HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Herr
Rico Hellwig

Topologievorhersage an Membranproteinen anhand eines auf Energieprofilen trainierten Hidden-Markov-Modells

2012

Fakultät **Mathematik** / **Naturwissenschaften** / **Informatik**

BACHELORARBEIT

Topologievorhersage an Membranproteinen anhand eines auf Energieprofilen trainierten Hidden-Markov-Modells

Autor:
Rico Hellwig

Studiengang:
Biotechnologie / Bioinformatik

Seminargruppe:
BI09w1-B

Erstprüfer:
Prof. Dr. rer. nat. Dirk Labudde

Zweitprüfer:
M.sc. Steffen Grunert

Mittweida, 08 2012

Bibliografische Angaben

Hellwig, Rico: Topologievorhersage an Membranproteinen anhand eines auf Energieprofilen trainierten Hidden-Markov-Modells, **??** Seiten, **??** Abbildungen, DVD mit Datensätzen und Software, sowie dieser Arbeit als PDF-Datei, Hochschule Mittweida (FH), Fakultät Mathematik / Naturwissenschaften / Informatik

Bachelorarbeit, 2012

Referat

Ziel der Arbeit war es ein Topologievorhersagetool zu entwickeln, dass anhand von Energieprofildaten die Topologie von Membranproteinen vorhersagen kann. Hierzu gliedert sich die Arbeit in zwei Hauptteile. Zum einen werden zunächst einige Grundlagen zu den Themen Proteine, Energieprofile und Hidden-Markov-Modelle angeführt, zum anderen erfolgt eine Erläuterung des Vorgehens in Bezug auf die Erstellung der Datensätze und des Modells selbst. Schließlich werden die gewonnenen Informationen in der Auswertung analysiert und diskutiert. Dies geschieht sowohl im Hinblick auf die Vorhersagegenauigkeit des Modells, als auch auf die Eigenschaften der erstellten Datensätze.

I. Inhaltsverzeichnis

II. Abbildungsverzeichnis

III. Tabellenverzeichnis

IV. Danksagung

An dieser Stelle möchte ich allen, die mich bei der Erstellung dieser Arbeit unterstützt haben, danken. Dies gilt vor allem meinen Betreuern Herr Prof. Dr. rer. nat. Dirk Labudde, B.sc Florian Heinke und M.sc. Steffen Grunert. Prof. Labudde danke ich dafür, dass er mir Möglichkeit gegeben hat diese Arbeit bei ihm anzufertigen und für seine Ideen in Bezug auf die Lösung einiger komplizierterer Probleme. Außerdem danke ich Florian Heinke für seine Anregungen zum Validieren der Datensätze. Mein besonderer Dank gilt an dieser Stelle Steffen Grunert. Dafür, dass er immer erreichbar war, wenn es Probleme gab und mir dann mit Rat und Tat zur Seite stand. Ohne seine Hilfe, Geduld und Beharrlichkeit im Ausbessern kleinerer und größerer Fehler in der Software, sowie seinen Elan bei der Diskussion über neue Lösungsansätze, hätte ich diese Arbeit niemals erstellen können.
Ebenso danke ich natürlich allen um mich herum. Besonders meiner Familie, meinen Freunden und meiner Freundin für ihre aufmunternden Worte, ihr Zuhören und ihre Geduld, wenn es bei mir ab und zu etwas stressiger war. Auch für die Unterstützung bei der Abrundung dieser Arbeit durch das Entfernen des einen oder anderen Rechtschreibfehlers beziehungsweise kryptischen Satzes möchte ich hiermit danken. Dies gilt vor allem meiner Mutter, die sich die Zeit nahm diese Arbeit nach bestem Wissen aufzuarbeiten.

1 Einleitung

Die Fähigkeit aktiv und gezielt Stoffe mit der Umgebung auszutauschen, um diese zu metabolisieren, ist per Definition einer der Hauptunterschiede, der tote Materie von Lebewesen unterscheidet. Gleich welche Organismen man betrachtet, zeigt sich ein annähernd identischer Grundbau. Zellen grenzen sich gegen ihre Umwelt durch eine Plasmamembran ab[?]. Pflanzenzellen und die meisten einzelligen Organismen besitzen zusätzlich noch eine Zellwand. Sie dient aber nur dem mechanischen Schutz der Zelle vor äußeren Einwirkungen.
Die Phospholipiddoppelschicht aus der die Zellmembran besteht, ist für die meisten Arten von Stoffen unpassierbar. Um den Lebewesen dennoch einen Stoffaustausch mit ihrer Umgebung zu ermöglichen, sind in die Membran Proteine eingelagert. Diese durchqueren die Membran und erzeugen Poren. Nur durch diese Kanäle ist es den Zellen möglich mit ihrer Umgebung zu interagieren. Membranproteine sind somit für Lebewesen ohnegleichen wichtig. Durch die enorme Vielfalt an unterschiedlichen Proteinstrukturen gibt es auch unzählige Möglichkeiten für die Morphologie an Membranproteinen. Somit ergeben sich auch unzählige Funktionen, die diese Proteine übernehmen. Zum einen agieren sie als Kanalproteine passiv im Stoffaustausch der Zelle, sie können aber auch als Pumpen- oder Carrierproteine aktiv in den Stoffaustausch eingreifen. Ebenso spielen Membranproteine eine wichtige Rolle als Oberflächenproteine und agieren somit als Rezeptoren und Signalproteine[?][?].
Zur Erkennung der jeweiligen Topologie eines Membranproteins gibt es zwei unterschiedliche Möglichkeiten. Eine davon sind in-vitro Verfahren. Diese experimentellen Laborverfahren sind sehr zeit- und kostenaufwendig und nicht für jedes Membranprotein durchführbar. Die zweite Möglichkeit sind in-silico Verfahren, welche die Topologie der Proteine am PC mit unterschiedlichsten Vorhersagealgorithmen bestimmen. Diese sind zwar nicht so genau wie die experimentelle Bestimmung, dafür aber kostengünstig und schnell.
Die Entwicklung derartiger Analysetools ist somit eine Hauptaufgabe der Bioinformatik. Es existieren hierbei die unterschiedlichsten algorithmischen Modelle. Zu den meist genutzten zählen hier, neben SVMs[1] und neuronalen Netzen, die Hidden-Markov-Modelle (HMM). Bestehende HMMs basieren auf den unterschiedlichen physiko-chemischen Eigenschaften der Aminosäuren. Das in der Arbeit behandelte Modell hingegen nimmt seine Informationen aus den Energieprofilen der Membranproteine und stellt somit eine völlig neue Herangehensweise dar.[?]

[1] Support Vector Machines

2 Grundlagen

2.1 Proteine

2.1.1 Aminosäuren

Proteine sind die Grundbausteine des Lebens. Sie geben den Lebewesen Struktur und die Fähigkeit unterschiedlichste Stoffe zu metabolisieren, und katalysieren die unterschiedlichsten Reaktionen. Außerdem können sie regulierend in den Stoffwechsel eingreifen und regeln den Stoffaustausch mit der Umgebung. Proteine sind lange Ketten aus Aminosäuremonomeren. Aminosäuren sind organische Säuren. Sie besitzen somit eine Carboxylgruppe. Zusätzlich zu dieser funktionellen Gruppe haben Aminosäuren noch eine Aminogruppe. Es exsistieren 20 kanonische, oder proteinogenen Aminosäuren. Diese sind in ihrem Grundaufbau, abgesehen von Prolin, identisch. An ein zentrales Kohlenstoffatom sind vier weitere Substituenten gebunden. Sie manifestieren sich als eine Aminogruppe, eine Carboxylgruppe, ein Wasserstoffatom und ein für jede kanonische Aminosäure spezifischer Rest. Im Falle von Glycin wäre dieser Rest ein Wasserstoffatom, bei Alanin eine Methylgruppe. Das zentrale Kohlenstoffatom wird auch als C_α-Atom bezeichnet (siehe Abbildung **??**).

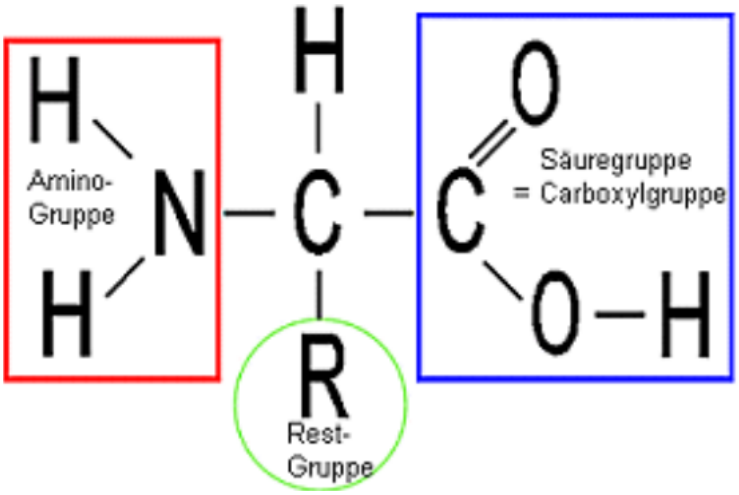

Abbildung 2.1: Grundaufbau einer Aminosäure: An das zentrale C_α-Atom ist links die Amino-, rechts die Carboxylgruppe, unten das Wasserstoffatom und mit R bezeichnet der individuelle Rest.[?]

Aminosäuren sind amphotere Verbindungen da sie sowohl als Bronstedt Säure als

auch als Bronstedt Base reagieren können. Den basischen Charakter bestimmt hierbei Aminogruppe, den Säurecharakter die Carboxylgruppe. Außerdem wird der physikochemnische Charakter einer Aminosäure durch ihre individuelle Restgruppe bestimmt. Damit lassen sich diese Verbindungen in unterschiedliche Gruppen clustern (siehe Abbildung **??**). Zur einfacheren Notation wurde für die 20 kanonischen Aminosäuren ein Einbuchstabencode und ein Dreibuchstabencode eingeführt. Dabei wird Aminosäure eineindeutig bezeichnet (siehe Tabelle: **??**). Besonders für die Informatik hat sich der Einbuchstabencode bewährt, da hier die gesammt Aminosäuresequenz eines Proteins als simpler Text-String[2] verarbeitet werden kann.

Tabelle 2.1: Die Tabelle zeigt die Übersetzung der Aminosäuren in ihren jeweiligen Ein-, und Dreilettercode

Name	Dreilettercode	Einlettercode
Alanin	Ala	A
Arginin	Arg	R
Asparagin	Asn	N
Asparaginsäure	Asp	D
Cystein	Cys	C
Glutamin	Gln	Q
Glutaminsäure	Glu	E
Glycin	Gly	G
Histidin	His	H
Isoleucin	Ile	I
Leucin	Leu	L
Lysin	Lys	K
Methionin	Met	M
Phenylalanin	Phe	F
Prolin	Pro	P
Serin	Ser	S
Threonin	Thr	T
Tryptophan	Trp	W
Tyrosin	Tyr	Y
Valin	Val	V

[2] Ein String ist unter Java ein komplexer Datentyp. Er beschreibt eine Zeichenkette.[?]

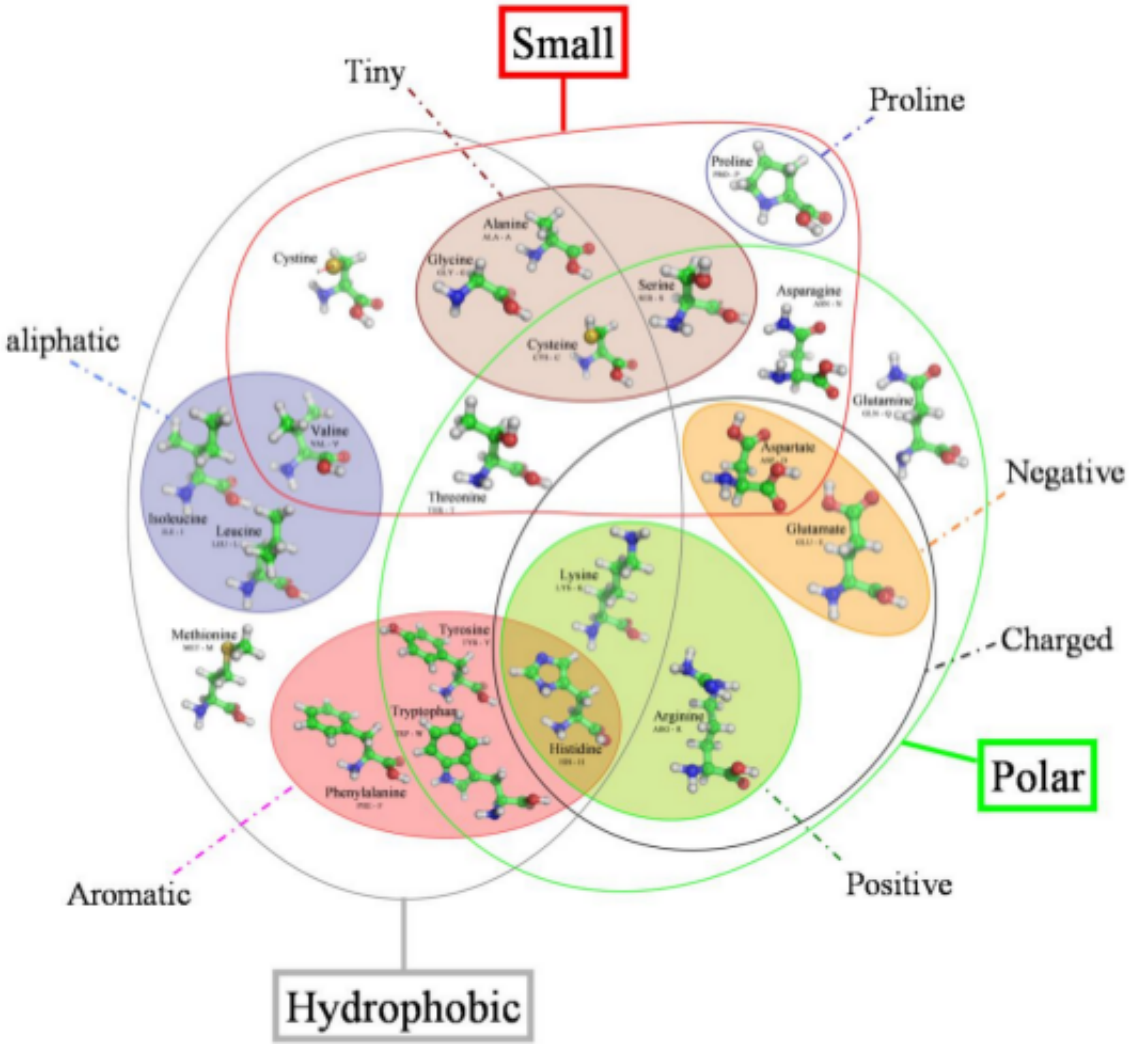

Abbildung 2.2: Aminosäuren geclustert nach ihren physikochemischen Eigenschaften[?]

2.1.2 Proteine

Proteine bestehen aus Aminosäuremonomeren. Durch die Kondensation der Carboxylgruppe einer Aminosäure mit der Aminogruppe einer weiteren Aminosäure bildet sich unter Entsehung eins Wassermoleküls eine Säureamidbindung zwischen beiden Aminosären aus, welche auch als Peptidbindung bezeichnet wird (siehe Abbildung **??**). Das entstandene Molekül ist ein Peptid und wird, da es nur aus zwei Aminosäuren besteht als Dipeptid bezeichnet. Dieser Vorgang ist beliebig oft wiederholbar. Das entstehende Polymer aus beliebig vielen Aminosäuremonomeren ergibt letztlich ein Protein. Jede Polypeptidkette besitzt bedingt durch die Polykondensationsreaktion ein Ende mit einer Aminogruppe und eines mit einer Carboxylgruppe. Dies verleiht dem Protein einen Richtungssinn. Das Carboxylende wird hierbei als C-Terminus bezeichnet, das Aminoende als N-Terminus.

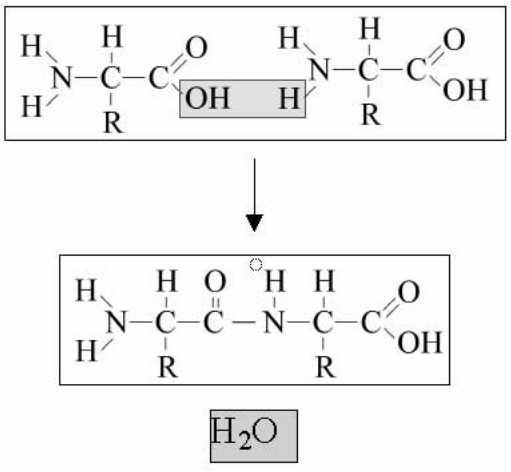

Abbildung 2.3: Diese Abbildung zeigt die Kondensationsreaktion zweier Aminosäuren zu einem Dipeptid[?].

Die unendlichen Kombinationsmöglichkeiten an Aminosäuremonomeren für ein Polypeptid sorgen für die große Funktions- und Formvielfalt von Proteinen. Hierbei bestimmen die Reste der Aminosäuren durch ihre unterschiedlichen physikochemischen Eigenschaften, die Eigenschaften des Proteins selbst. Prinzipiell kann zwischen zwei Arten von Proteinen unterschieden werden. Zum einen in globuläre Proteine. Diese befinden sich hauptsächlich im Cytoplasma. Globulär entstammt dem lateinischen globulus und bedeutet Kügelchen. Dies bezieht sich auf die häufig kugelförmige Gestalt dieser Proteinklasse. Diese erhalten sie vor allem dadurch, dass sich unpolare und damit hydrophobe Reste in Richtung Proteininnenraum wenden und geladene Reste nach außen zeigen. Die zweite Proteinklasse sind die bereits erwähnten Membranproteine. Durch ihre besonderen Funktionen im Stoffaustausch und als Rezeptor sind Membranproteine für Medizin besonders interessant.[?][?]

2.1.3 Proteinstrukturen

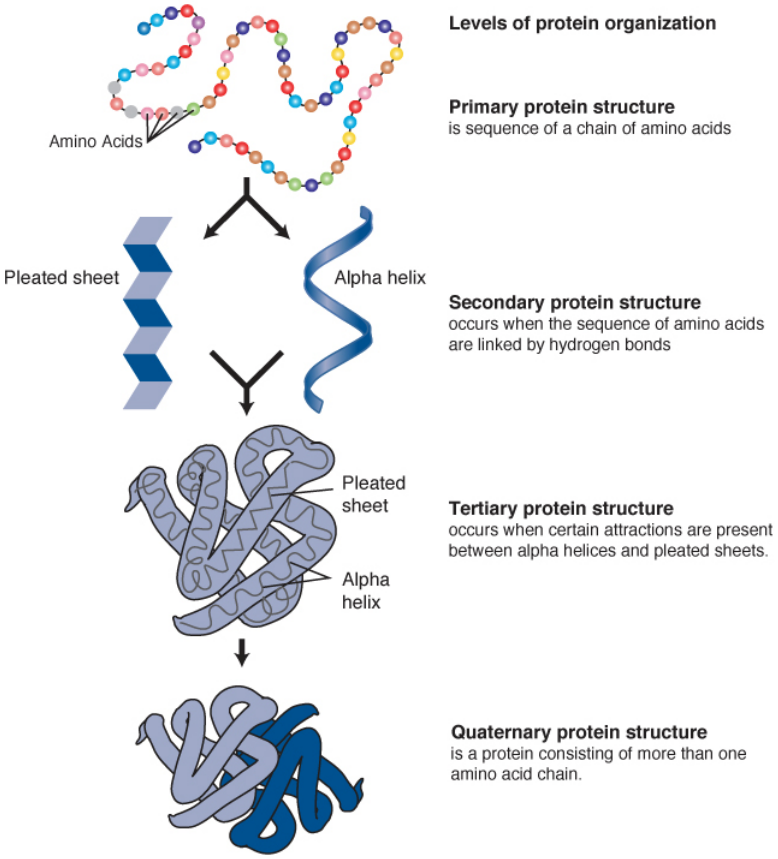

Abbildung 2.4: Diese Abbildung zeigt die einzelnen Organisatiosstrukturen einer Polypaptidkette. Beginnend mit der Pirmärstruktur als lineare Abfolge der Aminosäuremonomere über die Bildung von Helices und β-Faltblättern als Sekundärstruktur, deren Anordnung im Raum zur Tertiärstruktur, bis hin zur Quartärstruktur[?]

Proteine liegen in der Natur nicht als linearisierte Makromoleküle vor. Man unterscheidet zwischen vier Strukturebenen. Die Primärstruktur eines Proteins spiegelt die lineare Abfolge der einzelnen Aminosäuremonomere wieder. Durch Wasserstoffbrückenbindungen der Peptidgruppen und dem Bestreben hydrophober Reste dem wässrigem Milieu zu entgehen, faltet die Aminosäurekette in defnierte oder weniger definierte Formen im Raum. Dabei entstehen gewendelte Strukturen, sogenannte Helices, gestreckte Strukturen (β-Faltblätter) und wie bereits erwähnt, undefiniertere Formen, bezeichnte als coil. Dies bildet die Sekundärstruktur des Proteins. Durch unterschiedliche , nichtkovalente Wechselwirkungen lagern sich die Sekundärstrukturelemente zur Tertiärstrukur zusammen. Dabei werden die Wechselwirkungen hauptsächlich durch Wasserstoffbrückenbindungen, ionische, hydrophobe, und van-der-Waals-Kräfte induziert. Die vierte Gliederungsebene der Proteinstrukturen stellt die Quartärstruktur dar. Sie beschreibt

die Zusammenlagerung mehrerer Proteinuntereinheiten, deren Zusammenhalt ebenfalls durch nicht-kovalente Wechselwirkungen begründet wird (siehe Abbildung: **??**). Die morphologischen Möglichkeiten, welche sich aus den Strukurebenen ergeben, sind ein weiterer Grund für die hohe Variabilität an Funktionen von Proteinen[?].

2.1.4 Membranproteine

Wie bereits in Punkt **??** erwähnt sind Membranproteine für verschiedenste Forschungsgebiete von fundamentaler Bedeutung. Membranproteine sind Polypeptidketten, welche die Phospholipiddoppelmembran, die alle Zellen umgibt, durchdringen. Dabei können diese Proteine komplett oder auch nur zum Teil in der Membran liegen. Ebenso kann die Membran mehrfach vom Protein durchquert werden. Membranproteine erfüllen dabei unterschiedlichste Aufgaben. Zum einen dienen sie, wie bereits erwähnt als Kanalproteine und machen die Zellmembran permeabel für Ionen und größere Moleküle. Hierbei können sie passiv sein, aber auch aktiv in den Stofftransport eingreifen. Oberflächenproteine an denen zusätzlich Glykole und andere Stoffe gebunden sind dienen den Zellen als Rezeptor und ermöglichen den Zellen somit beispielsweise eine Freund-Feind Unterscheidung. So erkennen Zellen des menschlichen Immunsystems körpereigene Zellen am Major-Histocompatibility-Complex (MHC-Komplex) auf der Zellmembran aller Körperzellen. Außerdem können infizierte Zellen mittels dieses MHC-I-Komplexes Pathogene präsentieren und unterstützen somit die T-Zellen bei der Immunantwort[?][?].

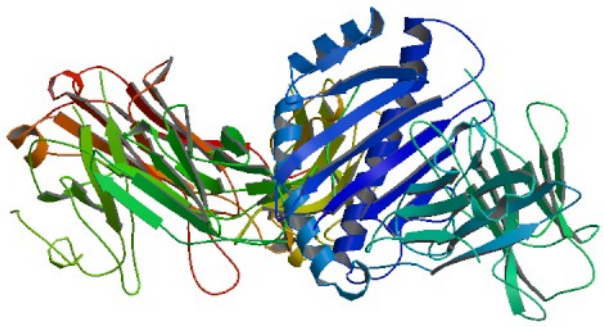

Abbildung 2.5: Diese Abbildung zeigt den T-Zellrezeptor eines MHC-I Proteinkomplexes (PDB_ID 1BD2)

Im Gegensatz zu globulären Proteinen sind nur sehr wenige Sekundär- beziehungsweise Tertiärstrukturen von Membranproteinen bekannt. Dies liegt hauptsächlich daran, dass die üblichen Methoden zur Strukturbestimmung wie beispielsweise NMR und Kristallröntgenspektroskopie für Membranproteine nicht verwendbar sind[?]. Zu begründen ist dies mit dem Umstand, dass die Proteine hierfür in Kristallform vorliegen müssen. Membranproteine lassen sich aber aufgrund ihrer Morphologie nicht adäquat kristallisieren, da sie in zwei verschiedenen Phasen existieren. Der transmembrane Bereich befindet sich in einer stark hydrophoben Phase, der nicht-transmembrane Bereich in einer stark hydrophilen Phase. Dieser Umstand beschreibt sehr gut die besonderen Eigenschaften von Membranproteinen. Da sie in zwei verschiedenen Milieus existieren müssen auch die Aminosäuren in ihren Primärstrukturen den unterschiedlichen Lösungsmittel entsprechen. So finden sich in transmembranen Bereichen hauptsächlich hydrophobe Aminosäuren und in nicht-transmembranen Bereichen hydrophile Aminosäuren. Außerdem ist zu beobachten, dass die Zellmembran nur von bestimmten Sekundärstrukturelementen durchquert wird. Dies sind zum einen α-Helices oder β-Faltblätter. Lagern sich viele β-Faltblätter zusammen so entstehen fassähnliche Strukturen, welche als Poren fungieren solche Strukturen werden als β-Barrels bezeichnet (siehe Abbildung **??**)[?].

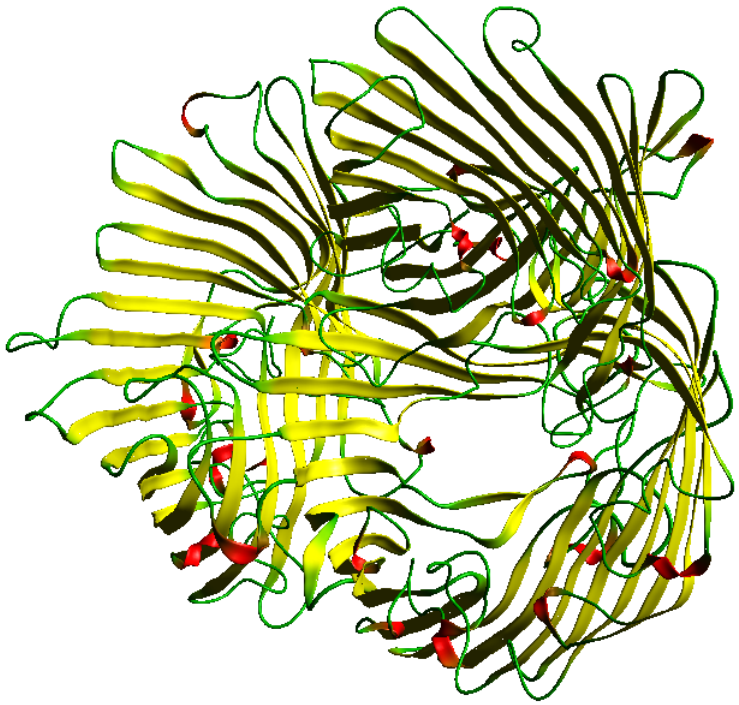

Abbildung 2.6: Diese Abbildung zeigt ein Saccharose spezifisches Porin (PDB_ID 1A0S). Gelb eingefärbt sind β-Faltblätter, welche sich zu β-Barrels zusammenlagern und somit eine Pore schaffen.

Ebenso können Poren aus der Interaktion von Helices hervorgehen.
Bei der Betrachtung von Membranproteinen ist es hauptsächlich von Bedeutung deren Topologie zu kennen. Die Topologie gibt die Lage des Proteins bezüglich seiner Orientierung zur Zellmembran an[?]. Es wird also im allgemeinen unterschieden zwischen intrazellulär, extrazelulär und transmembran. Das Wissen darüber legt bei der Erforschung neuer Membranproteine den Grundstein für das Erkennen von Funktionalitäten[?]. So können beispielsweise Veränderungen in der Primärstruktur, welche sich innerhalb eines transmembranen Bereiches befinden, Aufschluss über die fortbestehende Funktionsfähigkeit des Proteins liefern. Ebenso können für nicht-transmembrane Bereiche eventuelle Bindungstellen für andere wechselwirkende Stoffe identifiziert werden.
Das Auffinden von Topologien geschieht experimentell mit Hilfe eines Atomkraftmikroskops. Hierbei wird das Molekül mechanisch entfaltet und die Kraft gemessen, welche zur Entfaltung nötig ist. Hierbei schnellt die Kraftkurve bei jedem membranständigen Teil nach oben, da für das auslösen aus der Zellmembran mehr Kraft nötig ist als für das normale entfalten des Proteins[?]. Diese Prozeduren sind in der Regel recht zeitaufwendig. Die Bioinformatik bedient sich daher unterschiedlicher Tools zur Vorhersage der Topologie in-silico. Hierbei finden, wie bereits erwähnt, verschiedene Modelle zur Vorhersage Anwendung. Die meisten basieren dabei auf den unterschiedlichen physiko-chemischen Eigenschaften der Aminosäuren zur Determinierung der jeweiligen Topologie. Dabei liegt bei Membranproteinen das Hauptaugenmerk auf den Hydrophobizitätswerten der einzelnen Aminosäuren. Aminosäuren mit unpolaren und möglichst kleinen Seitenketten würden sich innerhalb der stark hydrophilen Umgebung innerhalb beziehungsweise außerhalb der Zelle ins innere des Proteins richten. Da sich Gleiches in Gleichem löst, sind diese Reste bestrebt, sich in stabilen Formen innerhalb einer stark hydrophoben Umgebung einzulagern. Eine solche Umgebung bietet die Zellmembran. Sie ist für hydrophobe Reste der energetisch günstigste Ort um in Lösung zu gehen. Ebenso verhält es sich mit hydrophilen Resten außerhalb der Membran. So kann also anhand einer Häufung von hydrophoben Aminosäuren auf einen transmembranen Bereich geschlossen werden.
Ein neuer Ansatz für die Vorhersage betrachtet nun nicht mehr nur einen einzelnen physiko-chemischen Aspekt, sondern verwendet alle Wechselwirkungen die ein Protein ausmachen in der komprimierten Form eines Energieprofils.

2.2 Energieprofie[?]

Energieprofile beschreiben die 3D-Struktur eines Proteins sowie alle Wechselwirkungen seiner Monomere und brechen diese in einen eindimensionalen Vektor herunter. Es erklärt sich somit die Energie eines gefalteten Proteins durch die Wechselwirkungen der einzelnen Residues[3] mit ihrer Umgebung. Aus diesem Ansatz lässt sich folgende Formel ableiten :

$$E_{Ges} = \sum_{ij} \left(e^*_{ij} f(r_{ij})\right) * \sum_{i} e^{'}_{i0} g(i) \tag{2.1}$$

Die Gesamtenergie (E_{Ges}) berechnet sich demnach aus der Summe der Wechselwirkungsenergien (e^*_{ij}) zwischen Aminosäure i und j, relativiert durch deren Abstandsfunktion ($f(r_{ij})$), multipliziert mit der Summe aus den Wechselwirkungsenergien der Aminosäure mit ihrer Umgebung ($e^{'}_i 0$). Zum Bestimmen der Abstandsfunktion $f(r_{ij})$ ist es nötig einen Fixpunkt zu wählen. Hierfür werden die-C_β-Atome der Seitenketten in Betracht gezogen. Diese sind relativ starr mit dem Backbone verbunden und weisen keine starken strukturellen Divergenzen auf. Zur Beschreibung der Aminosäure-Lösungsmittel-Wechselwirkung wird die Hydrophobizität der Reste betrachtet. Hierbei streben hydrophobe Reste aufgrund der umgebenden wässrigen Lösung ins Innere des Proteins. Hydrophile Reste hingegen streben auf die Außenseite des Proteins um mit ihrer Umgebung Wechselwirkungen und somit beispielsweise Wasserstoffbrückenbindungen einzugehen. Dieser Zustand beschreibt ein Innen-/Außenkriterium und wurde in der Formel **??** durch $g(i)$ definiert. Zur Definition des Proteininnen- beziehungsweise -außenraums wird um den Schwerpunkt c eine Kugel mit dem Radius 10 Å gelegt.
Es gilt eine Residue als innen liegend und $g(i) = 1$ wenn folgende Bedingung erfüllt ist:

$$|C_\alpha - c| < 5 \vee \left(C_\alpha - C_\beta\right)(C_\alpha - c) < 0 \tag{2.2}$$

Andernfalls ist sie als außen liegend mit $g(i) = 0$ definiert. Es wurde für ein Set von Membranproteinen das Innen-/Außenkriterium berechnet (siehe Tabelle **??**). Hiermit kann nun die Lösungsenergie $e^{'}_{i0}$ bestimmt werden. Dies erfolgt nach folgender Formel mit Hilfe der Boltzmann-Konstante:

$$e^{'}_{i0} = -k_B T \ln\left(\frac{n_{in}}{n_{out}}\right) \tag{2.3}$$

[3] Reste beziehungsweise Aminosäuremonomere

Zur Berechnung von e^*_{ij} kam analog zu Formel **??** folgende Formel zum Einsatz:

$$e^*_{ij} = -k_B T \ln\left(\frac{n_{ij}}{N_{contact} p_i p_j}\right) \tag{2.4}$$

Hierbei Beschreibt n_{ij} die Anzahl der Wechselwirkungen von Residue i mit Residue j. Daraus ergibt sich die Gesamtzahl aller Kontakte $N_{contacs}$ und die Wahrscheinlichkeiten p_i und p_j.

Tabelle 2.2: Diese Tabelle zeigt das statistische Auftreten der Innen/Außen Bewertung für alle 20 kanonischen Aminosäuren[?]

Aminosäure	Anzahl Innen	Anzahl Außen
CYS	4582	1016
ILE	20370	4141
SER	12576	10411
GLN	7373	7752
LYS	9285	15193
ASN	8225	8928
PRO	9135	9423
THR	12537	9622
PHE	13353	2813
ALA	22725	11052
HIS	6419	3366
GLY	16698	14326
ASP	10001	14327
LEU	30615	7107
ARG	11327	10441
TRP	4001	1193
VAL	23562	6551
GLU	11165	18091
TYR	11228	3529
MET	7003	1723

Zur Berechnung der reinen Paarpotentiale erfolgt noch die Umformung:

$$e_{i0} = \left(\frac{1}{i}\alpha_i\right) e'_{i0} \tag{2.5}$$

α_i beschreibt hierbei alle zur Residue i in Kontakt stehenden Aminosäuren j. Kontakt ist dabei definiert durch $r_{ij} < 8$ Å. Daraus folgt für das einfache Paarpotential:

$$e_{ij} = e_{i0} + e_{j0} + e^*_{ij} \tag{2.6}$$

Formel **??** spiegelt nun die Berechnung für das Energieprofil eines globulären Proteins wider. Für die Berechnung eines Membranproteinenergieprofils ist diese Formel nicht anwendbar, da nur sehr wenig 3D-Strukturen von Membranproteinen vorhanden sind. Daher würde der Ansatz über Paarpotentiale eine sehr schlechte Statistik liefern, weshalb bei der Berechnung von Energieprofilen für Membranproteine auf die Paarpotentiale verzichtet wird. Dabei ergibt sich aus Formel **??**:

$$e_{ij} = e_{i,mem} + e_{j,mem} \tag{2.7}$$

Die 3D-Struktur wird durch die Wiedergabe der Energie jedes einzelnen Restes in einen zweidimensionalen Vektor überführt. Damit enthält ein Energieprofil alle physikochemischen Eigenschaften eines Proteins, kondensiert in ein informatisch verarbeitbares Modell. Energieprofile sind für jedes Protein unique und spiegeln somit den energetischen Fingerprint eines Proteins wider. Damit ist es für Vorhersagemodelle bestens geeignet. Mit Hilfe dieses Werkzeugs soll nun ein solches Modell zur Bestimmung der Topologie von Membranproteinen erstellt werden, da hiermit erstmals alle physikochemischen Eigenschaften eines Proteins in die Vorhersage eingebunden werden können.

2.3 Hidden Markov Modelle

2.3.1 Markov Ketten

Abbildung 2.7: Dieses Bild zeigt Andrei Andreijewitsch Markov, den Begründer der Markov Ketten.[?]

Die Basis des Topologievorhersagemodells stellt ein Hidden Markov Modell dar. Diese Modelle beruhen auf den von Andrei Andreijewitsch entwickelten Markov Ketten. Eine Markov Kette ist ein stochastisches Modell, welches unterschiedlichste Zustandsänderungen eines Systems beschreibt. Zustände sind dabei als Elemente Z eines diskreten Alphabets Σ an Zeichen definiert (siehe Abbildung **??**). Man schreibt hierfür $Z \in \Sigma$. Es wird nun die Übergangswahrscheinlichkeit des Zeichens Z_i zum Zeichen Z_{i+1} beschrieben. Jeder folgende Zustand Z_{i+1} ist dabei nur abhängig von seinem voraus gehenden Zustand Z_i. Die Zustände am Start einer Markov Kette werden als Initialzustände bezeichnet und unterliegen einer eigenen Wahrscheinlichkeitsverteilung.

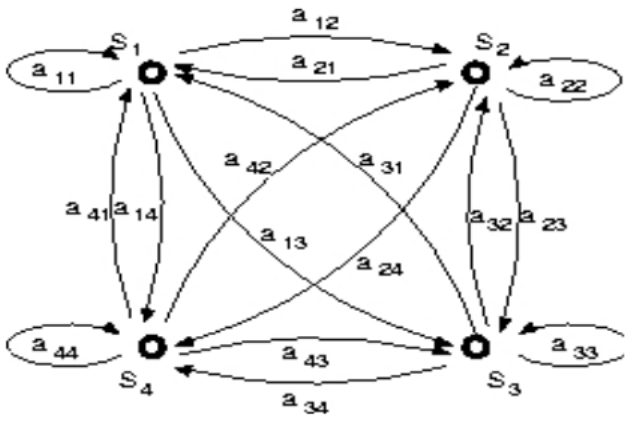

Abbildung 2.8: Die Abbildung zeigt eine Markov-Kette mit vier verschiedenen Zuständen S1-S4. Jede Linie verdeutlicht den Übergang in den jeweils anderen Zustand. Die entsprechenden Übergangswahrscheinlichkeiten a sind mit den Zuständen indexiert. So steht a_{12} für den Wechsel von S1 auf S2.

Die Übergangswahrscheinlichkeiten von Zustand Z_i in Z_{i+1} werden formal als p_{ij} beschrieben, wobei j den Term $i+1$ widerspiegelt. Definiert sind die Übergangs- oder Transitionswahrscheinlichkeiten durch die bedingte Wahrscheinlichkeit p_{ij}. Sie beschreibt die Wahrscheinlichkeit des Auftretens von Zeichen Z_j unter der Bedingung, dass zuvor das Zeichen Z_i beobachtet wurde. Die bedingte Wahrscheinlichkeit berechnet sich nach folgender Formel:

$$P(Z_j|Z_i) = p_{ij} = \frac{P(Z_i \cap Z_j)}{P(Z_j)} \tag{2.8}$$

Der große Vorteil für den Nutzer liegt bei Markov Ketten an ihrer Einfachheit und Überschaubarkeit und daran, dass die Wahrscheinlichkeitsberechnung für lange Ketten von Zustandsänderungen sehr schnell geschehen kann, weil jeder Zustand nur von seinem jeweils vorausgehenden Zustand abhängt. Dies gilt zumindest für Markov Ketten erster Ordnung. Die Übergangswahrscheinlichkeiten eines Markov Modells werden zur Vereinfachung in einer Übergangswahrscheinlichkeitsmatrix oder Transitionsmatrix dargestellt. Hierbei findet sich die Übergangswahrscheinlichkeit für p_{ij} in der i-ten Zeile und j-ten Spalte der Matrix. Unmögliche Übergange im Modell erhalten den Wahrscheinlichkeitswert $p_{ij} = 0$, Zustände die nur einen Wechsel zu einem bestimmten Zustand zulassen erhalten den Wahrscheinlichkeitswert $p_{ij} = 1$.

$$(p_{ij}(t)) = \mathbf{M} = \begin{pmatrix} p_{11}(t) & \dots & p_{1m}(t) \\ \vdots & \ddots & \vdots \\ p_{m1}(t) & \dots & p_{mm}(t) \end{pmatrix}$$

Abbildung 2.9: Diese Abbildung stellt schematisch eine Transitionsmatrix dar. Sie hat die Größe i*j, die Übergangswahrscheinlichkeit für p_{ij} findet sich in Zeile i Spalte j.[?]

Grundsätzliche kann zwischen verschiedenen Formen von Markov Ketten unterschieden werden. Hauptkriterium dabei ist die Zugänglichkeit der Stati im Modell. So werden Markov Ketten als irreduzibel bezeichnet, sollte es möglich sein jeden Punkt im Modell in einer endlichen Anzahl an Schritten zu erreichen. Absorbierende Markov Ketten hingegen besitzen Zustände die deterministisch für den weiteren Verlauf der Kette sind, da sie keine Übergänge in andere Stati zulassen. Sie enthalten also Übergangswahrscheinlichkeiten mit $p_{ij} = 1$. Zustände, welche keine Übergänge in andere Zustände zulassen, außer in sich selbst, werden als absorbierende Zustände bezeichnet. Es gilt ein Zustand als absorbierend wenn:

$$p_{ij} = p_{ji} = 1 \tag{2.9}$$

Wenn aber gilt:

$$p_{ij} = 1 \wedge p_{ji} = 0 \tag{2.10}$$

So bezeichnet man den Zustand als transient.

2.3.2 Hidden Markov Modelle

Einfache Markov Ketten lassen die Möglichkeit zu, die Wahrscheinlichkeit des Auftretens einer Sequenz an Zeichen zu bestimmen. Vergleichend ist dies ein praktikabler Anhaltspunkt zur Entscheidung welche Sequenz eventuell biologische Relevanz hat. Allein zur Vorhersage einer Sequenz an Zeichen beziehungsweise Zuständen eignen sich Markov Ketten nicht.
Hidden Markov Modelle erweitern Markov Ketten um ein zusätzliches Alphabet an Zuständen. Während bei Markov Ketten das Alphabet aus einsehbaren oder beobachtbaren Zuständen besteht, handelt es bei dem erweiterten Alphabet im Hidden Markov Modell (HMM) um versteckte (engl. hidden) Zustände. Diese Zustände werden auch als interne Zustände bezeichnet. Es kann nun zu jedem Element Z aus dem Alphabet oder Zustandsraum Σ ein Element aus dem erweiterten Zustandsraum Σ_{HMM}, welcher die

versteckten Zustände enthält, zugeordnet werden. Hierbei spielen erneut Übergangswahrscheinlichkeiten eine Rolle. Sie definieren die Wahrscheinlichkeit des Auftretens eines Zeichens aus dem versteckten Zustandsraum Σ_{HMM} unter der Bedingung, dass ein bestimmtes Zeichen aus dem beobachtbaren Zustandsraum Σ aufgetreten ist. Es wird somit eine Emission beschrieben. Daher wird diese Übergangswahrscheinlichkeit auch als Emissionswahrscheinlichkeit bezeichnet.

$$e_k(b) = P(x_i = b|\pi_i = k) \tag{2.11}$$

Die Emissionswahrscheinlichkeit $e_k(b)$ eines Zeichens $b \in \Sigma$ unter der Bedingung, dass das Zeichen $k \in \Sigma_{HMM}$ vorhanden ist, berechnet sich demnach aus der bedingten Wahrscheinlichkeit für das Zeichen x zum Zeitpunkt i aus Σ unter der Bedingung, dass Zeichen π zum Zeitpunkt i aus Σ_{HMM} eingetreten ist (siehe Formel **??**).
Hieraus folgt, dass für jede Sequenz $x = x_1, x_2, x_3, ..., x_l$ der Länge l mit Elementen aus Σ, eine ebenso lange Sequenz $\pi = \pi_1, \pi_2, \pi_3, ..., \pi_l$ mit Elementen aus Σ_{HMM} existiert. Die Abbildungsvorschrift dafür liefert $e_k(b)$.

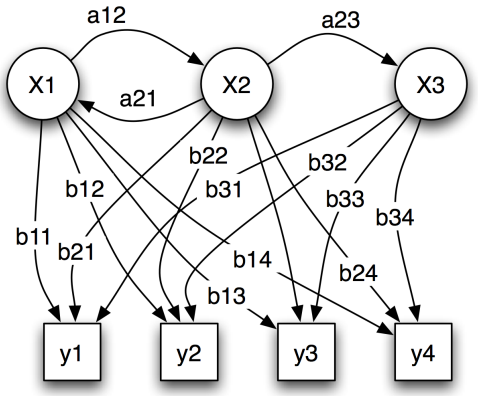

Abbildung 2.10: Diese Abbildung zeigt ein dreistufiges HMM. Die Übergangswahrscheinlichkeiten der beobachtbaren Zustände x_i sind mit a_{ij} gekennzeichnet. Die Elemente aus Σ_{HMM} sind mit y_i bezeichnet. Ebenso sind die Emissionswahrscheinlichkeiten $e_k(b)$ mit den Bezeichnungen b_{ij} zu sehen[?].

Damit ergibt sich die Möglichkeit, aus bekannten Übergangswahrscheinlichkeiten für den beobachtbaren Zustandsraum und den bekannten Emissionswahrscheinlichkeiten, bei gegebener Sequenz x, auf die versteckte Sequenz π zu schließen.
Es muss demnach möglich sein, aus einer bekannten Sequenz an Energiewerten für ein Membranprotein auf dessen Topologiesequenz zu schließen, wenn man annimmt,

dass jeder topologische Status ein Symbol in Σ_{HMM} widerspiegelt.

2.3.3 Training des HMM - Baum-Welch-Algorithmus[?]

Wie bereits erwähnt ist es nötig, die Übergangswahrscheinlichkeiten des HMMs zu ermitteln um ein Vorhersagetool zu erhalten. Der Grundsatz zum Erstellens eines solchen Modells beruht auf der Analyse von Daten, welche die gewünschten Informationen (also bekannte zusammengehörige Sequenzen x und π) enthalten. Aus diesen Daten werden die Übergangswahrscheinlichkeiten für das HMM bestimmt. Man bezeichnet den Vorgang auch als Training des Modells. Hierfür existieren unterschiedliche Algorithmen. Da in der verwendeten Software der Baum-Welch-Algorithmus implementiert ist, soll dieser im Folgendem näher erläutert werden.
Der genannte Algorithmus optimiert ein gelerntes bestehendes Modell und ermöglicht somit eine Verbesserung der Vorhersagegenauigkeit. Als Optimierungsgrundlage dient die Gesamtprodutionswahrscheinlichkeit $P(O|\lambda)$. Es wird dabei ein bestehendes Modell λ, abhängig von bestimmten Beispieldaten O, so optimiert, dass das optimierte Modell $\hat{\lambda}$ die Daten des Trainingsdatensatzes mit größerer oder gleicher Wahrscheinlichkeit generiert:

$$P(O|\hat{\lambda}) > P(O|\lambda) \tag{2.12}$$

Sollte mit dem Modell λ ein lokales Maximum der Optimierung unter Beachtung aller möglichen Modelle erreicht worden sein, so gilt analog zu Formel **??**:

$$P(O|\hat{\lambda}) = P(O|\lambda) \tag{2.13}$$

Bezogen auf das gegebene Ausgangsmodell λ und die Trainingsdaten O werden alle Modellparameter ersetzt. Der Baum-Welch-Algorithmus optimiert somit die Parameter eines mehrstufigen Zufallsprozesses mit versteckten Zustandsvariablen nach einem Maximum-Likelihood-Kriterium. Vorwärts- und Rückwärtsvariablen im statistischen Sinne, lassen Rückschlüsse auf die internen Abläufe des Modells λ in Hinsicht auf die Generierung der vorliegenden Daten O zu. Diese Größen sind das Fundament des Verfahrens. Zusätzlich zur a-posteriori[4] Wahrscheinlichkeit $\gamma_t(i) = P(S_t = i|O,\lambda)$[5] werden ebenso die a-posteriori Wahrscheinlichkeiten für die Zustandsübergänge benötigt. Es werden somit die für das Modell benötigten Wahrscheinlichkeiten berechnet. Aufbauend auf die bestimmten γ_t können im Anschluss die diskreten Emissionsverteilungen für $\hat{\lambda}$ berechnet werden. Zur Berechnung der verbesserten Schätzwerte $\hat{a}_{ij}$ nutzt man

[4] a-posteriori beschreibt die Wahrscheinlichkeit für das eintreten eines Zustandes A_i aus einer Sequenz S unter der Bedingung, dass B eingetreten ist. Formal: $P(A_i|B)$ [?]

[5] $\gamma_t(i)$ gibt die Wahrscheinlichkeit des Auftretens von Zustand i zum Zeitpunkt t an

die im statistischen Mittel zu erwartende Anzahl an Zustandsübergängen von i nach j, normiert über die Gesamtzahl von Übergängen aus Zustand i. Diese ergeben sich aus der Summe der Einzelübergangswahrscheinlichkeiten $\gamma_t(ij)$ über alle möglichen Zeitpunkte t.

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \gamma_t(ij)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{2.14}$$

Ebenso werden die verbesserten Startwahrscheinlichkeiten $\hat{\pi}_i$ bestimmt:

$$\hat{\pi}_i = \gamma_t(i) \tag{2.15}$$

Analog hierzu ergeben sich die verbesserten Emissionswahrscheinlichkeiten. Hierzu wird zunächst die Anzahl an Emissionen eines Symbols o_k im Zustand i berechnet. Dies geschieht durch die Forderung der Übereinstimmung von o_k mit dem entsprechenden Element der Observationssequenz. Schätzwerte für die erwarteten diskreten Emissionswahrscheinlicheiten $\hat{b}_j(o_k)$ ergeben sich aus der Normierung über die Gesamtzahl an Emissionen ausgehend von Zustand j.

$$\hat{b}_j(o_k) = \frac{\sum_{t:O_t=o_k} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{2.16}$$

Der iterative Lernprozess des Baum-Welch-Algorithmus entspricht hierbei der Anwendung der Schätzgleichungen (**??**), (**??**) und (**??**). Es findet nach jeder Iteration ein Abgleich nach Formel **??** statt, bis $\hat{\lambda}$ eine hinreichende Beschreibungsgüte erreicht hat.

2.3.4 Viterbi-Decodierung[?]

Nach dem Training des Modells ist es der letzte Schritt, eine Eingabesequenz zu analysieren und für diese Vorhersagen zu treffen.
Dabei müssten für jedes Element der Zeichenkette die Emissionswahrscheinlichkeiten bestimmt und untereinander abgeglichen werden. Dies hätte einen exponentiellen Rechenaufwand, sowohl im Hinblick auf die Sequenzlänge als auch auf die Zustände in den Zustandsräumen, zur Folge.
Der Viterbi-Algorithmus bestimmt zu einer Beobachtungssequenz X die wahrscheinlichste Emissionssequenz π^*. Mathematisch betrachtet ist dies ein relativ triviales Problem:

$$\pi^* = argmax_\pi = P(X, \pi) \tag{2.17}$$

Dabei beschreibt $argmax_\pi$ die Funktion, welche das Argument zum Maximum aller $P(X, \pi)$ liefert. Die Notation $P(X, \pi)$ ist dabei als eine Menge in Bezug auf π zu verstehen. Um das gegebene Problem zu lösen findet nun der Viterbi-Algorithmus Anwen-

dung. Der Hauptansatz liegt darin, dass jedes Segment $\pi_1^*, \pi_2^*, \pi_3^*, ..., \pi_n^*$ des optimalen Pfades durch das Modell unabhängig von seinem Folgesegment $\pi_n^*, \pi_{n+1}^*, ..., \pi_L^*$ gesucht werden kann. Voraussetzung ist, dass die beiden Segmente an der Stelle π_n^* übereinstimmen. Hieran zeigt sich der rekursive Charakter des Algorithmus, der es ermöglicht den Pfad π^* in einzelne Teilprobleme zu zerlegen und diese Schritt für Schritt zu lösen. Fundament des Algorithmus ist die Viterbi-Variable $v_k(i)$, wobei $k \in \Sigma_{HMM}$ und i die Position des Zustandes in der Sequenz beschreiben. Der Algorithmus setzt sich nun aus der Bildungsvorschrift für $v_k(i)$ und einem Rekursionsschema zur Auswertung zusammen.

$$v_l(i+1) = e_l(x_{i+1}) max_k\{v_k(i)a_{kl}\} \tag{2.18}$$

Diese Gleichung stellt den Zusammenhang zwischen der Viterbi-Variable an Stelle i und $i+1$ her. Es handelt sich dabei nicht nur um eine Variable, sondern um eine Liste an Variablen für jeden Zustand l des HMMs. Die Maximierungsfunktion max_k bezieht sich auf eben diese Liste $\{v_k(i)a_{kl}\}$. Sie spiegelt das Produkt der vorausgehenden Viterbi-Variablen zum Zustand k des Hidden-Markov-Modells an der Stelle i und des Übergangs von Zustand k auf l wider. Die Emissionswahrscheinlichkeiten $e_l(x_{i+1})$ liefern den Zusammenhang zwischen beobachteten und verborgenem Zustand des Hidden-Markov-Modells. Formal gliedert sich der Ablauf des Algorithmus in vier Schritte.

1. Initialisierung
2. Rekursion
3. Terminierung
4. Traceback

Die Initialisierung startet mit der Einführung eines Anfangszustandes 0. Dieser Zustand wird als Teil von Σ_{HMM} definiert. Der Start der Rekursion setzt damit alle Viterbi-Variablen für i gleich 0.

$$v_o(0) = 1, v_k(0) = 0, \forall k \in \Sigma_{HMM} \tag{2.19}$$

Der Wert für die Viterbi-Variable an der Stelle $i+1$ ergibt sich nun aus der zuvor berechneten Viterbi-Variable aus dem i-ten Rekursionschritt. Nun wird die Liste $\{v_k(i)a_{kl}\}$ erstellt und daraus das maximale Argument k entnommen. Dieser Wert wird mit der Emissionswahrscheinlichkeit für das $i+1$te Symbol der beobachteten Sequenz X, also $e_l(x_{i+1})$, multipliziert. Das ermittelte k wird schließlich in der Zeigervariable $Z_i(l)$ abgelegt. Die Terminierung der Rekursion erfordert erneut eine besondere Berechnung für die Viterbi-Variable. Es wird ein Übergang des letzten Symbols $k \in \Sigma_{HMM}$ auf dem Pfad π in einen Nullzustand angenommen. Die Übergangswahrscheinlichkeit für alle

Elemente aus Σ_{HMM} in den Nullzustand a_{k0} wird für alle als gleich groß angenommen. Sie berechnet sich demnach wie folgt:

$$a_{k0} = \frac{1}{|\Sigma_{HMM}|}; \forall k \in \Sigma_{HMM} \tag{2.20}$$

Die eigentliche Terminierung erfolgt formal nach folgender Formel:

$$max_k v_k(L) a_{k0} \tag{2.21}$$

Nach dem Abschluss der Rekursion muss nun nur noch im Tracebackverfahren der wahrscheinlichste Pfad π^* bestimmt werden. Dazu wird die Rekursion am Index i des Pfades zurückverfolgt. Die Zeigervariable $Z_i(l)$ liefert am Index i den wahrscheinlichsten Zustand $k \in \Sigma_{HMM}$ des Pfades π^* (siehe Formel).

$$\pi_i^* = Z_{i+1}(\pi_i^*) \tag{2.22}$$

2.4 e^{3}TMR

e^{3}TMR ist die bestehende Implementierung eines HMMs, welches anhand von Energieprofilen eine Vorhersage über die Topologie eines Membranproteins treffen kann. Es handelt sich hier um ein dreistufiges Hidden-Markov-Modell. Es unterscheidet sich insofern von allen anderen bekannten Vorhersagemodellen für Topologien, weil es als Beobachtungssequenz eine Sequenz aus Energiewerten erhält, während ähnliche Modelle (so zum Beispiel das TMHMM vgl. Abschnitt **??**) die Hydrophobizitätswerte der Aminosäuren als Ausgangspunkt für die Vorhersage verwenden. Der große Nachteil davon ist, dass nicht alle physiko-chemischen Eigenschaften, welche die Konformation und Topologie eines Membranproteins bestimmen, Beachtung finden. So werden bei der Verwendung von Hydrophobizitätswerten van-der-Waals-Kräfte oder ionische Wechselwirkungen gänzlich außer Acht gelassen. Daher sollte ein energiebasiertes Modell weitaus bessere oder zumindest vergleichbar gute Ergebnisse erzielen. Das bereits etablierte e^{2}TMR-Modell zeigte, dass mit Hilfe von Energieprofilen die Topologievorhersage mit Unterscheidung zwischen transmembranen und nicht-transmembranen Bereichen gute Vorhersageergebnisse erreichbar sind.

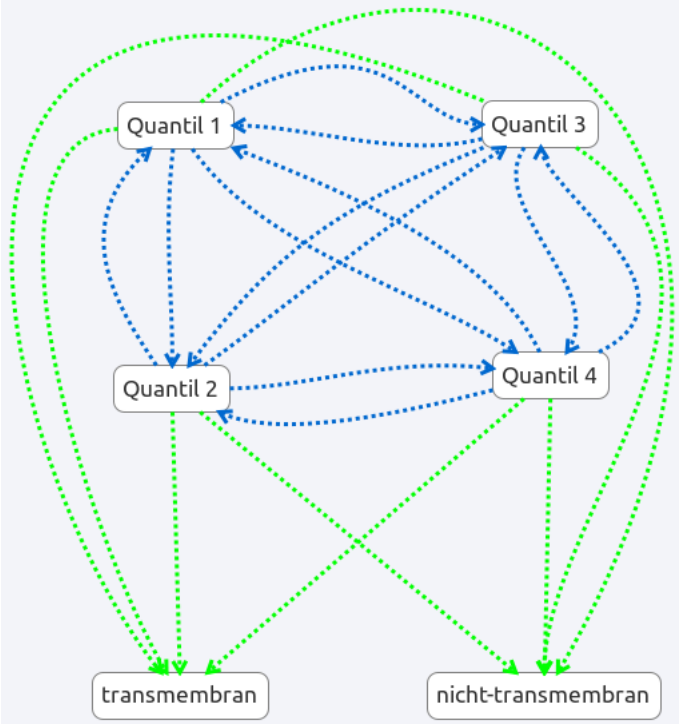

Abbildung 2.11: Diese Abbildung stellt den schematischen Aufbau des e^{2}TMR-Modells dar. Übergange der beobachtbaren Zustände sind als blaue Linien dargestellt, emittierte Zustände über grüne Linien.

Die Erweiterung dieses Modells um einen weiteren Status resultierte letztlich im bereits genannten e^{3}TMR-Modell, welches die Grundlage dieser Arbeit bildet. Das e^{3}TMR-Modell besitzt einen internen Zustandsraum Σ_{HMM} der Größe drei. Dieser enthält die Elemente $\{t\} \in \Sigma_{HMM}$ für transmembrane Bereiche, $\{i\} \in \Sigma_{HMM}$ für intrazelluläre Bereiche und $\{o\} \in \Sigma_{HMM}$ für extrazelluläre Bereiche. Der beobachtbare Zustandsraum enthält die Energiewerte die eine Aminosäure annehmen kann. Die Verwendung der Energiewerte ist dabei nicht trivial, da jeder Energiewert aus einem Energieprofil eine unterschiedliche Ausprägung annehmen kann. Zwar bewegen sich die Energiewerte in festen Grenzen, dazwischen jedoch sind sie variabel. Das HMM kann jedoch nur mit diskreten Werten als Zustandsraum verfahren. Daher mussten die Energiewerte aus den Energieprofilen zunächst noch normiert werden um für das HMM verwendbar zu sein. Hierfür wurden die Energien in Quantile übersetzt. Da unterschiedliche Datensätze mit einer unterschiedlichen Anzahl an Quantilen erstellt wurden, ist eine direkte Definition der Größe von Σ nicht möglich. Es bewegt sich allerdings in einem Bereich zwischen $0 \leq \Sigma \leq Quantile$. Dies ist ebenso eine Neuerung zum e^{2}TMR-Modell, da hier die Anzahl der Quantile auf 4 festgesetzt war. Der Vorteil liegt in einer dynamischeren Modellierung und somit einer flexibleren Anpassung zur Verbesserung der Vorhersagegenauigkeit. Das Modell verwendet als Trainingsalgorithmus den Baum-Welch-Algorithmus mit

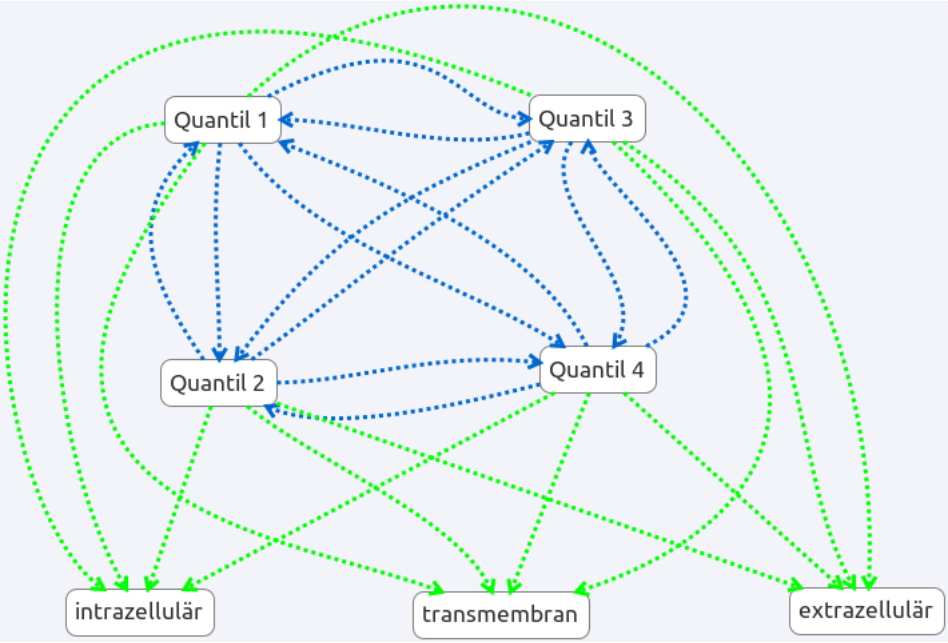

Abbildung 2.12: Diese Abbildung stellt den schematischen Aufbau des e^{3}TMR-Modells für 4 Quantile dar. Übergange der beobachtbaren Zustände sind als blaue Linien dargestellt, emittierte Zustände über grüne Linien. Im Vergleich zu e^{2}TMR (siehe Abbildung **??**) enthält dieses Modell einen weiteren emittierten Zustand.

variabler Iterationslänge. Dabei kann ein bereits trainiertes HMM gespeichert und bei Bedarf wiederverwendet werden. Die Decodierung des Modells erfolgt mit dem Viterbi-Algorithmus, wobei die Startwahrscheinlichkeiten a_{0k} entweder alle gleichgroß gesetzt werden, oder aber derjenige Status mit der höchsten Startwahrscheinlichkeit verwendet wird.

2.5 HMM_RA[?]

Das HMM_RA (Hidden-Markov-Modell_ReducedAlphabet) ist ein führendes Vorhersagetool für die Topologiebestimmung α-Helikaler Membranproteine. Es ist ähnlich dem HMMTOP-Modell aufgebaut, verwendet aber eine andere Eingabemodellierung. Grundlage für die Vorhersage von Topologien sind bei diesem Modell, im Gegensatz zum e^{3}TMR, Hydrophobizitäten. Wie HMMTOP besteht dieses Modell aus fünf Modulen. Sie werden bezeichnet als *inside-loops*, *inside helix tails*, *membrane helix*, *outside helix tail*, und *outside loop*. *Inside loop* beschreibt den Teil des Proteins, welcher sich innerhalb der Zelle befindet. An diesen schließt sich der inside *helix tail* an. Er bezeichnet den Beginn einer α-Helix innerhalb der Zelle, an der Grenzfläche zwischen Cytoplasma und Zellmembran. Dieser Teil geht in die *membrane helix*, also den transmembranen Helixteil des Membranproteins, über. Auf diesen folgt der *outside helix tail*, das Ende der

transmembranen Helix, welches sich an der Grenzfläche zwischen Phospholipiddoppelschicht und Extrazellulärraum befindet. Letztlich folgt der *outside-loop* für den extrazellulären Bereich des Proteins (siehe Abbildung **??**). Außerdem sind für die einzelnen Elemente bestimmte Richtungen und Größen vorgegeben. So sind die *tail*-Modi immer 15 Aminosäuren lang, sowohl aus der Helix kommend als auch in Richtung Helix. Der *transmembrane helix* Teil ist mindestens 17 Aminosäuren lang, höchstens aber 25 Aminosäuren. Die Richtungsangaben sind ebenso diskret bestimmt. So folgt auf eine nach innen[6] gerichtete Helix immer ein *inside helix tail*. An diesen höchstens 15 Aminosäuren langen Teil kann sich nun ein *inside loop* anschließen. Dieser Status wird in der Literatur auch auch "self-looping-state" bezeichnet, da er keine feste Größe vorweist sondern völlig variabel in seiner Länge ist. Wurde dieser Status einmal erreicht kann er beliebig oft in sich selbst übergehen oder, im Falle des *inside loops*, in einen nach außen gerichteten *inside helix tail*, an welchen eine nach außen gerichtete *transmembrane helix* anknüpfen würde. Die nach außen gerichtete *transmembrane helix* kann dann nur in einen *outside helix tail* übergehen und dieses wiederum nur in einen *outside loop*[7] oder aber einen nach innen gerichteten *outside helix tail*.

[6] Eine Helix deren Verlauf vom N-, zum C-Terminus gesehen von Extrazellulär nach Intrazellulär verläuft.
[7] Ebenso wie der *inside loop* ist dieser ein "self-looping state".

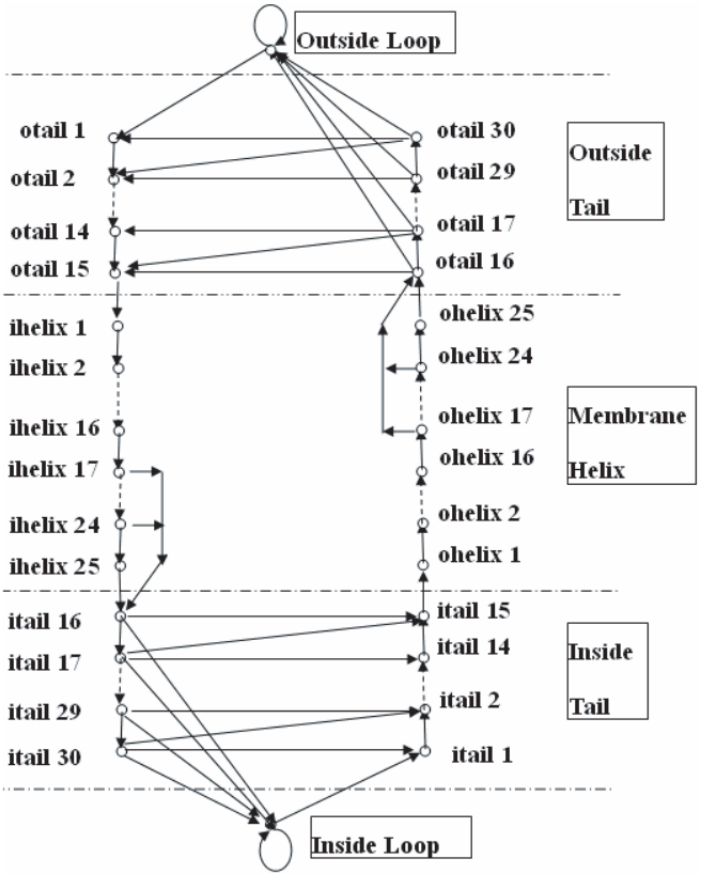

Abbildung 2.13: Diese Abbildung zeigt den Aufbau des HMM_RA-Modells. Es stellt die einzelnen Stati dar und deren mögliche Übergänge ineinander. Außerdem sind die festen Längen für beispielsweise Helices erkennbar(maximal 25, mindestens 17 Aminosäuren) .[?]

Verglichen mit dem e^{3}TMR-Modell besitzt das HMM_RA mehr beobachtbare Stati. Eine gleiche Tendenz ist trotzdem erkennbar, allerdings ist das HMM_RA feiner eingeteilt. Fasst man *outside-loop* und *outside helix tail* zusammen, so ergäbe dies den extrazellulären Status des e^{3}TMR. Durch Zusammenfassen des *inside helix tail* und *inside loop* erhielte man den intrazellulär Status des e^{3}TMR. Als übereinstimmend zu erachten sind *membrane helix* und transmembran. Jeder Status des HMM_RA ist, wie bei e^{3}TMR, assoziiert mit n-Emissionstati. Die unterschiedliche Zahl entsteht beim HMM_RA durch die verwendeten Alphabete, beim e^{3}TMR durch die unterschiedliche Quantilzahl.[?] Betrachtet man vergleichend den logischen Aufbau der beiden Modelle so lassen sich neben einigen grundsätzlichen Unterschieden auch Parallelen feststellen. Wie bereits erläutert, sind im HMM_RA-Modell Längen für bestimmte Elemente vorgeschrieben. Das bedeutet, innerhalb einer bestimmten Länge, hat das Modell keine Möglichkeit in einen anderen Status zu wechseln. Eine solche Regel enthält das e^{3}TMR-Modell nicht. Der

Vorteil in dieser Reglementierung ist, dass keine biologisch unrelevanten[8] Vorhersagen getroffen werden können. Andererseits ist im e^{3}TMR ebenso wie im HMM_RA implementiert, dass auf bestimmte Stati nur ein konkreter anderer Status folgen kann. So implementiert beispielsweise e^{3}TMR, dass auf einen intrazellulären beziehungsweise einen extrazellulären Bereich nur ein transmembraner Bereich folgen kann. Zählt man die *inside helix tails* und *outside helix tails* zum *inside loop* beziehungsweise *outside loop*, so könnte man beide Modelle sogar als identisch erachten, da der intrazellulär beziehungsweise extrazellulär Status prinzipiell auch als "self-looping state" angesehen werden kann (vgl. Abbildung **??** und Abbildung **??**).

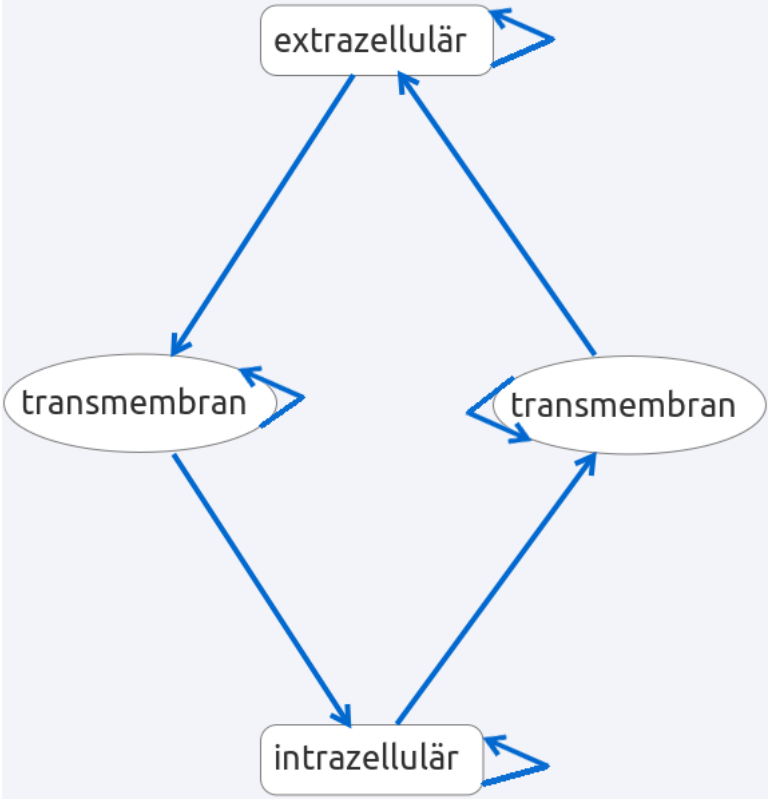

Abbildung 2.14: Diese Abbildung zeigt den Emissionsaufbau des 3^{3}TMR-Modells. Die blauen Linien verdeutlichen mögliche Statusübergänge.

Außerdem erlaubt keines der Modelle das Auftreten gleicher Topologien nach einem Membrandurchlauf. Dieser Sachverhalt impliziert, dass auch das 3^{3}TMR-Modell seinen Helices einen Richtungssinn verleiht. Es bedeutet, dass auf einen intrazellulären Bereich, nach einem transmembranen Bereich, immer ein extrazellulärer Bereich folgt und umgekehrt. Zu begründen ist dies mit der Tatsache, dass beide Modelle nur auf membrandurchspannende Proteine trainiert wurden. Dadurch kann für Membranproteine, die

[8] Unrelevant wäre zum Beispiel eine *transmembrane helix* aus lediglich zwei Aminosäuren, da eine solche Helix niemals die Membran durchqueren könnte.

innerhalb der Membran "umlenken" keine Vorhersage getroffen werden.[?]

2.6 TMHMM

Da das TMHMM[9] für die Erstellung der Datensätze für e^{3}TMR eine entscheidende Rolle spielte, soll der Aufbau und die Funktionsweise dieses Modells kurz erläutert werden. TMHMM wurde bereits 1998 von Erik Sonnhammer et al entwickelt. Es ist bis heute eines der genausten Tools zur Vorhersage von Topologien, begründet auf einem HMM. Es handelt sich hierbei um ein zyklisches HMM mit sieben unterscheidbaren Stati, zum Beispiel Helixkern (*helix core*), Helix Kappe (*helix cap*), zytoplasmatischen und extrazellulären loop. Die loops werden jeweils um einen Status für globuläre Domänen erweitert. Der Grundaufbau wird in Abbildung **??** noch einmal verdeutlicht.

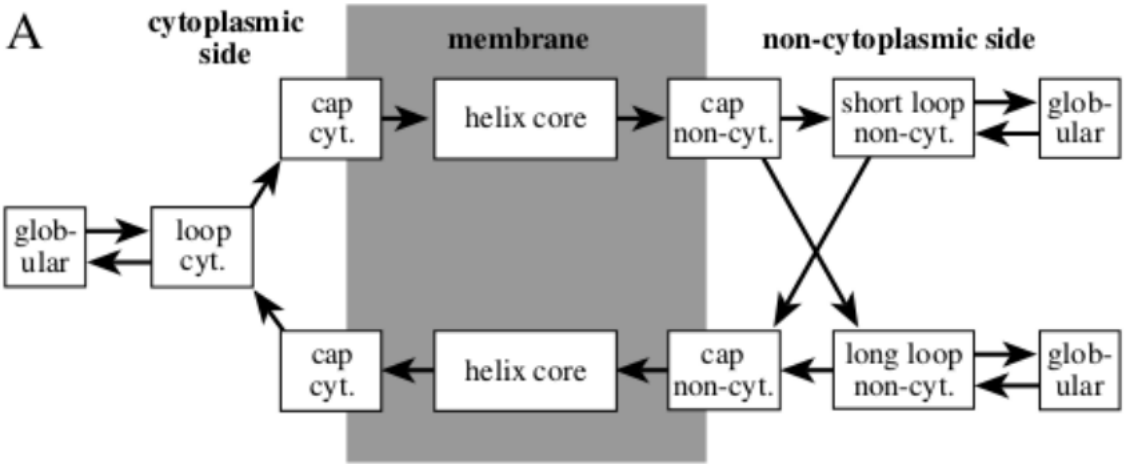

Abbildung 2.15: Diese Abbildung zeigt den modularen Aufbau des TMHMMs[?].

Aus der Abbildung wird besonders der zyklische Charakter des TMHMM's deutlich. Folgt man den Übergängen so bewegt man sich endlos durch das Modell. Ebenso fällt auf, dass nicht zu jeder Zeit jeder Übergang möglich ist. Es kann aus dem Cytoplasma nicht direkt in den extrazellulären Raum gesprungen werden. Außerdem ist ersichtlich, dass dieses Modell nur membrandurchspannende Proteine vorhersagen kann, da es keine Möglichkeit gibt, aus der Membran in in das vorausgehende Komparitment zu springen. In dieser Eigenschaft gleicht es den beiden zuvor vorgestellten Modellen.
Jedes Element des Modells besitzt seine eigenen Wahrscheinlichkeitsverteilungen, sodass nach dem Wechsel eines Zustandes die Vorhersagegenauigkeit weiter verbessert wird, da diese für jedes Element der Aminosäureverteilung spezifisch ist. Auch die Länge der einzelnen Stati ist wie beim HMM_RA festgesetzt. So wird eine TM-Helix nur

[9] Transmembrane Hidden Markov Modell

emittiert wenn die Kette mindestens fünf und höchstens 25 Aminosäuren lang ist. Deutlich erkennbar ist dieser Sachverhalt in der folgenden Abbildung.

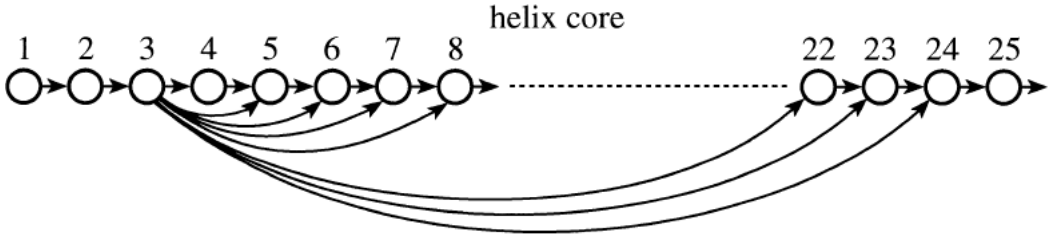

Abbildung 2.16: Diese Abbildung verdeutlicht den Aufbau des Helixkern Moduls (*helix core*)[?]

Zu Beginn jeder Helix befindet sich die Helixkappe (*helicx cap*) bestehend aus fünf Aminosäuren. Wie bei HMM_RA und e^{3}TMR wird zwischen nach innen gerichteter Helix und nach außen gerichteter Helix unterschieden. Dies hängt aber nicht von unterschiedlichen Übergangswahrscheinlichkeitsmatrizen ab, sondern nur vom vorhergehenden Zustand. Es wird demzufolge geprüft ob der Status vor dem Helixstart zytoplasmatisch oder extrazellulär war. Die Transitionswahrscheinlichkeiten für den Helixkern entstammen somit der selben Übergangswahrscheinlichkeitsmatrix. Die zwischen den Helices befindlichen loops wurden durch zwei Module modelliert. Diese enthalten zwei mal zehn Stati arrangiert in einer Leiterkonfiguration, verbunden durch einen "selbstloopenden"-Status (*self-looping-state*) (siehe Abbildung **??**[?]).

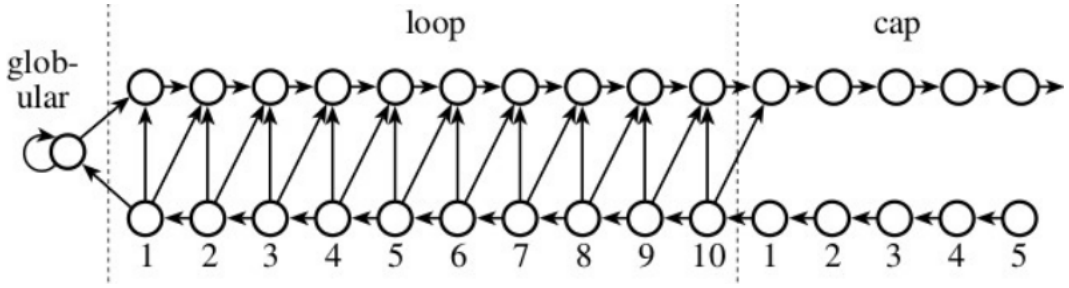

Abbildung 2.17: Diese Abbildung verdeutlicht die leiterförmige Konfiguration des loop-Moduls[?].

Der Aufbau für die intra und extrazellulären Modelle verhalten sich analog zueinander. Eine Unterscheidung ist lediglich auf Basis der Aminosäureverteilung möglich. Der Aufbau dieses Modells begründet sich mit der Tatsache, dass die Aminosäureverteilung kurz nach, beziehungsweise kurz vor dem Membraneintritt charakteristischer ist als in den langen globulären Domänen. Es findet sich hauptsächlich eine neutrale Aminosäureverteilung. Ebenso war auffällig, dass lange loops im extrazellulären Raum andere

Aminosäuren bevorzugt beinhalteten, als kurze loops in diesem Kompartiment. So findet sich eine vergleichbar höhere Anzahl positiver Reste in langen loops, als man sie in den kürzeren beobachten kann. Das Modell wurde mit dem Baum-Welch-Algorithmus in mehreren Iterationen geschätzt. Für das Training kamen zwei unterschiedliche Datensätze zum Einsatz. Ein Datensatz bestehend aus insgesamt 83 Membranproteinen wovon 38 mehrfach durchspannende Proteine und 45 einfach durchspannende Proteine die Basis bildeten. Der zweite Datensatz bestand aus insgesamt 160 Proteinen. Davon waren 108 mehrfach durchspannende Proteine und 52 einfach durchspannend. Beide Datensätze enthielten Sequenzen deren Topologie experimentell bestimmt und annotiert wurde. Das TMHMM bestitzt nach Angaben von Sonnhammer et al. eine Genauigkeit in der Topologievorhersage von 77,1% beim Training mit dem Datensatz 1 (83 Proteine) und 76,9% beim Training auf Datensatz 2 (160 Proteine).[?]

3 Durchführung

3.1 Erstellen der Datensätze

Eines der Hauptprobleme beim Erweitern des e^{2}TMR Modells ergab sich aus dem Erstellen eines neuen Trainingsdatensatzes. Der zunächst verwendete Datensatz basierte auf experimentell ermittelten und in der MP-topo-Datenbank hinterlegten Topologiedaten[?]. Dieser Datensatz bestand aus 144 Membranproteinen. Dabei handelte es sich allerdings um redundante Daten, wie in einer nachträglichen Untersuchung deutlich wurde. Nach dem Herausfiltern der mehrfach vorkommenden Sequenzen, verringerte sich die Datensatzgröße auf circa 60 Proteine. Die verbliebenen Proteine wurde erneut auf das Vorkommen von wenigstens zwei verschiedenen Topologien überprüft, woraufhin weitere 20 Proteine ausschieden. Der übrige Datensatz mit seiner geringen Größe reichte nicht aus um das Modell adäquat zu trainieren. Eine weitere Überlegung war das verwenden der PDBTM[10]. Diese Datenbank beinhaltet Transmembranproteine geclustert nach deren Klasse. Für das Modell kamen dabei nur α-helikale Proteine in Frage. Außerdem beinhaltet die PDBTM eine nicht-redundante Liste an α-helikalen Membranproteinen. Die Datenbank steht als xml-File zum Download zur Verfügung. Nach eingehender Betrachtung schied die Möglichkeit der Nutzung dieser Datenbank aus, da die hier annotierten Topologien nicht die gewünschten Informationen enthielten. So waren zwar die Sequenzen mit Topologieinformationen versehen, aber es gab keine Möglichkeit zwischen extra und intrazellulär zu unterscheiden. Der Grund hierfür war, dass die PDBTM prinzipiell die folgenden Topologien verwendete:

1. Side1
2. Side2
3. Beta-strand
4. alpha-helix
5. coil
6. membrane-inside
7. membrane-loop
8. interfacial helix

Dabei definieren *Side 1* und *Side 2* jeweils eine Seite der Membran (also intra beziehungsweise extrazellulär), *Beta-strand* ein β-Faltblatt, *alpha-helix* eine transmembrane

[10] *Protein Database for Transmembraneproteines*

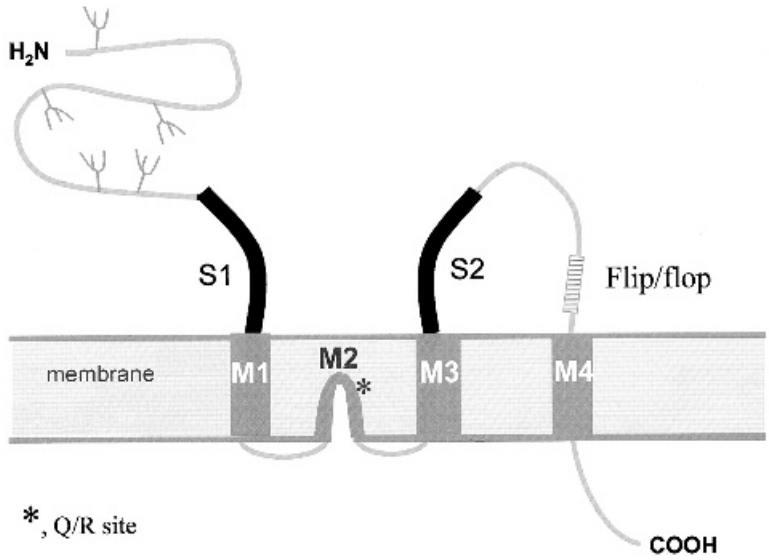

Abbildung 3.1: Diese Abbildung soll die membrane-loop Topologie verdeutlichen. Sie ist mit M2 in der Abbildung gekennzeichnet.[?]

α-Helix, *coil* definiert eine coil-Region außerhalb der Membran. Des weiteren bezeichnet *membrane inside* den membranständigen Teil eines β-Faltblattes, *mambrane loop* einen über wenige Residues in die Membran eintauchenden loop und *interfacial helix*. Diese Topologie wurde neu eingeführt. Sie beschreibt eine transmembrane Helix die aus mindestens sechs Residues besteht. Außerdem sind die Atome mit weniger als 9 Å aufgelöst und besitzen einem Durchgangswinkel durch die Membran von mindestens 30°[?]. Als problematisch erwiesen sich hierbei die beiden ersten Topologieannotationen "*Side 1*" und "*Side 2* " da sie nicht direkt ein bestimmtes Kompartiment referenzieren. Dadurch wurde eine Unterscheidung in intra-, extrazellulär unmöglich, weshalb die PDBTM für das weitere Vorgehen, zumindest als Topologiegrundlage, ausschied. Daraufhin folgte die Überlegung die gewünschten Topologieinformationen durch eine Software generieren zu lassen. Der Vorteil dieser Methode besteht darin, dass für nahezu jedes Protein mit bekannter Sequenz eine annähernd gute Topologie bestimmt werde könnte, und somit der Datensatz eine verwertbare Größe erhalten würde. Als Vorhersagetool diente letztlich der TMHMM-Server. Grundlage für den Datensatz bildete die Liste Nichtredundanter α-helikaler Proteine der PDBTM. Hierfür wurde die PDBTM als XML-Datei heruntergeladen und ausgelesen. Es wurde eine Software erstellt, die das XML-File parste und die PDB-Identifier sowie die Aminosäuresequenz im fasta-Format als Liste abspeicherte. Dabei wurden die Sequenzen nach Redundanz gefiltert. Außerdem wurden alle Proteine ausgeschlossen, welche in ihrer Topologieannotation "unbekannte"-Bereiche beinhalteten. Diese Liste enthielt 687 Proteine im fasta-Format und konnte direkt an den TMHMM-Server übergeben werden. Als Ausgabeformat wurde "extensive, no graphics" gewählt. Die Ausgabe lieferte eine Textdatei, welche alle

Proteine mit deren Topologieannotationen enthielt. Zum einfacheren Auslesen der Liste musste diese zunächst noch angepasst werden. Hierfür wurde ein Java-Applet erstellt, welches die unregelmäßig auftretenden Leerzeichen in den Annotationszeilen zwischen Topologieinformation und Sequenzlänge durch einfache Tabulatoren ersetzte. Analog zu den Topologieinformationen waren ebenso die Energieprofile der Membranproteine nötig. Zur Berechnung dieser Profile stand bereits eine funktionierende Software zur Verfügung, welche nur noch gering modifiziert werden musste um die gewünschte Aufgabe zu erfüllen. Die Modifikation bezog in diesem Falle lediglich auf die Implementierung des Einlesens der nötigen PDB-Dateien, welche von der Software zur Berechnung der Energieprofile benötigt wurden. Die soeben erwähnten PDB-Dateien sind ebenfalls mit Hilfe einer hierfür erstellten Software, anhand der Proteinliste der PDBTM, welche auch schon beim TMHMM-Server zum Einsatz kam, heruntergeladen worden. Da die Berechnung der Energieprofile viel Zeit in Anspruch nahm, wurde diese durch mehrere Instanzen der Energieprofilsoftware realisiert. Hierfür ist die PDBTM-Liste in fünf gleich große Teillisten zerlegt worden. Jede Energieprofilberechnungsinstanz erhielt als Quelle für ihre Proteine eine der Listen. Anhand der Listeneinträge konnte somit jede Berechnungsinstanz auf eine bestimmt PDB-Datei zugreifen. Die Berechnung lieferte nun 687 Energieprofile.

1BOJ_A.ep2

```
NAME    1BOJ
TYPE    TM
HEAD    Chain   ResNo   Res     SS      Energy
ENGY    A       35      W       H       3.5096594369051615
ENGY    A       36      Q       H       6.467727851656664
ENGY    A       37      F       H       4.790636518372085
ENGY    A       38      S       H       5.33369657392808
ENGY    A       39      M       H       -0.9734031535205343
ENGY    A       40      L       H       0.21526658098726215
ENGY    A       41      A       H       -3.902238021483862
ENGY    A       42      A       H       -3.526702730560577
ENGY    A       43      Y       H       -1.4270539762090226
ENGY    A       44      M       H       -3.7741160603632387
ENGY    A       45      F       H       -1.5455273988767124
ENGY    A       46      L       H       -1.7186989469951577
ENGY    A       47      L       H       -0.14705074779768443
ENGY    A       48      I       H       0.6651841331093992
ENGY    A       49      M       H       -2.762810796840659
ENGY    A       50      L       H       -0.23653982458094547
ENGY    A       51      G       H       3.348247536658717
ENGY    A       52      F       H       0.5257199549855454
ENGY    A       53      P       H       5.013404167857013
ENGY    A       54      I       H       2.1380323668458217
ENGY    A       55      N       H       11.672017835773575
ENGY    A       56      F       H       0.8022349571622753
```

Abbildung 3.2: Diese Abbildung zeigt den Ausschnitt aus einem Energieprofil. Erkennbar sind in den ersten Zeilen die Informationen zum Protein, wie beispielsweise die PDB-ID, darunter, dass es sich um ein Membranprotein handelt. Danach folgt eine Kopfzeile mit den Definitionen der folgenden Spalten. Deutlich erkennbar ist der Sequenzstart erst bei Residue 35.

Nun erfolgte das eigentliche Erstellen der Trainingsdaten. Diese wurden aus den Energieprofilen und den Topologieinformationen generiert. Hierfür ist erneut eine Software entwickelt worden. Es wurde dabei für jedes Energieprofil ein Proteinobjekt angelegt, welches eine Liste aller Energiewerte des Proteins, den Dateinamen, die PDB-ID[11], die Chain-ID[12], den Sequenzbeginn aus dem dem ep-File, dem Sequenzende aus dem ep-File sowie aus der TMHMM-Ausgabe und einer Liste mit Längenangaben für die jeweiligen Topologien bestand. Aus den zuletzt Genannten wurde direkt die Topologie, als Liste an Strings, iterativ generiert. Mit diesen Informationen konnten letztlich die Trainingsdateien, bezeichnet als ep3-Dateien, erstellt werden. Diese sind ähnlich dem ep-File aufgebaut (siehe Abbildung **??**), mit dem Unterschied, dass eine weitere Spalte mit der Topologieinformation für die jeweilige Residue eingeführt wurde und der Energiewert durch das Quantil ersetzt wurde (siehe Abbildung **??**). Hohe Energiewerte erhielten dabei niedrige Quantilwerte, niedrige Energien hohe Quantilwerte.

1BOJ_A.ep3

```
NAME    1BOJ
TYPE    TM
HEAD    Chain   ResNo   Res     Energy  Topology
ENGY    A       35      W       3       outside
ENGY    A       36      Q       1       outside
ENGY    A       37      F       2       outside
ENGY    A       38      S       2       outside
ENGY    A       39      M       9       transmembrane
ENGY    A       40      L       7       transmembrane
ENGY    A       41      A       11      transmembrane
ENGY    A       42      A       11      transmembrane
ENGY    A       43      Y       10      transmembrane
ENGY    A       44      M       11      transmembrane
ENGY    A       45      F       10      transmembrane
ENGY    A       46      L       10      transmembrane
ENGY    A       47      L       8       transmembrane
ENGY    A       48      I       6       transmembrane
ENGY    A       49      M       11      transmembrane
ENGY    A       50      L       8       transmembrane
ENGY    A       51      G       3       transmembrane
ENGY    A       52      F       6       transmembrane
ENGY    A       53      P       2       transmembrane
ENGY    A       54      I       4       transmembrane
ENGY    A       55      N       0       transmembrane
ENGY    A       56      F       6       transmembrane
```

Abbildung 3.3: Diese Abbildung zeigt den Ausschnitt eines ep3-Files, analog zu Abbildung **??**. Erkennbar sind die Veränderungen der Struktur hinsichtlich der Topologieinformationen und Energiewerte.

Ein zusätzliches Problem bestand darin, dass das Modell möglichst mit unterschiedlichen Quantilen trainiert werden sollte. Daher musste das Erzeugen der Quantile möglichst generisch erfolgen. Hierfür wurde eine Methode implementiert, die anhand einer

[11] PDB-ID oder PDB-Identifier ist eine genaue nicht-redundante Bezeichnung die von der PDB für jedes dort abgelegte Protein vergibt. Der Identifier besteht aus vier Zeichen bestehend aus Buchstaben und Zahlen

[12] ChainID bezeichnet die Kette des jeweiligen Proteins. Hierfür werden ebenfalls Buchstaben vergeben.

Nutzereingabe die Anzahl der Quantile anpasst. Somit war es möglich innerhalb kurzer Zeit viele unterschiedliche Datensätze zu erstellen. Für alle Datensätze wurde für die spätere Validierung ein Datensatz angelegt, welcher die Quantile und Topologieinformationen, zur einfacheren Verarbeitung, jeweils in einer separaten Zeile in einer Datei beinhaltet. Um diesen Datensatz zu erstellen wurde ebenfalls eine kleine Softwareroutine entwickelt. Des weiteren wurde zur Analyse der Datensätze ein Applet erstellt, welches die Übergangswahrscheinlichkeiten der Quantile berechnet und die Ausgabe in einer Datei abspeichert. Die Berechnung erfolgte analog zur Formel **??**. Außerdem wurden die log-odds für "*transmembrane*" gegen das "*nicht-transmembrane*" Modell und für das "*inside*" und das "*outside*" Modell bestimmt. Die log-odds ergaben sich allgemein nach folgender Formel:

$$log - odd = \frac{p_{ij}^{Modell1}}{p_{ij}^{Modell2}} \tag{3.1}$$

Für die Berechnung der log-odds für"*transmembran*"/"*nicht-transmembran*" wurden die "*inside* " und "*outside*" Übergangswahrscheinlichkeiten zusammengefasst. Es wurde dabei "*transmembran*" als $Modell1$ und "*nicht-transmembran*" als $Modell2$ gesetzt. Bei *inside*" / "*outside*" wurde *inside*" als $Modell1$ und *outside*" als $Modell2$ gesetzt.

3.2 Erweiterung des e^{2}TMR-Modells

Das bestehende e^{2}TMR-Modell erlaubte zunächst nur die Vorhersage zweier Topologien. Um dieses Modell zu erweitern musste demnach ein weiterer Status in die Software integriert werden. Hierfür war die Anpassung mehrerer Klassen nötig. Hauptsächlich handelte es sich dabei um die Klassen "IOHmmTrainer", welche vorwiegend Methoden zum Aufbau und Training des HMMs beinhaltet, und "IOHmmCalculator". Diese Klasse enthält Methoden zur Berechnung und Ausgabe des wahrscheinlichsten Pfades durch das Modell. Hierin mussten die Modellingalgorithmen umgeschrieben werden. Anfangs basierte die Ausgabe auf einem Binärcode. Das Modell wurde in "*inside*" und "*outside*" geteilt und jedem dieser Stati sind vier unterscheidbare Zustände (entsprechend der vier Quantile) zugeteilt worden. Diese Zuteilung erfolgte dabei numerisch. Der "*inside*" Zustand erhielt dabei die Werte eins bis vier, der "*outside*" Zustand die Werte fünf bis acht. Dieser Algorithmus musste nun um einen Status ergänzt werden. Dazu wurde ein "transmembran" Status eingeführt. Dieser ist zwischen den beiden bestehenden Stati angesiedelt worden, sodass die Verteilung wie folgt verändert wurde:

- inside: 1,2,3,4
- transmembrane: 5,6,7,8

- outside: 9,10,11,12

An dieser Stelle zeigte sich ein Problem der bestehenden Software, bedingt durch geringe Flexibilität in Bezug auf das Eingabealphabet. Es war für dieses Modell noch nicht möglich, ohne größere Eingriffe in den Quellcode, einen Datensatz mit veränderter Quantilzahl zu verarbeiten. Daher wurde diese festgesetzte Definition durch einen Algorithmus ersetzt, der in der Lage ist die Zustände generisch zu verwalten. Hierfür ist die folgende Annahme getroffen worden:

- Die Anzahl der Zustände des Systems ergibt sich aus der Quantilzahl multipliziert mit der Anzahl an vorherzusagenden Zuständen

Im Falle von e^{3}TMR sind dies drei Zustände multipliziert mit der Quantilzahl. Für vier Quantile ergeben sich somit 12 Zustände insgesamt. Mit diesem Wissen konnten jeweils die Variablen iterativ aufgefüllt werden. Diese Modellierung musste sowohl zum Aufbau des Modells verwendet werden als auch beim Auslesen des wahrscheinlichsten Pfades. Außerdem kam es zu Problemen beim Einlesen der Testdaten, verursacht durch das Auftreten von zweistelligen Zahlenwerten für die Quantile. Da das Ausgangsmodell auf die Verwendung von vier Quantilen ausgelegt war, erfolgte die Eingabe der Testsequenz als einfacher String aus aneinandergereihten Zahlen. Dabei wurde jede Position des String iterativ in einen Integerwert übersetzt. Dies führte bei zweistelligen Eingabewerten zu Fehlern, da diese durch das Iterieren über den String nicht als solche erkannt werden konnten. Daher wurde die Regel eingeführt, dass jeder Zahlenwert durch ein Semikolon getrennt werden muss. Somit konnte der String an diesem definierten Zeichen aufgetrennt, in ein Array überführt und letztlich iterativ auf seine Integerinhalte geparst werden. Des weiteren ergaben sich Probleme beim einlesen der Trainingsdaten. Hierbei wurde zur Vermeidung von Fehlern das Quantil zunächst durch ein Regex[13] geparst und danach in einen Integer umgewandelt. Als Fehlerquelle stellte sich der reguläre Ausdruck heraus. Dieser war auf eine einstellige Zahl ausgelegt und erkannte somit die zweistelligen Zahlen bei Quantilzahlen über 10 nicht. Behoben wurde das Problem durch das Entfernen des regulären Ausdrucks und das direkte Übersetzen des Quantilwertes in einen Integerwert. Um die Fehlersicherheit trotzdem aufrecht zu erhalten wird der Wert nur übernommen, sollte er innerhalb der definierten Quantilgröße liegen. Andernfalls gibt das Programm einen Fehler aus und weißt den Benutzer darauf hin. In sehr frühen Teststadien des Programms zeigt die grafische Ausgabe oft sehr stark wechselnde Emissionen an. So traten beispielsweise unmögliche Zustandskombi-

[13] regulärer Ausdruck (*regular expression)*

nationen auf. Diese manifestierten sich als Wechsel von "*inside*[14]" zu "*outside*[15]" ohne dabei dazwischen einen "*transmembrane*" Bereich zu zeigen. Oder es wechselte von "*inside*" beziehungsweise "*outside*" über "*transmembrane*" in den selben Bereich. Darum ist eine Regel eingeführt worden, welche die Ausgabe des Modells überprüft. Hierfür wurde implementiert, dass jeder Zustand vor einer Membran in einer Variable abgelegt wird, ebenso jeder nicht-transmembrane Status. Wird nun nach einem Transmembransegment der gleiche Status emittiert wie vor der Membran, so wird dieser Status in den erforderlichen Status geändert. Gleiches geschieht, sollte der Status innerhalb eines "*inside*" oder "*outside*" -Kompartimentes plötzlich in ein konträres Kompartiment springen (ausgenommen natürlich "*transmembrane*").
Anfangs existierten zwei eigenständige Softwarepakete für das e^{3}TMR Modell, eines mit grafischer Benutzeroberfläche und eines ohne. Beide Modelle hatten einen unterschiedlichen Stand und wurden im Laufe der Arbeit angeglichen. Ein großer Vorteil des nicht-grafischen Programms lag darin, dass das trainierte Modell gespeichert werden konnte und somit nicht bei jeder Benutzung neu trainiert werden musste. Besonders bei hohen Quantilzahlen erwies sich ein permanentes Training als sehr zeitraubend. Daher ist die Möglichkeit der Speicherung des HMMs auch in die nicht-grafische Variante implementiert worden. Anfangs war diese Speichermöglichkeit noch unzuverlässig im Programm verankert, sodass bei jedem Start nicht geprüft wurde ob ein trainiertes Modell vorhanden ist. Das Programm versuchte ein Modell zu Laden und warf einen Fehler sollte es keine entsprechende Datei finden. Erst daraufhin begann das Training. Diese Programmierung wurde umgangen, indem das Programm am Anfang prüft, ob ein verwendbares HMM zur Verfügung steht.

[14] intrazellulär
[15] extrazellulär

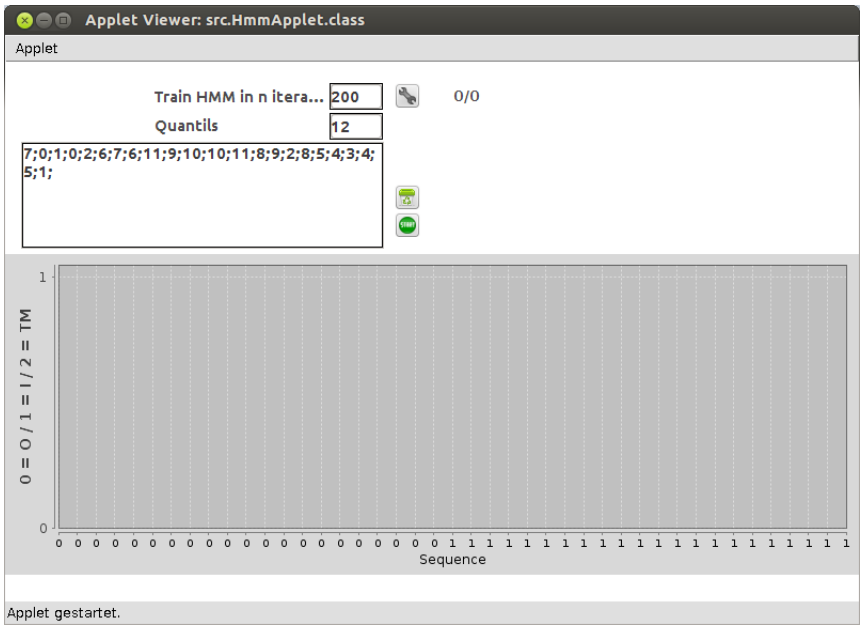

Abbildung 3.4: Diese Abbildung zeigt die grafische Benutzeroberfläche des e^{3}TMR Modells. Im unterem Bereich erscheint die Ausgabe des wahrscheinlichsten Pfades. Im oberen Bereich werden die gewünschte Trainingsiterationszahl, Quantile sowie die Testsequenz angegeben.

Dabei wird geprüft ob die angegebenen Trainingsiterationen und Quantile den Nutzereingaben entsprechen. Sollte dies nicht der Fall sein so beginnt das Modell mit dem Training.
Das grafische Paket wurde außerdem um die Möglichkeit ergänzt, die Anzahl der Quantile im Datensatz anzugeben. Die grafische Ausgabe des wahrscheinlichsten Pfades ist gegenüber dem alten Modell verbessert worden. Der Aufbau der GUI[16] wird in Abbildung **??** verdeutlicht.

3.3 Validierung

Um zu bestimmen wie gut die Vorhersagen des fertigen Modells sind, ist dieses in mehreren Schritten validiert worden. Da zur Validierung eine gewisse Automatisierung nötig war, wurde dazu das nicht-grafische Softwarepaket verwendet. Als Anspruch für die Validierung galt:

- Der Datensatz zum Training muss größer sein als der Testdatensatz
- Der Testdatensatz sollte aus möglichst unterschiedlichen Proteinen bestehen

[16] GUI grafische Benutzeroberfläche *engl. graphical user interface*

- Die Auswahl der Test- sowie Trainingsdaten sollte zufällig geschehen
- Die Verwendeten Datensätze sollten zur Nachvollziehbarkeit erhalten bleiben
- Die trainierten Modelle sollten zur späteren erneuten Verwendung erhalten bleiben

Nach diesen Erfordernissen ist das nicht-grafische e^{3}TMR Modell modifiziert worden. Zunächst wurde implementiert, dass für jeden Validierungslauf zehn verschiedene Test- und Trainingssets erstellt werden. Ein Validierungslauf setzt sich demnach aus zehn Testläufen zusammen. Hierbei wird automatisch ein Ordner angelegt dessen Name spezifisch für den Lauf und die verwendeten Daten ist. Diese Bezeichnung setzt sich zusammen aus:

1. Dem aktuellem Datum
2. Der aktuellen Zeit
3. Dem verwendeten Datensatz (in Hinblick auf die Quantile)

Ein Beispiel dafür sähe wie folgt aus:

"17-07-2012at09-45-48_Qunatiles_12"

Innerhalb dieses Ordners wird für jeden Testlauf ein spezifischer Unterordner angelegt, benannt nach dem Lauf, also von Lauf eins bis Lauf zehn. Dabei sind die Ordner mit *"run_1" bis "run_10 "* beziffert. Jeder dieser Ordner besitzt ebenso eine feste Struktur. So enthalten sie einen Ordner für die Testdaten, bezeichnet mit *"testData"* , einen Ordner für die Trainingsdaten, bezeichnet mit *"trainData"* und einen Ordner für das gelernte HMM, bezeichnet mit *"HMM"*.
Am Anfang jedes Laufes werden die gesamten Trainingsdaten für jeden Testlauf in den jeweiligen *"trainData"* Order kopiert. Aus diesem Ordner werden zufällig zehn Proteine ausgewählt, und in den *"testData"* Ordner verschoben. Danach wird das Modell für jeden Testlauf neu trainiert und das jeweilige HMM im *"HMM"* Ordner abgelegt. Da unterschiedliche Trainingsiterationen und Quantile verwendet werden, erhält das HMM diese Informationen ebenfalls im Dateinamen. Dieser setzt sich aus der Iterationszahl, einem Unterstrich, gefolgt vom verwendeten Datensatz hinsichtlich der Quantile zusammen (zum Beispiel *"100_Quantiles_12"* für 100 Iterationen bei 12 Quantilen).
Zur Validierung wurde zunächst für alle Quantilsequenzen des Testdatensatzes mittels der gelernten Modelle, die Topologiesequenz vorhergesagt. Mit diesen Vorhersagesequenzen wurden zwei unterschiedliche Werte berechnet. Zunächst der Q2-Wert zur Bestimmung der Vorhersagegenauigkeit transmembraner und nicht-transmembraner Be-

reiche. Hierfür wurden die intra- und extrazellulären Bereiche zu einer Topologie zusammengefasst.Dies erfolgte durch das umbenennen aller emittierten "*i*" für " *inside*" und "*o*" für "*outside*" in "*n*" für "*nicht-transmembran*". Daraus resultierte eine Zeichenkette bestehend aus "*t*" für transmembrane Bereiche und "*n*" für nicht-transmembrane Bereiche. Der Q2-Wert selbst spiegelt die Wahrscheinlichkeit wider, dass ein bestimmtes Zeichen Z zum Zeitpunkt t der emittierten Sequenz mit dem Zeichen X zum Zeitpunkt t der realen Sequenz übereinstimmt. Berechnet wird dies nach folgender Formel:

$$Q_x = \frac{N_{Z_t}}{N_{X_t}} \; wenn \; gilt : Z = X \tag{3.2}$$

Dabei steht N_{Z_t} für alle richtig emittierten Zeichen zum Zeitpunkt t und N_{X_t} für alle möglichen Zeichen die richtig vorhergesagt werden konnten. Gleiches gilt für den ebenso berechneten Q3-Wert. Hierbei werden lediglich drei Stati, anstatt von nur zweien betrachtet.
Beide Werte wurden jeweils zusätzlich für zwei verschiedene Modellsysteme bestimmt. Zum einen für das angepasste Modell, zum anderen für die direkte Ausgabe des Modells. Alle Validierungsergebnisse sind zur weiteren Verarbeitung in einem File abgespeichert worden. Das File erhielt dabei einen durch Tabulatoren getrennten Aufbau zur besseren Auswertung mit Statistiksoftware. Außerdem erhielt jeder Lauf zur Identifikation eine Überschrift bestehend aus dem aktuellen Datum. Gefolgt wurde diese durch die Definition des jeweiligen Datensatzes und der Verwendeten Iterationszahl. Danach erfolgte zeilenweise die Ausgabe der einzelnen Läufe mit ihren jeweiligen Ergebnissen für den Q2 und Q3-Wert, aufgeteilt in das angepasste und das direkte Modell. Einen Auszug aus der Validierungsdatei zeigt Abbildung **??**.

result_neu.txt

```
ThuatJulat19at19-24-17atCESTat2012
Q3 and Q2 for 4 Quantiles @ 100 Iterations
QValues original        adapted
Q2 RUN_1        69.77867203219316       69.77867203219316
Q3 RUN_1        42.25352112676056       45.99597585513079
Q2 RUN_2        73.10734463276836       73.10734463276836
Q3 RUN_2        54.80225988700565       46.03389830508475
Q2 RUN_3        74.74207107374856       74.74207107374856
Q3 RUN_3        55.52158960641957       47.00038211692778
Q2 RUN_4        72.78366502972344       72.78366502972344
Q3 RUN_4        61.798914448177825      45.56733005944689
Q2 RUN_5        68.79457917261055       68.79457917261055
Q3 RUN_5        32.16833095577746       39.51497860199715
Q2 RUN_6        75.25533890436398       75.25533890436398
Q3 RUN_6        49.721448467966574      50.41782729805014
Q2 RUN_7        73.14178638351031       73.14178638351031
Q3 RUN_7        42.97314178638351       44.47220487195503
Q2 RUN_8        68.54274414359598       68.54274414359598
Q3 RUN_8        40.00608457560085       43.86979008214177
Q2 RUN_9        71.56208277703605       71.56208277703605
Q3 RUN_9        41.57543391188251       42.80373831775701
Q2 RUN_10       72.10854092526691       72.10854092526691
Q3 RUN_10       55.071174377224196      49.86654804270463
Q3 and Q2 for 4 Quantiles @ 200 Iterations
QValues original        adapted
Q2 RUN_1        70.22132796780684       70.22132796780684
Q3 RUN_1        42.93762575452716       50.42253521126761
Q2 RUN_2        73.71751412429379       73.71751412429379
Q3 RUN_2        55.163841807909606      46.89265536723164
Q2 RUN_3        74.93312953763852       74.93312953763852
Q3 RUN_3        55.750859763087504      42.911730989682844
Q2 RUN_4        72.55104678211424       72.55104678211424
Q3 RUN_4        61.69552856035151       48.56552080640993
Q2 RUN_5        67.68901569186876       67.68901569186876
Q3 RUN_5        32.02567760342368       40.69186875891583
Q2 RUN_6        75.99814298978644       75.99814298978644
```

Abbildung 3.5: Diese Abbildung zeigt einen Ausschnitt aus der Validierungsdatei. Deutlich zu erkennen sind die Bezeichnungen für den jeweiligen Lauf sowie den genutzten Datensatz und die Ergebnisse.

4 Auswertung

4.1 Datensätze

Die Annahme, dass Energieprofile spezifisch für bestimmte Topologien sind, findet man bereits in der Betrachtung der Energieprofile bestätigt. So deuten Bereiche mit besonders niedrigen Energien auf transmembrane Bereiche hin. Abbildung **??** zeigt deutlich die eben beschriebenen Eigenschaften. Transmembrane Bereiche sind in der Abbildung rot unterlegt. Erkennbar sind die niedrigen Energien in diesem Bereich. Dieses Beispiel ist exemplarisch und steht für alle α-helikalen Membranproteine. Ein genauerer Nachweis dieses Sachverhaltes zeigt sich später in der Auswertung der Datensätze.

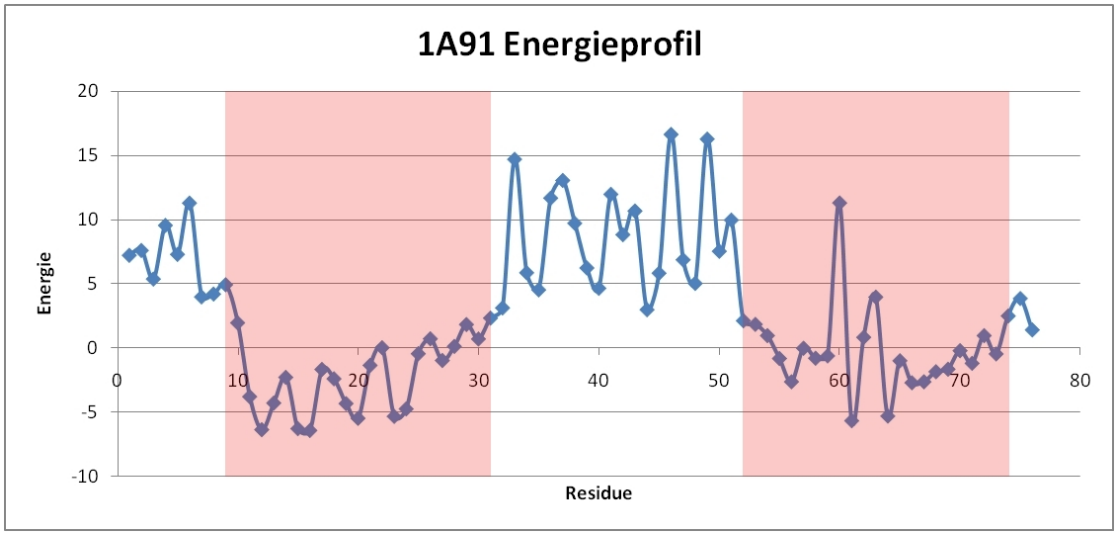

Abbildung 4.1: Diese Abbildung zeigt ein Membranproteinenergieprofil (PDB ID 1A91). Transmembrane Bereiche sind rot unterlegt und enthalten deutlich niedrigere Energien als der Rest.

Zusätzlich sollen an dieser Stelle kurz die Ergebnisse der verschiedenen Energieprofile für unterschiedliche Quantilzahlen erläutert werden. Das Verwenden von unterschiedlich großen Quantilräumen ging auf die Vermutung zurück, dass ein Energieprofil umso deutlicher mit diskreten Werten abgebildet wird, je feiner der Quantilraum ist dem das neue Profil zu Grunde liegt. Betrachtet man hierzu die Abbildungen **??**, **??** und **??** so lässt sich feststellen, dass diese Annahme nicht vollständig zutrifft. So sind zum Beispiel im 12 Quantile Profil im Bereich zwischen den Residues 10 und 20 kleine Schwankungen im Profil erkennbar, welche sich bei 4 oder 20 Quantilen nicht einstellen. Gleiches

lässt sich im Bereich von 66 bis 70 Residues erkennen. Trotzdem fällt eine Glättung des Graphen in Abbildung **??** im Bereich von 7 bis 10 Quantilen auf, welche in Abbildung **??** nicht zu beobachten ist. Dieser Sachverhalt widerspricht der aufgestellten These. Vergleicht man aber das Profil für 4 Quantile mit dem für 12 und 20, findet sich die Annahme bestätigt. Zurückzuführen ist dies auf die Tatsache , dass bei weniger Quantilen mehr Werte dem gleiche Zustandsraum (also Quantil) zugeordnet werden. Somit werden Schwankungen im Energieprofil geglättet und damit aber auch Informationen aus dem Profil entfernt. Es resultiert letztlich ein sehr rauscharmes Energieprofil, in dem sich deutlich große Sprünge aus dem Ausgangsprofil abzeichnen, kleine Energieunterschiede jedoch werden eliminiert. Das gleiche gilt im Prinzip auch beim Vergleich der beiden Energieprofile für 12 und 20 Quantile, wobei allerdings die Glättung im Bereich von 7 bis 10 und 66 bis 70 Residues im 20 Quantilprofil nicht erklärbar ist.

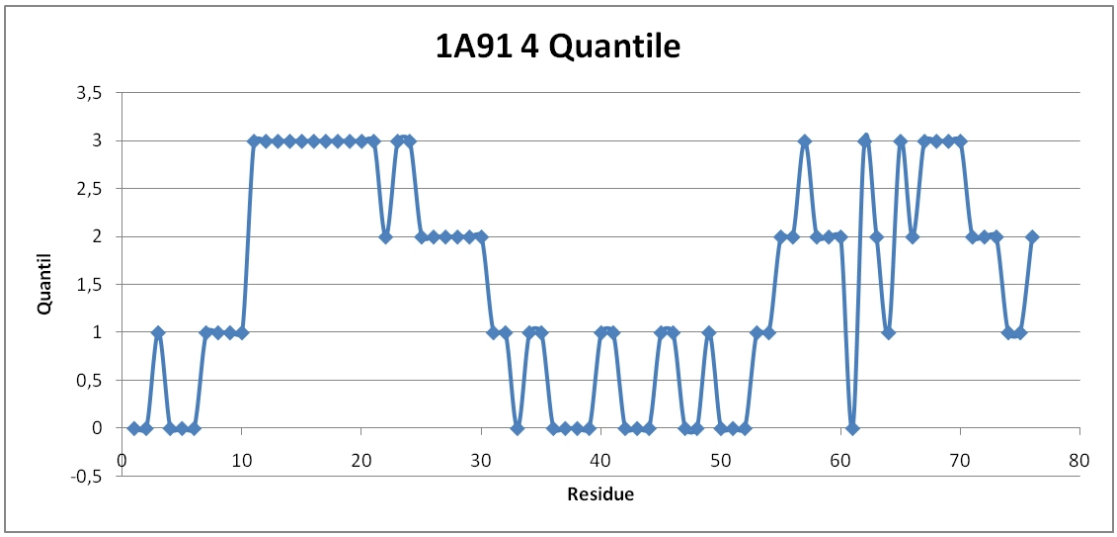

Abbildung 4.2: Energieprofil für 1A91 mit 4 Quantilen.

Bei der Betrachtung der Datensatzanalysen ergaben sich unterschiedliche Ergebnisse. Das Ziel war hierbei hauptsächlich, die unterschiedlichen Übergangswahrscheinlichkeiten zu analysieren um Rückschlüsse auf die internen Abläufe des Modells ziehen zu können. Die statistische Auswertung wurde mit zunehmender Quantilzahl sehr unübersichtlich. Da die Tendenzen für alle Datensätze ähnlich sind, wird an dieser Stelle exemplarisch für alle Datensätze, die Auswertung am Datensatz mit 12 Quantilen ausführlich erörtert. Dieser Datensatz wurde ausgewählt, da er für die Validierung die besten Ergebnisse lieferte. Trotzdem soll an späterer Stelle kurz der Datensatz mit 4 Quantilen vergleichend erwähnt werden. Transmembrane Bereiche nehmen im Energieprofil die niedrigsten Energiewerte an, denen später die höchsten Quantile zugeordnet werden. Somit deuten Übergänge von hohen Quantilen in hohe zumeist auf trans-

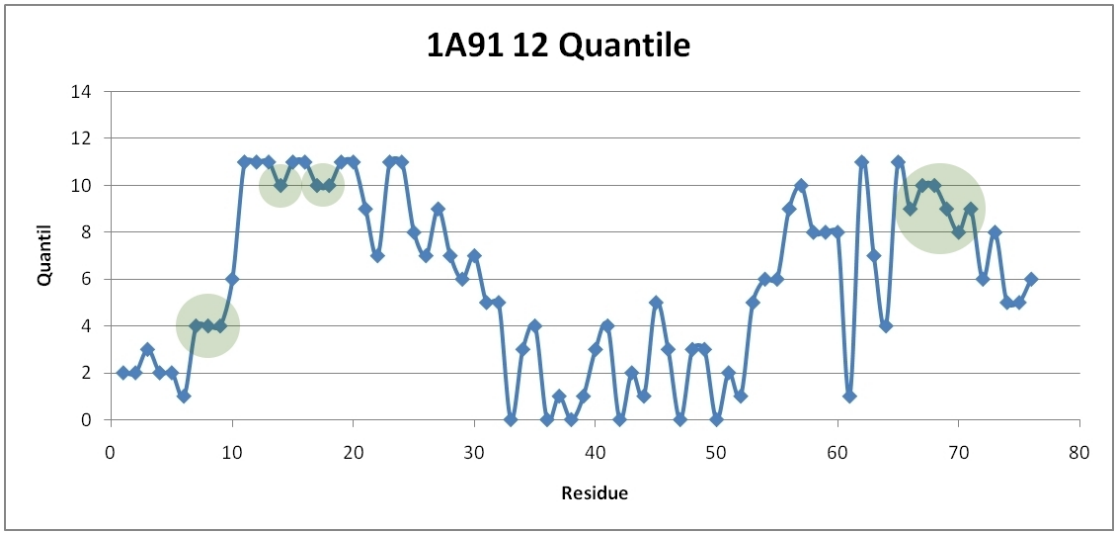

Abbildung 4.3: Energieprofil für 1A91 mit 12 Quantilen. Die gravierende Unterschiede zum 20 Quantile Profil sind grün eingefärbt.

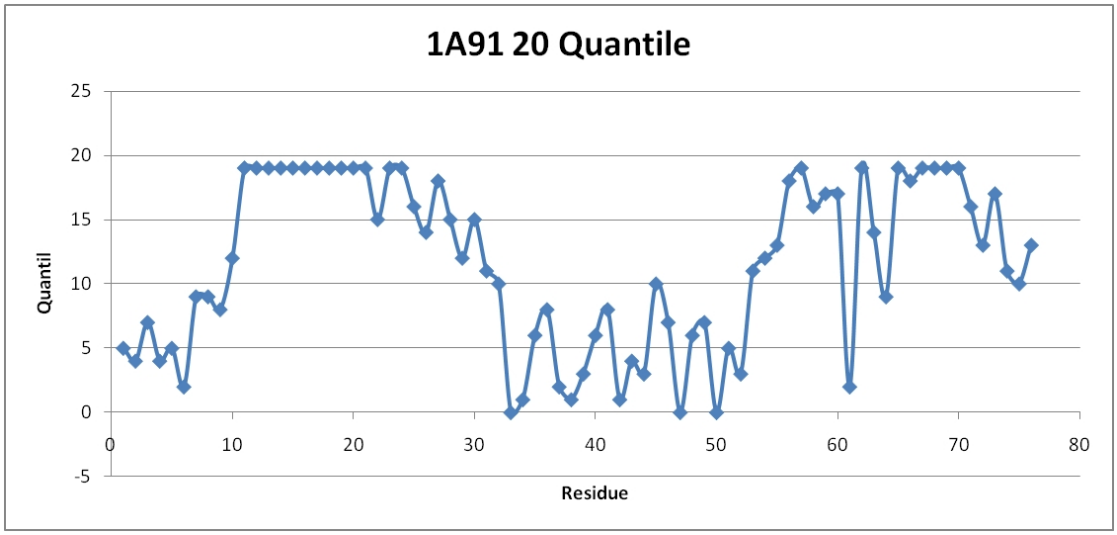

Abbildung 4.4: Energieprofil für 1A91 mit 20 Quantilen.

membrane Bereiche hin. Bei niedrigen Quantilen gilt dieser Sachverhalt in umgekehrter Weise. Diese deutliche Trennung sollte die Grundlage für die genauen Vorhersagen dieser Topologien sein. Deutlich wird dies in Abbildung **??**. Sie veranschaulicht die Übergangswahrscheinlichkeiten des Datensatzes, jeweils für das "*inside*- " das "*outside*- " und das "*transmembran*- "Modell. Es ist dabei die vermutete Grundtendenz erkennbar. So zeigt sich eine deutlich höhere Wahrscheinlichkeit für hohe Quantilübergänge in transmembranen Bereichen als in den niederen Übergängen. Sehr deutlich erkennt man diese Ausprägung bereits ab dem siebten Quantil. Ausgehend von diesem Quantil sind alle Übergänge in Quantile größer als sechs in den Transmembranbereich gewertet. Für die "*inside*- " und "*outside* "-Übergänge ist die Entscheidung schwerer zu treffen, da sich die Wahrscheinlichkeiten zwischen diesen beiden kaum unterscheiden. Es lässt sich eher feststellen, dass sich beide Kurven stark annähern, was auf eine ähnliche Energielandschaft schließen lässt. Dies ist logisch und erklärt sich durch die sehr ähnlichen Lösemitteleigenschaften in den nicht-transmembranen Bereichen, woraus eine ähnliche Energiestruktur resultiert. Des weiteren ist erkennbar, dass die "*inside* "-Übergangswahrscheinlichkeiten zumeist stärker ausgeprägt sind als die "*outside* "-Übergangswahrscheinlichkeiten.

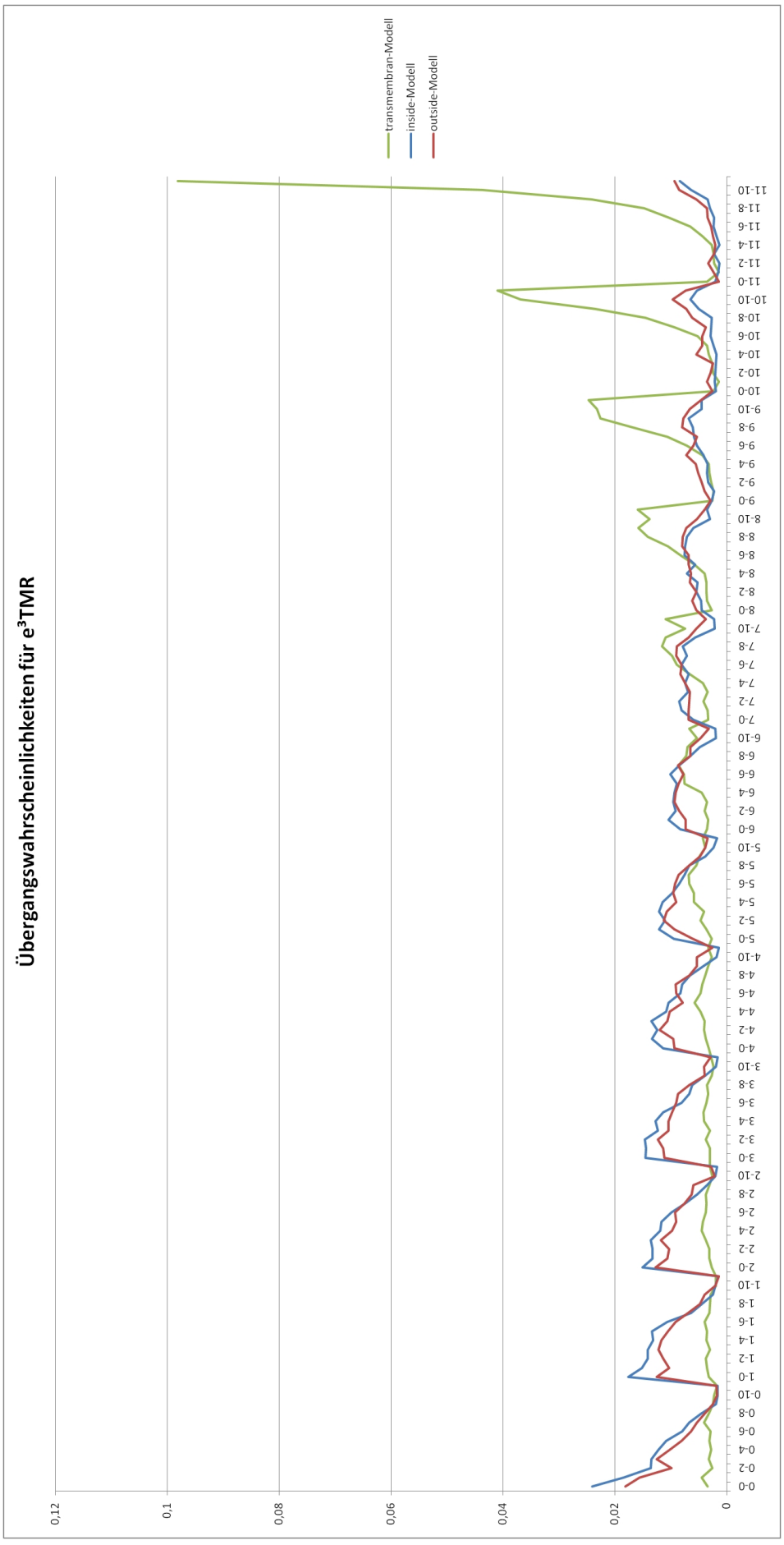

Abbildung 4.5: Diese Abbildung zeigt die Übergangswahrscheinlichkeiten für e^3TMR. Auf der Abszisse sind die Quantilübergänge, auf der Ordinate die Wahrscheinlichkeiten aufgetragen. Die Übergangswahrscheinlichktsentwicklung für das transmembrane Modell zeigt der grüne, für inside der blaue und für outside der rote Graph.

Zur besseren Veranschaulichung und um einen tieferen und genaueren Einblick in das Modell zu erhalten, wurden außerdem die log-odds des Datensatzes betrachtet (siehe Punkt **??**). Die Betrachtung der log-odd-Matrizen für "*transmembran* " und "*nicht-transmembran*" bestätigt die bisherigen Annahmen. Es zeigt sich deutlich die erhöhte Tendenz des Modells auf "*transmembrane*" zu entscheiden, sollten die Quantile hohe Werte aufweisen. Betrachtet man Tabelle **??** stellt man fest, dass sich die Matrix diagonal in zwei Hälften teilen lässt. Negative Werte sind rot eingefärbt und deuten auf eine Entscheidung des Modells in den "*nicht-transmembranen*" Bereich hin, grüne Werte sind größer null und zeigen eine Entscheidung in den "*transmembranen*" Bereich an. Mathematisch begründet sich dieser Umstand mit dem verwendeten Logarithmus. Sollte der Wert im Logarithmus größer null sein, so wird der Logarithmus positiv. Dies ist der Fall, wenn der Dividend größer ist als der Divisor, also wenn die Wahrscheinlichkeit für $Modell1$ höher ist als für $Modell2$ (siehe Formel **??**).

Tabelle 4.1: Diese Tabelle zeigt die log-odd-Matrix für das transmembran/nicht-transmembran Modell. Rote Werte sind kleiner null und deuten auf nicht-transmembran hin. Grüne Werte sind größer null und deuten auf transmembran hin.

	0	1	2	3	4	5	6	7	8	9	10	11
0	-1,83	-1,33	-1,52	-1,39	-1,40	-1,12	-0,89	-0,39	-0,27	0,12	0,33	0,00
1	-1,55	-1,27	-1,23	-1,46	-1,23	-1,21	-0,91	-0,74	-0,40	-0,06	-0,04	0,34
2	-1,65	-1,36	-1,33	-1,21	-0,87	-0,88	-0,93	-0,71	-0,43	-0,43	0,26	0,29
3	-1,45	-1,45	-1,27	-1,32	-1,06	-0,94	-0,85	-0,84	-0,59	-0,34	-0,21	0,24
4	-1,18	-1,10	-1,11	-1,12	-0,80	-0,47	-0,61	-0,67	-0,53	-0,35	-0,26	0,54
5	-1,07	-1,07	-0,86	-1,04	-0,55	-0,50	-0,29	-0,16	-0,19	0,12	0,22	0,51
6	-0,78	-0,97	-0,78	-0,97	-0,72	-0,15	-0,16	0,01	0,12	0,24	0,48	0,96
7	-0,65	-0,77	-0,58	-0,69	-0,54	-0,13	0,10	0,20	0,32	0,57	0,68	1,32
8	-0,59	-0,43	-0,40	-0,46	-0,53	-0,07	0,15	0,31	0,64	0,88	1,20	1,47
9	-0,06	-0,25	-0,32	-0,34	-0,31	-0,25	0,24	0,65	0,88	1,14	1,44	1,71
10	0,05	-0,67	0,01	0,18	-0,10	0,07	0,39	1,06	1,19	1,35	1,52	1,88
11	0,73	-0,25	0,02	0,07	0,51	0,73	0,95	1,32	1,53	1,71	1,78	2,41

Tabelle 4.2: Diese Tabelle veranschaulicht die Ausprägung der log-odds. Je dunkler eine Zelle gefärbt ist umso stärker ist ihre Ausprägung. Rot deutet auf nicht-transmembran, grün auf transmembran hin. Es ist eine deutliche Verstärkung des Signal zu den Enden hin sichtbar.

	0	1	2	3	4	5	6	7	8	9	10	11
0	-1,83	-1,33	-1,52	-1,39	-1,40	-1,12	-0,89	-0,39	-0,27	0,12	0,33	0,00
1	-1,55	-1,27	-1,23	-1,46	-1,23	-1,21	-0,91	-0,74	-0,40	-0,06	-0,04	0,34
2	-1,65	-1,36	-1,33	-1,21	-0,87	-0,88	-0,93	-0,71	-0,43	-0,43	0,26	0,29
3	-1,45	-1,45	-1,27	-1,32	-1,06	-0,94	-0,85	-0,84	-0,59	-0,34	-0,21	0,24
4	-1,18	-1,10	-1,11	-1,12	-0,80	-0,47	-0,61	-0,67	-0,53	-0,35	-0,26	0,54
5	-1,07	-1,07	-0,86	-1,04	-0,55	-0,50	-0,29	-0,16	-0,19	0,12	0,22	0,51
6	-0,78	-0,97	-0,78	-0,97	-0,72	-0,15	-0,16	0,01	0,12	0,24	0,48	0,96
7	-0,65	-0,77	-0,58	-0,69	-0,54	-0,13	0,10	0,20	0,32	0,57	0,68	1,32
8	-0,59	-0,43	-0,40	-0,46	-0,53	-0,07	0,15	0,31	0,64	0,88	1,20	1,47
9	-0,06	-0,25	-0,32	-0,34	-0,31	-0,25	0,24	0,65	0,88	1,14	1,44	1,71
10	0,05	-0,67	0,01	0,18	-0,10	0,07	0,39	1,06	1,19	1,35	1,52	1,88
11	0,73	-0,25	0,02	0,07	0,51	0,73	0,95	1,32	1,53	1,71	1,78	2,41

Tabelle 4.3: Diese Tabelle zeigt die log-odd-Matrix für das inside/outside-Modell. Rote Werte sind kleiner null und deuten auf outside grüne auf inside hin. Grau hinterlegt sind die eindeutig als transmembran wertenden Zellen.

	0	1	2	3	4	5	6	7	8	9	10	11
0	0,28	0,17	0,32	0,07	0,18	0,29	0,22	0,23	0,15	-0,24	-0,05	-0,11
1	0,34	0,38	0,21	0,14	0,12	0,24	0,15	-0,07	-0,07	-0,49	-0,02	0,07
2	0,17	0,22	0,25	0,14	0,20	0,26	0,06	-0,02	-0,18	-0,55	-0,02	-0,49
3	0,26	0,24	0,17	0,17	0,20	0,15	-0,11	-0,27	-0,09	-0,01	-0,76	-0,57
4	0,20	0,33	0,03	0,24	0,06	0,28	-0,09	-0,15	-0,02	-0,24	-1,08	-0,59
5	0,43	0,25	-0,02	0,12	0,23	0,02	-0,09	-0,14	-0,01	-0,22	-0,50	-0,67
6	0,13	0,35	0,07	0,01	0,02	0,04	0,25	-0,02	-0,01	-0,30	-0,88	-0,44
7	-0,14	0,18	0,24	0,05	0,00	-0,20	0,01	-0,24	-0,14	-0,20	-0,94	-0,54
8	-0,20	-0,30	0,01	-0,24	0,11	-0,18	0,11	-0,08	-0,11	-0,20	-0,58	-0,12
9	-0,06	-0,58	-0,30	-0,37	-0,50	-0,56	-0,12	0,08	-0,27	-0,14	-0,41	-0,02
10	-0,25	-0,51	-0,34	-0,24	-1,14	-0,62	-0,42	-0,28	-0,86	-0,38	-0,40	-0,33
11	0,40	-0,49	-0,94	0,01	-0,48	-0,30	-0,16	-0,43	-0,16	-0,46	-0,30	-0,12

Um die Ausprägungen der log-odds zu visualisieren wurde die selbe Matrix (Tabelle **??**) mit einer von rot nach grün reichenden Farbskala erstellt. Je dunkler die Farbe ist, umso stärker ist dieser Übergang ausgeprägt. Rot deutet dabei wieder auf "*nicht-transmembran*" und grün auf "*transmembran*" hin (siehe Tabelle **??**). Es wird dabei deutlich, dass sich das rote beziehungsweise grüne Farbsignal zum Start beziehungsweise

Endpunkt der Matrix verstärkt. So befindet sich dunkelste rote Feld an Position $[0,0]$ das dunkelste grüne Feld an Stelle $[11,11]$.
Die Betrachtung der log-odd-Matrizen für "*inside* " und "*outside*" lassen im Vergleich zum Diagramm Rückschlüsse auf die Entscheidungen des Modells zu. Zur besseren Erkennung wurden die Teile der Matrix grau unterlegt, welche eindeutig dem "*transmembran*"-Bereich zuzuordnen waren (vergleiche Tabelle **??**). Es fällt auf, dass hauptsächlich die höheren Quantile auf einen "*outside*"-Bereich hindeuten. Ganz niedrige Quantile zeigen einen "*inside*"-Bereich an. Betrachtet man allerdings die gewichtete Matrix, stellt man fest, dass die log-odds viel weniger differenziert sind und sich kaum starke Entwicklungen heraus kristallisieren. Besonders deutlich wird dieser Sachverhalt, vergleicht man die Tabellen **??** und **??**.

Tabelle 4.4: Diese Matrix zeigt die gewichteten log-odds des inside/outside Modells an. Je dunkler eine Farbe desto stärker ist der Übergang ausgeprägt. Rot steht für outside, grün für inside.

	0	1	2	3	4	5	6	7	8	9	10	11
0	0,28	0,17	0,32	0,07	0,18	0,29	0,22	0,23	0,15	-0,24	-0,05	-0,11
1	0,34	0,38	0,21	0,14	0,12	0,24	0,15	-0,07	-0,07	-0,49	-0,02	0,07
2	0,17	0,22	0,25	0,14	0,20	0,26	0,06	-0,02	-0,18	-0,55	-0,02	-0,49
3	0,26	0,24	0,17	0,17	0,20	0,15	-0,11	-0,27	-0,09	-0,01	-0,76	-0,57
4	0,20	0,33	0,03	0,24	0,06	0,28	-0,09	-0,15	-0,02	-0,24	-1,08	-0,59
5	0,43	0,25	-0,02	0,12	0,23	0,02	-0,09	-0,14	-0,01	-0,22	-0,50	-0,67
6	0,13	0,35	0,07	0,01	0,02	0,04	0,25	-0,02	-0,01	-0,30	-0,88	-0,44
7	-0,14	0,18	0,24	0,05	0,00	-0,20	0,01	-0,24	-0,14	-0,20	-0,94	-0,54
8	-0,20	-0,30	0,01	-0,24	0,11	-0,18	0,11	-0,08	-0,11	-0,20	-0,58	-0,12
9	-0,06	-0,58	-0,30	-0,37	-0,50	-0,56	-0,12	0,08	-0,27	-0,14	-0,41	-0,02
10	-0,25	-0,51	-0,34	-0,24	-1,14	-0,62	-0,42	-0,28	-0,86	-0,38	-0,40	-0,33
11	0,40	-0,49	-0,94	0,01	-0,48	-0,30	-0,16	-0,43	-0,16	-0,46	-0,30	-0,12

Wie bereits erwähnt soll an dieser Stelle kurz auf den Datensatz mit 4 Quantilen eingegangen werden. Dabei wird verdeutlicht, dass das Ausgangsmodell[17] nicht genügte um den Erfordernissen der Vorhersage von drei verschiedenen Stati gerecht zu werden. Zunächst ist eine klare Tendenz für transmembrane Bereiche erkennbar. Diese befinden sich hauptsächlich in den hohen Quantilen, wohingegen niedrige Quantile auf "*inside*" beziehungsweise "*outside*" hindeuten (siehe Abbildung **??**). Ebenso finden sich parallelen bei der Betrachtung der log-odds. Die Matrix für die "*transmembran*"/"*nicht-transmembran*"-Modelle lässt sich wieder mittig teilen (siehe Tabelle **??**).

[17] e^{2}TMR und das frühe e^{3}TMR Modell waren zunächst auf vier Quantile ausgelegt.

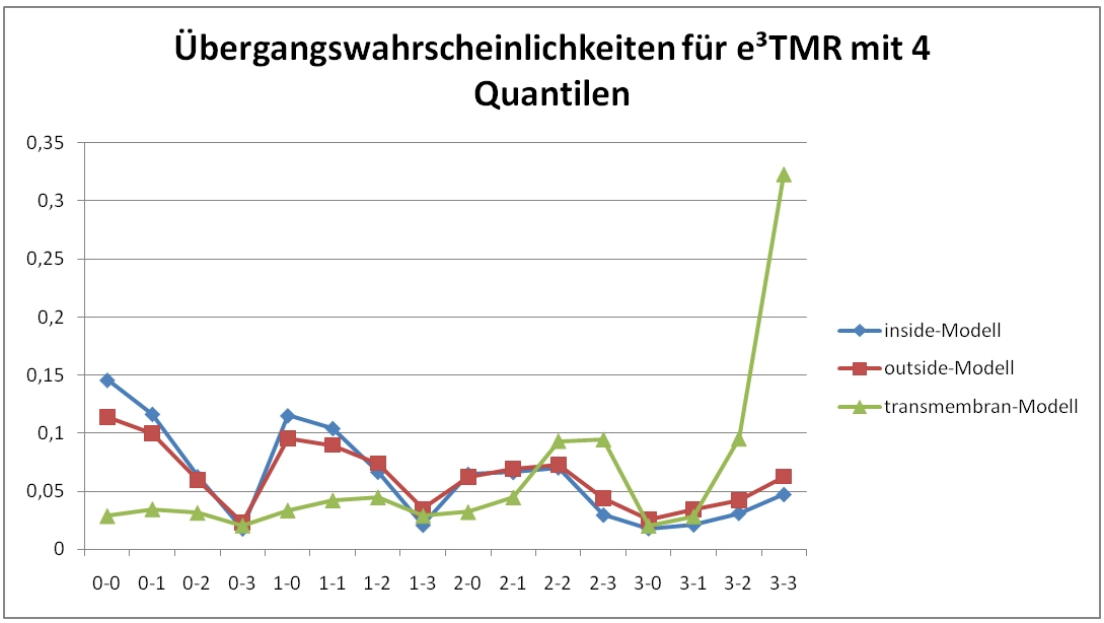

Abbildung 4.6: Diese Abbildung zeigt die Übergangswahrscheinlichkeiten für e^3TMR bei 4 Quantilen. Auf der Abszisse sind die Quantilübergänge, auf der Ordinate die Wahrscheinlichkeiten aufgetragen. Die Übergangswahrscheinlichktsentwicklung für das transmembrane Modell zeigt der grüne, für inside der blaue und für outside der rote Graph.

Betrachtet man die Wichtung der Matrixwerte in Tabelle **??** ist ebenso eine Konzentration der Farbintensität, bezogen auf rot und grün, in Richtung der Eckpunkte der Matrix zu beobachten. Daraus lässt sich schließen, dass die Vorhersage von Topologien bezogen auf die zwei Stati "*transmembran*" und "*nicht-transmembran*" zuverlässig durchführbar sein müsste, da diese Intensivierung eine Verstärkung der Übergangswahrscheinlichkeiten anzeigt. Bestätigt wird diese Vermutung durch die gute Funktionalität des e^2TMR Modells. Betrachtet man hingegen die log-odds für "*inside*" und "*outside*" , so lassen sich Unzulänglichkeiten gegenüber dem Datensatz mit 12 Quantilen feststellen. Es ist zwar eine Unterscheidung zwischen "*inside*" und "*outside*" möglich, allerdings ist diese nur sehr schwach ausgeprägt. Daran zeigt sich der Vorteil eines Datensatzes mit einer höheren Anzahl an Quantilen. Diese Gegebenheit ermöglicht eine genauere Abbildung des Energieprofils und somit werden auch mehr Informationen in das HMM eingegeben.

Tabelle 4.5: Diese Tabelle zeigt die log-odd-Matrix für das transmembran/nicht-transmembran Modell bei 4 Quantilen. Rote Werte sind kleiner null und deuten auf nicht-transmembran hin. Grüne Werte sind größer null und deuten auf transmembran hin.

	0	1	2	3
0	-1,50	-1,13	-0,66	0,03
1	-1,13	-0,83	-0,44	0,05
2	-0,67	-0,41	0,27	0,95
3	-0,06	0,03	0,96	1,77

Tabelle 4.6: Diese Tabelle veranschaulicht die Ausprägung der log-odds für 4 Quantile für das transmembran/nicht-transmembran-Modell. Je dunkler eine Zelle gefärbt ist umso stärker ist ihre Ausprägung. Rot deutet auf nicht-transmembran, grün auf transmembran hin. Es ist eine deutliche Verstärkung des Signal zu den Enden hin sichtbar.

	0	1	2	3
0	-1,50	-1,13	-0,66	0,03
1	-1,13	-0,83	-0,44	0,05
2	-0,67	-0,41	0,27	0,95
3	-0,06	0,03	0,96	1,77

Tabelle 4.7: Diese Tabelle veranschaulicht die log-odds für 4 Quantile für das inside/outside-Modell. Rot deutet auf outside mit Werten kleiner null, grün auf inside mit Werten größer null hin. Eindeutig als transmembran gewertete Zellen sind grau unterlegt.

	0	1	2	3
0	0,25	0,16	0,05	-0,24
1	0,19	0,15	-0,10	-0,50
2	0,04	-0,04	-0,03	-0,39
3	-0,33	-0,47	-0,29	-0,27

Tabelle 4.8: Diese Tabelle veranschaulicht die Ausprägung der log-odds für 4 Quantile für das inside/outside-Modell. Je dunkler eine Zelle gefärbt ist umso stärker ist ihre Ausprägung. Rot deutet auf outside, grün auf inside hin. Eindeutig als transmembran gewertete Zellen sind grau unterlegt.

	0	1	2	3
0	0,25	0,16	0,05	-0,24
1	0,19	0,15	-0,10	-0,50
2	0,04	-0,04	-0,03	-0,39
3	-0,33	-0,47	-0,29	-0,27

4.2 Validierung des Modells

Die Validierung der Vorhersagegenauigkeit des Modells lieferte eine Liste mit vielen Werten, welche sich in ihrer Ausprägung relativ ähnlich sind. So liegen die Q2 Vorhersagegenauigkeiten im Allgemeinen zwischen 60 und 80 Prozent. Die Q3 Werte bewegen sich in einem Bereich zwischen 30 und 60 Prozent. Es war auffällig, dass verschiedene Test- beziehungsweise Trainingsdatensätze unterschiedliche Qualitäten hervorbrachten. Ebenso war die Anzahl der Quantile im Datensatz sowie die verwendeten Trainingsiterationen ein ausschlaggebender Faktor für die Vorhersagegenauigkeit des Modells. So ließ sich beobachten, dass Datensätze mit geringeren Quantilzahlen bereits nach weniger Trainingsiterationen vertretbare Werte lieferten, wohingegen Datensätze mit einer höheren Quantilzahl mehr Iterationen benötigten um vergleichbare Werte zu generieren. Anfangs stand die Vermutung, dass das Modell besser wird, je mehr Quantile im Datensatz verwendet werden, da somit das Energieprofil genauer abgebildet werden kann. Diese These lies sich nach der Validierung widerlegen. Betrachtet man die Entwicklung der Vorhersagegenauigkeit über verschiedene Läufe bei 100 Iterationen und variabler Quantilzahl, so stellt man fest, dass die Genauigkeit von den eben genannten Parametern abhängig ist, nicht aber vom jeweiligen Lauf. Abbildung **??** verdeutlicht die Entwicklung der Q-Werte für 100 Iterationen bei unterschiedlichen Test- beziehungsweise Trainingsdaten. Bei *run_1* lieferte der 20-Quantil-Datensatz die besten Ergebnisse mit 73,2% für den Q2-Wert und 47,8% für den Q3-Wert. Der Datensatz mit 10 Quantilen erreicht in *run_10* den besten Q2 Wert mit 72,5% . Der beste Q3 Wert geht auf den Datensatz mit 4 Quantilen mit einem Wert von 49,9% zurück. In *run_2* liegt der beste Q2 Wert bei 77% und stammt aus dem 20-Quantile-Datensatz, der höchste Q3-Wert in diesem Lauf geht auf den Datensatz mit 10 Quantilen und einem Betrag von 47,1% zurück.

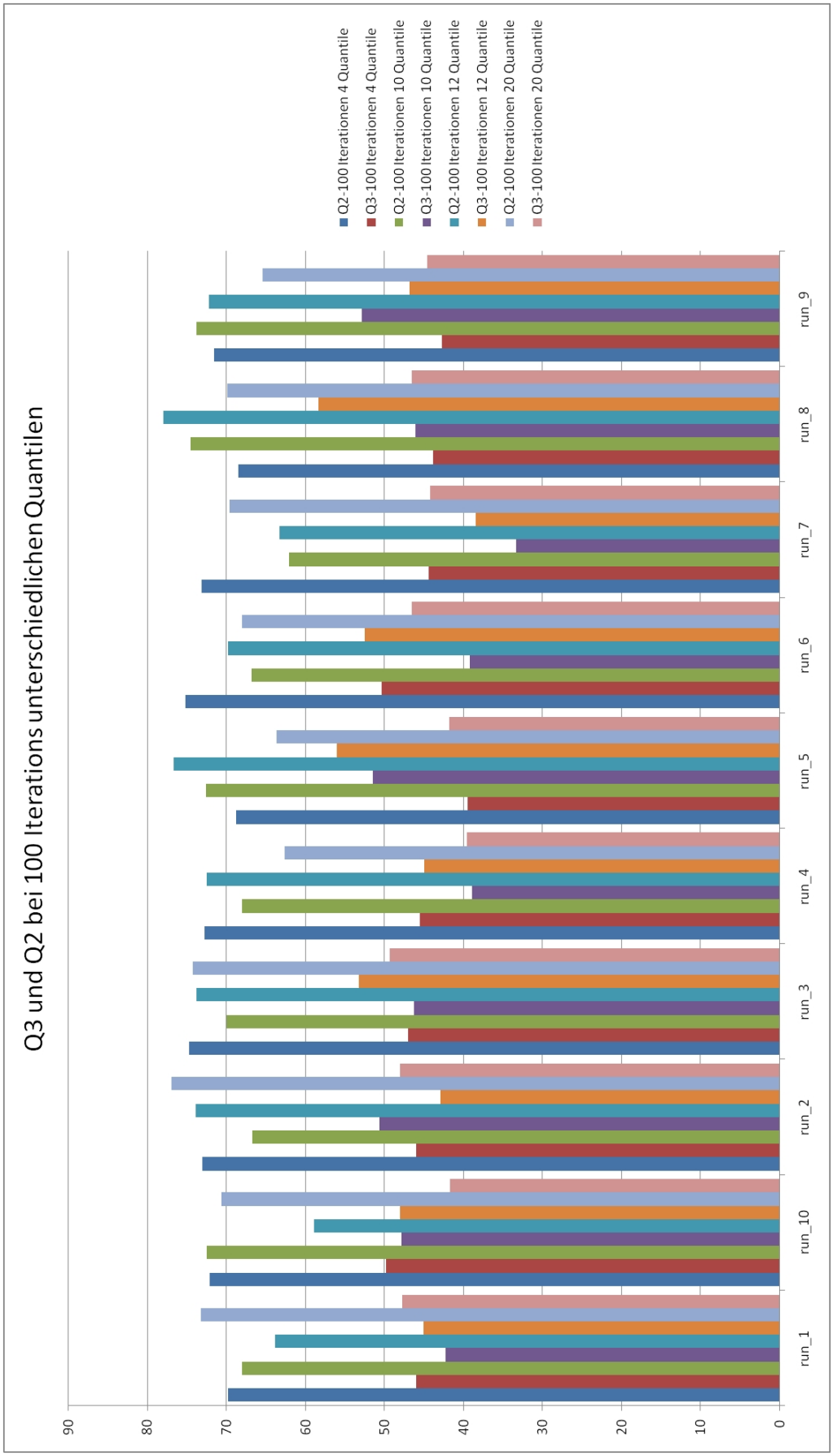

Abbildung 4.7: Diese Abbildung zeigt die Entwicklung der Q-Werte für 100 Iterationen bei unterschiedlichen Test- beziehungsweise Trainingssets für verschiedene Datensätze.

Für *run_3* ergeben sich die Werte von 74,7% für den Q2-Wert im 4-Quantil-Datensatz und 53,3% für den Q3-Wert im 12-Quantil-Datensatz. In *run_4* liegt der Datensatz mit 4 Quantilen bei einem Q2-Wert von 72,8% an und einem Q3-Wert von 45,6% an der Spitze. Bei *run_5* stellte sich der höchste Q2-Wert mit 76,7% und der höchste Q3-Wert bei 56% ein. Beide Werte entstammten dem 12-Quantil-Datensatz. Den höchsten Q2-Wert im *run_6* lieferte der 4-Quantil-Datensatz mit einem Wert von 75,3%. Den maximalen Q3-Wert mit 52,5% generierte der 12-Quantil-Datensatz. Bei *run_7* zeigte sich ein maximaler Q2-Wert von 73,1% und ein Q3-Wert von 44,5%, wobei beide Werte aus dem Datensatz mit 4 Quantilen stammen. Bei der Betrachtung von *run_8* lässt sich feststellen, dass der höchste Q2- und Q3-Wert dem 12-Quantil-Datensatz entstammt. Dabei liegt der Q2-Wert bei 78% und der Q3-Wert bei 58,4%. Der Q2- und Q3-Wert in *run_9* liegen bei 73,8% und 52,9%. Beide Wert gehen auf den 10-Quantil-Datensatz zurück. Diese Sachverhalte zeigen auch, dass der Datensatz mit der höchsten Genauigkeit für "*transmembran*" / "*nicht-transmembran*" nicht zwingend den höchsten Q3-Wert liefern muss. Daher muss für die Auswahl des entsprechenden Datensatzes letztlich ein Kompromiss zwischen Q2- und Q3-Wert eingegangen werden. Betrachtet man die maximalen Q2-Werte der Datensätze bei den verschiedenen Iterationen, so kristallisiert sich der Datensatz mit 12 Quantilen bei 100 Iterationen mit einem Wert von 78% heraus (siehe Abbildung **??**).

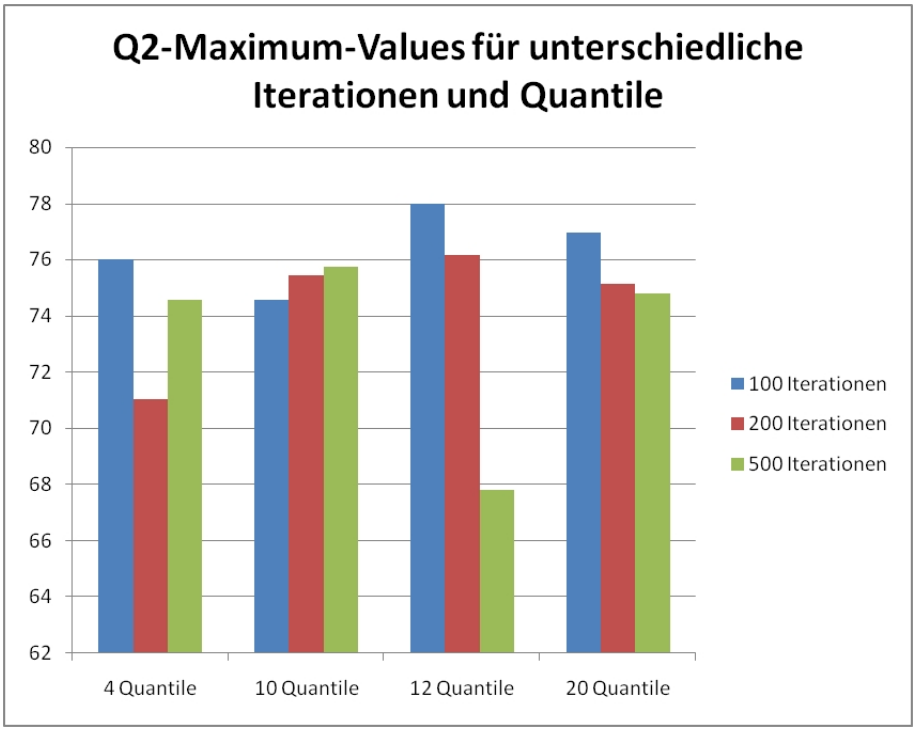

Abbildung 4.8: Diese Abbildung zeigt die maximalen Q2-Werte für die einzelnen Iterationen der unterschiedlichen Datensätze.

Ähnliches lässt sich bei den maximalen Q3-Werte erkennen (siehe Abbildung **??**). Hier sticht ebenfalls der Datensatz mit 12 Quantilen hervor mit einem Wert von 60%. Allerdings wird dieses Maximum nicht bei 100 Iterationen wie beim Q2-Wert erreicht, sondern bei 200 Iterationen.
Letztlich lässt sich aus dieser Analyse ableiten, dass der Datensatz mit 12 Quantilen die besten Ergebnisse liefert und somit als Standard für des 3^3TMR-Modell verwendet werden sollte.

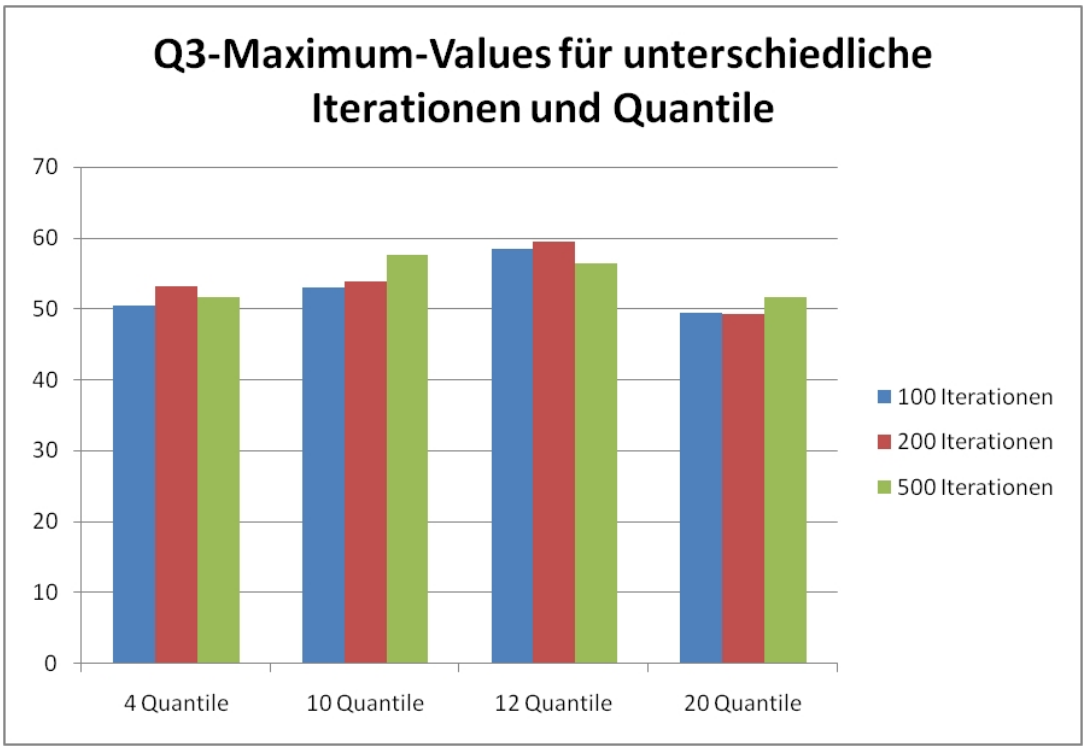

Abbildung 4.9: Diese Abbildung zeigt die maximalen Q3-Werte für die einzelnen Iterationen der unterschiedlichen Datensätze.

Betrachtet man vergleichend die Ergebnisse e^{3}TMR Modells mit denen des TMHMMs so lässt sich feststellen, dass die Genauigkeit von e^{3}TMR mit 60% noch weit unter der des TMHMMs mit 77,1%[?] liegt . Ebenso liegt der Q3-Wert deutlich unter dem des HMM_RA-Modells mit 80,6%[?]. Trotzdem ist dieses Ergebnis bei einem statistischen Erwartungswert von 33% sehr gut.

5 Fazit

Die Arbeit hat gezeigt, welche Möglichkeiten sich aus Topologievorhersagen mit Energieprofilen eröffnen.
Es konnten wertvolle Erkenntnisse für die Erstellung der Datensätze gewonnen werden. Hier stand anfangs die Vermutung, dass sich die Vorhersagequalität des Modells direkt proportional zur Anzahl der verwendeten Quantile im Datensatz verhält. Diese Annahme beruhte wiederum auf der Vermutung, dass steigende Quantilzahlen ein Energieprofil genauer abbilden können. Es ließ sich feststellen, dass zwar das Energieprofil deutlicher abgebildet werden konnte, eine Verbesserung der Vorhersagegenauigkeit stand damit aber nicht im direkten im Zusammenhang. Vielmehr war es möglich einen Datensatz heraus zu kristallisieren, der den Erfordernissen am ehesten entsprach.
Des weiteren wurde festgestellt, dass bei höheren Quantilzahlen die Rechenzeit für das Training des Modells stark zunimmt, womit eine Speicherung des trainierten Modells für eine spätere Webimplementierung der Software unablässig ist. Die Implementierte Fähigkeit mit unterschiedlichen Datensätzen zu arbeiten, ermöglichte an dieser Stelle eine zusätzliche Nutzerfreundlichkeit. Es ist dem Webuser nun möglich selbst auszuwählen welchen Datensatz er verwenden möchte und ermöglicht außerdem eine weitere Verbesserung des Modells ohne großen Aufwand.
Das Hauptziel der Arbeit bestand darin ein Vorhersagemodell zu entwickeln, welches anhand von Energieprofilen gute Topologievorhersagen trifft. Dies ist im Rahmen dieser Arbeit auch gelungen. Ein Q3-Wert von 60% liegt weit über dem Erwartungswert und beweist damit, dass Energieprofile gut für Topologievorhersagen geeignet sind.

6 Ausblick

Ausblickend lässt sich sagen, dass die Vorhersagegenauigkeit des Modells noch weiter verbessert werden kann. Hierfür soll eine neuer Trainingsdatensatz, basierend auf den Trainingsdaten des TMHMM, angelegt werden. Außerdem könnten die Regeln des Modells zusätzlich verfeinert werden. Als Vorbild hierfür könnte ebenfalls die Architektur des TMHMM oder HMMTOP dienen. Des weiteren wird der Ansatz verfolgt, basierend auf der Arbeit von Jing Hu[?] ein reduziertes Alphabet für Energieprofile einzuführen. Dies würde weitere statistische Analysen der Datensätze in Bezug auf die Ausprägungen des Quantilauftretens in definierten Topologien erfordern. Zusätzlich soll die Software als Webanwendung zur Verfügung gestellt werden. Hierfür müsste das eGOR-System, welches die Energiedaten für unbekannte Sequenzen liefert, an die neuen Erfordernisse in Bezug auf Quantile und deren Trennung, zur Analyse für das Programm angepasst werden.

Anhang A: Validierungsergebnisse

Tabelle A.1:

Q3 and Q2 for 4 Quantiles @ 100 Iterations			
QValues	adapted		
Q2 RUN_1	69,778672	Q3 RUN_1	45,9959759
Q2 RUN_10	72,1085409	Q3 RUN_10	49,866548
Q2 RUN_2	73,1073446	Q3 RUN_2	46,0338983
Q2 RUN_3	74,7420711	Q3 RUN_3	47,0003821
Q2 RUN_4	72,783665	Q3 RUN_4	45,5673301
Q2 RUN_5	68,7945792	Q3 RUN_5	39,5149786
Q2 RUN_6	75,2553389	Q3 RUN_6	50,4178273
Q2 RUN_7	73,1417864	Q3 RUN_7	44,4722049
Q2 RUN_8	68,5427441	Q3 RUN_8	43,8697901
Q2 RUN_9	71,5620828	Q3 RUN_9	42,8037383
Q3 and Q2 for 4 Quantiles @ 200 Iterations			
QValues	adapted		
Q2 RUN_1	70,221328	Q3 RUN_1	50,4225352
Q2 RUN_10	72,4644128	Q3 RUN_10	52,2686833
Q2 RUN_2	73,7175141	Q3 RUN_2	46,8926554
Q2 RUN_3	74,9331295	Q3 RUN_3	42,911731
Q2 RUN_4	72,5510468	Q3 RUN_4	48,5655208
Q2 RUN_5	67,6890157	Q3 RUN_5	40,6918688
Q2 RUN_6	75,998143	Q3 RUN_6	53,2033426
Q2 RUN_7	73,4540912	Q3 RUN_7	44,1599001
Q2 RUN_8	68,8469729	Q3 RUN_8	44,4782476
Q2 RUN_9	71,8558077	Q3 RUN_9	42,670227

Tabelle A.2:

Q3 and Q2 for 4 Quantiles @ 500 Iterations			
QValues	adapted		
Q2 RUN_1	55,1710262	Q3 RUN_1	38,6317907
Q2 RUN_10	66,3701068	Q3 RUN_10	49,2882562
Q2 RUN_2	50,8248588	Q3 RUN_2	37,8983051
Q2 RUN_3	68,9721055	Q3 RUN_3	40,5043943
Q2 RUN_4	60,8167485	Q3 RUN_4	46,988886
Q2 RUN_5	47,681883	Q3 RUN_5	41,2268188
Q2 RUN_6	71,0306407	Q3 RUN_6	48,2358403
Q2 RUN_7	60,8994379	Q3 RUN_7	34,8532167
Q2 RUN_8	63,5229693	Q3 RUN_8	
Q2 RUN_9	55,0600801	Q3 RUN_9	38,4512684
Q3 and Q2 for 10 Quantiles @ 100 Iterations			
QValues	adapted		
Q2 RUN_1	68,0783818	Q3 RUN_1	42,2882427
Q2 RUN_10	72,4965894	Q3 RUN_10	47,9126876
Q2 RUN_2	66,7377399	Q3 RUN_2	50,6929638
Q2 RUN_3	69,9703655	Q3 RUN_3	46,2956865
Q2 RUN_4	68,0491551	Q3 RUN_4	38,9912954
Q2 RUN_5	72,5901398	Q3 RUN_5	51,5084621
Q2 RUN_6	66,8051708	Q3 RUN_6	39,242844
Q2 RUN_7	62,0952381	Q3 RUN_7	33,3333333
Q2 RUN_8	74,5793901	Q3 RUN_8	46,1619348
Q2 RUN_9	73,8262911	Q3 RUN_9	52,9342723

Tabelle A.3:

Q3 and Q2 for 10 Quantiles @ 200 Iterations			
QValues	adapted		
Q2 RUN_1	69,721871	Q3 RUN_1	42,9835651
Q2 RUN_10	71,8417462	Q3 RUN_10	51,3233288
Q2 RUN_2	68,1769723	Q3 RUN_2	47,1215352
Q2 RUN_3	69,8386566	Q3 RUN_3	45,3078696
Q2 RUN_4	67,4859191	Q3 RUN_4	41,0394265
Q2 RUN_5	70,6033848	Q3 RUN_5	48,8226637
Q2 RUN_6	65,6509695	Q3 RUN_6	37,8116343
Q2 RUN_7	63,6190476	Q3 RUN_7	39,047619
Q2 RUN_8	75,446898	Q3 RUN_8	47,0031546
Q2 RUN_9	73,2981221	Q3 RUN_9	53,9319249
Q3 and Q2 for 10 Quantiles @ 500 Iterations			
QValues	adapted		
Q2 RUN_1	69,4690265	Q3 RUN_1	39,6965866
Q2 RUN_10	73,0968622	Q3 RUN_10	49,9317872
Q2 RUN_2	68,3901919	Q3 RUN_2	48,6140725
Q2 RUN_3	72,0777083	Q3 RUN_3	45,1103062
Q2 RUN_4	68,8684076	Q3 RUN_4	46,1853559
Q2 RUN_5	71,6335541	Q3 RUN_5	40,544518
Q2 RUN_6	68,6980609	Q3 RUN_6	37,6731302
Q2 RUN_7	65,1047619	Q3 RUN_7	34,2095238
Q2 RUN_8	75,7360673	Q3 RUN_8	50,7886435
Q2 RUN_9	73,8262911	Q3 RUN_9	57,5704225
Q3 and Q2 for 12 Quantiles @ 100 Iterations			
QValues	adapted		
Q2 RUN_1	63,8441364	Q3 RUN_1	45,0730611
Q2 RUN_10	58,9353612	Q3 RUN_10	48,035488
Q2 RUN_2	73,9130435	Q3 RUN_2	42,9385307
Q2 RUN_3	73,854898	Q3 RUN_3	53,293213
Q2 RUN_4	72,5104127	Q3 RUN_4	44,982961
Q2 RUN_5	76,7334361	Q3 RUN_5	56,05547
Q2 RUN_6	69,8060942	Q3 RUN_6	52,5392428
Q2 RUN_7	63,2987095	Q3 RUN_7	38,4639597
Q2 RUN_8	77,9941981	Q3 RUN_8	58,4334853
Q2 RUN_9	72,2683706	Q3 RUN_9	46,8370607

Tabelle A.4:

12 Quantiles @ 200 Iterations			
QValues	adapted		
Q2 RUN_1	63,8441364	Q3 RUN_1	44,3237167
Q2 RUN_10	58,2382763	Q3 RUN_10	45,0570342
Q2 RUN_2	71,844078	Q3 RUN_2	50,6146927
Q2 RUN_3	72,1832163	Q3 RUN_3	51,0197258
Q2 RUN_4	71,601666	Q3 RUN_4	47,8985233
Q2 RUN_5	75,1001541	Q3 RUN_5	51,155624
Q2 RUN_6	70,2677747	Q3 RUN_6	57,6638966
Q2 RUN_7	61,5360403	Q3 RUN_7	38,4954359
Q2 RUN_8	76,1707418	Q3 RUN_8	59,5109822
Q2 RUN_9	71,4376997	Q3 RUN_9	46,8051118
Q3 and Q2 for 12 Quantiles @ 500 Iterations			
QValues	adapted		
Q2 RUN_1	60,9216935	Q3 RUN_1	44,6234545
Q2 RUN_10	55,6717364	Q3 RUN_10	45,5006337
Q2 RUN_2	62,7586207	Q3 RUN_2	39,4302849
Q2 RUN_3	67,0678703	Q3 RUN_3	49,180876
Q2 RUN_4	65,391897	Q3 RUN_4	45,0586899
Q2 RUN_5	67,7966102	Q3 RUN_5	50,1078582
Q2 RUN_6	69,0674054	Q3 RUN_6	51,6158818
Q2 RUN_7	51,4321687	Q3 RUN_7	29,7765187
Q2 RUN_8	69,4571073	Q3 RUN_8	56,4028181
Q2 RUN_9	66,9648562	Q3 RUN_9	47,5399361

Tabelle A.5:

Q3 and Q2 for 20 Quantiles @ 100 Iterations			
QValues	adapted		
Q2 RUN_1	73,2818107	Q3 RUN_1	47,8056859
Q2 RUN_10	70,6720978	Q3 RUN_10	41,7515275
Q2 RUN_2	76,9699517	Q3 RUN_2	48,0773272
Q2 RUN_3	74,243215	Q3 RUN_3	49,3997912
Q2 RUN_4	62,6216968	Q3 RUN_4	39,6383866
Q2 RUN_5	63,7217391	Q3 RUN_5	41,8086957
Q2 RUN_6	68,0136112	Q3 RUN_6	46,6184602
Q2 RUN_7	69,6465696	Q3 RUN_7	44,2233442
Q2 RUN_8	69,8758704	Q3 RUN_8	46,5940054
Q2 RUN_9	65,4939107	Q3 RUN_9	44,6549391
Q3 and Q2 for 20 Quantiles @ 200 Iterations			
QValues	adapted		
Q2 RUN_1	73,6682307	Q3 RUN_1	44,5211151
Q2 RUN_10	70,0203666	Q3 RUN_10	44,5621181
Q2 RUN_2	75,1418365	Q3 RUN_2	48,9598655
Q2 RUN_3	73,9300626	Q3 RUN_3	49,3736952
Q2 RUN_4	63,2475661	Q3 RUN_4	39,8470097
Q2 RUN_5	63,6869565	Q3 RUN_5	41,7391304
Q2 RUN_6	69,5448745	Q3 RUN_6	49,2556359
Q2 RUN_7	70,1217701	Q3 RUN_7	46,3320463
Q2 RUN_8	70,0575235	Q3 RUN_8	47,9261278
Q2 RUN_9	63,9715832	Q3 RUN_9	43,0987821
FriatJulat27at10-12-01atCESTat2012			
Q3 and Q2 for 20 Quantiles @ 500 Iterations			
QValues	adapted		
Q2 RUN_1	67,9736025	Q3 RUN_1	46,5450311
Q2 RUN_2	70,4196933	Q3 RUN_2	44,5520581
Q2 RUN_3	74,456407	Q3 RUN_3	50,4116529
Q2 RUN_4	66,2286465	Q3 RUN_4	43,9224704
Q2 RUN_5	74,8152977	Q3 RUN_5	51,1516732
Q2 RUN_6	71,4621533	Q3 RUN_6	51,6220028

Literaturverzeichnis

[] http://daten.didaktikchemie.uni-bayreuth.de/umat/proteine/proteine_strukturen.htm

[] *Markov-Modell 1. Ordnung*. http://upload.wikimedia.org/wikipedia/de/math/1/2/1/1212b4c6fcf4828ba5833ce28b80c078.png

[] *Andrei Andreijewitsch Markov*. http://upload.wikimedia.org/wikipedia/commons/a/a8/Andrei_Markov.jpg. Version: 01.07.2012

[] *Hidden Markov Modell*. http://upload.wikimedia.org/wikipedia/commons/2/2e/HiddenMarkovModel.png. Version: 03.07.2012

[] *National Human Genome Research Institute*. http://www.genome.gov/Glossary/index.cfm?id=27. Version: 06.06.2012

[] *Datentyp String*. www.iti.fh-flensburg.de/lang/prog/string.htm. Version: 12.06.2012

[] *Peptidbindung-Kondensationsreaktion*. http://www.u-helmich.de/bio/stw/biokatalyse/katalyse02.html. Version: 12.06.2012

[] *The Transmembrane Protein Databank*. http://pdbtm.enzim.hu/?m=manual&man=116. Version: 20.07.2012

[] ANDERS KROGH1, Gunnar von H. Bjorn Larsson L. Bjorn Larsson ; SONNHAMMER, Erik L. L.: Predicting Transmembrane Protein Topology with a Hidden Markov Model: Application to Complete Genomes. In: *J. Mol. Biol.* 305 (2001), S. 567–580

[] BERLIN, Humboldt-Universitaet zu: *Wahrscheinlichkeitstheorie*. http://mars.wiwi.hu-berlin.de/mediawiki/statwiki/index.php/Wahrscheinlichkeitstheorie. Version: 25.07.2012

[] DIMMELER, S.: *Zellulaere und molekulare Grundlagen des Immunsystems*. http://campus.doccheck.com/uploads/tx_dcmedstudscripts/5807_zellulaere_und_molekulare_grundlagen_des_immunsystems_abgabe_version_.pdf. Version: 30.06.2012

[] DRESSEL, F.: *Sequenz, Energie, Struktur-Untersuchungen zur Beziehung zwischen Primär- und Tertiärstruktur in globulären und Membran-Proteinen*, Technische Universität Dresden, Diss., 2008

[] FINK, Gernot A.: *Mustererkennung mit Markov-Modellen*. Bd. 1. Teubner Verlag : Wiesbaden, 2003

[] FLORIAN, Heinke: *Energieprofilbasierende Analysemethoden von Proteinfamilien*. 2010

[] GRUMBT, Barbara: *Der MHC-Locus und das HLA-System*. `http://www.medizinische-genetik.de/index.php?id=1475`. Version: 28.06.2012

[] HUETT, Marc-Thorsten: *Methoden der Bioinformatik*. Heidelberg : Springer Verlag, 2006

[] JING HU, Changhui Y.: HMM_RA: An Improved Method for Alpha-Helical Transmembrane Protein Topology Prediction. (2008), Nr. 2, S. 67–74

[] MICHEL, Hartmut: *Molekulare-Membranbiologie*. `http://www.biophys.mpg.de/es/institut/molekulare-membranbiologie.html`. Version: 12.06.2012

[] NELSON, David; Micheal C.: *Lehninger Biochemie*. Berlin : Springer Verlag, 2009

[] RAYMOND DINGLEDINE, DEREK BOWIE STEPHEN F. T. KARIN BORGES B. KARIN BORGES: The Glutamate Receptor Ion Channels. (1999), Nr. 1

[] S. JAYASINGHE, K. H. ; WHITE, S. H.: MPtopo: A database of membrane protein topology. In: *Protein Sci.* 10 (2001), S. 455–458

[] SCHEIBE, Renate: *Biologie*. Bd. 9. München : Pearson, 2009

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Mittweida, 22.08.2012

Rico Hellwig