whole period is $d$. This system is shown in Figure 2.1.

At physical site $(i, s)$, the motor is in the same mechanochemical conformation or state as at site $(i, s')$ for all $i$, $s$ and $s'$. Therefore this system can also be described in state space in which there are only $n$ states. Denoting the probability of being
in state \( i \) by \( P_i \) we have

\[
P_i = \sum_s p_{i,s}.
\] (2.0.1)

Note that when \( i = 0 \), \( \sum_{j=0}^{i-1} \) is defined to give 0 and similarly when \( i = n - 1 \) for \( \sum_{j=i+1}^{n-1} \). In this way the stepping mechanism can be related to the movement of the protein through physical space.

A chosen stepping mechanism is encoded into a discrete-stochastic model by the form of the governing master equations and the values of the transition rates. A system of \( n \) states can be described by \( n \) master equations. In matrix form we have

\[
\dot{P}(t) = MP(t)
\] (2.0.2)

where \( M \) is a \( n \times n \) transition rate matrix and the \( i^{th} \) component of the vector of state occupancy probability \( \dot{P}(t) \) is \( P_i(t) \). The transition rate matrix can be written

\[
M_{kl} = W_{i,k} - \delta_{kl} \left( \delta_k + \sum_{j=1}^{n} W_{k,j} \right)
\] (2.0.3)

where \( \delta_{kl} \) is the Kronecker delta function, with the transition rate from state \( i \) to state \( j \) denoted by \( W_{i,j} \) and the detachment rate from state \( i \) being given by \( \delta_i \). We have \( W_{i,j} = 0 \) if there is no possible transition between state \( i \) and state \( j \).

Solving these master equations in the steady state (\( \dot{P} = 0 \)) give the steady-state state occupancy probabilities. Assuming the system is closed (that molecules are not leaving the track) there exist non-trivial solutions as probabilities must sum to unity. The case where this is not true is addressed in section 2.4.1.

Note that there is always one non-trivial steady-state solution to equations (2.0.2) coupled with probability normalisation. As \( \phi_0 = (1, ..., 1) \) is an eigenvector of \( M \) with eigenvalue \( \lambda_0 = 0 \) there is at least one steady-state. Probabilities summing to unity means that any solution must lie on a \( n \) dimensional hypersphere of radius
2.1. A TOY MODEL

Here I present a derivation to demonstrate how one could derive the velocity and dispersion for a specific system from the steady-state probabilities. This is based on the work of Hoyle [21].

2.1.1 Two States, One Branch

A simple two-state system with two substeps and only one reaction pathway is shown in Figure 2.2. In order to determine the experimentally observable quantities of this system the probability, $P_i$, of a molecule being in biomechanochemical state $i$ must be determined.

The master equations are given by

$$\dot{P}(t) = M P(t),$$

1, hence the steady state is not trivial. Solutions are never unstable as the trace of $M$ is always negative and the determinant is non negative; thus there can be at most one solution.

In this chapter a detailed analysis of the calculations for the steady-state probabilities, cycle fluxes and average velocities is presented as well as a derivation for the run length. The calculation for the dispersion is much more involved. Whilst many of the studies mentioned in this chapter do present dispersion derivations (each of which is specific to a given discrete stochastic architecture), I present a novel and general method of doing this in the next chapter so the pre-existing lengthy calculations are omitted here.

Firstly a simple toy model is used to demonstrate how to calculate the velocity and dispersion from the steady-state probabilities. Next, multiple existing methods and original extensions to calculate the probabilities for different system structures are explained. Finally, a method to account for molecules leaving the system known as renormalisation is discussed.

Figure 2.2: A simple periodic two-state system with two substeps and one pathways in state space. Forward rates are denoted by $u_i$, backward rates by $w_i$. On the physical lattice, all molecules at odd sites are in state 1 and all molecules at even sites are in state 2. The substep distances in physical space are denoted by $d_i$. 
where $M$ is the transition rate matrix,

$$
\begin{pmatrix}
-(u_1 + w_1) & u_2 + w_2 \\
(u_1 + w_1) & -(u_2 + w_2)
\end{pmatrix}
$$

and the probabilities satisfy the normalisation condition

$$P_1(t) + P_2(t) = 1 \tag{2.1.3}$$

because the detachment rates $\delta_1$ and $\delta_2$ are both zero.

Systems that do not satisfy this condition can be renormalised into a system that does [27].

The rate of change of the state occupancy probabilities for molecules on the track is

$$\dot{P}_1 = bP_2 - aP_1, \tag{2.1.4}$$
$$\dot{P}_2 = aP_1 - bP_2, \tag{2.1.5}$$

where $a = u_1 + w_1$ and $b = u_2 + w_2$. $P_1$ and $P_2$ in the steady-state can be determined from equations (2.4.16) and (2.4.17) and probability normalisation to be

$$P_1 = \frac{b}{a + b}, \tag{2.1.6}$$
$$P_2 = \frac{a}{a + b}. \tag{2.1.7}$$

**Velocity**

The average velocity of molecules in such a system in the steady-state can be determined by considering physical site occupancy probabilities $p_{1,s}$ of a molecule as it moves along an uneven one-dimensional lattice in physical space. These are governed by the equations

$$\frac{dp_{2,s}}{dt} = u_1 p_{1,s} + w_1 p_{1,s+1} - (u_2 + w_2) p_{2,s}, \tag{2.1.8}$$
$$\frac{dp_{1,s+1}}{dt} = u_2 p_{2,s} + w_2 p_{2,s+1} - (u_1 + w_1) p_{1,s+1}. \tag{2.1.9}$$

Here each $s$ is associated with each repeat of the physical lattice of even and odd positions.

The average displacement of a molecule on the track is given by

$$<x> = \sum_s p_{1,s+1} [(d_1 + d_2)s - d_2] + \sum_s p_{2,s}(d_1 + d_2)s,$$

$$= (d_1 + d_2) \left[ \sum_s sp_{2,s} + \sum_s sp_{1,s+1} \right] + d_2 P_2,$$

$$= (d_1 + d_2) [X + Y] - d_2 P_2. \tag{2.1.10}$$
2.1. A TOY MODEL

where

\[ Y \equiv \sum_s sp_{1,s+1} \text{ and } X \equiv \sum_s sp_{2,s} \text{ satisfy} \]

\[
\frac{dX}{dt} = (w_1 + u_1)Y + u_1 P_1 - (w_2 + u_2)X, \quad \text{ (2.1.11)}
\]

\[
\frac{dY}{dt} = (w_2 + u_2)X - w_2 P_2 - (w_1 + u_1)Y. \quad \text{ (2.1.12)}
\]

The velocity of molecule in such a system in the steady state is

\[
V = \frac{d < x >}{dt} = (d_1 + d_2) \left[ \frac{dX}{dt} + \frac{dY}{dt} \right],
\]

\[
= (d_1 + d_2) \left[ u_1 P_1 - w_2 P_2 \right],
\]

\[
= (d_1 + d_2) \frac{u_1 u_2 - w_1 w_2}{u_1 + u_2 + w_1 + w_2}. \quad \text{ (2.1.13)}
\]

This is just the flux multiplied by the total step size. Interestingly flux of molecules within the system is similar to Kirchhoff’s laws for current conservation. There appears to be no analogy for potentials

Dispersions

The dispersion is a measure of the width of the distribution of the velocities around mean \( V \). The dispersion requires

\[
\frac{d < x^2 >}{dt} = 2 \nu \left[ (d_1 + d_2) [X + Y] - d_2 P_2 \right], \quad \text{ (2.1.14)}
\]

\[
= 2(d_1 + d_2)^2 [X + Y] [u_1 P_1 - w_2 P_2] - 2d_2(d_1 + d_2) P_2 [u_1 P_1 - w_2 P_2], \quad \text{ (2.1.15)}
\]

and

\[
\frac{d < x^2 >}{dt} = \sum_s \frac{dp_{2,s}}{dt} [(d_1 + d_2) s - d_2]^2 + \sum_s \frac{dp_{1,s+1}}{dt} (d_1 + d_2)^2 s^2,
\]

\[
= \sum_s (u_1 p_{1,s} + w_1 p_{1,s+1} - (u_2 + w_2) p_{2,s}) [(d_1 + d_2) s - d_2]^2
\]

\[
+ \sum_s (u_2 p_{2,s} + w_2 p_{2,s+1} - (u_1 + w_1) p_{1,s+1}) (d_1 + d_2)^2 s^2 \quad \text{(2.1.16)}
\]

\[
= \sum_s (w_1 p_{1,s+1} - (u_2 + w_2) p_{2,s}) [(d_1 + d_2) s - d_2]^2
\]

\[
+ \sum_s (u_2 p_{2,s} - (u_1 + w_1) p_{1,s+1}) (d_1 + d_2)^2 s^2
\]

\[
+ \sum_s u_1 p_{1,s+1} [(d_1 + d_2) (s + 1) - d_2]^2
\]

\[
+ \sum_s w_2 p_{2,s} (d_1 + d_2)^2 (s - 1)^2 \quad \text{(2.1.17)}
\]
motion. This can be obtained from the above expressions. The randomness ratio and so therefore the dispersion is 

\[ D = (d_1 + d_2)^2 \sum_s [(2s + 1)u_1p_{1,s+1} - (2s - 1)w_2p_{2,s}] \]

\[ -2d_2(d_1 + d_2) [u_1P_1 + (u_1 + w_1)Y - (u_2 + w_2)X] \]

\[ + d_2^2 [(u_1 + w_1)P_1 - (u_2 + w_2)P_2] \]

\[ = (d_1 + d_2)^2 [2u_1Y - w_2X] + u_1P_1 + w_2P_2 \]

\[ - 2d_2(d_1 + d_2) [u_1P_1 + (u_1 + w_1)Y - (u_2 + w_2)X]. \] (2.1.18)

Assuming the \( P_i \) are in the steady state and \( X(0) = Y(0) = 0 \) (at time 0 all molecules are located in the \( s = 0 \) period), the solutions to equations 2.1.11 and 2.1.12 are

\[ Y = \frac{(w_2 + u_2)(u_1P_1 - w_2P_2)}{w_1 + u_1 + w_2 + u_2} t \]

\[ - \frac{(w_2 + u_2)u_1P_1 + (w_1 + u_1)w_2P_2}{(w_1 + u_1 + w_2 + u_2)^2} (1 - e^{-t(w_1+u_1+w_2+u_2)}) , \] (2.1.20)

\[ X = \frac{(w_1 + u_1)(u_1P_1 - w_2P_2)}{w_1 + u_1 + w_2 + u_2} t \]

\[ + \frac{(w_2 + u_2)u_1P_1 + (w_1 + u_1)w_2P_2}{(w_1 + u_1 + w_2 + u_2)^2} (1 - e^{-t(w_1+u_1+w_2+u_2)}) . \] (2.1.21)

Therefore the dispersion is

\[ D \equiv \frac{1}{2} \lim_{t \to \infty} \frac{d}{dt} \left( <x^2> - <x>^2 \right) , \]

\[ = (d_1 + d_2)^2 \lim_{t \to \infty} [(u_1Y - w_2X) - (X + Y)(u_1P_1 - w_2P_2)] \]

\[ + \frac{1}{2}(d_1 + d_2)^2 [u_1P_1 + w_2P_2] \]

\[ - d_2(d_1 + d_2) [u_1P_1 - u_1P_1P_2 + w_2P_2^2] \]

\[ - d_2(d_1 + d_2) \lim_{t \to \infty} [(u_1 + w_1)Y - (u_2 + w_2)X] , \] (2.1.22)

and so

\[ = -(d_1 + d_2)^2 \left[ (u_1 + w_2)\frac{(w_2 + u_2)u_1P_1 + (w_1 + u_1)w_2P_2}{(w_1 + u_1 + w_2 + u_2)^2} \right] \]

\[ + \frac{1}{2}(d_1 + d_2)^2 [u_1P_1 + w_2P_2] \]

\[ - d_2(d_1 + d_2) [u_1P_1 - u_1P_1P_2 + w_2P_2^2] \]

\[ + d_2(d_1 + d_2) \left[ \frac{(w_2 + w_2)u_1P_1 + (u_1 + w_1)w_2P_2}{u_1 + u_1 + w_2 + u_2} \right] . \] (2.1.23)

The randomness ratio \( \rho = \frac{2D}{V_d} \) is a measure of the deviations from constant-speed motion. This can be obtained from the above expressions.
2.1. A TOY MODEL

Figure 2.3: A simple two-state system with two substeps and two pathways. Forward rates are denoted by $u_i$, backward rates by $w_i$ and the alternative pathway rates have primes. The substep distances in physical space are denoted by $d_i$. 

2.1.2 Two States, Two Branches

The following is an extension of the work by Hoyle [21] and demonstrates how a small change in the system changes the derived quantities. A simple two-state system with two substeps and two reaction pathways is shown in Figure 2.3. Again, in order to determine the experimentally observable quantities of this system the probability, $P_i$, of a molecule being in biomechanochemical state $i$ must be determined.

The master equations and their steady state solutions remain the same as in the two-state one-branch case, however the dynamic properties are calculated differently.

Velocity

The average velocity of molecules in such a system in the steady-state can be determined by considering physical site occupancy probabilities $p_{i,s}$ of a molecule as it moves along an uneven one-dimensional lattice in physical space. These are governed by the equations

$$\frac{dp_{2,s}}{dt} = u_1p_{1,s} + w_1p_{1,s+1} - (u_2 + w_2)p_{2,s}, \quad (2.1.24)$$

$$\frac{dp_{1,s+1}}{dt} = u_2p_{2,s} + u'_1p_{1,s} + w_2p_{2,s+1} + w'_1p_{2,s+3} - (u_1 + u'_1 + w_1 + w'_1)p_{1,s+1}, \quad (2.1.25)$$

where the primed rates denote jumps from one repeat of the system to another.

Here each $s$ is associated with a physical location each being either an even or an odd position.
The average displacement of a molecule on the track is given by

\[
< x > = \sum_s p_{1,s+1} [(d_1 + d_2)s - d_2] + \sum_s p_{2,s}(d_1 + d_2)s,
\]

\[
= (d_1 + d_2) \left[ \sum_s sp_{2,s} + \sum_s sp_{1,s+1} \right] - d_2P_2,
\]

\[
= (d_1 + d_2) [X + Y] - d_2P_2, \quad (2.1.26)
\]

where \( Y \equiv \sum_s sp_{1,s+1} \) and \( X \equiv \sum_s sp_{2,s} \) satisfy

\[
\frac{dX}{dt} = (w_1 + u_1)Y + u_1P_1 - (w_2 + u_2)X, \quad (2.1.27)
\]

\[
\frac{dY}{dt} = (w_2 + u_2)X + (u'_1 - w'_1)P_1 - w_2P_2 - (w_1 + u_1)Y. \quad (2.1.28)
\]

The solutions are

\[
Y = \frac{(w_2 + u_2) (u_1P_1 - w_2P_2 + (u'_1 - w'_1)P_1)}{w_1 + u_1 + w_2 + u_2}
\]

\[
	imes \left( 1 - e^{-t(w_1+u_1+w_2+u_2)} \right), \quad (2.1.29)
\]

\[
X = \frac{(w_1 + u_1) (u_1P_1 - w_2P_2 + (u'_1 - w'_1)P_1)}{w_1 + u_1 + w_2 + u_2}
\]

\[
+ \frac{(w_2 + u_2) u_1P_1 - (w_1 + u_1) ((u'_1 - w'_1)P_1 - w_2P_2)}{w_1 + u_1 + w_2 + u_2}
\]

\[
	imes \left( 1 - e^{-t(w_1+u_1+w_2+u_2)} \right), \quad (2.1.30)
\]

assuming \( \dot{P}_1 = 0 \).

The velocity of molecule in such a system in the steady state is

\[
V = \frac{d < x >}{dt} = (d_1 + d_2) \left[ \frac{dX}{dt} + \frac{dY}{dt} \right],
\]

\[
= (d_1 + d_2) [(u_1 + u'_1 - w'_1)P_1 - w_2P_2],
\]

\[
= (d_1 + d_2) \frac{u_1u_2 - w_1w_2 + (u'_1 - w'_1)(u_2 + w_2)}{u_1 + u_2 + w_1 + w_2}. \quad (2.1.31)
\]

This is just the flux multiplied by the total step size.

**Dispersion**

The dispersion requires

\[
\frac{d < x >^2}{dt} = 2v [(d_1 + d_2) [X + Y] - d_2P_2], \quad (2.1.32)
\]

\[
= 2(d_1 + d_2)^2 [X + Y] [(u_1 + u'_1 - w'_1)P_1 - w_2P_2]
\]

\[
-2d_2(d_1 + d_2)P_2 [(u_1 + u'_1 - w'_1)P_1 - w_2P_2], \quad (2.1.33)
\]
and

\[
\frac{d < x^2 >}{dt} = \sum_s \frac{dp_{2,s}}{dt} [(d_1 + d_2)s - d_2]^2 + \sum_s \frac{dp_{1,s+1}}{dt} (d_1 + d_2)^2 s^2,
\]

\[
= \sum_s [u_1 p_{1,s} + w_1 p_{1,s+1} - (u_2 + w_2)p_{2,s}] [(d_1 + d_2)s - d_2]^2
\]

\[
+ \sum_s [u_2 p_{2,s} + u'_1 p_{1,s} + w_2 p_{2,s+1} + w'_1 p_{2,s+3}] (d_1 + d_2)^2 s^2
\]

\[
- \sum_s [(u_1 + u'_1 + w_1 + w'_1) p_{1,s+1}] (d_1 + d_2)^2 s^2
\]

\[
= \sum_s (w_1 p_{1,s+1} - (u_2 + w_2)p_{2,s}) [(d_1 + d_2)s - d_2]^2
\]

\[
+ \sum_s [u_2 p_{2,s} - (u_1 + u'_1 + w_1 + w'_1) p_{1,s+1}] (d_1 + d_2)^2 s^2
\]

\[
+ \sum_s u_1 p_{1,s+1} [(d_1 + d_2)(s + 1) - d_2]^2
\]

\[
+ (d_1 + d_2)^2 \sum_s (u'_1 p_{1,s+1}(s + 1)^2 + w_2 p_{2,s}(s - 1)^2 + w'_1 p_{1,s+1}(s - 1)^2),
\]

thus

\[
= (d_1 + d_2)^2 \sum_s [(2s + 1)u_1 p_{1,s+1} + (2s + 1)u'_1 p_{1,s+1}]
\]

\[
- (d_1 + d_2)^2 \sum_s [(2s - 1)w_2 p_{2,s} - (2s - 1)w'_1 p_{1,s+1}]
\]

\[
- 2d_2 (d_1 + d_2) [u_1 P_1 + (u_1 + w_1)Y - (u_2 + w_2)X]
\]

\[
+ d_2^2 [(u_1 + w_1)P_1 - (u_2 + w_2)P_2]
\]

\[
= (d_1 + d_2)^2 [2((u_1 + u'_1 - w'_1)Y - w_2 X) + (u_1 + u'_1 - w'_1)P_1 + w_2 P_2]
\]

\[
- 2d_2 (d_1 + d_2) [u_1 P_1 + (u_1 + w_1)Y - (u_2 + w_2)X].
\]

Therefore the dispersion is

\[
D \equiv \frac{1}{2} \lim_{t \to \infty} \frac{d}{dt} (d < x^2 > - < x >^2),
\]

\[
= (d_1 + d_2)^2 \lim_{t \to \infty} \left( [(u_1 + u'_1 - w'_1)Y - w_2 X] - [X + Y] [(u_1 + u'_1 - w'_1)P_1 - w_2 P_2] \right)
\]

\[
+ \frac{1}{2} (d_1 + d_2)^2 [(u_1 + u'_1 - w'_1)P_1 + w_2 P_2]
\]

\[
- 2d_2 (d_1 + d_2) [u_1 P_1 - (u_1 + u'_1 - w'_1)P_1 P_2 + w_2 P_2^2]
\]

\[
- d_2 (d_1 + d_2) \lim_{t \to \infty} [(u_1 + w_1)Y - (u_2 + w_2)X],
\]

\[
= \lim_{t \to \infty} \left( (d_1 + d_2)^2 \left( [(u_1 + u'_1 - w'_1)Y - w_2 X] - [(u_1 + u'_1 - w'_1)P_1 - w_2 P_2]^2 t \right) \right)
\]

\[
+ \frac{1}{2} (d_1 + d_2)^2 [(u_1 + u'_1 - w'_1)P_1 + w_2 P_2]
\]

\[
- 2d_2 (d_1 + d_2) [u_1 P_1 - (u_1 + u'_1 - w'_1)P_1 P_2 + w_2 P_2^2]
\]

\[
+ d_2 (d_1 + d_2) \left( \frac{(u_2 + w_2)u_1 P_1 + (u_1 + w_1)(u'_1 - w'_1)P_1 + w_2 P_2}{u_1 + w_1 + u_2 + w_2} \right).
\]
and so

\[
\begin{align*}
= & \quad (d_1 + d_2)^2 \left[ (u_1 + w_2) \frac{(w_2 + u_2) u_1 P_1 + (w_1 + u_1) ((u'_1 - w'_1) P_1 + w_2 P_2)}{(w_1 + u_1 + w_2 + u_2)^2} \right] \\
& \quad + \frac{1}{2} (d_1 + d_2)^2 \left[ (u_1 + u'_1 - w'_1) P_1 + w_2 P_2 \right] \\
& \quad - d_2 (d_1 + d_2) \left[ u_1 P_1 - (u_1 + u'_1 - w'_1) P_1 P_2 + w_2 P_2 ^2 \right] \\
& \quad + d_2 (d_1 + d_2) \left[ \frac{(u_2 + w_2) u_1 P_1 + (w_1 + u_1)((u'_1 - w'_1) P_1 + w_2 P_2)}{u_1 + w_1 + w_2 + u_2} \right]. \quad (2.1.37)
\end{align*}
\]

Deriving results for individual systems can be time consuming. In the rest of this chapter I focus on methods for arbitrarily sized systems.

## 2.2 Existing Work on System Structures

### 2.2.1 Single Pathway Model

The simplest \textit{general} stepping mechanism architecture is one in which there is a single reaction pathway with \( n \) states in which the molecule moves a total distance \( d \) in one cycle as originally studied by Derrida [13]. A molecule can only pass from one state to an adjacent state and the system is spatially periodic.

The master equations for such a system are given by

\[
\dot{P}_j = u_{j-1} P_{j-1} + w_{j+1} P_{j+1} - (u_j + w_j) P_j, \quad (2.2.1)
\]

with \( j \in \{1, \ldots, n\} \), state \( 1 \equiv n + 1 \), \( u_j \) being the rate from state \( j \) to state \( j + 1 \) and \( w_j \) being the rate from state \( j \) to state \( j - 1 \). This system is shown in Figure 2.4.

\begin{center}
Figure 2.4: A simple periodic reaction network.
\end{center}

Following Derrida [13], one can define

\[
\begin{align*}
r_j &= \frac{1}{u_j} \left( 1 + \sum_{i=j+1}^{n-1} \prod_{l=1}^{j+l} \frac{w_i}{u_i} \right) \quad (2.2.2)
\end{align*}
\]
again taking indices modulo \( n \) and see that
\[
\begin{align*}
&u_{j-1}r_{j-1} + w_{j+1}r_{j+1} - u_jr_j - w_jr_j \\
&= \frac{w_{j+1}}{u_{j+1}} \left( 1 + \sum_{l=1}^{j-1} \prod_{i=j+2}^{j+l} \frac{w_i}{u_i} \right) + \left( 1 + \sum_{l=1}^{j-1} \prod_{i=j}^{j+l} \frac{w_i}{u_i} \right) \\
&\quad - \left( 1 + \sum_{l=1}^{j-1} \prod_{i=j+1}^{j+l} \frac{w_i}{u_i} \right) - w_j \left( 1 + \sum_{l=1}^{j-1} \prod_{i=j}^{j+l} \frac{w_i}{u_i} \right) \\
&= \left( \frac{w_{j+1}}{u_{j+1}} + \sum_{l=1}^{j-1} \prod_{i=j+1}^{j+l+1} \frac{w_i}{u_i} \right) + \left( \sum_{l=1}^{j-1} \prod_{i=j}^{j+l} \frac{w_i}{u_i} \right) \\
&\quad - \left( \sum_{l=1}^{j-1} \prod_{i=j}^{j+l} \frac{w_i}{u_i} \right) - \left( w_j + \sum_{l=1}^{j-1} \prod_{i=j+1}^{j+l+1} \frac{w_i}{u_i} \right) \\
&= \frac{w_{j+1}}{u_{j+1}} - \frac{w_j}{u_j} + \sum_{l=1}^{j-1} \left( \prod_{i=j+1}^{j+l+1} \frac{w_i}{u_i} - \prod_{i=j}^{j+l} \frac{w_i}{u_i} + \prod_{i=j}^{j+l+1} \frac{w_i}{u_i} - \prod_{i=j}^{j+l} \frac{w_i}{u_i} \right) \\
&= \prod_{i=j+1}^{j+n} \frac{w_i}{u_i} - \prod_{i=j}^{j+n-1} \frac{w_i}{u_i} \\
&= 0. \quad (2.2.3)
\end{align*}
\]

The steady state of Eqn. 2.2.1 is then given by
\[
P_j = Nr_j, \quad (2.2.4)
\]
since then
\[
\dot{P}_j = N (u_{j-1}r_{j-1} + w_{j+1}r_{j+1} - (u_j + w_j)r_j) = 0, \quad (2.2.5)
\]
with the normalisation condition
\[
\sum_{j=1}^{n} P_j = 1, \quad (2.2.6)
\]
where \( N \) is the normalisation constant given by
\[
N = \frac{1}{\sum_{j=1}^{n} r_j}. \quad (2.2.7)
\]

The steady state gives certain quantities describing the system. The flux, \( J \), of molecules between states, their average velocity, \( V \), and the dispersion, \( D \), of the
velocity around the mean value. The flux around the cycle is given by

\[
J = \frac{V}{d} = P_1 u_1 - P_2 w_2,
\]

\[
= \frac{1}{\sum_{j=1}^{n} r_j} (u_1 r_1 - w_2 r_2), \tag{2.2.8}
\]

The difference term can be simplified

\[
u_1 r_1 - w_2 r_2 = 1 + \sum_{k=1}^{n-1} \prod_{i=2}^{k} \frac{w_i}{u_i} - \frac{w_2}{u_2} \left(1 + \sum_{k=1}^{n-1} \prod_{i=3}^{k} \frac{w_i}{u_i}\right),
\]

\[
= 1 + \sum_{k=1}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}} - \frac{w_2}{u_2} \left(1 + \sum_{k=1}^{n-1} \prod_{i=1}^{k} \frac{w_{i+2}}{u_{i+2}}\right),
\]

\[
= 1 + \frac{w_2}{u_2} + \sum_{k=2}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}} - \left(\frac{w_2}{u_2} + \sum_{k=1}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}}\right),
\]

\[
= 1 + \sum_{k=1}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}} - \sum_{k=1}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}},
\]

\[
= 1 + \sum_{k=2}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}} - \sum_{k=2}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}},
\]

\[
= 1 + \sum_{k=2}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}} - \sum_{k=2}^{n-1} \prod_{i=1}^{k} \frac{w_{i+1}}{u_{i+1}} - \prod_{i=1}^{n} \frac{w_{i+1}}{u_{i+1}},
\]

\[
= 1 - \prod_{i=1}^{n} \frac{u_i}{u_i}
\]

\[
\text{giving}
\]

\[
J = \frac{1}{\sum_{j=1}^{n} r_j} \left(1 - \prod_{i=1}^{n} \frac{u_i}{w_i}\right), \tag{2.2.10}
\]

where the reciprocal sum of \(r_j\) and product of forward reaction rates normalise the probabilities. This is exactly the result shown by Derrida [13].

The probabilities and the flux (and therefore also the velocity) have been explicitly calculated from a single-chain discrete stochastic model. Fitting such a model against experiment involves choosing the reaction rates so that these dynamic properties (and the rates) match up with experimental data. This dispersion calculation has been omitted for simplicity. Further analysis of different architectures in the rest of the chapter follow the same principle.

### 2.2.2 Branching Models

The results for a simple periodic system have been extended by Kolomeisky and Fisher [27] by considering branching states. Let each state \(j\) have a transition to a
finite branch of states $j_{k}, k \in [1,m_{j}]$, separate from the main cycle. State $j_{k}$ is state $j$ on the main cycle and only nearest neighbour transitions are allowed on the branch, i.e. only $u_{j_{k}}$ and $w_{j_{k}}$ transitions exist from a state $j_{k}$. This system is shown in Figure 2.5.

![Figure 2.5: A simple periodic reaction network with branching from every state.](image)

The master equations for states on the main cycle are

$$
\dot{P}_{j} = u_{j-1}P_{j-1} + w_{j+1}P_{j+1} + w_{j}P_{j,1} - (u_{j} + w_{j} + u_{j,0})P_{j},
$$

(2.2.11)

where $P_{j} \equiv P_{j0}$. The master equations for states on the branches are given by

$$
\dot{P}_{j_{k}} = u_{j_{k-1}}P_{j_{k-1}} + w_{j_{k+1}}P_{j_{k+1}} - (u_{j_{k}} + w_{j_{k}})P_{j_{k}},
$$

(2.2.12)

for $k \in [1,m_{j} - 1]$ and

$$
\dot{P}_{j_{m_{j}}} = u_{j_{m_{j}-1}}P_{j_{m_{j}-1}} - w_{j_{m_{j}}}P_{j_{m_{j}}},
$$

(2.2.13)

Using the ansatz

$$
P_{j} = Nr_{j},
$$

(2.2.14)

where $r_{j}$ is defined in equation 2.2.2, equation 2.2.11 becomes

$$
\dot{P}_{j} = w_{j,1}P_{j,1} - u_{j,0}P_{j}.
$$

(2.2.15)

If

$$
P_{j_{k}} = \prod_{i=1}^{k} \frac{u_{j_{k-l}}}{w_{j_{k-l}}},
$$

(2.2.16)
for \( k \in [1, m_j] \), then equation 2.2.15 becomes
\[
\dot{P}_j = w_{j,(1)} P_{j,(1)} - u_{j,(0)} P_j,
\]
\[
= u_{j,(0)} P_j - u_{j,(0)} P_j,
\]
\[
= 0.
\] (2.2.17)

Equation 2.2.12 becomes
\[
\dot{P}_j(k) = u_{j,(k-1)} P_{j,(k-1)} + w_{j,(k+1)} P_{j,(k+1)} - (u_{j,(k)} + w_{j,(k)}) P_{j,(k)},
\]
\[
= \left[ u_{j,(k-1)} \prod_{i=1}^{k-1} \frac{u_{j,(i-1)}}{w_{j,(i)}} + w_{j,(k+1)} \prod_{i=1}^{k+1} \frac{u_{j,(i-1)}}{w_{j,(i)}} - (u_{j,(k)} + w_{j,(k)}) \prod_{i=1}^{k} \frac{u_{j,(i-1)}}{w_{j,(i)}} \right] P_j,
\]
\[
= 0,
\] (2.2.18)

for \( k \in [1, m_j - 1] \). Equation 2.2.13 is evaluated to be
\[
\dot{P}_j(m_j) = u_{j,(m_j-1)} P_{j,(m_j-1)} - w_{j,(m_j)} P_{j,(m_j)},
\]
\[
= \left[ u_{j,(m_j-1)} \prod_{i=1}^{m_j-1} \frac{u_{j,(i-1)}}{w_{j,(i)}} - w_{j,(m_j)} \prod_{i=1}^{m_j} \frac{u_{j,(i-1)}}{w_{j,(i)}} \right] P_j,
\]
\[
= \left[ w_{j,(m_j)} \prod_{i=1}^{m_j} \frac{u_{j,(i-1)}}{w_{j,(i)}} - w_{j,(m_j)} \prod_{i=1}^{m_j} \frac{u_{j,(i-1)}}{w_{j,(i)}} \right] P_j,
\]
\[
= 0.
\] (2.2.19)

Normalisation gives the unity condition and thus the condition for the steady state is satisfied.

The flux for the system with branching becomes
\[
J = \frac{V}{d} = P_1 u_1 - P_2 w_2,
\]
\[
= \frac{1}{\sum_{j=1}^{n} r_j \left( 1 + \sum_{k=1}^{m_j} \prod_{i=1}^{k} \frac{u_{j,(i-1)}}{w_{j,(i)}} \right)} \left( u_1 r_1 - w_2 r_2 \right),
\]
\[
= \frac{1}{\sum_{j=1}^{n} r_j \left( 1 + \sum_{k=1}^{m_j} \prod_{i=1}^{k} \frac{u_{j,(i-1)}}{w_{j,(i)}} \right)} \left( 1 - \prod_{i=1}^{n} \frac{u_i}{w_i} \right).
\] (2.2.20)

Therefore only the normalisation constant is changed by the existence of branches.

### 2.3 Extensions and Original Work

#### 2.3.1 Parallel Pathway Models

The results for single reaction pathways have been extended by Kolomeisky [26] to two parallel paths. This section extends them further to \( k \) parallel paths rather
than just two. Consider $k$ separate reaction pathways coupled at state 1. The master equations for the non-coupled states are given by

$$\dot{P}_j^{(k)} = u_{j-1}^{(k)} P_{j-1}^{(k)} + w_{j+1}^{(k)} P_{j+1}^{(k)} - (u_j^{(k)} + w_j^{(k)}) P_j^{(k)},$$

(2.3.1)

with $j \in \{2, ..., n^{(k)}\}$, $u_j^{(k)}$ being the rate from state $j$ to state $j+1$ on the pathway $k$ and $w_j^{(k)}$ being the rate from state $j$ to state $j-1$ on the pathway $k$ and $P_1^{(k)} = P_1$. The coupling state master equation can be written

$$\dot{P}_1 = \sum_k \left[ u_{n^{(k)}}^{(k)} P_{n^{(k)}}^{(k)} + w_2^{(k)} P_2^{(k)} - (u_1^{(k)} + w_1^{(k)}) P_1^{(k)} \right],$$

(2.3.2)

with the aim that each element of the sum should be zero for the steady state as the flux into each branch should equal the flux out. This system is shown in Figure 2.6.

![Figure 2.6: A periodic parallel pathway reaction network.](image)

Following Kolomeisky [26],

$$r_j^{(k)} = \frac{1}{u_j^{(k)}} \left( 1 + \sum_{l=1}^{n^{(k)}-1} \prod_{i=j+1}^{j+l} \frac{w_i^{(k)}}{u_i^{(k)}} \right),$$

(2.3.3)

it follows from Equation 2.2.3 that

$$u_j^{(k)} r_j^{(k)} + w_{j+1}^{(k)} r_j^{(k)} - u_j^{(k)} r_j^{(k)} - w_j^{(k)} r_j^{(k)} = 0.$$

(2.3.4)

Therefore, choose the ansatz

$$P_j^{(k)} = Ne_j r_j^{(k)}$$

(2.3.5)

for $j \in \{2, ..., n^{(k)}\}$ and require that $P_1^{(k)} = P_1 \forall k$ to get the steady state solution. The $e_k$ are therefore chosen so that

$$\frac{P_1}{N} = \frac{P_1}{N r_1^{(k)}} \forall k$$

(2.3.6)

giving

$$e_k = \frac{P_1}{N r_1^{(k)}} \forall k$$

(2.3.7)
to ensure the coupling conditions are met. Without loss of generality it can be assumed that $e_1 = 1$ and so

$$P_1 = Nr_1^{(1)}$$  \hspace{1cm} (2.3.8)

giving

$$e_k = \frac{r_1^{(1)}}{r_1^{(k)}} \forall k.$$  \hspace{1cm} (2.3.9)

$N$ is chosen so that

$$N = \frac{1}{r_1^{(1)} \left( 1 + \sum_k \sum_{j=2}^{n(k)} \frac{r_j^{(k)}}{r_1^{(k)}} \right)}$$  \hspace{1cm} (2.3.10)

to ensure the probabilities are normalised so that they sum to unity.

It follows directly from Equation 2.3.4 that Equation 2.3.1 gives

$$\dot{P}_j^{(k)} = 0$$  \hspace{1cm} (2.3.11)

for $j \in \{2, ..., n(k)\}$. With ansatz 2.3.5 Equation 2.3.2 becomes

$$\dot{P}_1 = \sum_k \left[ u_{n(k)}^{(k)} P_{n(k)}^{(k)} + w_2^{(k)} P_2^{(k)} - (u_1^{(k)} + w_1^{(k)}) P_1 \right],$$

$$= N \sum_k e_k \left[ u_{n(k)}^{(k)} r_{n(k)}^{(k)} + w_2^{(k)} r_2^{(k)} - u_1^{(k)} r_1^{(k)} - w_1^{(k)} r_1^{(k)} \right],$$

$$= 0.$$  \hspace{1cm} (2.3.12)

The ansatz 2.3.5 thus satisfies the condition required for the steady state of 2.3.1.

Therefore, the steady state solution is

$$P_j^{(k)} = Ne_k r_j^{(k)},$$  \hspace{1cm} (2.3.13)

$$N = \frac{1}{P_1 + \sum_k \sum_{j=2}^{n(k)} P_j^{(k)}},$$  \hspace{1cm} (2.3.14)

$$e_k = \frac{r_1^{(1)}}{r_1^{(k)}}.$$  \hspace{1cm} (2.3.15)

In the steady state the flux through state 1 is the sum of the fluxes through each pathway. Therefore

$$J = \frac{V}{d} = \sum_k J^{(k)} = \sum_k \frac{V^{(k)}}{d^{(k)}},$$

$$= \sum_k \left( u_1^{(k)} P_1 - w_2^{(k)} P_2^{(k)} \right),$$

$$= N \sum_k e_k \left( u_1^{(k)} r_1^{(k)} - w_2^{(k)} r_2^{(k)} \right),$$

$$= N \sum_k e_k \left( 1 - \prod_{i=1}^{n(k)} \frac{u_i^{(k)}}{w_i^{(k)}} \right).$$  \hspace{1cm} (2.3.16)
2.3. EXTENSIONS AND ORIGINAL WORK

2.3.2 Divided Pathway Models

Das and Kolomeisky [12] claimed to extend the single pathway model to include a section where the path divides into two pathways. This was achieved by mapping the divided pathway directly onto the parallel pathway model, reducing the number of coupling states to one. The analysis presented here demonstrates that there must be at least two coupled states for a divided pathway model. This section generalises the results to \( k \) divided pathways and presents an alternative analysis to that provided by Das and Kolomeisky [12].

The master equations for the single pathway states are given by

\[
\dot{P}_j = u_{j-1}P_{j-1} + w_{j+1}P_{j+1} - (u_j + w_j)P_j, \tag{2.3.17}
\]

for \( j \in \{2, ..., m-1\} \), the master equations for the states where the pathways are divided are given by

\[
\dot{P}^{(k)}_j = u_{j-1}^{(k)}P^{(k)}_{j-1} + w_{j+1}^{(k)}P^{(k)}_{j+1} - (u_j^{(k)} + w_j^{(k)})P^{(k)}_j, \tag{2.3.18}
\]

with \( j \in \{m+1, ..., n^{(k)}\} \), and the master equations for the coupling states are given by

\[
\dot{P}_1 = w_2P_2 - u_1P_1 + \sum_k \left( u_{n^{(k)}}^{(k)}P^{(k)}_{n^{(k)}} - w_1^{(k)}P_1 \right), \tag{2.3.19}
\]

\[
\dot{P}_m = u_{m-1}P_{m-1} - w_mP_m + \sum_k \left( w_{m+1}^{(k)}P^{(k)}_{m+1} - u_m^{(k)}P_m \right). \tag{2.3.20}
\]

This system is shown in Figure 2.7.

![Figure 2.7: A periodic divided pathway reaction network.](image)

This system is projected onto the parallel pathway model coupled at two states - rather than the one coupled state used by Das and Kolomeisky [12] - shown in Figure 2.8. The single pathway between states 1 and \( m \) is split so that the resulting system is a parallel pathway coupled at states 1 and \( m \). If the system is projected so that it is not coupled at state \( m \) (or equivalently at state 1 as in the analysis by Das and Kolomeisky [12]) then \( P_m = \sum_k P^{(k)}_m \) which introduces cross path coupling.
Figure 2.8: The periodic divided pathway reaction network in Figure 2.7 mapped onto a parallel pathway model coupled at two states.

terms in the \(-u_w^{(k)} P_m\) term in Equation 2.3.20, such as \(u_{m}^{(1)} P_m^{(2)}\), that make little physical sense.

Setting \(w_j^{(k)} = w_j\) and \(u_j^{(k)} = u_j\) with \(\sum_k P_j^{(k)} = P_j\) for \(j \in \{2, ..., m-1\}\), \(P_m^{(k)} = P_m\) and \(P_1^{(k)} = P_1\ \forall k\), \(\sum_k u_1^{(k)} = u_1\) and \(\sum_k w_m^{(k)} = w_m\), Equation 2.3.19 that describes state 1 becomes

\[
\dot{P}_1 = \sum_k \left( u_n^{(k)} P_n^{(k)} + w_2^{(k)} P_2^{(k)} - \left[ u_1^{(k)} + w_1^{(k)} \right] P_1^{(k)} \right),
\]

Equation 2.3.20 that describes state \(m\) becomes

\[
\dot{P}_m = \sum_k \left( u_{m-1}^{(k)} P_{m-1}^{(k)} + w_{m+1}^{(k)} P_{m+1}^{(k)} - \left[ u_m^{(k)} + w_m^{(k)} \right] P_m^{(k)} \right)
\]

and Equation 2.3.17 that describes states \(j \in \{2, ..., m-1\}\) becomes

\[
\sum_k \dot{P}_j^{(k)} = \sum_k \left( u_{j-1}^{(k)} P_{j-1}^{(k)} + w_{j+1}^{(k)} P_{j+1}^{(k)} - \left[ u_j^{(k)} + w_j^{(k)} \right] P_j^{(k)} \right).
\]

Therefore, solutions to the parallel chain model

\[
\dot{P}_j^{(k)} = u_{j-1}^{(k)} P_{j-1}^{(k)} + w_{j+1}^{(k)} P_{j+1}^{(k)} - \left[ u_j^{(k)} + w_j^{(k)} \right] P_j^{(k)},
\]

with \(j \in \{2, ..., n^{(k)}\}\) and

\[
\dot{P}_1 = \sum_k u_n^{(k)} P_n^{(k)} + w_2^{(k)} P_2^{(k)} - \left[ u_1^{(k)} + w_1^{(k)} \right] P_1
\]

are also solutions to the divided pathway system if \(w_j^{(k)} = w_j\) and \(u_j^{(k)} = u_j\) with \(\sum_k P_j^{(k)} = P_j\) for \(j \in \{2, ..., m-1\}\), \(P_m^{(k)} = P_m\) and \(P_1^{(k)} = P_1\ \forall k\), \(\sum_k u_1^{(k)} = u_1\) and \(\sum_k w_m^{(k)} = w_m\).

Therefore, the steady state solution is

\[
P_j^{(k)} = N e_k r_j^{(k)},
\]

\[
N = \frac{1}{r_1^{(1)} + r_1^{(m)} + \sum_k \left( \sum_{j=2}^{m-1} + \sum_{j=m+1}^{n^{(k)}} \right) e_k r_j^{(k)}},
\]

\[
e_k = \frac{r_1^{(1)}}{r_1^{(k)}} = \frac{r_1^{(m)}}{r_1^{(k)}},
\]
There remain $2 \times \max(k)$ unfixed reaction rates: $u_1^{(k)}$ and $w_m^{(k)}$. These must satisfy
max($k$) + 1 constraints:

$$\frac{r_1^{(1)}}{r_1^{(k)}} = \frac{r_m^{(1)}}{r_m^{(k)}}, \quad \sum_k u_1^{(k)} = u_1, \quad \sum_k w_m^{(k)} = w_m. \quad (2.3.29)$$

Therefore the system is potentially soluble for all $k \in \mathbb{N}$.

Defining

$$k \Xi_a^b := \sum_{j=a}^{b} \prod_{i=a}^{j} \frac{w_i^{(k)}}{u_i^{(k)}}, \quad (2.3.31)$$

$$k \prod_{a}^{b} := \prod_{i=a}^{b} \frac{w_i^{(k)}}{u_i^{(k)}}, \quad (2.3.32)$$

with indices being taken modulo $n^{(k)}$ and states $0 \equiv n^{(k)}$, the $u_1^{(k)}$ can be found in terms of $u_1$:

$$\frac{r_1^{(1)}}{r_1^{(k)}} = \frac{r_m^{(1)}}{r_m^{(k)}}, \quad \frac{u_1^{(k)} (1 + 1 \Xi_2^{n^{(1)}})}{u_1^{(1)} (1 + k \Xi_2^{n^{(k)}})} = \frac{u_m^{(k)} (1 + 1 \Xi_{m+1}^{n^{(1)} + m - 1})}{u_m^{(1)} (1 + k \Xi_{m+1}^{n^{(k)} + m - 1})},$$

$$\frac{u_1^{(k)} (1 + 1 \Xi_2^{n^{(1)}})}{u_1^{(1)} (1 + k \Xi_2^{n^{(k)}})} = \frac{u_m^{(k)} \left[ 1 + 1 \Xi_{m+1}^{n^{(1)}} + \frac{w_1^{(1)}}{u_1^{(1)}} \prod_{i=m+1}^{n^{(1)}} (1 + 1 \Xi_2^{m-1}) \right]}{u_m^{(1)} \left[ 1 + k \Xi_{m+1}^{n^{(k)}} + \frac{w_1^{(k)}}{u_1^{(k)}} \prod_{i=m+1}^{n^{(k)}} (1 + k \Xi_2^{m-1}) \right]},$$

which gives

$$\frac{u_1^{(k)} (1 + k \Xi_{m+1}^{n^{(k)}}) + w_1^{(k)} \prod_{i=m+1}^{n^{(k)}} (1 + 1 \Xi_2^{n^{(k)-1}})}{u_1^{(1)} (1 + 1 \Xi_{m+1}^{n^{(1)}}) + w_1^{(1)} \prod_{i=m+1}^{n^{(1)}} (1 + 1 \Xi_2^{m-1})} = \frac{u_1^{(1)} (1 + k \Xi_2^{n^{(1)}})}{u_1^{(1)} (1 + 1 \Xi_2^{n^{(1)}})},$$

and so

$$u_1^{(k)} = u_1^{(1)} \psi^{(k)} + \Psi^{(k)} \quad (2.3.33)$$

with

$$\psi^{(k)} = \frac{u_m^{(k)} (1 + k \Xi_2^{n^{(k)}})}{u_m^{(1)} (1 + 1 \Xi_{2}^{n^{(1)}})} \frac{(1 + 1 \Xi_{m+1}^{n^{(1)})}}{(1 + k \Xi_{m+1}^{n^{(k)}})} \quad (2.3.34)$$

$$\Psi^{(k)} = w_1^{(k)} \prod_{i=m+1}^{n^{(k)}} \frac{1 + k \Xi_2^{n^{(k)-1}}}{1 + k \Xi_{m+1}^{n^{(k)}}} \frac{u_1^{(k)} (1 + k \Xi_2^{n^{(k)}})}{u_1^{(1)} (1 + 1 \Xi_2^{n^{(1)}})}, \quad (2.3.35)$$
It is known that
\[
\sum_k u_1^{(k)} = u_1, \tag{2.3.36}
\]
therefore
\[
u_1^{(1)} + \sum_{k \neq 1} u_1^{(k)} = u_1,\]
\[
u_1^{(1)} = \frac{u_1 - \sum_{k \neq 1} \Psi^{(k)}}{1 + \sum_{k \neq 1} \psi^{(k)}}, \tag{2.3.37}
\]
and so
\[
u_1^{(k)} = \left[\frac{u_1 - \sum_{k \neq 1} \Psi^{(k)}}{1 + \sum_{k \neq 1} \psi^{(k)}}\right] \psi^{(k)} + \Psi^{(k)}, \tag{2.3.38}
\]
distinct from the work of Das and Kolomeisky [12]. The \(w_m^{(k)}\) do not need to be fixed for relation 2.3.29 to be satisfied. However \(\sum_k w_m^{(k)} = w_m\) is required.

Steady state velocities can be derived in a similar manner to those in the parallel pathway model.

### 2.4 Molecular Detachment

The work already discussed assumes that a motor remains attached to its track however, many biological models of processive motor proteins allow detachment. If such a system is described by a discrete stochastic model then the normalisation condition of state probability occupancy does not hold; molecules are leaving the modelled system and so as time progresses, the sum of all the probabilities decreases. A trapping state (where all detachments transition to) solves this issue but requires a highly involved investigation of the transient behaviour of the system before it reaches steady state. Another method is needed.

Wu et al. [61] treated the detachment as a perturbation: it was assumed that the detachment dynamics were infinitesimal when compared to the dynamics of the rest of the system. This assumption however can break down in some regions of parameter space (e.g. close to stall force).

Kolomeisky and Fisher [27] developed a method to account for the loss of molecules by renormalising the system. The renormalised system then satisfies the normalisation condition. Firstly this is presented and then a simple two state system with unequal detachment rates is investigated to explore the validity of renormalisation.

#### 2.4.1 Renormalisation

A discrete stochastic system of \(n\) states can be described by \(n\) master equations as described at the start of this chapter. To renormalise the system into one
which conserves probability the detachment terms $\delta_i$ must be removed. Following Kolomeisky and Fisher [27]

$$P_i \approx \frac{1}{\phi_i} e^{-\lambda \tilde{P}_i}$$  \hspace{1cm} (2.4.1)

$$\dot{\tilde{P}} = \tilde{M}\tilde{P}$$  \hspace{1cm} (2.4.2)

where $\tilde{M}$ is a renormalised reaction-rate matrix with the renormalised detachment rate $\tilde{\delta}_i = 0$, $\forall i$. We assume that one eigenvalue is dominant and the other processes decay away quickly, therefore the solution is determined by dominant (closest to zero) $\lambda$.

Substituting ansatz 2.4.1 into the governing master equations and multiplying by $\phi_i e^{\lambda t}$ for state $i$ gives equations for $\tilde{P}_i$. Matching up the terms with the desired form of the renormalised master equations gives the renormalised rate constants

$$\tilde{W}_{k,l} = W_{k,l} \frac{\phi_l}{\phi_k}.$$  \hspace{1cm} (2.4.3)

An additional condition also arises: the equations

$$\tilde{\delta}_i + \sum_{j=1}^{n} \tilde{W}_{i,j} = \sum_{j=1}^{n} W_{i,j} + \delta_i - \lambda$$  \hspace{1cm} (2.4.4)

must be satisfied for $i \in \{1, 2, ..., n\}$. Using the renormalised rate relations given in Eqn 2.4.3 and $\tilde{\delta}_i = 0$, $\forall i$ in the renormalised case,

$$\sum_{j=1}^{n} W_{i,j} \phi_j - \sum_{j=1}^{n} W_{i,j} \phi_i - \delta_i \phi_i = -\lambda \phi_i.$$  \hspace{1cm} (2.4.5)

So the condition for our system to be renormalised using Eqn. 2.4.1 into Eqn. 2.4.2 which is probability conserving, can be expressed as the eigenvalue equation

$$\tilde{M}^T \phi = -\lambda \phi.$$  \hspace{1cm} (2.4.6)

where $-\lambda$ is the dominant (closest to zero) eigenvalue of $\tilde{M}$. Note the assumption that the detachment is linked to the slowest eigenvalue and thus is slower than the other processes in the system. The system can now be described in terms of the renormalised reaction-rate matrix that is given by the renormalised transition rates (equation 2.4.3).

The average duration of a run is $1/\lambda$ and thus the distance a motor protein travels before detaching - the run length $L$ - is:

$$L = \frac{V}{\lambda},$$  \hspace{1cm} (2.4.7)

where $V$ is the average velocity of motor proteins in the statistical limit (renormalised if relevant).
2.4.2 Two State, Two Substep System with Unequal Detachment

A toy two state system with two substeps and two different molecular detachment pathways is shown in Figure 2.2 and will be used in this section to investigate renormalisation. Experimentally observable quantities, such as the dispersion and velocity of this system, are calculated from transition rates between states and the probability $\hat{P}_i$ of a molecule being on the track and in biomechanochemical state $i$.

The master equations are given by

$$\mathbf{P}(t) = \mathbf{M}\mathbf{P}(t), \quad (2.4.8)$$

where $\mathbf{M}$ is the transition rate matrix,

$$
\begin{pmatrix}
-\left(u_1 + w_1 + \delta_1\right) & u_2 + w_2 \\
\left(u_1 + w_1\right) & -\left(u_2 + w_2 + \delta_2\right)
\end{pmatrix}
\quad (2.4.9)
$$

and the probabilities satisfy the normalisation condition

$$P_1(t) + P_2(t) + P_{\text{off}}(t) = 1, \quad (2.4.10)$$

with $P_{\text{off}}(t)$ being the probability of a molecule having already detached from the system.

Renormalised reaction rates of this system are given by

$$
\begin{align*}
\tilde{u}_1 &= \frac{u_1}{\phi_1}, \\
\tilde{w}_1 &= \frac{w_1}{\phi_1}, \\
\tilde{u}_2 &= \frac{u_2}{\phi_2}, \\
\tilde{w}_2 &= \frac{w_2}{\phi_2}
\end{align*}
$$
with the $\phi_i$ being elements of the dominant eigenvector of the transposed reaction rate matrix. The renormalised probabilities are

$$\tilde{P}_1 = \frac{\bar{u}_1 + \bar{w}_1}{\bar{u}_1 + \bar{w}_1 + \bar{u}_2 + \bar{w}_2},$$

(2.4.11)

$$\tilde{P}_2 = \frac{\bar{u}_2 + \bar{w}_2}{\bar{u}_1 + \bar{w}_1 + \bar{u}_2 + \bar{w}_2},$$

(2.4.12)

or

$$\tilde{P}_1 = \frac{b}{\sqrt{\phi_1^2 a + b}},$$

(2.4.13)

$$\tilde{P}_2 = \frac{a}{a + b \phi_2^2},$$

(2.4.14)

where $a = u_1 + w_1$ and $b = u_2 + w_2$.

The probability that a molecule is in state $i$ given that it remains on the track is

$$\hat{P}_i = \frac{P_i}{\sum_{j=1}^2 P_j},$$

(2.4.15)

with $\sum_{i=1}^2 \hat{P}_i = 1$.

The rate of change of the state occupancy probabilities for molecules on the track is

$$\dot{\hat{P}}_1 = \frac{\hat{P}_1}{\hat{P}_1 + \hat{P}_2} - \hat{P}_1 \frac{\hat{P}_1 + \hat{P}_2}{(\hat{P}_1 + \hat{P}_2)^2},$$

(2.4.16)

$$\dot{\hat{P}}_2 = a \hat{P}_1 - \delta \hat{P}_1 \hat{P}_2,$$

(2.4.17)

where $\delta = \delta_1 - \delta_2$.

$\hat{P}_1$ and $\hat{P}_2$ in the steady state can be determined from equations (2.4.16) and (2.4.17) and probability normalisation:

$$\hat{P}_1 = \frac{a + b + \delta - \sqrt{(a + b + \delta)^2 - 4bd}}{2\delta},$$

(2.4.18)

$$\hat{P}_2 = \frac{-a - b + \delta + \sqrt{(a + b + \delta)^2 - 4bd}}{2\delta},$$

(2.4.19)

Using the decay ansatz in equation 2.4.1 and relation 2.4.15:

$$\hat{P}_i = \frac{\tilde{P}_i}{\phi_i \sum_j \tilde{P}_j},$$

(2.4.20)
in general. Thus $\hat{P}_i = \hat{P}_i$ when

$$\phi_i \sum_j \frac{\hat{P}_j}{\phi_j} = 1$$

(2.4.21)

for all $i$, and hence $\phi_j = \phi_i$ for all $i, j$. Clearly this will not always be true. To understand the consequences of this, we calculate the dominant eigenvector of the system shown in Figure 2.2

$$\phi_1 = 1 - \frac{1}{2b} \left[ A - \sqrt{\Delta} \right],$$

(2.4.22)

$$\phi_2 = 1,$$

(2.4.23)

with $A = a + b + \delta$ and $\Delta = (a + b + \delta)^2 - 4\delta b$. The condition that $\phi_1 = \phi_2$ therefore becomes

$$\delta_1 = \delta_2.$$  

(2.4.24)

This is not always true, but $\phi_1 \approx \phi_2$ may hold approximately for $|\delta|$ small. If $|\delta|$ is small, then expanding the root in equation (2.4.18) gives

$$\sqrt{\Delta} = a + b + \frac{a - b}{a + b}\delta + \frac{2ab}{(a + b)^3}\delta^2 - \frac{2ab(a - b)}{(a + b)^5}\delta^2 + O(\delta^4),$$

(2.4.25)

and so

$$\hat{P}_1 = \frac{b}{a + b} - \frac{ab}{(a + b)^3}\delta + \frac{ab(a - b)}{(a + b)^5}\delta^2 + O(\delta^3),$$

(2.4.26)

which $P_1|_{\delta_1=\delta_2=0}$ are the probabilities for the system without detachment.

In this system $\phi_2 = 1$ and $\phi_1$ can be expanded up to order $\delta$ to give

$$\phi_1 = 1 - \frac{1}{a + b}\delta + \frac{a}{(a + b)^3}\delta^2 + O(\delta^3),$$

(2.4.27)

$$\phi_1^2 = 1 - \frac{2}{a + b}\delta + \frac{3a + b}{(a + b)^3}\delta^2 + O(\delta^3),$$

(2.4.28)

and

$$(a + b\phi_1^2)^{-1} = \frac{1}{a + b} + \frac{4b}{(a + b)^3}\delta - \frac{4b(2a - 3b)}{(a + b)^5}\delta^2 + O(\delta^3),$$

(2.4.29)

therefore

$$\phi_1^2(a + b\phi_1^2)^{-1} = \frac{1}{a + b} - \frac{4a}{(a + b)^3}\delta + \frac{4a(2a - 3b)}{(a + b)^5}\delta^2 + O(\delta^3).$$

(2.4.30)
2.4. MOLECULAR DETACHMENT

Substituting this in the expressions for $\tilde{P}_i$ gives

$$
\tilde{P}_1 = P_1|_{\delta_1=\delta_2=0} - \frac{4ab}{(a+b)^3}\delta + \frac{4ab(2a-3b)}{(a+b)^5}\delta^2 + O(\delta^3).
$$

(2.4.31)

Therefore,

$$
\tilde{P}_i \neq \bar{P}_i
$$

(2.4.32)

at order $\delta$ or above. Therefore, if the detachment rates from each state are not the same then the renormalised probabilities are not equivalent to the on track probabilities. This contradicts Kolomeisky and Fisher’s claim that the renormalised probabilities are the same as the on-track probabilities[27].

It is now important to establish whether the on track velocity is the same as the renormalised velocity. The average velocity of molecules in a renormalised system are governed by the physical space equations

$$
\frac{d\tilde{p}_{2,s}}{dt} = \tilde{u}_1\tilde{p}_{1,s} + \tilde{w}_1\tilde{p}_{1,s+1} - (\tilde{u}_2 + \tilde{w}_2)\tilde{p}_{2,s},
$$

(2.4.33)

$$
\frac{d\tilde{p}_{1,s}}{dt} = \tilde{u}_2\tilde{p}_{2,s} + \tilde{w}_2\tilde{p}_{2,s} - (\tilde{u}_1 + \tilde{w}_1)\tilde{p}_{1,s}.
$$

(2.4.34)

Here each $s$ is associated with a physical location each being either an even or an odd position.

The average displacement of a molecule on the track in the renormalised system is given by

$$
<\tilde{x}> = \sum_s \tilde{p}_{2,s}[(d_1 + d_2)s - d_2] + \sum_s \tilde{p}_{1,s}(d_1 + d_2)s,
$$

$$
= (d_1 + d_2) \left[ \sum_s s\tilde{p}_{2,s} + \sum_s s\tilde{p}_{1,s} \right] - d_2\tilde{P}_2,
$$

$$
= (d_1 + d_2) \left[ \tilde{X} + \tilde{Y} \right] - d_2\tilde{P}_2.
$$

(2.4.35)

$\tilde{Y} \equiv \sum_s s\tilde{p}_{1,s}$ and $\tilde{X} \equiv \sum_s s\tilde{p}_{2,s}$ satisfy

$$
\frac{d\tilde{X}}{dt} = \tilde{a}\tilde{Y} + \tilde{u}_1\tilde{P}_1 - \tilde{b}\tilde{X},
$$

(2.4.36)

$$
\frac{d\tilde{Y}}{dt} = \tilde{b}\tilde{X} - \tilde{w}_2\tilde{P}_2 - \tilde{a}\tilde{Y}.
$$

(2.4.37)

In the steady state, assuming a constant velocity solution and initial condition $<\tilde{x}(0)> = 0$ the set of solutions is

$$
\tilde{X} = \frac{\tilde{a} \left( \tilde{u}_1\tilde{P}_1 - \tilde{w}_2\tilde{P}_2 \right)}{\tilde{a} + \tilde{b}}t + \frac{\tilde{a}d_2\tilde{P}_2}{(d_1 + d_2)(\tilde{a} + \tilde{b})} + \frac{\tilde{b}\tilde{u}_1\tilde{P}_1 + \tilde{a}\tilde{w}_2\tilde{P}_2}{(\tilde{a} + \tilde{b})^2},
$$

(2.4.38)

$$
\tilde{Y} = \frac{\tilde{b} \left( \tilde{u}_1\tilde{P}_1 - \tilde{w}_2\tilde{P}_2 \right)}{\tilde{a} + \tilde{b}}t + \frac{\tilde{b}d_2\tilde{P}_2}{(d_1 + d_2)(\tilde{a} + \tilde{b})} - \frac{\tilde{b}\tilde{u}_1\tilde{P}_1 + \tilde{a}\tilde{w}_2\tilde{P}_2}{(\tilde{a} + \tilde{b})^2}.
$$

(2.4.39)
The velocity of molecule in such a system in the steady state is
\[
\tilde{V} = \frac{d}{dt} <\tilde{x}> = (d_1 + d_2) \left[ \frac{d\tilde{X}}{dt} + \frac{d\tilde{Y}}{dt} \right],
\]
\[
= (d_1 + d_2) \left[ \tilde{u}_1 \tilde{P}_1 - \tilde{w}_2 \tilde{P}_2 \right].
\]
(2.4.40)

This is just the flux multiplied by the total step size. Expanding up to \(O(\delta)\) the renormalised velocity is
\[
\tilde{V} = d(\tilde{u}_1 \tilde{P}_1 - \tilde{w}_2 \tilde{P}_2),
\]
\[
= v|_{\delta_1=\delta_2=0} + d \frac{(u_1 u_2 - w_1 w_2)(b - a)}{(a + b)^3} \delta + O(\delta^2),
\]
(2.4.41)

where \(d = d_1 + d_2\).

Next the velocity of motors, given that they remain on the track, is calculated. The probability that a molecule is in position \(i\) given that it remains on the track is
\[
\hat{p}_i = \frac{p_i}{\sum_j p_j},
\]
(2.4.42)

and so average velocity of on-track molecules is governed by the physical space equations
\[
\frac{d\hat{p}_{2,s}}{dt} = u_1 \hat{p}_{1,s} + w_1 \hat{p}_{1,s+1} - (u_2 + w_2) \hat{p}_{2,s} + (\delta_1 - \delta_2) \hat{p}_{2,s} \hat{P}_1,
\]
(2.4.43)
\[
\frac{d\hat{p}_{1,s}}{dt} = u_2 \hat{p}_{2,s} + w_2 \hat{p}_{2,s+1} - (u_1 + w_1) \hat{p}_{1,s} - (\delta_1 - \delta_2) \hat{p}_{1,s} \hat{P}_2.
\]
(2.4.44)

Therefore, the average displacement of a molecule given that it is on the track is given by
\[
<\tilde{x}> = \sum_s \hat{p}_{2,s} [(d_1 + d_2)s - d_2] + \sum_s \hat{p}_{1,s}(d_1 + d_2)s,
\]
\[
= (d_1 + d_2) \left[ \sum_s s\hat{p}_{2,s} + \sum_s s\hat{p}_{1,s} \right] - d_2 \hat{P}_2,
\]
\[
= (d_1 + d_2) \left[ \tilde{X} + \tilde{Y} \right] - d_2 \hat{P}_2,
\]
(2.4.45)

where hatted quantities pertain to molecules still on the track.

\(\hat{Y} \equiv \sum_s s\hat{p}_{1,s}\) and \(\hat{X} \equiv \sum_s s\hat{p}_{2,s}\) satisfy
\[
\frac{d\hat{X}}{dt} = a\hat{Y} + u_1 \hat{P}_1 - b\hat{X} + \delta \hat{P}_1 \hat{X},
\]
(2.4.46)
\[
\frac{d\hat{Y}}{dt} = b\hat{X} - w_2 \hat{P}_2 - a\hat{Y} - \delta \hat{P}_2 \hat{Y}.
\]
(2.4.47)
Assuming
\[
\dot{X} = g_1 t + h_1, \quad (2.4.48)
\]
\[
\dot{Y} = g_2 t + h_2, \quad (2.4.49)
\]
and a constant velocity solution \( \frac{d\dot{X}}{dt} + \frac{d\dot{Y}}{dt} = \dot{V} \), we get \( g_1 + g_2 = \dot{V} \). Equations 2.4.46 and 2.4.47 then give
\[
\dot{V} = u_1 \dot{P}_1 - w_2 \dot{P}_2 + \delta (\dot{P}_1 h_1 - \dot{P}_2 h_2) + \delta t (\dot{P}_1 g_1 - \dot{P}_2 g_2). \quad (2.4.50)
\]
Therefore
\[
\dot{P}_1 g_1 - \dot{P}_2 g_2 = 0, \quad (2.4.51)
\]
and setting \( g_2 = (\dot{P}_1 / \dot{P}_2) g_1 \) gives the linear-with-time components of equations 2.4.46 and 2.4.47. Initial conditions \( <\dot{x}(0)> = 0 \) give \( h_2 = d^{-1} d_2 \dot{P}_2 - h_1 \). Thus constant terms provide
\[
h_1 = \frac{d^{-1} d_2 \dot{P}_2 \left( a + \delta \dot{P}_2^2 \right) + u_1 \dot{P}_1^2 + w_2 \dot{P}_2^2}{a + b + \delta (\dot{P}_2^2 - \dot{P}_1^2)}, \quad (2.4.52)
\]
\[
h_2 = \frac{d^{-1} d_2 \dot{P}_2 \left( a + \delta \dot{P}_2^2 \right) + u_1 \dot{P}_1^2 + w_2 \dot{P}_2^2}{a + b + \delta (\dot{P}_2 - \dot{P}_1)}, \quad (2.4.53)
\]
Therefore
\[
g_1 = a d^{-1} d_2 \dot{P}_2 + u_1 \dot{P}_1 - (a + b - \delta \dot{P}_1) h_1, \quad (2.4.54)
\]
\[
g_2 = (\dot{P}_1 / \dot{P}_2) g_1. \quad (2.4.56)
\]
So the velocity for molecules on the track is
\[
\dot{V} = \frac{d <\dot{x}>}{dt} = (d_1 + d_2) \dot{V} + d_1 \dot{P}_2, \quad (2.4.55)
\]
\[
= (d_1 + d_2) \left[ u_1 \dot{P}_1 - w_2 \dot{P}_2 + \delta h_1 - \delta d^{-1} d_2 \dot{P}_2 \right],
\]
\[
= (d_1 + d_2) \left[ u_1 \dot{P}_1 - w_2 \dot{P}_2 + \delta h_1 \right] - d_2 \delta \dot{P}_2. \quad (2.4.57)
\]
Thus the velocity of motors, given that they remain on the track is
\[
\dot{V} = v \big|_{\delta_1 = \delta_2 = 0} + \frac{d\delta}{(a + b)^2} \left[ u_1 u_2 - w_1 w_2 \right] (b - a) + O(\delta^2), \quad (2.4.58)
\]
and so it is equal to the renormalised velocity up to order $\delta$:

$$\hat{V} = \hat{V}. \tag{2.4.59}$$

In this section I have investigated the differences between renormalised and on-track quantities. If the detachment rates from each state are slightly different, it has been demonstrated that the velocities are the same to leading order (equation 2.4.59). However, the probabilities (equation 2.4.32) are not. This implies the renormalisation method presented by Kolomeisky and Fisher [27] warrants further analysis. It is not pursued any further in this thesis, but would be an interesting avenue for further investigation.

Assuming renormalisation works in general for calculating the velocities and the dispersions as assumed in the literature, a system is entirely characterised by the renormalised reaction rate matrix and the velocities, dispersions and run lengths follow from the results for the same system without molecular detachment.

## 2.5 Summary

A method to calculate the velocity and dispersion relation of a molecular motor from its steady-state state occupancy probabilities has been shown for a simple toy model (section 2.1). Existing methods for calculating the probabilities were presented for different generalised system architectures in section 2.2. It is important to note the use of symmetry so that each system can be treated in an analogous manner to the single chain. Existing derivations [12, 26] have also been extended and corrected in sections 2.3.1 and 2.3.2 respectively. Fluxes and average velocities have been calculated from the probabilities for all of these systems. The renormalisation method and the calculation for the run length have also been demonstrated although it has been shown to depart from some of the claims of its creators [27] if detachment rates from distinct states are different.

A major issue with these methods is that the probabilities, and therefore the dispersion, must be recalculated for each system class. This is quite a large task, especially for those not mathematically inclined. The essence of the methods discussed here involves identifying the symmetries of a given architecture. In the next chapter I present a novel generalisation of this idea that gives the probabilities, velocities and dispersions for general molecular motor reaction pathways.
Chapter 3

Calculating Dynamic Properties
Generally

“A pessimist sees the difficulty in every opportunity; an optimist sees the opportunity in every difficulty”

Winston Churchill

Molecular motors play important roles within a biological cell, performing functions such as intracellular transport and gene transcription. Recent experimental work suggests that there are many plausible biochemical mechanisms that molecules such as myosin-V could use to achieve motion. To account for the abundance of possible discrete-stochastic frameworks that can arise when modelling molecular motor walks, a generalised and straightforward graphical method for calculating their dynamic properties is presented. It allows the calculation of the velocity, dispersion and randomness ratio for any proposed system through analysis of its structure. This chapter extends King and Altman’s [24] work on networks of enzymatic reactions by calculating additional dynamic properties for spatially hopping systems. Results for $n$-state systems are presented: single chain, parallel pathway, divided pathway, divided pathway with a chain. A novel technique for combining multiple system architectures coupled at a reference state is also demonstrated. 4-state examples illustrate the effectiveness and simplicity of these methods.

The work in this chapter has been published in the Journal of Chemical Physics [3].

3.1 Introduction

A discrete stochastic model of a molecular motor walk assumes that we can split the mechanochemical pathways into discrete states and that transitions between these states occur probabilistically. It is assumed to approach its steady state
(equilibrium in time) rapidly if molecular detachments are neglected. The steady-state solution of the governing master equations therefore can be used to determine the relevant mean behaviour of the molecules, for example their average velocity or the dispersion, from the transition rates. This is important in the investigation of many molecular motor models, for example those for myosin-V [1, 41, 44].

The flux balance method [50] allows calculation of quantities such as the velocity or the dwell times, without the need for explicit solutions for the state probabilities. However, it cannot give quantities such as the dispersion or randomness ratio, the reciprocal of which is the number of rate-limiting steps [48]. A method presented by Chemla et al. [6] allows the calculation of velocities, dispersions for any given biochemical pathway but cannot give general formulae. The calculations, particularly for large systems with reversible transitions, can require computationally expensive calculations and are mathematically quite involved.

An approach based on Derrida [13] has proved useful in calculating exact steady states and dynamic properties for specific classes of system architectures of arbitrary size. The simpler examples of these include single chains [13], parallel chains [26] and divided pathways [12]. Periodic parallel lattices have also been studied [47] in the limit of strong coupling between each branch. Each class can be modified to include branches and molecular detachment [27]. The average velocity and its dispersion is calculated individually for each system architecture. The method presented here simplifies, consolidates and extends all this work by presenting a general graphical method for any system.

A method for finding the steady-state probabilities of enzymatic networks graphically was first presented by King and Altman [24] and developed by Hill [18]. I have tailored it here to the context of molecular motors and extended the method to give additional dynamic quantities such as the dispersion presented in section 3.4, that was previously difficult to calculate.

A novel and mathematically straightforward method for calculating dynamic quantities for any biochemical pathway with any distribution of stepping sizes is presented. Explicit and exact expressions for the steady-state probabilities, average velocity and dispersion relation are given. These all depend on a set of variables $C_{ij}$ that can be determined in an intuitive graphical manner from the system architecture. The ability to calculate results for any generalised structure in such a straightforward manner distinguishes this method from all others; my methods enable the derivation of general formulae for specific system structures reducing potentially expensive calculations. The structure of the system is preserved in the calculations allowing existing general results to be analysed and modified; I demonstrate a method of combining several general architectures together by coupling them at a reference state. For smaller systems the dispersion relation that is usually complicated can be written down simply thus reducing the level of mathematical complexity. The methods and expressions presented here are therefore powerful tools in investigating the steady states and dynamic properties of theoretical models for molecular motor stepping cycles.

The calculation of steady-state probabilities and dynamic properties using this
3.2 General Steady-State Probabilities

We consider a system of \( n \) states, each representing a biomechanochemical state of a molecule, described by \( n \) master equations whose form is determined by the proposed set of biochemical pathways. In matrix form we have

\[
\dot{\mathbf{P}}(t) = \mathbf{M}\mathbf{P}(t)
\]  

(3.2.1)

where \( \mathbf{M} \) is a \( n \times n \) transition rate matrix and the \( i \)th component of the vector of state occupancy probability \( \mathbf{P}(t) \) is \( P_i(t) \). This can be written

\[
\frac{dP_i}{dt} = \sum_j [W_{ji}P_j - W_{ij}P_i] - \delta_i P_i
\]  

(3.2.2)

where the transition rate from state \( i \) to state \( j \) is denoted by \( W_{ij} \) and the detachment rate (the rate at which molecules leave the pathway) from state \( i \) is given by \( \delta_i \). We have \( W_{ij} = 0 \) if there is no possible transition between state \( i \) and state \( j \). A system with non-zero detachment rates can be renormalised into a system without detachments using the procedure (section 2.4.1) outlined by Kolomeisky and Fisher [27]. Therefore only systems without molecular detachment (\( \delta_i = 0 \)) are considered in this chapter.

Thus in the steady state

\[
\sum_j [W_{ji}P_j - P_iW_{ij}] = 0 \quad \forall j.
\]  

(3.2.3)

A molecular motor can be assumed to pass through a repeating sequence of changes to achieve motion. There is therefore a periodicity to the system with the transition from one period to another carrying some notion of direction: forwards moves to the next period and backwards moves to the previous. For example, the master equations for states along a single chain (nearest-neighbour coupled states) are of the form

\[
\frac{dP_i}{dt} = u_{i-1}P_{i-1} + w_{i+1}P_{i+1} - [u_i + w_i]P_i.
\]  

(3.2.4)
with forwards and backwards transition rates denoted by \( u_i \) and \( w_i \) respectively. State \( n \equiv 0 \) and indices are taken modulo \( n \). This is exactly the system studied by Derrida [13].

We now introduce some useful definitions.

A **branch** is a sequence of states with only nearest-neighbour transitions between them.

A **coupling state** is a state that connects two or more branches.

A **rate path** from \( a \) to \( b \) is a product of rates that are directed along a connected path from \( a \) to \( b \). For the system described in equation (3.2.4), \( u_a u_{a+1} u_{a+2} \) is a rate path from \( a \) to \( a + 3 \). A rate path from \( a \) to \( b \) is **closed** if it also contains a rate path from \( b \) to \( a \). A rate path can be directed: if we define a rate path from \( a \) to \( b \) as being a *forwards* rate path then the rate path from \( b \) to \( a \), that passes through each of the same states but in the reverse order, is known as a *backwards* rate path.

A **rate tree** of \( b \) is a product of reaction rates that are directed from unique states and contains a rate path term that is directed from each state in the rate tree to \( b \).

A **configuration** of \( b \) is a non-unique rate tree of \( b \) containing one rate directed from every state in the system except \( b \) and a **configuration** of \( b \) is a non-unique rate tree of \( b \) containing one rate directed from every state including \( b \). Thus a configuration cannot contain any closed rate paths, however a configuration of \( b \) must contain exactly one closed rate path.

In the single chain \( n = 4 \) example, a configuration of state 1 is \( w_2 u_3 u_0 \), another configuration is \( w_2 u_4 u_0 \). The remaining two are \( w_2 w_3 u_0 \) and \( w_2 w_3 u_0 \). This is shown in Figure 3.1. Note that \( u_2 u_3 u_0 \) is not a configuration of state 1 because it does not contain a rate path from 0 or 3 to 1. In this example a configuration of state 1 is simply a configuration multiplied by either \( u_1 \) or \( w_1 \).

A **rate path reversal** is a rate path within a rate tree with forwards rates changed into backwards rates so that the rate tree retains its properties. In the previous example \( w_2 u_3 u_0 \) and \( w_2 w_3 u_0 \) are rate path reversals of \( w_2 u_3 u_0 \) but \( w_2 w_3 w_0 \) is not.

If \( Q_i \) is the sum of all possible configurations for state \( i \) and \( Q_i^\ast \) is the sum of all possible configurations* for state \( i \) then

\[
Q_i^\ast = \sum_j W_{ji} Q_j
\]  

(3.2.5)

and

\[
Q_i^\ast = Q_i \sum_j W_{ij},
\]  

(3.2.6)

with both giving a relation between a sum over all configurations* of \( i \) and a sum over all configurations of \( i \). The first relation being a sum over each \( Q_j \) multiplied by the reaction rate from state \( j \) to state \( i \). This gives a rate tree for \( i \) that contains a rate from every state: a sum over all configurations* of \( i \). The second being a
3.2. GENERAL STEADY-STATE PROBABILITIES

Figure 3.1: All the configurations of state 1 for the Derrida [13] $n = 4$ example. $A$ represents configuration $c_A = u_2u_3u_0$, $B$ represents $c_B = w_2u_3u_0$, $C$ represents $c_C = w_2w_3u_0$ and $c_D = w_2w_3w_0$. The steady-state solution for state 1 is therefore $P_1/N = Q_1 = c_A + c_B + c_C + c_D$. The configurations of state 1 are given by $u_1c_i$ or $w_1c_i$ with $i \in \{A, B, C, D\}$. 
CHAPTER 3. CALCULATING DYNAMIC PROPERTIES GENERALLY

sum over all configurations of \( i \) state multiplied by a sum over rates from \( i \) to every other state. This is also a sum over all configurations* of \( i \).

Thus

\[
\sum_j [W_{ji}Q_j - W_{ij}Q_i] = 0 \quad \forall i. \tag{3.2.7}
\]

For example, in the single chain \( n = 4 \) case taking indices modulo 4,

\[
\sum_j W_{ji}Q_j = u_{i-1}Q_{i-1} + w_{i+1}Q_{i+1},
\]

\[
= u_{i-1}(u_{i+1}u_{i+1} + w_{i+1}w_{i+1}) + u_{i-1}(w_{i+1}u_{i+1} + w_{i+1}w_i) + w_{i+1}(w_{i+1}u_{i+1} + w_{i+1}w_{i+1})
\]

\[
= Q_i(u_i + w_i),
\]

\[
= Q_i \sum_j W_{ij}.
\]

Equation (3.2.7) shows that the \( Q_i \) satisfy the steady-state equation (3.2.3). Therefore the sum of all possible configurations for state \( i \) is the non-normalised steady-state probability for state \( i \) and so

\[
P_i = NQ_i, \tag{3.2.8}
\]

where \( N \) is a normalisation constant that ensures the probabilities sum to unity. This result was first shown by King and Altman [24] and developed by Hill [18].

In the steady-state, the governing master equations investigated by Derrida [13] become

\[
0 = u_{i-1}P_{i-1} + w_{i+1}P_{i+1} - [u_i + w_i] P_i. \tag{3.2.9}
\]

Using the method above, taking all indices modulo \( n \) these have the solution

\[
\frac{P_i}{N} = Q_i = \prod_{j=i+1}^{i+1} u_j + \prod_{j=i+1}^{i-1} u_j + w_{i+1}w_{i+2} \prod_{j=i+3}^{i-1} u_j + \ldots + \prod_{j=i+1}^{i-1} w_j,
\]

\[
\frac{P_i}{Ne_1} = \frac{1}{u_i} \left( 1 + \sum_{j=i+1}^{i-1} \prod_{j=i+1}^{i-1} \frac{w_j}{u_j} \right). \tag{3.2.10}
\]

a forwards rate path from \( i + 1 \) to \( i \) plus all its path reversals, where \( e_1 = \prod_{j=i+1}^{i-1} u_j \).

This is exactly the solution shown by Derrida [13].

The probabilistic steady-state can be found using this method for any closed system. Physically, the relative sizes of these values informs us as to where in the biochemical pathways the molecules tend to dwell.
3.3 Configurational Methods

In this section relationships are derived that are important for the derivation of the dynamic quantities. The calculations presented are a little involved and thus this section can be omitted for readers by interested in the underlying mathematics.

We shall define $Z_{ik}$ and $Z_{ik}^{*}$ to be the sum over all configurations and configurations* respectively of state $i$ that include a $W_{0k}$. Thus we have $Z_{i0} = Z_{0i} = Z_{i0}^{*} = Z_{0i}^{*} = 0$ for all $i$. In the example shown in Figure 3.1, $Z_{1k} = 0$ for $k \neq 1, 3$, $Z_{13} = w_{0}w_{3}w_{2}$ and $Z_{11} = u_{0}(u_{2}u_{3} + w_{2}u_{3} + w_{2}w_{3})$.

It can be deduced that the sum over all configurations of $i \neq 0$ satisfies

$$Q_{i} = \sum_{k \neq 0} Z_{ik}, \ i \neq 0.$$  \hspace{1cm} (3.3.1)

Then $Q_{0}$ satisfies

$$Q_{0} = \sum_{j \neq 0} W_{j0} \frac{Z_{jk}}{W_{0k}}, \ \forall k \neq 0.$$  \hspace{1cm} (3.3.2)

This can be seen by considering each element of the summation in turn. Each element $j$ is the sum over all possible configurations of 0 containing the rate $W_{j0}$. Summing over all $j$ must therefore give the sum over all configurations of 0, $Q_{0}$.

Denote a rate path from $a$ to $b$ that does not pass through state 0 by $W_{a \rightarrow b}$ and from $a$ to $b$ that does pass through state 0 by $W_{a \rightarrow 0 \rightarrow b}$. The corresponding closed rate paths (constructed through multiplying by an additional rate that connects the last state to the first) are $\tilde{W}_{a \rightarrow b}$ and $\tilde{W}_{a \rightarrow 0 \rightarrow b}$ respectively.

A configuration* contains exactly one closed rate path by definition. Denote the sum over all configurations* of $j$ that include a rate $W_{0i}$ given that each term contains a $W_{j \rightarrow i}$ by $Z_{ji}(\tilde{W}_{j \rightarrow i})$ and similarly for $Z_{ji}^{*}(\tilde{W}_{j \rightarrow 0 \rightarrow i})$ and $Z_{ji}^{*}(\tilde{W}_{j \rightarrow i}$ or $\tilde{W}_{j \rightarrow 0 \rightarrow i}$).

By definition, $Z_{ii}$ is the sum over all configurations of $i$ that include a non-zero rate $W_{0i}$. Therefore for any $j \neq 0, i$, each term contains either a $W_{j \rightarrow i}$, or a $W_{j \rightarrow 0 \rightarrow i}$. $Z_{ii} \sum_{j} W_{ij}$ is therefore the sum over all configurations* of $i$ that include a non-zero rate $W_{0i}$ given that each term contains either a $\tilde{W}_{i \rightarrow i}$ or a $\tilde{W}_{i \rightarrow 0 \rightarrow i}$ (Figure 3.2) and so

$$\sum_{j} W_{ij} Z_{ii} = Z_{ii}(\tilde{W}_{i \rightarrow i} \ or \ \tilde{W}_{i \rightarrow 0 \rightarrow i}),$$

$$= Z_{ii}(\tilde{W}_{i \rightarrow i}) + Z_{ii}^{*}(\tilde{W}_{i \rightarrow 0 \rightarrow i}),$$  \hspace{1cm} (3.3.3)

as a configuration* contains exactly one closed rate path.

$Z_{ji}$ is the sum over all configurations of $j$ that include a non-zero rate $W_{0i}$. Therefore, as configurations cannot contain closed rate paths, each term must contain a $W_{i \rightarrow j}$. Thus, each term in $\sum_{j} W_{ji} Z_{ji}$ must contain a $\tilde{W}_{i \rightarrow i}$ and so $\sum_{j} W_{ji} Z_{ji}$ is the
CHAPTER 3. CALCULATING DYNAMIC PROPERTIES GENERALLY

Figure 3.2: The two possible classes of closed rate paths within the sum over all configurations* of i that include a rate \( W_{0i} \), \( Z_{ii}^* \). Each element of \( Z_{ii}^* \) either has a closed rate path from i to i through 0 (\( \bar{W}_{i\rightarrow0\rightarrow i} \), dotted) or not through zero (\( \bar{W}_{i\rightarrow i} \), dashed).

sum over all configurations* of i that include a non-zero rate \( W_{0i} \) given that each term contains a \( \bar{W}_{i\rightarrow i} \) and so

\[
Z_{ii}^*(\bar{W}_{i\rightarrow i}) = \sum_j W_{ji}Z_{ji}. \tag{3.3.4}
\]

Each term in \( \sum_j W_{j0}Z_{ji} \) must contain a \( \bar{W}_{i\rightarrow0\rightarrow i} \) and so \( \sum_j W_{j0}Z_{ji} \) is the sum over all configurations* of i that include a non-zero rate \( W_{0i} \) given that each term contains a \( \bar{W}_{i\rightarrow0\rightarrow i} \) and so

\[
Z_{ii}^*(\bar{W}_{i\rightarrow0\rightarrow i}) = \sum_j W_{j0}Z_{ji}. \tag{3.3.5}
\]

Using the above and then equation (3.3.2),

\[
\sum_j W_{ij}Z_{ii} = \sum_j (W_{j0} + W_{ji}) Z_{ji},
\]

\[
= W_{0i}Q_{0} + \sum_j W_{ji}Z_{ji}. \tag{3.3.6}
\]

Therefore

\[
\sum_j [W_{ij}Z_{ii} - W_{ji}Z_{ji}] = W_{0i}Q_{0}. \tag{3.3.7}
\]

Using the same argument used to show relation (3.2.7), it can be seen that

\[
Z_{ik}^* = \sum_{j \neq 0,k} W_{ji}Z_{jk} \quad \text{and} \tag{3.3.8}
\]

\[
Z_{ik}^* = Z_{ik} \sum_{j \neq 0,k} W_{ij}, \tag{3.3.9}
\]
for \( i \neq 0, k \). Therefore

\[
\sum_{j \neq 0, k} [W_{ij}Z_{ik} - W_{ji}Z_{jk}] = 0, \quad \forall i \neq 0, k. \quad (3.3.10)
\]

Using the relations between \( Q_i \) and \( Z_{ij} \) in equations (3.3.1) and (3.3.2), equation (3.2.7) gives

\[
W_{i0} \sum_{j \neq 0} Z_{ij} - W_{0i}Q_0 + \sum_{j \neq 0} \sum_{k \neq 0} [W_{ij}Z_{ik} - W_{ji}Z_{jk}] = 0, \quad (3.3.11)
\]

for all \( i \neq 0 \). Equations (3.3.7) and (3.3.10) then simplify this to

\[
\sum_{j \neq 0} [(W_{i0} + W_{ij})Z_{ij} - W_{ji}Z_{jj}] = 0. \quad (3.3.12)
\]

Each component of the sum over \( j \) contains a unique element \( W_{0j} \) and thus equation 3.3.12 becomes

\[
(W_{i0} + W_{ij})Z_{ij} - W_{ji}Z_{jj} = 0, \quad (3.3.13)
\]

for all \( i, j \neq 0 \).

Therefore using equations (3.3.10) and (3.3.13)

\[
\sum_{j} [W_{ij}Z_{ik} - W_{ji}Z_{jk}] = 0, \quad \forall i \neq 0, k. \quad (3.3.14)
\]

3.4 Dynamic Properties

The probabilistic steady state allows the calculation of the dynamic properties of the system. Again we have a periodic system with \( n \) states and a rate from state \( i \) to state \( j \) is denoted by the directionless rate \( W_{ij} \). However, now physical distances between states must also be specified. State \( i \) is a distance \( d_i \) from reference state \( 0 \) and the total physical distance over the whole period is \( d \) (Figure 2.1).

We want to calculate the average velocity \( v \) and the dispersion \( D \) of molecules in the system and so we consider the movement of molecules in physical space along a periodically repeating lattice of physical sites. The probability of being in the \( ith \) site on the \( sth \) cycle is denoted by \( p_{is} \). Each site is connected to \( n \) sites forwards and \( n - 1 \) sites backwards, assuming that a molecule cannot jump a cycle length or longer for simplicity. Jumping a cycle length does not modify the governing equations and so is omitted. For example, site \((0, s)\) is connected to each site \((i, s)\) and \((i, s - 1)\) for all \( i \neq 0 \). The forwards and backwards transition rates from site \( i \) to site \( j \) are \( u_{ij} \) and \( w_{ij} \) respectively. The distance from site \((0, s)\) to site \((i, s)\) is denoted by \( d_i \) and \( d_0 = 0 \). This system is shown in Figure 2.1.
The site occupancy probabilities \( p_{i,s} \) are given by
\[
\frac{dp_{i,s}}{dt} = \sum_{j=i+1}^{n-1} (w_{ji}p_{j,s} + u_{ji}p_{j,s-1}) + \sum_{j=0}^{i-1} (w_{ji}p_{j,s+1} + u_{ji}p_{j,s}) - \sum_j (u_{ij} + w_{ij})p_{i,s}.
\] (3.4.1)

Here each \( s \) is associated with one repeat of the physical-space lattice, and \( \sum_s p_{i,s} = P_i \). Note that when \( i = 0 \), \( \sum_{j=0}^{i-1} \) is defined to give 0 and similarly when \( i = n - 1 \) for \( \sum_{j=i+1}^{n-1} \).

### 3.4.1 Velocity

The average displacement of a molecule along the track is given by
\[
< x > = \sum_s \sum_i p_{i,s} (ds + d_i) = \sum_i (dX_i + d_i P_i),
\] (3.4.2)

where
\[
X_i \equiv \sum_s sp_{i,s}.
\] (3.4.3)

Assuming that the system is in its steady state we have \( \frac{dP_i}{dt} = 0 \) and so
\[
V \equiv \frac{d < x >}{dt} = d \sum_i \frac{dX_i}{dt}.
\] (3.4.4)

Multiplying equation (3.4.1) by \( s \) and summing over \( s \) gives
\[
\frac{dX_i}{dt} = \sum_{j=i+1}^{n-1} \left( w_{ji}X_j + u_{ji} \sum_s (s + 1)p_{j,s} \right) + \sum_{j=0}^{i-1} \left( w_{ji} \sum_s (s - 1)p_{j,s} + u_{ji}X_j \right) - \sum_j (u_{ij} + w_{ij})X_i,
\] (3.4.5)
and so
\[
\frac{dX_i}{dt} = \sum_j (W_{ji}X_j - W_{ij}X_i) + \sum_{j=i+1}^{n-1} u_{ji}P_j - \sum_{j=0}^{i-1} w_{ji}P_j, \quad (3.4.6)
\]
recognizing that \( W_{ij} = u_{ij} + w_{ij} \) when mapping the physical-space system onto the state-space one.

Therefore,
\[
\sum_i \frac{dX_i}{dt} = \sum_i \sum_{j=0}^{i-1} (u_{ji} - w_{ji}) P_j. \quad (3.4.7)
\]

Equations 3.4.7 and 3.4.4 give
\[
V = d \sum_i \sum_{j=0}^{i-1} (u_{ji}P_j - w_{ji}P_j), \quad (3.4.8)
\]
exactly as expected from flux balance [50]. Note the velocity is independent of the \( d_i \) and only depends on the total step size \( d \).

### 3.4.2 Dispersion

The dispersion of the velocities around their mean is defined to be
\[
D \equiv \frac{1}{2} \lim_{t \to \infty} \frac{d}{dt} \left( < x^2 > - < x >^2 \right). \quad (3.4.9)
\]
The mean squared displacement is given by
\[
<x^2> = \sum_s \sum_i p_{i,s}(ds + d_i)^2,
\]
and
\[
= \sum_i \left( d^2 \alpha_i + 2dd_iX_i + d_i^2 P_i \right), \quad (3.4.10)
\]
where \( \alpha_i = \sum_s s^2 p_{i,s} \). Therefore, using equation (3.4.4) we have
\[
D = \sum_i \left[ d^2 \frac{d\alpha_i}{dt} + 2dd_i \frac{dX_i}{dt} - 2VdX_i - 2Vd_iP_i \right] \quad (3.4.11)
\]
at steady state.

Similarly to equation (3.4.6) we have
\[
\frac{d\alpha_i}{dt} = \sum_j (W_{ji}\alpha_j - W_{ij}\alpha_i) + \sum_{j=0}^{i-1} [w_{ji}(P_j - 2X_j) + u_{ji}(P_j + 2X_j)] \quad (3.4.12)
\]
and hence

\[ D = d^2 \sum_{i=0}^{i-1} \sum_{j=0}^{i-1} [w_{ji}(P_j - 2X_j) + u_{ji}(P_j + 2X_j)] \]

\[ + 2 \sum_{i} \left( \frac{d}{dt} dX_i - V dX_i - V d_i P_i \right). \]  

(3.4.13)

Therefore to calculate the dispersion we must first find the \( X_i \).

Following experimental observations [16] we assume a constant velocity solution \( X_i = g_i t + h_i \). The balance of constant terms in equation (3.4.6) gives solutions for the \( h_i \) in terms of the \( g_i \). We also have a a normalisation condition for the \( g_i \):

\[ \sum_i g_i = \sum_i \frac{dX_i}{dt} = \sum_i \sum_{j=0}^{i-1} (u_{ji} - w_{ji}) P_j. \]  

(3.4.14)

Linear with time terms in equation (3.4.6) are the same as the governing equations (3.2.3) for the steady-state probabilities \( P_i \). Therefore \( g_i \propto P_i \) and normalizing gives

\[ g_i = P_i \sum_k \sum_{j=0}^{k-1} (u_{jk} - w_{jk}) P_j = P_i \frac{V}{d}. \]  

(3.4.15)

Only the \( h_i \) remain to be determined in order to calculate \( X_i \). Matching up constant terms in equation (3.4.6) gives

\[ g_i = \sum_j [W_{ji} h_j - W_{ij} h_i] \]

\[ + \sum_{j=i+1}^{n-1} u_{ji} P_j - \sum_{j=0}^{i-1} w_{ji} P_j \]  

(3.4.16)

which can be written

\[ G_i = \sum_j W_{ji} h_j - \sum_j W_{ij} h_i, \]  

(3.4.17)

defining \( G_i \). Therefore \( \sum_j G_j = 0 \) and can be written

\[ G = Mh. \]  

(3.4.18)

\( M \) is a singular matrix as \( MP = 0 \) with \( \sum_i P_i = 1 \). Equation (3.4.18) cannot be solved explicitly for an arbitrarily sized system by standard matrix methods.

We have \( \sum_j G_j = 0 \) and choose the ansatz

\[ h_0 = 0, \quad h_i = -\frac{\sum_{j \neq 0} G_j C_{ij}}{Q_0}, \quad i \neq 0, \]  

(3.4.19)
where again $Q_0$ is the sum over all configurations of state 0 with the $C_{ij}$ defined as

$$C_{ij} = \frac{Z_{ij}}{W_{0j}}, \quad i \neq 0, \quad (3.4.20)$$

where the $Z_{ij}$ are the sum over all configurations of state $i$ that include a rate $W_{0j}$.

Equation (3.4.17) and (3.4.19) require that the $C_{ij}$ satisfy

$$G_i Q_0 = \sum_k G_k \left[ \sum_j W_{ij} C_{ik} - \sum_j W_{ji} C_{jk} \right] \quad (3.4.21)$$

where again $Q_0$ is a sum over all configurations of state 0. Note the $k = 0$ term in the summation is 0 by definition of the $C_{ij}$.

For $i = 0$, we have

$$\sum_k G_k \left[ \sum_j W_{0j} C_{0k} - \sum_j W_{j0} C_{jk} \right]$$

$$= -\sum_{k \neq 0} G_k \sum_j W_{j0} C_{jk},$$

$$= -\sum_{k \neq 0} G_k Q_0, \text{ from equation 3.3.2}$$

$$= G_0 Q_0, \quad (3.4.22)$$

since $\sum_j G_j = 0$. For $i \neq 0$, we have

$$\sum_k G_k \sum_j \left[ W_{ij} C_{ik} - W_{ji} C_{jk} \right]$$

$$= G_i \sum_j \left[ W_{ij} C_{ii} - W_{ji} C_{ji} \right]$$

$$+ \sum_{k \neq i} G_k \sum_j \left[ W_{ij} C_{ik} - W_{ji} C_{jk} \right],$$

$$= \frac{G_i}{W_{0i}} \sum_j \left[ W_{ij} Z_{ii} - W_{ji} Z_{ji} \right]$$

$$+ \sum_{k \neq i} \frac{G_k}{W_{0k}} \sum_j \left[ W_{ij} Z_{ik} - W_{ji} Z_{jk} \right],$$

$$= G_i Q_0, \quad (3.4.23)$$

using equations (3.3.7) and (3.3.14). Thus the choice of the $C_{ij}$ in equation (3.4.20) satisfies equation (3.4.21) and therefore the ansatz (3.4.19) is correct.
Using $g_i = P_i V/d$ it can be seen that

$$D = \frac{d^2}{2} \lim_{t \to \infty} \sum_{i,j=0}^{i-1} [w_{ji}(P_j - 2X_j) + u_{ji}(P_j + 2X_j)]$$

$$-d \lim_{t \to \infty} V \sum_i X_i,$$

$$= d^2 \sum_i \lim_{t \to \infty} \left( \sum_{j=0}^{i-1} [u_{ji}h_j - w_{ji}h_j] - \frac{V}{d} h_i \right)$$

$$+ d^2 \sum_i \frac{1}{2} \sum_{j=0}^{i-1} [u_{ji}P_j + w_{ji}P_j]. \quad (3.4.24)$$

The dispersion in terms of the $C_{ij}$ is therefore

$$D = d^2 \sum_i \sum_{j=0}^{i-1} \frac{1}{2}(u_{ji} + w_{ji})P_j$$

$$+ \frac{d^2}{Q_0} \sum_i \sum_{j=0}^{i-1} [w_{ji} - u_{ji}] \sum_{k \neq 0} G_{k,j}C_{j,k}$$

$$+ \frac{dV}{Q_0} \sum_i \sum_{j \neq 0} G_{j}C_{i,j}, \quad (3.4.25)$$

with

$$G_i = \frac{V}{d} P_i - \sum_{j=i+1}^{n-1} u_{ji}P_j + \sum_{j=0}^{i-1} w_{ji}P_j; \quad (3.4.26)$$

and again the $C_{ij}$ are the sum over all configurations of state $i \neq 0$ given a non-zero rate from 0 to $j$ and divided by that rate. It can be deduced from the definition of the $C_{ij}$ that

$$Q_0 = \sum_{j \neq 0} W_{j0}C_{j,k}, \quad \forall k \neq 0, \quad (3.4.27)$$

$$Q_i = \sum_{k \neq 0} W_{0k}C_{ik} i \neq 0, \quad (3.4.28)$$

which become the steady-state probabilities $P_i$ once normalised.

Equations for the velocity (3.4.8) and the dispersion(3.4.25) are general under the assumptions that the states lie on a 1D physical lattice and that it is not possible for a molecule cannot jump a cycle length or longer. Only the $C_{ij}$ need to be calculated for each individual system as general explicit equations have been derived in terms of them for the state-occupancy probabilities (equations (3.4.27) and (3.4.28)), the velocity (equation (3.4.8)) and the dispersion (equation (3.4.25)). Systems with high degrees of symmetry can greatly simplify these calculations as will be shown in section 3.5.
The randomness ratio is given by

\[ \rho = \frac{2D}{dV}. \] (3.4.29)

Note that assuming that the transition rates are independent of the substeps \( d_i \), the velocity, dispersion and therefore randomness ratio are also.

\( C_{ij} = \frac{Z_{ij}}{W_{0,j}} \), they are the sums over all configurations of state \( i \neq 0 \) given a rate from 0 to \( j \) and dividing through by that rate. Only these need to be calculated to give the state-occupancy probabilities, the velocity (equation (3.4.8)) and the dispersion (equation (3.4.25)). Systems with high degrees of symmetry can greatly simplify the calculation of the \( C_{ij} \).

### 3.5 Calculating the \( C_{ij} \)

Equations (3.4.8) and (3.4.25) give general expressions for the velocity and the dispersion respectively for any 1D hopping system assuming a molecule cannot jump one repeat or more of the lattice of physical sites. The \( C_{ij} \) must be derived for any individual system structure, however analysis of the architecture can greatly simplify the calculation.

Each term of a given \( C_{ij} \) must obey three rules. Firstly it must contain a rate path from \( j \) to \( i \) not through 0. Secondly for any state \( a \neq 0, i, j \) it must contain exactly one rate path from \( a \) to \( i \) or \( a \) to 0. Thirdly it must contain exactly one transition rate from each state except 0 and \( i \) - from which there should be none.

For illustrative purposes, the \( C_{ij} \) for the three smallest completely general systems are given. The \( n = 2 \) system has

\[ C_{11} = 1, \]

the \( n = 3 \) system has

\[ C_{11} = W_{20} + W_{21}, \]
\[ C_{12} = W_{21}, \]
\[ C_{21} = W_{12}, \]
\[ C_{22} = W_{10} + W_{12}, \]
and the $n = 4$ system has

\[
C_{11} = (W_{20} + W_{21})(W_{30} + W_{31} + W_{32}) + W_{23}(W_{30} + W_{31}), \quad (3.5.1)
\]
\[
C_{12} = W_{21}(W_{30} + W_{31} + W_{32}) + W_{23}W_{31}, \quad (3.5.2)
\]
\[
C_{13} = W_{31}(W_{20} + W_{21} + W_{23}) + W_{32}W_{21}, \quad (3.5.3)
\]
\[
C_{21} = W_{12}(W_{30} + W_{31} + W_{32}) + W_{13}W_{32}, \quad (3.5.4)
\]
\[
C_{22} = (W_{10} + W_{12})(W_{30} + W_{31} + W_{32}) + W_{13}(W_{32} + W_{30}), \quad (3.5.5)
\]
\[
C_{23} = W_{32}(W_{12} + W_{10} + W_{13}) + W_{31}W_{12}, \quad (3.5.6)
\]
\[
C_{31} = W_{13}(W_{20} + W_{21} + W_{23}) + W_{12}W_{23}, \quad (3.5.7)
\]
\[
C_{32} = W_{23}(W_{10} + W_{12} + W_{13}) + W_{21}W_{13}, \quad (3.5.8)
\]
\[
C_{33} = (W_{10} + W_{13})(W_{20} + W_{21} + W_{23}) + W_{12}(W_{20} + W_{23}). \quad (3.5.9)
\]

The calculation of $C_{13}$ is explained as an example in Figure 3.3.
Figure 3.3: A graphical example of how $C_{13}$ is calculated for the $n = 4$ system. Each rate $W_{ij}$ included in $C_{13}$ is shown as a solid arrow. $C_{13}$ is the sum over all configurations of state 1 given a non-zero rate from 0 to 3 shown as a dashed arrow. The algebraic interpretation is displayed below the graphical one. Note that to satisfy the definition of a configuration of 1 there is a rate path from any state to state 1. Each $C_{ij}$ can be interpreted in this manner.
It is also possible to consider arbitrary-sized systems in terms of branched states and coupling states. Considering a branch \( k \) of states with nearest neighbour interactions and defining \( u_i^k \) as the rate from \( i \) to \( i+1 \) and \( w_i^k \) as the rate from \( i \) to \( i-1 \), the following notation is useful when writing down configurations:

\[
\begin{align*}
\Lambda_a^b &= \prod_{i=a}^{b} u_i^k, \\
\Pi_a^b &= \prod_{i=a}^{b} w_i^k, \\
\Xi_a^b &= k\Lambda_a^b \left( 1 + \sum_{j=a}^{b} \prod_{i=a}^{j} \frac{u_i^k}{w_i^k} \right),
\end{align*}
\]

and define that each expression becomes unity if no rates are included, for example \( \Xi_a^{a-1} = 1 \) unless otherwise stated.

The \( \Lambda_a^b \) and the \( \Pi_a^b \) represent a path and a reversed rate path respectively between \( a \) and \( b \) on branch \( k \). This is shown graphically in Figure 3.4. Multiplying several of these components together and ensuring the indices do not overlap gives the properties of all the terms. For example, \( k\Xi_a^b\Xi_{b+1}^a \) represents the sum of all combinations of path reversals between \( a \) and \( b \) and between \( b+1 \) and \( c \) on branch \( k \). \( k\Xi_a^b\Xi_c^d \) represents the sum of all combinations of path reversals between \( a \) and \( b \) on branch \( k \) and all combinations of path reversals between \( c \) and \( d \) on branch \( k' \).

A given \( C_{ij} \) is written in terms of the rates from coupling states and rates from states on a branch. Coupling-state rates appear explicitly in the equations, whilst branch-state rates can be grouped together using relations (3.5.10), (3.5.11) and (3.5.12). For example, considering the simplest system architecture of only one branch and no coupling states known as the single chain (Figure 3.5a) we have

\[
\begin{align*}
C_{ij} &= u_j\Lambda_{j+1}^{i-1}\Xi_{i+1}^{j-1}\Xi_i^{n-1}, \quad i > j, \\
C_{ij} &= w_j\Pi_{j+1}^{i-1}\Xi_{i+1}^{j-1}\Xi_i^{n-1}, \quad i < j,
\end{align*}
\]

with the \( i = j \) case given by equation (3.5.13) with \( u_j\Lambda_{j+1}^{i-1} := 1 \). In this notation \( C_{ij} \) must be written as three separate equations because the relative locations of states \( i \) and \( j \) are important. A system with two branches and one coupling states defined to be the reference state 0, the parallel pathway (Figure 3.5b), has

\[
\begin{align*}
C_{ij}^k &= u_j^k\Lambda_{j+1}^{i-1}k\Xi_{i+1}^{j-1}k^{-1}\Xi_i^{n-1}k^{-1}, \quad i > j, \\
C_{ij}^k &= w_j^k\Pi_{j+1}^{i-1}k\Xi_{i+1}^{j-1}k^{-1}\Xi_i^{n-1}k^{-1}, \quad i < j,
\end{align*}
\]

with \( k, k' \in \{1, 2\} \) and the \( i = j \) case given by equation (3.5.15) with \( u_j^k\Lambda_{j+1}^{i-1} := 1 \). The case where the \( j, k \) state lies on a different branch to the \( i, k' \) state (i.e. \( k \neq k' \))
3.5. **CALCULATING THE C\textsubscript{ij}**

Figure 3.4: A graphical representation of $k_\text{ab}$ the rate path from a to b, $k_\text{ba}$ the reversed rate path from b to a and $k_\text{ab} + k_\text{ba}$ the sum of all rate path reversals between a and b all along a branch k.

Figure 3.5: A structural representation of a) the single chain, b) the two branch parallel pathway, c) the divided pathway and d) the divided pathway with a chain coupled at state 0. Dots represent coupling states and lines represent branches.
CHAPTER 3. CALCULATING DYNAMIC PROPERTIES GENERALLY

has $C_{ij} = 0$ because no configurations of $i$ given a rate $W_{0,(j,k)}$ exist as there is no rate path from $j,k$ to $i,k'$ that does not pass through 0.

Note that each expression can be derived from the other; in changing the order of the $i$ and the $j$, $i \geq j \leftrightarrow i < j$, three transformations must be applied:

\begin{align*}
    u_j^k & \leftrightarrow w_j^k, \quad (3.5.17) \\
    k\Xi_j^l, k\Lambda_j^l, k\Pi_j^l & \leftrightarrow k\Xi_l^i, k\Lambda_l^i, k\Pi_l^i, \quad (3.5.18) \\
    k\Lambda & \leftrightarrow k\Pi, \quad (3.5.19)
\end{align*}

namely the rate from the $j$th state changes direction, the $i$s and $j$s in the indices of the grouped branch terms are swapped and the $\Lambda$ and the $\Pi$ are swapped. In this manner only a few expressions for the $C_{ij}$ need to be written down, the rest can be deduced from these.

Introducing another coupling state adds another level of complexity. For a given $C_{ij}$, states $i$ and $j$ can now be this coupling state and not just branch states and so more than three expressions are required to fully define all the $C_{ij}$. For a system with two branches and two coupling states, defining one of the coupling states as the reference state 0 and denoting the other by $m$ we have the divided pathway [12], Figure 3.5c. For $0 < j < i < m$

\begin{equation}
    C_{ij} = u_j k\Lambda_{j+1}^{l-1} \Xi_{l+1}^{j-1} [w_m k\Pi_{i+1}^{m-1} k\Xi_{i+1}^{m-1} k\Pi_{i+1}^{m-1} k\Xi_{i+1}^{m-1} k\Pi_{i+1}^{m-1} k\Xi_{i+1}^{m-1} k\Pi_{i+1}^{m-1}]
\end{equation}

however it is much simpler to consider this in graphical form as in Figure 3.6. The case where $0 < j = i < m$ is recovered by sending $u_j k\Lambda_{j+1}^{l-1} \Xi_{l+1}^{j-1} \rightarrow 1$. The case $0 < i < j < m$ is given by applying transformations given by relations (3.5.17), (3.5.18) and (3.5.19) and an additional $u_m^k \leftrightarrow w_m$ transformation to take into account the extra coupling state.

In this manner all the $C_{ij}$ for the general $n$-state divided pathway can be written down. This is presented in section 3.6.3

### 3.5.1 Modifying System Structures

The method discussed in this chapter preserves the structure of a given architecture within the calculations. This allows solutions for one architecture to be deduced from solutions to another, for example by modifying the structure of the system as demonstrated in this section.

Modifying the single chain model into the parallel pathway model is akin to adding additional single chains into the system coupled at 0. Note that the parallel chain results given in equations (3.5.15) and (3.5.16) are the single chain results multiplied by the product of each additional branch of the configuration of state 0 restricted to that branch.
3.5. CALCULATING THE $C_{ij}$

Figure 3.6: A graphical representation of the $C_{ij}$ for a divided pathway model for $0 < j < i < m$. Each figure part represents a term in equation (3.5.20) and contains all possible components of this $C_{ij}$ where the coupling state rate points a) backwards, b) forwards along branch $k$ and c) forwards along branch $k'$. Indices on the branched terms have been omitted - each symbol pertains to the branch states bracketed by the vertical dashed lines.
It is straightforward to see that if we define $S^k_0$ to be the sum of all configurations of $0$ restricted to the single chain branch $k$ and $S^k_{ij}$ to be the single chain $C_{ij}$ restricted to branch $k$, we can rewrite equations (3.5.15) and (3.5.16) to get the parallel pathway $PC^k_{ij}$ from the single chain:

$$PC^k_{ij} = S^k_{ij} \prod_{k' \neq k} S^{k'}_0.$$  \hfill (3.5.21)

Further results can be derived using these combinatorial considerations of the $C_{ij}$. For example the divided pathway with an additional single chain coupled at $0$ (Figure 3.5d) would have

$$D^+SC_{ij} = D^C_{ij}S^D_0, \quad i,j \in D \text{ section},$$

$$D^+SC_{ij} = S^C_{ij}D^Q_0, \quad i,j \in S \text{ section},$$

$$D^+SC_{ij} = 0, \quad \text{otherwise},$$  \hfill (3.5.22) \hfill (3.5.23) \hfill (3.5.24)

where $D$ represents the divided pathway, $S$ represents the single chain, $D^C_{ij}, D^Q_0$ are the $C_{ij}$ and the $Q_0$ respectively for just the divided pathway section and $S^C_{ij}, S^Q_0$ are the $C_{ij}$ and the $Q_0$ respectively for just the single chain section.

In general for two structures $A$ and $B$ coupled only at the reference state, the $A^+B^C_{ij}$ for the combined structure can be written in terms of individual component structure variables

$$A^+B^C_{ij} = A^C_{ij}B^Q_0, \quad i,j \in A \text{ section},$$

$$A^+B^C_{ij} = B^C_{ij}A^Q_0, \quad i,j \in B \text{ section},$$

$$A^+B^C_{ij} = 0, \quad \text{otherwise}.$$  \hfill (3.5.25) \hfill (3.5.26) \hfill (3.5.27)

In this manner one can derive results for the probabilistic steady state and dispersion for arbitrarily large and highly complex systems by splitting them into individual component systems coupled at the reference state and computing the $C_{ij}$ for each system individually. This is a very useful tool for investigating complex underlying biochemical pathways of a molecular motor.

### 3.6 Generalised System Architectures

Many biologically-inspired systems suggest that there is at least one state shared between all the forwards mechanochemical cycles, for example with myosin-V [1, 5, 41, 42, 44, 52, 55]. Defining this state to be $0$ we can use this as a boundary of the periodic lattice of physical sites. Systems that do not have this property require a minimum of 3 branches and 4 coupling states and for simplicity we will not consider these here. Therefore physical sites $(0, s)$ are connected to $2n - 2$ other sites and sites $(i \neq 0, s)$ are connected to $n - 1$ other sites. This is similar to the system in Figure 2.1 except sites $(i \neq 0, s)$ only have possible transitions to other
sites \((j \neq i, s)\) and \((0, s + 1)\). The equation for the velocity can now be simplified to

\[
V = d \sum_{i \neq 0} [u_{i0} P_i - w_{0i} P_0],
\]

and the equation for the dispersion can also be simplified to

\[
D = \frac{d^2}{2} \sum_i (u_{i0} P_i + w_{0i} P_0) - \frac{d^2}{Q_0} \sum_i u_{i0} \sum_{j \neq 0} G_{ij} C_{ij} + \frac{dV}{Q_0} \sum_i \sum_{j \neq 0} G_{ij} C_{ij},
\]

with

\[
G_i = \frac{V}{d} P_i + w_{0i} P_0, \text{ for } i \neq 0.
\]

From now on we will only consider systems of this type.

There are many different classes of \(n\)-state system architectures, each defined by the conditions on the transition rates.

### 3.6.1 Single Chain Model

A single chain model is a system with only nearest neighbour transitions as shown in Figure 3.7a. The state occupancy probabilities \(P_i\) are governed by \(n\) master equations

\[
\frac{dP_0}{dt} = u_{n-1} P_{n-1} + w_1 P_1 - (u_0 + w_0) P_0, \quad (3.6.4)
\]

\[
\frac{dP_1}{dt} = u_0 P_0 + w_2 P_2 - (u_1 + w_1) P_1, \quad (3.6.5)
\]

\[
\vdots
\]

\[
\frac{dP_{n-2}}{dt} = u_{n-2} P_{n-2} + w_0 P_0 - (u_{n-1} + w_{n-1}) P_{n-1}, \quad (3.6.6)
\]

The steady state of the system is given by the methods in section 3.2. In this case it is exactly the result shown by Derrida [13]

\[
Q_i = \frac{P_i}{N} = \frac{e_1}{u_i} \left( 1 + \sum_{j=i+1}^{i-1} \prod_{l=i+1}^{j} \frac{w_l}{u_l} \right), \quad (3.6.7)
\]
Figure 3.7: Three potential molecular motor model structures. a) A $n$-state single chain system with $n$ substeps and $n$ different molecular detachment pathways. Forward rates are denoted by $u_i$ and the backward rates by $w_i$. b) A parallel pathway reaction network with an arbitrary number of branches. c) A generalised divided pathway reaction network. Distances between states are not shown.
where $N$ is the normalising factor so that the probabilities sum to unity and

$$e_1 = \prod_{i=0}^{n-1} u_i.$$  

(3.6.8)

Therefore,

$$Q_i = \Xi_{i+1}^{-1},$$  

(3.6.9)

with indices being taken modulo $n$ as state $0 \equiv n$.

Equation (3.6.1) gives the velocity of the system

$$V = d [u_{n-1}P_{n-1} - w_0P_0],$$  

(3.6.10)

and equation (3.6.2) gives the dispersion of the system

$$D = \frac{d^2}{2} (u_{n-1}P_{n-1} + w_0P_0)$$

$$- \frac{d^2}{Q_0} u_{n-1} \sum_{j \neq 0} G_j C_{n-1,j},$$

$$+ \frac{dV}{Q_0} \sum_i \sum_{j \neq 0} G_j C_{ij},$$

(3.6.11)

with

$$G_i = \frac{V}{d} P_i + w_0P_0, \text{ for } i \neq 0,$$

(3.6.12)

and

$$C_{ij} = \Lambda_{i+1}^{-1} \Xi_i^{-1} \Xi_{i+1}^{-1}, \text{ for } i \geq j,$$

(3.6.13)

$$C_{ij} = \Pi_{i+1}^{-1} \Xi_i^{-1} \Xi_{j+1}^{-1}, \text{ for } i < j.$$  

(3.6.14)

This is exactly the result shown by Derrida [13].

### 3.6.2 Parallel Pathway Model

The parallel pathway model is a simple modification to the single chain model. In this section we demonstrate how the single chain solution can be modified to give us the parallel pathway result. For the parallel pathway model with an arbitrary number of branches (enumerated by superscript $k$) the architecture is shown in Figure 3.7b and the steady state is given by

$$Q_0 = \frac{P_0}{N} = \sum_k k \Xi_1^{k-1}$$  

(3.6.15)

$$Q_i^k = \frac{P_i^k}{N} = \frac{k \Xi_i^{-1}}{i+1} \sum_{k' \neq k} k' \Xi_1^{k'-1},$$  

(3.6.16)
Equation (3.6.1) gives the velocity

\[ V = d \sum_k \left[ u_{nk-1}^k P_{nk-1}^k - w_0^k P_0^k \right]. \]  

(3.6.17)

The dispersion is given by equation (3.6.2)

\[ D = \frac{d^2}{2} \sum_k (u_{nk-1}^k P_{nk-1}^k + w_0^k P_0^k) \]
\[ - \frac{d^2}{Q_0} \sum_k u_{nk-1}^k \sum_{j \neq 0} G_{i}^{k} C_{j}^{k}_{nk-1,j} \]
\[ + \frac{dv}{Q_0} \sum_k \sum_i \sum_{j \neq 0} G_{j}^{k} C_{i}^{k}_{ij}, \]  

(3.6.18)

with

\[ G_{i}^{k} = \frac{v}{d} P_{i}^{k} + w_0^k P_0^k, \] for \( i \neq 0, \)  

(3.6.19)

where

\[ C_{i}^{k}_{ij} = k_{i}^{j-1} k_{j}^{i-1} k_{j+1}^{i+n-1} \prod_{k' \neq k} k'_{i+j}'_{j+1}', \]  

(3.6.20)

for \( i \geq j, \) and

\[ C_{i}^{k}_{ij} = k_{i}^{j-1} k_{j}^{i-1} k_{j+1}^{i+n-1} \prod_{k' \neq k} k'_{i+j}'_{j+1}', \]  

(3.6.21)

for \( i < j. \)

Restricting to only two branches, \( k \in [1, 2], \) the resulting dispersion gives exactly the result shown by Kolomeisky [26]. This result is general for any number of parallel branches.

### 3.6.3 The Divided Pathway

The divided pathway [12] (see Figure 3.5c) has two coupling states and three branches. The velocity, dispersion and the \( G_{i}^{k} \) relations for divided pathway are the same as those for the parallel pathway system and are given by equations (3.6.17), (3.6.18) and (3.6.19) respectively: here \( k \in \{1, 2\} \) where each \( k \) represents a different divided branch. The \( C_{i,j} \) for the divided pathway however are different to those for the parallel pathway.

For a given \( C_{ij} \), states \( i \) and \( j \) can represent a coupling state and not just branch states. There are fourteen different expressions for the \( C_{ij} \), however it is sufficient to write down six and describe the transformations needed to produce the others.
The first 5 are given by

\[ C_{ij} = u_j \Lambda_{j+1}^{i-1} \Xi_1^{j-i} \left[ w_m \Pi_{i+1}^{m-k} \Xi_{m+1}^{n-1-k'} \Xi_{m+1}^{n'-1} \right. \]
\[ + \Xi_{i+1}^{m-1} (u_m^k \Lambda_{m+1}^{i-1-k'} \Xi_{m+1}^{n'-1} \Xi_{m+1}^{n'-1}) \]
\[ \left. + u_m^{k'} \Lambda_{m+1}^{i-1-k} \Xi_{m+1}^{n-1} \Xi_{m+1}^{n'-1} \right], \tag{3.6.22} \]

for 0 < j < i < m,

\[ C_{ij} = u_j \Lambda_{j+1}^{i-1} \Xi_1^{j-i} \Xi_{i+1}^{n-1-k'} \Xi_{i+1}^{n'-1}, \tag{3.6.23} \]

for 0 < j < i = m,

\[ C_{ij} = u_m^k \Lambda_{i+1}^{m-1-k} \Lambda_{m-1}^{i-1-k} \Xi_{i+1}^{m-k} \Xi_{i+1}^{n-1-k'} \Xi_{i+1}^{n'-1}, \tag{3.6.24} \]

for 0 < j < m < i with i on branch k,

\[ C_{ij} = u_j^{kk} \Lambda_{j+1}^{i-1-k} \Xi_1^{j-i} \Xi_{i+1}^{n-1-k'} \Xi_{i+1}^{n'-1}, \tag{3.6.25} \]

for 0 < j = m < i and

\[ C_{ij} = u_j^{kk} \Lambda_{j+1}^{i-1-k} \Xi_1^{j-i} \Xi_{i+1}^{m-1-k} \Xi_{i+1}^{n-1-k'} \Xi_{i+1}^{n'-1} \Xi_{i+1}^{j+1} \]
\[ \times \left[ w_m^k \Xi_{m+1}^{j-1} \Pi_{m+1}^{i-k'} \Xi_{m+1}^{n-1-k'} \Xi_{m+1}^{n'-1} \Xi_{m+1}^{j+1} \right. \]
\[ + u_m^k \Lambda_{m+1}^{i-i-1-k} \Xi_{m+1}^{n-1-k'} \Xi_{m+1}^{n'-1} \Xi_{m+1}^{j+1} \Xi_{m+1}^{j+1} \]
\[ \left. + u_m^{k'} \Lambda_{m+1}^{i-i-1-k} \Xi_{m+1}^{j-i-1-k} \Xi_{m+1}^{n-1-k'} \Xi_{m+1}^{n'-1} \Xi_{m+1}^{j+1} \right], \tag{3.6.26} \]

for 0 < m < j < i.

The next five terms 0 ≤ i ≤ j ≤ m, 0 < i ≤ j = m, 0 < i < m < j, 0 < i = m < j, 0 < m < i ≤ j are given by applying four transformations. The transformations given by relations (3.5.17), (3.5.18) and (3.5.19) and an additional \( u_m^k \leftrightarrow w_m \) transformation to take into account the extra coupling state.

The cases where 0 < j = i < m, 0 < j = i = m and 0 < m < j = i are given by equations (3.6.22), (3.6.23) and (3.6.26) respectively with \( u_j^{kk} \Lambda_{j+1}^{i-1} \rightarrow 1 \).

The last term for \( m < i, j \) with i on branch k and j on branch \( k' \) is

\[ C_{ij} = u_m^k u_j^{k'} \Pi_{m+1}^{j-i-1-k} \Lambda_{m+1}^{i-1-k} \Xi_{m+1}^{j-i-1-k} \Xi_{m+1}^{n-1} \Xi_{j+1}^{n'-1}. \tag{3.6.27} \]

The steady-state probabilities are calculated from the \( C_{ij} \) using equations (3.4.27) and (3.4.28).

### 3.6.4 Divided Pathway with a Chain

The divided pathway with a chain is constructed by coupling a single chain to the reference state shown schematically in Figure 3.5d.
The velocity, dispersion and the $G_k^i$ relations for divided pathway with a chain are the same as those for the parallel pathway system (although the $C_{ij}$ are different) and are given by equations (3.6.17), (3.6.18) and (3.6.19) respectively. In contrast to the divided pathway $k \in \{1, 2, 3\}$ where the additional $k = 3$ branch represents the single chain and the $C_{ij}$ are derived from those of the single chain and the divided pathway using the method given in section 3.5.1.

The steady-state probabilities are again calculated from the $C_{ij}$ using equations (3.4.27) and (3.4.28).

### 3.7 Four State Model Example

Biologically interesting models for molecular motors exist that involve relatively few states [1, 41, 44, 61]. This more intuitive framework gives the dispersion much more readily than existing approaches for systems with smaller number of states. The generalised $n = 4$ system is shown in Figure 3.8 and has unnormalised probabilities

\[
Q_0 = W_{10}C_{1k} + W_{20}C_{2k} + W_{30}C_{3k}, \quad \forall k \neq 0
\]

\[
Q_1 = W_{01}C_{11} + W_{02}C_{12} + W_{03}C_{13},
\]

\[
Q_2 = W_{01}C_{21} + W_{02}C_{22} + W_{03}C_{23},
\]

\[
Q_3 = W_{01}C_{31} + W_{02}C_{32} + W_{03}C_{33},
\]

from equations (3.4.27) and (3.4.28), with normalised probabilities given by $P_i = Q_i / \sum_j Q_j$.

Equation (3.6.1) gives the velocity and the dispersion is given by equation (3.6.2) with the $C_{ij}$ as given in equations (3.5.1) - (3.5.9).
3.7. FOUR STATE MODEL EXAMPLE

![Figure 3.9: A toy single chain model with no backwards stepping.](image)

3.7.1 Single Chain

A toy 4-state single chain model with no backwards steps is shown in Figure 3.9 and has

\[ Q_{0}^{S} = u_{3}^{S} C_{31}^{S}, \]  \hspace{1cm} (3.7.5)

\[ Q_{1}^{S} = u_{0}^{S} C_{11}^{S}, \]  \hspace{1cm} (3.7.6)

\[ Q_{2}^{S} = u_{0}^{S} C_{21}^{S}, \]  \hspace{1cm} (3.7.7)

\[ Q_{3}^{S} = u_{0}^{S} C_{31}^{S}, \]  \hspace{1cm} (3.7.8)

with \( P_{i}^{S} = N_{i}^{S} Q_{i}^{S} \) with \( N_{i}^{S} = 1/\sum Q_{j}^{S} \). Equation (3.6.1) gives the velocity

\[ V^{S} = d u_{3}^{S} P_{3}^{S}. \]  \hspace{1cm} (3.7.9)

The dispersion is given by equation (3.6.2)

\[ D^{S} = \frac{d^{2}}{2} u_{n-1}^{S} P_{n-1}^{S} \]

\[ - \frac{dV^{S}}{Q_{0}^{S}} u_{n-1}^{S} \sum_{j \neq 0} P_{j}^{S} C_{n-1,j}^{S} \]

\[ + \frac{V^{S} V^{S}}{Q_{0}^{S}} \sum_{i} \sum_{j \neq 0} P_{j}^{S} C_{ij}^{S}, \]  \hspace{1cm} (3.7.10)

with

\[ C_{11}^{S} = u_{2}^{S} u_{3}^{S}, \]  \hspace{1cm} (3.7.11)

\[ C_{12}^{S} = 0, \]  \hspace{1cm} (3.7.12)

\[ C_{13}^{S} = 0, \]  \hspace{1cm} (3.7.13)

\[ C_{21}^{S} = u_{1}^{S} u_{3}^{S}, \]  \hspace{1cm} (3.7.14)

\[ C_{22}^{S} = u_{1}^{S} u_{3}^{S}, \]  \hspace{1cm} (3.7.15)

\[ C_{23}^{S} = 0, \]  \hspace{1cm} (3.7.16)

\[ C_{31}^{S} = u_{1}^{S} u_{2}^{S}, \]  \hspace{1cm} (3.7.17)

\[ C_{32}^{S} = u_{1}^{S} u_{2}^{S}, \]  \hspace{1cm} (3.7.18)

\[ C_{33}^{S} = u_{1}^{S} u_{2}^{S}. \]  \hspace{1cm} (3.7.19)

3.7.2 Divided Pathway with a Jump

The toy 4-state divided pathway model with a jump and no backwards steps is shown in Figure 3.10. When \( u_{3}^{D} = 0 \), this is a divided pathway model. For
simplicity, rates are chosen so that $u_{12} = u_{20}$ and $u_{13} = u_{30}$. For both of these systems

$$Q_0^D = u_1^D C_{21}^D + u_2^D C_{31}^D,$$  \hfill (3.7.20)

$$Q_1^D = u_0^D C_{11}^D,$$  \hfill (3.7.21)

$$Q_2^D = u_0^D C_{21}^D,$$  \hfill (3.7.22)

$$Q_3^D = u_0^D C_{31}^D,$$  \hfill (3.7.23)

with $P_i^D = N_i^D Q_i^D$ with $N_i^D = 1/\sum_j Q_j^D$. Equation (3.6.1) gives the velocity

$$V^D = d \left[ u_1^D P_2^D + u_2^D P_3^D \right].$$  \hfill (3.7.24)

The dispersion is given by equation (3.6.2)

$$D^D = \frac{d^2}{2} (u_1^D P_2^D + u_2^D P_3^D) - \frac{dV^D}{Q_0^D} \sum_{j \neq 0} P_j^D (u_1^D C_{1j}^D + u_2^D C_{2j}^D) + \frac{V^D V^D}{Q_0^D} \sum_i \sum_{j \neq 0} P_j^D C_{ij}^D,$$  \hfill (3.7.25)

The randomness ratio is then expressed as

$$\rho^D = \frac{2D^D}{dV^D}.$$  \hfill (3.7.26)
For the divided pathway with a jump:

\[
\begin{align*}
C_{11}^D &= u_1^D u_2^D + u_2^D u_3^D, \\
C_{12}^D &= 0, \\
C_{13}^D &= 0, \\
C_{21}^D &= u_1^D u_2^D, \\
C_{22}^D &= u_2^D (u_1^D + u_2^D), \\
C_{23}^D &= 0, \\
C_{31}^D &= u_2^D (u_1^D + u_3^D) + u_1^D u_3^D, \\
C_{32}^D &= u_3^D (u_1^D + u_2^D), \\
C_{33}^D &= (u_1^D + u_3^D)(u_1^D + u_2^D),
\end{align*}
\]

with the divided pathway result recovered when \( u_3^D = 0 \). It should be noted that \( C_{ij}^D = 0 \) when all rate paths from \( j \) to \( i \) pass through 0, in this architecture this is rate paths 2 to 1, 3 to 1 and 3 to 2.

### 3.7.3 Combining Model Structures

A divided pathway with a jump and a single chain (and no backwards stepping) is shown in Figure 3.11. As described in section 3.5.1 we can deduce the unnormalised probabilities to be

\[
\begin{align*}
Q_0 &= Q_0^S Q_0^D, \\
Q_1^S &= Q_0^D Q_1^S, \\
Q_2^S &= Q_0^D Q_2^S, \\
Q_3^S &= Q_0^D Q_3^S, \\
Q_1^D &= Q_3^S Q_1^D, \\
Q_2^D &= Q_3^S Q_2^D, \\
Q_3^D &= Q_3^S Q_3^D,
\end{align*}
\]

with \( P_i = N Q_i \) with \( N = 1/\sum_j Q_j \). Equation (3.6.1) gives the velocity

\[
V = N \left[ \frac{v^S}{N^S} + \frac{v^D}{N^D} \right].
\]

The dispersion is given by equation (3.6.2)

\[
D = \frac{d^2}{2} \left( u_3^S P_{3S}^S + u_2^D P_{2S}^S + u_3^D P_{3S}^S \right) \\
- \frac{dV}{Q_0} \sum_{j \neq 0} P_j \left( u_3^S C_{3Sj}^S + u_2^D C_{2Sj}^D + u_3^D C_{3Sj}^D \right) \\
+ \frac{V^2}{Q_0} \sum_{j \neq 0} P_j \sum_i C_{ij}.
\]
with

\[ C_{ij} = C_{ij}^D Q_0^S, \ i, j \in DJ \text{ section}, \]  \quad (3.7.45)

\[ C_{ij} = C_{ij}^S Q_0^D, \ i, j \in S \text{ section}, \]  \quad (3.7.46)

\[ C_{ij} = 0, \ \text{otherwise}, \]  \quad (3.7.47)

where DJ represents the divided pathway with a jump and S represents the single chain.

### 3.8 Discussion

A method to calculate measurable quantities of molecular motors described by discrete stochastic models has been presented in this chapter. The approach is based on the work of Derrida [13] that gives dynamic properties for a general n-state single chain model. Derrida's work has in the past been extended to other system structures [12, 26, 27] and in each case the velocity and dispersion were rederived. However this method - extending the work by King and Altman [24] - gives explicit expressions for the completely general steady-state probabilities, velocities and dispersions in terms of variables \( C_{ij} \) that depend on the system structure.

The expressions for the average velocity and dispersion were derived without any constraint on the distance between physical stepping sites. It was shown that the resulting equations were independent of the substep size \( d_i \) (assuming that the transition rates are also) and only depend on the total step size \( d \). Whilst apparent for systems without detachment, this was not obvious a priori for systems with unequal detachments between between successive substeps. Assuming the renormalisation procedure [27] is correct, we can map a system with detachment to one without by scaling reaction rates and steady-state probabilities. Thus these results still hold with rescaled rates and probabilities and so the velocity and dispersion are independent of the substep sizes \( d_i \) regardless of detachment rates.
3.8. **DISCUSSION**

It was shown that the $C_{ij}$ are derivable in a simple graphical manner from the structure of a proposed system and have written them down explicitly for several example structures. This approach gives general $n$-state system expressions for multiple system structures. Modifications of the generalised structure and their effect on the $C_{ij}$ have been explored. Results for a simple 4-state system demonstrate the simplicity of the calculations relative to other methods. Results for two separate 4-state systems can be combined in a simple manner to give the dispersion and velocity of an 8-state system.

Generalised results have been given for the single chain, parallel pathway and divided pathway systems and have used a technique for combining structures coupled at the reference state to derive the novel divided pathway with a chain results.

Alternative methods of calculating the dynamic properties exist. Tsygankov et al. [50] provide a flux-balance method to calculate the velocities for any system and Chemla et al. [6] use matrix methods to calculate the velocities and dispersions from a given system. However none can provide results for general model structures or solutions that can be interpreted in such a simple graphical way.

These methods provide powerful theoretical tools for investigating how the underlying transition rates of a molecular motor affect its dynamic properties. The dynamic properties of smaller models can now be calculated simply. This is used in the rest of this thesis to calculate relevant dynamic properties in a simple and timesaving manner.
Experimental work on molecular motors seeks to provide evidence for or against a particular hypothesis about the underlying mechanisms of the motor proteins using measurable observables. Theory seeks to understand the consequences of these ideas using mathematical descriptions (models) of this and informs the next generation of experimental investigation; this was most famously demonstrated by Kolomeisky and Fisher’s mathematical description of myosin-V [28] that accurately predicted the stepping sizes of its walk. For a given motor protein many such models exist [29, 57] but unfortunately can be in conflict with one another. This can be due to a fundamental difference in the assumed underlying biology but also can be due to the various different mathematical approaches that different groups take. It is very important to resolve the biological conflicts as they are central to establishing how the motor proteins function. When comparing models of molecular motors to identify biological ideas such as the stepping mechanism it is important to remove the mathematical differences to separate them from the biological ones. Despite much work on the subject, none of it does this by directly assessing the validity of competing underlying biological ideas in the same theoretical framework.

As discussed in section 1.3 there are many ideas as to how myosin-V functions. Rief et al. [41] used optical trap measurements to identify a possible stepping mechanism and constructed a model for the myosin-V protein in which both heads of the protein are sites for ATP hydrolysis and are coupled in such a way as to induce procession along the actin filament. In my work this particular mechanism
CHAPTER 4. MYOSIN-V STEPPING MODELS

is contained within all models and so I shall refer to this as the hydrolysis cycle. Veigel et al. [55] introduced the idea that strain had an important role to play in the coordination of the heads. Rosenfeld and Sweeney [42] identified the possibility of failed steps or a futile cycle. Skau et al. [44] took all these ideas and constructed a model that reproduced all but one of the accepted velocity and run length relations to varying nucleotide concentrations. The rogue result - a run length against ADP relationship - was produced by Baker et al. [1] who suggest an alternative model [61] that does give this result but then, as I demonstrate in the next chapter, fails to give the run length against ATP relationship. Uemura et al. [52] and Capello et al. [5] suggested that the protein moves in two and three movements or substeps respectively per hydrolysis cycle, the consequences of these competing ideas with relation to observable quantities is not known. In order to investigate the implications of these different model assumptions and compare the success of different models in explaining experimental data, a common framework is needed.

Mathematical models can have features that cannot be directly measured experimentally; these can be quantified by parameters that are often chosen numerically to ensure that the model reproduces key results. When comparing two models designed in a similar manner one can look at the differences between the two sets of parameters to help identify what each model is doing differently. In this chapter I set out a mathematical framework to encode an idea as to how a motor functions - its stepping mechanism - so that different models for myosin-V can be compared against each other. The aim is to identify the plausibility of different stepping mechanisms and to establish how consistent they are with experimental data.

Firstly a method developed by Skau et al. [44] to fit a discrete stochastic model of a molecular motor to experimental data numerically is described in section 4.1. A model containing many submodels of myosin-V (our metamodel) is presented in section 4.2 and will be used in future chapters as a framework to compare different models. The measure of how well a model of myosin-V reproduces an experimental result - the cost function - is then described in section 4.3. Finally in section 4.4 a modified optimisation routine is introduced to make this method more efficient numerically and therefore make this comparison of multiple models more feasible.

4.1 The Skau Optimisation Procedure

Skau et al. [44] constructed an optimisation procedure to fit a discrete stochastic model for a molecular motor to a set of experimental results. The parameters were separated into chemical and mechanical energy barriers and differences and the reaction rates were formulated energetically using Arrhenius expressions containing these parameters. The rate to go from the less energetic state \( j \) to the more energetic state \( i \) (usually backwards) is given by

\[
w_{ji} = \tau^{-1} e^{-\beta(G_{ij}^c + \Delta G_{ij})},
\]

where \( \tau \) is the fundamental timescale of the reaction (with the differences between timescales of the different reactions absorbed into the exponentials), \( G_{ij}^c \) is the
chemical energy barrier between the states \( i \) and \( j \), \( \Delta G_{ij} \) is the energy difference (both chemical and mechanical) between the states as shown in Figure 4.1. The Boltzmann factor is denoted by \( \beta = \frac{1}{k_B T} \). Rates to go from a more energetic state \( i \) to the less energetic state \( j \) (usually forwards) are given by

\[
u_{ij} = \tau^{-1} e^{-\beta G_{ij}^z}.
\]

(4.1.2)

Typically, forwards movement through a cycle corresponds to transitions to lower energy states. If not at least one \( \Delta G_{ij} < 0 \).

![Figure 4.1: The energy landscape between states \( i \) and \( j \) with the energy barrier \( G_{ij}^z \) and energy difference \( \Delta G_{ij} \) labelled. \( x \) is some metric that measures the progress of the reaction.](image)

There were ten model parameters in the work by Skau et al. [44]. Four distinct changes in the chemical state of the myosin-V motor can occur during each hydrolysis cycle giving the first eight parameters. The release of a phosphate as the front head binds strongly to the filament has associated energy difference \( \Delta G_{Dw-Ds} \) with energy barrier \( G_{Dw-Ds}^z \), the reaction of an empty head with ATP from the bulk has energy difference \( \Delta G_{E-T} \) and barrier \( G_{E-T}^z \), the release of ADP from a head has energy difference \( \Delta G_{Ds-E} \) with barrier \( G_{Ds-E}^z \) and the weak binding of a head to the actin track with conversion of ATP to ADP has energy difference \( \Delta G_{T-Dw} \) and barrier \( G_{T-Dw}^z \).

The ninth parameter is related to intra-molecular strain. There exist three levels of molecular strain that contribute to the energy differences between states: unstrained, partly strained (energy \( bE_s \)) and fully strained (energy \( E_s \)). \( bE_s \) was determined under the assumption that the molecule is a Hookean spring, using \( E_s \) to give the spring constant and the step size (quoted from experimental results) to determine \( b \).

The final parameter is given by an additional energy barrier \( \alpha E_s \) relating to the energy required to open a pocket in a myosin-V head to allow a nucleotide to arrive or leave [10]. This is associated with the transition \( 4 \to 6 \) in the Skau model (see Figure 4.2) and is assumed to be proportional to the strain of the molecule and gives the lead head gating effect discussed in chapter 1.
CHAPTER 4. MYOSIN-V STEPPING MODELS

Figure 4.2: The discrete stochastic model developed by Skau et al. [44] to describe myosin-V stepping. States 1-7, the chemical configuration of each head and the dominant direction in which the molecule moves through state space are labelled. There are two cycles, one main hydrolysis cycle in which the molecule moves forwards and takes a step (passing through states 1, 2, 3, 4 and 5) and one futile cycle in which the molecule fails to take a step (passing through states 2, 3, 4 and 6). The molecule can disassociate from the track from state 7, a result of a loss of coordination between the heads. Each state has an associated amount of molecular strain: states 1, 4, 5 and 6 have the maximum amount of strain (the molecule is in the telemark stance), states 2 and 7 have no associated strain and state 3 has an intermediate amount of strain.

A choice of a set of these parameters can be interpreted geometrically as a point in a 10-dimensional parameter space. A model associated with that point has parameters given by the point coordinates and therefore produces certain results for observable quantities such as the motor velocity and run length. In order to compare these results against experiment numerically a function is defined that associates a real positive number with each point in parameter space based on how well the model reproduces experimental results at that point. This is known as the cost function.

The cost function was evaluated at 50,000,000 points in parameter space in this study; these points were chosen using a Sobol quasi-random number generator [35] to ensure they were evenly spread throughout the space. The 50 points with the lowest values of the cost function were then passed as initial points to a standard numerical simulated annealing Monte Carlo routine that explored the ten-dimensional energetic parameter space to minimise the cost function. The point corresponding to the lowest value of the cost function achieved in those 50 simulated annealings was then chosen to give the model parameters (Table 4.1).

This method allowed Skau et al. to fit their model parameters so that their model reproduced experimental results. In this chapter I propose a that similar method
4.2. THE METAMODEL OF MYOSIN-V

Table 4.1: The optimised energy barriers, differences and the strain and gating values chosen using the optimisation routine developed by Skau et al. [44].

<table>
<thead>
<tr>
<th>Energy Barriers</th>
<th>Strain</th>
<th>Gating</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G_{T-Dw}^t$</td>
<td>$G_{Dw-Ds}^t$</td>
<td>$G_{Ds-E}^t$</td>
</tr>
<tr>
<td>0.3</td>
<td>10.4</td>
<td>15.7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Energy Differences</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta G_{T-Dw}$</td>
<td>0.14</td>
</tr>
<tr>
<td>$\Delta G_{Dw-Ds}$</td>
<td>9.9</td>
</tr>
<tr>
<td>$\Delta G_{Ds-E}$</td>
<td>-10</td>
</tr>
<tr>
<td>$\Delta G_{E-T}$</td>
<td>13.1</td>
</tr>
<tr>
<td></td>
<td>71.4</td>
</tr>
</tbody>
</table>

can be used not just to fit an individual model but to compare different models against each other. I then formulate several proposed myosin-V mechanisms into a single framework to demonstrate this method.

4.2 The Metamodel of Myosin-V

Several stepping models for the myosin-V protein have been proposed as discussed in Chapter 1. A selection of these models are brought together into the same framework in this section. The metamodel is shown in Figure 4.3. Each individual model can be defined by taking the metamodel and only allowing certain state transitions. Here I extend the framework used by Skau et al. [44] to include the additional mechanisms proposed by Baker et al. [1] and Cappello et al. [5]. The key feature of this work is that this is all performed in a single mathematical framework in which I can choose the mechanisms that I want to include in a given model. The aim is to investigate how the underlying assumptions affect measurable results.

Myosin-V moves along its actin track by hydrolysing ATP. The hydrolysis cycle is shown in Figure 4.3 in black. Starting from the state in which both heads are attached to the actin track and are bound to ADP (state 4 in Figure 4.3) the molecule can release ADP from its rear head to pass into state 5 where the rear head is in the rigor state - where intra-molecular strain is at maximum. It detaches rapidly from the actin track after reacting with ATP from the bulk transitioning the molecule into state 1. Molecular strain is then released causing the motor to take its powerstroke step moving the ATP-attached head to the front and the molecule into state 2. A diffusive search for the actin filament followed by binding to the actin track and breakdown of the ATP in the front head into ADP and $P_i$ is the transition to state 3. Release of the phosphate from the front head completes the hydrolysis cycle, bringing the myosin back into state 4.

Failed stepping has been postulated to play a major role in the function of myosin-V [1, 44]. This futile cycle in our model corresponds to release of ADP from the front head in state 4 rather than the rear head. This is the transition from state 4 to 6. The front head then detaches from the track upon binding of ATP bringing it back into state 2. This cycle corresponds to the protein lifting its front head off
Figure 4.3: The myosin-V metamodel. The black arrows represent the basic hydrolysis cycle with strain-release-generated stepping. Additional pathways are colour coded: blue (A) represents the ideas from the work of Rosenfeld and Sweeney [42] - chemical detachment and the futile step - red (B) and light green (C) represent alternative hydrolysis cycle states, purple (D) shows possible mechanical detachments and yellow (E) is an additional pathway from the pre-detachment state back to the hydrolysis cycle. Dark green loops (F) correspond to transitions in which the molecule slips $d = 36$nm along the actin filament. The model proposed by Skau et al. [44] includes A. The model proposed by Baker et al. [1] and developed by Wu et al. [61] includes B, C and D.
4.2. THE METAMODEL OF MYOSIN-V

the actin and placing it down again in the same position. In this manner a protein can fail to move along its track as it cycles through states $2 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 2$.

A postulated method for the motor to detach from the track is through a loss of coordination of the chemical reactions in the myosin heads [44]. If the molecule transitions into a state in which only one head is attached to the track and this head is nucleotide free (transition between states 2 and 9 in the metamodel) then binding of $ATP$ to the attached head will cause the molecule to detach from the track completely (transition from state 9 to detached). Chemical detachment and futile cycling are represented in Figure 4.3 by the blue (A) transitions.

Additional cycles have been suggested to be possible [2, 61]; these consist of the mechanochemical transitions occurring in different orders. Release of $ADP$ from the rear head in state 3 transitions the molecule away from the main hydrolysis cycle into state 7. Release of $Pi$ from the front head then moves the molecule back to the main cycle and into the telemark position in state 5. This cycle is represented in Figure 4.3 by the red (B) transitions. For a molecule in state 7, binding of $ATP$ to the rear head before release of the phosphate from the front head causes a transition into state 8. Subsequent release of the phosphate then brings the molecule back to the main hydrolysis cycle in state 1. This is represented in Figure 4.3 by the light green (C) transitions. Including states 7 and 8 in a model gives another potential pathway through the reaction cycle. Binding of the front head to the actin from the pre-detachment state 9 brings the molecule back into the cycle at state 7. This is represented in Figure 4.3 by the yellow (E) transitions. Slipping along the track has been suggested by experimentalists [17] and was successfully incorporated into a model by Bierbaum and Lipowsky [2]. This is represented in the metamodel framework by dark green (F) loops corresponding to $d = 36nm$ slipping along the actin filament. Here we assume a slip of one helical repeat of the actin as experimental studies have yet to confirm the step size of slipping.

Detachment at a constant rate from particular states has been suggested as a possibility. In this metamodel this mechanical detachment is represented by constant transition rates leaving the system from states 3 and 8 as suggested by Baker et al. [1].

4.2.1 Master Equations

Each state in the model has an associated differential equation that describes how the probability of state occupancy evolves with time. The master equations for
CHAPTER 4. MYOSIN-V STEPPING MODELS

This system are:

\[ \dot{P}_1 = w_{21}P_2 + u_{51}P_5 + u_{81}P_8 - (u_{12} + w_{15} + w_{18})P_1, \]
\[ \dot{P}_2 = u_{12}P_1 + w_{32}P_3 + u_{62}P_6 + w_{92}P_9 - (w_{21} + u_{23} + w_{26} + w_{29})P_2, \]
\[ \dot{P}_3 = u_{23}P_2 + w_{43}P_4 + w_{73}P_7 - (w_{32} + u_{34} + u_{37} + \delta_3)P_3, \]
\[ \dot{P}_4 = u_{34}P_3 + w_{54}P_5 + w_{64}P_6 - (w_{43} + u_{45} + u_{46})P_4, \]
\[ \dot{P}_5 = w_{15}P_1 + u_{45}P_4 + u_{75}P_7 - (u_{51} + w_{54} + w_{57})P_5, \]
\[ \dot{P}_6 = w_{26}P_2 + u_{46}P_4 - (u_{62} + w_{64})P_6, \]
\[ \dot{P}_7 = u_{37}P_3 + w_{57}P_5 + w_{87}P_8 - (u_{75} + w_{78} + w_{79})P_7, \]
\[ \dot{P}_8 = w_{18}P_1 + u_{78}P_8 - (u_{81} + w_{87} + \delta_8)P_8, \]
\[ \dot{P}_9 = u_{29}P_2 + w_{79}P_7 - (w_{92} + u_{97} + \delta_9)P_9, \]

where \( u_{ij} \) and \( w_{ij} \) describe the reaction rates going from state \( i \) to state \( j \) forwards and backwards respectively and the number of states \( n \) is 9. The terms \( \delta_3, \delta_8 \) and \( \delta_9 \) are the rates at which molecules in state 3, 8 and 9 respectively fall off the track and are lost to the bulk. These detachment rates can be set to zero using the renormalisation methods described in section 2.4.1.

This system can be written in matrix notation as

\[ \dot{P} = M \dot{P} \] (4.2.1)

where \( M \) is a \( n \times n \) reaction rate matrix and the \( i^{th} \) component of vector \( P \) is \( P_i \). The detachments are renormalised away [27] and we drop the tilde.

The state-occupancy probabilities \( P_i \) and the dynamic properties \( (V \text{ and } D) \) are given by the methods described in chapter 3 and the run length \( L \) is given by equation 2.4.7. All that is required to determine these values are the transition rates.

### 4.2.2 Energetics

The reaction of two molecules \( X \) and \( Y \) to produce \( Z \) can be written

\[ X + Y \rightarrow Z. \] (4.2.2)

The reaction will generally occur in the forwards direction if

\[ G_Z + W - G_X - G_Y < 0, \] (4.2.3)

where \( G_i \) is the free energy of molecule \( i \) and \( W \) is the work done to the surroundings by the reaction.

It is a fundamental notion that total energy is conserved. Therefore the sum of all the energies in a closed system is always constant and so the total energy before and after a reaction is the same. This is energy balance.
To consider the energy balance of a myosin-V stepping cycle some notation must be introduced. I shall denote:

\[ G_0^b \]  Free energy of head bound to actin with no nucleotide attached,
\[ G^D_b \]  Free energy of head bound to actin with ADP attached,
\[ G_{DP}^b \]  Free energy of head bound to actin with ADP and Pi attached,
\[ G^T \]  Free energy of unbound head with ATP attached,
\[ G^T_f \]  Free energy of ATP in bulk,
\[ G^D_f \]  Free energy of ADP in bulk,
\[ G^P_f \]  Free energy of Pi in bulk,
\[ E_s \]  Energy stored in fully strained motor,
\[ bE_s \]  Energy stored a partially strained motor,
\[ W \]  Work done to the surroundings - the dissipation,
\[ m \]  the number of ATP molecules in the bulk.

The energies for each state in the metamodel can then be written relative to that of state 5 (the maximally strained state) as shown in Table 4.2. The free energy of a head with ATP attached is assumed not to change between the states whether the head is on or off the track; the differences between these states are measured in the difference in internal strain of the molecule (\( E_s \) for strong binding of the front head and \( bE_s \) for weak binding). It should also be noted that work done in the energy balance is only relevant to the forwards reactions; for backwards reactions this term should be omitted.
The energy balance can then be written as follows

\[ \Delta G_1,2 = W - E_s, \quad (4.2.4) \]
\[ \Delta G_2,3 = bE_s + G_b^{DP} - G^T, \quad (4.2.5) \]
\[ \Delta G_{3,4} = E_s(1 - b) + G_b^D + G_f^P - G_b^{DP}, \quad (4.2.6) \]
\[ \Delta G_{4,5} = G_b^0 + G_f^D - G_b^0, \quad (4.2.7) \]
\[ \Delta G_{5,1} = G^T - G_f^T - G_b^0, \quad (4.2.8) \]
\[ \Delta G_{4,6} = G_b^0 + G_f^D - G_b^0, \quad (4.2.9) \]
\[ \Delta G_{6,2} = G^T - G_b^0 - E_s - G_f^P - G_f^T, \quad (4.2.10) \]
\[ \Delta G_{3,7} = G_b^0 + G_f^D - G_b^0, \quad (4.2.11) \]
\[ \Delta G_{7,5} = E_s(1 - b) + G_b^D + G_f^P - G_b^{DP}, \quad (4.2.12) \]
\[ \Delta G_{7,8} = G^T - G_f^T - G_b^0, \quad (4.2.13) \]
\[ \Delta G_{8,1} = E_s(1 - b) + G_b^D + G_f^P - G_b^{DP}, \quad (4.2.14) \]
\[ \Delta G_{2,9} = G_b^0 + G_f^D - G_b^0, \quad (4.2.15) \]
\[ \Delta G_{9,7} = bE_s + G_b^{DP} - G^T. \quad (4.2.16) \]

where \( \Delta G_{i,j} \) denotes the difference in free energies between state \( i \) and state \( j \).

In order to optimise this model, the independent parameters must first be identified and the defined energy differences are not all independent of each other. If \( \Delta G_{X-Y} \) denotes the energy difference between a myosin-V head going from chemical state \( X \) to state \( Y \) then,

\[ \Delta G_{Dw-Ds} = G_b^D + G_f^P - G_b^{DP}, \]
\[ = \Delta G_{3,4} - E_s(1 - b), \]
\[ = \Delta G_{7,5} - E_s(1 - b), \]
\[ = \Delta G_{8,1} - E_s(1 - b), \quad (4.2.17) \]
\[ \Delta G_{Ds-E} = G_b^0 + G_f^D - G_b^0 \]
\[ = \Delta G_{4,5}, \]
\[ = \Delta G_{4,6}, \]
\[ = \Delta G_{3,7}, \]
\[ = \Delta G_{2,9}, \quad (4.2.18) \]
\[ \Delta G_{E-T} = G^T - G_f^T - G_b^0, \]
\[ = \Delta G_{5,1}, \]
\[ = \Delta G_{7,8}, \]
\[ = \Delta G_{6,2} + E_s, \quad (4.2.19) \]
\[ \Delta G_{T-Dw} = G_b^{DP} - G^T \]
\[ = \Delta G_{2,3} - bE_s, \]
\[ = \Delta G_{9,7} - bE_s. \quad (4.2.20) \]
4.2. THE METAMODEL OF MYOSIN-V

The total energy balance can then be expressed for each cycle in which an ATP hydrolysis reaction occurs, for example:

\[ \sum_{\text{hydrolysis}} \Delta G_{i,j} = \Delta G_{1,2} + \Delta G_{2,3} + \Delta G_{3,4} + \Delta G_{4,5} + \Delta G_{5,1} \]

\[ = G_f^P + G_f^D + W - G_f^T. \quad (4.2.21) \]

The total energy from hydrolysis cycle is assumed to be the same through each cycle. For this reaction network to exist physically it is required that the free energy of the one ATP molecule in the bulk is larger than the sum of the free energies of its products plus the work done for each run through the hydrolysis cycle \[ \sum_h \Delta G_{i,j} < 0. \]

Note that for the futile cycle an ATP nucleotide from the bulk has been converted into ADP and phosphate in the bulk but no work has been done to the surroundings:

\[ \sum_{\text{futile}} \Delta G_{i,j} = \Delta G_{2,3} + \Delta G_{3,4} + \Delta G_{4,6} + \Delta G_{6,2} \]

\[ = G_f^P + G_f^D - G_f^T. \quad (4.2.22) \]

The futile cycle is therefore energetically a more favourable path. For the hydrolysis cycle to dominate there must be an additional energy barrier that must be crossed to enter the futile cycle. It has been suggested by experimentalists that the energy barrier to release an ADP nucleotide from the front head is higher than that to release it from the rear head \[ [42]. \]

This is known as gating and is postulated to physically correspond to the energy required to open a pocket in which the nucleotides sit.

The molecular strain constant \( b \) relates the total molecular strain to the partial. In state 2 the molecule is unstrained, and \( bE_s \) is strain gained from a diffusive step into state 3. \( E_s \) is the total strain when there are two heads attached to the track with the molecule in the telemark stance and the release of this strain generates the powerstroke. Therefore we have \( 0 < b < 1 \). Furthermore, assuming the molecule behaves like a perfect spring it has a Hookean spring constant \( k_H \) defined by the relation between the strain energy difference before and after a movement and the distance traveled. Therefore

\[ bE_s = \frac{1}{2}k_Hd_D^2, \quad (4.2.23) \]

\[ E_s = \frac{1}{2}k_Hd_P^2, \quad (4.2.24) \]

where \( d_D \) and \( d_P \) are the distances covered by the diffusive step and the powerstroke respectively. Thus

\[ b = \frac{d_D^2}{d_P^2}. \quad (4.2.25) \]

The work done to the surroundings through mechanical movement is given by

\[ W = f_{ex}d_P, \quad (4.2.26) \]
where $f_{ex}$ is the magnitude of the external force.

The Skau model only takes into account two substeps: $d_P = 24\, \text{nm}$ and $d_D = 12\, \text{nm}$ as shown by some experimentalists \[52\]. Cappello et al. \[5\] suggest that there is a third substep, $d_B$, corresponding to the release of phosphate from the front head and strong binding to the track. The substeps were experimentally measured to be $d_P = 23\, \text{nm}$, $d_D = 8\, \text{nm}$ and $d_B = 5\, \text{nm}$. The metamodel only has the two substep configuration but can be extended to include three in order to investigate the effect of different numbers of substeps.

### 4.2.3 Transition Rates

The transition rates in this work are based closely on those developed by Skau et al. \[44\]. Forwards and backwards transitions from state $i$ to state $j$ are denoted by $u_{ij}$ and $w_{ij}$ respectively. Transition rates in the main hydrolysis cycle are given by

\[
\begin{align*}
    u_{12} &= \tau_P^{-1}, \quad (4.2.27) \\
    w_{21} &= \tau_P^{-1}e^{-(E_s-f_{ex}d_P+EH)/k_BT}, \quad (4.2.28) \\
    u_{23} &= \tau_P^{-1}e^{-(G_{T-Dw}^1-+\Delta G_{T-Dw})/k_BT}, \quad (4.2.29) \\
    w_{32} &= \tau_P^{-1}e^{-(G_{T-Dw}^1+\Delta G_{T-Dw})/k_BT}, \quad (4.2.30) \\
    u_{34} &= \tau^{-1}e^{-(G_{T-Dw}^1-+\Delta G_{D-Dw})/k_BT}e^{-(f_{ex}d_B)/k_BT}, \quad (4.2.31) \\
    w_{43} &= [\Pi]\tau^{-1}e^{-(G_{T-Dw}^1-+\Delta G_{D-Dw})/k_BT}\Phi(f_{ex}), \quad (4.2.32) \\
    u_{45} &= \tau^{-1}e^{-(G_{D-E}^1-)/k_BT}, \quad (4.2.33) \\
    w_{54} &= [\text{ADP}]\tau^{-1}e^{-(G_{D-E}^1-+\Delta G_{D-E})/k_BT}\Phi(f_{ex}), \quad (4.2.34) \\
    u_{51} &= [\text{ATP}]\tau^{-1}e^{-(G_{E-T}^1-)/k_BT}\Phi(f_{ex}), \quad (4.2.35) \\
    w_{15} &= \tau^{-1}e^{-(G_{E-T}^1-+\Delta G_{D-E})/k_BT}, \quad (4.2.36)
\end{align*}
\]

where $E_H = 0.5f_{ex}^2/k_B$, the energy corresponding to pulling on the cargo of the protein assuming that the molecule behaves as a Hookean spring. Here it is assumed that this only has an effect on the largest substep: the powerstroke. $\tau$ is the fundamental timescale of the reaction, $\tau_P$ is the hydrodynamic time scale related to movement over step length $d_P$ and $f_{ex}$ is the component of the pico-newton size external force parallel to the direction of motion of the motor. $[X]$ represents the concentration of nucleotide $X$ in the bulk assumed to be constant. If it is always more energetically favourable to move to the next state, the optimised values for the $\Delta G$ will be positive. $\Phi(f_{ex})$ is a parameterisation of the force dependence associated with nucleotide binding used in the study by Bierbaum et al. \[2\] where

$$
\Phi(f_{ex}) = \frac{1 + e^{-d_XF'}/k_BT}{1 + e^{d_X(f_{ex}-F')}/k_BT}.
$$

$F'$ represents the threshold value that the external force must exceed to have an effect and $\chi$ encodes the scale of that effect. In this work we choose the same values as in \[2\]: $\chi = 4$ and $F' = 1.6\text{pN}$. 


Skau et al. [44] calculate the fundamental timescales to be:

\[ \tau \approx 10^{-8} \text{s}, \quad (4.2.38) \]
\[ \tau_P \approx 10^{-5} \text{s}, \quad (4.2.39) \]

and any difference between \( \tau \) or \( \tau_P \) for different steps is absorbed into the optimised energy barriers.

The transition rates for the futile cycle are given by

\[ u_{46} = u_{45} e^{-\alpha E_s/k_B T}, \quad (4.2.40) \]
\[ w_{64} = w_{54} e^{-\alpha E_s/k_B T}, \quad (4.2.41) \]
\[ u_{62} = u_{51}, \quad (4.2.42) \]
\[ w_{26} = w_{15} e^{-(E_s+fexdD)/k_B T}. \quad (4.2.43) \]

Here \( \alpha E_s \) is the energy barrier for opening a pocket in which the ADP nucleotide sits in order to let it in and out. This gating was suggested by Rosenfeld and Sweeney [42] to explain the difference in ADP release rates between the front and rear heads. It is assumed that we have symmetric gating - that the energy required to open the pocket is the same whether a nucleotide sits within it or not.

The additional rates for extra cycles can be related to the rates on the main hydrolysis cycle and so are given by

\[ u_{37} = u_{45} e^{-\beta E_s/k_B T}, \quad (4.2.44) \]
\[ w_{73} = w_{54} e^{-\beta E_s/k_B T}, \quad (4.2.45) \]
\[ u_{78} = u_{51}, \quad (4.2.46) \]
\[ w_{87} = w_{15}, \quad (4.2.47) \]
\[ u_{81} = u_{34}, \quad (4.2.48) \]
\[ w_{18} = w_{43}, \quad (4.2.49) \]
\[ u_{75} = u_{34}, \quad (4.2.50) \]
\[ w_{57} = w_{43}, \quad (4.2.51) \]
\[ u_{97} = w_{23}, \quad (4.2.52) \]
\[ w_{79} = w_{32}. \quad (4.2.53) \]

The \( \beta E_s \) represents the energy barrier for opening the pocket within which the ADP nucleotides sit for state 3.

The chemical detachment mechanism rates are defined to be

\[ u_{29} = u_{45}, \quad (4.2.54) \]
\[ w_{92} = w_{54}, \quad (4.2.55) \]
\[ \delta_9 = u_{51}. \quad (4.2.56) \]

An additional pathway to reproduce high forcing data - a jump from one cycle repeat to another - is adapted from Bierbaum et al. [2]

\[ w_{\text{slip}} = \frac{D'(fexd-U^2)}{d^2k_B T}(1-e^{(U^2-fexd)/k_B T})^{-1}, \quad (4.2.57) \]
\[ u_{\text{slip}} = w_{\text{slip}} e^{-fexd/k_B T}. \quad (4.2.58) \]
where \( D' \) is the diffusion constant, \( U_\uparrow \) is the corresponding energy barrier. The authors in [2] chose \( D' = 470 \text{nm}^2/\text{s} \) and \( U_\uparrow = 20k_BT \) in their study. Physically these transitions correspond to a postulated forwards and backwards slipping respectively from one binding state to the next of the myosin-V molecule [17]. It is assumed that this can only happen from states in which only one head of the molecule is bound to the track (states 2 and 9). Therefore,

\[
\begin{align*}
w_{22} &= w_{99} = w_{\text{slip}}, \\
u_{22} &= u_{99} = u_{\text{slip}}.
\end{align*}
\]

These rates have no effect on the governing state-space master equations as all relating terms cancel, however they do have an influence on the velocity as each molecule to undergo such a transition slips 36\text{nm} along the actin filament.

Mechanical detachment is determined by optimised parameters \( \delta_3 \) and \( \delta_8 \), the detachment rates from state 3 and 8 respectively.

A more sophisticated gating mechanism takes into account the different amounts of energy required to open the pocket dependent on whether a nucleotide sits within it and in which state the motor resides. This can be defined by modifying existing rates:

\[
\begin{align*}
w_{64}^* &= w_{54}e^{-\alpha^* E_s/k_BT}, \\
w_{37}^* &= w_{45}e^{-\beta E_s/k_BT}, \\
w_{73}^* &= w_{54}e^{-\beta^* E_s/k_BT}, \\
w_{29}^* &= w_{45}e^{-\gamma E_s/k_BT}, \\
w_{92}^* &= w_{54}e^{-\gamma^* E_s/k_BT}, \\
w_{54}^* &= w_{54}e^{-\omega^* E_s/k_BT}.
\end{align*}
\]

therefore, the gating of ADP need no longer be symmetric. Note that letting the ADP nucleotide in and out in all four sets of transitions \( 3 \leftrightarrow 7, 4 \leftrightarrow 6, 2 \leftrightarrow 9 \) and \( 4 \leftrightarrow 5 \) involves opening a pocket, but in the last case we absorb the forward barrier into \( G_{Ds-E} \) as we are interested in the difference between these energy barriers.

The velocity of molecules in the metamodel is given by the relations derived in equation 3.4.8:

\[
V = d [u_{12}P_1 - w21P_2 + (u_{\text{slip}} - w_{\text{slip}})(P_9 + P_2)].
\]

The runlength is given by equation 2.4.7 assuming renormalisation holds

\[
L = \frac{V}{\lambda}
\]

where again \( \lambda \) is the dominant eigenvalue of the reaction rate matrix \( \mathbb{M} \).
4.3. The Cost Function

The cost function contains 16 terms

\[
\Delta = \sum_{i=1}^{16} \Delta_i([ATP], [ADP], [P_i], f_{ex}),
\]

and each compares a result from the model against experimental data using a least-squares method

\[
\Delta_i([ATP], [ADP], [P_i], f_{ex}) = \frac{(R - R_E)^2}{\sigma_{R_E}^2},
\]

where \(R\) is the model result, \(R_E\) is the experimental result and \(\sigma_{R_E}^2\) is the mean-squared uncertainty in the experimental result; each is dependent on conditions \([ATP], [ADP], [P_i]\) and \(f_{ex}\).

All cost function terms pertaining to dynamic quantities are contained within Table 4.3. Experimentalists have measured dynamic quantities of myosin-V under varying nucleotide concentrations [1, 10, 11, 16, 41, 52, 59, 64] and terms 1-9 in the cost function represent these measurements. The next two terms \(\Delta_{10}\) and \(\Delta_{11}\) are based on the velocity and the run length measurements of myosin-V molecules under external forcing by Uemura et al. [52].

Rosenfeld and Sweeny [42] observe that release of ADP from the front head is 50 times slower than from the rear head and so

\[
\Delta_{12}(1mM, 0.1\mu M, 0.1\mu M, 0) = \frac{1}{50^2} \left( \frac{(J_{fusile})}{V} \right)^2.
\]

The last four terms of the cost function represent energetic restrictions on the

<table>
<thead>
<tr>
<th>(\Delta_i([ATP], [ADP], [P_i], f_{ex}))</th>
<th>(R)</th>
<th>(R_E)</th>
<th>(\sigma_{R_E})</th>
<th>Studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\Delta_1(1mM, 0.1\mu M, 0.1\mu M, 0))</td>
<td>(L)</td>
<td>0.8(\mu m)</td>
<td>0.15</td>
<td>[1]</td>
</tr>
<tr>
<td>(\Delta_2(1mM, 0.1\mu M, 0.1\mu M, 0))</td>
<td>(V)</td>
<td>0.54(\mu m s^{-1})</td>
<td>0.054</td>
<td>[1, 10, 11, 41, 52]</td>
</tr>
<tr>
<td>(\Delta_3(100\mu M, 0.1\mu M, 0.1\mu M, 0))</td>
<td>(L)</td>
<td>1.15(\mu m)</td>
<td>0.15</td>
<td>[1]</td>
</tr>
<tr>
<td>(\Delta_4(10\mu M, 0.1\mu M, 0.1\mu M, 0))</td>
<td>(V)</td>
<td>0.075(\mu m s^{-1})</td>
<td>0.01</td>
<td>[1]</td>
</tr>
<tr>
<td>(\Delta_5(1mM, 200\mu M, 0.1\mu M, 0))</td>
<td>(V)</td>
<td>0.32(\mu m s^{-1})</td>
<td>0.032</td>
<td>[1]</td>
</tr>
<tr>
<td>(\Delta_6(1mM, 2.5mM, 0.1\mu M, 0))</td>
<td>(L)</td>
<td>0.4(\mu m)</td>
<td>0.15</td>
<td>[1]</td>
</tr>
<tr>
<td>(\Delta_7(1mM, 2.5mM, 0.1\mu M, 0))</td>
<td>(V)</td>
<td>0.13(\mu m s^{-1})</td>
<td>0.013</td>
<td>[1]</td>
</tr>
<tr>
<td>(\Delta_8(1mM, 0.1\mu M, 4mM, 0))</td>
<td>(L)</td>
<td>0.5(\mu m)</td>
<td>0.15</td>
<td>[1, 36]</td>
</tr>
<tr>
<td>(\Delta_9(1mM, 0.1\mu M, 4mM, 0))</td>
<td>(V)</td>
<td>0.44(\mu m s^{-1})</td>
<td>0.044</td>
<td>[1, 36]</td>
</tr>
<tr>
<td>(\Delta_{10}(1mM, 200\mu M, 0.1\mu M, 0.75pN))</td>
<td>(L)</td>
<td>0.4(\mu m)</td>
<td>0.15</td>
<td>[52]</td>
</tr>
<tr>
<td>(\Delta_{11}(1mM, 200\mu M, 0.1\mu M, 0.75pN))</td>
<td>(V)</td>
<td>0.32(\mu m s^{-1})</td>
<td>0.05</td>
<td>[52]</td>
</tr>
<tr>
<td>(\Delta_{12}(1mM, 0.1\mu M, 0.1\mu M, 0))</td>
<td>(V)</td>
<td>0.32(\mu m s^{-1})</td>
<td>0.05</td>
<td>[52]</td>
</tr>
</tbody>
</table>

Table 4.3: The experimentally measured data points for the run length and molecular velocity included in the cost function
interaction of the protein with the actin track [19]:

\[
\Delta_{13} = \left( \frac{\Delta G_{T-Dw} - 2k_BT}{3k_BT} \right)^2,
\]

\[
\Delta_{14} = \left( \frac{\Delta G_{Dw-Ds} - 5.7k_BT}{3k_BT} \right)^2,
\]

\[
\Delta_{15} = \left( \frac{\Delta G_{Ds-E} + 7.7k_BT}{3k_BT} \right)^2,
\]

\[
\Delta_{16} = \left( \frac{\Delta G_{E-T} - 15.3k_BT}{3k_BT} \right)^2.
\]

The reaction energy differences are restricted so that they sum to the total energy given by the ATP hydrolysis reaction [19]:

\[
\Delta G_{T-Dw} + \Delta G_{Dw-Ds} + \Delta G_{Ds-E} + \Delta G_{E-T} = 13.1k_BT.
\]

This cost function is identical to that used by Skau et al. [44] with the exception that one term - the dwell time result - has been removed. The dwell time calculation has since been shown to be inaccurate [51] for models with backwards transitions.

### 4.4 A Modified Optimisation

In addition to the free parameters in the Skau model ($G^T_{T-Dw}$, $G^T_{Dw-Ds}$, $G^T_{Ds-E}$, $G^T_{E-T}$, $E_s$, $\alpha E_s$, $\Delta G_{T-Dw}$, $\Delta G_{Dw-Ds}$, $\Delta G_{Ds-E}$, $\Delta G_{E-T}$), the metamodel of myosin-V has mechanical detachment parameters $\delta_3$ and $\delta_8$, the additional gating parameter $\beta E_s$, and $\alpha^* E_s$, $\beta^* E_s$, $\gamma E_s$, $\gamma^* E_s$ and $\omega^* E_s$ if the symmetric gating of ADP release assumption is dropped. These all need to be determined by the optimisation.

The cost function $\Delta$ defines a hypersurface in parameter hypspace, within our metamodel framework this is at most a 18 dimensional surface in 19 dimensional parameter-cost function space. The closer $\Delta$ is to zero for any given point, the better the model reproduces experimental data for those parameters. A comparison of these values for different models within our metamodel allows one to compare quantitatively the degree to which each model reproduces experimental results. Moreover, local minima of $\Delta$ or local regions of a relatively flat cost function can be identified across models allowing a measure of inter-model compatibility to be introduced.

In this study I am not only interested in the global point of lowest cost. The cost function is somewhat arbitrary as the experimental results that are included are chosen by the modeller and there is some debate as to what data is the best (see Chapter 1). A small modification of the cost function could make a local minimum - or a point nearby - a global minimum. Therefore as we are more interested in the experimentally measurable behaviour of the model, the point of lowest cost is not
4.4. A MODIFIED OPTIMISATION

necessarily the one we want to investigate: we are interested in points of relatively low cost around which small changes in the cost have little effect on the model results. Assuming that we have chosen a reasonably valid cost function we want to understand the shape of its hypersurface close to minimum points. From this it must be decided which of the local minimum points should be used. A spiky landscape around a minimum is a sign that the model is not very robust there and therefore less likely to be valid [38]. Within a small enough region around a low-cost point the model should behave consistently and the relative sizes of these regions indicate the robustness of the model at each point.

A numerically expensive optimisation - such as the one used by Skau et al. [44] - identifies minima well within a very spiky landscape. This is not necessary as spiky regions are not of interest; a far cheaper optimisation that gives reasonably low-cost points (not necessarily minima), allows the behaviour of the model in low-cost regions to be investigated quickly and efficiently. A region can be said to be low cost a-priori if the cost function takes a value similar to or less than the original Skau lowest-cost point in the metamodel optimisation.

In a given optimisation run firstly an initial point is determined. Then this is passed to the Monte Carlo simulated annealing routine that explores parameter space from the initial values to a point of relative low cost. A sensitivity analysis of the parameters can then be performed to describe the shape of the hypersurface in this region.

Exploring parameter space extensively using the Sobol quasi-random sequence to find starting points for the simulated annealing routine is numerically expensive in the metamodel framework. To improve computational efficiency the start point chosen for the routine is estimated sensibly. In their numerically intense optimisation scheme Skau et al. [44] established an interesting parameter region that agreed with most experimental data. In the modified optimisation for the metamodel, these are the initial values for the corresponding parameters. The initial values of the additional parameters in the metamodel are chosen using additional data. Baker et al. [1] estimate

\[ \delta_3 = 1.1 \text{ms}^{-1}, \]
\[ \delta_8 = 0.032 \text{ms}^{-1} \]

and the values within the model formulated by Skau et al. [44] give

\[ \beta E_s = 0, \]
\[ \alpha^* E_s = \alpha E_s, \]
\[ \beta^* E_s = 0, \]
\[ \gamma E_s = 0, \]
\[ \gamma^* E_s = 0, \]
\[ \omega^* E_s = 0. \]

These are then passed to a bespoke Monte Carlo simulated annealing routine presented in appendix A. Whilst there are possibly other low-cost regions for models
within the metamodel framework it is impractical computationally to search for them. For each model considered, in addition to the Skau initial point, 10,000 random start points were also selected and optimised as a crude check for other low cost regions for each model. Each run with a low cost result brought the parameters back towards the region in which the Skau parameters are location implying that using these as the initial point is a sensible strategy.

Once a low cost point $h$ for a particular model has been identified, an analysis of the surrounding $n$-dimensional hypersurface must be performed. If the value of the cost function at that point is $\Delta(h)$ then we want to establish the set of largest $\delta h_i^+$ and $\delta h_i^-$ such that

$$\Delta(h + \delta h_i^+ \hat{n}_i) < \Delta(h) + \epsilon \quad \text{and} \quad \Delta(h - \delta h_i^- \hat{n}_i) < \Delta(h) + \epsilon,$$

(4.4.9)

(4.4.10)

for a given $i \in \{1, 2, ..., n\}$ where $\epsilon$ is a constant relating to cost function variation and $\hat{n}_i$ is the unit vector in the $i$th direction.

Defining the variability of parameter $i$ by $\Delta H_i = \delta h_i^+ + \delta h_i^-$, parameters with large $\Delta H_i$ are more stable optimisationally than parameters with a small $\Delta H_i$. Therefore, models that have larger $\Delta H_i$ are more robust than models with smaller $\Delta H_i$. We are only interested in the relative variability between parameters and models; thus $\epsilon$ can be chosen somewhat arbitrarily, so long as it is small and remains constant across the parameters and models investigated.

### 4.5 Optimisation Validation

There are very few components ($\Delta_i$) in the cost function defined for the metamodel optimisation: essentially only a small number of available experimental data points are chosen to be optimised against. This is partly to reduce computational expense but also to reduce the likelihood of what is known to be a major pitfall of numerical optimisation: over constraint and over fitting [38].

An overly constrained optimisation is one in which either the objective function (in this case our cost function) contains too many data points or the model itself has too many restrictions and is not relevant to the objective function. Both prevent the optimisation from picking sensible parameter values [38].

An optimisation result that has been over fitted is one in which the optimisation routine has picked specific parameter values that have an extremised objective function value but the parameter values themselves make little sense. This usually occurs if the objective function is poorly defined or if the space is particularly spiky and has the symptom that the objective function is highly sensitive to a small change in a particular parameter. One symptom of this is that results change dramatically for small changes in the model parameters or constants.

In order to combat these issues several steps are taken. Firstly very few experimental data points are defined in the cost function so that the optimisation routine...
simply chooses the scale of the parameters rather than the experimentally inferred
trends and relationships. A low-cost selection of results are examined then manually
after the optimisation run (see the following chapter) and a sensitivity analysis
on the parameters is performed after the optimisation as defined in the previous
section.

Certain parameters pertain to some experimental kinetic measurements not in-
cluded in the optimisation. ATP binding to the rear head has been found to exist
within a range [55]. In our model this corresponds to

\[
\frac{u_{51}}{[ATP]} \in [0.6, 1.5] \mu M^{-1} s^{-1}.
\]  

Thus it is required that

\[
G_{E-T}^i \in [18.0, 18.9] k_B T,
\]  

to match experimental results. Phosphate release has been shown to be very high
with \(u_{34} > 250 s^{-1}\) [11]. This corresponds to \(G_{Dw-Ds}^i < 12.9 k_B T\). ADP release
from the rear head has been measured experimentally to have a maximum values
of about 30 s\(^{-1}\) [42]. Unfortunately in the metamodel framework there are several
possible values for the ADP release rate and each is dependent on the internal
molecular strain. Assuming that the ADP-release transition occurs from the max-
imally strained state, this corresponds to

\[
u_{45} \in [10, 20] \mu s^{-1},
\]  

and so

\[
G_{D-T}^i \in [15.4, 16.1] k_B T.
\]

Many studies compare the magnitude of the transitions rates against kinetic stud-
ies [2, 28, 44, 61]. Here the energetics of the system is optimised rather than the
rates themselves; a small change energetically will make a large difference to the
transition rates due to the exponential terms. Additionally, all transitions are as-
sumed reversible meaning that in the transition between states a high forwards
and backwards rate can correspond to a much lower experimentally measured rate
in only one direction due to the fact that kinetic studies measure the statistically
averaged drift from one state to another. Therefore rates determined in the meta-
model optimisation framework are not necessarily expected to match experimental
data - instead the overall trend of dynamic properties against cellular conditions
([ATP], [ADP], \(f_{ex}\) etc.) is sought after given physical the energetic constrictions
and model structure. The system is ensured to be realistic in careful choosing of
the states, transitions and their corresponding energetic restrictions.

### 4.6 Summary

There are many experimental ideas as to how myosin-V functions and thus many
models to describe such function. Models constructed in different mathematical
frameworks can show different results but it is often difficult to determine whether this is a result of the underlying assumptions and model architecture or the biological mechanisms.

In this chapter a metamodel for myosin-V is presented that takes into account many proposed mechanisms all under the same framework. These can be activated or deactivated to create models that include various different biological mechanisms. Several different reaction pathways have been included in the metamodel as well as multiple routes for the molecular motor to detach from its track. An optimisation scheme has been proposed to establish a particular model’s compatibility with various experimental results.

The framework presented in this chapter will be used in the rest of this thesis to investigate multiple stepping mechanisms of myosin-V.
In the previous chapter a mathematical framework was presented under which to construct multiple models of myosin-V in order to compare them against each other. Here, this is applied to isolate and identify individual characteristics of a model and improve agreement with experimental data: in particular, the ability of a given model to reproduce experimentally observed velocity and run length against \([ADP]\) and \([ATP]\) relationships. Note that in this chapter the number of substeps that myosin-V takes in its 36nm step is fixed to two \([52]\).

Optimisation procedures are applied to modified versions of the models proposed by Skau et al. \([44]\) and Wu et al. \([61]\). A discrepancy in the reproduction of experimentally observed run length against nucleotide concentration results is observed and modifications of the models are postulated to resolve this conflict.

The aim of this chapter is to identify why different models produce conflicting results and to construct a model that reproduces as many of the experimentally observed results as possible. Note that the work in this chapter is done under zero forcing and so the corresponding cost function values \(\Delta_{10}\) and \(\Delta_{11}\) (defined in the previous chapter) are set to zero.

## 5.1 A Comparison of Two Stepping Models

While it is well established that coordination of the chemical states of myosin-V heads leads to processive motion the circumstances under which the walk breaks down is not understood. Furthermore, there are many sets of experimental data and no model can currently reproduce the trends inferred from all of them. In order
to identify which model can reproduce which results it is necessary to analyse them under the same mathematical framework: the metamodel framework presented in the previous chapter. Once this is done it is possible to establish what it is within these descriptions that leads to agreement or disagreement with the experimental results.

It is important to note that each experimental result is not equally balanced in the cost function due to the nature of the experimental data; there are more values for the velocities and the $L$ against $[ATP]$ relationship than for the $L$ against $[ADP]$ one. Thus the $L$ against $[ADP]$ relationship will be the most difficult to reproduce and this must be taken into account in the analysis of models in the metamodel framework.

In this section the model proposed by Skau et al. [44] and the model proposed by Baker et al. [1] but developed by Wu et al. [61] are compared against each other. In order to minimise conflicting modelling assumptions between them this comparison is repeated when they are both encoded within the metamodel framework. The aim is to identify the similarities and differences between the models in order to establish their relative validity and therefore improve the understanding of myosin-V.

### 5.1.1 The Skau Model Versus The Wu Model

The model of myosin-V developed by Skau et al. [44] (the Skau Model) contains one hydrolysis cycle through which the protein steps along the actin filament, a futile stepping cycle and detachment from the actin track that corresponds to a loss of coordination between the heads. This matches the metamodel described in the previous chapter with pathway $A$ included as shown in Figure 5.1. An important aspect of this model is that it was originally constructed under an optimisation framework similar to that described within the previous chapter but restricted to the relevant reaction pathways. As previously discussed, the cost function used in that optimisation was not correct (see section 4.3 for more details) and so this model will be reoptimised within this new framework.

The Wu Model was developed by Wu et al. [61] and contains three possible hydrolysis cycles through which the myosin-V protein could step along the actin track (pathways B and C in the metamodel framework). This model also contains an alternative motor detachment mechanism: molecules leave the track at a constant rate from states in which the front head has an ADP nucleotide bound to it (pathway $D$).

The original discrete-stochastic model proposed by Wu et al. [61] was very simple. The transition rates of the governing master equations were simply taken directly from experiment or chosen to allow the model to function as required. This shall be referred to as the Original Wu Model.

The optimisation results for the Skau Model in the metamodel framework are shown in Table 5.1. This particular submodel has the fewest parameters of all
5.1. A COMPARISON OF TWO STEPPING MODELS

Figure 5.1: The discrete stochastic model developed by Skau et al. [44] to describe myosin-V stepping. States 1-6 and 9, the chemical configuration of each head and the dominant direction in which the molecule moves through state space are labelled. There are two cycles, one main hydrolysis cycle in which the molecule moves forwards and takes a step (passing through states 1, 2, 3, 4 and 5) and one futile cycle in which the molecule fails to take a step (passing through states 2, 3, 4 and 6). The molecule can disassociate from the track from state 9, a result of a loss of coordination between the heads. Each state has an associated amount of molecular strain: states 1, 4, 5 and 6 have the maximum amount of strain (the molecule is in the telemark stance), states 2 and 9 have no associated strain and state 3 has an intermediate amount of strain. This corresponds to our metamodel of myosin-V shown in Figure 4.3 with only cycle A (blue) included.

those contained within the metamodel - save that which only includes the main hydrolysis cycle - as it only includes pathway A. The optimised parameters are very similar to those obtained by Skau et al. [44] with the exception that the strain-related parameters have been reduced. This suggests that the original inaccurate optimisation (due to the dwell time cost function result since shown to be incorrect [50]) performed by the authors had a significant effect on the intra-molecular strain results obtained and thus a given optimisation’s observable results can be very sensitive to its objective function. The cost function must therefore be chosen carefully to ensure accurate results.

The velocity and run length against nucleotide concentration relationships for the re-optimised metamodel version of the Skau Model and the Original Wu Model are shown in Figures 5.2 and 5.3 respectively. The experimental velocity against [ADP] and [ATP] results are both reproduced well, although the Original Wu Model is a better fit to the former and the Skau Model is a better fit to the latter. The run length relationships however are far more interesting. The reoptimised
Table 5.1: Optimised parameters and final cost function values for models of myosin-V included in this chapter. The Original Wu model is omitted as the authors did not use a compatible optimisation framework. Parameters relating to the energy barriers and differences between states are shown as well as intramolecular strain values and optimised detachment rates.
5.1. A COMPARISON OF TWO STEPPING MODELS

RESULT REPRODUCTION

<table>
<thead>
<tr>
<th>Model</th>
<th>Cost</th>
<th>$[ATP]$</th>
<th>$[ADP]$</th>
<th>$[ATP]$</th>
<th>$[ADP]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Wu</td>
<td>206.77</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Original Skau</td>
<td>214.94</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Wu (BCD)</td>
<td>4.29</td>
<td>yes</td>
<td>yes</td>
<td>limited</td>
<td>yes</td>
</tr>
<tr>
<td>Skau (A)</td>
<td>59.42</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>SAGM (A*)</td>
<td>5.10</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>SAGEM (ABE*)</td>
<td>1.29</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table 5.2: A list of models investigated including their cost function value (the cost) and whether they reproduce certain experimental trends. All models are in the metamodel framework except the Original Wu and the Original Skau models that produce the conflicting run length against nucleotide concentration results. For these stepping mechanisms in the metamodel framework (the Wu Model and the Skau model respectively) this holds, although the Wu Model does give the $L$ vs $[ATP]$ experimental trend for a small region of $[ADP]$ concentrations. With regards to run length the Skau Model only gives the $L$ vs $[ATP]$ result; introducing asymmetric gating in the SAGM stops this but allows it to give the $L$ vs $[ADP]$ result. Introducing an extra pathway through which myosin-V molecules can move in the SAGEM allows all considered results to be reproduced.

Skau Model still does not reproduce the $[ADP]$ relationship but does reproduce the $[ATP]$ relationship similarly to the original as previously discussed [44]). As $[ADP]$ increases, run length increases proportionally rather then asymptoting at saturating concentration. Conversely the Original Wu Model reproduces the $[ADP]$ experimentally observed trend and not the $[ATP]$ one: as $[ATP]$ tends to zero the run length drops rather than increasing as experimentally observed. These points, and details of the further models investigated later in this chapter, are summarised in Table 5.2.
Figure 5.2: A comparison of the velocity relationships against ATP and ADP nucleotide concentration between the model proposed by Skau et al. [44] in the metamodel framework (blue) and the original model by Wu et al. [61] (red). Model results are at three nucleotide concentrations: 1 mM (solid curves) and 10 µM (dashed) for results against [ADP] and [ATP], and [ATP] = 500 µM and [ADP] = 100 µM (dotted) for results against [ADP] and [ATP] respectively. Experimental data points demonstrating relationships (circles) were measured by Baker et al. [1] against [ADP] (performed at 1 mM [ATP]) and by Forkey et al. [16] against [ATP] (at 10 µM [ADP]). Experimental results used in the optimisation (only the Skau Model in this case) are marked by all other symbols. Data points are at [ATP] = 1 mM (pluses: Δ₂, Δ₅ and Δ₇), [ATP] = 10 µM (right-pointing triangle: Δ₄), [ADP] = 2.5 mM (asterisk: Δ₇), [ADP] = 200 µM (left-pointing triangle: Δ₅), [ADP] = 0.1 µM (crosses: Δ₃ and Δ₄). (a): The solid line model results correspond to the experimentally observed circles but only optimised to match up with the pluses. The dashed line is optimised to match up with the right-pointing triangle. (b): The dashed curves correspond to the circles and are optimised to match up with the crosses. High [ADP] corresponds to the asterisk and the solid line.
5.1. A COMPARISON OF TWO STEPPING MODELS

(a) Run length against ADP concentration
(b) Run length against ATP concentration

Figure 5.3: A comparison of the run length relationships against ATP and ADP nucleotide concentration between the model proposed by Skau et al. [44] in the metamodel framework (blue) and the original model by Wu et al. [61] (red). Model results are as in Figure 5.2. Experimental data points demonstrating relationships (circles) were measured by Baker et al. [1]. Measurements against varying [ADP] are performed at 1 mM [ATP], and those against [ATP] are at 10μM [ADP]. Experimental results used in the optimisations are marked by all other symbols. Data points are at [ATP] = 1mM (pluses: ∆1 and ∆6), [ADP] = 2.5mM (asterisk: ∆6), [ADP] = 0.1μM (crosses: ∆1 and ∆3). (a): The solid line model results correspond to the experimentally observed circles but only optimised to match up with the pluses. The dashed line is optimised to match up with the right-pointing triangle. (b): The dashed curve correspond to the circles and are optimised to match up with the crosses. High [ADP] corresponds to the asterisk and the solid line.
In our metamodel framework the new Wu Model (shown in Figure 5.4) is defined in an energetic manner and transition rates are derived from optimised energetic parameters and includes defined pathways B, C and D. A comparison between the results of this model and the Skau Model is more valid as both models are in the same theoretical framework; the difference in modelling frameworks has been removed and so the various proposed mechanisms can be investigated more easily.

Figure 5.4: The discrete stochastic model developed by Wu et al. [61]. There are three pathways a molecule move along its actin track. Molecules can disassociate from the track mechanically from states 3 and 8. Each state has an associated amount of strain similar to that in the Skau model. This corresponds to our metamodel of myosin-V shown in Figure 4.3 with pathways B (red), C (green) and D (purple) included.

The optimised parameters for the Wu Model are shown in Table 5.1. The Wu Model has a much smaller cost (the final cost function value) than the Skau Model immediately implying that the results are much closer to the experimental data and that the data points it misses are less prominent. However, the greater number of optimised parameters could be the predominant cause of this. Interestingly most of the parameters and their variability are very similar between the two models with the exception of $G_{T-Dw}^+$, $G_{Dw-Ds}^+$ and $E_s$, which are much greater, and $G_{Ds-E}^+$ which is much smaller. The variability measure $\Delta H$ (see Table: 5.3) of each these parameters in the Wu Model is much smaller than in the Skau Model. This suggests that the low-cost region in which the extremised parameter point lies is more restricted and the likelihood of the Wu Model result being over fitted is greater (see discussion in section 4.5).
5.1. A COMPARISON OF TWO STEPPING MODELS

### Table 5.3

<table>
<thead>
<tr>
<th>VARIABILITY</th>
<th>Energy Differences $\Delta G (k_B T) \quad \Delta H_{T-Dw} \quad \Delta H_{Dw-Ds} \quad \Delta H_{Ds-E} \quad \Delta H_{E-T}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Skau (A)</td>
</tr>
<tr>
<td></td>
<td>Wu (BCD)</td>
</tr>
<tr>
<td></td>
<td>SAGM (A*)</td>
</tr>
<tr>
<td></td>
<td>SAGEM (ABE*)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VARIABILITY</th>
<th>Energy Barriers $G^I (k_B T) \quad \Delta H_{T-Dw}^I \quad \Delta H_{Dw-Ds}^I \quad \Delta H_{Ds-E}^I \quad \Delta H_{E-T}^I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Skau (A)</td>
</tr>
<tr>
<td></td>
<td>Wu (BCD)</td>
</tr>
<tr>
<td></td>
<td>SAGM (A*)</td>
</tr>
<tr>
<td></td>
<td>SAGEM (ABE*)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VARIABILITY</th>
<th>Detachment ($s^{-1}$) \quad Strain energies and barriers ($k_B T$) $\Delta H_{3s} \quad \Delta H_{8s} \quad \Delta H_{E_s} \quad \Delta H_{E_{s_a}} \quad \Delta H_{E_{s_s}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Skau (A)</td>
</tr>
<tr>
<td></td>
<td>Wu (BCD)</td>
</tr>
<tr>
<td></td>
<td>SAGM (A*)</td>
</tr>
<tr>
<td></td>
<td>SAGEM (ABE*)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VARIABILITY</th>
<th>Strain energies and barriers ($k_B T$) $\Delta H_{\alpha_s E_s} \quad \Delta H_{\beta_s E_s} \quad \Delta H_{\gamma_s E_s} \quad \Delta H_{\epsilon_s E_s} \quad \Delta H_{\omega_s E_s}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Skau (A)</td>
</tr>
<tr>
<td></td>
<td>Wu (BCD)</td>
</tr>
<tr>
<td></td>
<td>SAGM (A*)</td>
</tr>
<tr>
<td></td>
<td>SAGEM (ABE*)</td>
</tr>
</tbody>
</table>

Table 5.3: The variability $\Delta H_i$ with $\epsilon = 1$ of the final optimised parameter values for models of myosin-V optimised in the metamodel framework. In the region of the optimised parameter values, the larger the number, the flatter the cost function surface is in that particular direction in parameter space.

The velocity and run length against nucleotide concentration relationships for the reoptimised metamodel version of the Skau Model and the Wu Model are shown in Figures 5.5 and 5.6 respectively. The velocity against $[ADP]$ and $[ATP]$ results for the Wu Model are almost identical to the Original Wu Model (see figures 5.2 and 5.3) but the run length results are different. The run length against $[ATP]$ relationship is reproduced for the Wu Model under exactly 10 $\mu M [ADP]$ but the established behaviour deteriorates rapidly as the ADP concentration is increased. This suggests that whilst the optimisation routine has found a solution, the model has had to over fit the model. Without experimental data for run length at these concentrations it is difficult to establish the validity of this result. Greater analysis of how this model produces these trends will however provide more understanding.
CHAPTER 5. MYOSIN-V STEPPING MODELS: ZERO FORCING

Figure 5.5: A comparison of the velocity relationships against ATP and ADP nucleotide concentrations between the model proposed by Skau et al. [44] and the model developed by Wu et al. [61] now optimised within the metamodel framework presented in the previous chapter. Nucleotide concentrations are as in Figure 5.2. The Wu Model (WM) is distinct from the Original Wu Model (OWM) because it is now within the metamodel framework rather than the original authors mathematical structure.
5.1. A COMPARISON OF TWO STEPPING MODELS

(a) Run length against ADP concentration

(b) Run length against ATP concentration

Figure 5.6: A comparison of the run length relationships against ATP and ADP nucleotide concentration between the model proposed by Skau et al. [44] and the model developed by Wu et al. [61] now optimised within the metamodel framework presented in the previous chapter. Nucleotide concentrations are as in Figure 5.3. The Wu Model (WM) is distinct from the Original Wu Model (OWM) because it is now within the metamodel framework rather than the original authors mathematical structure.
The Skau Model and the Wu Model under zero forcing both reproduce the expected velocity against nucleotide concentration results. However neither one adequately reproduces both sets of run length data. As the run length of a given myosin-V molecule primarily depends on both its velocity and its detachment mechanism/rate, further understanding of the detachment mechanism is required. With this information one can begin to identify how to construct a model that reproduces these experimental results and so further the understanding of myosin-V.

### 5.2 Molecular Detachment Mechanisms

There are two different motor detachment mechanisms contained within the meta-model framework. The Wu Model has a constant mechanical detachment rate associated with two states, the assumption being that molecule in a certain conformation leaves the track at a constant rate. The Skau Model incorporates a slightly more intuitive idea of molecular detachment: chemical detachment. It has a parameter-dependent detachment rate associated with a particular branched state. Physically, the detachment corresponds to the breakdown of coordination between the ATP hydrolysis reactions in the two heads - the process that allows myosin-V to take successive steps. Analysis in the previous section suggests that the mechanisms are antagonistic in terms of the results that they reproduce.

#### 5.2.1 Chemical Detachment

An intuitive idea as to how the myosin-V molecule detaches from its track is through a loss of coordination between the chemical reactions at its heads. The Skau Model incorporates this idea by including a pre-detachment state - state 9 in Figure 5.1. Entry into this state from the main hydrolysis cycle corresponds to the rear head releasing an ADP nucleotide before the front head hydrolyses its bound ATP nucleotide. Chemical detachment occurs when the rear head dissociates from the actin track before the front head binds.

The Skau system reproduces the run length against $[ATP]$ result shown by Baker et al. [1]: as $[ATP]$ decreases, run length tends to a very large value. The Skau Model has a chemical detachment rate associated with state 9 that increases proportionally with $[ATP]$, therefore as $[ATP]$ decreases the chemical detachment rate decreases, meaning the probability that the motor remains on the main reaction cycle increases. Run length is determined by this process in competition with the rate at which molecules pass through the hydrolysis cycle (the hydrolysis flux). In the Skau Model the detachment flux goes to zero faster than the hydrolysis flux as $[ATP]$ decreases and so the run length increases as $[ATP]$ decreases as experimentally observed.
5.2.2 Mechanical Detachment

It has been suggested [1] that there exists at least one mechanochemical conformation of the molecule which is subject to a constant detachment rate. This can also be described as a non-zero probability of the molecule being mechanically removed from the actin in a particular state. The model suggested by Baker et al. [1] and developed by Wu et al. [61] contains this form of detachment. Baker et al. suggest that their results show that the state in which the front head was weakly bound and the rear head was strongly bound is most vulnerable to this constant-rate detachment in their model as well as the weakly bound and free state (states 3 and 8 in Figure 5.4).

The Wu Model reproduces the run length against $[ADP]$ result shown by Baker et al. [1]: as $[ADP]$ increases, run length decreases, reaching a non-zero value at saturating concentrations. The feature of the model that reproduces this result is the constant rate of motor detachment from states that have increased probability of occupancy as $[ADP]$ increases. This increases the probability of a motor dissociating from the track and so decreases its run length. This effect is eventually balanced by a lack of molecules in the pre-detachment states.

5.2.3 Cycle with Chemical and Mechanical Detachment

A straightforward way of trying to produce both results would be to create a model with both detachment mechanisms. A simple model with both chemical and mechanical detachment is shown in Figure 5.7.

![Figure 5.7: A simple model with both chemical and mechanical detachment.](image)

Considering this toy model in Figure 5.7 it can be seen that a system that includes both the mechanical and the chemical detachment would not reproduce...
both run length results. Chemical detachment alone has the run length increasing for decreasing $[ATP]$ because the detachment rate goes to zero and so the molecules become more constrained to the track faster than the hydrolysis cycle flux goes to zero. The mechanical detachment is defined so that the run length against $[ADP]$ result is produced and thus ensures that molecules always detach from the track. For constant non-zero $[ADP]$, as $[ATP]$ tends to zero for the combined model molecules would still be detaching mechanically and total detachment would not tend to zero. Therefore, the run length would not get very large for small concentrations of $[ATP]$, and this experimentally measured trend would not be reproduced.

Thus, whilst explaining one result, the mechanical detachment prevents another. Therefore either a modified form of chemical detachment or an alternative detachment mechanism must be included in a model of myosin-V to give both the required run length against $[ADP]$ and $[ATP]$ results.

5.3 The Skau Asymmetric Gating Model

It is experimentally observed that for a given non-zero concentration of ATP, as $[ADP]$ is increased, run length decreases. At saturating concentrations of ADP run length stabilises at a non-zero value.

Baker et al. [1] discuss the possible underlying mechanisms that could give the $[ADP]$ against run length relationship. Detachment from a state that increases in occupancy as $[ADP]$ is increased with state transition rates chosen so that this balances with a sufficient number of hydrolysis cycle completions at saturating $[ADP]$ is one example. This was the method they proposed in their model that was adapted by Wu et al. [61] and is the Wu Model in our metamodel framework. Baker et al. [1] mention an alternative method but do not include it in their model: futile cycling from an $[ADP]$ bound state again with the transition rates chosen so that this balances with the hydrolysis cycle completions could also give the desired result.

The Skau Model has futile cycling from an $[ADP]$ bound state but does not reproduce the $[ADP]$-run length relationship. This suggests that some other part of the model is restricting the production of this result and consequently a modification of the model is required.

As ADP concentration decreases, the run length must increase and so the probability for a molecule to be occupying the futile cycling states must also decrease. Considering Figure 5.1 this corresponds to the transition $4 \rightarrow 6$ decreasing or the corresponding return transition $6 \rightarrow 4$ increasing. However these rates (defined in section 4.2.3) are related to the $4 \leftrightarrow 5$ and the $2 \leftrightarrow 9$ transitions. The former plays a large part in the velocity of the molecules around the hydrolysis cycle and the latter the detachment rate; both of which also have a large effect on the run length. Therefore these transitions need to be decoupled in a physical manner in order for the Skau Model to be able to give the desired result.
Rosenfeld and Sweeny [42] measured a difference in the kinetic rate of release of ADP from the front and rear heads of the myosin-V molecule: this is known as gating. Skau et al. [44] assumed that there is some sort of pocket in which the nucleotide sits that must be opened against the internal strain of the molecule. This is different for each head and so gives the gating effect. It was assumed by the authors that gating is symmetric - that the energy required to open the pocket when the nucleotide is held within it is the same as when it is empty. Relaxing this assumption, helps to decouple the rates as discussed above.

The Skau Asymmetric Gating Model (SAGM) is identical to the Skau Model except that the assumption of symmetric gating is relaxed and the modified starred rates in section 4.2.3 replace their non-starred counterparts. The Skau Model will be used as a benchmark to establish the influence asymmetric gating is having on the results.

The optimised parameters are given in Table 5.1 and their corresponding variability measures $\Delta H_i$ are shown in Table 5.3. The cost of the SAGM is much lower than that of the Skau Model implying that the inclusion of asymmetric gating improves the models agreement with experimental results. The energy difference and energy barrier parameters and their variability for the SAGM are very similar to those for the Skau Model. This is encouraging as it means that the introduction of asymmetric gating has not modified the basic chemical functioning of the model as one would expect physically. The total internal strain energy $E_s$ is much higher and more restricted implying that strain plays a much bigger part in this model as intended. $\alpha E_s$ - the additional energy barrier for entering the futile cycle - has a lower $\Delta H_i$ and is much smaller suggesting the futile cycle plays a much larger role in this model as intended. $\alpha^* E_s$ is larger than $\alpha E_s$ suggesting that getting a nucleotide into the pocket is more difficult than getting it out when the molecule is under strain. $\omega^* E_s$ is very small implying that gating in the main cycle is not asymmetric and the variability of this parameter is small. This could correspond to the pocket being opened sufficiently as this is the only case of the ADP release transition where the myosin-V molecule is fully strained. This suggests that gating in partially strained myosin-V could be far more complex than for the fully strained molecule.

The run length results for the SAGM are shown in Figure 5.8. The velocity results are as before and are omitted for simplicity. The SAGM now gives the experimentally observed run length against $[ADP]$ result. The change in flux of molecules through the futile cycle compared to the hydrolysis cycle (see Figure 5.9) causes this result as postulated above: for low concentrations of ADP the futile flux decreases leading to an increase in run length; as $[ADP]$ increases the futile flux eventually stabilises at a fixed non-zero value.
Figure 5.8: A comparison of the run length relationship against ATP and ADP nucleotide concentrations between two models in the metamodel framework: the original model proposed by Skau et al. [44] (used as a benchmark) and Skau model with asymmetric gating. Nucleotide concentrations are as in Figure 5.3.
5.4. THE SKAU ASYMMETRIC GATING EXTRA MODEL

Relaxing the symmetric gating assumption in the Skau Model enables the run length-[ADP] result but negates the [ATP] one. In the Skau Model the latter result was caused by a low detachment rate at low [ATP] (the detachment rate was proportional to [ATP] and zero at [ATP] = 0) - resulting in a high run length for small [ATP] - and for increasing [ATP], detachment was balanced by an increased molecular velocity resulting in a non-zero run length at saturating ATP concentrations. Introducing flexibility for the optimisation to precisely choose the transition to and from the pre-detachment state was the most likely cause of the switchover of the production of the run length against [ADP]/[ATP] results. As this transitions is clearly an important one, introducing a new path from the pre-detachment state to the main cycle would be a sensible next step.

The Skau Asymmetric Gating Extra Model (SAGEM) is based on the metamodel with pathways A, B, and E included and is shown in Figure 5.10. Pathway E
has been included to enable an additional path from the pre-detachment state in the metamodel; this transition $9 \rightarrow 7$ requires the inclusion of pathway B and is assumed to be identical to the transition $2 \rightarrow 3$. As with the SAGM in the last section the assumption of symmetric gating is again relaxed and the modified starred rates in section 4.2.3 replace their non-starred counterparts.

The optimised parameters for the SAGEM are given in Table 5.1. Interestingly the cost of the SAGEM model is the lowest of all models investigated, implying that it reproduces the experimental results best. When compared to the Skau Model results the energy barriers and differences are very similar with the exception of the ATP hydrolysis reaction values $\Delta G_{T-Dw}^T$ and $G_{T-Dw}^4$ which are both larger. This implies that in order to achieve the low cost parameter point the optimisation had to slow down the rate of ATP hydrolysis and increase the relative energy given from this transition. The energy value chosen by the optimisation ($3.72k_BT$) for the $ATP \rightarrow ADP + P_i$ reaction in the SAGEM is close to the experimentally determined value [19] of $2k_BT$ compared with the other models ($0.12 - 1.03k_BT$). This model is therefore more valid with respect to the chemical energetics data. It is important to note that the energy barrier for this transition is now much more restricted by the optimisation whereas the energy difference is less so. Again, similarly to the SAGM the total internal strain energy $E_s$ is greater than in the Skau model and the variability is less implying strain plays a much larger role.
The energy barrier to enter the futile cycle and its variability is also much less suggesting the futile cycle is also much more important.

Comparing parameters to experiment: $G_{E-T}^4 = 6.05k_B T$ means $u_5$ is much larger than the $0.6 - 1.5k_B T$ range predicted by experiment [55] but smaller than that predicted by the Skau model. $G_{Dw-Ds}^4 = 8.05k_B T$ and so $u_3 \gg 1000s^{-1}$ which is bigger than $250s^{-1}$, just as predicted by experiment [11]. $G_{Ds-E}^4 = 8.36k_B T$ means $u_{45}$ is larger than $30s^{-1}$ the maximum value predicted for this transition by experiment [42]; all the ADP release rates in the different states also vary wildly. $u_{29} \approx 1.06s^{-1}$, $u_{46} \approx 80s^{-1}$ and $u_{37} \gg 100s^{-1}$. This suggests that in this model a variability in ADP release rates is important to produce the experimental trends. $u_{37}$ is the only rate that falls below the experimentally observed kinetic limit; this suggests that perhaps the kinetic experiments restrict the myosin-V dimer in some manner to this or a similar transition. Experimentally measured kinetic rates and a model that accounts for the overall behaviour of the protein fitted to larger scale observables may be fundamentally incompatible - different experiments impose different restrictions on the motor. The parameters for the minimum optimisation point may also not give some experimentally measured kinetic rates because these values do not consider return rates; the corresponding backwards rates for large forwards rates in this model are also very large. This is possibly relevant as it is unclear whether experimental results pertain only to a forwards rate or, more likely, a statistically averaged drift from one state to another.

Many studies compare the magnitude of the transition rates against kinetic studies [2, 28, 44, 61]. Here the energetics of the system is optimised rather than the rates themselves; a small change energetically will make a large difference to the transition rates. Rates determined in the metamodel optimisation framework are not necessarily expected to match experimental data - the behaviour of dynamic properties against cellular conditions ([ATP], [ADP], $f_{ex}$ etc.) is more pertinent given physical the energetic constrictions and model structure.

The velocity and run length against nucleotide concentration results are shown in Figures 5.11 and 5.12 respectively. The SAGEM (metamodel with pathways A B and E included) is the first investigated that reproduces all four sets of velocity and run length against nucleotide results robustly.
Figure 5.11: A comparison of the velocity relationships against ATP and ADP nucleotide concentration between two models in the metamodel framework: the original model proposed by Skau et al. [44] and Skau model with asymmetric gating and an extra pathway from the pre-detachment state. Nucleotide concentrations are as in Figure 5.2.
5.4. THE SKAU ASYMMETRIC GATING EXTRA MODEL

Figure 5.12: A comparison of the run length relationships against ATP and ADP nucleotide concentration between two models in the metamodel framework: the original model proposed by Skau et al. [44] and Skau model with asymmetric gating and an extra pathway from the pre-detachment state. Nucleotide concentrations are as in Figure 5.3. The Skau Asymmetric Gating Extra Model (metamodel with pathways A B and E included) is the first investigated that reproduces both sets of run length against nucleotide results robustly.
This model suggests that the release of ADP is a very important process in the walk of myosin-V. Figure 5.13 shows the relative fluxes across transitions involving ADP release against $[\text{ADP}]$. Flux to the detachment state $J_{29}$ is very low for all values and flux through the additional hydrolysis pathway $J_{37}$ remains relatively constant. This is interesting because these transitions are dependent of $[\text{ADP}]$ meaning the model must have chosen parameters to ensure this. The flux through the main hydrolysis pathway $J_{45}$ and the futile cycle $J_{37}$ exhibit an interesting switchover unlike in the Skau Model (not shown). This is most likely why the SAGEM reproduces the run length against $[\text{ADP}]$ result and the Skau Model does not.

![Figure 5.13](image)

*Figure 5.13: The proportion of fluxes involving ADP release relative to the total flux against $[\text{ADP}]$ for 1mM $[\text{ATP}]$ and low $[\text{P}_i]$ for the SAGEM. Flux to the detachment state $J_{29}$ is very low for all values, suggesting that the model requires the motor to be highly processive as expected. The flux through the additional hydrolysis pathway $J_{37}$ remains relatively constant. The relative importance of flux through the main hydrolysis pathway $J_{45}$ and the futile cycle $J_{\text{fut}}$ switchover as $[\text{ADP}]$ increases.*

### 5.4.1 Critique of the SAGEM

As the SAGM and the SAGEM have the largest number of optimisable parameters it can be argued that the optimisation has over fitted the model to the data due to under-constraint of the parameters. However I would argue that the Skau Model optimisation was likely *over-constrained* by the symmetric gating and reaction pathway assumptions. Furthermore, as can be seen in Figures 5.11 and 5.12, the models were not optimised against experimental trends. They were optimised against very few experimental data points simply for the routine to choose the
correct scale of the model dynamic-property/nucleotide relationships. Lastly one of the major symptoms of overfitting is not present: a narrow region of nucleotide concentration within which the trends are produced (unlike the results given for the Wu Model in section 5.1.1). Thus, it is reasonably safe to conclude that these models shown that asymmetric gating and futile cycling with the additional hydrolysis pathways B and E are likely required for myosin-V to reproduce the experimentally observed trends of run length against [ADP] and [ATP].

5.5 Conclusions

Many discrete-stochastic models of myosin-V have been investigated in this chapter. Existing stepping cycles [1, 44, 61] have been incorporated into the metamodel framework that optimises relevant parameters so that the model reproduces experimental data (see Chapter 4). This allowed the models to be compared against each other and has shown some interesting results summarised in Table 5.2.

Six models under zero forcing were investigated in terms of their reproduction of certain experimental trends: velocity versus [ADP] and [ATP] and run length versus [ADP] and [ATP]. The cost (the cost function value as defined in chapter 4) shows how closely a model reproduces certain key experimental data points. An optimisation routine chose model parameters for the models in the metamodel framework in order to minimise this value. All models considered produce the velocity results suggesting that the main ATP hydrolysis pathway, that is included in all the models, is the mechanism that gives these results.

The Original Wu model [1, 61] and Original Skau model fed into the metamodel cost function routine had a high value as they did not reproduce the run length versus [ATP] and run length versus [ADP] results respectively. The corresponding metamodel versions of these models (the Wu Model and the Skau Model) had much lower costs due to the optimisation applied.

The Wu Model includes additional hydrolysis pathways (B and C) and allows mechanical detachment (D): a constant rate of detachment from certain states. All experimental trends considered in this section were reproduced although the run length versus [ATP] results only held for a very small window of [ADP] concentrations. This suggests that the optimisation routine found a solution by overfitting the parameters to the data. This result was however produced very well by including only chemical detachment (detachment though a loss of coordination between the heads of the protein) and futile cycling (a pathway that accounts for failed stepping) in the Skau Model. However the Skau Model failed to reproduce the run length versus [ADP] results.

The differing detachment mechanisms were identified as the primary cause of the different run length results. Further analysis of the two detachment processes in section 1.3 revealed that the mechanical detachment precluded reproduction of the results caused by the chemical detachment. Therefore the Skau Model had to be modified in a manner that did not include mechanical detachment in order for it
to reproduce the run length versus $[ADP]$ result.

Baker et al. [1] suggested that futile cycling from an ADP-bound state could also give the run length versus $[ADP]$ result (although they failed to take this any further and included mechanical detachment in their model instead). The Skau Model contains such a process but when optimised fails to reproduce the result. Relaxing the symmetric gating assumption in the Skau Model gives the SAGM (see section 5.3). This model gives the run length versus $[ADP]$ trend as shown in Figure 5.8, however no longer gives the run length versus $[ATP]$ relationship. The latter result depends on the transitions to and from pre-detachment state 9 which were changed by including asymmetric gating parameters in the optimisation function. Adding additional hydrolysis pathways B and E again modifies this dependence and gives the SAGEM.

The SAGEM reproduces all sets of zero-forcing experimental trends considered (see Figures 5.11 and 5.12). Its also has the lowest cost function out of all models investigated. As it is the only model to reproduce all of the experimental trends, asymmetric gating and futile cycling and the additional hydrolysis pathways B and E are likely to be vital in the function of myosin-V.

Different mechanisms within the models can now be suggested to enable a model to give certain results. As the velocity experimental trends are reproduced in all models investigated, these are likely to be caused by the common feature: the main hydrolysis pathway. The decrease of run length as $[ATP]$ increases is only observed in models with the loss-of-chemical-coordination detachment, thus suggests that this is the enabling mechanism that allows a model to reproduce the $L$ versus $[ATP]$ experimental trend. The decrease of run length as $[ADP]$ increases corresponds to behaviour switchover of the molecules akin to wheelspin: as $[ADP]$ increases there is a decrease in the main hydrolysis cycle flux and an increase in the futile cycle flux (see figure 5.13). As this was not the case in the Skau Model, the inclusion of asymmetric gating is likely to enable the model to reproduce this result.

Physically, asymmetric gating of ADP corresponds to a change in the conformation of the pocket in which the nucleotide resides after ADP release, for example when ADP is released from the front head before the rear head, the strain in the molecule collapses the front pocket making it much more difficult for ADP to reenter when compared to the same process occurring in the rear head. The collapse in the front head is then communicated to the rear head through a change in intermolecular strain preventing the rear head from releasing ADP and so the molecule progresses through the futile cycle. The futile cycle corresponds to a failed step in which the front head of myosin-V unbinds chemically before the rear head then re-attaches. This has been experimentally observed in recent experiments and is known as the foot stomp [25]. In the SAGEM a mechanism for this is presented and it is postulated that this occurs more under high $[ADP]$ and thus causes the experimentally measured decrease in run length for increasing $[ADP]$. Experimental investigation is now required to confirm these predictions.

The SAGEM has been optimised under the assumption of zero external force. The next chapter explores the influence of the introduction of this.
Chapter 6

Myosin-V Models Under Force

“We live on an island surrounded by a sea of ignorance. As our island of knowledge grows, so does the shore of our ignorance.”

John Archibald Wheeler

In the previous chapter a set of mechanisms were postulated that enabled the molecular motor myosin-V to reproduce important experimentally observed trends under the assumption of zero forcing. This was achieved through analysing several different models of the protein. Velocity trends [1, 16] were reproduced well across all models and so were suggested to result directly from the underlying well-known hydrolysis cycle. Experimental run length results [1] were also investigated and it was shown that chemical detachment from the track reproduced the $L$ versus $[ATP]$ trend and futile cycling (or foot stomping) combined with asymmetric gating of ADP release/binding enabled models to reproduce the $L$ versus $[ADP]$ observed trends.

Further experimental investigations have been conducted on the myosin-V protein to investigate its behaviour when an external force is applied. The transition from zero forcing, with the motor at full speed, to the point at which force balances this and the motor stalls (the stall force) is well understood [52]. The velocity has also been shown to stay constant as the motor is pulled in the direction of movement and goes negative as the external force is increased beyond stall against the direction of movement [17] with a smooth transition in between [52] (see figure 6.2). This negative velocity has been postulated to be caused by myosin-V slipping backwards along the track rather than stepping backwards through the hydrolysis cycle creating ATP.

In this chapter we introduce the external forcing components into the optimisation (terms $\Delta_{10}$ and $\Delta_{11}$) and investigate the effect this has on the SAGEM. The aim is to identify and test suggested mechanisms [2] that enable a model to reproduce experimentally observed trends for varying external force that are compatible with
the mechanisms that give the previously discussed zero-forcing results.
Figure 6.1: The run length relationships against ATP and ADP nucleotide concentration for the Bierbaum Model. Nucleotide concentrations are taken from Baker et al. [1] at \([\text{ATP}] = 1\text{mM}\). In this model the run length relationship against \([\text{ADP}]\) matches up with the Baker’s data for small values however at saturating concentrations, the run length goes to zero which is contrary to the experimental data. The relationship against \([\text{ATP}]\) is not reproduced; the run length is small for small \([\text{ATP}]\).
Firstly a model suggested by Bierbaum [2] that reproduces many forcing results for myosin-V is discussed and compared with the forcing results of the Skau Model [44]. A selection of mechanisms from both studies has been included in the metamodel framework and the SAGEM is optimised under forcing to give the Skau Asymmetric Gating Extra Forcing Model. Two variants are considered: the model with two (SAGEFM2) and three (SAGEFM3) substeps.

### 6.1 The Bierbaum Model and the Skau Model

The Bierbaum Model [2] was the first discrete-stochastic model to reproduce high positive and negative external forcing results for myosin-V. Mechanisms were adapted from models of kinesin and applied to give the experimentally observed trends. The model included two mechanochemical transition cycles: one corresponding to the main hydrolysis cycle and the other describing the process that leads to molecular slip. The main hydrolysis cycle enabled the model to reproduce the observed velocity against nucleotide concentration results [1, 16]. Mechanical detachment (see section 5.2.2) similar to that used in the Wu Model was included within the model that gave accurate run length results for low concentrations of ADP, however at saturating concentrations this reproduction breaks down. Due to the lack of an ATP-dependent detachment or futile cycling mechanism (as discussed in the previous chapter) the Bierbaum Model also does not reproduce the run length against ATP data. These results are shown in Figure 6.1.

In this chapter the mechanisms of myosin-V established in the previous chapter are developed to address the run length issue in the Bierbaum Model by adapting the SAGEM to give forcing results.

There are two important discrete stochastic models that attempt to model external forcing on myosin-V, the Skau Model and the Bierbaum Model. The velocity results for both are shown in Figure 6.2. Stall force to high negative forcing velocity results [17, 52] were enabled in the Bierbaum Model by terms that encoded the force-driven reversal of the powerstroke that has a decaying effect as force become more negative. In the Skau Model this was caused by an interaction between this process and pulling the molecule in a spring-like manner, this however breaks down as external forcing becomes highly negative as the spring effect starts to dominate. At high positive forcing, the velocity of the Skau Model goes to zero. In the Bierbaum Model a switchover between the main hydrolysis cycle and the slipping cycle is captured through the transition rate terms that cause certain rates (corresponding to the binding of nucleotides to the myosin heads) to decrease with increasing force. At high forces this causes slipping backwards along the track and enables the reproduction of measured velocity results for high forcing. Both models reproduce velocity against small forcing results by Uemura et al. [52] reasonably well. Only the Bierbaum Model gives the higher-force relationships shown by Gebhardt et al. [17]. Neither gives a particularly accurate stall force result - both predict it to be about 1.7pN rather than the 2.5pN shown by experiment.
6.2 SAGEM UNDER FORCING

The Skau Asymmetric Gating Extra Model under forcing (SAGEFM) is defined to be the metamodel described in section 4.2 with pathways A (futile cycling and chemical detachment), B, E (additional hydrolysis cycles) and F (molecular slip from states in which the molecule is attached to the actin with only one head) included. This is shown in Figure 6.3. There are two variants: one that describes the molecule taking two substeps per 36nm step (SAGEFM2) as postulated by Uemura et al. [52] and one with three substeps (SAGEFM3) as shown experimentally by Capello [5]. Terms that change with changing external force encode its effect on the powerstroke, the elastic nature of the protein, nucleotide binding to the heads and slipping down the track.

It is important to recalculate the parameters of the new forcing variants of the SAGEM as the reproduction of forcing results may depend on the relative occupancy of different states. These models are optimised with the complete cost function (including the forcing terms, unlike in the last chapter) to match the chosen experimental data points described in section 4.3. The results from these optimisations are given in Table 6.1.

### 6.2.1 Two Substeps

The SAGEFM2 reproduces the experimentally observed velocity against nucleotide concentration trends very well and these are not shown here for succinctness. The more interesting run length results are shown in Figure 6.4 and demonstrate that the SAGEFM2 reproduces measured run length data much better than the Bierbaum Model. The $L$ versus $[ATP]$ trend is well produced due to the inclusion of

<table>
<thead>
<tr>
<th>Model</th>
<th>Cost</th>
<th>$\Delta G_{T-Dw}$</th>
<th>$\Delta G_{Dw-Ds}$</th>
<th>$\Delta G_{Ds-E}$</th>
<th>$\Delta G_{E-T}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAGEFM2</td>
<td>19.3</td>
<td>3.11</td>
<td>6.88</td>
<td>-10.17</td>
<td>13.31</td>
</tr>
<tr>
<td>SAGEFM3</td>
<td>13.4</td>
<td>2.65</td>
<td>6.79</td>
<td>-10.04</td>
<td>13.72</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>$G_{T-Dw}^\dagger$</th>
<th>$G_{Dw-Ds}^\dagger$</th>
<th>$G_{Ds-E}^\dagger$</th>
<th>$G_{E-T}^\dagger$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAGEFM2</td>
<td>3.83</td>
<td>14.72</td>
<td>15.15</td>
<td>5.79</td>
</tr>
<tr>
<td>SAGEFM3</td>
<td>4.56</td>
<td>15.35</td>
<td>12.93</td>
<td>5.82</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>$E_s$</th>
<th>$\alpha E_s$</th>
<th>$\beta E_s$</th>
<th>$\gamma E_s$</th>
<th>$\alpha^* E_s$</th>
<th>$\beta^* E_s$</th>
<th>$\gamma^* E_s$</th>
<th>$\omega^* E_s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAGEFM2</td>
<td>11.27</td>
<td>1.98</td>
<td>1.59</td>
<td>16.49</td>
<td>1.23</td>
<td>5.66</td>
<td>13.69</td>
<td>0.11</td>
</tr>
<tr>
<td>SAGEFM3</td>
<td>12.19</td>
<td>2.53</td>
<td>4.69</td>
<td>11.07</td>
<td>2.53</td>
<td>7.05</td>
<td>17.86</td>
<td>0.14</td>
</tr>
</tbody>
</table>

Table 6.1: Optimised parameters and final cost function values for the Skau Asymmetric Gating Extra Forcing Model with two and three substeps. Parameters relating to the energy barriers and differences between states are shown as well as intramolecular strain values.
CHAPTER 6. MYOSIN-V MODELS UNDER FORCE

chemical detachment and excluding mechanical detachment from the model. The $L$ versus $[ADP]$ results are well reproduced for high concentrations of ADP but not for low concentrations. This is due to the optimisation picking parameters that do not give the flux switchover between the main hydrolysis cycle and the futile cycle (see Figure 6.5) as required for this result. The requirement for the optimisation now to fit to forcing data has interfered with this process.

The optimisation results for the SAGEFM2 are given in Table 6.1. The SAGEFM2 has a higher cost function value than the SAGEM but this is expected as there are additional forcing terms included. $\alpha E_s > \alpha^* E_s$ and so the gating terms for entry into the futile cycle imply that rebinding of an ADP nucleotide occurs at a higher rate than disassociation. This is most likely what causes the model not to reproduce the $L$ versus $[ADP]$ result. One would expect that the collapse of the pocket after nucleotide release would require $\alpha E_s < \alpha^* E_s$, $\beta E_s < \beta^* E_s$, $\gamma E_s < \gamma^* E_s$ and $\omega^* E_s > 0$. Only the second and the last of these conditions are satisfied at this parameter point for this model.

The velocity against external forcing results for the SAGEFM2 are shown in Figure 6.6. For high negative forcing, velocity remains constant - unlike in the Skau Model this is caused by the force driven reversal of the powerstroke, coupled with a modified spring-like extension of the protein as it is pulled backwards through the powerstroke. These effects decay to zero as forcing become heavily negative. As forcing increases through $f_{ex} = 0pN$ these effects start to dominate and the motor stalls at $f_{ex} \approx 2pN$, slightly below experimentally observed values [5, 52]. Beyond this these effects become even greater and the molecule begins to slip backwards along the track.
6.2. SAGEM UNDER FORCING

Figure 6.2: A comparison of the velocity relationship against external forcing between the Original Skau Model (blue) and the Bierbaum Model (red). Three sets of nucleotide concentrations are plotted, with the corresponding experimental data points: \([\text{ATP}] = 1\text{mM}\) and \([\text{ADP}] = 1\) (solid, circles \cite{52}), \([\text{ATP}] = 1\text{mM}\) and \([\text{ADP}] = 200\) (dotted, upwards-pointing triangles \cite{52}) and \([\text{ATP}] = 10\) and \([\text{ADP}] = 1\) (dashed, squares \cite{52}). All are at low \([\text{Pi}]\). Data points from Gebhardt et al. \cite{17} at \([\text{ATP}] = 1\text{mM}\) are also plotted (downwards-pointing triangles). At external forces below the stall force the models have positive velocity. The force driven reversal of the powerstroke causes a transition from stall at \(1.7\text{pN}\) to a large positive velocity below \(0\text{pN}\). For large negative forces this is held constant in the Bierbaum model through force-dependent nucleotide-binding terms and the relative occupancy of different states, whereas in the Skau Model this is due to terms that encode the elastic nature of the protein. Above stall force the Skau model has velocity going to zero whereas in the Bierbaum Model the slipping mechanism becomes dominant and the motor slips backwards along the track.
Figure 6.3: The Skau Asymmetric Gating Extra Forcing Model in the metamodel framework with the hydrolysis pathway, futile cycling and chemical detachment (blue), additional pathways (yellow and red) and molecular slip (green). This model can have two modes - a two substep mode with physical movement in transitions $1 \leftrightarrow 2$ and $2 \leftrightarrow 3$ or a three substep mode with these and an additional move in transitions $3 \leftrightarrow 4$ or $7 \leftrightarrow 5$ depending on the hydrolysis pathway.
Figure 6.4: The run length relationships against ATP and ADP nucleotide concentration for the Skau Asymmetric Gating Extra Forcing Model with two substeps (SAGEFM2). Nucleotide concentrations are as in Figure 5.3. The run length relationship is reproduced poorly for low [ADP] but well at saturating concentrations. The model gives a good fit for the run length [ATP] results.
CHAPTER 6. MYOSIN-V MODELS UNDER FORCE

Figure 6.5: The proportion of fluxes involving ADP release relative to the total flux against [ADP] for 1mM [ATP] and low [Pi] for the SAGEFM2. Flux to the detachment state $J_{29}$ is very low for all values and flux through the additional hydrolysis pathway $J_{37}$ remains relatively constant. The relative importance of flux through the main hydrolysis pathway $J_{45}$ and the futile cycle $J_{fut}$ change slightly as [ADP] increases but not as dramatically as in the SAGEFM3 (shown later). This explains why the SAGEFM2 does not reproduce the $L$ vs [ADP] experimental findings as well as the SAGEFM3.

This model reproduces many important results but does not give the $L$ versus [ADP] run length result. It also fails to give the expected physical conditions upon the asymmetric gating parameters and, as with all models investigated thus far, gives a stall force that is lower than experimentally measured values.

6.2.2 Three Substeps

The SAGEFM3 assumes that the myosin-V motor takes three substeps in every 36nm move along its actin track. This modifies the forcing terms within the model that are contained within the main hydrolysis cycle and introduces an additional forcing term related to moving into the telemark stance. This has a profound effect on the basic postulated functioning of the protein.

The results of the optimisation are given in Table 6.1 and show that the cost function for the SAGEFM3 is lower than that for the SAGEFM2. Energy barriers, differences and the total internal strain energy value are similar in both models. However, the asymmetric gating values vary significantly. Most importantly conditions $\alpha E_s < \alpha^* E_s$, $\beta E_s < \beta^* E_s$, $\gamma E_s < \gamma^* E_s$ and $\omega^* E_s > 0$ are all satisfied for the SAGEFM3.

The velocity against nucleotide concentration results are given in Figure 6.8 and
6.2. SAGEM UNDER FORCING

Figure 6.6: The velocity relationship against external forcing for the Skau Asymmetric Gating Extra Forcing Model with two substeps. Three sets of nucleotide concentrations are plotted, with the corresponding experimental data points: $[ATP] = 1\, \text{mM}$ and $[ADP] = 1$ (solid, circles [52]), $[ATP] = 1\, \text{mM}$ and $[ADP] = 200$ (dotted, upwards-pointing triangles [52]) and $[ATP] = 10$ and $[ADP] = 1$ (dashed, squares [52]). All are at low $[Pi]$. Data points from Gebhardt et al. [17] at $[ATP] = 1\, \text{mM}$ are also plotted (downwards-pointing triangles). Relevant data points in the optimisation are also shown: $\Delta_1$ (plus), $\Delta_4$ (left-pointing triangle), $\Delta_5$ (asterisk) and $\Delta_{11}$ (cross). At external forces below the stall force the model has a positive velocity that I postulate is due to a complex interaction between the elastic properties of the molecule and the effect of the external force on the powerstroke. This causes a transition from stall at $2\, \text{pN}$ to constant positive velocity below $-1\, \text{pN}$. Above stall force the slipping mechanism becomes dominant and the motor slips backwards along the track.

shows that again these experimental trends are well reproduced. The run length results against nucleotide concentration results are given in Figure 6.9. Again the $[ATP]$ result is well captured. The $[ADP]$ result is reproduced well at saturating concentrations and at low concentration except for very close to zero. Figure 6.7 shows that there remains a large gearing effect: as $[ADP]$ increases the probability that motors pass through the futile cycle and although not quite the switchover observed in the SAGEM, sufficient to cause a reasonable decrease in run length as $[ADP]$ is increased and to a much greater extent than in the SAGEFM2. This suggests that an increased number of substeps could also have some role to play in the reproduction of the $L$ versus $[ADP]$ result observed experimentally.

The velocity against forcing results are shown in Figure 6.10. At external forces below $-1\, \text{pN}$ velocity remains constant and positive as shown by experiment [17].
Between this value and stall force at 2.5 pN the velocity decays to zero due to a complex interaction between terms that encode the elastic properties of the molecule, the effect of external force on the powerstroke as well as nucleotide binding terms. This stall force result is similar to experimental findings [5, 52] and a much better approximation than any model considered so far. Above stall force the slipping mechanism becomes dominant and the motor slips backwards along the track.

Figure 6.7: The proportion of fluxes involving ADP release relative to the total flux against [ADP] for 1mM [ATP] and low [P_i] for the SAGEFM3. Flux to the detachment state J_{29} is very low for all values, suggesting that the model requires the motor to be highly processive as expected. The flux through the additional hydrolysis pathway J_{37} remains relatively constant. The relative importance of flux through the main hydrolysis pathway J_{45} and the futile cycle J_{fut} approach switchover as [ADP] increases. This is in contrast to the SAGEFM2 (Figure 6.5), suggesting that a model with 3 substeps is more accurate as suggested by some experimentalists [5].
6.2. SAGEM UNDER FORCING

The velocity relationships against ATP and ADP nucleotide concentration for the Skau Asymmetric Gating Extra Forcing Model with three substeps (SAGEFM3). Nucleotide concentrations are as in Figure 5.2. The velocity experimental trends are again reproduced very well as with all models considered. The most likely reason for this is that they result from the main hydrolysis cycle as it is the fundamental process by which the myosin-V protein steps in all models.
Figure 6.9: The run length relationships against ATP and ADP nucleotide concentration for the Skau Asymmetric Gating Extra Forcing Model with three substeps (SAGEFM3). Nucleotide concentrations are as in Figure 5.3. The L vs [ATP] experimental trend is again reproduced very well as with all models investigated in this thesis with the detachment mechanism that corresponds to a loss of chemical coordination between the heads. Importantly, the L vs [ADP] experimentally observed trend is also reproduced well.
6.3. DISCUSSION

Figure 6.10: The relationship of velocity to external forcing for the Skau Asymmetric Gating Extra Forcing Model with three substeps. Three sets of nucleotide concentrations are plotted, with the corresponding experimental data points: [ATP] = 1mM and [ADP] = 1 (solid, circles [52]), [ATP] = 1mM and [ADP] = 200 (dotted, upwards-pointing triangles [52]) and [ATP] = 10 and [ADP] = 1 (dashed, squares [52]). All are at low [Pi]. Data points from Gebhardt et al. [17] at [ATP] = 1mM are also plotted (downwards-pointing triangles). Relevant data points in the optimisation are also shown: $\Delta_1$ (plus), $\Delta_4$ (left-pointing triangle), $\Delta_5$ (asterisk) and $\Delta_11$ (cross). These results are similar to the SAGEFM2 except that the stall force is higher at 2.5pN. At external forces below the stall force the model has a positive velocity that I postulate is due to a complex interaction between the elastic properties of the molecule and the effect of external force on the powerstroke as well as nucleotide binding terms. This causes a transition from stall at 2.5pN (similar to experimental findings [5, 52]) to constant positive velocity below −1pN. Above stall force the slipping mechanism becomes dominant and the motor slips backwards along the track. This model has the best fit to experimental data thus far.

6.3 Discussion

Several models of myosin-V have been investigated under external forcing. The Skau Model and the Bierbaum Model were the only complex models of the myosin-V walk under forcing that existed within the literature prior to this study. Both used different proposed mechanisms to reproduce experimental data shown by Uemura et al. [52] and Gebhardt et al. [17]. Both included terms that caused external forcing to affect the powerstroke in one direction only: the Skau Model included
terms that also encoded the elastic properties of the protein and the Bierbaum model included forcing terms that encoded force-dependent nucleotide binding and molecular slip down the track for high forces. Both showed good agreement with low forcing results however only the Bierbaum model functioned at high forcing results. With regards to run length against nucleotide concentration results, the detachment mechanism in the Skau model was chemical and that in the Bierbaum model was mechanical although the Skau Model was much more accurate overall. Different competing mechanisms for reproducing these results were investigated.

Applying the metamodel framework to the problem I postulate that an alternative hybrid set of mechanisms causes the experimentally observed trends. They are all described in the Skau Asymmetric Gating Extra Forcing Model with three substeps. I postulate that the well-understood main hydrolysis cycle gives the velocity against nucleotide concentration trends as all models considered give these results. Myosin-V molecules detach from the track when their heads lose chemical coordination and this gives the run length against $[ATP]$ results. Ideally more experimental data points are required to confirm this - particularly at higher concentrations; however, molecules are difficult to track experimentally in these regimes due to their speed. An asymmetrically gated release of ADP coupled with futile cycling (also known as foot stomping) gives the run length against $[ADP]$ results. An interaction between the elastic properties of the molecule, the effect of force on the powerstroke and the nucleotide binding rates gives the velocity against forcing results for when $f_{ex}$ is both low and highly negative. When external forcing becomes highly positive slipping along the track gives the velocity results. Three substeps per 36nm myosin-V step are required to make the protein robust enough to have such a high stall force as $2.5pN$.

The SAGEFM3 encodes all of these features and provides the best fit to the experimental data out of all models investigated in this thesis. Importantly the velocity against force data was so well reproduced despite the fact that there were only two terms in the optimisation that included forcing. These were set at $f_{ex} = 2.5pN$ and the model reproduced the results closely from $-5pN$ up to $5pN$ and beyond. This is compelling evidence to suggest that the mechanisms themselves are responsible for the good fit to the data rather than the optimisation and suggest that this is a valid model. Experimental work is now required to confirm this.
Chapter 7

Conclusion

“If mankind minus one were of one opinion, then mankind is no more justified in silencing the one than the one - if he had the power - would be justified in silencing mankind”

John Stuart Mill

Molecular motors fulfil many important roles within biological cells. This thesis has focused upon mathematical descriptions of stepping motors that move along tracks, in particular myosin-V that steps along actin filaments. These descriptions are vital in order to understand and predict the behaviour of these proteins. The majority of previous studies have been experimental although there has been some theoretical work [2, 8, 26, 27, 28, 29, 30, 44, 56, 57, 61]. These studies encode postulated mechanisms of myosin-V into mathematical descriptions and demonstrate how such ideas can give experimentally observed results such as step sizes, velocities, dispersions, run lengths, dwell times etc. In this thesis I have investigated and developed novel methods for calculating observable properties of myosin-V and have postulated mechanisms that give previously measured but poorly understood velocity and run length relationships.

In section 2.4.1 I discussed a well-known method to account mathematically for molecular motors detaching from their tracks known as renormalisation [27]. This process involved rescaling quantities within the calculations so that required results could be derived from a different description in which the motors did not detach. I have shown that, in a two state system, this is an approximation that breaks down when the difference in the detachment rate from different states becomes large. The results depart from the original authors’ claims but somehow still gives some observables such as the velocity results. This method warrants further investigation to establish how this problem relates to systems with an arbitrary number of states, how large the error can become and whether a modification to the method is needed to correct it.
In a discrete-stochastic model of myosin-V the mechanochemical conformations of
the molecule are discretised into states. Molecules pass from one state to another
with a certain probability that can be characterised by a transition rate. The dy-
namic properties of the motors can be calculated from these rates. A fundamental
aspect of the observable quantities in discrete-stochastic models is that their calcu-
lation for any given model is dependent on its underlying state structure. Previous
studies gave these calculations for several specific structures. In Chapter 3 I pre-
sented a novel, generalised and straightforward graphical method for calculating
dynamic properties of molecular motors in discrete-stochastic systems. It allows
the calculation of the steady-state probabilities, velocity, dispersion and random-
ness ratio for any proposed model through analysis of its structure. Results for
n-state model types are presented: single chain, parallel pathway, divided path-
way and divided pathway with a chain. A novel technique for combining multiple
model architectures coupled at a reference state is also demonstrated. I also pre-
sented many 4-state examples to illustrate the effectiveness and simplicity of the
methods. They provide powerful theoretical tools for investigating how the un-
derlying transition rates of a molecular motor affect its dynamic properties. The
dynamic properties of smaller models can now be calculated simply. Large and
highly complex models can be classified by their structure and these methods can
give the steady-state probabilities and dynamic properties readily without the need
to perform computationally expensive calculations.

Understanding how to calculate dynamic properties very quickly and in a straight-
forward manner was a tremendous advantage in the rest of the work in this thesis.
The detail of stepping mechanisms of the linearly processive motor protein myosin-
V is not well understood [1, 5, 10, 11, 15, 16, 36, 37, 43, 45, 52, 54, 55, 57, 62]
and many of the existing models reflect this. My results allowed the calculation of
dynamic properties for many different system architectures, and greatly aided their
analysis. The calculations became much more rapid computationally compared to
the work by Skau et al. [44] and allowed me to apply the sophisticated optimisation
techniques in [44] to complex models in order to distinguish between them on the
basis of the fit to experimental data.

A metamodel of myosin-V was presented in Chapter 4. This mathematical descrip-
tion of the molecular motor encoded many mechanisms that have been postulated
by various studies to give experimentally observed dynamic properties. Central to
the system is the main hydrolysis cycle in which myosin-V hydrolyses ATP into
ADP and Pi at two coordinated reaction sites to drive the stepping motion along
the track [41, 55]. Alternative and less well-understood hydrolysis pathways that
fulfilled the same function but in a slightly different manner were included [1, 2, 61]
as well as a mechanism that encoded failed or futile stepping (also known as foot
stomping) [44]. Two notions of molecular detachment from the track were also
included - one that involved a loss of chemical coordination between reaction sites
[44] and one that involved a constant probability of being ripped off the track me-
chanically from certain states [1, 2, 61]. Each step that the myosin-V molecule
makes from one binding site on the actin track to the next has been shown to
occur in either two [52] or three [5] substeps. The metamodel can be configured
with either of these settings. Forcing dependance was included in powerstroke (the large substep that myosin-V takes) reversal [2, 44], nucleotide binding [2], elastic stretching of the molecule [44] and molecular slip [2, 17] (where the motor slips backwards down the track).

The many mechanisms that the metamodel of myosin-V captures work well in isolation as shown by several studies [1, 2, 44, 61] but their effects may overlap and interfere with each other. Therefore a method based on the optimisation procedure developed by Skau et al. [44] was defined in Chapter 4 to pick a combination of mechanisms in the metamodel framework, defined as a model, and optimise the free parameters to get a best fit to experimental results. These models then can be compared quantitatively against each other to establish their relative validity.

The metamodel was applied in Chapter 5 to investigate models and mechanisms of myosin-V without external forcing. Experimental velocity results against varying nucleotide concentration were found to be reproduced by every model considered, suggesting that the mechanism that causes these results is fundamental to all the models - most likely the most well-understood aspect: the main hydrolysis cycle. Run length results however proved to be a little more interesting. Before this work, two mechanisms to reproduce experimentally shown run length against nucleotide concentration data had been postulated. Both involved detachment from the track, one was due to a loss of chemical coordination between the myosin-V heads [44] and the other was due to a constant probability of detachment from certain states known as mechanical detachment. Chemical detachment was shown to give run length against $[\text{ATP}]$ observed trends and mechanical detachment was shown to give run length against $[\text{ADP}]$ observed trends. This was discussed and demonstrated in Chapter 5 using the metamodel. Interestingly models with a combination of both of these mechanisms only give the $[\text{ADP}]$ result and it was argued in the Chapter 5 that this was due to interference between the competing mechanisms. Several further models were investigated and one was found that gave all velocity and run length zero-forcing experimental trends. In this model (the SAGEM) chemical detachment caused the run length against $[\text{ATP}]$ result and futile cycling (or foot stomping) coupled with an effect coined asymmetric gating (different energy barriers for both releasing and binding nucleotides to the heads) and an alternative hydrolysis pathway gave the $[\text{ADP}]$ result. The SAGEM was the first model in this thesis and the literature to reproduce all the velocity and run length results against nucleotide concentrations.

The metamodel was applied to myosin-V models that included external forcing in Chapter 6. Again, experimental velocity results against varying nucleotide concentration were found to be reproduced by every model considered, however, key to an accurate external forcing model is the reproduction of the extensively studied and well-characterised velocity against force curves [17, 52]. Two sophisticated models that had already approached this were investigated [2, 44]. Both included terms that caused external forcing to affect the powerstroke in one direction only: one included terms that also encoded the elastic properties of the protein and the other included forcing terms that encoded force-dependent nucleotide binding and molecular slip down the track for high forces. Both showed good agreement with
low forcing results however only one reproduced the high forcing results. Both predicted lower stall forces (at about 1.7pN) than shown by experiment [52]. With regard to run length against nucleotide concentration results, one of the models’ detachment mechanism was chemical and the other was mechanical and so the same issues arose that were discussed in Chapter 5.

The SAGEM description of myosin-V was considered under forcing to create the SAGEFM2 and the SAGEFM3 for the two and three substep cases respectively. Both included forcing terms that encoded an interaction between the elastic properties of the molecule, the effect of force on the powerstroke and the nucleotide binding rates. These give the velocity against forcing results for when $f_{ex}$ is both low and highly negative. When external forcing becomes highly positive, both models also encoded slipping backwards along the track and once optimised, the models gave a very good fit to the velocity-forcing data. However, the 2 substep case (the SAGEFM2) reproduced the $L$ versus $[ADP]$ results poorly due to the requirement for the optimisation to fit to the new forcing data. The stall force was also slightly lower than measured in experiment [52] at 1.7pN. The SAGEFM3 - with three substeps - once optimised gave a better fit to the $L$ versus $[ADP]$ results, a much better fit to the velocity-forcing results and the stall force matched up with experiment at 2.5pN. This suggests three smaller rather than two larger substeps are required to make the protein robust enough to have such a high stall force.

The work in chapters 5 and 6 of this thesis applied the metamodel framework to investigate how possible stepping mechanisms of myosin-V can give experimentally observed trends. From these results I postulate that the nature of the mechanochemical coupling of the heads gives the velocity against nucleotide concentration results, the run length against $[ATP]$ result is given by detachment corresponding to a breakdown of the chemical coordination and the run length against $[ADP]$ result is given by an additional hydrolysis pathway, futile cycling and the effects of asymmetric gating. Measured velocity against external forcing behaviour is given through a complex interaction of the elastic nature of the protein, a lack of a force-driven reversal of the powerstroke and the effect of forcing on the nucleotide binding rates. High positive forcing-velocity results are given by the molecule slipping along the track. Finally I postulate that the molecule takes at least three substeps as it passes from one binding site to another in order to be robust enough to have a high stall force.

Much of this work can be taken further. The renormalisation techniques described in section 2.4.1 have been shown to depart from the authors’ claims - the renormalised probabilities are not the same as the on-track probabilities. It would be of interest to investigate why and to what extent in general by applying it to the methods presented in chapter 3 to investigate the validity of renormalisation for generalised probabilities, velocities and dispersions, as well as to other work in the field on dwell times [50]. The methods used to investigate the underlying mechanism of myosin-V can be applied to other walking molecular motors such as myosin-VI or kinesin, potentially providing unique insight into the underlying mechanisms of additional motor proteins.
Understanding how molecular motors function is essential in understanding the inner workings of the cell. With a greater understanding of these proteins conditions such as Griscelli disease [39] can be understood better and perhaps this can lead to treatments being formulated. Understanding how to replicate these proteins can lead to synthetic biological motors [23, 34] that can be created as part of nanoscale machinery. In this work I have addressed part of this complex puzzle. Further growth of the field is essential in understanding human and animal biology; it can ultimately have important technological and medical implications for society.
Appendices
A. MATLAB Code

In this thesis numerics have been used to maximise a given model’s agreement with experimental results. The code is presented here.

A bespoke optimisation routine based on a simulated annealing Monte Carlo method was used. Given a set parameter space and an objective or *cost* function that assigns each point in that space a scalar value, a minimising Monte Carlo routine takes an initial point and randomly selects a point nearby. If it is of lower cost then this becomes the initial point for the next run; if it is of higher cost then it calculates whether or not to use this based on the relative cost and a predefined tolerance analogous to temperature. This process is performed a set number of times to calculate numerically the minimum cost function point for that run. The whole routine is then run many times over from many initial positions in parameter space and the lowest cost result is assumed to be the minimum point.

*Annealing* is a metallurgy technique in which a given material is heated above a critical temperature and cooled slowly to alter properties such as hardness and ductility. *Simulated annealing* with a Monte Carlo optimisation routine is a computational technique to improve the accuracy of a Monte Carlo search in a parameter space that has a highly variable cost function surface. The routine is run iteratively - feeding the result from the previous run into the next - and the probability of jumping to a parameter point with a higher cost in a Monte Carlo move is gradually reduced for each successive run.

In this appendix, the numerical routines used to optimise models discussed in this thesis are presented. Note that particularly lengthy expressions have been truncated for neatness.

### A.1 Running the Code

To calculate the results presented in Chapter 5, the following MATLAB script was run:

```matlab
1 initialParametersSkau=[0.14,9.9,-10,13.1, 0.3,10.4,15.7,5.8, ... 12.8, 5.4];
2 initialParametersWu=[0.14,9.9,-10,13.1, 0.3,10.4,15.7,5.8, ... 12.8, 5.4,0, 1.1,0.032];
3 initialParametersSkauExtra=[0.14,9.9,-10,13.1, ... 0.3,10.4,15.7,5.8,12.8, 5.4,0,0, 5.4,0,0,0];
4
5 %need to run this command for matlab to work in parallel.
6 matlabpool
7
8 %time how long the parallelised code takes to run
9 tStart=tic;
10
11 parfor i=1:4
12    if(i==1)
```
ResultsSkau = ...
  OptimiseModelControl('Skau', initialParametersSkau, 'yes')
  a1 = ParameterVariability('Skau', ResultsSkau.parameters)
  ResultsSkau.∆Plus = a1.∆Plus;
  ResultsSkau.∆Minus = a1.∆Minus;
  ResultsSkau.variability = a1.variability;
elseif (i==2)
  ResultsWu = ...
  OptimiseModelControl('Wu', initialParametersWu, 'yes')
  a2 = ParameterVariability('Wu', Wu.parameters)
  ResultsWu.∆Plus = a2.∆Plus;
  ResultsWu.∆Minus = a2.∆Minus;
  ResultsWu.variability = a2.variability;
elseif (i==3)
  ResultsSkauAssg = ...
  OptimiseModelControl('SAGM', initialParametersSkauExtra, 'yes')
  a3 = ...
  ParameterVariability('SAGM', ResultsSkauAssg.parameters)
  ResultsSkauAssg.∆Plus = a3.∆Plus;
  ResultsSkauAssg.∆Minus = a3.∆Minus;
  ResultsSkauAssg.variability = a3.variability;
elseif (i==4)
  ResultsSkauAssgExtra = ...
  OptimiseModelControl('SAGEM', initialParametersSkauExtra, 'yes')
  a4 = ...
  ParameterVariability('SAGEM', ResultsSkauAssgExtra.parameters)
  ResultsSkauAssgExtra.∆Plus = a4.∆Plus;
  ResultsSkauAssgExtra.∆Minus = a4.∆Minus;
  ResultsSkauAssgExtra.variability = a4.variability;
elseif (i==5)
  ResultsSkauAssgExtraForcing = ...
  OptimiseModelControl('SAGEFM', initialParametersSkauExtra, 'yes')
  a4 = ...
  ParameterVariability('SAGEFM', ResultsSkauAssgExtraForcing.parameters)
  ResultsSkauAssgExtraForcing.∆Plus = a4.∆Plus;
  ResultsSkauAssgExtraForcing.∆Minus = a4.∆Minus;
  ResultsSkauAssgExtraForcing.variability = a4.variability;
end
end
tElapsed = toc(tStart)

Running this many times and selecting the lowest cost results gave the following:

% 2 substep
ResultsSkau.cost = 59.421510;
ResultsSkau.parameters = [0.120323 7.975044 -9.891250 ...
 14.920883 1.464954 11.974763 15.802450 5.673358 ...
 6.603029 1.206639];
ResultsSkau.∆Plus = [0.1061 0.9656 0.1112 3.3793 1.8860 ...
 0.1047 0.0573 0.0903 0.5251 0.3125];
A. MATLAB CODE

6 ResultsSkau.variability = [0.2604 4.2022 0.2222 6.0011 24.1272 ...
   0.2588 0.1592 0.1785 1.3015 0.6416];
7
8 ResultsWu.cost = 4.289203;
9 ResultsWu.parameters = [0.142158 6.196856 -9.243228 ...
   16.029214 5.592443 14.636281 7.920319 6.017208 ...
   11.288592 5.111996 4.996000 4.388956 2.452831];
10 ResultsWu.∆Plus = [0.1249 2.4693 0.2115 2.2735 0.1767 0.0795 ...
   0.1831 0.1164 1.1066 0.5043];
11 ResultsWu.∆Minus = [0.1014 4.1978 0.1663 3.7241 0.2077 0.0792 ...
   0.1914 0.1073 0.3957 22.2412 0.1914 0.7860 0.3608];
12
13 ResultsWu.variability = [0.2263 6.6671 0.3779 5.9975 0.3844 ...
   0.1587 0.3745 0.2237 0.7548 33.0515 0.3745 1.8926 0.8651];
14
15 ResultsSkauAssg.cost = 5.104224;
16 ResultsSkauAssg.parameters = [1.032676 7.144348 -9.301711 ...
   14.294687 3.039098 12.825381 12.103438 5.650370 ...
   16.194874 0.802715 7.841641 6.607836 2.303096 ...
   1.345814 4.571505 0.084000];
17 ResultsSkauAssg.∆Plus = [0.0452 0.3619 0.1756 4.0615 ...
   0.4815 0.0862 0.0709 0.1010 0.2347 0.0822 ...
   11.4159 0.0723 0.2381 11.1081 10.8746 0.0853];
18 ResultsSkauAssg.∆Minus = [0.0846 0.3099 0.2103 1.9609 ...
   0.3520 0.0439 0.1337 0.1240 0.1565 0.1251 ...
   22.2412 0.1343 0.1619 22.2412 0.6901 0.1199];
19
20 ResultsSkauAssg.variability = [0.1298 0.6718 0.3859 ...
   6.0224 0.8335 0.1301 0.2046 0.2250 0.3911 ...
   0.2073 33.6571 0.2066 0.4000 33.3493 11.5647 ...
   0.2052];
21
22 ResultsSkauAssgExtra.cost = 1.289902;
23 ResultsSkauAssgExtra.parameters = [3.723154 4.297383 ...
   19.44009 14.548472 6.252743 8.051137 8.358097 ...
   6.047789 13.104478 5.688226 2.158541 10.308806 ...
   6.579233 5.715260 9.001286 0.291936 ];
24 ResultsSkauAssgExtra.∆Plus = [1.2248 1.0237 0.1504 ...
   3.7545 0.0498 3.7367 0.1028 0.0847 0.1926 ...
   0.1545 11.2387 0.1090 0.3681 10.8075 10.9565 ...
   0.0901];
25 ResultsSkauAssgExtra.∆Minus = [0.4909 0.3893 0.2257 ...
   2.2514 0.0479 22.2412 0.1005 0.1299 0.2364 ...
   0.2410 22.2412 0.0976 0.2276 10.7101 0.8121 ...
   0.1513];
26
27 ResultsSkauAssgExtra.variability = [1.7157 1.4130 0.3762 ...
   6.0059 0.0977 25.9779 0.2033 0.2146 0.4291 ...
   0.3955 33.4799 0.2066 0.5957 21.5177 11.7686 ...
   0.2415];
28
29 %2 substep
30 ResultsSkauAssgExtraForcing.cost = 19.299103;
31 ResultsSkauAssgExtraForcing.parameters = [3.110229 6.878378 ...
   -10.170327 13.306720 3.832779 14.724728 ...
   15.152939 5.786497 11.265485 1.975519 1.594382 ...
16.486856  1.234722  5.655274  13.687410  0.110341  

%3 substep
ResultsSkauAssgExtraForcing.cost = 13.383594;
ResultsSkauAssgExtraForcing.parameters = [2.649319  6.787624  ...
 -10.036763  13.724820  4.559759  15.351292  ...
  12.925318  5.817426  12.192726  2.534746  4.687403  ...
  11.071215  2.530185  7.046122  17.858560  0.143797] ;
A. MATLAB CODE

A.2 Bespoke Functions

OptimiseModelControl

```matlab
function OptimisationOutput = ...
    OptimiseModelControl(model, initialParameters, print)

    constants = CalculateConstants();
    % set number of substeps (2 - Uemura et al., 3 - Capello et al.)
    if ((constants.steps==2)||(constants.steps==3))
    else
        error('steps incorrectly defined in call of ...
            OptimiseModelControl')
    end
    % CHOOSE MODEL!!
    % model='Skau' % Skau Only (hydrolysis cycle, futile ...
    % cycling, chemical detachment)
    % model='Wu' % Wu Only (hydro cycle, extra cycles, mech ...
    % detachment)
    % model='SAGM' % Skau with assymmetric gating
    % model='SAGEM' % Skau with asymm. gat. and additional cycle
    % model='SAGEFM' % SAGEFM with focing and slippage
    if(strcmp(model,'Skau')==1)
        noRequiredParameters=10;
        elseif((strcmp(model,'Wu')==1)||(strcmp(model,'WuExtraOpt')==1))
        noRequiredParameters=13; % include beta, ∆3 and ∆8
    elseif(strcmp(model,'SAGM')==1)
        noRequiredParameters=16; % don't include ∆3 and ∆8
    elseif((strcmp(model,'SAGEM')==1)||(strcmp(model,'SkauAssgExtraOpt')==1))
        noRequiredParameters=16; % don't include ∆3 and ∆8
    elseif((strcmp(model,'SAGEFM')==1)||(strcmp(model,'SkauAssgExtraOpt')==1))
        noRequiredParameters=16; % don't include ∆3 and ∆8
        else
            error('model incorrectly defined in call of ...
                OptimiseModelControl')
    end
    if(length(initialParameters)̸=noRequiredParameters)
        error('incorrect number of parameters passed to ...
            OptimiseModelControl')
    end
    if(strcmp(print,'yes')==1)
        printResults=1;
```
```matlab
elseif (strcmp(print,'no') == 1)
    printResults = 0;
else
    error('print defined incorrectly in ...
    OptiniseModelControl. Should be yes or no.')</end

%--------------------------Expect parameters in this ...
% form-------------------------- %

%initialParameters={DGtdw,DGdws,DGdse,DGet, ...
GDDtdw,GDDdws,GDDdse,GDDet, Es, alphaEs,betaEs, ... 
gammaEs/Δ3, alphaSEs/Δ8,betaSEs,omegaSEs} 
%initialParameters=[0.14,9.9,-10,13.1, 0.3,10.4,15.7,5.8, ... 
12.8, 5.4,0,0, 5.4,0,0, 1.1,0.032]; %Skau Results ...
combined with Baker

%initialParametersSkau=[0.14,9.9,-10,13.1, ... 
0.3,10.4,15.7,5.8, 12.8, 5.4];
%initialParametersWu=[0.14,9.9,-10,13.1, ... 
0.3,10.4,15.7,5.8, 12.8, 5.4,0, 1.1,0.032];
%initialParametersSkauExtra=[0.14,9.9,-10,13.1, ... 
0.3,10.4,15.7,5.8,12.8, 5.4,0,0, 5.4,0,0,0];
%initialParametersSkauExtraForcing=[0.14,9.9,-10,13.1, ... 
0.3,10.4,15.7,5.8,12.8, 5.4,0,0, 5.4,0,0,0];

%--------------------------Optimise model
OptimisationOutput = ...
    OptimiseModel(model,initialParameters,constants);
OptimisationOutput.title = strcat(model, ', steps=', ... 
    num2str(constants.steps));
minParameters = OptimisationOutput.parameters;
if (printResults==1)
    PrintAllResults(model,minParameters,constants,'yes')
end
end
```

function constants = CalculateConstants()
    constants.steps=3;
    
    % two sub step sizes – Umeura et al.
    constants.d1=25;  % nm
    constants.d2=11;  % nm
    constants.d3=0;  % nm

    % three sub step sizes – Capello et al.
    if (constants.steps==3)
        constants.d1=23;  % nm
        constants.d2=8;  % nm
        constants.d3=5;  % nm
    end
    constants.d=36;
    
    % tau_p=1*10^(-5); % tau_d
    % tau_P=tau_p*((dP*dP)/(dp*dp));
    % tau_D=tau_p*((DD*dD)/(dp*dp));
    constants.tau = 1*10^(-8);
    constants.invtau = 1/(constants.tau);
    constants.tauD = 1*10^(-5);
    constants.invtauD = 1/(constants.tauD);
    constants.kB = 1.3806505*10^(-23);
    constants.Temperature = 298;
    constants.BJ ...
        = constants.kB * constants.Temperature * 10^21;
    constants.kBT = 1;
    constants.∆ = 2.4;  % nm
    constants.DTOT=13.125;  % Standard state at 298 K
    
    % OPTIMISATION CONSTANTS
    % number of times to run an optimisations for a given run
    constants.numberopts=1;
    % simulated annealing steps
    constants.annealingSteps=150;
    % how many steps to optimise over during each annealing run
    constants.optSteps=2500;
    
    % Parameter variability constants
    constants.εpsilon=1;  % amount the cost function is allowed ...
    % to vary
    constants.minstep=0.000001;  % minimum step size for finding ∆...
constants.maxcount=1000; %1000; %maximum number of iterations to look for the ∆s

end
%Optimisation function – optimises parameters for a given model
%Applies a customised Monte Carlo moves algorithm with simulated annealing
% so a set of parameters and an objective/cost function

function OptimiseModelOutput = ...
    OptimiseModel(model,initialParameters,constants)

    cost = CostFunction(model,initialParameters,constants);
    cost_old = cost;
    mincost = cost;
    parameters = initialParameters;
    minParameters = parameters;
    lengthParameters=length(parameters);

    %simulated Annealing
    for j=1:constants.annealingSteps
        %T=0.298-(0.298/m)*j;
        T=10/j;

        %standard Monte Carlo moves algorithm
        for i=1:constants.optSteps
            %maximum displacement is restricted by Delta_max.
            Delta_max=0.137*sqrt(T);

            %create a new point
            for k=1:lengthParameters
                oldParameters(k)=parameters(k);
                parameters(k)=parameters(k)+(rand-0.5)*Delta_max;
            end

            %Assuming the first four parameters are the energy ...
            % Ensure others are bigger than zero
            for k=5:lengthParameters
                while parameters(k)<0
                    parameters(k)=oldParameters(k)+(rand-0.5)*Delta_max;
                end
            end

            %Assume the first four parameters are the energy ...
            % Ensure the Delta G sum to DTOT the total energy ...
            % available in
            % ATP hydrolysis
            SUM DG=0;
            for k=1:4
                SUM DG=SUM DG+parameters(k);
            end
            for k=1:4
                parameters(k)=parameters(k)*(constants.DTOT)/SUM DG;
            end

end
% calculate the cost function for that point
  cost = CostFunction(model, parameters, constants);
  cost_new = cost;

  % Monte Carlo criteria for accepting the new point
  if (cost_new < cost_old)
    % accept move
    cost_old = cost_new;
  elseif (rand < \exp(-\{(cost\_new-cost\_old)/T\}))
    % accept move
    cost_old = cost_new;
  else
    % reject move
    for k = 1:lengthParameters
      parameters(k) = oldParameters(k);
    end
  end
  end

  cost = CostFunction(model, parameters, constants);
  mincost = cost;
  minParameters = parameters;

  % What is the absolute minimum from all runs? Return this
  OptimisationOutput.mincost = mincost;
  OptimisationOutput.parameters = minParameters;

end
CostFunction

1 %Calculates cost, the degree to which the model results ... 
2 % least squared difference
3
4 function cost = CostFunction(model,parameters,constants)
5
6 % V against fex – Uemura et al. Mechnochemical coupling of ... 
7 % two substeps in a single myosin-V motor Nat. Struct. ... 
8 % Mol. Biol. 11:877–883 2004
9 % L, V vs ATP, Baker et al.
10 cost=0;
11
12 state.fex=0.0;
13 state.cPi=0.1*10^-6;
14 state.cADP=0.1*10^-6;
15 state.cATP=1000*10^-6;
16
17 dynamicProperties = ... 
18 CalculateDynamicProperties(model,parameters,constants,state);
19 least=((dynamicProperties.rl-0.8)/(0.15)); %Delta1
20 cost=cost+least*least;
21 least=((dynamicProperties.v-0.54)/(0.054)); %Delta2
22 cost=cost+least*least;
23
24 least=0.02*((dynamicProperties.vFutile*constants.d)/(dynamicProperties.v*constants.d2)); ... 
25 % Rosenfeld and Sweeny – ADP release from the front head ... 
26 % than that of the rear head
27 least=0.02*(((dynamicProperties.vFutile*constants.d)/(dynamicProperties.v*constants.d2)));
28 %Delta12 %Skau version
29 cost=cost+least*least;
30
31 state.cATP=100*10^-6;
32
33 dynamicProperties = ... 
34 CalculateDynamicProperties(model,parameters,constants,state);
35 least=((dynamicProperties.rl-1.15)/(0.150)); %Delta3
36 cost=cost+least*least;
37
38 state.cATP=10*10^-6;
39
40 dynamicProperties = ... 
41 CalculateDynamicProperties(model,parameters,constants,state);
42 least=((dynamicProperties.v-0.075)/(0.010)); %Delta4
43 cost=cost+least*least;
44
45 state.cADP=200*10^-6;
46 state.cATP=1000*10^-6;
47
48 dynamicProperties = ... 
49 CalculateDynamicProperties(model,parameters,constants,state);
50 least=((dynamicProperties.v-0.32)/(0.032)); %Delta5
51 cost=cost+least*least;
52
53 state.cADP=2500*10^-6;
54 state.cATP=1000*10^-6;
55
56 dynamicProperties = ... 
57 CalculateDynamicProperties(model,parameters,constants,state);
leat=((dynamicProperties.rl-0.4)/(0.150)); %Delta6
   cost=cost+leat*leat;
leat=((dynamicProperties.v-0.13)/(0.013)); %Delta7
   cost=cost+leat*leat;

state.cPi=4000*10^(-6);
state.cADP=0.1*10^(-6);
state.cATP=1000*10^(-6);
dynamicProperties = ...
   CalculateDynamicProperties(model,parameters,constants,state);
leat=((dynamicProperties.rl-0.500)/(0.15)); %Delta8
   cost=cost+leat*leat;
leat=((dynamicProperties.v-0.44)/(0.044)); %Delta9
   cost=cost+leat*leat;

if(strcmp(model,'SAGEFM')==1)
   state.cPi=0.1*10^(-6);
   state.cADP=200*10^(-6);
   state.cATP=1000*10^(-6);
   f0=0.75;
   state.fex=f0/(constants.BJ);
   dynamicProperties = ...
   CalculateDynamicProperties(model,parameters,constants,state);
   cost=cost+leat*leat;
end

%Energy constraints
DG2=parameters(1);
DG3=parameters(2);
DG4=parameters(3);
DG5=parameters(4);

   least=((DG2-2)^2)/(9); %Delta14
   cost=cost+leat*leat;
leat=((DG3-5.7)^2)/(9); %Delta15
   cost=cost+leat*leat;
leat=((DG4+7.7)^2)/(9); %Delta16
   cost=cost+leat*leat;
leat=((DG5-15.3)^2)/(9); %Delta17
   cost=cost+leat*leat;
end
% Uses model to call the required functions to calculate the ...

function dynamicProperties = ...
    CalculateDynamicProperties(model,parameters,constants,state)
    
    if (strcmp(model,'Skau')==1)
        dynamicProperties = ...
            SkauModel(model,parameters,constants,state);
    elseif ((strcmp(model,'Wu')==1)||(strcmp(model,'WuExtraOpt')==1))
        dynamicProperties = ...
            WuModel(model,parameters,constants,state);
    elseif (strcmp(model,'SAGM')==1)
        dynamicProperties = ...
            SkauModel(model,parameters,constants,state);
    elseif ((strcmp(model,'SAGEM') ==1)||(strcmp(model,'SkauAssgExtraOpt')==1))
        dynamicProperties = ...
            SkauAssgExtraModel(model,parameters,constants,state);
    elseif ((strcmp(model,'SAGEFM') ==1)||(strcmp(model,'SkauAssgExtraOpt')==1))
        dynamicProperties = ...
            SkauAssgExtraModel(model,parameters,constants,state);
    elseif (strcmp(model,'OriginalWu')==1)
        dynamicProperties = OriginalWuModel(state);
    elseif (strcmp(model,'OriginalBierbaum')==1)
        dynamicProperties = BierbaumOriginalModel(constants,state);
    else
        error('model incorrectly defined in call of ...
            CalculateDynamicProperties')
    end

end
function dynamicProperties = OriginalWuModel(state)

cATP = state.cATP * 10^6;
cADP = state.cADP * 10^6;

%——Calculate the reaction rates——

d = 0.036; % mum

% termination rates taken from Wu paper (who got them from ...
    Baker)
kterm1 = 0.032; % term1
kterm2 = 1.1; % term2

% rates
k1 = 12;
k1prime = 30;
kmin1 = cADP * 4.5;
kmin1prime = kmin1;

k2 = cATP * 0.9;
k2prime = k2;
k3 = 870;
k3prime = 200;

k4 = 166;
k5 = 15;
k6 = 4;
k7 = 200;

% probabilities
x = zeros(8);

B = kmin1 + k6 + k2;
x(2) = 1 / ( ((k5*B + k1*k6)/(k1)) * ... 
    ((k2prime + kmin1prime)/(k1prime*k2prime) + (1/k2prime) + ... 
    (1/k3prime)) + k2*((1/k3) + (1/k4) + (1/k7)) - ... 
    (k6/k1prime) + (B/k1) + 1 ); % EDw

x(1) = (B/k1)*x(2); % DDw
x(3) = (k2/k3)*x(2); % TDw
x(4) = (k2/k4)*x(2); % T'Dw
x(5) = (k2/k7)*x(2); % T'Ds
x(6) = (((k5*B + k1*k6)/(k1)) * ... 
    ((k2prime + kmin1prime)/(k1prime*k2prime)) - ... 
    k6/k1prime)*x(2); % DDS
x(7) = ((k5*B + k1*k6)/(k1*k2prime))*x(2); % EDs
x(8) = ((k5*B + k1*k6)/(k1*k3prime))*x(7); % TDs

% calculate run lengths and velocities
% Wu version
dynamicProperties.v = (k2prime*x(7)+k2*x(2))*d;
vterm = (kterm1*x(4)+kterm2*x(1));
dynamicProperties.rl = dynamicProperties.v / vterm;
dynamicProperties.x = x;
dynamicProperties.vFutile = 0;
end
% Calculates the dynamic properties for the Wu model
function dynamicProperties = ...

    % Calculate the reaction rates
    rates = CalculateRates(model, parameters, constants, state);

    u = rates.u;
    w = rates.w;
    \Delta^3 = rates.\Delta^3;
    \Delta^8 = rates.\Delta^8;

    % reaction rate matrix
    M = [
        -(u(1) + w(5) + w(10)), w(1), 0, 0, u(5), 0, u(10),
        u(1), -(u(2) + w(1)), w(2), 0, 0, 0,
        0, u(2), -(u(3) + u(8) + w(2) + \Delta^3), w(3), 0, w(8), 0,
        0, 0, u(3), -(u(4) + w(3)), w(4), 0, 0,
        w(5), 0, 0, u(4), -(u(5) + w(4) + w(11)), u(11), 0,
        0, 0, u(8), 0, w(11), -(u(9) + w(8) + u(11)), w(9),
        w(10), 0, 0, 0, u(9), -(u(10) + w(9) + \Delta^8)
    ];

    % eigenvalues and vectors
    MT = transpose(M);
    [eigvec, eigval] = eig(MT);
    maxeigval = -9999;
    for i = 1:7
        maxeigvaltemp = max(real(eigval(i, i)));
        if (isreal(eigval(i, i)) == 1)
            maxeigvaltemp = max(eigval(i, i));
            if (maxeigvaltemp > maxeigval)
                maxeigval = maxeigvaltemp;
                maxeigvec = eigvec(:, i);
            end
        end
    end
    % have largest eigenvalue and corresponding eigenvector
    evec = maxeigvec;

    % renormalise the system
    u(1) = u(1) * evec(2) / evec(1); % 12
    u(2) = u(2) * evec(3) / evec(2); % 23
    u(3) = u(3) * evec(4) / evec(3); % 34
    u(4) = u(4) * evec(5) / evec(4); % 45
    u(5) = u(5) * evec(1) / evec(5); % u51
A. **MATLAB CODE**

51 \[ u(8) = u(8) \cdot evec(6) / evec(3); \quad \% u37 \]
52 \[ u(9) = u(9) \cdot evec(7) / evec(6); \quad \% u78 \]
53 \[ u(10) = u(10) \cdot evec(1) / evec(7); \quad \% u81 \]
54 \[ u(11) = u(11) \cdot evec(5) / evec(6); \quad \% u75 \]
55
56 \[ w(1) = w(1) \cdot evec(1) / evec(2); \quad \% w21 \]
57 \[ w(2) = w(2) \cdot evec(2) / evec(3); \quad \% w32 \]
58 \[ w(3) = w(3) \cdot evec(3) / evec(4); \quad \% w43 \]
59 \[ w(4) = w(4) \cdot evec(4) / evec(5); \quad \% w54 \]
60 \[ w(5) = w(5) \cdot evec(5) / evec(1); \quad \% w15 \]
61
62 \[ w(8) = w(8) \cdot evec(3) / evec(6); \quad \% w73 \]
63 \[ w(9) = w(9) \cdot evec(6) / evec(7); \quad \% w87 \]
64 \[ w(10) = w(10) \cdot evec(7) / evec(1); \quad \% w18 \]
65 \[ w(11) = w(11) \cdot evec(6) / evec(5); \quad \% w57 \]
66
67 \[ \Delta 9 = 0; \quad \% \text{detachment from state 9} \]
68 \[ \Delta 3 = 0; \quad \% \text{detachment from state 3} \]
69 \[ \Delta 8 = 0; \quad \% \text{detachment from state 8} \]
70
71 %-------------------steady state solutions from Maple---------------------
72 \[ y(1) = \text{TRUNCATED DUE TO LENGTH} \]
73 \[ y(2) = \text{TRUNCATED DUE TO LENGTH} \]
74 \[ y(3) = \text{TRUNCATED DUE TO LENGTH} \]
75 \[ y(4) = \text{TRUNCATED DUE TO LENGTH} \]
76 \[ y(6) = 0; \]
77 \[ y(7) = \text{TRUNCATED DUE TO LENGTH} \]
78 \[ y(8) = \text{TRUNCATED DUE TO LENGTH} \]
79 \[ y(9) = 0; \]
80
81 \[ \text{invSUM} = 1 / (\text{sum(sum(y))}); \]
82 \[ x = y \cdot \text{invSUM}; \]
83
84 \[ \text{dynamicProperties.x} = x; \]
85 % Velocity and convert to the correct units
86 \[ \text{dynamicProperties.v} = (\text{constants.d}) \cdot (x(1) \cdot u(1) - x(2) \cdot w(1)) / 1000; \]
87 \[ \text{dynamicProperties.vFutile} = \ldots \]
88 \[ (\text{constants.d2}) \cdot (x(6) \cdot u(7) - x(2) \cdot w(7)) / 1000; \quad \% \text{Skau version} \]
89
90 % runlength µ m
91 \[ \text{dynamicProperties.rl} = (\text{dynamicProperties.v}) / (-\text{maxeigval}); \]
92
93 \[ \text{dynamicProperties.rates} = \text{rates}; \]
94 \[ \text{end} \]
CalculateRates

% Calculates the reaction rates for a given model

function rates = CalculateRates(model,parameters,constants,state)

% parameters=[DGtdw,DGdwds,DGdse,DGet, ...
  GDDtdw,GDDdwds,GDDdse,GDDet, Es, alphaEs,betaEs, ...
  gammaEs/△3, alphaSEs/△8, betaSEs, gammaSEs, omegaSEs]

% PULL OUT PARAMETERS
DGtdw = parameters(1);
DGdwds = parameters(2);
DGdse = parameters(3);
DGet = parameters(4);
GDDtdw = parameters(5);
GDDdwds = parameters(6);
GDDdse = parameters(7);
GDDet = parameters(8);
Es = parameters(9);
alphaEs = parameters(10);

% DEFINE CONSTANTS
kH = 2*Es/(constants.d1*constants.d1);
b = (0.5*kH*constants.d2*constants.d2)/Es;
fex = state.fex;
invtauD = constants.invtauD;
invtau = constants.invtau;
d1 = constants.d1;
d2 = constants.d2;
spring = 0; % 0.5*fex/kH;
spring1 = 0.5*fex/kH;

% DEFINE RATES
% Mechanical Movement Rate - Skau
rates.u(1) = invtauD; % u12
rates.w(1) = invtauD * exp(-(Es-fex*(d1-spring))); % w21
rates.u(2) = invtauD * exp(-(GDDtdw+fex*(d2+spring)+b*Es)); ...
% u23
rates.w(2) = invtauD * exp(-(GDDtdw+DGet)); % w32

% Both Heads Attached - Skau
rates.u(3) = invtau*exp(-GDDdwds); % u34
rates.w(3) = ...
state.cPi*invtau*exp(-GDDdwds+DGdwds-(1-b)*Es)); % w43
rates.u(4) = invtau*exp(-GDDdse)); % u45
rates.w(4) = state.cADP*invtau*exp(-(GDDdse+DGdse)); % w54
rates.u(5) = state.cATP*invtau*exp(-(GDDet)); % u51
rates.w(5) = invtau*exp(-(GDDet+DGet)); % w15
if(strcmp(model,'Skau')==1) || (strcmp(model,'SAGM')==1))

% Futile Cycle - Skau
rates.u(6)=rates.u(4)*exp(-(alphaEs)); % u46
rates.w(6)=rates.w(4)*exp(-(alphaEs)); % w64
rates.u(7)=rates.u(5); % u62
rates.w(7)=rates.w(5)*exp(-Es-fex*(d2+spring)); % w26

% Extra Cycle - Wu % % OFF % %
rates.u(8)=0; % u37
rates.w(8)=0; % w73
rates.u(9)=9999; % u78
rates.w(9)=0; % w87
rates.u(10)=9999; % u81
rates.w(10)=0; % w18
rates.u(11)=0; % u75
rates.w(11)=0; % w57

% Chemical Detachment - Skau
rates.u(12)=rates.u(4); % u29
rates.w(12)=rates.w(4); % w92
rates.∆9=rates.u(5); %*exp(abs(fex)*constants.∆); ...
% detachment from state 9

% Mechanical Detachment - Wu % % OFF % %
rates.∆3=0; % detachment from state 3
rates.∆8=0; % detachment from state 8

% Extra pre-detachment state rate % % OFF % %
rates.u(13)=0; % u97
rates.w(13)=0; % w79

if(strcmp(model,'SAGM')==1)
% PULL OUT PARAMETERS
gammaEs=parameters(12);
alphaSEs=parameters(13);
gammaSEs=parameters(15);
omegaSEs=parameters(16);

% Assymetric gating
rates.w(6)=rates.w(4)*exp(-(alphaSEs)); % w64
% rates.w(8)=rates.w(4)*exp(-(betaSEs)); % w73
rates.u(12)=rates.u(12)*exp(-(gammaEs)); % w29
rates.w(12)=rates.w(4)*exp(-(gammaSEs)); % w92
rates.w(4) = rates.w(4)*exp(-(omegaSEs)); % w54
end

elseif((strcmp(model,'Wu')==1)||(strcmp(model,'WuExtraOpt')==1))
% PULL OUT PARAMETERS
betaEs=parameters(11);
∆3=parameters(12);
∆8=parameters(13);
%Futile Cycle - Skau
rates.u(6)=0; \%u46
rates.w(6)=0; \%w64
rates.u(7)=9999; \%u62
rates.w(7)=0; \%w26

%Extra Cycle - Wu
rates.u(8)=rates.u(4)*exp(-\(\beta_{Es}\)); \%u37
rates.w(8)=rates.w(4)*exp(-\(\beta_{Es}\)); \%w73
rates.u(9)=rates.u(5); \%u78
\text{ASSUME STRAIN HAS ...}
%NO PART TO PLAY IN ATP BINDING
rates.w(9)=rates.w(5); \%w87
\text{ASSUME STRAIN HAS NO PART ...}
%TO PLAY IN ATP BINDING
rates.u(10)=rates.u(3); \%u81
\text{ASSUME STRAIN HAS NO PART ...}
%TO PLAY IN Pi RELEASE
rates.w(10)=rates.w(3); \%w18
rates.u(11)=rates.u(3); \%u75
rates.w(11)=rates.w(3); \%w57

%Chemical Detachment - Skau
rates.u(12)=0; \%u29
rates.w(12)=9999; \%w92
\Delta9=0; \%detachment from state 9

%Mechanical Detachment - Wu
rates.\Delta3=\Delta3; \%detachment from state 3
rates.\Delta8=\Delta8; \%detachment from state 8

%Extra pre-detachment state rate
\text{\%OFF \% \%}
rates.u(13)=0; \%u97
rates.w(13)=0; \%w79

elseif((strcmp(model,'SAGEM')==1)||(strcmp(model,'SAGEFM')==1))
\text{\%PULL OUT PARAMETERS}
betaEs=parameters(11);
gammaEs=parameters(12);
alphaSEs=parameters(13);
betaSEs=parameters(14);
gammaSEs=parameters(15);
omegaSEs=parameters(16);

%Futile Cycle - Skau
rates.u(6)=rates.u(4)*exp(-\(\alpha_{Es}\)); \%u46
rates.w(6)=rates.w(4)*exp(-\(\alpha_{Es}\)); \%w64
rates.u(7)=rates.u(5); \%u62
rates.w(7)=rates.w(5)*exp(-\(Es-fex*\(d2\+spring\))); \%w26

%Extra Cycle - Wu
rates.u(8)=rates.u(4)*exp(-\(\beta_{Es}\)); \%u37
rate.w(8) = rate.w(4) * exp(-betaEs);  % w73
rate.u(9) = 0;  % u78 ASSUME STRAIN HAS NO PART TO PLAY IN ...

% ATP BINDING
rate.w(9) = 0;  % w73  % state 8 is off
rate.u(10) = 9999;  % u81  % state 8 is off
rate.w(10) = 0;  % w78  % state 8 is off
rate.u(11) = rate.u(3);  % u75
rate.w(11) = rate.w(3);  % w57

% Chemical Detachment - Skau
rate.u(12) = rate.u(4);  % u29
rate.w(12) = rate.w(4);  % w92
rate.d9 = rate.u(5);  % exp(abs(fex)*constants.d); ...
% detachment from state 9

% Mechanical Detachment - Wu  % % OFF  % %
rate.d3 = 0;  % detachment from state 3
rate.d8 = 0;  % detachment from state 8

% Extra pre-detachment state rate
rate.u(13) = rate.u(2);  % u97
rate.w(13) = rate.w(2);  % w79

% Assymetric gating
rate.w(6) = rate.w(4) * exp(-alphaEs);  % w64
rate.w(8) = rate.w(4) * exp(-betaEs);  % w73
rate.u(12) = rate.u(12) * exp(-gammaEs);  % w29
rate.w(12) = rate.w(4) * exp(-gammaEs);  % w92
rate.w(4) = rate.w(4) * exp(-omegaEs);  % w54

% implement modified slipping mechanism from Bierbaum ...
and Lipowski (2011)
if(strcmp(model,'SAGEFM')==1)
    D = 470;  % chosen by the authors
    Uba = 20;  % chosen by the authors
    rate.w(14) = (D *((state.fex)*constants.d - ... 
        Uba) / (constants.d*constants.d) / (1 - ... 
        exp((Uba - (state.fex)*constants.d))) );
    rate.u(14) = rate.w(14) * exp(-constants.d*(state.fex));

    % force dependence on nucleotide release mechanism ...
    % from Bierbaum and Lipowski (2011)
    chi = 4;
    d = d1 + d2;
    Fchem = 1.6;
    Fprime = 1.6;
    Fchem = (1 + exp(-chi*d*Fprime)) / (1 + exp(chi*d*(state.fex - Fprime)))
    rate.w(4) = rate.w(4) * Fchem;
    rate.u(5) = rate.u(5) * Fchem;
else
    rate.w(14) = 0;
    rate.u(14) = 0;
end
else
error('model incorrectly defined in call of ...
CalculateRates')
end
end
SkauModel

```matlab
function dynamicProperties = ...
    SkauModel(model,parameters,constants,state)
    %-------Calculate the reaction rates-------
    rates = CalculateRates(model,parameters,constants,state);
    u = rates.u;
    w = rates.w;
    Delta9=rates.Delta9;

    %--------reaction rate matrix--------
    M = [ ...
        -(u(1) + w(5)),w(1),0,0,u(5),0,0,
        u(1),-(u(2)+u(12)+w(1)+w(7)),w(2),0,0,u(7),w(12),
        0,u(2),-(u(3) + w(2)),w(3),0,0,0,
        0,0,u(3),-(u(4) + u(6) + w(3)),w(4),w(6),0,
        w(5),0,0,u(4),-( u(5) + w(4) ),0,0,
        0,w(7),0,u(6),0,-(u(7) + w(6)),0,
        0,u(12),0,0,0,0,-( w(12) + Delta9 ) % ];
    %ignore states 7 and 8 so 7==9
    MT=transpose(M);
    [eigvec,eigval]=eig(MT);
    maxeigval=-9999;
    for i=1:7
        maxeigvaltemp=max(real(eigval(i,i)));
        if (isreal(eigval(i,i))==1)
            maxeigvaltemp=max(eigval(i,i));
            if (maxeigvaltemp>maxeigval)
                maxeigval=maxeigvaltemp;
                maxeigvec=eigvec(:,i);
            end
            %
        else
            % error('eigenvalues not real!')
        end
    end
    %have largest eigenvalue and corresponding eigenvector
    evec=maxeigvec;

    %--------renormalise the system--------
    u(1)=u(1)*evec(2)/evec(1); %12
    u(2)=u(2)*evec(3)/evec(2); %23
    u(3)=u(3)*evec(4)/evec(3); %34
```

u(4) = u(4) * evec(5) / evec(4);  % 45
u(5) = u(5) * evec(1) / evec(5);  % u51
u(6) = u(6) * evec(6) / evec(4);  % u46
u(7) = u(7) * evec(2) / evec(6);  % u62
u(12) = u(12) * evec(7) / evec(2);  % 29, == 9

w(1) = w(1) * evec(1) / evec(2);  % w21
w(2) = w(2) * evec(2) / evec(3);  % w32
w(3) = w(3) * evec(3) / evec(4);  % w43
w(4) = w(4) * evec(4) / evec(5);  % w54
w(5) = w(5) * evec(5) / evec(1);  % w15
w(6) = w(6) * evec(4) / evec(6);  % w64
w(7) = w(7) * evec(6) / evec(2);  % w26
w(12) = w(12) * evec(2) / evec(7);  % w92, == 9

Δ9 = 0;  % detachment from state 9
Δ3 = 0;  % detachment from state 3
Δ8 = 0;  % detachment from state 8

% steady state solutions from Maple
y(1) = TRUNCATED DUE TO LENGTH
y(2) = TRUNCATED DUE TO LENGTH
y(3) = TRUNCATED DUE TO LENGTH
y(4) = TRUNCATED DUE TO LENGTH
y(5) = TRUNCATED DUE TO LENGTH
y(6) = TRUNCATED DUE TO LENGTH
y(7) = 0;
y(8) = 0;
y(9) = (u(12) * (y(2) / w(12)));

invSUM = 1 / (sum(sum(y)));
x = y * invSUM;
dynamicProperties.x = x;
% Velocity and convert to the correct units
dynamicProperties.v ...
   = (constants.d) * (x(1) * u(1) - x(2) * w(1)) / 1000;
dynamicProperties.vFutile ...
   = (constants.d2) * (x(6) * u(7) - x(2) * w(7)) / 1000;  % Skau version
%dynamicProperties.detachFlux = x(9) * rates.Δ9;

% runlength \mu m
dynamicProperties.rl = (dynamicProperties.v / (-maxeigval));
dynamicProperties.rates = rates;
end
SkauAssgExtraModel

```matlab
function dynamicProperties = ...
SkauAGxtrPathModel(model,parameters,constants,state)

% Skau system with alternative gating and added state 7 and a ...
% rate 97 and 79
rate 97 and 79
rates = CalculateRates(model,parameters,constants,state);

u = rates.u;
w = rates.w;
\[ A \delta \theta = rates.A \delta \theta \]

% Calculate the reaction rates
rates = CalculateRates(model,parameters,constants,state);

u(1), -(u(1) + w(5)), w(1), 0, 0, u(5), 0, 0, 0
u(1), -(u(2) + u(12) + w(1) + w(7)), w(2), 0, 0, u(7), 0, w(12)
0, w(12), 0, u(3) + w(2) + u(8)), w(3), 0, 0, w(8), 0
0, 0, u(3), -(u(4) + u(6) + w(3)), w(4), w(6), 0, 0
w(5), 0, 0, u(4), -(u(5) + w(4) + w(11)), 0, u(11), 0
0, w(7), 0, u(6), 0, -(u(7) + w(6)), 0, 0
0, 0, u(8), 0, w(11), 0, -(w(8) + u(11) + w(13)), u(13)
0, u(12), 0, 0, 0, 0, w(13), -(u(13) + w(12) + A \delta \theta)

% Ignore state 8 so 8 == 9
M = [
  u(1), -(u(1) + w(5)), w(1), 0, 0, u(5), 0, 0, 0
  u(1), -(u(2) + u(12) + w(1) + w(7)), w(2), 0, 0, u(7), 0, w(12)
  0, w(12), 0, u(3) + w(2) + u(8)), w(3), 0, 0, w(8), 0
  0, 0, u(3), -(u(4) + u(6) + w(3)), w(4), w(6), 0, 0
  w(5), 0, 0, u(4), -(u(5) + w(4) + w(11)), 0, u(11), 0
  0, w(7), 0, u(6), 0, -(u(7) + w(6)), 0, 0
  0, 0, u(8), 0, w(11), 0, -(w(8) + u(11) + w(13)), u(13)
  0, u(12), 0, 0, 0, 0, w(13), -(u(13) + w(12) + A \delta \theta)
];

% Calculate the eigen values and vectors from the ...
% transposed reaction rate matrix
MT = transpose(M);
[eigvec, eigval] = eig(MT);
maxeigval = -9999;
for i = 1:8
  maxeigvaltemp = max(real(eigval(i, i)));
  if (isreal(eigval(i, i)) == 1)
    maxeigvaltemp = max(eigval(i, i));
    if (maxeigvaltemp > maxeigval)
      maxeigval = maxeigvaltemp;
      maxeigvec = eigvec(:, i);
    end
  end
end
% Have largest eigenvalue and corresponding eigenvector

evec = maxeigvec;

% Renormalise the system
u(1) = u(1) * evec(2) / evec(1); % 12
u(2) = u(2) * evec(3) / evec(2); % 23
u(3) = u(3) * evec(4) / evec(3); % 34
u(4) = u(4) * evec(5) / evec(4); % 45
```
u(5) = u(5) * evec(1) / evec(5);  \% u51
u(6) = u(6) * evec(6) / evec(4);  \% u46
u(7) = u(7) * evec(2) / evec(6);  \% u62
u(12) = u(12) * evec(8) / evec(2);  \% 29, 8==9
u(13) = u(13) * evec(7) / evec(8);  \% 97

w(1) = w(1) * evec(1) / evec(2);  \% w21
w(2) = w(2) * evec(2) / evec(3);  \% w32
w(3) = w(3) * evec(3) / evec(4);  \% w43
w(4) = w(4) * evec(4) / evec(5);  \% w54
w(5) = w(5) * evec(5) / evec(1);  \% w15
w(6) = w(6) * evec(4) / evec(6);  \% w64
w(7) = w(7) * evec(6) / evec(2);  \% w26
w(12) = w(12) * evec(8) / evec(2);  \% w92, 8==9
w(13) = w(13) * evec(8) / evec(7);  \% 79

\Delta 9 = 0;  \% detachment from state 9

%--------------------------------- steady state solutions from Maple ---------------------------------
y(1) = TRUNCATED DUE TO LENGTH
y(2) = TRUNCATED DUE TO LENGTH
y(3) = TRUNCATED DUE TO LENGTH
y(4) = TRUNCATED DUE TO LENGTH
y(5) = TRUNCATED DUE TO LENGTH
y(6) = TRUNCATED DUE TO LENGTH
y(7) = TRUNCATED DUE TO LENGTH
y(8) = 0;
y(9) = TRUNCATED DUE TO LENGTH

invSUM = 1 / (sum(sum(y)));
x = x * invSUM;
dynamicProperties.x = x;

\% Velocity and convert to the correct units
\%  \% or 9
dynamicProperties.v = (constants.d) * (x(1) * u(1) - x(2) * w(1) + ...
  (u(14) - w(14)) * (x(2) + x(9))) / 1000;  \%slip from states 2 ...
dynamicProperties.vFutile = (constants.d2) * (x(6) * u(7) - x(2) * w(7)) / 1000; ...  
\% Skau version

\%runlength \mu m
dynamicProperties.rl = (dynamicProperties.v / (-maxeigval));

\% Additional Js
dynamicProperties.J45 = (u(4) * x(4) - w(4) * x(5)) / 1000;
dynamicProperties.J37 = (u(8) * x(3) - w(8) * x(7)) / 1000;
dynamicProperties.J29 = (u(12) * x(2) - w(12) * x(9)) / 1000;
dynamicProperties.Jdetach = (rates.\Delta 9 * x(9)) / 1000;
dynamicProperties.Jslip = (u(14) - w(14)) * (x(2) + x(9)) / 1000;
dynamicProperties.rates = rates;
ed
BierbaumOriginalModel

```matlab
function dynamicProperties = BierbaumOriginalModel(constants,state)

theta=0.65;
D=470;
Uba=20;
d=36;
chi=4;
Fprime=1.6/constants.BJ;
Fchem=(1+exp(-chi*d*Fprime))/(1+exp(chi*d*(fex-Fprime)));
FmechFor=exp(-theta*d*fex);
FmechBack=exp((1-theta)*d*fex);

rates.u(1)=12; %u12
rates.w(1)=cADP*4.5; %w21
rates.u(2)=cATP*0.9; %u23
rates.w(2)=0.00002; %w32
rates.u(3)=7000*FmechFor; %u34
rates.w(3)=0.65*FmechBack; %w43
rates.u(4)=250; %u41
rates.w(4)=cPi*0.65; %w14
rates.u(5)=1.2; %u25
rates.w(5)=cADP*4.5*Fchem; %w52
rates.u(6)=cATP*0.9*Fchem; %u56
rates.w(6)=0.00002; %w65
rates.u(7)=250; %u62
rates.w(7)=cPi*0.0000006; %w26
rates.w(8)=((D*(fex*d - Uba))/(d*d)) / { 1 - ... \exp((Uba-fex*d)) }; %u5'5
rates.u(8)=rates.w(8)*exp(-d*fex); %u55'
rates.\Delta t=0.4;

u = rates.u;
w = rates.w;
\Delta t=rates.\Delta t;

%---steady state solutions Cij format---
cycle1=u(3)*u(4)*u(1) + w(2)*u(4)*u(1) + w(2)*w(3)*u(1) + ... 
w(2)*w(3)*w(4); 

cycle2=u(7)*u(6)+w(5)*u(7)+w(5)*w(6); 
```

A. MATLAB CODE

1. function dynamicProperties = BierbaumOriginalModel(constants,state)
2. theta=0.65;
3. D=470;
4. Uba=20;
5. d=36;
6. chi=4;
7. Fprime=1.6/constants.BJ;
8. Fchem=(1+exp(-chi*d*Fprime))/(1+exp(chi*d*(fex-Fprime)));
9. FmechFor=exp(-theta*d*fex);
10. FmechBack=exp((1-theta)*d*fex);
11. rates.u(1)=12; %u12
12. rates.w(1)=cADP*4.5; %w21
13. rates.u(2)=cATP*0.9; %u23
14. rates.w(2)=0.00002; %w32
15. rates.u(3)=7000*FmechFor; %u34
16. rates.w(3)=0.65*FmechBack; %w43
17. rates.u(4)=250; %u41
18. rates.w(4)=cPi*0.65; %w14
19. rates.u(5)=1.2; %u25
20. rates.w(5)=cADP*4.5*Fchem; %w52
21. rates.u(6)=cATP*0.9*Fchem; %u56
22. rates.w(6)=0.00002; %w65
23. rates.u(7)=250; %u62
24. rates.w(7)=cPi*0.0000006; %w26
25. rates.w(8)=((D*(fex*d - Uba))/(d*d)) / { 1 - ... \exp((Uba-fex*d)) }; %u5'5
26. rates.u(8)=rates.w(8)*exp(-d*fex); %u55'
27. rates.\Delta t=0.4;
28. u = rates.u;
29. w = rates.w;
30. \Delta t=rates.\Delta t;
31. cycle1=u(3)*u(4)*u(1) + w(2)*u(4)*u(1) + w(2)*w(3)*u(1) + ... 
w(2)*w(3)*w(4); 
32. cycle2=u(7)*u(6)+w(5)*u(7)+w(5)*w(6); 

\[ y(1) = \left( u(2) * u(3) * u(4) + w(1) * (u(3) * u(4) + w(2) * u(4) + \ldots \right) \) * cycle2; \%
\[ y(2) = cycle1 * cycle2; \%
\[ y(3) = \left( u(2) * (u(4) * u(1) + w(3) * u(1) + w(3) * w(4)) + \ldots \right) \) * cycle2; \%
\[ y(4) = u(2) * u(3) * (u(1) + w(4)) * cycle2 + \ldots \)
\[ y(5) = u(5) * (u(7) + w(6)) * cycle1 + w(7) * w(6) * cycle1; \%
\[ y(6) = u(5) * u(6) * cycle1 + w(7) * (u(6) + w(5)) * cycle1; \%
\]
\]
\[ invSUM = 1 / \{ sum(y) \}; \]
\[ x = y * invSUM; \]
\[ dynamicProperties.x = x; \]
\[ % Velocity and convert to the correct units \]
\[ dynamicProperties.v = \left( \text{constants.d} * \ldots \right) \left( (x(3) * u(3) - x(4) * w(3)) + (u(8) - w(8)) * x(5) \right) / 1000; \]
\[ dynamicProperties.vFutile = 0; \]
\[ % runlength \mu m \]
\[ dynamicProperties.rl = dynamicProperties.v / \{ rates.\Delta1 * x(1) \}; \]
\[ dynamicProperties.rates = rates; \]
\[ end \]
%returns the sensitivity of the parameters with regards to the costfunction

function parameterResults = ParameterVariability(model,parameters)

if(strcmp(model,'Skau')==1)
   noRequiredParameters=10;
elseif(strcmp(model,'Wu')==1)
   noRequiredParameters=13;  % include beta, Δ3 and Δ8
elseif(strcmp(model,'SkauAssg')==1)
   noRequiredParameters=16;  % don't include Δ3 and Δ8
elseif(strcmp(model,'SkauAssgExtra')==1)
   noRequiredParameters=16;  % don't include Δ3 and Δ8
else
   error('model incorrectly defined in call of ...ParameterVariability')
end

if(length(parameters)~=noRequiredParameters)
   error('incorrect number of parameters passed to ...ParameterVariability')
end

constants = CalculateConstants();

minCost = CostFunction(model,parameters,constants);

% preallocation of memory
ΔPlus=parameters;
ΔMinus=parameters;

% not an efficient method but works for now
for i = 1:noRequiredParameters
   maxParameters=parameters;
   minParameters=parameters;
   for div = 1:10
      % start with a big step size and gradually get smaller
      step = constants.minstep*exp(10/div);
      cost=minCost;
      count=0;
      while((cost<minCost+constants.epsilon) && ...
         (count<constants.maxcount))
         % contains a random element – must be run several times to understand results
         maxParameters(i) = maxParameters(i)+rand*step;
      end
   end
end
```matlab
cost = CostFunction(model, maxParameters, constants);
count = count+1;
end

cost = minCost;
count = 0;

while((cost <= minCost + constants.epsilon) & ... (count <= constants.maxcount))
    minParameters(i) = minParameters(i) - step;
    cost = CostFunction(model, minParameters, constants);
    count = count+1;
end

% counter running through the boundary
maxParameters(i) = maxParameters(i) - step;
minParameters(i) = minParameters(i) + step;
end

% output the Δs
ΔPlus(i) = maxParameters(i) - parameters(i);
ΔMinus(i) = parameters(i) - minParameters(i);
end

parameterResults.parameters = parameters;
parameterResults.ΔPlus = ΔPlus;
parameterResults.ΔMinus = ΔMinus;
parameterResults.variability = ΔPlus + ΔMinus;
end
```
A. MATLAB CODE

A.3 Printing the Visuals

PrintAllResults

```matlab
function PrintAllResults(model,parameters,constants,separate)

n=500;
state.fex=0.0;
state.cPi=0.1*10^(-6);
state.cADP=0.1*10^(-6);
state.cATP=1000*10^(-6);

forceFrom=-5;
forceTo=5;

%Plot Graphs - generate data files
%Forcing vs...
for cATP=[10*10^(-6),1000*10^(-6)]
    for cADP=[1*10^(-6),200*10^(-6)]
        for j=1:n
            state.cATP=cATP;
            state.cADP=cADP;

            f0=(j/n)*(forceTo-forceFrom) + forceFrom;
            state.fex=f0/parameters.BJ;

            dynamicProperties = ...
            CalculateDynamicProperties(model,parameters,constants,state);
            x=dynamicProperties.x;
            rl=dynamicProperties.rl;
            v=dynamicProperties.v;
            u=dynamicProperties.rates.u;
            w=dynamicProperties.rates.w;

            %Do for each set of concentrations
            if((state.cADP==1*10^(-6))&&(state.cATP==1000*10^(-6)))
                resultsForcing.D1.T1000(j,1)=rl;
                resultsForcing.D1.T1000(j,2)=v;
            elseif((state.cADP==1*10^(-6))&&(state.cATP==10^(-6)))
                resultsForcing.D1.T10(j,1)=rl;
                resultsForcing.D1.T10(j,2)=v;
            elseif((state.cADP==200*10^(-6))&&(state.cATP==1000*10^(-6)))
                resultsForcing.D200.T1000(j,1)=rl;
                resultsForcing.D200.T1000(j,2)=v;
            end
        end
    end
end
```
% Quantities against ADP
for cATP=[10*10^(-6), 500*10^(-6), 1000*10^(-6)]
    for j=1:n
        state.cATP=cATP;
        state.fex=0.0;
        state.cADP=j*(12.5)*10^(-6);
        
        dynamicProperties = ...
        CalculateDynamicProperties(model, parameters, constants, state);
        x=dynamicProperties.x;
        rl=dynamicProperties.rl;
        v=dynamicProperties.v;
        u=dynamicProperties.rates.u;
        w=dynamicProperties.rates.w;
        
        if ((state.cATP==1000*10^(-6)))
            resultsADP.T1000(j,1)=rl;
            resultsADP.T1000(j,2)=v;
        elseif ((state.cATP==500*10^(-6)))
            resultsADP.T500(j,1)=rl;
            resultsADP.T500(j,2)=v;
        elseif ((state.cATP==10*10^(-6)))
            resultsADP.T10(j,1)=rl;
            resultsADP.T10(j,2)=v;
        end
    end
end

% against ATP
for cADP=[10*10^(-6), 100*10^(-6), 1000*10^(-6)]
    for j=1:n
        state.cATP=cATP;
        state.cADP=cADP;
        state.fex=0.0;
        state.cATP=j*(12.5)*10^(-6);
        
        dynamicProperties = ...
        CalculateDynamicProperties(model, parameters, constants, state);
        x=dynamicProperties.x;
        rl=dynamicProperties.rl;
        v=dynamicProperties.v;
        u=dynamicProperties.rates.u;
        w=dynamicProperties.rates.w;
        
        if (state.cADP==1000*10^(-6))
            resultsATP.D1000(j,1)=rl;
            resultsATP.D1000(j,2)=v;
        elseif (cADP==100*10^(-6))
A. MATLAB CODE

```matlab
101 resultsATP.D100(j,1)=rl;
102 resultsATP.D100(j,2)=v;
103 elseif((cADP==10*10^-6))
104 resultsATP.D10(j,1)=rl;
105 resultsATP.D10(j,2)=v;
106 end
107 end
108
110 om=linspace(0,n1,n);
111 close;
113
115 %Plot data files
116 %Experimental Results
117 %Uemura fex vs Velocity, 1mM [ATP], 1muM [ADP]
118 x1 = [(0.40-forceFrom)*n/(forceTo-forceFrom) ...
(0.90-forceFrom)*n/(forceTo-forceFrom) ...
(1.30-forceFrom)*n/(forceTo-forceFrom) ...
(1.50-forceFrom)*n/(forceTo-forceFrom) ...
(1.80-forceFrom)*n/(forceTo-forceFrom) ...
(2.20-forceFrom)*n/(forceTo-forceFrom) ...
(2.45-forceFrom)*n/(forceTo-forceFrom)];
119 y1 = [0.46 0.45 0.37 0.15 0.09 0.05 0.025];
120 e1 = [0.110 0.150 0.09 0.015 0.01 0 0];
123 %Uemura fex vs Velocity, 10muM [ATP], 1muM [ADP]
124 x2 = [(0.45-forceFrom)*n/(forceTo-forceFrom) ...
(0.80-forceFrom)*n/(forceTo-forceFrom) ...
(1.35-forceFrom)*n/(forceTo-forceFrom) ...
(1.7-forceFrom)*n/(forceTo-forceFrom) ...
(2.5-forceFrom)*n/(forceTo-forceFrom)];
125 y2 = [0.22 0.215 0.15 0.055 0];
128 e2 = [0.03 0.03 0.04 0.005 0];
132 %Baker [ADP] vs Run Length, 1mM [ATP], 0.0 fex
134 x3 = [(0.55-forceFrom)*n/(forceTo-forceFrom) ...
(1.1-forceFrom)*n/(forceTo-forceFrom) ...
(1.5-forceFrom)*n/(forceTo-forceFrom) ...
(1.8-forceFrom)*n/(forceTo-forceFrom) ...
(2.3-forceFrom)*n/(forceTo-forceFrom)];
135 y3 = [0.07 0.065 0.055 0.050 0.035];
138 e3 = [0.01 0.01 0.01 0 0];
140 %Baker [ADP] vs Velocity, 1mM [ATP], 0.0 fex
142 x4 = [0 50 100 200 350 400 600 1000 2500 5000];
145 y4 = [0.82 0.62 0.54 0.48 0.35 0.37 0.31 0.38 0.44];
148 e4 = [];
151 %Baker [ADP] vs Velocity, 10muM [ATP]
153 x5 = [0 50 100 200 350 400 600 1000 2500 5000];
156 y5 = [0.54 0.53 0.37 0.37 0.3 0.26 0.19 0.13 0.14 0.02];
159 e5 = [];
165 %Forkey [ATP] vs V, 10mM [ADP]
167 x7 = [10 20 50 400 4000];
170 y7 = [0.035 0.06 0.14 0.35 0.42];
```
\[ e^7 = []; \]

\[ x^8 = \{ (-10 \text{--} \text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) \ldots \]
\[ (-5 \text{--} \text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) \ldots \]
\[ (1 \text{--} \text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) \ldots \]
\[ (3 \text{--} \text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) \ldots \]
\[ (5 \text{--} \text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) \ldots \]
\[ (10 \text{--} \text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) \}; \]
\[ y^8 = [0.28 0.33 0.06 -0.1 -0.18 -0.43]; \]

\[ \text{Cost function data points} \]
\[ xC1 = [0 2500]; \]
\[ yC1 = [0.82 0.38]; \]
\[ \% \Delta 1 \text{ and } \Delta 6 \text{ Lvs[ADP]} \text{ [ATP]} = 1000 \ldots \]
\[ \text{PLUS SIGN} \]
\[ xC11 = [0.1]; \]
\[ yC11 = [1.15]; \]
\[ \% \Delta 3 \text{ Lvs[ADP]} \text{ [ATP]} = 100 \ldots \]
\[ \text{Downward-pointing triangle} \]
\[ xC2 = [100 1000]; \]
\[ yC2 = [1.15 0.82]; \]
\[ \% \Delta 3 \text{ Lvs[ATP]} \text{ [ADP]} = 0.1 \text{ CROSS} \]
\[ xC21 = [1000]; \]
\[ yC21 = [0.38]; \]
\[ \% \Delta 6 \text{ Lvs[ATP]} \text{ [ADP]} = 2500 \text{ ASTERISK} \]
\[ xC3 = [0.1 200 2500]; \]
\[ \% \Delta 1, \Delta 5 \text{ and } \Delta 7 \text{ Lvs[ATP]} \text{ [ADP]} = 1000 \text{ PLUS SIGN} \]
\[ xC31 = [0.1]; \]
\[ yC31 = [0.075]; \]
\[ \% \Delta 4 \text{ Vvs[ADP]} \text{ [ATP]} = 10 \ldots \]
\[ \text{Right-pointing triangle} \]
\[ xC4 = [10 1000]; \]
\[ yC4 = [0.075 0.54]; \]
\[ \% \Delta 4 \text{ and } \Delta 1 \text{ Vvs[ADP]} \text{ [ATP]} = 0.1 \text{ CROSS} \]
\[ xC41 = [1000]; \]
\[ yC41 = [0.32]; \]
\[ \% \Delta 5 \text{ Vvs[ATP]} \text{ [ADP]} = 200 \text{ Left-pointing ... triangle} \]
\[ xC42 = [1000]; \]
\[ yC42 = [0.14]; \]
\[ \% \Delta 7 \text{ Vvs[ATP]} \text{ [ADP]} = 2500 \text{ ASTERISK} \]
\[ xC5 = [(0.75--\text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) ]; \]
\[ \text{Lvsfex ... ATP}=1 \text{mM}, \text{ ADP}=200 \text{muM}, \text{ Pi}=0.1 \text{muM}, \text{ fex}=0.75 \text{pN} \]
\[ yC5 = [0.4]; \]
\[ \% \Delta 10 \text{ CROSS} \]
\[ xC51 = [(0--\text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) ]; \]
\[ \text{Lvsfex ... ATP}=0.1 \text{muM}, \text{ Pi}=0.1 \text{muM}, \text{ fex}=0 \text{pN} \]
\[ yC51 = [0.82]; \]
\[ \% \Delta 1 \text{ PLUS SIGN} \]
\[ xC52 = [(0--\text{forceFrom}) \cdot n/(\text{forceTo}--\text{forceFrom}) ]; \]
\[ \text{Lvsfex ... ATP}=100 \text{muM}, \text{ ADP}=0.1 \text{muM}, \text{ Pi}=0.1 \text{muM}, \text{ fex}=0 \text{pN} \]
\[ yC52 = [1.15]; \]
\[ \% \Delta 3 \text{ ASTERISK} \]
A. MATLAB CODE

\[
x_{C6} = \left( 0.75 - forceFrom \right) \times n / \left( forceTo - forceFrom \right);
\]

\[
y_{C6} = [0.32];
\]

\[
x_{C61} = \left( 0 - forceFrom \right) \times n / \left( forceTo - forceFrom \right);
\]

\[
y_{C61} = [0.54];
\]

\[
x_{C62} = \left( 0 - forceFrom \right) \times n / \left( forceTo - forceFrom \right);
\]

\[
y_{C62} = [0.32];
\]

\[
x_{C63} = \left( 0 - forceFrom \right) \times n / \left( forceTo - forceFrom \right);
\]

\[
y_{C63} = [0.075];
\]

\[
if(strcmp(separate,'no') == 1)
\]

\[
\text{rows}=2;
\]
\[
\text{columns}=3;
\]

\[
\text{subplot(rows,columns,1)};
\]

\[
\text{plot(om*12.5, resultsADP.T1000(:,1),'k', ...}
\]
\[
\text{om*12.5, resultsADP.T500(:,1),'k', om*12.5, resultsADP.T10(:,1),'--k', ...}
\]
\[
x4,y4,'ko');
\]

\[
\text{% xlabel('[ADP] (\mu M)');}
\]
\[
\text{ylabel('Run Length \mu m');}
\]
\[
\text{axis([0 5000 0 1.5]);}
\]
\[
\text{% hleg1 = legend('[ATP]=1mM','[ATP]=500 ...}
\]
\[
\text{\mu M','[ATP]=10\mu M');}
\]

\[
\text{subplot(rows,columns,2)};
\]

\[
\text{plot(om*12.5, resultsATP.D1000(:,1),'k', ...}
\]
\[
\text{om*12.5, resultsATP.D100(:,1),'k', om*12.5, resultsATP.D10(:,1),'--k', ...}
\]
\[
x6,y6,'ko');
\]

\[
\text{% xlabel('[ATP] (\mu M)');}
\]
\[
\text{ylabel('Run Length \mu m');}
\]
\[
\text{axis([0 5000 0 1.5]);}
\]
\[
\text{% hleg1 = legend('[ADP]=1mM','[ADP]=100 ...}
\]
\[
\text{\mu M','[ADP]=10\mu M');}
\]

\[
\text{subplot(rows,columns,3)};
\]

\[
\text{plot(om, resultsForcing.D1.T1000(:,1),'k', ...}
\]
\[
\text{om, resultsForcing.D200.T1000(:,1),'k', ...}
\]
\[
\text{om, resultsForcing.D1.T10(:,1),'--k');}
\]

\[
\text{% xlabel('fex (pN)');}
\]
\[
\text{ylabel('Run Length \mu m');}
\]
\[
\text{set(gca,'XTick',1:n/(forceTo-forceFrom):n+1);}
\]
\[
\text{set(gca,'XTickLabel',[forceFrom:forceTo]);}
\]
\[
\text{axis([1 n+1 0 1.5]);}
\]
% hleg1 = legend('[ATP]=1mM, [ADP]=1\mu M', ' [ATP]=1mM, [ADP]=200\mu M', ' [ATP]=10\mu M, [ADP]=1\mu M');

% plot [ADP] against Velocity
subplot(rows,columns,4);
plot(om*12.5,resultsADP.T1000(:,2), 'k', ...
  om*12.5,resultsADP.T500(:,2), ':k', ...
  om*12.5,resultsADP.T10(:,2), '--k', ...
  x5,y5, 'ko');
xlabel('[ADP] (\mu M)');
ylabel('Velocity \mu m s^{-1}');
axis([0 5000 0 0.6]);
% hleg1 = legend('[ATP]=1mM', '[ATP]=500 ...
  \mu M', '[ATP]=10\mu M');

% plot [ATP] against velocity
subplot(rows,columns,5);
plot(om*12.5,resultsATP.D1000(:,2), 'k', ...
  om*12.5,resultsATP.D100(:,2), ':k', ...
  om*12.5,resultsATP.D10(:,2), '--k', ...
  x7,y7, 'ko');
xlabel('[ATP] (\mu M)');
ylabel('Velocity \mu m s^{-1}');
axis([0 5000 0 0.6]);
% hleg1 = legend('[ADP]=1\mu M', ' [ADP]=100 ...
  \mu M', '[ADP]=10\mu M');

% plot force against velocity
subplot(rows,columns,6);
plot(om,resultsForcing.D1.T1000(:,2), 'k', ...
  om,resultsForcing.D200.T1000(:,2), ':k', ...
  om,resultsForcing.D1.T10(:,2), '--k', ...
  x1,y1, 'ko', x2,y2, 'k^', x3,y3, 'ks', x8,y8, 'ks');
xlabel('fex (pN)');
ylabel('Velocity \mu m s^{-1}');
set(gca, 'XTick',1:n/(forceTo
  forceFrom):n+1); set(gca, 'XTickLabel',{forceFrom:forceTo});
axis([1 n+1 -0.2 0.6]);
% hleg1 = legend('[ATP]=1mM, [ADP]=1\mu M', '[ATP]=10mM, ... 
  [ADP]=200\mu M', '[ATP]=10\mu M, [ADP]=1\mu M');

ha = axes('Position',[0 0 1 1], 'Xlim',[0 1], 'Ylim',[0 ... 1],'Box','off', 'Visible', 'off', 'Units', 'normalized', ...
  'clipping', 'off');
title = strcat(model, '; steps=', ...
um2str(constants.steps));
text(0.5, 1,title, 'HorizontalAlignment' ... 
  'center', 'VerticalAlignment', 'top');
if((strcmp(model,'SAGBM')==1)) %Skau Asg Bierbaum Model
  cost = CostFunctionSAGBM(model,parameters,constants);
else
  cost = CostFunction(model,parameters,constants);
end
fname = sprintf('graph.All_%s_%0.0f', model, cost*10);
print('-dpng',fname);
close;

%Write the data to a file
fname = sprintf('results.All_%s_%0.0f', model, cost*10);
fid = fopen(fname, 'w');
fprintf(fid, title);
fprintf(fid, 'n
');
for i=1:length(parameters)
    fprintf(fid, 'f
', parameters(i));
end
fclose(fid);

elseif(strcmp(separate,'yes')==1)

%set line width
lw=1.5;

%set text size
tss=23;
tsl=27;

%marker size
markerSize=20;

%plot [ADP] against runlength
p = plot(om*12.5,resultsADP.T1000(:,1),'k', ...
         om*12.5,resultsADP.T500(:,1),':k',om*12.5,resultsADP.T10(:,1),'--k', ...
         x4,y4,'ko',xC1,yC1,'k+',xC11,yC11,'kv');
set(p,'LineWidth',lw);
hold on;
xlabel('[ADP] (\mu M)','fontsize',tss);
ylabel('Run Length (\mu m)','fontsize',tss);
axis([0 5000 0 1.5]);
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
set(p, 'MarkerSize', markerSize);
hlleg1 = legend('[ATP]=1mM','[ATP]=500 ... 
\mu M','[ATP]=10\mu M');

cost = CostFunction(model,parameters,constants);
fname = sprintf('graph.LADP_%s_steps %0.f_%0.0f', ... 
model,constants.steps,cost*10);
print('-dpng',fname);
close;

%plot [ATP] against runlength
p = plot(om*12.5,resultsATP.D1000(:,1),'k', ...
         om*12.5,resultsATP.D100(:,1),':k',om*12.5,resultsATP.D10(:,1),'--k', ...
         x6,y6,'ko',xC2,yC2,'kx',xC21,yC21,'k*');
set(p,'LineWidth',lw);
hold on;
xlabel('[$\text{ATP (\mu M)}$]', 'fontsize', tsl);
ylabel('Run Length [\mu m]', 'fontsize', tsl);
axis([0 5000 0 1.5]);
set(gca, 'Fontsize', tss);
set(gca, 'LineWidth', lw);
set(p, 'MarkerSize', markerSize);

% hleg1 = legend('[$\text{ATP}=1 \text{mM}$]', ...
%                  '[$\text{ATP}=100 \ldots \mu M$]');

cost = CostFunction(model, parameters, constants);
fname = sprintf('graph_LATP_%s_steps %0.0f', ...
model, constants.steps, cost*10);
print('-dpng', fname);
close;

% plot force against run length
p = plot(om, resultsForcing.D1.T1000(;,1), 'k', ...
         om, resultsForcing.D200.T1000(;,1), ':k', ...
         om, resultsForcing.D1.T100(:,1), '--k', ...
         xC5, yC5, 'kx', xC51, yC51, 'k+', xC52, yC52, 'k*');
set(p, 'LineWidth', lw);
hold on;
xlabel('fex (pN)', 'fontsize', tsl);
ylabel('Run Length [\mu m]', 'fontsize', tsl);
set(gca, 'XTick', 1:n/(forceTo-forceFrom):n+1);
set(gca, 'XTickLabel', {forceFrom:forceTo});
axis([1 n+1 0 1.5]);
set(gca, 'FontSize', tss);
set(gca, 'LineWidth', lw);
set(p, 'MarkerSize', markerSize);

% hleg1 = legend('[$\text{ATP}=1 \text{mM}$]', '[$\text{ATP}=100 \ldots \mu M$]', ...
%                  '[$\text{ATP}=200 \mu M$]');

cost = CostFunction(model, parameters, constants);
fname = sprintf('graph_Lfex_%s_steps %0.0f', ...
model, constants.steps, cost*10);
print('-dpng', fname);
close;

% plot [ADP] against Velocity
p = plot(om*12.5, resultsADP.T1000(:,2), 'k', ...
         om*12.5, resultsADP.T500(:,2), ':k', om*12.5, resultsADP.T10(:,2), '--k', ...
         x5, y5, 'ko', xC3, yC3, 'k+', xC31, yC31, 'k*');
set(p, 'LineWidth', lw);
hold on;
xlabel('[$\text{ADP (\mu M)}$]', 'fontsize', tsl);
ylabel('Velocity [\mu m/s]', '-1', 'fontsize', tsl);
axis([0 5000 0 0.6]);
set(gca, 'Fontsize', tss);
set(gca, 'LineWidth', lw);
set(p, 'MarkerSize', markerSize);
% hleg1 = legend('[$\text{ATP}=1 \text{mM}$]', '[$\text{ATP}=500 \ldots \mu M$]');
cost = CostFunction(model, parameters, constants);
fname = sprintf('graph_VADP_ %s_steps %0.f_ %0.0f', ...  
    model, constants.steps, cost*10);
print('-dpng', fname);
close;

% plot [ATP] against velocity
p = plot(om*12.5, resultsATP.D100(:,2), 'k', ...  
    om*12.5, resultsATP.D100(:,2), ':k', ...  
    x7, y7, 'ko', xC4, yC4, 'kx', xC41, yC41, 'k>', xC42, yC42, 'k*');
set(p, 'LineWidth', lw);
hold on;
xlabel('ATP (\muM)', 'fontsize', tsl);
axis([0 5000 0 0.6]);
set(gca, 'FontSize', tss);
set(gca, 'LineWidth', lw);
set(p, 'MarkerSize', markerSize);

% hleg1 = legend('[ATP]=1mM', '[ATP]=100mM ... 
\muM', '([ADP]=1mM', '([ADP]=100 ... 
\muM');

% plot force against velocity
p = plot(om, resultsForcing.D1.T1000(:,2), 'k', ...  
    om, resultsForcing.D200.T1000(:,2), ':k', ...  
    x1, y1, 'ko', x2, y2, 'k>', x3, y3, 'ks', ...  
    x8, y8, 'kv', xC6, yC6, 'kx', ...  
    xC61, yC61, 'k+', xC62, yC62, 'k*', xC63, yC63, 'k<');
set(p, 'LineWidth', lw);
hold on;
xlabel('fex (pN)', 'fontsize', tsl);
ylabel('Velocity \mu m s^{-1}', 'fontsize', tsl);
set(gca, 'XTick', 1:n/(forceTo-forceFrom):n+1);
set(gca, 'XTickLabel', [forceFrom:forceTo]);
axis([1 n+1 -0.2 0.7]);
set(gca, 'FontSize', tss);
set(gca, 'LineWidth', lw);
set(p, 'MarkerSize', markerSize);

% hleg1 = legend('[ATP]=1mM, [ADP]=1\muM', '[ATP]=1mM, ... 
\muM', '[ATP]=200\muM', '([ATP]=10\muM, [ADP]=1\muM');

% plot velocity against ATP
p = plot(om*12.5, resultsATP.D1000(:,2), 'k', ...  
    om*12.5, resultsATP.D100(:,2), ':k', ...  
    x7, y7, 'ko', xC4, yC4, 'kx', xC41, yC41, 'k>', xC42, yC42, 'k*');
set(p, 'LineWidth', lw);
hold on;
xlabel('ATP (n\muM)', 'fontsize', tsl);
axis([0 5000 0 0.6]);
set(gca, 'FontSize', tss);
set(gca, 'LineWidth', lw);
set(p, 'MarkerSize', markerSize);

% hleg1 = legend('[ATP]=1mM', '[ATP]=100 ... 
\muM', '([ADP]=1mM', '([ADP]=100 ... 
\muM');
%Write the data to a file
fname = sprintf('parameters_ %s_steps %0.f_ %0.f', ...  
    model, constants.steps, cost*10);
fid = fopen(fname, 'w');
fprintf(fid, model);
  fprintf(fid, ' \
');
  fprintf(fid, ' %f \n', cost);
for i=1:length(parameters)
  fprintf(fid, ' %f \t', parameters(i));
end
fclose(fid);
else
  error('separate incorrectly defined in call of ...  
    PrintAllResults')
end
%Prints the required results for a given set of parameters and ... models

function ...
    PrintCompareResults(model_1, model_2, parameters_1, parameters_2)

    constants = CalculateConstants();

    n=500;
    state.fex=0.0;
    state.cPi=0.1*10^(-6);
    state.cADP=0.1*10^(-6);
    state.cATP=1000*10^(-6);

    %Quantities against ADP
    state.fex=0.0;
    state.cPi=0.1*10^(-6);
    state.cATP=1000*10^(-6);

    for j=1:n
        state.cADP=j*(12.5)*10^(-6);

        dynamicProperties = ...
            CalculateDynamicProperties(model_1, parameters_1, constants, state);
        results1ADP.ATP1000.rl(j)=dynamicProperties.rl;
        results1ADP.ATP1000.v(j)=dynamicProperties.v;

        dynamicProperties = ...
            CalculateDynamicProperties(model_2, parameters_2, constants, state);
        results2ADP.ATP1000.rl(j)=dynamicProperties.rl;
        results2ADP.ATP1000.v(j)=dynamicProperties.v;
    end

    state.cATP=500*10^(-6);

    for j=1:n
        state.cADP=j*(12.5)*10^(-6);

        dynamicProperties = ...
            CalculateDynamicProperties(model_1, parameters_1, constants, state);
        results1ADP.ATP500.rl(j)=dynamicProperties.rl;
        results1ADP.ATP500.v(j)=dynamicProperties.v;

        dynamicProperties = ...
            CalculateDynamicProperties(model_2, parameters_2, constants, state);
        results2ADP.ATP500.rl(j)=dynamicProperties.rl;
        results2ADP.ATP500.v(j)=dynamicProperties.v;
    end
state.cATP=10*10^(-6);
for j=1:n
    state.cADP=j*(12.5)*10^(-6);

    dynamicProperties = ...
    CalculateDynamicProperties(model_1,parameters_1,constants,state);
    results1ADP.ATP10.rl(j)=dynamicProperties.rl;
    results1ADP.ATP10.v(j)=dynamicProperties.v;

    dynamicProperties = ...
    CalculateDynamicProperties(model_2,parameters_2,constants,state);
    results2ADP.ATP10.rl(j)=dynamicProperties.rl;
    results2ADP.ATP10.v(j)=dynamicProperties.v;
end

%Quantities against ATP
state.fex=0.0;
state.cPi=0.1*10^(-6);
state.cADP=10*10^(-6);
for j=1:n
    state.cATP=j*(12.5)*10^(-6);

    dynamicProperties = ...
    CalculateDynamicProperties(model_1,parameters_1,constants,state);
    results1ATP.ADP10.rl(j)=dynamicProperties.rl;
    results1ATP.ADP10.v(j)=dynamicProperties.v;

    dynamicProperties = ...
    CalculateDynamicProperties(model_2,parameters_2,constants,state);
    results2ATP.ADP10.rl(j)=dynamicProperties.rl;
    results2ATP.ADP10.v(j)=dynamicProperties.v;
end

state.cADP=100*10^(-6);
for j=1:n
    state.cATP=j*(12.5)*10^(-6);

    dynamicProperties = ...
    CalculateDynamicProperties(model_1,parameters_1,constants,state);
    results1ATP.ADP1000.rl(j)=dynamicProperties.rl;
    results1ATP.ADP1000.v(j)=dynamicProperties.v;

    dynamicProperties = ...
    CalculateDynamicProperties(model_2,parameters_2,constants,state);
    results2ATP.ADP1000.rl(j)=dynamicProperties.rl;
    results2ATP.ADP1000.v(j)=dynamicProperties.v;
end

state.cADP=1000*10^(-6);
for j=1:n
A. MATLAB CODE

96 state.cATP=j*(12.5)*10^(-6);
97
98 dynamicProperties = ...;
99 results1ATP.ADP1000.rl(j)=dynamicProperties.rl;
100 results1ATP.ADP1000.v(j)=dynamicProperties.v;
101
102 dynamicProperties = ...;
103 results2ATP.ADP1000.rl(j)=dynamicProperties.rl;
104 results2ATP.ADP1000.v(j)=dynamicProperties.v;
105
106 om=linspace(0,n,n);
107 close;
108
109
110 %Plot data files
111 %Experimental Results
112 %Uemura fex vs Velocity, 1mM [ATP], 1muM [ADP]
113 x1 = [(0.40+10) *25 (0.90+10)*25 (1.30+10)*25 (1.50+10)*25 ...(1.80+10)*25 (2.20+10)*25 (2.45+10)*25];
114 y1 = [0.46 0.45 0.37 0.15 0.09 0.05 0.025];
115 e1 = [0.110 0.150 0.09 0.015 0.01 0 0];
116 %Uemura fex vs Velocity, 1mM [ATP], 200muM [ADP]
117 x2 = [(0.45+10) *25 (0.80+10)*25 (1.35+10)*25 (1.7+10)*25 ...(2.5+10)*25];
118 y2 = [0.22 0.215 0.15 0.055 0];
119 e2 = [0.03 0.03 0.04 0.005 0];
120 %Uemura fex vs Velocity, 10muM [ATP], 1muM [ADP]
121 x3 = [(0.55+10)*25 (1.1+10)*25 (1.5+10)*25 (1.8+10)*25 ...(2.3+10)*25];
122 y3 = [0.07 0.065 0.055 0.050 0.035];
123 e3 = [0.01 0.01 0.01 0 0];
124 %Baker [ADP] vs Run Length, 1mM [ATP], 0.0 fex
125 x4 = (0 50 100 200 350 400 600 1000 2500 5000);
126 y4 = (0.82 0.62 0.54 0.48 0.35 0.37 0.37 0.31 0.38 0.44);
127 e4 = [ ];
128 %Baker [ADP] vs Velocity, 1mM [ATP], 0.0 fex
129 x5 = (0 50 100 200 350 400 600 1000 2500 5000);
130 y5 = (0.54 0.53 0.37 0.32 0.3 0.26 0.19 0.13 0.14 0.02);
131 e5 = [ ];
132 %Baker [ATP] vs RL, 10muM [ADP]
133 x6 = (50 100 1000);
134 y6 = (1.4 1.15 0.82);
135 e6 = [ ];
136 %Forkey [ATP] vs V, 10mM [ADP]
137 x7 = (10 20 50 400 4000);
138 y7 = (0.035 0.06 0.14 0.35 0.42);
139 e7 = [ ];
140 %Gebhardt 2006 velocity vs high forcing
141 x8 = ((-5+10)*25 (-5+10)*25 (5+10)*25);
142 y8 = (0.28 0.33 -0.12 );
143
144 %Cost function data points
xC1 = [0 2500];
yC1 = [0.82 0.38]; % Delta1 and Delta6 Lvs[ADP] [ATP]=1000 ...
PLUS SIGN

xC11 = [ 0.1 ]; %
yC11 = [ 1.15 ]; % Delta3 Lvs[ADP] [ATP]=100 ...
Downward-pointing triangle

xC2 = [100 1000]; %
yC2 = [1.15 0.82]; % Delta3 and Delta1 Lvs[ATP] [ADP]=0.1 CROSS

xC21 = [ 1000 ]; %
yC21 = [ 0.38 ]; % Delta6 Lvs[ATP] [ADP]=2500 ASTERISK

xC3 = [0.1 200 2500]; % Delta2, Delta5 and Delta7
yC3 = [0.54 0.32 0.14]; % Vvs[ADP] [ATP] = 1000 PLUS SIGN

xC31 = [ 0.1 ]; %
yC31 = [ 0.075 ]; % Delta4 Vvs[ADP] [ATP] = 10 ...
Right-pointing triangle

xC4 = [10 1000]; % Delta4 and Delta1
yC4 = [0.075 0.54]; % Vvs[ATP] [ADP] = 0.1 CROSS

xC41 = [ 1000 ]; %
yC41 = [ 0.32 ]; % Delta5 Vvs[ATP] [ADP] = 200 Left-pointing ...
triangle

xC42 = [ 1000 ]; %
yC42 = [ 0.14 ]; % Delta7 Vvs[ATP] [ADP] = 2500 ASTERISK

% set line width
lw=1.5;
% set text size
tss=23;
tsl=27;
% marker size
markerSize=20;
% plot [ADP] against runlength
hold off;
p = ...;
plot(om*12.5,results1ADP.ATP1000.rl,'b',om*12.5,results2ADP.ATP1000.rl,'r',
    set(p,'LineWidth',lw);
hold on;
xlabel('[ADP] (\mu M)','fontsize',tsl);
ylabel('Run Length \mu mum','fontsize',tsl);
axis([0 5000 0 1])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
set(p, 'MarkerSize', markerSize);
hleg1 = legend(model_1,model_2);
A. MATLAB CODE

```matlab
name=strcat('CompareResults_ADPsRL', model_1, '.', model_2);
print('-dpng', name);
close;

% plot [ATP] against runlength
hold off;
p = ...
  plot(om*12.5,results1ATP.ADP1000.rl,'b',om*12.5,results2ATP.ADP1000.rl,'r',
       p = ...
  plot(om*12.5,results1ATP.ADP10.rl,om*12.5,results2ATP.ADP10.rl,x6,y6,'ko')
set(p,'LineWidth',lw);
hold on;
xlabel('[ATP] (\mu M)','fontsize',tsl);
ylabel('Run Length (\mu m)','fontsize',tsl);
axis([0 5000 0 1.5])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
set(p, 'MarkerSize', markerSize);
hleg1 = legend(model_1,model_2);
name=strcat('CompareResults_ATPvsRL', model_1, '.', model_2);
print('-dpng', name);
close;

% plot [ADP] against velocity
hold off;
p = ...
  plot(om*12.5,results1ADP.ADP1000.v,'b',om*12.5,results2ADP.ADP1000.v,'r',om
set(p,'LineWidth',lw);
hold on;
xlabel('[ADP] (\mu M)','fontsize',tsl);
ylabel('Velocity (\mu ms^{-1})','fontsize',tsl);
axis([0 5000 0 1])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
set(p, 'MarkerSize', markerSize);
hleg1 = legend(model_1,model_2);
name=strcat('CompareResults_ADPsV', model_1, '.', model_2);
print('-dpng', name);
close;

% plot [ATP] against velocity
hold off;
p = ...
  plot(om*12.5,results1ATP.ADP1000.v,'b',om*12.5,results2ATP.ADP1000.v,'r',om
set(p,'LineWidth',lw);
hold on;
xlabel('[ATP] (\mu M)','fontsize',tsl);
ylabel('Velocity (\mu ms^{-1})','fontsize',tsl);
axis([0 5000 0 1])
set(gca,'FontSize',tss);
```
set(gca,'LineWidth',lw);
set(p,'MarkerSize',markerSize);
hleg1 = legend(model_1,model_2);
name=strcat('CompareResults_ATPvsV', model_1, '.', model_2);
print('-dpng', name);
close;
end
A. MATLAB CODE

PrintFluxComparison

%Prints the required results for a given set of parameters and ...
%model

function ...
PrintFluxComparison(model_1,model_2,model_3,parameters_1,parameters_2,parameters_3)

constants = CalculateConstants();

n=500;
state.fex=0.0;
state.cPi=0.1*10^(-6);
state.cADP=0.1*10^(-6);
state.cATP=1000*10^(-6);

%Quantities against ADP
state.fex=0.0;
state.cPi=0.1*10^(-6);
state.cATP=1000*10^(-6);

for j=1:n
state.cADP=j*(12.5)*10^(-6);

dynamicProperties = ...
    CalculateDynamicProperties(model_1,parameters_1,constants,state);
    Jhyd=dynamicProperties.v/constantsd;
    Jfut=dynamicProperties.vFutile/constantsd2;
    Jtot=Jhyd+Jfut;
    JhydNormal=Jhyd/Jtot;
    JfutNormal=Jfut/Jtot;
    results1ADP.ATP1000.Jhyd(j)=JhydNormal;
    results1ADP.ATP1000.Jfut(j)=JfutNormal;
    results1ATP.ATP1000.JRelativeDetach(j) = ...
        dynamicProperties.x(5)/dynamicProperties.x(9);

    dynamicProperties = ...
        CalculateDynamicProperties(model_2,parameters_2,constants,state);
    Jhyd=dynamicProperties.v/constantsd;
    Jfut=dynamicProperties.vFutile/constantsd2;
    Jtot=Jhyd+Jfut;
    JhydNormal=Jhyd/Jtot;
    JfutNormal=Jfut/Jtot;
    results2ADP.ATP1000.Jhyd(j)=JhydNormal;
    results2ADP.ATP1000.Jfut(j)=JfutNormal;
    results2ATP.ATP1000.JRelativeDetach(j) = ...
        dynamicProperties.x(5)/dynamicProperties.x(9);

    dynamicProperties = ...
        CalculateDynamicProperties(model_3,parameters_3,constants,state);
    Jhyd=dynamicProperties.v/constantsd;
Jfut=dynamicProperties.vFutile/constants.d2;
Jtot=Jhyd+Jfut;
JhydNormal=Jhyd/Jtot;
JfutNormal=Jfut/Jtot;
results3ADP.ATP1000.Jhyd(j)=JhydNormal;
results3ADP.ATP1000.Jfut(j)=JfutNormal;
results3ATP.ATP1000.JRelativeDetach(j) = ...
    dynamicProperties.x(5)/dynamicProperties.x(9);
end

%Quantities against ATP
state.cADP=0.1*10^(-6);
state.cATP=1000*10^(-6);
for j=1:n
    state.cATP=j*(12.5)*10^(-6);
    dynamicProperties = ...;
    CalculateDynamicProperties(model1,parameters1,constants,state);
    Jhyd=dynamicProperties.v/constants.d;
    Jfut=dynamicProperties.vFutile/constants.d2;
    Jtot=Jhyd+Jfut;
    JhydNormal=Jhyd/Jtot;
    JfutNormal=Jfut/Jtot;
    results1ATP.ADP01.Jhyd(j)=JhydNormal;
    results1ATP.ADP01.Jfut(j)=JfutNormal;
    results1ATP.ADP01.JRelativeDetach(j) = ...;
    dynamicProperties.x(5)/dynamicProperties.x(9);
    dynamicProperties = ...;
    CalculateDynamicProperties(model2,parameters2,constants,state);
    Jhyd=dynamicProperties.v/constants.d;
    Jfut=dynamicProperties.vFutile/constants.d2;
    Jtot=Jhyd+Jfut;
    JhydNormal=Jhyd/Jtot;
    JfutNormal=Jfut/Jtot;
    results2ATP.ADP01.Jhyd(j)=JhydNormal;
    results2ATP.ADP01.Jfut(j)=JfutNormal;
    results2ATP.ADP01.JRelativeDetach(j) = ...;
    dynamicProperties.x(5)/dynamicProperties.x(9);
    dynamicProperties = ...;
    CalculateDynamicProperties(model3,parameters3,constants,state);
    Jhyd=dynamicProperties.v/constants.d;
    Jfut=dynamicProperties.vFutile/constants.d2;
    Jtot=Jhyd+Jfut;
    JhydNormal=Jhyd/Jtot;
    JfutNormal=Jfut/Jtot;
    results3ATP.ADP01.Jhyd(j)=JhydNormal;
    results3ATP.ADP01.Jfut(j)=JfutNormal;
    results3ATP.ADP01.JRelativeDetach(j) = ...;
    dynamicProperties.x(5)/dynamicProperties.x(9);
end
om=linspace(0,n-1,n);
close;

% set line width
lw=1.5;

% set text size
tss=18;
tsl=22;

% plot [ADP] against futile cycling
hold off;
p = plot(om*12.5,results1ADP.ADP1000.Jfut,'b', ...
     om*12.5,results2ADP.ATP1000.Jfut,':b', ...
     om*12.5,results3ADP.ATP1000.Jfut,'--b');
set(p,'LineWidth',lw);
hold on;
xlabel('
     \textit{[ADP]} (\textmu M)', 'fontsize',tsl);
ylabel('Futile Flux Proportion of Total', 'fontsize',tsl);
axis([0 5000 0 1])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);

hleg1 = legend(model1,model2,model3);
name=strcat('CompareJFut\_ADP', model1, ' ', model2, ' ', model3);
print('-dpng', name);
close;

% plot [ATP] against futile cycling
hold off;
p = plot(om*12.5,results1ATP.ADP01.Jfut,'b', ...
     om*12.5,results2ATP.ADP01.Jfut,':b', ...
     om*12.5,results3ATP.ADP01.Jfut,'--b');
set(p,'LineWidth',lw);
hold on;
xlabel('
     \textit{[ATP]} (\textmu M)', 'fontsize',tsl);
ylabel('Futile Flux Proportion of Total', 'fontsize',tsl);
axis([0 5000 0 1])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);

hleg1 = legend(model1,model2,model3);
name=strcat('CompareJFut\_ATP', model1, ' ', model2, ' ', model3);
print('-dpng', name);
close;

% plot [ATP] against detachment vs hydrolysis states
hold off;
p = plot(om*12.5,results1ATP.ADP01.JRelativeDetach,'b', ...
     om*12.5,results2ATP.ADP01.JRelativeDetach,':b', ...
om*12.5, results3ATP.ADP01.JRelativeDetach,'--b');

set(p,'LineWidth',lw);
hold on;
xlabel('ATP (\muM)','','fontsize',tsl);
ylabel('Detachment vs Hydrolysis states','fontsize',tsl);

axis([0 5000 0 1])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
hleg1 = legend(model1,model2,model3);
name = strcat('CompareRelDetach_ATP', model1, '_', model2, ... '
'_, model3);
print('-dpng', name);
close;

end

PrintModelFluxes

function PrintModelFluxes(model,parameters)

constants = CalculateConstants();

n=500;
state.fex=0.0;
state.cPi=0.1*10^(-6);
state.cADP=0.1*10^(-6);
state.cATP=1000*10^(-6);

% Quantities against ADP
state.fex=0.0;
state.cPi=0.1*10^(-6);
state.cATP=1000*10^(-6);

for j=1:n
    state.cADP=j*(12.5)*10^(-6);
    dynamicProperties = ...
        CalculateDynamicProperties(model,parameters,constants,state);
    J45=dynamicProperties.J45;
    J37=dynamicProperties.J37;
    J29=dynamicProperties.J29;
    Jfut=dynamicProperties.vFutile/constants.d2;
    Jtot=J45+J37+J29+Jfut;
    J45Normal=J45/Jtot;
    J37Normal=J37/Jtot;
    J29Normal=J29/Jtot;
    JfutNormal=Jfut/Jtot;
% Quantities against ATP
state.cADP = 0.1 * 10^(-6);
state.cATP = 1000 * 10^(-6);

for j = 1:n
    state.cATP = j * (12.5) * 10^(-6);

    dynamicProperties = ...
        CalculateDynamicProperties(model, parameters, constants, state);
    J45 = dynamicProperties.J45;
    J37 = dynamicProperties.J37;
    J29 = dynamicProperties.J29;
    Jfut = dynamicProperties.vFutile / constants.d2;
    Jtot = J45 + J37 + J29 + Jfut;
    J45Normal = J45 / Jtot;
    J37Normal = J37 / Jtot;
    J29Normal = J29 / Jtot;
    JfutNormal = Jfut / Jtot;
    resultsATP.ADP01.J45(j) = J45Normal;
    resultsATP.ADP01.J37(j) = J37Normal;
    resultsATP.ADP01.J29(j) = J29Normal;
    resultsATP.ADP01.Jfut(j) = JfutNormal;
end

cm = linspace(0, n-1, n);
close;

% set line width
lw = 1.5;

% set text size
tss = 18;
tsl = 22;

% plot [ADP] against flux
hold off;
p = ...
    plot(cm*12.5, resultsADP.ATP1000.J45,'b', cm*12.5, resultsADP.ATP1000.J37,'b'
        set(p,'LineWidth',lw);
    hold on;
xlabel('[ADP] (\mu M)', 'fontSize', tsl);
ylabel('Flux Proportion of Total', 'fontSize', tsl);
axis([0 5000 0 1])
    set(gca, 'FontSize', tss);
    set(gca, 'LineWidth', lw);
\begin{verbatim}
 hailed = legend('J45','J37','J29','Jfut');
 name=strcat('CompareJADP', model);
 print('-dpng', name) ;
close;

%plot [ATP] against flux
hold off;
p = ...
    plot(om*12.5,resultsATP.ADP01.J45,'b',om*12.5,resultsATP.ADP01.J37,':b',om*12.5,resultsATP.ADP01.J29,'-b',om*12.5,resultsATP.ADP01.Jfut,'r');
set(p,'LineWidth',lw);
hold on;
xlabel('[ATP] (\mu M)','fontsize',tsl);
ylabel('Flux Proportion of Total','fontsize',tsl);
axis([0 5000 0 1])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
hleg1 = legend('J45','J37','J29','Jfut');
name=strcat('CompareJATP', model);
print('-dpng', name) ;
close;

%plot [ATP] against detachment flux
hold off;
p = plot(om*12.5,resultsATP.ADP01.Jdetach,'b');
set(p,'LineWidth',lw);
hold on;
xlabel('[ATP] (\mu M)','fontsize',tsl);
ylabel('Detachment Flux (s^{-1})','fontsizet',tsl);
axis([0 1000 0 0.00015])
set(gca,'FontSize',tss);
set(gca,'LineWidth',lw);
name=strcat('CompareJdetach_ATP', model);
print('-dpng', name) ;
close;
end
\end{verbatim}

\section*{B The Author’s Further Doctoral Activities}

\subsection*{B.1 Publications}

- The Role of Futile Cycling and Asymmetric Gating in Myosin-V, N.J. Boon and R.B. Hoyle, in preparation

B. THE AUTHOR’S FURTHER DOCTORAL ACTIVITIES

B.2 Awards

- KTN TakeAim 2012 Communicating Mathematics competition winner
- Best Presentation Award, University of Surrey Research Conference 2010

B.3 Conferences Attended

- MAGIC 2010: University of Leeds UK - attendance
- YRM 2010: University of Cambridge UK - oral presentation
- Dynamic Days Europe 2010: University of Bristol UK - oral presentation
- Surrey Postgraduate Conference 2010 - oral presentation
- YRM 2011: University of Warwick UK - oral presentation
- MAGIC 2011: University of Reading UK - oral presentation
- Surrey Postgraduate Conference 2012 - oral presentation
- Biophysical Society Annual General Meeting 2012: San Diego USA - poster presentation
- KTN TakeAim 2012: University of Oxford UK - oral presentation
- Biophysical Society Annual General Meeting 2013: Philadelphia USA - poster presentation

B.4 Courses Attended

- PGSDP: Basic Presentation Skills
- PGSDP: PhD Project Management
- Maths Stats and OR network: undergraduate teaching workshop
- PGSDP: Managing the Literature Search
- PGSDP: Writing the Confirmation Report
- PGSDP: Presenting Research at Conferences
- Level 3 French
- PGSDP: Getting Published
- SPLASH: Publication writing retreat
- SPLASH: Thesis writing retreat 1
- SPLASH: Thesis writing retreat 2
Bibliography


