Optimization in Bioinformatics

Dissertation

zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)
der Naturwissenschaftlich-Technischen Fakultät I
– Mathematik und Informatik –
der Universität des Saarlandes

vorgelegt von

Dipl. Inform. Alexander Rurainski

Saarbrücken
2010

Datum des Kolloquiums:	07.05.2010
Dekan:	Prof. Dr. Holger Hermanns, Saarbrücken
Mitglieder des Promotionsausschusses:	Prof. Dr. Reinhard Wilhelm (Vorsitzender) Prof. Dr. Hans-Peter Lenhof Prof. Dr. Matthias Hein Dr. Thomas In der Rieden

Eidesstattliche Erklärung

"Hiermit versichere ich an Eides statt, dass ich diese Arbeit 'Optimization in Bioinformatics' selbständig und nur unter Verwendung der angegebenen Quellen angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quellen gekennzeichnet. Ich habe diese Arbeit bisher weder im Innoch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt."

Saarbrücken, Mai 2010

Alexander Rurainski

Danksagungen

Ich möchte mich an dieser Stelle ganz herzlich bei Herrn Prof. Dr. Hans-Peter Lenhof für die Aufgabenstellung und die Freiheit, die ich in den vergangenen Jahren genießen durfte, bedanken. Ebenso möchte ich mich für das Vertrauen bedanken, das er in mich gesetzt hat.

Weiterhin möchte ich mich bei Prof. Dr. Matthias Hein für viele fruchtbare Gespräche und seine Unterstützung bedanken. Er hatte in den vergangenen Jahren immer ein offenes Ohr und hat bei verschiedenen Projekten mit seiner Erfahrung auf dem Gebiet des maschinellen Lernens sehr weitergeholfen.

Natürlich möchte ich nicht versäumen, Dr. Dirk Neumann für die vielen inspirierenden Gespräche und gute Zusammenarbeit zu danken. Großer Dank gilt hierfür insbesondere auch Dr. Jan Fuhrmann, der als jahrelanger Zimmernachbar in vielen unnachahmlichen Situationen das gemeinsame Leben am Zentrum für Bioinformatik angenehmer gestaltete.

Danken möchte ich auch vielen Kollegen am Zentrum für Bioinformatik, die durch die gute Zusammenarbeit viele Ergebnisse beschleunigt haben. Ihnen möchte ich auch für eine lustige und angenehme Arbeitsatmosphäre danken. Insbesondere möchte ich hier Christina Backes, Dr. Jan Küntzer, Stephan Roth, Thomas Schackmann, Anne Dehof, Dr. Andreas Hildebrandt, Dr. Andreas Moll, Rene Hussong, Sophie Weggler, Dr. Andreas Keller und Benny Kneissl nennen.

Abschließend möchte ich meiner Familie danken, die mich auf meinem Weg immer unterstützt hat. Auch ohne direktes Zutun wäre diese Arbeit ohne sie wohl in der vorliegenden Form nie zustande gekommen.

Abstract

In this work, we present novel optimization approaches for important bioinformatical problems. The first part deals mainly with the local optimization of molecular structures and its applications to molecular docking, while the second part discusses discrete global optimization. In the first part, we present a novel algorithm to an old task: find the next local optimum into a given direction on a molecular potential energy function (line search). We show that replacing a standard line search method with the new algorithm reduced the number of function/gradient evaluations in our test runs down to 47.7% (down to 85% on average) . Then, we include this method into our novel approach for locally optimizing flexible ligands in the presence of their receptors, which we describe in detail, avoiding the singularity problem of orientational parameters. We extend this approach to a full ligand-receptor docking program using a Lamarckian genetic algorithm. Our validation runs show that we gained an up to tenfold speedup in comparison to other tested methods. Then, we further incorporate side chain flexibility of the receptor into our approach and introduce limited backbone flexibility by interpolating between known extremal conformations using spherical linear extrapolation. Our results show that this approach is very promising for flexible ligand-receptor docking. However, the drawback is that we need known extremal backbone conformations for the interpolation. In the last section of the first part, we allow a loop region to be fully flexible. We present a new method to find all possible conformations using the Gō-Scheraga ring closure equations and interval arithmetic. Our results show that this algorithm reliably finds alternative conformations and is able to identify promising loop/ligand complexes of the studied example. In the second part of this work, we describe the bond order assignment problem for molecular structures. We present our novel linear 0-1-programming formulation for the very efficient computation of all optimal and suboptimal bond order assignments and show that our approach does not only outperform the original heuristic approach of Wang et al. but also commonly used software for determining bond orders on our test set considering all optimal results. This test set consists of 761 thoroughly prepared drug like molecules that were originally used for the validation of the Merck Molecular Force Field. Then, we present our filter method for feature subset selection that is based on mutual information and uses second order information. We show our mathematically well motivated criterion and, in contrast to other methods, solve the resulting optimization problem exactly by quadratic 0-1-programming. In the validation runs, our method could achieve in 18 out of 21 test scenarios the best classification accuracies. In the last section, we give our integer linear programming formulation for the detection of deregulated subgraphs in regulatory networks using expression profiles. Our approach identifies the subnetwork of a certain size of the regulatory network with the highest sum of node scores. To demonstrate the capabilities of our algorithm, we analyzed expression profiles from nonmalignant primary mammary epithelial cells derived from BRCA1 mutation carriers and epithelial cells without BRCA1 mutation. Our results suggest that oxidative stress plays an important role in epithelial cells with BRCA1 mutations that may contribute to the later development of breast cancer. The application of our algorithm to already published data can yield new insights. As expression data and network data are still growing, methods as our algorithm will be valuable to detect deregulated subgraphs in different conditions and help contribute to a better understanding of diseases.

German Abstract

In der vorliegenden Arbeit präsentieren wir neue Optimierungsansätze für wichtige Probleme der Bioinformatik. Der erste Teil behandelt vorwiegend die lokale Optimierung von Molekülen und die Anwendung beim molekularen Docking. Der zweite Teil diskutiert diskrete globale Optimierung. Im ersten Teil präsentieren wir einen neuartigen Algorithmus für ein altes Problem: finde das nächste lokale Optimum in einer gegebenen Richtung auf einer Energiefunktion (Liniensuche, "line search"). Wir zeigen, dass die Ersetzung einer Standardliniensuche mit unserer neuen Methode die Anzahl der Funktions- und Gradientauswertungen in unseren Testläufen auf bis zu 47.7% reduzierte (85% im Mittel). Danach nehmen wir diese Methode in unseren neuen Ansatz zur lokalen Optimierung von flexiblen Liganden im Beisein ihres Rezeptors auf, den wir im Detail beschreiben. Unser Verfahren vermeidet das Singularitätsproblem von Orientierungsparametern. Wir erweitern diese Methode zu einem vollständigen Liganden-Rezeptor-Dockingprogramm, indem wir einen Lamarck'schen genetischen Algorithmus einsetzen. Unsere Validierungsläufe zeigen, dass wir im Vergleich zu anderen getesteten Methoden einen bis zu zehnfachen Geschwindigkeitszuwachs erreichen. Danach arbeiten wir in unseren Ansatz Seitenketten- und begrenzte Backboneflexibilität ein, indem wir zwischen bekannten Extremkonformationen mittels sphärischer linearer Extrapolation interpolieren. Unsere Resultate zeigen, dass unsere Methode sehr viel versprechend für flexibles Liganden-Rezeptor-Docking ist. Dennoch hat dieser Ansatz den Nachteil, dass man bekannte Extremkonformationen des Backbones für die Interpolation benötigt. Im letzten Abschnitt des ersten Teils behandeln wir eine Loopregion voll flexibel. Wir zeigen eine neue Methode, die die Gō-Scheraga Ringschlussgleichungen und Intervalarithmetik nutzt, um alle möglichen Konformationen zu finden. Unsere Resultate zeigen, dass dieser Algorithmus zuverlässig in der Lage ist, alternative Konformationen zu finden. Er identifiziert sehr vielversprechende Loop-Ligandenkomplexe unseres Testbeispiels. Im zweiten Teil dieser Arbeit beschreiben wir das Bindungsordnungszuweisungsproblem von Molekülen. Wir präsentieren unsere neuartige Formulierung, die auf linearer 0-1-Programmierung basiert. Dieser Ansatz ist in der Lage sehr effizient alle optimalen und suboptimalen Bindngsordnungszuweisungen zu berechnen. Unsere Methode ist nicht nur besser als der ursprüngliche Ansatz von Wang et al., sondern auch weitverbreiteter Software zur Bindungszuordnung auf unserem Testdatensatz überlegen. Dieser Datensatz besteht aus 761 sorgfältig präparierten, arzneimittelähnlichen Molekülen, die ursprünglich zur Validierung des Merck-Kraftfeldes eingesetzt wurden. Danach präsentieren wir unsere Filtermethode zur "Feature Subset Selection", die auf "Mutual Information" basiert und Informationen zweiter Ordnung nutzt. Wir geben unser mathematisch motiviertes Kriterium an und lösen das resultierende Optimierungsproblem global optimal im Gegensatz zu anderen Ansätzen. In unseren Validierungsläufen konnte unsere Methode in 18 von 21 Testszenarien die beste Klassifizierungsrate erreichen. Im letzten Abschnitt geben wir unsere, auf linearer 0-1-Programmierung basierende Formulierung zur Berechnung von deregulierten Untergraphen in regulatorischen Netzwerken an. Die Basisdaten für diese Methode sind Expressionsprofile. Unser Ansatz identifiziert die Unternetze einer gewissen Größe mit der höchsten Summe der Knotenscores. Wir analysierten Expressionsprofile von nicht bösartigen Brustepithelzellen von BRCA1 Mutationsträgern und Epithelzellen ohne BRCA1 Mutation, um die Fähigkeiten unseres Algorithmuses zu demonstrieren. Unsere Resultate legen nahe, dass oxidativer Stress eine wichtige Rolle bei Epithelzellen mit BRCA1 Mutation spielt, der zur späteren Entwicklung von Brustkrebs beitragen könnte. Die Anwendung unseres Ansatzes auf bereits publizierte Daten kann zu neuen Erkenntnissen führen. Da sowohl

Expressions- wie auch Netzwerkdaten ständig anwachsen, sind es Methoden wie unser Algorithmus die wertvoll sein werden, um deregulierte Subgraphen in verschiedenen Situationen zu entdecken. Damit trägt unser Ansatz zu einem besseren Verständnis von Krankheiten und deren Verlauf bei.

German Summary

Bioinformatik ist ein schnell wachsendes und interdisziplinäres Gebiet der Wissenschaft. Zielsetzung ist die Anwendung von Informationstechnologien auf biologische Fragestellungen. Heutzutage umfasst Bioinformatik die Anwendung und Verbesserung oder gar ganz die Neuentwicklung von Berechnungsmodellen, statistische Techniken, Algorithmen, Datenbanken und Theorien, um Probleme zu lösen, die sich bei der Verwaltung und Analyse von biologischen Daten stellen. Die Verbesserung unseres Verständnises von biologischen Prozessen, sowie die Unterstützung von Laborpersonal mit einer vorberechneten Auswahl an sinnvollen Versuchsabläufen sind Hauptziele in diesem Gebiet. Dadurch führt Bioinformatik nicht nur zu neuen wissenschaftlichen Erkenntnissen sondern auch zu einer Reduktion der Kosten durch den massiven Einsatz von Computern und Berechnungsmodellen.

Eine zentrale Aufgabe bei vielen bioinformatischen Problemstellungen ist Optimierung. Eine optimale Position, eine Struktur, eine Konformation, ein relevantes regulatorisches oder metabolischen Netzwerk, u.s.w. zu finden, ist der Schlüssel zu vielen bioinformatischen Herausforderungen. In der vorliegenden Arbeit behandeln wir Optimierungsprobleme von zentralen bioinformatischen Fragestellungen und deren Lösung.

Wir verbessern aktuelle numerische Methoden für die lokale Optimierung im Kontext von Energiefunktionen. Wir präsentieren einen neuartigen Ansatz für ein altes Problem: finde das nächste lokale Optimum in einer gegebenen Richtung auf einer Energiefunktion (Liniensuche, "line search"). Diese Aufgabenstellung tritt als Teilproblem bei vielen lokalen Optimierungsansätzen auf. Wir präsentieren eine neuartige Konsensusmethode die einfach an die lokalen Gegebenheiten der Zielfunktion angepasst werden kann, aber dennoch nicht mehr Funktionsauswertungen als Standardverfahren benötigt. Der zusätzliche Berechnungsaufwand ist vernachlässigbar. Unsere neue Methode hat nur durch die Ersetzung einer Standardliniensuche die Anzahl der Funktions- und Gradientauswertungen in unseren Testläufen auf bis zu 47.7% reduziert (85% im Mittel).

Im Folgenden gehen wir das Parametrisierungsproblem von molekularen Strukturen an. Unser Ansatz einen Liganden zu parametrisieren benutzt eine kompakte Darstellung, um die Anzahl der Freiheitsgrade zu reduzieren. Aufgrund des bekannten Singularitätsproblems von Orientierungsparametern kommt der Optimierungsprozess in der Praxis jedoch häufig an nicht-optimalen Positionen schon zum Erliegen. Wir zeigen, dass unsere Methode dieses Problem durch sehr effizientes Reparametrisieren vermeidet. Sie erlaubt es, hocheffiziente, gradientenbasierte Optimierungsverfahren zusammen mit der kompakten Darstellung beim molekularen Docking zu verwenden. Wir beschreiben dazu im Detail, wie unser Ansatz in die Optimierungsprozedur eingearbeitet wird. Insbesondere zeigen wir auf, dass unsere Liniensuche dazu sehr gut geeignet ist. Unsere Resultate verdeutlichen, dass diese Methode der in der Praxis weit verbreiteten stochastischen Methode von Solis und Wets weit überlegen ist. Das zeigt sich umso deutlicher, wenn Liganden mit wachsender Anzahl an Freiheitsgraden betrachtet werden.

Wir erweitern in dieser Arbeit unseren Ansatz zu einem Liganden-Rezeptor-Dockingprogramm, indem wir einen sog. Lamarck'schen genetischen Algorithmus einsetzen. Diese Heuristik wird benutzt, um verschiedene Startpositionen für unser Verfahren zur lokalen Optimierung zu generieren. Wir zeigen, dass diese kombinierte Methode deutlich anderen Ansätzen, die einen stochastischen lokalen Optimierungsansatz verwenden, überlegen ist. Der neue Algorithmus führt zu kürzeren Laufzeiten und liefert bemerkenswert bessere Resultate, insbesondere bei einer steigenden Anzahl von Freiheitsgraden der Liganden. Wir erhielten bis

zu zehnfache Geschwindigkeitszuwächse im Vergleich zu den anderen getesteten Methoden. Daher ist es durch diesen Algorithmus heute möglich, Dockingexperimente durchzuführen, bei denen die Liganden viele rotierbare Bindungen haben. Dieses Verfahren behandelt in seiner ursprünglichen Form jedoch den Rezeptor als starr, was eine sehr starke Annahme ist. Daher erweitern wir ihn und führen Seitenkettenflexibilität für den Rezeptor ein. Die Ideen dazu basieren auf der gleichen Parametrisierung wie wir sie für die Liganden einsetzen. Weiterhin erreichen wir eine beschränkte Backbone Bewegung indem wir zwischen zwei bekannten extremen Konformationen mittels sphärischer linearer Extrapolation interpoliern. Wir zeigen im Detail auf, wie wir für diesen Ansatz die Gradient berechnen, so dass unsere effizienten Methoden anwendbar sind. Dadurch sind wir in der Lage voll flexibles Liganden-Rezepor-Docking durchzuführen, wenn verschiedene Backbone-Konformationen bekannt sind. Für unsere Studie haben wir Human Serum Albumin gewählt, da dieses Protein für seine Fähigkeit verschiedene Arten von Liganden zu binden bekannt ist und dabei große Backbone-Bewegungen eine Rolle spielen. Unsere Resultate zeigen, dass dieser Ansatz sehr vielversprechend für flexibles Liganden-Rezeptor-Docking ist. Dennoch ist der Hauptnachteil, dass verschiedene Backbonekonformationen bekannt sein müssen, was gerade bei Loop-Regionen ein Problem darstellt, da diese hochflexibel und schlecht zu kristallisieren sind. Daher verwenden wir noch einen weiteren Ansatz, der die Loop-Region voll flexible behandelt. Wir modifizieren die Torsionswinkel des Backbones für flexibles Docking und bedienen uns dabei globalen Optimierungstechniken. Die spezielle Herausforderung hierbei ist, "reale" Konformationen zu generieren, d.h. die erhaltenen Loops müssen fest definierte Start- und Endpositionen haben. Gō und Scheraga haben 1970 ihre Gleichungen publiziert, um geschlossene Loops in Polypeptiden zu berechnen. Um diese Gleichungen zu lösen, benutzten sie die Newton-Methode. Dabei handelt es sich aber nur um eine lokale Suche. Nach unserem Wissen haben sie diese Methode mit gleichverteilten Startpositionen immer wieder gestartet. Daher gibt es keine Garantie, dass alle Lösungen gefuden werden. Höchstwahrscheinlich waren dazu auch viele Funktions- und Gradientauswertungen nötig. Wir präsentieren hier unsere, auf Intervalarithmetik basierende Methode alle Lösungen zu finden. In unseren Testläufen wurden die Resultate binnen Mikrosekunden gefunden. Für diese Studie haben wir "Human 17β-Hydroxysteroid Dehydrogenase Type 1" (17β-HSD 1) gewählt, da in vorbereitenden Studien, andere Ansätze mit 17β-HSD 1 flexibles Docking durchzuführen, fehlschlugen. Der Grund ist die Flexibilität der Loop-Region. Unsere Resultate belegen, dass dieser Algorithmus zuverlässig alternative Konformationen findet und in der Lage ist, vielversprechende Ligand-Loop-Komplexe des Testbeispiels zu berechnen.

Der letzte Teil dieser Arbeit behandelt diskrete Optimierungsprobleme in der Bioinformatik. Wir beschreiben zunächst das Problem der Zuweisung von Bindungsordnungen bei einem gegebenen Molekül. Bindungsordnungsinformationen können oft nicht direkt aus den vorhandenen experimentellen Daten erschlossen werden. Sogar wichtige molekulare Datenbanken wie die Cambridge Structural Database und die Protein Data Bank sind dafür bekannt, fehlerhafte Bindungsinformationen zu enthalten. In manchen Fällen sind gar keine Informationen enthalten. Für Aminosäuren und Proteine können Bindungsordnungen einfach durch ihren besonderen Aufbau bestimmt werden. Das gilt allerdings nicht für andere Arten von Molekülen wie beispielsweise Liganden. Es ist auch nicht praktikabel, die Ordnungen von Hand durch Experten im Falle von Tausenden von Molekülen beispielsweise für Virtual Screening zuzuweisen. Dadurch ist die automatisierte Zuweisung von Bindungsordnungen eine fundamentale Aufgabe bei der Arbeit mit Molekülen. Zu diesem Zweck wurden über die Jahre viele verschiedene Ansätze entwickelt, von denen die meisten auf korrekte Atomkoordinaten

angewiesen sind. Wir erweitern einen Ansatz von Wang et al., der heuristische Strafscores nur basierend auf Konnektivitätsinformationen zuweist. Wang et al. versuchen dieses Kriterium heuristisch zu optimieren, was zwei Nachteile hat: Die Scores der resultierenden Zuweisung sind nicht notwendigerweise optimal und es wird nur eine Zuweisung gefunden. Es kommt in der Praxis jedoch häufig vor, dass mehrere optimale Zuweisungen existieren. In dieser Arbeit präsentieren wir unsere neue, auf linearer 0-1-Programmierung basierende Formulierung, die sehr effizient alle optimalen und auch suboptimalen Zuweisungen berechnen kann. Unsere Resultate zeigen nicht nur, dass die neue Methode dem Originalansatz von Wang et al. klar überlegen ist, sondern unter Berücksichtigung aller optimalen Zuweisungen auch zwei weitverbreiteten Programmen für diese Aufgabenstellung, obwohl sie für unseren Testdatensatz ideal geeignet sein sollten. Sie vertrauen auf korrekte Atompositionen, und unser Testdatensatz besteht aus 761 sorgfältig präparierten, arzneistoffähnlichen Molekülen, die ursprünglich für die Validierung des Merck-Kraftfeldes eingesetzt wurden.

Im Weiteren behandeln wir die Aufgabenstellung eine optimale Untermenge aus einer Menge von gegebenen Features zu selektieren. Beim maschinellen Lernen besteht die Aufgabe der Klassifizierung darin, anhand von beschrifteten Beispielen ein Modell zu erlernen, das unbekannten Objekten in eine Klasse einordnet. In der Bioinformatik treten zahlreiche solcher Problemstellungen auf. Die Anwendungen reichen von der Unterscheidung von Krebs- und Normalgewebe bis zur Vorhersage der “Splice Site”. Es ist wichtig irrelevante oder redundante Features zu vermeiden, da sie einen negativen Effekt auf die Genauigkeit des Klassifizierers haben können. Anstatt mit allen vorhandenen Features zu arbeiten wird nur eine kleine Untermenge verwendet. Die Ziele dabei sind: (1) Reduzierung von Übertraining und dadurch Verbesserung der Klassifizierungsgenauigkeit, (2) die selektierten Features sind besser interpretierbar, was helfen kann, verschiedene Funktionstypen im Krankheitsverlauf zu identifizieren, (3) Dimensionsreduktion senkt die Anzahl der Berechnungen der Klassifizierungsalgoithmen. Die vorherrschenden Methoden sind Filteransätze und Wrapperverfahren. Letztere sind sehr aufwändig im Vergleich zu Filter. In dieser Arbeit präsentieren wir unsere Filter Methode, die “Mutual Information” und Informationen zweiter Ordnung verwendet, whrend andere Ansätze nur Informationen erster Ordnung benutzen. Unser Kriterium ist mathematisch gut motiviert und wird von uns exakt mittels quadratischer 0-1-Programmierung gelöst. In unseren Validierungsläufen konnte unsere Methode in 18 von 21 Testszenarien die beste Klassifizierungsgenauigkeit erreichen.

Zum Schluss präsentieren wir unsere Branch-and-Cut Methode für die Berechnung von deregulierten Untergraphen in regulatorischen Netzwerken basierend auf Expressionsprofilen. Dazu werden Scores auf die Gene des Netzwerks abgebildet, die die Deregulierung symbolisieren. In dieser Studie haben wir das menschliche, regulatorische Netz von KEGG verwendet. Die Unternetze werden anhand der Summe ihrer Knoten-Scores bewertet. Unser Ansatz identifiziert das Unternetz einer gewissen Grösse mit der höchsten Score-Summe. Diese Methode (und ihre Weiterentwicklungen) können entscheidend sein, um z.B. Therapien gegen verschiedene Arten von Krebs zu optimieren, indem Schlüsselgene durch unseren Ansatz identifiziert werden. Wir analysierten Expressionsprofile von nicht bösartigen Brustepithelzellen von BRCA1 Mutationsträgern und Epithelzellen ohne BRCA1 Mutation, um die Fähigkeiten unseres Verfahrens zu demonstrieren. Unsere Resultate legen nahe, dass oxidativer Stress eine wichtige Rolle bei Epithelzellen mit BRCA1 Mutation spielt, der zur späteren Entwicklung von Brustkrebs beitragen könnte. Die Anwendung unseres Ansatzes auf bereits publizierte Daten kann zu neuen Erkenntnissen führen. Da sowohl Expressions- wie auch Netzwerkdaten ständig anwachsen, sind es Methoden wie unser Algorithmus die wertvoll sein werden,

um deregulierte Subgraphen in verschiedenen Situationen zu entdecken. Damit trägt unser Ansatz zu einem besseren Verständnis von Krankheiten und deren Verlauf bei.

Abschließend müssen wir sagen, dass durch die vorliegende Arbeit die angegangenen Probleme zwar nicht als gegenstandslos betrachtet werden können, jedoch große Fortschritte in die richtige Richtung erzielt werden konnten. Es ist auch möglich, die entwickelten Ansätze und Algorithmen weiter zu verbessern. Bereits als diese Arbeit entstand, haben wir schon an Weiterentwicklungen gearbeitet, die aus Zeitgründen leider nicht mehr ihren Weg in diese Dissertation fanden. Weitere Untersuchungen werden zeigen, inwieweit Verbesserungen möglich sind.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Bioinformatics is a fast growing interdisciplinary field of science. It consists of the application of information technology to the field of molecular biology. Nowadays, bioinformatics entails the creation and advancement of computational and statistical techniques, algorithms, databases, and theory to solve problems arising from the management and analysis of biological data. Increasing our understanding of biological processes and supporting scientists in laboratories with precalculated selections of experimental setups ("virtual lab", see Figure 1.1 for a typical workflow) are main goals in this area. Hence, bioinformatics leads to new scientific insights on the one hand and cost reduction on the other hand by employing massive computer usage and computational models, which will be even more accurate in the future.

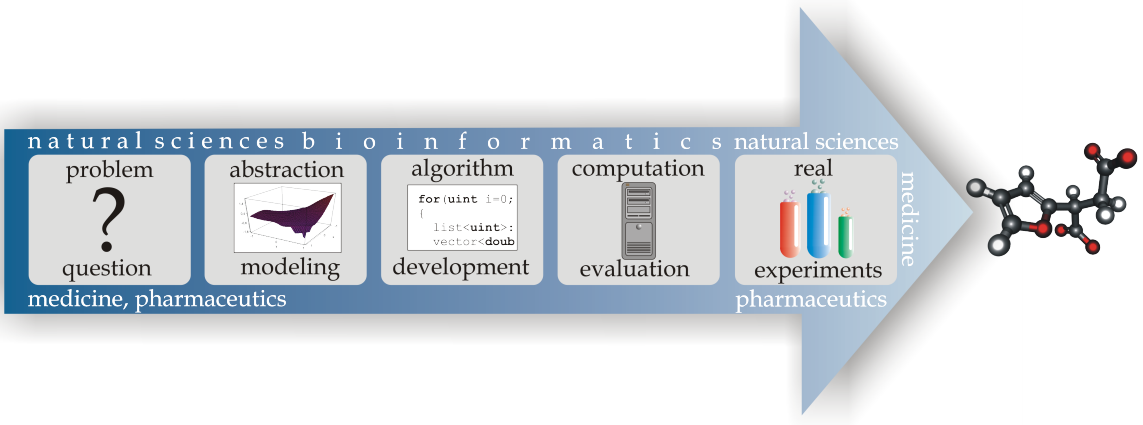

Figure 1.1: Typical workflow in drug design, nowadays with the aid of bioinformatics. From the question/task (left) to the real drug molecule(s) (right) ready for biological/pharmaceutical testing. During these studies, typically new/other questions arise and this workflow may be iterated until a new remedy could be identified.

A core task in many bioinformatical problems is optimization. Finding an optimal or nearly optimal position, a structure, a conformation, a relevant regulatory or metabolic (sub)network, etc. under certain conditions is the key to many bioinformatical challenges. In this work, we address optimization problems for central bioinformatical topics. We illustrate the basic questions by a small example, which is borrowed from biochemistry and

pharmaceutics. A laboratory head may ask:

(Q1) "How can we reduce experimental costs?"

(Q2) "What is the cheapest approach to a certain experiment (maybe under several constraints like the given equipment)?"

Each single answer saves money in practice. Both questions represent different kinds of optimization problems: *local optimization*, and *(provable) global optimization*. In order to address such a problem, the workflow has to be prepared for applying optimization techniques.

Parametrization and Scoring: If a computer is supposed to solve problems, e.g. questions (Q1) and (Q2), it needs to be able to handle the different experimental approaches and setups. Basically, a computer can only deal with numbers. Thus, the situation has to be mathematically modeled and parameters ("variables") have to be found that represent and describe the problem, see Figure 1.2. These values must uniquely define the different states. While the task of describing simple geometric objects in space might be obvious, e.g. the spatial arrangement of a molecule might be represented by the Cartesian coordinates of all its atoms as well as a set of bonds that defines their interconnectivity, the parametrization problem is not trivial at all for a whole experimental setup.

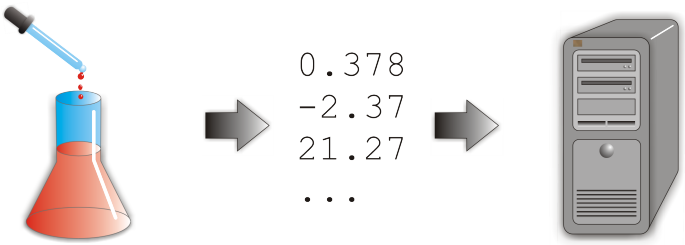

Figure 1.2: The parametrization problem. Before a computer can handle a situation, numbers have to be found that fully describe the corresponding state (parametrization).

In addition to the parametrization that permits a computer to distinguish between different states, a so-called *scoring function* is required that maps each set of variables onto a score (a real value) allowing the computer to assess the quality of solutions. For instance, in our laboratory example this function may measure the costs for carrying out the experimental procedure and, thus, lower values are better than higher values. Depending on the setup this function might be without theoretical margin. In bioinformatics, a scoring function mostly only approximates reality. If this function is improper for the considered question, we cannot expect to obtain a good outcome. However, we use well known scoring functions for several tasks like molecular structure optimization. Nevertheless, the scoring functions in Sections 4.2 and 4.3 are own developments and directly related to the corresponding optimization tasks.

A good visualization of the initial situation obtained so far is to identify the scoring function with mountains, see Figure 1.3. The height of each point is the value of the scoring function, e.g. costs in the laboratory example, whereas the coordinates are the parameters, e.g. they represent the associated setup. We want to reduce costs and, thus, we are looking for a deeper position, in fact, the deepest position we can find.

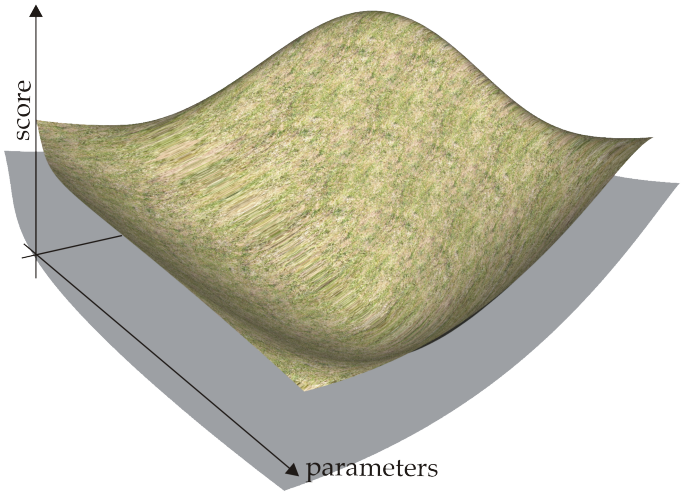

Figure 1.3: Sketch of a scoring function depicted as a grass covered mountain part. The geographical coordinates are the parameters, the height of each coordinate pair is the associated score.

Local Optimization: Question (Q1) symbolizes the task of *local optimization*. We already know a working setup, i.e. a position on the scoring function, and we want to improve it, i.e. we want to find a position with better score. Certainly, more cost reduction is better than less reduction in Question (Q1). Thus, we look for better scores until we cannot find directly a further improvement[1]. Human hikers in the mountains can receive much information through their eyes, whereas a computer is blind. A computer knows the current position and can evaluate the height at that point. Either by primitive testing or a bit more information it knows which direction leads to a lower score. Then the process of local optimization is given in a simplified manner by walking downhill in the mountains until no further direction will result in a lower score. Maybe the best real world correspondence is a ball manually positioned in the mountains, which rolls into the next hollow after its release, see Figure 1.4.

Heuristic Global Optimization: Certainly, any improvement is welcome. Thus, methods that help us to find deeper points are of high interest. Clearly, our hope is to find the global optimum, in our mountain example the deepest point at all. In this illustration we could escape from local minima and throw the ball into random directions, see Figure 1.5, let it roll, and store the found position if it is deeper than all previous identified positions. Such and other improvements fall into the category of *heuristics*. However, even if we find the global optimum, we are usually not able to prove that we found the deepest point at all. To be exact, we do not have any guarantee that the deepest point we identified is indeed the global optimum.

Global Optimization: The answer to question (Q2) might be crucial if new laboratory material or equipment has to be purchased. The costs can reach millions of euro and any however guarantee not to spend too much money is welcome. *Global optimization* stands for the aim to find the guaranteed global optimum, i.e. the deepest point at all with the proof

[1]It is important to note that in this case *only* a reduction is requested. Question (Q1) does not cover Question (Q2), which is difficult to answer in general.

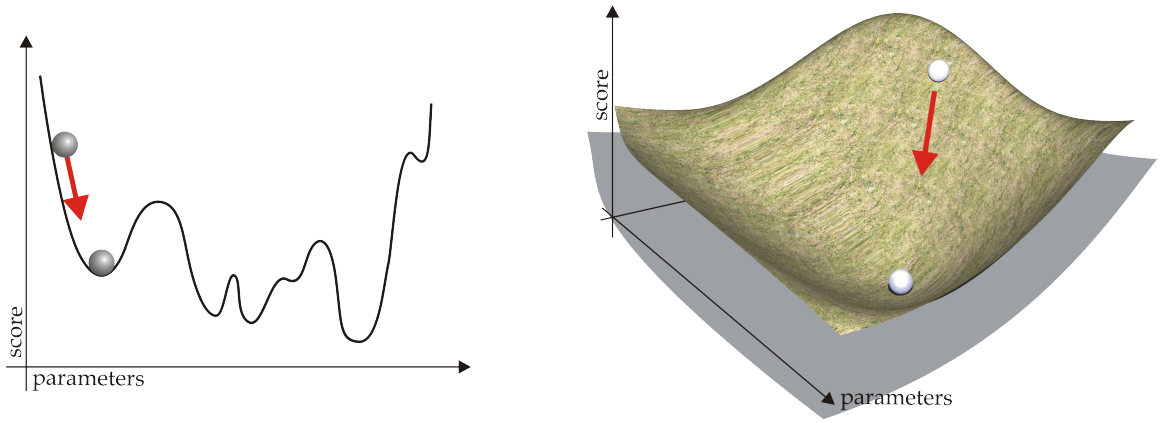

Figure 1.4: Procedure of local minimization depicted as a ball rolling into the next hollow. A computer calculates iteratively every position of the ball on its way using the current point to predict the following location.

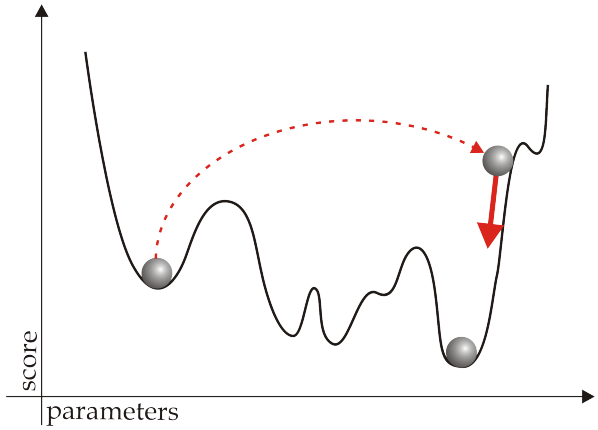

Figure 1.5: Possible heuristic. Throw the ball into a random direction and start a new local optimization process. Repeat this process for a predetermined number of times.

of global optimality. Of course, this is the hardest task at all. Only for special classes of optimization tasks this problem can be seen as "solved". For most other classes, not even in theory algorithms exist for general problems without special properties. Under mild assumptions the problem is solved if only one parameter is involved [70], but this does not hold with increasing number of degrees of freedom. Certainly, we can (and do) use heuristics to obtain acceptable results in practice. However, in global optimization, the most time consuming part is usually the proof that an already identified position is globally optimal.

In this thesis, we present novel optimization approaches for important bioinformatical problems. The subsequent chapters are organized as follows: Chapter 2 gives the mathematical background and associated formalism without special view of bioinformatical applications. Chapter 3 deals mainly with the local optimization of molecular structures and its applications to molecular docking, while Chapter 4 discusses discrete global optimization.

In Chapter 2, we distinguish several optimization problems considered in this work and

sketch the solution strategies. Where necessary, we prove facts that cannot be found in the literature, but in general we only state formulas that are important for an understanding of the topic or are used in the remaining chapters. This chapter can also be seen as an overview of up-to-date techniques for the addressed questions.

In Section 3.1, we improve current computational approaches for *local optimization* in the context of molecular potential energy functions. In fact, we present a novel, more modern algorithm [154] to an old task: find the next local optimum into a given direction on a molecular potential energy function (*line search*). This task arises as a part of many local optimization approaches. We present a new consensus like approach that can be easily adapted to the local behavior of objective functions, while it does not require additional function evaluations, imposing only negligible computational overhead. Replacing a standard line search method with the new algorithm reduced the number of function/gradient evaluations in our test runs down to 47.7% (down to 85% on average).

In Section 3.2, we deal with the above mentioned *parametrization problem* in the context of molecular representations [54]. Our approach to parameterize a ligand uses a compact representation to reduce the number of variables. However, due to the well-known singularity problem of orientational parameters, the optimization process in ligand-receptor docking may get stuck at non-optimum positions. We show that our method avoids this problem by computationally efficient reparametrization and enables gradient-based optimization of molecular complexes using the compact molecular representation. We give details on the incorporation of our orientational parametrization [54] into the *local optimization* procedure in Section 3.2. Especially, we show that our line search method is well suited for this task. Our results indicate that this approach is clearly superior to the stochastic method of Solis and Wets [164] widely used in practice. This becomes even more substantial if we consider ligands of increasing complexity.

We extend this local optimization procedure to a ligand-receptor docking approach [55] in Section 3.3. We present a so-called Lamarckian genetic algorithm, a *heuristic* for providing different start positions for the local optimization procedure. We show that this combined approach is clearly superior to other approaches employing a stochastic local optimization method. The new algorithm features shorter run times and gives substantially better results, especially with increasing complexity of the ligands. In our validation runs, we gained an up to tenfold speedup in comparison to other tested methods. Thus, it may be used to dock ligands with many rotatable bonds with high efficiency.

While the ligand is fully flexible in our docking method of Section 3.3, the receptor is kept rigid. In Section 3.4, we further incorporate side chain flexibility of the receptor with the same parametrization as we introduced for the ligand. Additionally, we achieve limited backbone movement by interpolating between two extremal conformations using spherical linear extrapolation [163] and give details on how the resulting gradient can be calculated. This enables application of the whole range of gradient-based optimization methods to flexible ligand-receptor docking when two or more extremal backbone conformations are known. In our study of Schackmann [156], see also Rurainski et al. [152], we chose human serum albumin (HSA), which is the most abundant transport protein of the human blood plasma. It is known for its promiscuity to bind different ligand species, and it is one of the most extensively studied proteins. The fact that HSA undergoes tremendous backbone movements upon binding to fatty acids, facilitating enormous topological and structural changes over the whole protein, rendered this molecule to an ideal candidate for our study. Our results show that this approach is very promising for flexible ligand-receptor docking.

Our approach of Section 3.4 needs known extremal backbone conformations for the interpolation. In general, we cannot assume that all necessary conformations can be obtained by interpolation between already known structures. In Section 3.5 we allow a loop region to be fully flexible, see also Rurainski et al. [151] and Roth [148]. We deal with the modification of backbone torsion angles for flexible docking [148] using *global optimization* techniques. The special challenge is here to calculate "real" conformations, i.e. the obtained loops must start and end at given positions. Crucial for this approach is to find all possible conformations. In 1970, Gō and Scheraga published equations whose solutions represent the loop closure in polypeptides [58]. They addressed the problem of solving these equations with Newton's method. Since this is only a local search method, they may have sampled the search space by providing different equally distributed start positions. Thus, there is no guarantee to find all solutions and, most probably, many function and derivative evaluations had to be performed. We present a new method to find all possible conformations using interval arithmetic [120, 129]. In our test runs, all results were obtained nearly instantly. For this study, we chose the human 17β-hydroxysteroid dehydrogenase type 1 (17β-HSD 1). In preliminary studies, different approaches to perform flexible docking with 17β-HSD 1 failed because of the high flexibility of the loop region. Our results show that this algorithm reliably finds alternative conformations and is able to identify promising loop/ligand complexes of the studied example.

Dealing with global optimization, we describe the bond order assignment problem for molecular structures in Section 4.1. Bond order information can often not be directly inferred from the available experimental data. Even important molecular databases, like the Cambridge Structural Database [6] and the Protein Data Bank (PDB) [18, 17], are known to contain erroneous data for connectivity and bond order information [106] or to omit them entirely. For nucleic acids and proteins bond orders can easily obtained due to their building block nature, but this does not hold for other kinds of molecules like ligands. Furthermore, it is not practicable to assign bond orders manually for, e.g., virtual screening purposes, where thousands of molecules are to be considered. Hence, automated bond order assingment is often a fundamental task for the work with molecules. Very different strategies have been applied to derive bond order information, most of them relying on the correctness of the atom coordinates. We extend an ansatz proposed by Wang et al. [182] that assigns heuristic molecular penalty scores solely based on connectivity information and tries to heuristically approximate its optimum. This procedure has two drawbacks: the scores of the resulting assignments are not guaranteed to be optimal and the algorithm provides only one solution while there can be more than one assignment with optimal score. Here, we present our novel linear 0-1-programming formulation for the very efficient computation of all optimal and suboptimal bond order assignments. We show that our approach does not only outperform the original heuristic approach of Wang et al. [182], but also commonly used software for determining bond orders on our test set considering all optimal results. It consists of 761 thoroughly prepared drug like molecules that were originally used for the validation of the Merck Molecular Force Field (MMFF94).

In Section 4.2, we address the task of selecting a subset of a given set of features based on mutual information, see also Rurainski et al. [153]. In machine learning, the problem of supervised classification is concerned with using labeled examples to induce a model that classifies objects into a finite set of known classes. Copious classification tasks occur in bioinformatics, such as distinguishing cancer tissues from normal tissues [7] or one cancer subtype vs another [5], predicting protein fold or super-family from its sequence [85, 43], etc.. Avoiding irrelevant or redundant features is important because they may have a negative effect

on the accuracy of the classifier. Instead of using all available features, only a subset is employed for classification tasks mainly with the following aims: (1) reduction of overfitting of the used learning methods and, hence, improvement of the classification accuracy, (2) the obtained features are more interpretable that can help identifying and monitoring the target diseases or function types, and, finally, (3) dimension reduction decreases the computational costs for the classification algorithms. The prevalent methods are filter approaches and wrapper type methods [35, 100]. The last-named methods are computationally very expensive in comparison to filters. Here, we present our filter method that uses second order information while other methods strongly rely only on first order information, see Sections 4.2.2 to 4.2.4. Furthermore, our criterion is mathematically well motivated and, in contrast to other methods, exactly solved by quadratic 0-1-programming. In the validation runs, our method could achieve in 18 out of 21 test scenarios the best classification accuracies.

In Section 4.3, we present our novel branch-and-cut approach for the determination of deregulated subgraphs in regulatory networks using expression profiles. In this study, scores indicating the deregulation of the genes are mapped onto the vertices of the KEGG [93, 94] human regulatory network. The subnetworks are assessed by the sum of their participating vertex scores. Our approach identifies the subnetwork of a certain size with the highest sum of node scores. The vision implicated by the proposed connectivity model is to identify – besides the most deregulated components – the root node that may represent a key player in the pathogenic process. This key player may be responsible for the observed differences between the investigated conditions and may serve as a potential target for therapy purposes. To demonstrate the capabilities of our algorithm, we analyzed expression profiles from nonmalignant primary mammary epithelial cells derived from BRCA1 mutation carriers and epithelial cells without BRCA1 mutation. Our results suggest that oxidative stress plays an important role in epithelial cells with BRCA1 mutations that may contribute to the later development of breast cancer. It is important to note that the application of our algorithm to already published data can yield new insights. As expression data and network data are still growing, methods as our algorithm will be valuable to detect deregulated subgraphs in different conditions and help contribute to a better understanding of diseases.

Chapter 2

Mathematical Background

In this chapter, we put the problems on a firm mathematical basis and formalize the different tasks we address in this work without special view of bioinformatical applications. Furthermore, we give a short overview of algorithms for the different tasks covered by this work and, dependent on the individual case, present them on a detailed level if it is required for the understanding of the following chapters.

2.1 Notation

In the rest of this work, we use the following notations: $\mathbb{R}$ denotes the field of real numbers and $\mathbb{Z} \subset \mathbb{R}$ the subset of integer values. We define $\mathbb{N} := \{x \in \mathbb{Z} \mid x > 0\}$ and $\mathbb{R}^n$, $n \in \mathbb{N}$, denotes the Euclidean n-dimensional vector space over $\mathbb{R}$ (n-tupel of real values). The $n \times m$ matrices with real valued entries are represented by $\mathbb{R}^{n \times m}$, $n, m \in \mathbb{N}$. Bold lower case letters, e.g. $\mathbf{x} \in \mathbb{R}^n$, symbolize vectors, bold upper case letters, e.g. $\mathbf{A} \in \mathbb{R}^{n \times m}$, matrices. Single real values are indicated by small italic letters, e.g. $y \in \mathbb{R}$, vector and matrix entries with additional corresponding index (indices), for example $\mathbf{x} = (x_i)_{i=1}^n$ and $\mathbf{A} = (a_{i,j})_{i,j=1}^n$. We denote the set of symmetric real $n \times n$ matrices by $\mathbb{S}^n$ and set

$$\langle \mathbf{A}, \mathbf{B} \rangle := \mathrm{tr}(\mathbf{A}^T \mathbf{B}) \quad \text{for all } \mathbf{A}, \mathbf{B} \in \mathbb{S}^n,$$

where tr symbolizes the trace, i.e.

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} a_{i,i}$$

for a matrix $\mathbf{A} = (a_{i,j})_{i,j=1}^n \in \mathbb{S}^n$. We denote the fact that $\mathbf{A}$ is positive semidefinite (definite) by

$$\mathbf{A} \succeq 0 \qquad (\mathbf{A} \succ 0)$$

based on the Löwner partial ordering [150]. The identity matrix is denoted by

$$\mathbf{I} := \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & \ddots & & 0 \\ \vdots & & \ddots & \\ 0 & \cdots & 0 & 1 \end{pmatrix} \in \mathbb{S}^n$$

and we expand "$\leq$" component-by-component to vectors: Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we set

$$\mathbf{x} \leq \mathbf{y} \quad \Leftrightarrow \quad x_i \leq y_i,\ 1 \leq i \leq n.$$

The gradient of a real valued function f is denoted by ∇f and $\nabla^2 f$ symbolizes the matrix of the second derivatives, its Hessian (matrix).

2.2 Problem Definition

Definition 2.2.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function that maps from $\mathbb{R}^n$, $n \in \mathbb{N}$, into $\mathbb{R}$.*

(a) The problem of finding $\mathbf{x}^* \in \mathbb{R}^n$ for which

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \qquad \text{for all } \mathbf{x} \in \mathbb{R}^n$$

*holds is called an **unconstrained global minimization problem**. Usually the problem is abbreviated to* [1]

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}). \tag{2.1}$$

*(b) The pair $(\mathbf{x}^*, f(\mathbf{x}^*))$ is called a **global minimum** and the point $\mathbf{x}^*$ a **global minimizer**, usually denoted by*

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}).$$

(c) If there exists $\varepsilon > 0$ and $\mathbf{x}^* \in \mathbb{R}^n$ for which

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \qquad \text{for all } \mathbf{x} \in \mathbb{R}^n \text{ with } \|\mathbf{x} - \mathbf{x}^*\| < \varepsilon$$

*holds, then $(\mathbf{x}^*, f(\mathbf{x}^*))$ is called a **local minimum** and the point $\mathbf{x}^*$ a **local minimizer**.*

(d) The set

$$\mathcal{B}_\varepsilon(\mathbf{x}^*) = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}^*\| < \varepsilon\}$$

*is called the **ε-neighborhood of $\mathbf{x}^*$** or the **ε-ball around $\mathbf{x}^*$**.*

*(e) If f is continuous (continuously differentiable), the minimization problem is called **continuous (continuously differentiable)**.*

Obviously, every global minimum is a local minimum, but not vice versa. Figure 2.1 shows the different types of problems. In general, there are two classes of optimization problems: *minimization* and *maximization*. Each kind can be easily transformed into the other because of

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) = -\max_{\mathbf{x} \in \mathbb{R}^n} \{-f(\mathbf{x})\}.$$

Thus, we focus on minimization problems in this chapter. Obviously, every method can also be applied to maximization problems.

[1] It would be mathematically more precise to write $\inf_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$ because the minimum does not have to exist but the infimum always exists, e.g. if $f \to -\infty$ we have $\inf_{\mathbf{x} \in \mathbb{R}^n} f = -\infty$. However, we follow the usual notation. Such unpleasant behavior should be checked before an optimization algorithm is applied in practice.

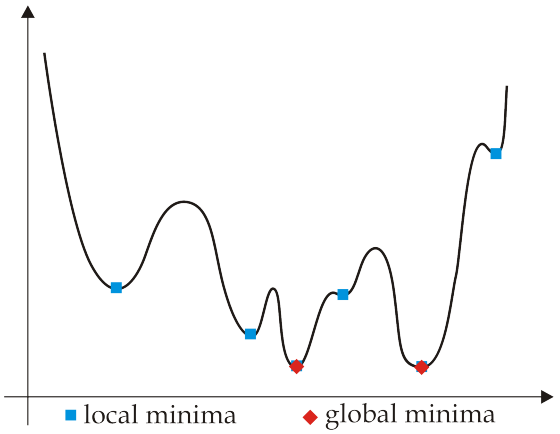

Figure 2.1: Local and global minima. Note that we cannot guarantee the marked global minima being indeed global on entire $\mathbb{R}$ because we do not know the function left and right of the shown part.

Definition 2.2.2. *Let $n \in \mathbb{N}$ and $f : \mathbb{R}^n \to \mathbb{R}$ be a function that maps from $\mathbb{R}^n$ into $\mathbb{R}$.*

(a) Let $\mathcal{F} \subset \mathbb{R}^n$, then the problem

$$\min_{\mathbf{x} \in \mathcal{F}} f(\mathbf{x}) \tag{2.2}$$

*is called a **constrained global minimization problem**.*

*(b) The region $\mathcal{F}$ is called the **feasible region** of the minimization problem.*

*(c) If $\mathcal{F}$ contains only a countable quantity of elements, problem (2.2) is called a **discrete (global) minimization problem**.*

Terms like local and global minimizer are analogously defined. If $\mathcal{F}$ is a connected region – especially if it is innumerable – and the minimizer lies in the interior of $\mathcal{F}$, then we can still treat (2.2) as an unconstrained minimization problem. This allows us to apply methods for unconstrained optimization to force fields in the next chapter, even though force field functions are undefined when the energy tends to ∞.

Example 2.2.3.

(a) The problem

$$\min_{(x,y) \in \mathbb{R}^2} x^2 - xy^3 + \sin(x^2 y)$$

is a continuously differentiable global minimization problem.

(b) The problem

$$\begin{aligned} \min_{(x,y) \in \mathbb{R}^2} \quad & x^2 - xy^3 + \sin(x^2 y) \\ \textit{subject to} \quad & x + y \leq 1, \\ & x - y \geq -1 \end{aligned}$$

is a constrained global minimization problem. The feasible region is given by

$$\mathcal{F} = \{(x, x) \in \mathbb{R}^2 \mid x + y \leq 1, x - y \geq -1\}.$$

(c) The problem

$$\begin{aligned} \min_{(x,y)\in\mathbb{R}^2} \quad & x^2 - xy^3 + \sin(x^2 y) \\ \textit{subject to} \quad & x + y \leq 1, \\ & x - y \geq -1, \\ & x, y \in \{0, 1\} \end{aligned}$$

is a discrete minimization problem (in this case a so-called 0-1-problem). The feasible region is given by

$$\mathcal{F} = \{(x, x) \in \{0, 1\}^2 \mid x + y \leq 1, x - y \geq -1\}.$$

The problem can also be denoted by

$$\begin{aligned} \min_{(x,y)\in\{0,1\}^2} \quad & x^2 - xy^3 + \sin(x^2 y) \\ \textit{subject to} \quad & x + y \leq 1, \\ & x - y \geq -1. \end{aligned}$$

2.3 Local Minimization of Smooth Functions

In general, if no special class of functions is given, the only way to find out whether a point $\mathbf{x}^*$ is a local minimum is to examine all the points in its immediate vicinity and to check that none of them has a lower function value. However, if the considered function f is *smooth*, we have more efficient and practical ways to identify local minima. If f is twice continuously differentiable, we may be able to identify a local minimizer by exploring its gradient ∇f and the Hessian $\nabla^2 f$.

2.3.1 Optimality Criteria

We only state here basic theorems for local optimization purposes. The proofs can be found in any calculus textbook, see, e.g., [133].

The fundamental and central tool to study local minimizers of smooth functions is Taylor's Theorem.

Theorem 2.3.1. (Taylor's Theorem)
Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and $\mathbf{p} \in \mathbb{R}^n$. Then it holds

$$f(\mathbf{x} + \mathbf{p}) = f(\mathbf{x}) + \nabla f(\mathbf{x} + \alpha\mathbf{p})^T \mathbf{p}, \quad \textit{for some } \alpha \in (0, 1).$$

Moreover, if f is twice continuously differentiable, we have that

$$\nabla f(\mathbf{x} + \mathbf{p}) = \nabla f(\mathbf{x}) + \int_0^1 \nabla^2 f(\mathbf{x} + t\mathbf{p})\mathbf{p}\, dt,$$

and that

$$f(\mathbf{x} + \mathbf{p}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \nabla^2 f(\mathbf{x} + \alpha\mathbf{p})\mathbf{p}, \quad \textit{for some } \alpha \in (0, 1).$$

Necessary conditions for local optimality can be directly derived from Taylor's Theorem.

Theorem 2.3.2. (First-Order Necessary Condition).
If $\mathbf{x}^*$ is a local minimizer of f and there exists $\varepsilon > 0$ with f is continuously differentiable in $\mathcal{B}_\varepsilon(\mathbf{x}^*)$, then

$$\nabla f(\mathbf{x}^*) = 0.$$

Note that in this case $\nabla f = 0$ is a necessary, but not sufficient condition, i.e. there may be points $\mathbf{x}$ with $\nabla f(\mathbf{x}) = 0$ but $\mathbf{x}$ is not a local minimizer of f.

Definition 2.3.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and $\mathbf{x} \in \mathbb{R}^n$ with*

$$\nabla f(\mathbf{x}) = 0,$$

*then we call $\mathbf{x}$ a **stationary point** (of f).*

Considering second derivatives, we obtain the following necessary condition.

Theorem 2.3.4. (Second-Order Necessary Condition).
If $\mathbf{x}^*$ is a local minimizer of f and there exists $\varepsilon > 0$ with f is twice continuously differentiable in $\mathcal{B}_\varepsilon(\mathbf{x}^*)$, then

$$\nabla f(\mathbf{x}^*) = 0 \quad \textit{and} \quad \nabla^2 f(\mathbf{x}^*) \succeq 0.$$

Taylor's Theorem can also be consulted to derive a sufficient condition.

Theorem 2.3.5. (Second-Order Sufficient Condition).
If $\mathbf{x}^*$ is a local minimizer of f and there exists $\varepsilon > 0$ with f is twice continuously differentiable in $\mathcal{B}_\varepsilon(\mathbf{x}^*)$ and it holds

$$\nabla f(\mathbf{x}^*) = 0 \quad \textit{and} \quad \nabla^2 f(\mathbf{x}^*) \succ 0,$$

*then $\mathbf{x}^*$ is a **strict** local minimizer of f.*

If the above sufficient condition holds, the corresponding minimizer $\mathbf{x}^*$ is called *strict*, i.e. there exists $\varepsilon > 0$ with

$$f(\mathbf{x}^*) < f(\mathbf{x}) \quad \text{for all } \mathbf{x} \in \mathcal{B}_\varepsilon(\mathbf{x}^*) \backslash \{\mathbf{x}^*\}.$$

Thus, this condition is stronger than the necessary conditions in a certain sense because it guarantees the uniqueness of the local minimizer in an ε-neighborhood. In addition, the second-order sufficient conditions are not necessary:

Example 2.3.6. *Consider $f(x) = x^4$. Certainly, $x^* = 0$ is a strict local (and global in this case) minimizer. However, $\nabla^2 f(0) = f''(0) = 0$ and, hence, the Hessian matrix is not positive definite, but semidefinite as required by Theorem 2.3.4.*

2.3.2 Overview of Algorithms

Numerical local minimization algorithms proceed in an iterative fashion by trying to find a new position $\mathbf{x}_{k+1}$ based on data at the current point $\mathbf{x}_k$ and, dependent on the kind of algorithm, further information collected during the optimization process. As mentioned in the previous chapter, the general approach to find local minima of smooth functions is to start at a given position and to "walk downhill" until one of the above criteria is fulfilled.

The start position of this process should be a reasonable estimate of the solution. In the case of reoptimization of molecules in Section 3.1, the given conformation is such an estimation. However, if no sensible information is given, we have to deal with the problem in a heuristic manner as in Section 3.2.

There are two fundamental strategies for moving from the current point $\mathbf{x}_k$ to a new iterate $\mathbf{x}_{k+1}$: trust region approaches and line search-based algorithms.

2.3.2.1 Line Search Methods

Line search-based algorithms calculate in each step a search direction $\mathbf{d}_k$ and start a line search looking for the next local minimizer along this direction. Let $f = f(\mathbf{x})$, f continuously differentiable, be a scalar function of the vector $\mathbf{x}$ and $\mathbf{x}_k$ the current iterate at iteration k. Then, a typical iteration of a line search-based minimization algorithm can be described as follows:

1: Compute a search direction $\mathbf{d}_k$ from current and/or collected data ($\mathbf{d}_k$ needs to be a descent direction)

2: Compute the minimum of f from $\mathbf{x}_k$ along $\mathbf{d}_k$, i.e.

$$\lambda_k = \arg\min_{\lambda>0} f(\mathbf{x}_k + \lambda \mathbf{d}_k)$$

3: Set

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k.$$

The second step is called an *exact line search*. Since the computation of the exact minimum is expensive and unnecessary, inexact line searches are used in practice. Thus, the second and third step are replaced by

2': Compute $\lambda_k > 0$ that yields an acceptable next iterate $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k$.

Widely used conditions for accepting an iterate [123] are the so-called *strong Wolfe conditions*

$$f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_k) + \alpha \lambda_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k \tag{2.3}$$

$$|\nabla f(\mathbf{x}_{k+1})^T \mathbf{d}_k| \leq \beta |\nabla f(\mathbf{x}_k)^T \mathbf{d}_k| \tag{2.4}$$

for accepting λ_k, where $0 < \alpha < \frac{1}{2}$ and $\alpha < \beta < 1$. The first inequality ensures to be slightly better than simply $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$, which is crucial for the convergence theory behind line search algorithms, see Dennis and Schnabel [84]. The second inequality is a curvature condition [123]. This condition is important in quasi-Newton methods since it guarantees that a positive definite quasi-Newton update is possible allowing the incorporation of current local information into the approximated Hessian. See for example Dennis and Schnabel [84] and Fletcher [49], see also Section 2.3.2.3. It is always possible to fulfill the second criterion if the objective function is bounded from below and does not tend to $-\infty$. For more information the reader is referred to Dennis and Schnabel [84] and Moré and Thuente [123].

The strong Wolfe conditions have the advantage that by decreasing β we can directly control the quality of the search by forcing the accepted $\mathbf{x}_{k+1}$ to lie closer to a local minimum along $\mathbf{d}_k$. This tuning is not possible using the *regular (or weak) Wolfe conditions*, which only require

$$\nabla f(\mathbf{x}_{k+1})^T \mathbf{d}_k \geq \beta \nabla f(\mathbf{x}_k)^T \mathbf{d}_k,$$

instead of (2.4). Therefore, a step selection routine that enforces the strong Wolfe conditions is of wider applicability.

Standard line search algorithms use a two stage strategy. In the first stage, a local minimizer is bracketed (extrapolation stage). In the second stage, the bracketing interval is iteratively decreased until a suitable position can be found (interpolation stage). The next trial step in each stage is estimated by interpolating given data and setting the trial step to the minimizer of the interpolant. The common standard line search algorithms use quadratic and cubic polynomial interpolation schemes discussed later. This is due to the simplicity of the calculations and uniqueness of the interpolants. Note that the choice between possibilities only serves to guarantee interpolation and extrapolation criteria. It is, however, usually not utilized to adapt optimally to the behavior of the objective function in the region of interest. If both, quadratic and cubic steps, fulfill the required interpolation and extrapolation criteria, the cubic step is often taken since it additionally fulfills all derivative conditions. We give in Section 3.1 details on our consensus line search method, which follows a novel approach to adapt interpolants to the behavior of the objective function. We show that it is optimally adapted to molecular potential energy functions and has the power to dramatically reduce computational costs.

2.3.2.2 Trust-Region Methods

In this work, we focus on local optimization techniques in a molecular context and, to the best of our knowledge, all current molecular minimization techniques use line search-based approaches. The main reason for this is computational cost: large problem instances as they occur in molecular structure optimization require the use of limited memory schemes that are entirely unsuited for efficient trust region computation. In this case, trust region methods also sometimes fail to incorporate local information about the objective function, potentially resulting in degraded convergence behavior of the algorithm. Thus, we do not discuss trust-region methods in this work. However, we sketch the basic ideas for completeness.

Such methods use information gathered about the objective function f to construct a model function m_k whose behavior near the current point $\mathbf{x}_k$ is similar to that of f. This model m_k may not be a good approximation of f if a position $\mathbf{x}$ is far from $\mathbf{x}_k$. Thus, the search for a minimizer of m_k is restricted to some region around $\mathbf{x}_k$. The advantage is clearly that a trust region approach makes further use of this n-dimensional model while a line search only constructs one-dimensional models into the previously determined search direction. The candidate step $\mathbf{p}$ is then found by approximately solving

$$\begin{aligned} \min_{\mathbf{p}\in\mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{p}) \\ \text{subject to} \quad & \|\mathbf{p}\| \leq \delta, \end{aligned}$$

where $\delta > 0$ is the so-called trust region radius. If the candidate solution does not produce a sufficient decrease in f, the usual approach is to conclude that the trust region is too large. In this case, the region will be shrunk and the corresponding problem has to be (re)solved. If the model m_k perfectly fits the objective function f at the candidate solution, the region is increased. For comprehensive treatment on trust-region methods, the reader may be referred to Conn et al. [29].

2.3.2.3 Search Directions for Line Search Methods

In this section, we sketch different approaches to calculate the search direction $\mathbf{d}_k$ in (2.3) and (2.4). We use these (or derived) schemes in Section 3.1 for the optimization of molecular structures.

Steepest Descent. Mathematically, a direction $\mathbf{d}$ is a descent direction from $\mathbf{x}_k$ if the directional derivative of f at $\mathbf{x}_k$ in direction $\mathbf{d}$ is negative, i.e.

$$\nabla f(\mathbf{x}_k)^T \mathbf{d} < 0.$$

The most obvious search direction choice for a line search method is then the direction of steepest descent. It is the direction along which the objective function f decreases most rapidly for a given norm, i.e.

$$\begin{aligned} \min_{\mathbf{d}\in\mathbb{R}^n} \quad & \nabla f(\mathbf{x}_k)^T \mathbf{d} \\ \text{subject to} \quad & \|\mathbf{d}\| = 1. \end{aligned}$$

In l_2 norm, this problem has the unique solution

$$\mathbf{d} = -\frac{1}{\|\nabla f(\mathbf{x}_k)\|} \nabla f(\mathbf{x}_k),$$

see for example Nocedal and Wright [133]. However, the convergence of a steepest descent approach in practice is only linear and should be avoided in practice [84]. This approach tends to circle round a local minimum and is, thus, excruciatingly slow on difficult problems. However, the methods in the following sections use the steepest descent direction in the first iteration when the whole optimization procedure starts. The advantage is clearly the low computational costs and storage requirements are linear in the number of variables.

Newton's Method. Another important search direction – perhaps the most important one of all – is the Newton direction. Using Taylor's Theorem 2.3.1 yields

$$f(\mathbf{x}_k + \mathbf{d}) \approx f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T \nabla^2 f(\mathbf{x_k}) \mathbf{d} =: m_k(\mathbf{d}).$$

m_k is a quadratic model of f around $\mathbf{x}_k$. The *Newton direction* [84] is the minimizer of m_k. If $\nabla^2 f(\mathbf{x}_k)$ is positive definite, m_k has a unique minimizer and the Newton direction is given by

$$\mathbf{d} = -\nabla^2 f(\mathbf{x}_k)^{-1} \nabla f(\mathbf{x}_k), \tag{2.5}$$

which is a descent direction, since due to the positive definiteness of $\nabla^2 f(\mathbf{x}_k)$ it follows

$$-\nabla f(\mathbf{x}_k)^T \nabla^2 f(\mathbf{x}_k)^{-1} \nabla f(\mathbf{x}_k) < 0.$$

However, if m_k is not convex, direction (2.5) does not have to be a descent direction. To overcome this problem either $\nabla^2 f(\mathbf{x}_k) + \mu \mathbf{I}$, $\mu > 0$, is used in practice, where $\mu \mathbf{I}$ is a small perturbation in a way that $\nabla^2 f(\mathbf{x}_k) + \mu \mathbf{I} \succ 0$ [84], or trust-region methods, see Section 2.3.2.2, are applied to the full (non-convex) model.

The disadvantages of Newton's method are twofold: (1) the second derivatives have to be computed, which may be expensive, and (2) they have to be stored, i.e. the storage requirements are quadratic in the number of variables. Thus, methods have been developed to overcome these problems.

Quasi-Newton Procedures. An attempt to avoid second derivatives (but still with quadratic storage requirements) are the widely used *quasi-Newton methods*. They form the basis of limited memory methods we used in our study in Section 3.1. The model is still quadratic, but an approximation $\mathbf{B}_k$ to the Hessian $\nabla^2 f(\mathbf{x}_k)$ is used instead of the exact second derivatives. The matrix $\mathbf{B}_k$ is updated in every iteration of the local minimization procedure based on the observation [133] that

$$\nabla^2 f(\mathbf{x}_{k+1})(\mathbf{x}_{k+1} - \mathbf{x}_k) \approx \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k),$$

when $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ lie in a region near to a local minimum where $\nabla^2 f$ is positive definite. Many approaches for unconstrained minimization update $\mathbf{B}_k$ that (1) $\mathbf{B}_{k+1}$ is symmetric, (2) it mimics this property, i.e.

$$\mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k \tag{2.6}$$

(the so-called *secant equation*), where

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \qquad \mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k),$$

and (3) that

$$\mathbf{B}_{k+1} \succ 0.$$

It is always possible [84] to update $\mathbf{B}_k$ in a way that $\mathbf{B}_{k+1}$ is positive definite if $\mathbf{B}_k$ itself is positive definite and it holds

$$\mathbf{s}_k^T \mathbf{y}_k > 0. \tag{2.7}$$

If condition (2.4) is fulfilled in a line search framework, see Section 2.3.2.1, the above condition (2.7) is also fulfilled. Thus, in a line search context an update of $\mathbf{B}_k$ is always possible, whereas this might fail in a trust-region approach. One of the most popular update schemes is

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{\mathbf{B}_k \mathbf{s}_k \mathbf{s}_k^T \mathbf{B}_k}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k} + \frac{\mathbf{y}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k},$$

the so-called *BFGS formula*, named after its inventors, Broyden, Fletcher, Goldfarb, and Shanno. Typically, $\mathbf{B}_0$ is the identity matrix or any scaled version of it, see Liu and Nocedal [110], and the search direction in the first iteration becomes the steepest descent direction.

A more modern approach are so-called *shifted variable metric methods* [180], which chose

$$\mathbf{B}_k^{-1} = \zeta_k \mathbf{I} + \mathbf{A}_k,$$

where $\mathbf{A}_k$ is a symmetric positive semidefinite matrix and $\zeta_k > 0$ is the so-called *shift parameter*. Vlček and Lukšan [180] give two update schemes for $\mathbf{A}$ where the scheme based on our calculations in Section 3.1 is given by

$$\mathbf{A}_{k+1} = \mathbf{A}_k + \left(\rho_k + \frac{\mathbf{y}_k^T \mathbf{A}_k \mathbf{y}_k}{\mathbf{p}_k^T \mathbf{y}_k}\right) \frac{\mathbf{p}_k \mathbf{p}_k^T}{\mathbf{p}_k^T \mathbf{y}_k} - \frac{\mathbf{p}_k \mathbf{y}_k^T \mathbf{A}_k + \mathbf{A}_k \mathbf{y}_k \mathbf{p}_k^T}{\mathbf{p}_k^T \mathbf{y}_k},$$

with $\mathbf{p}_k = \mathbf{s}_k - \zeta_{k+1}\mathbf{y}_k$ and $\rho_k > 0$ is a correction parameter. Vlček and Lukšan [180] prove that it is advantageous if the shift parameter satisfies

$$0 < \zeta_{k+1} < \frac{\mathbf{s}_k^T \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{y}_k}$$

and give several update schemes for ζ_k and ρ_k. In our study in Section 3.1, we used the recommended update

$$\rho_k = \frac{\zeta_k}{\zeta_k + \zeta_{k+1}} \quad \text{and} \quad \zeta_{k+1} = \frac{\sqrt{1 - \frac{\mathbf{y}_k^T \mathbf{A}_k \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{B}_k^{-1} \mathbf{y}_k}}}{1 + \sqrt{1 - \frac{\left(\mathbf{s}_k^T \mathbf{y}_k\right)^2}{\|\mathbf{y}_k\|^2 \|\mathbf{s}_k\|^2}}} \frac{\mathbf{y}_k^T \mathbf{s}_k}{\|\mathbf{y}_k\|}.$$

Shanno's Conjugate Gradient Algorithm. Probably the oldest approach to avoid second derivatives and the quadratic storage requirements are conjugate gradient algorithms [50]. These methods are based on the concept of conjugate directions and the assumption of a quadratic objective function. If the objective function is quadratic, then a conjugate gradient approach finds (in theory) the minimum in n steps, where n is the number of variables. However, in practice we have to deal with roundoff errors of the floating point arithmetic and the objective functions are not quadratic. Nevertheless, conjugate gradient methods are commonly used in many applications. An interesting extension has been developed by Watowich et al. [187] and is called *Shanno's Conjugate Gradient Algorithm*. The basic idea is to incorporate ideas for update schemes from quasi-Newton methods into a conjugate gradient approach. The search direction is then calculated as follows:

$$\mathbf{d}_{k+1} = -\mathbf{b}_k + \frac{\mathbf{s}_t^T \nabla f(\mathbf{x}_{k+1})}{\mathbf{s}_k^T \mathbf{y}_k} \mathbf{a}_k - \left(\left(1 + \frac{\mathbf{a}_k^T \mathbf{y}_k}{\mathbf{s}_k^T \mathbf{y}_k} \right) \frac{\mathbf{s}_t^T \nabla f(\mathbf{x}_{k+1})}{\mathbf{s}_k^T \mathbf{y}_k} - \frac{\mathbf{a}_k^T \nabla f(\mathbf{x}_{k+1})}{\mathbf{s}_k^T \mathbf{y}_k} \right) \mathbf{s}_k,$$

where

$$\mathbf{a}_k = \frac{\mathbf{s}_t^T \mathbf{y}_t}{\mathbf{y}_t^T \mathbf{y}_t} \mathbf{y}_k - \frac{\mathbf{s}_t^T \mathbf{y}_k}{\mathbf{y}_t^T \mathbf{y}_t} \mathbf{y}_t + \left(\frac{2\mathbf{s}_t^T \mathbf{y}_k}{\mathbf{s}_t^T \mathbf{y}_t} - \frac{\mathbf{y}_t^T \mathbf{y}_k}{\mathbf{y}_t^T \mathbf{y}_t} \right) \mathbf{s}_t, \tag{2.8}$$

$$\mathbf{b}_k = \frac{\mathbf{s}_t^T \mathbf{y}_t}{\mathbf{y}_t^T \mathbf{y}_t} \nabla f(\mathbf{x}_{k+1}) - \frac{\mathbf{s}_t^T \nabla f(\mathbf{x}_{k+1})}{\mathbf{y}_t^T \mathbf{y}_t} \mathbf{y}_t + \left(\frac{2\mathbf{s}_t^T \nabla f(\mathbf{x}_{k+1})}{\mathbf{s}_t^T \mathbf{y}_t} - \frac{\mathbf{y}_t^T \nabla f(\mathbf{x}_{k+1})}{\mathbf{y}_t^T \mathbf{y}_t} \right) \mathbf{s}_t. \tag{2.9}$$

After a certain number of steps a restart is performed. Here, t is the index of the iteration of the most recent restart. Since this interesting approach is known to perform very well on molecular potential energy functions [187], we also included this approach in our set of tested methods.

Limited-Memory Quasi-Newton Methods. The most modern approach are limited-memory quasi-Newton methods. They are based on the aforementioned update schemes. In the case of the BFGS method the matrix updates are stored separately (and not directly applied to $\mathbf{B}_k$ in each iteration), and when the available storage is used up, the oldest correction is deleted to make space for the new one. To be exact, the pairs $(\mathbf{s}_k, \mathbf{y}_k)$ are stored (they are the only required data for an update) until a fixed amount of storage would be exceeded and in all subsequent iterations the oldest pair is deleted and the new one is inserted [132, 110]. Due to the Strang recurrences [116] the solution $\mathbf{d}$ of

$$\mathbf{B}_k \mathbf{d} = -\nabla f(\mathbf{x}_k), \quad \text{i.e.} \quad \mathbf{d} = -\mathbf{B}_k^{-1} \nabla f(\mathbf{x}_k)$$

can be calculated only based on the stored vector pairs and without explicit knowledge of $\mathbf{B}_k$. Let us assume we have stored the recent m correction pairs, which we label for simplicity $(\mathbf{s}_0, \mathbf{y}_0), \ldots, (\mathbf{s}_{m-1}, \mathbf{y}_{m-1})$ and $\rho_i = \frac{1}{\mathbf{y}_i^T \mathbf{s}_i}$, then the following algorithm calculates $\mathbf{d}$ at the current position $\mathbf{x}_k$:

1. $\mathbf{d} \leftarrow \nabla f(\mathbf{x}_k)$
2. For $i = m-1, \ldots, 0$:
   $\alpha_i \leftarrow \rho_i \mathbf{s}_i^T \mathbf{d}$ (store α_i)
   $\mathbf{d} \leftarrow \mathbf{d} - \alpha_i \mathbf{y}_i$
3. $\mathbf{d} \leftarrow \mathbf{B}_0^{-1}\mathbf{d}$
4. For $i = 0, \ldots, m-1$:
   $\beta_i \leftarrow \rho_i \mathbf{y}_i^T \mathbf{d}$
   $\mathbf{d} \leftarrow \mathbf{d} + \mathbf{s}_i(\alpha_i - \beta_i)$

A very good choice for $\mathbf{B}_0^{-1}$ is the scaled identity matrix

$$\mathbf{B}_0^{-1} = \frac{\mathbf{y}_k^T \mathbf{s}_k}{\|\mathbf{y}_k\|^2}\mathbf{I},$$

see Liu and Nocedal [110]. Note that the computation of $\mathbf{d}$ in this case needs only $2m(2n+1)+3n$ multiplications and one division, and is, hence, very efficient.

Limited memory methods for the shifted variable metric method can be derived by the update scheme given in the previous section about quasi-Newton methods and the choice

$$\mathbf{B}_k^{-1} = \zeta_k \mathbf{I} + \mathbf{U}_k \mathbf{U}_k^T,$$

where the $n \times m$ matrix $\mathbf{U}_k$ is updated. Thus, the limited memory approximation $\mathbf{U}_k\mathbf{U}_k^T$ replaces $\mathbf{A}_k$. With our previous notation the resulting update scheme is given by

$$\begin{aligned}\mathbf{U}_{k+1} = \; & \mathbf{U}_k - \frac{\mathbf{p}_k \mathbf{y}_k^T \mathbf{U}_k}{\mathbf{y}_k^T \mathbf{p}_k} \\ & - \left(\rho_k \frac{\mathbf{p}_k}{\sqrt{\rho_k \frac{\mathbf{y}_k^T \mathbf{p}_k}{\|\lambda_k \mathbf{U}_k^T \nabla f(\mathbf{x}_k)\|^2}}} + \lambda_k \mathbf{U}_k \mathbf{U}_k^T \nabla f(\mathbf{x}_k) - \lambda_k \frac{\mathbf{y}_k^T \mathbf{U}_k \mathbf{U}_k^T \nabla f(\mathbf{x}_k)}{\mathbf{y}_k^T \mathbf{p}_k} \right) \\ & \cdot \frac{\mathbf{U}_k^T \nabla f(\mathbf{x}_k)}{\lambda_k \|\mathbf{U}_k^T \nabla f(\mathbf{x}_k)\|^2},\end{aligned}$$

where λ_k is the solution (2.3), (2.4) of the most recent line search.

2.4 Local Minimization of Non-Smooth Functions

In Section 3.2, we consider a non-smooth scoring function. Thus, we have to deal with that topic although the focus of this work is more on smooth functions, i.e., the functions themselves are at least continuously differentiable[2].

Functions of the non-smooth type may be non-smooth but continuous, see Figure 2.2, or even discontinuous. It is not possible in general to identify a minimizer of a general discontinuous function. If the function consists of smooth pieces (with discontinuities between them), it may be possible to minimize each piece individually. If the function is continuous everywhere but non-differentiable at certain points, we can

[2]Note that even in the case of the discrete optimization problems we consider in Chapter 4, the functions themselves are smooth. The problems are continuous (and constrained) if we drop the integrality constraints.

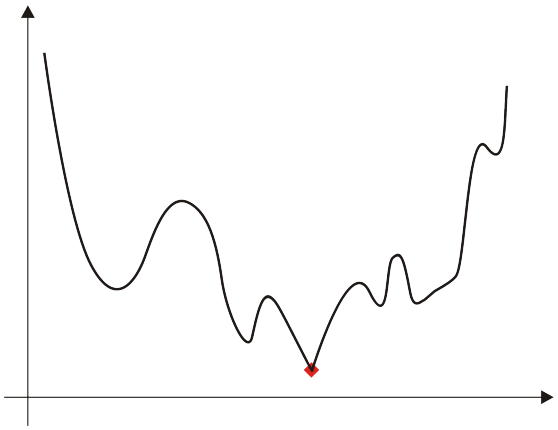

Figure 2.2: Non-smooth function with minimum at a kink.

(a) identify the solution by exploring *subgradients*, or *generalized gradients*[3].

(b) force the function to be continuously differentiable by inserting transition patches between the smooth pieces, see Figure 2.3, if this is applicable, i.e. the non-differentiable positions are known before optimization and can be efficiently caught during the optimization process. We choose this approach in Section 3.2 in order to apply the full machinery of smooth optimization techniques while we do not substantially change the results.

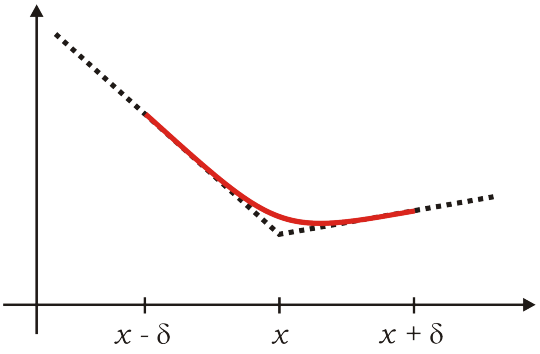

Figure 2.3: Added smooth transition patch between two differentiable pieces. The dashed line is the original function, which has been modified (red solid line) in a δ-neighborhood of the non-smooth point x.

For more information about non-smooth optimization, the reader may be referred to Hiriart-Urruty and Lemaréchal [77]. Note that if the function to be minimized does not have any special property, but there are simple bound constraints on the variables, the approach described in the next section may be worth being tried out finding global minimizers. However, this approach may fail in practice due to too many "intermingled" terms in the objective function resulting in a too high number of boxes to be considered.

[3]Subgradients are a generalization of the concept of gradients to the non-smooth case.

2.5 Global Optimization in General

As mentioned before, if the objective function does not have any special property, there is no known method to determine global minima. We have to distinguish between the task of finding positions as deep as possible (hopefully they are global minima) and proving that positions we could identify are indeed global minima (or if they are not global the task is to find (other) deeper positions).

The first task can be addressed with heuristics, e.g., monte-carlo based search, simulated annealing, genetic algorithms, particle swarm, differential evolution, etc.. All of them profit from efficient local optimization techniques as subroutines if the considered problems are continuously differential. However, although we incorporate our techniques in heuristics in Chapter 3 in the context of molecular docking, a survey of heuristics is beyond the scope of this work. For a good overview of this topic the reader may be referred to Fuhrmann [53], a detailed description can be found in Michalewicz and Fogel [117].

The proof of global optimality (and to identify deeper positions if the proof fails) can only succeed in special cases. In the next section, we consider a special class of problems that allow for efficient detection of global minima and the proof of global optimality.

The most general problems that can be solved, at least in theory, are problems without special structure but with so-called "box-constraints" on every variable, i.e., there is a restriction $x \in [a, b]$, $a \leq b$ on every parameter x.[4] The most general approaches for this kind of problems are branch-and-bound algorithms using interval arithmetic [114]. It is interesting to note that by interval arithmetic the global optimization problem can be seen as "solved" when the objective function is twice continuously differentiable and only one parameter x with $x \in [a, b]$, $a \leq b$ is considered [193, 70, 69]. In other words, there exists an algorithm for one-dimensional twice continuously differentiable problems under box-constraints, which is successfull not only in theory but also in practice, see Hansen [70]. However, this does not hold if more than one variable is involved or any other prerequisite is not fulfilled.

A typical branch-and-bound algorithm is given in Figure 2.4. The main prerequisite for such an approach (as is for optimization in general) is that

$$\min_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x}) \geq \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad \text{if } \mathcal{S} \subseteq \mathcal{X}$$

holds. Certainly, this fact (with exact minima) does hold in general but any used estimation approach has also to fulfill this fundamental principle, i.e. let $\operatorname{est}\min$ be the estimated minimum (an estimated lower bound) then the method to obtain this bound has to fulfill

$$\operatorname{est}\min_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x}) \geq \operatorname{est}\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad \text{if } \mathcal{S} \subseteq \mathcal{X}. \tag{2.10}$$

Obviously, since every region is reduced by splitting and the bounds obtained become tighter the branch-and-bound algorithm identifies global minima (if these minima are strict it returns regions with only one element) and proves that they are global by discarding provably non-optimum regions. This approach has to deal with many problems in practice. If there are non-discrete global optima, this algorithm will never terminate because it tries to enumerate an innumerable number of solutions. Even if there are only a finite number of solutions,

[4] In practice, numbers are represented by floating point numbers in a computer, which have only a limited range. Thus, this restriction is more of theoretical interest. It may be sufficient to set, e.g., $a = -10^{10}$ and $b = 10^{10}$.

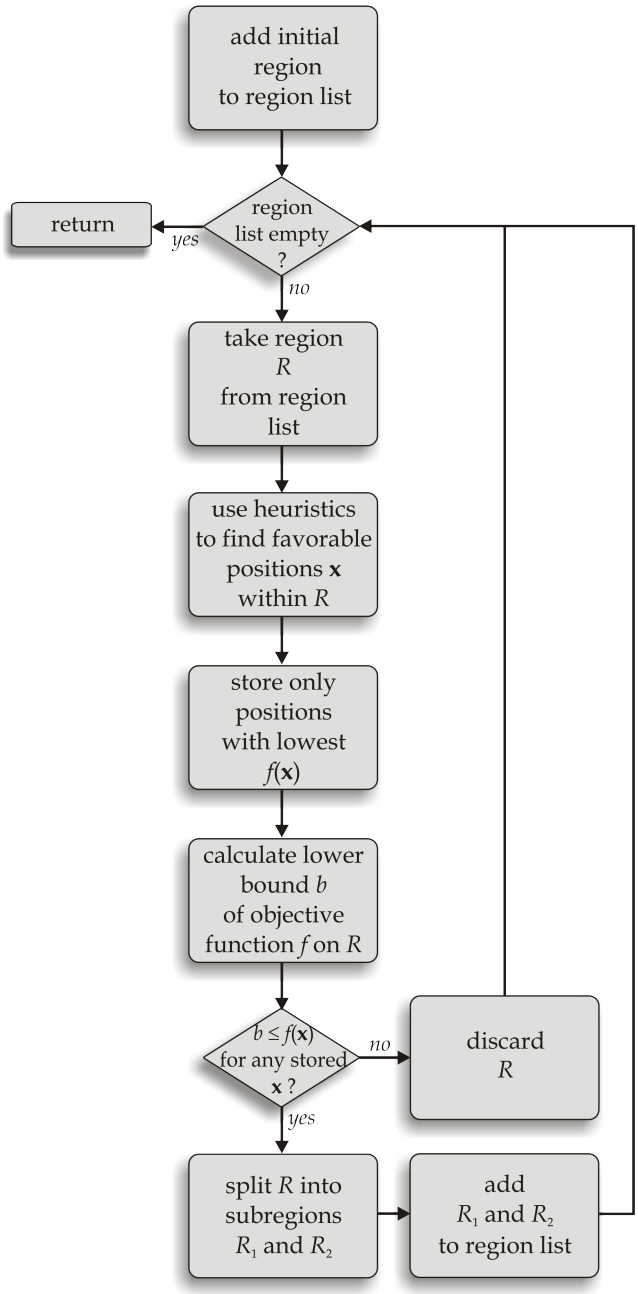

Figure 2.4: Typical branch-and-bound algorithm using heuristics to improve a pure branch-and-bound procedure.

the number of regions to be considered during the branch-process might be too many to be stored. Nevertheless, this approach is in practical use and the most general way to look for global minimizer.

The main problem is to calculate a lower bound on the objective function within a given region (fulfilling 2.10). The most general approach is interval analysis and the subsequent interval arithmetic [120]. We sketch the basic ideas in the following section. In Section 3.5, we show how to find all solutions that models a backbone loop of a protein based on this technique. It should be mentioned here that another related method is affine arithmetic [37], which is often viewed as an improvement of interval arithmetic. However, the errors of affine arithmetic are quadratic, whereas interval arithmetic errors are linear [165]. Thus, there is usually a critical width for the input intervals beyond which affine arithmetic is not accurate enough to be worth its added expense. We restrict ourself to interval arithmetic in this work.

2.5.1 Interval Arithmetic

R. E. Moore [120] derived an interval extension of Newton's method and showed that in a neighborhood of a simple root this extension converges quadratically to the root. Nickel [130, 131] showed that the algorithm converges globally provided the derivative of the objective function is non-zero in some interval containing the root. Hansen [68] extended the interval analysis and developed a method that isolates and bounds all real roots of a continuously differentiable function in a given interval. This method never fails to converge and led to the algorithm for identifying all global minima of a function with one variable when the first and the second derivatives have a finite number of isolated zeros [70]. In this section we give the basic ideas behind interval arithmetic.

Classical arithmetic defines operations on individual numbers. Interval arithmetic defines these operations on intervals in the sense that if $\mathrm{op}(I) = [a, b]$ for an interval I and an operator op, it is guaranteed that it holds

$$a \leq \min\{\mathrm{op}(x) \mid x \in I\} \quad \text{and} \quad b \geq \max\{\mathrm{op}(x) \mid x \in I\},$$

see Figure 2.5. In this context, the basic operations on intervals $[a, b], [c, d]$ are given by

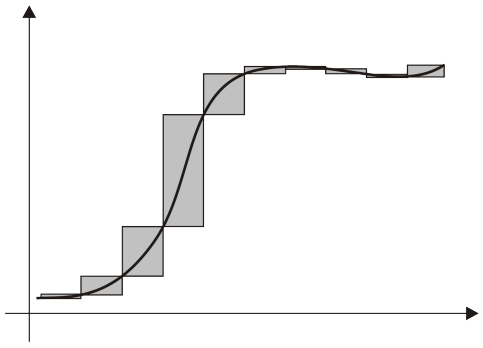

Figure 2.5: Example of interval arithmetic. The given function with the tightest lower and upper bounds on the intervals (gray boxes).

- $[a, b] + [c, d] = [a + c, b + d]$,
- $[a, b] - [c, d] = [a - d, b - c]$,

- $[a,b] \cdot [c,d] = [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}]$,
- $\frac{[a,b]}{[c,d]} = \left[\min\left\{\frac{a}{c}, \frac{a}{d}, \frac{b}{c}, \frac{b}{d}\right\}, \max\left\{\frac{a}{c}, \frac{a}{d}, \frac{b}{c}, \frac{b}{d}\right\}\right]$, if $0 \notin [c,d]$.

Operations with scalars are defined by intervals with identical upper and lower bounds, e.g. we have

$$\alpha + [a,b] = [\alpha,\alpha] + [a,b] = [\alpha + a, \alpha + b],$$

for a scalar $\alpha \in \mathbb{R}$. More complex functions can be evaluated by the chain rule, e.g. $x^2 - 2 \cdot x^3$ can be evaluated by $(x \cdot x) - (2 \cdot ((x \cdot x) \cdot x))$. Thus, the resulting interval is based on an accumulation and evaluation of basic operations. Considering the case where 0 is contained in the divisor interval leads to extended interval arithmetic [68]. Theoretically, all primitive functions like sin, cos, exp, etc. can be extended to their interval equivalents purely by performing the numerical algorithms that calculate their function values. However, the bounds obtained this way are usually too inaccurate. For example, consider the square function. We have

$$[-1,2]^2 = [-1,2] \cdot [-1,2] = [-2,4],$$

where a rough estimation would lead to $[0,4]$. Thus, specialized versions for interval arithmetic have been developed for all standard functions, which provide tighter bounds.

Using this arithmetic, branch-and-bound methods for either global optimization or solving systems of equalities may be applied. However, due to the above mentioned problems in practice we cannot assume that we obtain the desired results by applying such an approach. "Intermingled" terms in the objective function are the main problem. If the terms are linear, the bounds obtained are exact. Highly non-linear terms lead usually to very inaccurate bounds (although these bounds are guaranteed lower and upper bounds) resulting in (too) many splitting steps in a branch-and-bound framework. Imagine that if we have n parameters and we want every parameter only to be splitted once, we have to consider 2^n boxes, see Figure 2.6. For example, if our problem has only the modest number of 20 variables, we have to

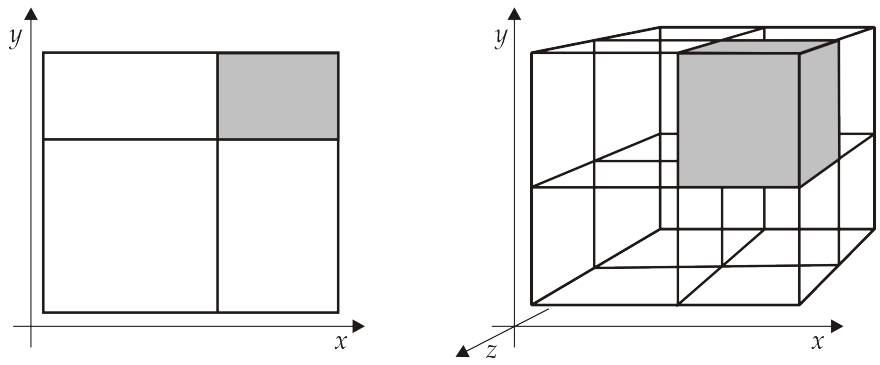

Figure 2.6: Splitting of boxes. On the left two parameters each splitted once: $2^2 = 4$ boxes. On the right three parameters each splitted once: $2^3 = 8$ boxes.

consider $2^{20} = 1.048.576$ boxes when every variable is splitted once. Certainly, there are approaches to improve a primitive branch-and-bound approach, e.g. see [194, 195], but we cannot expect them to be successful in general. Nevertheless, finding all solutions that models a backbone loop of a protein is a task where interval arithmetic can be successfully applied, see Section 3.5.

2.6 Convex Optimization

The concept of convexity is fundamental in optimization. It implies that the problem is benign in several respects. The term *convex* can be applied both to sets and to functions.

Definition 2.6.1. *(a) A set $\mathcal{S} \subset \mathbb{R}^n$ is called **convex** if for any two points $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{S}$ we have*

$$\alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2 \in \mathcal{S} \quad \text{for all } \alpha \in [0,1].$$

*(b) Let $f : \mathcal{S} \to \mathbb{R}$ be a function with domain $\mathcal{S} \subseteq \mathbb{R}^n$. f is called **convex** if $\mathcal{S}$ is convex and for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{S}$ it holds*

$$f(\alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1-\alpha)f(\mathbf{x}_2), \quad \text{for all } \alpha \in [0,1].$$

*(c) A function f is called **concave** if $(-f)$ is convex.*

Informally speaking, a set $\mathcal{S}$ is convex if the straight line segment connecting any two points in $\mathcal{S}$ lies itself fully inside $\mathcal{S}$. In other words, convex sets cannot have hollows and dips, see Figure 2.7. Sets not fulfilling this condition are usually called *non-convex*, see Figure 2.8. An analogous imagination of a convex function f is that f is convex if for any two points $\mathbf{x}_1$

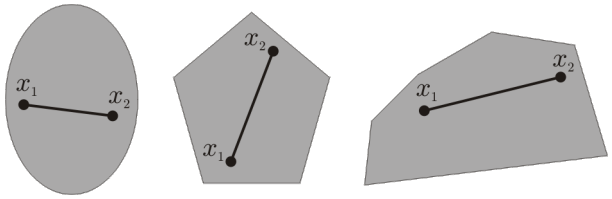

Figure 2.7: Convex sets.

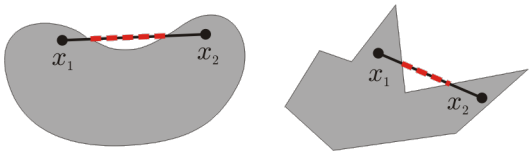

Figure 2.8: Non-convex sets.

and $\mathbf{x}_2$ within its domain the graph of f lies below the straight line connecting $(\mathbf{x}_1, f(\mathbf{x}_1))$ to $(\mathbf{x}_2, f(\mathbf{x}_2))$ in the space $\mathbb{R}^{n+1}$, see Figure 2.9. Otherwise, the function is called *non-convex*, see Figure 2.10.

Definition 2.6.2. *Let $f : \mathcal{F} \to \mathbb{R}$ be a convex function and $\mathcal{F}$ a convex set, then the problem*

$$\min_{\mathbf{x} \in \mathcal{F}} f(\mathbf{x}) \tag{2.11}$$

*is called a **convex (constrained global) minimization problem** or **convex program**.*

By the fact that the region

$$\mathcal{F} = \{\mathbf{x} \in \mathbb{R}^n \mid f_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, m\}$$

is convex if $f_i : \mathbb{R}^n \to \mathbb{R}$ are convex functions, we have the more common representation:

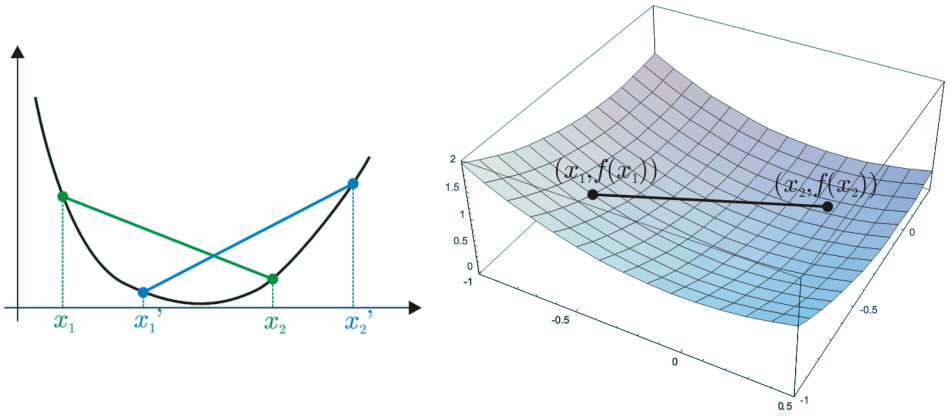

Figure 2.9: Convex functions.

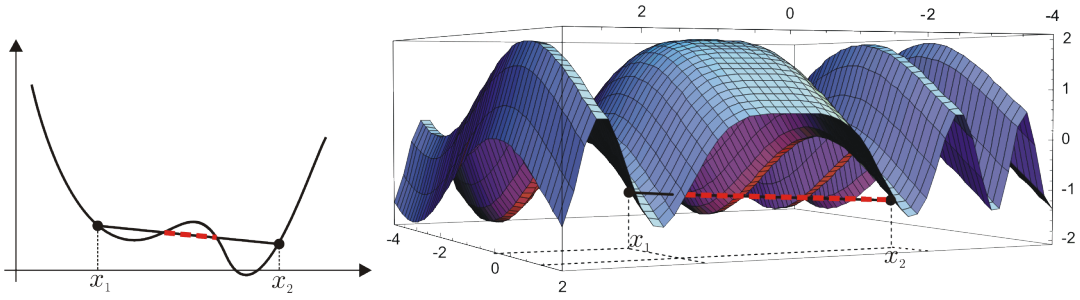

Figure 2.10: Non-convex functions.

Corollary 2.6.3. *Let $f_i : \mathbb{R}^n \to \mathbb{R}$ be convex functions, $i = 0, \ldots, m$, then the problem*

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & f_0(\mathbf{x}) \\ \textit{subject to} \quad & f_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, m \end{aligned} \tag{2.12}$$

is a convex minimization problem (convex program).

The importance of convex optimization is given in the following theorem.

Theorem 2.6.4. *Let $f : \mathcal{S} \to \mathbb{R}$ be a convex function. Then, any local minimizer $\mathbf{x}^*$ of f is a global minimizer of f. If, in addition, f is differentiable, then any stationary point $\mathbf{x}^*$ is a global minimizer of f.*

Thus, looking for local minimizer – in fact, a local stationary point in the unconstrained case – is sufficient to identify global minimizer in the case of convex optimization. Thus, the simple strategy to start from somewhere (within the feasible region) and to walk downhill (within the feasible region) until no further improvement can be achieved will lead to a global minimum[5].

In general, any constrained convex program can be solved by (local) methods for constrained optimization problems. Usual approaches are augmented Lagrangian methods, see [86] for a good survey, and filter approaches, see for example [181]. More efficient techniques

[5]In practice, numerical instabilities might lead to incorrect results.

are so-called *interior-point methods*. These iterative algorithms converge to the solution out of the interior of the feasible region and make use of the convexity of the problem [86]. However, general constrained convex programs are beyond the scope of this work. In the subsequent sections, we describe special classes of constrained convex problems that will be applied in the following chapters.

2.6.1 Linear Programs

If all functions in (2.12) are linear, the resulting problem is called a linear program. In explicit notation we have

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & c_1x_1 \quad + ... + c_nx_n \\ \text{s. t.} \quad & a_{11}x_1 \quad + ... + a_{1n}x_n \leq b_1, \\ & \vdots \qquad\qquad\quad \vdots \\ & a_{m1}x_1 + ... + a_{mn}x_n \leq b_m, \end{aligned}$$

with $c_i, a_{j,i} \in \mathbb{R}$. Usually, linear programs are represented by matrix/vector notation:

Definition 2.6.5. (Linear Program)
Let $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, *and* $\mathbf{A} \in \mathbb{R}^{n\times m}$, *then the problem*

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{c}^T\mathbf{x} \\ s.\ t. \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{aligned}$$

is called ***linear program (LP)****.*

There are several methods to solve linear programs. To the best of our knowledge the oldest approach is the so-called *simplex algorithm* of G. B. Dantzig [34], which led to a "development stimulus" of linear programming in the 50s and 60s. Nowadays, this method is widely used in practice. The idea of this algorithm is to make use of the fact that the feasible set of a linear program is a polyhedron and can be sketched as follows:

(1) Find a corner in the feasible region.

(2) Walk along a descending edge (reduce $\mathbf{c}^T\mathbf{x}$) to a neighboring corner.

(3) Repeat (2) until no descending edge can be found.

The simplex algorithm is the dominating method for solving integer linear programs, see Section 2.7.1. Its approach allows easily for the incorporation of additional linear constraints ("cutting planes") during the optimization process.

Since 1984 another class of techniques for solving linear programs has been developed: the so-called interior-point methods. As mentioned before, interior-point methods are a class of techniques that can be applied to general convex optimization problems. The basic ideas can be adapted to many problems, in this case to linear programs, making use of special properties of each problem. They are based on non-linear programming techniques. In contrast to the simplex method they do not work on the edges and corners of the feasible polyhedron; they converge to the solution out of the interior of the feasible region. This approach to

converge to the solution is usually based on an adapted Newton method for solving systems of nonlinear equations. The advantage of interior-point methods is their guaranteed polynomial run time. Due to Khachiyan [98] and its skillful rounding techniques, these methods do not need an exponential number of calculations. This is also guaranteed in practice. However, the disadvantage of this class of algorithms is their enormous storage requirements and the exact calculations that have to be performed numerically stably. The accumulation of rounding errors can lead to incorrect results. Certainly, this also holds true for the simplex algorithm but its comparatively easy kind of calculations is less affected. For a survey on interior-point methods in the context of semidefinite programming (an extension to linear programming in a certain sense, see Section 2.6.3), the reader is referred to Todd [172] and Rurainski [150].

2.6.2 Quadratic Programs

If all functions in (2.12) are quadratic, the resulting problem is called a quadratic program. Then, all functions can be represented by

$$f_i(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} + c_i$$

with

$$\mathbf{A}_i \in \mathbb{S}^n,\ \mathbf{b_i} \in \mathbb{R}^n,\ \text{and } c_i \in \mathbb{R}.$$

If $\mathbf{A}_i$ is not positive semidefinite, the corresponding f_i is not convex and, thus, solving a quadratic program to global optimality in general can be hard. In fact, Pardalos and Vavasis [138] have shown that quadratic programs are NP-hard, even when $\mathbf{A}_0$ has just one negative eigenvalue and when the constraint functions f_i, $i = 1, \ldots, n$ are linear. See for example Sherali and Tuncbilek [162] for an approach to deal with non-convex quadratic programs. If all $\mathbf{A}_i$ are positive semidefinite, the resulting quadratic program is a convex optimization problem.

Convex quadratic programs with only linear constraints, i.e. f_i, $i = 1, \ldots, m$ are linear or equivalently $\mathbf{A}_i = 0$, $i = 1, \ldots, m$, can be solved by the simplex method for quadratic programming, see, e.g., [190, 155, 178]. Linear constraints are a prerequisite for the simplex algorithm since it makes use of the polyhedron property of the feasible region as mention in the previous section.

General convex quadratic programs can be solved by interior-point methods. Although interior-point methods specialized to quadratic problems exist, each convex quadratic program can be transformed into a semidefinite program, see next section, and can be solved by a semidefinite program solver. Certainly, this software uses also an interior-point method. The advantage is that some excellent semidefinite program solvers are freely available [21].

2.6.3 Semidefinite Programs

Semidefinite programs can be seen as a generalization of linear programs to matrix variables. They are the most general kind of constrained optimization problems we consider in this work. Among others, linear programs as well as convex quadratic programs can be formulated as semidefinite programs, see Jarre and Stoer [86].

Definition 2.6.6. (Primal Semidefinite Program)

Let $\mathbf{A}_i \in \mathbb{S}^n$, $i = 1, \ldots, m$, $\mathbf{C} \in \mathbb{S}^n$, *and* $\mathbf{b} = (b_1, \ldots, b_m)^T \in \mathbb{R}^m$. *Then, we call the problem*

$$\begin{aligned} \min_{\mathbf{X} \in \mathbb{S}^n} \quad & \langle \mathbf{C}, \mathbf{X} \rangle \\ s.\ t. \quad & \langle \mathbf{A}_i, \mathbf{X} \rangle = b_i, \quad i = 1, \ldots, m, \\ & \mathbf{X} \succeq 0 \end{aligned}$$

a ***semidefinite program*** *in* ***primal standard form***.

Using techniques of convex duality, see Rockafellar [147], we obtain the dual of the above primal problem.

Definition 2.6.7. (Dual Semidefinite Program)
Let $\mathbf{A}_i \in \mathbb{S}^n$, $i = 1, \ldots, m$, $\mathbf{C} \in \mathbb{S}^n$, *and* $\mathbf{b} = (b_1, \ldots, b_m)^T \in \mathbb{R}^m$. *Then, we call the problem*

$$\begin{aligned} \max_{\substack{\mathbf{y} \in \mathbb{R}^n \\ \mathbf{Z} \in \mathbb{S}^n}} \quad & \mathbf{b}^T \mathbf{y} \\ s.\ t. \quad & \sum_{i=1}^{m} y_i \mathbf{A}_i + \mathbf{Z} = \mathbf{C}, \\ & \mathbf{Z} \succeq 0 \end{aligned}$$

a ***semidefinite program*** *in* ***dual standard form***.

Primal and dual forms are connected by duality theory and can be transformed into each other. Consequently, both formulations are valid for semidefinite program solvers. These solvers use specialized interior-point methods to iteratively follow the so-called *central path* and to converge to the solution, see, e.g., [21]. For a good survey on semidefinite programming see for example Todd [172], Jarre and Stoer [86], and Rurainski [150]. Because of its relevance in practice, we give exemplary a semidefinite programming formulation of a quadratically constrained convex quadratic program.

Example 2.6.8. (Quadratically constrained convex quadratic program)
Let $\mathbf{A}_i \in \mathbb{R}^{n \times n}$, $\mathbf{b}_i, \mathbf{c}_i \in \mathbb{R}^n$, *and* $d_i \in \mathbb{R}$. *In addition, let* f_i *be*

$$f_i(\mathbf{x}) := (\mathbf{A}_i \mathbf{x} + \mathbf{b}_i)^T (\mathbf{A}_i \mathbf{x} + \mathbf{b}_i) - \mathbf{c}_i^T \mathbf{x} - d_i, \quad i = 0, \ldots, m,$$

then the convex quadratic optimization problem

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & f_0(\mathbf{x}) \\ \text{subject to } & f_i(\mathbf{x}) \leq 0, \ i = 1, \ldots, m \end{aligned}$$

is equivalent to the semidefinite program

$$\begin{aligned} \min_{\substack{\mathbf{x} \in \mathbb{R}^n \\ t \in \mathbb{R}}} \quad & t \\ s.\ t. \quad & \begin{pmatrix} \mathbf{I} & \mathbf{A}_0 \mathbf{x} + \mathbf{b}_0 \\ (\mathbf{A}_0 \mathbf{x} + \mathbf{b}_0)^T & \mathbf{c}_0^T \mathbf{x} + d_0 + t \end{pmatrix} \succeq 0, \\ & \begin{pmatrix} \mathbf{I} & \mathbf{A}_i \mathbf{x} + \mathbf{b}_i \\ (\mathbf{A}_i \mathbf{x} + \mathbf{b}_i)^T & \mathbf{c}_i^T \mathbf{x} + d_i \end{pmatrix} \succeq 0, \ i = 1, \ldots, m, \end{aligned}$$

where $\mathbf{I}$ *denotes the identity matrix. The necessary decompositions of the quadratic functions can be obtained by singular value decompositions of the corresponding matrices. The equivalence proof can be found in Rurainski [150].*

2.7 Discrete Optimization

If the feasible region of a optimization problem contains only a countable quantity of elements, the problem is called a *discrete optimization problem*. Since there exists an isomorphism to a subset of $\mathbb{Z}^n$ in this case, we restrict ourself to integer problems.

Definition 2.7.1. (Discrete Problems)

(a) If some conditions restrict all variables of an optimization problem to be integer, this problem is called an ***integer (optimization) problem****.*

(b) If only a few variables are restricted to be integer, the problem is called a ***mixed integer (optimization) problem****.*

(c) If some conditions restrict all variables to be only 0 or 1, this problem is called a ***0-1 (optimization) problem****.*

If the feasible region of a discrete minimization problem contains only a finite modest number of elements, we may test the values of the objective function at all these positions and take the points with the minimum value. In this situation, such a problem might be easier to solve than a continuous one where this approach is not possible. However, even if the feasible region contains only a finite number of elements, the problem to only identify feasible points can be NP-hard [104]. Furthermore, if there is an infinite number of feasible points, this enumeration-and-test approach cannot succeed. If there are simple bounds (upper and lower bounds) on every variable, we may apply the interval subdivision technique (with the known difficulties) mentioned in Section 2.5, but in general no method exists to solve discrete optimization problems without special properties. In the following sections, we describe two common discrete optimization problems crucial for this work. For general discrete problems the reader may be referred to Korte and Vygen [104], Cook et al. [30], Nemhauser and Wolsey [128], Papadimitriou and Steiglitz [137], Schrijver [160], and Wolsey [191] for comprehensive treatments of this subject.

2.7.1 Integer Linear Programs

Definition 2.7.2. (Integer Linear Program (ILP))
Let $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, *and* $\mathbf{A} \in \mathbb{R}^{n \times m}$, *then the problem*

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & \mathbf{c}^T \mathbf{x} \\ s.\ t. \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}, \\ & x_i \in \mathbb{Z},\ i = 1, \ldots, n \end{aligned}$$

is called an ***integer linear program (ILP)****.*

Integer linear programs are usually solved by Lagrange relaxation [104], branch-and-bound, or branch-and-cut methods. The basic idea in a branch-and-bound and branch-and-cut framework is to drop the integrality constraints and to solve the underlying LP (LP-relaxation). If this solution is already integral, we have found the solution. Otherwise:

(a) *Branch-and-bound*: If the problem could not yet be solved to integral optimality, a branching step is performed: For example a variable with non-integral value, e.g., $x_i = 1.276$, at the current solution is chosen and two subproblems are generated. The first contains an additional integral upper bound on x_i, in our example $x_i \leq 1$, the second an additional integral lower bound, here $x_i \geq 2$. Both (sub)problems are solved. The obtained values at the solution are lower bounds for the sought integral solution of the original problem. To this end, a list of subproblems has to be stored during the optimization process. If the solution of a subproblem is integral, it is stored if it is the first integral feasible solution or its value is lower than any previous found integral position. Non-integral solutions are lower bounds to the sought integral solution and, thus, subproblems whose lower bound is higher than an already found integral solution can be discarded. Finally, the original problem is solved if all appearing subproblems are fully processed. More general branching strategies can be found in Schrijver [160].

(b) *Cutting-plane approach*: If the problem could not yet be solved to integral optimality, so-called cutting planes (linear inequalities) are generated, which separate the non-integral solution from the feasible integer positions, see Figure 2.11. These constraints are added to the problem and the (new) problem is solved again. This approach is iterated until no further cutting plane could be identified. See for example [137] for a good survey about cutting planes.

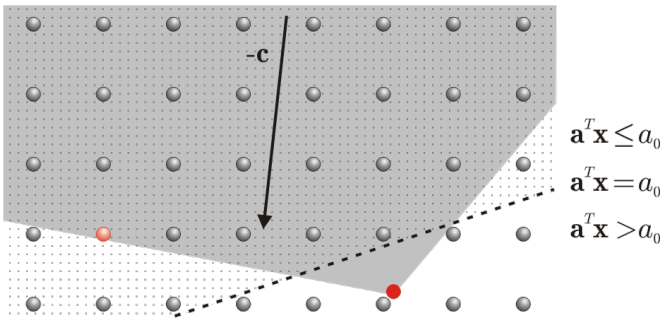

Figure 2.11: Separation of a non-integral solution from the feasible integer positions by a cutting plane. Grey: the feasible region; red ball: sought integer solution; red circle: non-integral LP solution; $-\mathbf{c}$: negative gradient of the objective function. The inequality $\mathbf{a}^T\mathbf{x} \leq a_0$ will be added to the LP-relaxation.

A pure cutting-plane method does not have to find any integral solution of the problem. Thus, the combination of both methods is widely used, i.e. use the cutting-plane method for each subproblem of a branching step. Branch only if no valid cutting plane could be identified. Methods following this combined approach are called *branch-and-cut* algorithms. A survey on how ILPs can be solved via branch-and-cut can be found in Papadimitriou and Steiglitz [137], Nemhauser and Wolsey [128], Korte and Vygen [104], see also Rurainski [149].

ILPs are an important class of problems for combinatorial optimization. In some cases, the full problem description has an exponential number of constraint inequalities. This is the case in Section 4.3, where we determine interesting subnetworks based on microarray data. In this case, we start with a basic set of inequalities, solve this problem, identify violated problem inequalities at this solution and add these constraints to the ILP. This process is iterated until no further violated inequality could be identified and, hence, the problem is

solved. Note that in the worst case all inequalities have to be added. Fortunately, this has not been the case in our calculations.

2.7.2 Quadratic 0-1-Programs

Definition 2.7.3. (Quadratic 0-1-Program)
Let $\mathbf{A} \in \mathbb{S}^n$ *and* $\mathbf{c} \in \mathbb{R}^n$. *The problem*

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ s.\ t. \quad & x_i \in \{0,1\},\ i = 1,\ldots,n \end{aligned} \tag{2.13}$$

is called a ***quadratic 0-1-program****.*

Quadratic 0-1 problems are another important class of discrete optimization problems. We have to solve this kind of problem in Section 4.2, where the problem of selecting an optimum subset out of a set of features based on a mutual information score is expressed as quadratic 0-1-program. The usual approaches to address these problems are branch-and-bound and branch-and-cut methods as they are sketched in the previous section. A summary of theoretical properties of the cut polytope for this class of problems can be found in [42].

The subproblems arising in a branch-and-cut framework can be solved by, e.g., the simplex method for quadratic programming [190, 155, 178], if the matrix $\mathbf{A}$ is positive semidefinite, i.e. the objective function is convex, see Section 2.6.2. If the quadratic function is not convex, the problem is known to be NP-hard. In fact, Pardalos et al. [138] have shown that this kind of problem is NP-hard, even if $\mathbf{A}$ has just one negative eigenvalue. As a consequence, if the quadratic function is not convex, the subproblems cannot be solved easily to guaranteed global optimality [71, 177, 176]. Numerical methods that try to achieve only local optimality are described in [60, 61, 59].

The key to solve (2.13) with arbitrary symmetric $\mathbf{A}$ is the observation that $x_i - x_i^2 = 0$ for $x_i \in \{0,1\}$, or equivalently, $u_i \cdot x_i - u_i \cdot x_i^2 = 0$ for all $u_i \in \mathbb{R}$ and $x_i \in \{0,1\}$. Thus, all problems of the class

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \mathbf{x}^T(\mathbf{A} - \mathrm{Diag}(\mathbf{u}))\mathbf{x} + (\mathbf{c}+\mathbf{u})^T\mathbf{x} \\ \text{s. t.} \quad & x_i \in \{0,1\},\ i = 1,\ldots,n \end{aligned} \tag{2.14}$$

with $\mathbf{u} \in \mathbb{R}^n$, where $\mathrm{Diag}(\mathbf{u})$ denotes the matrix with $\mathbf{u}$ on its main diagonal and zero off-diagonal entries, are equivalent to (2.13). The aim is to choose $\mathbf{u}$ in a way that (1) $\mathbf{A} - \mathrm{Diag}(\mathbf{u}) \succeq 0$ (to ensure that efficient techniques are applicable) and (2) we obtain good lower bounds for any relaxed version of (2.13). Higher lower bounds will improve the total run time of a branch-and-bound or branch-and-cut approach since subproblems can be discarded earlier. Certainly, it holds $\mathbf{A} - \mathrm{Diag}(\lambda_{\min}(\mathbf{A})) \succeq 0$ where $\mathrm{Diag}(\lambda_{\min}(\mathbf{A}))$ denotes the matrix with the minimum eigenvalue of $\mathbf{A}$ on its diagonal and zero off-diagonal entries, but preliminary studies have shown that the obtained bounds with this approach are bad. Branching can be very expensive in practice. Thus, it is crucial that we obtain the best existing lower bound. In fact, we solve

$$\begin{aligned} \mathbf{u} = \quad \arg\max_{\mathbf{v}\in\mathbb{R}^n} \quad & \min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{x}^T(\mathbf{A} - \mathrm{Diag}(\mathbf{v}))\mathbf{x} + (\mathbf{c}+\mathbf{v})^T\mathbf{x} \\ & \text{s. t.} \quad \mathbf{A} - \mathrm{Diag}(\mathbf{v}) \succeq 0 \end{aligned} \tag{2.15}$$

to global optimality by semidefinite programming [19] as a preprocessing step. The following theorem is central to this task. We present the proof here because it is omitted in [19].

Theorem 2.7.4. *Problem (2.15) is equivalent to the semidefinite program*

$$\begin{aligned} \max_{\substack{\mathbf{u}\in\mathbb{R}^n \\ r\in\mathbb{R}}} \quad & r \\ s.\ t. \quad & \begin{pmatrix} -r & \frac{1}{2}(\mathbf{c}+\mathbf{u})^T \\ \frac{1}{2}(\mathbf{c}+\mathbf{u}) & \mathbf{A}-\mathrm{Diag}(\mathbf{u}) \end{pmatrix} \succeq 0. \end{aligned} \tag{2.16}$$

Proof. Let $\mathbf{U}$ be

$$\mathbf{U} = \begin{pmatrix} -r & \frac{1}{2}(\mathbf{c}+\mathbf{u})^T \\ \frac{1}{2}(\mathbf{c}+\mathbf{u}) & \mathbf{A}-\mathrm{Diag}(\mathbf{u}) \end{pmatrix}.$$

Since $\mathbf{A}-\mathrm{Diag}(\mathbf{u})$ is a principal submatrix of $\mathbf{U}$, we have

$$\mathbf{U} \succeq 0 \quad \Rightarrow \quad \mathbf{A}-\mathrm{Diag}(\mathbf{u}) \succeq 0$$

by Theorem A.13 in [150], see also Fact 8 in [172]. Let us assume now

$$\mathbf{A}-\mathrm{Diag}(\mathbf{u}) \succ 0$$

and, hence, (2.15) has a unique minimum. (The case $\mathbf{A}-\mathrm{Diag}(\mathbf{u}) \succeq 0$ is covered by the fact that the solution of a semidefinite program always lies on the boundary of the semidefinite cone.) The minimizer $\mathbf{x}^*$ of (2.15) fulfills

$$\mathbf{x}^* = -\frac{1}{2}(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u}).$$

By inserting $\mathbf{x}^*$ into (2.15), we obtain the value of (2.15) at $\mathbf{x}^*$

$$\begin{aligned} & \mathbf{x}^{*T}(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))\mathbf{x}^* + (\mathbf{c}+\mathbf{u})^T\mathbf{x}^* \\ = \ & \frac{1}{4}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u}) - \frac{1}{2}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u}) \\ = \ & -\frac{1}{4}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u}). \end{aligned}$$

We want to maximize this minimum. Thus, we maximize $r \in \mathbb{R}$ with the constraint

$$-\frac{1}{4}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u}) \geq r$$

or equivalently

$$-r - \frac{1}{4}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u}) \geq 0. \tag{$*$}$$

By the Schur complement, see Fact 11 in [172] or Theorem 1.18 in [150], it follows

$$(*) \quad \Leftrightarrow \quad \mathbf{U} \succeq 0.$$

Thus, the solution of (2.16) solves (2.15). The other way round, if $\mathbf{u}$ solves (2.15), we have that

$$-\frac{1}{4}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}(\mathbf{c}+\mathbf{u})$$

is at its maximum. Since (2.16) maximizes r, inequality $(*)$ becomes to

$$-\frac{1}{4}(\mathbf{c}+\mathbf{u})^T(\mathbf{A}-\mathrm{Diag}(\mathbf{u}))^{-1}\frac{1}{2}(\mathbf{c}+\mathbf{u}) = r$$

and r takes its maximum possible value. Hence, the solution of (2.15) solves (2.16). □

After this preprocessing step, we solve the resulting convex problem by a quadratic integer program solver via branch-and-cut. Note that $\mathbf{u}$ found in this preprocessing stage is the best vector we can expect, because Poljak et al. [142] prove that the maximum lower bounds of different types of relaxation are equal.

Chapter 3

Optimizing Molecular Structures

One of the most important problems arising in Computational Chemistry and Biology is the optimization of molecular structures by minimization of empirical potential energy functions (force fields). The applications stretch from (re)optimization of manually drawn molecules, e.g. in a drug design framework, see Figure 3.1, to molecular docking purposes. To this end,

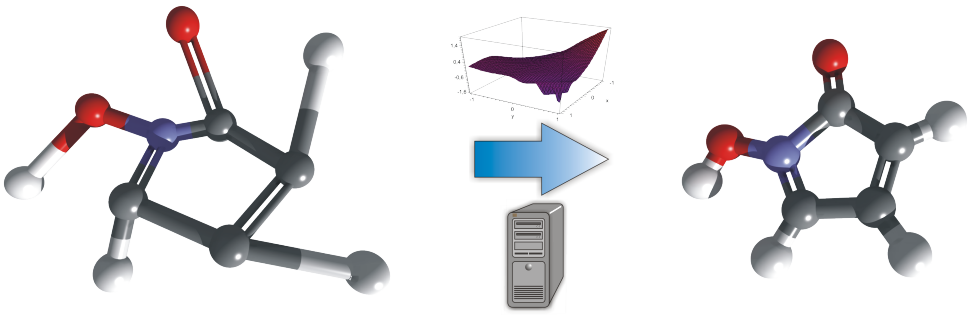

Figure 3.1: Optimizing a manually drawn molecule by "minimizing the structure" using a molecular potential energy function.

the molecules, i.e. their atomic conformations and the physicochemical properties, must be encoded as a set of real valued variables (parameters). The obvious way to represent the spatial arrangement of a molecule is by defining the Cartesian coordinates of all its atoms as well as a set of bonds that defines their interconnectivity. This parametrization is widely used in practice. The next section uses this obvious parametrization where we present our method to locally optimize molecular structures. This algorithm adapts optimally to the behavior of the objective function in the region of interest during the optimization procedure. We show that our approach is clearly superior to other commonly used standard methods.

For several applications in molecular modeling, like docking, it is reasonable to constrain molecular flexibility by fixing bond angles and bond lengths and restricting torsional flexibility to rotation around single bonds that connect rigid entities, like ring structures [15, 47, 124]. Alternatively to Cartesian coordinates, a molecule may be represented by its translation, orientation, and a set of torsional angles, bond angles, and bond lengths (compact representation) where such nonrelevant degrees of freedom can easily be frozen. However, this parametrization has a major drawback: there is no orientational parametrization that is at the same time minimal and free of singularities [159, 62]. In the course of optimization, a

molecule can reach positions in which not all possible further alterations of its orientation are achievable by variations of the orientational parameters. Thus, the optimization process may get stuck at non-optimum positions. Therefore, the main challenge with respect to the compact molecular representation is to find an orientational parametrization that is well suited for the efficient gradient-based local optimization techniques we described in Section 2.3. In Section 3.2, we present our method to overcome the problems with the compact molecular representation and to efficiently apply gradient-based local optimization techniques.

We extend this approach to a ligand-receptor docking technique in Section 3.3. We combine a multi-deme genetic algorithm with a Lamarckian genetic algorithm using our local search method. We show that this method is clearly superior to commonly employed approaches.

However, while treating the ligand fully flexible with respect to its torsional angles, this method still keeps the receptor fixed. We present our promising attempts to overcome this strong assumption in Section 3.4. Our approach extents the ideas behind the flexibility of the ligand (parametrization) to the receptor's side chains. Furthermore, we introduce limited backbone movement by interpolating between known extremal conformations. We present our formulas in detail and give analytical derivatives, necessary for our gradient-based optimization method. We show that our approach is able to dock ligands into regions with highly flexible side chains on our test example, the Human Serum Albumin (HSA). This protein is known for its promiscuity to bind different ligand species and, furthermore, undergoes tremendous backbone movements upon binding to fatty acids. HSA is the most abundant transport protein of the human blood plasma.

The last part of this chapter deals with our approach to allow a loop region of the receptor to be fully flexible, i.e. not only limited movement is possible as in our previous study requiring at least two different high resolution structures of the backbone, but all possible conformations can be considered, see Section 3.5. To this end, we use the Gō-Scheraga loop closure equations and present our interval arithmetic based method to guarantee that we can calculate all possible conformations. In this study, we chose the human 17β-hydroxysteroid dehydrogenase type 1 (17β-HSD 1), which plays an important role in the development and proliferation of breast cancer and other estrogen-dependent diseases. We show that our approach is able to dock estradiol into the highly flexible loop region of 17β-HSD 1, where other approaches failed in our preliminary studies.

3.1 The Consensus Line Search Approach

The local optimization of molecular structures is a well understood and studied task in bioinformatics and computational chemistry. Solving this problem usually requires efficient local minimization techniques, i.e. iterative two-step methods that search first for a descent direction and then try to estimate the step width as described in Section 2.3. In Section 2.3.2.1, we showed the general procedure of a line search-based numerical local optimization method and mentioned the so-called strong Wolfe conditions (2.3) and (2.4) for accepting an iterate.

To the best of our knowledge, all line search algorithms employed in molecular minimization software use quadratic and cubic polynomial interpolation schemes to estimate the next trial step. This is due to the simplicity of the calculations and uniqueness of the interpolants. It is important to note that the choice between possibilities only serves to guarantee interpolation and extrapolation criteria. It is, however, usually not utilized to adapt optimally to

the behavior of the objective function in the region of interest. If both, quadratic and cubic steps, fulfill the required interpolation and extrapolation criteria, the cubic step is often taken since it additionally fulfills all derivative conditions.

However, none of these choices is able to accurately represent the rational or exponential behavior that often dominates force field based potential energy functions. To model this effect, we adopt ideas of conic interpolation [36, 161, 20], i.e. an interpolation method using the conic model function

$$\psi_{\text{conic}}(\lambda) = \frac{a\lambda^2 + b\lambda + c}{(d\lambda + 1)^2},$$

and rational interpolation

$$\psi_{\text{rational}}(\lambda) = \frac{p\lambda^2 + q\lambda + r}{s\lambda + 1}.$$

Bjørstad and Nocedal [20] prove local convergence with Q-order equal 2 for one-dimensional minimization based on the pure conic interpolation method, rendering the proposed interpolation scheme an alternative to polynomial interpolation.

But conic and rational interpolation alone will not "globally" improve convergence: Dependent on the current position on the energy surface, polynomial, rational, or exponential behavior may dominate the potential function. This motivates the idea to locally decide on the most suitable interpolant based on the comparison of the interpolation results, see Figure 3.2. However, a naive approach – using multiple separate interpolation results – would significantly increase the number of evaluations required.

In this section, we show how the decision for the most reliable interpolation can be made with negligible computational overhead, see also Rurainski et al. [154]. We demonstrate that the approach leads to substantially improved performance in comparison to two standard line search algorithms on almost all test cases. We also show that the scheme can be easily extended. For example, to handle the case of non-smooth functions (e.g. the famous Gehlhaar scoring/energy function [56]) we have included two additional interpolating/approximating schemes specially suited to this case: two side linear piecewise interpolations (derivative of both sides and function values, the intersection of both interpolants is the sought minimizer if there is any) and a four point quadratic least squares approximation (four collected function values, the function is the quadratic least squares approximation). The user can choose which of these functions to include into the set of interpolants.

3.1.1 The Line Search Procedure

Recalling the task of a line search from Section 2.3.2.1, where we drop the iteration subscript k in order to simplify the notation and use the objective function f, the current position $\mathbf{x}$, and the current search direction $\mathbf{d}$, the problem addressed by a line search can be seen as a one-dimensional approximative minimization problem with the objective function $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ with

$$\phi(\lambda) := f(\mathbf{x} + \lambda\mathbf{d}), \tag{3.1}$$

The derivative of ϕ is then given by

$$\phi'(\lambda) = (\nabla f(\mathbf{x} + \lambda\mathbf{d}))^T \mathbf{d}$$

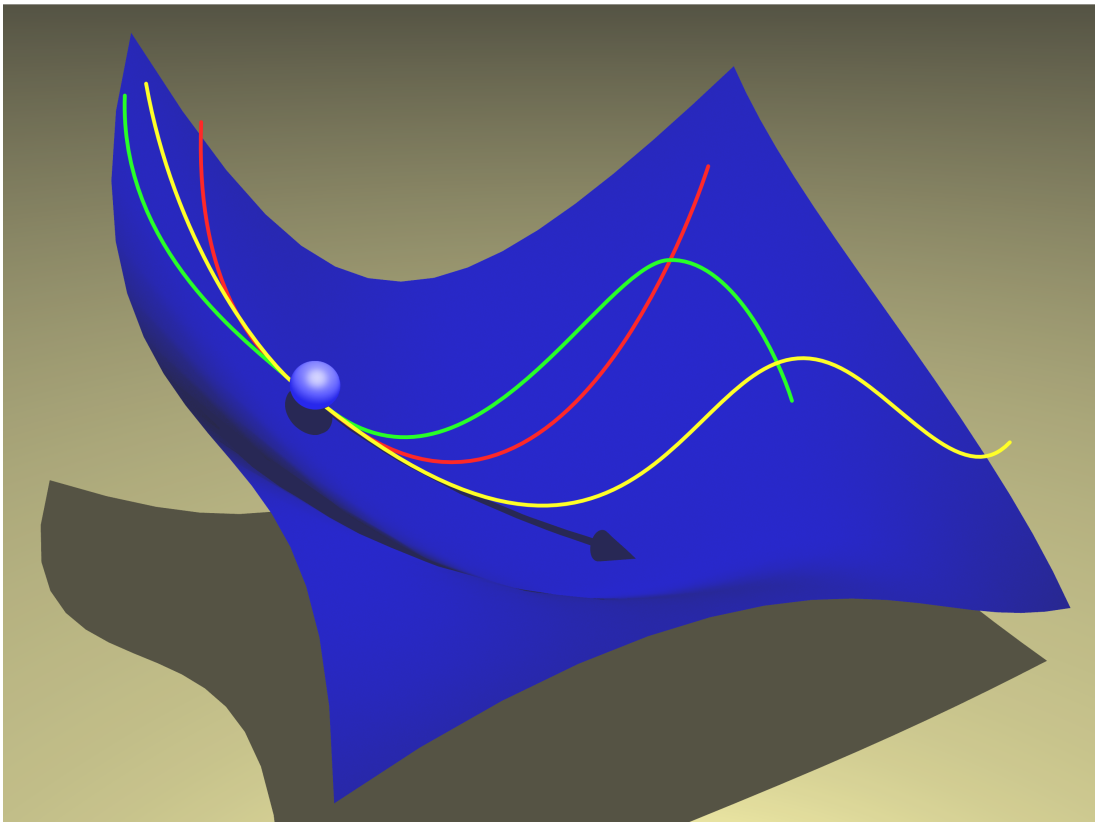

Figure 3.2: Searching the next local minimum on an energy surface by our consensus approach. This graphical abstract appeared on the cover of the Journal of Computational Chemistry, Vol 30, July 15, 2009.

and the strong Wolfe conditions become

$$\phi(\lambda) \leq \phi(0) + \alpha\lambda\phi'(0) \tag{3.2}$$

$$|\phi'(\lambda)| \leq \beta|\phi'(0)|. \tag{3.3}$$

Finding points that fulfill the strong Wolfe conditions does require not only points close to a local minimum (fulfilling condition (3.3)) but also satisfying sufficient decrease (condition (3.2)). Line search methods often choose to use the auxiliary function

$$\widetilde{\phi}(\lambda) := \phi(\lambda) - \phi(0) - \alpha\lambda\phi'(0) \tag{3.4}$$

instead of ϕ if problems with insufficient decrease occur: A local minimum of $\widetilde{\phi}$ always corresponds to a point that fulfills conditions (3.2) and (3.3) for the function ϕ.

Theorem 3.1.1. *Let $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ with $\phi'(0) < 0$ denote the one dimensional objective function defined in (3.1) and $\widetilde{\phi}$ the auxiliary function defined in (3.4). Then it holds*

(a) $\widetilde{\phi}'(0) < 0$

(b) any local minimizer λ^* of $\widetilde{\phi}$ identified by the two stage bracketing approach fulfills the convergence criteria (3.2) and (3.3).

Proof. (a) Using the definition (3.4) of $\widetilde{\phi}$ yields

$$\widetilde{\phi}'(0) = \phi'(0) - \alpha\phi'(0) = (1-\alpha)\phi'(0) < 0$$

since $0 < \alpha < \frac{1}{2}$ and $\phi'(0) < 0$, see Section 2.3.2.1.

(b) Let λ^* be a minimizer of $\widetilde{\phi}$, then we have

$$\widetilde{\phi}'(\lambda^*) = \phi'(\lambda^*) - \alpha\phi'(0) = 0.$$

It follows

$$|\phi'(\lambda^*)| = \alpha|\phi'(0)|$$

and hence with $0 < \alpha < \beta$

$$|\phi'(\lambda^*)| \leq \beta|\phi'(0)|,$$

i.e. λ^* fulfills the curvature criterion (3.3). Finally, we have (from (3.4))

$$\widetilde{\phi}(0) = 0$$

and the update procedure looking for a minimizer guarantees to find a point λ^* whose function value is less than or at least equal to the initial value, i.e.

$$\widetilde{\phi}(\lambda^*) \leq \widetilde{\phi}(0) = 0.$$

Thus, it follows

$$\phi(\lambda^*) \leq \phi(0) + \alpha\phi'(0)\lambda^*$$

and condition (3.2) is fulfilled. □

By merely minimizing a one-dimensional function, either $\widetilde{\phi}$ or ϕ, we can always guarantee the strong Wolfe conditions. How to minimize such a function in practice will be discussed in the following. To simplify notation, we will from now on refer to the objective function by ϕ always.

All existing inexact line search algorithms perform the minimization by using interpolation schemes and setting the next trial step to the minimizer of the current interpolant. The best known approaches are two-stage bracketing algorithms. In the first stage, a point that fulfills criteria (3.2) and (3.3) is bracketed (in an ideal case a minimum) by moving or enlarging an interval, say I, until it can be ensured that a suitable point lies within I. From a mathematical point of view, this is an extrapolation stage where increasing steps are performed and I is updated dependent on these steps and local information of ϕ. In the second stage, the size of I is iteratively decreased until a $\lambda \in I$ could be found that fulfills (3.2) and (3.3). This second stage can be seen as an interpolation stage. The main reason for the good performance of the two-stage bracketing approach in practice is its provable convergence, while non-bracketing methods cannot guarantee convergence. Therefore, in this work we are focusing on two stage approaches only.

When the line search starts, we do not yet have an initial trial step that will become the upper bound of the initial interval I. The standard choice for this initial estimate in the context of (quasi-)Newton methods is $\lambda = 1$. This means our first trial step is a full step as recommended by the convergence theory behind Newton methods [84]. However, other choices for initial estimates have been proposed in the literature [36, 50, 133]. Since (quasi-)

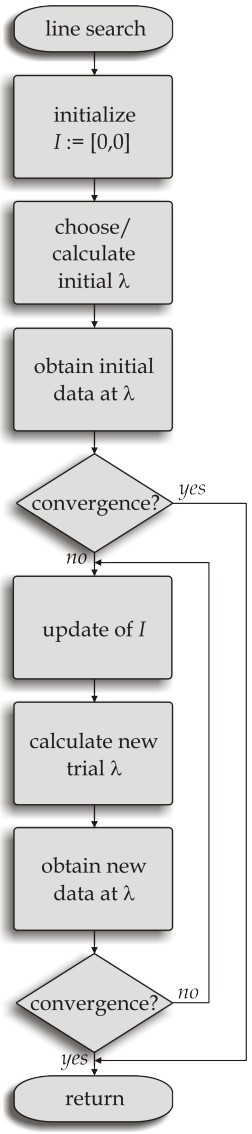

Figure 3.3: The general scheme of a (bracketing) line search algorithm.

Newton methods are the most efficient approaches, we focus on these algorithms. Thus, $\lambda = 1$ is a suitable initial estimate widely used in practice. In summary, the general scheme of a line search algorithm is given in Figure 3.3. Note that the initial choice $I = [0, 0]$ is legitimate.

In a two stage (bracketing) approach, different functions are computed, which interpolate the data at the bounds of I. Afterwards, one bound is replaced (update of I) by the new trial step (minimum of the interpolant). The choice of interpolating functions in common line search algorithms is only driven by the idea of using functions, which provide steps in the right direction, i.e., its minimizer lies right of I in the first (bracketing) stage (to ensure extrapolation) and inside of I in the second stage (to ensure reduction of the size of I). In the case, when more than one interpolating function provide suitable steps (say quadratic and cubic), there are two common choices how to proceed:

(A) If there are functions not fulfilling all interpolation conditions, the function that satisfies

most of the conditions is chosen, e.g., cubic interpolation is preferred to quadratic interpolation.

(B) The function, which provides a step closest to the bound of I with the lower energy value, is chosen.

Thus, existing two-stage bracketing algorithms discard local behavior of the objective function.

3.1.2 The New Approach

In this section, we describe our consensus line search approach that utilizes the local behavior of the objective function in order to accelerate the optimization process, see also Rurainski et al. [154].

As stated before, the interval I is updated by replacing one bound with the minimizer of the chosen interpolant. While other line search approaches discard old data, we propose to store the knowledge about the behavior of the objective function at the replaced bound of I, i.e. the bound itself (denoted by λ_{ctrl}) and the energy value at this bound (denoted by $\phi_{\text{ctrl}} := \phi(\lambda_{\text{ctrl}})$). We call these data *control data*. Whenever I is updated, we replace the control data with the values at the replaced bound. In summary, our approach stores two additional floating point numbers in comparison to standard line search methods. We use these data to decide which interpolating function might provide the most reliable result.

Calculation of λ

We use five different interpolation schemes that model the typical behavior of potential molecular energy functions: (1) conic, (2) rational, (3) cubic, (4) quadratic based on two energy values and one derivative, and (5) quadratic based on two derivatives and one energy value. We denote the interpolants behind these schemes with ψ_{conic}, ψ_{rational}, ψ_{cubic}, $\psi_{\text{quadratic}}$, and ψ_{secant}, respectively. At each step, the different interpolation schemes need to be fitted to the available data using the interpolation criteria. For rational and conic functions, however, this fit is not unique but only one of the possible solutions has a minimum.

At any given step, some of the interpolants might violate certain consistency criteria. If such a violation is detected, we filter out the corresponding interpolant. The filtering procedure will be described in the following section.

The obvious approach for choosing the best of the remaining schemes would be to compare the minimizer of all appropriate interpolants with the function values at the minimizing positions. This, however, would result in additional energy evaluations. Instead, we first choose the interpolant, say $\widetilde{\psi}$, which best fits the control data, i.e.

$$\widetilde{\psi} := \arg\min_{\psi_s}\{|\psi_s(\lambda_{\text{ctrl}}) - \phi_{\text{ctrl}}| \ : \ \psi_s \text{ appropriate}\}. \tag{3.5}$$

where $s \in \{\text{conic, rational, cubic, quadratic, secant}\}$, and the best candidate λ for the next update of I is then calculated via

$$\lambda := \arg\min_{\gamma>0} \widetilde{\psi}(\gamma)$$

assuming $\widetilde{\psi}$ represents best the local behavior of ϕ in the current region.

The consistency criteria mentioned above are required to guarantee two different properties. First, we need to ensure that the interpolant leads to extrapolation in the extrapolation stage and vice versa. This is a standard requirement of line search approaches, see for example Moré and Thuente [123]. In some cases this can be decided based on interpolant values and derivatives. Second, in contrast to established line search approaches, we introduce an additional criterion that guarantees the presence of inflection points if this can be inferred from the function values. The full case differentiation used by our approach is given in Figure 3.4, where $I := [\lambda_{\text{lo}}, \lambda_{\text{up}}]$.

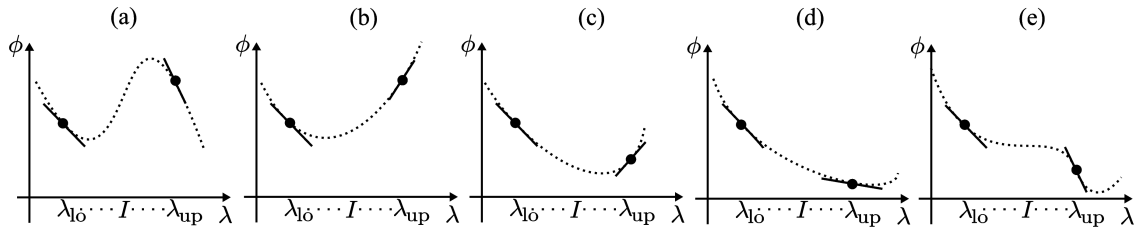

Figure 3.4: The five different cases for interpolation/update procedures. Points: given values. Solid lines: given tangents (derivatives). Dotted lines: assumed curve progression of ϕ. (a) $\phi(\lambda_{\text{lo}}) < \phi(\lambda_{\text{up}})$, minimum bracketed, $\phi'(\lambda_{\text{lo}}) \cdot \phi'(\lambda_{\text{up}}) > 0$, ϕ must have an inflection point, sensible interpolations: conic, rational, cubic. (b) $\phi(\lambda_{\text{lo}}) < \phi(\lambda_{\text{up}})$, minimum bracketed, $\phi'(\lambda_{\text{lo}}) \cdot \phi'(\lambda_{\text{up}}) < 0$, sensible interpolations: conic, rational, cubic, quadratic (two derivatives), quadratic (one derivative, two values). (c) $\phi(\lambda_{\text{lo}}) > \phi(\lambda_{\text{up}})$, $\phi'(\lambda_{\text{lo}}) \cdot \phi'(\lambda_{\text{up}}) < 0$, minimum bracketed, sensible interpolations: conic, rational, cubic, quadratic (two derivatives). (d) $\phi(\lambda_{\text{lo}}) > \phi(\lambda_{\text{up}})$, $|\phi'(\lambda_{\text{lo}})| > |\phi'(\lambda_{\text{up}})|$, minimum not necessarily bracketed, sensible interpolations: conic, rational, cubic, quadratic (two derivatives). (e) $\phi(\lambda_{\text{lo}}) > \phi(\lambda_{\text{up}})$, $|\phi'(\lambda_{\text{lo}})| < |\phi'(\lambda_{\text{up}})|$, minimum not necessarily bracketed, sensible interpolations: conic, rational, cubic.

Even though the efficient filtering described in the last section removes the substantial amount of inconsistencies the resulting λ might still be inappropriate, e.g. it might fall out of the interval of interest I. If such a violation is detected, the next best interpolant is tested until either a suitable step can be identified or all interpolation schemes have failed.

In the latter case, we proceed by an "increasing extrapolation strategy" in the first stage, i.e. $\lambda := \sigma \cdot \lambda_{\text{up}}$, $\sigma := \xi \cdot \sigma_{\text{prev}}$ (the previous extrapolation factor), $\xi > 1$. In the second stage we proceed by a bisection step. This calculation of the next trial step λ is given in Figure 3.5. It is worth noting that none of the safeguard cases have occurred in our molecular test runs.

Our experiments indicate that control data should be updated in every step. This typically results in control steps very close to the current interval I. Thus, the decision whether polynomial or exponential behavior dominates ϕ is very reliable. At the start of the optimization, where we do not have control data, we use the standard choice (B).

Update of I

Once we have calculated a new trial step that does not yet lead to convergence, we have to update the current interval I. Some update schemes have been proposed in the literature, see, e.g., Moré and Thuente [123] and Al-Baali and Fletcher [3]. We decided to use the update rules of Moré and Thuente, as these rules guarantee that the interval always contains an

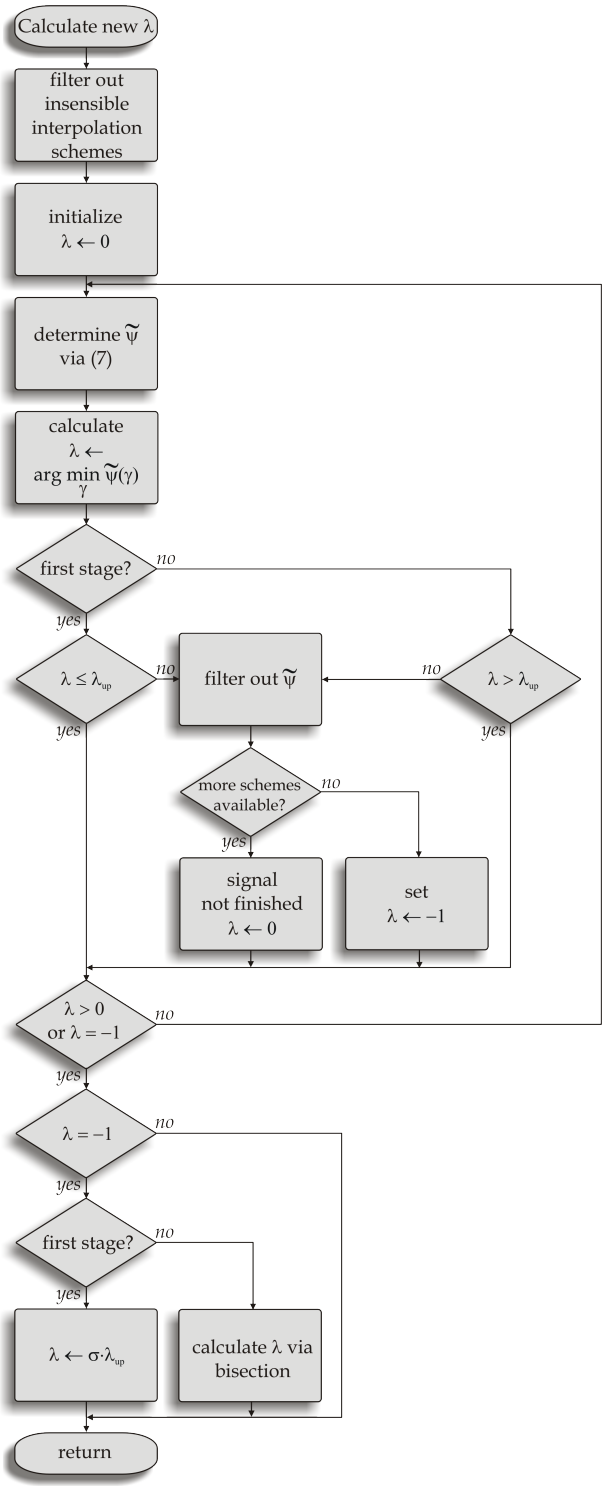

Figure 3.5: The Consensus Approach: Calculation of λ after the initial case.

acceptable point.[1]

[1]Note that the original scheme uses relative steps, i.e. the bounds of I can be switched. This results in an additional case differentiation when our next λ is computed. Thus, we avoid relative steps.

In the initial step, where $\lambda_{\text{up}} = 0$, we simply store the initial data. In the subsequent steps, we have to distinguish five cases that are shown in Figure 3.4. The first three cases in Figure 3.4 correspond to the standard update scheme of Moré and Thuente [123]. The update scheme for these cases is shown in Figure 3.6. In the last two cases, we modify the

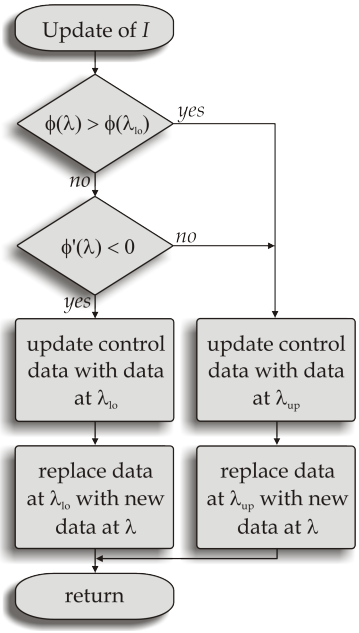

Figure 3.6: The Consensus Approach: Update of I in interpolation stage, cases (a), (b), and (c) of Figure 3.4. Replacing data means replace the position, the energy value and the derivative at that position.

scheme of Moré and Thuente. Our observations on molecular energy functions have shown that in these cases the deeper minimum is usually located to the right of I. Thus, we move I instead of increasing it (see Figure 3.7). This replacement yielded improved convergence in our experiments.

Use of $\widetilde{\phi}$

Whenever insufficient decrease was detected the auxiliary function $\widetilde{\phi}$ can be employed in several ways. Moré and Thuente [123] use $\widetilde{\phi}$ only in the first stage of the algorithm. They argue that the interpolation schemes might work better on the unmodified function ϕ and they prove that employing $\widetilde{\phi}$ in the second stage is not required to achieve convergence. Our experiments, however, revealed that whenever $\widetilde{\phi}$ was used for bracketing using it for interpolation as well improves convergence. Consequentely, we use $\widetilde{\phi}$ whenever we detect insufficient decrease independent of the current stage. Similarly, data from ϕ and $\widetilde{\phi}$ should not be mixed during the interpolation process.

3.1.3 Comparison to Other Line Search Methods

The consensus line search approach has been implemented in C++ using the C++ software library BALL [101, 102, 76]. We used GNU g++, version 4.2.3, with -O2 optimization flag.

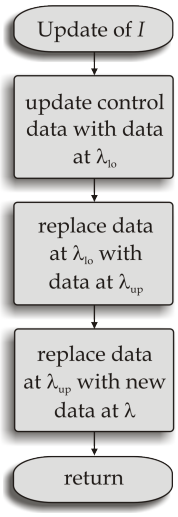

Figure 3.7: The Consensus Approach: Update of I in extrapolation stage, cases (d) and (e) of Figure 3.4. Replacing data means replace position, the energy value and the derivative at that position.

All test runs were performed on an Intel Core2 processor (2.0 GHz, 2GByte RAM), Linux, kernel 2.6.24.

In order to approve that our approach is highly efficient, we included all possible interpolants in all test runs, even though the functions for the non-smooth case are expected to be useless for molecular potentials and rather expensive to evaluate (note that the least squares solution requires a singular value decomposition). We compared the performance of our line search method to two standard line search approaches: the algorithm of Moré and Thuente [123] (MT line search) and the method of Wright and Nocedal [133] (WN line search).

The first algorithm has been widely used on many kinds of optimization problems and is widely accepted as the one of the best line search algorithms currently known. It has been used in chemistry, see for example Xie and Schlick [192] and Schlick and Fogelson [157, 158], as well as on mathematical test problems, see for example Gilbert and Nocedal [57] and Liu and Nocedal [110]. The original Fortran implementation of Moré and Thuente can be found in MINPACK-2 [122, 9]. In order the combine Fortran code with BALL, we used the tool f2c [48] to convert the Fortran code to plain C and adapted the resulting code to BALL's data structures. In order to ensure that we did not modify performance and accuracy of the algorithm with this conversion, we compared the results of our implementation to the results published by Moré and Thuente [123] on their test functions. As shown in Table 3.1 the results are completely equivalent. We also compared the performance of our method and the second line search approach on these test functions. Although our interpolations have not been designed for these kinds of functions but rather for molecular functions, our method performed better in 13 of 24 test runs and performed slightly worse in 6 runs compared to the algorithm of Moré and Thuente [123]. We do not report run times here because all test runs finished nearly instantly. Since for the method of Wright and Nocedal [133] no suitable implementation seems to be available, we implemented the algorithm from scratch. Our implementation closely follows the recommendations made in Wright and Nocedal [133], in particular Chapter 3 of line searches. Table 3.1 shows that this algorithm is inferior to our method and the method of Moré and Thuente [123] on these test functions. This is due

		MTlit			MT			WN			CLS		
Table[123]	x_0	m	x_m	ϕ'	m	x_m	ϕ'	m	x_m	ϕ'	m	x_m	ϕ'
5.1	10^{-3}	6	1.4	$-9.2\ 10^{-3}$	6	1.365	$-9.16\ 10^{-3}$	13	1.426	$2.10\ 10^{-3}$	**4**	8.158	$1.37\ 10^{-2}$
	10^{-1}	3	1.4	$4.7\ 10^{-3}$	3	1.441	$4.66\ 10^{-3}$	5	1.6	$2.69\ 10^{-2}$	4	1.497	$1.34\ 10^{-2}$
	10^{+1}	1	10	$9.4\ 10^{-3}$	1	10.0	$9.41\ 10^{-3}$	1	10.0	$9.42\ 10^{-3}$	1	10.0	$9.4\ 10^{-3}$
	10^{+3}	4	37	$7.3\ 10^{-4}$	4	36.888	$7.31\ 10^{-4}$	4	37.054	$7.25\ 10^{-4}$	**2**	28.004	$1.27\ 10^{-3}$
5.2	10^{-3}	12	1.6	$7.1\ 10^{-9}$	12	1.596	$3.81\ 10^{-9}$	16	1.596	$7.57\ 10^{-10}$	**10**	1.596	$9.93\ 10^{-10}$
	10^{-1}	8	1.6	$1.0\ 10^{-10}$	8	1.596	$1.01\ 10^{-10}$	x	x	x	11	1.596	$2.06\ 10^{-15}$
	10^{+1}	8	1.6	$-5.0\ 10^{-9}$	8	1.596	$-4.97\ 10^{-9}$	9	1.596	$-1.05\ 10^{-11}$	8	1.596	$1.64\ 10^{-12}$
	10^{+3}	11	1.6	$-2.3\ 10^{-8}$	11	1.596	$-2.31\ 10^{-8}$	x	x	x	14	1.596	$3.71\ 10^{-12}$
5.3	10^{-3}	12	1.0	$-5.1\ 10^{-5}$	12	1.0	$-5.14\ 10^{-5}$	16	1.0	$-3.21\ 10^{-6}$	14	1.0	$-1.72\ 10^{-7}$
	10^{-1}	12	1.0	$-1.9\ 10^{-4}$	12	1.0	$-1.92\ 10^{-4}$	14	0.999	$-3.83\ 10^{-4}$	**10**	1.0	$3.39\ 10^{-5}$
	10^{+1}	10	1.0	$-2.0\ 10^{-6}$	10	1.0	$-1.98\ 10^{-6}$	8	1.0	$-3.21\ 10^{-6}$	13	1.0	$8.32\ 10^{-3}$
	10^{+3}	13	1.0	$-1.6\ 10^{-5}$	13	1.0	$-1.58\ 10^{-5}$	13	1.0	$-1.66\ 10^{-5}$	14	1.0	$5.21\ 10^{-3}$
5.4	10^{-3}	4	0.08	$-6.9\ 10^{-5}$	4	0.085	$-6.85\ 10^{-5}$	6	0.032	$-4.87\ 10^{-4}$	4	0.327	$-3.56\ 10^{-6}$
	10^{-1}	1	0.10	$-4.9\ 10^{-5}$	1	0.10	$-4.93\ 10^{-5}$	1	0.1	$-4.93\ 10^{-5}$	1	0.1	$-4.93\ 10^{-5}$
	10^{+1}	3	0.35	$-2.9\ 10^{-6}$	3	0.349	$-2.92\ 10^{-6}$	3	0.349	$-2.91\ 10^{-6}$	**2**	0.553	$8.62\ 10^{-7}$
	10^{+3}	4	0.83	$1.6\ 10^{-5}$	4	0.829	$1.64\ 10^{-5}$	4	0.829	$1.64\ 10^{-5}$	**3**	0.667	$3.38\ 10^{-6}$
5.5	10^{-3}	6	0.075	$1.9\ 10^{-4}$	6	0.075	$1.90\ 10^{-4}$	9	0.076	$4.16\ 10^{-4}$	**5**	0.074	$3.86\ 10^{-5}$
	10^{-1}	3	0.078	$7.4\ 10^{-4}$	3	0.078	$7.39\ 10^{-4}$	3	0.071	$-6.13\ 10^{-4}$	3	0.074	$6.65\ 10^{-5}$
	10^{+1}	7	0.073	$-2.6\ 10^{-4}$	7	0.073	$-2.56\ 10^{-4}$	x	x	x	**6**	0.071	$-7.30\ 10^{-4}$
	10^{+3}	8	0.076	$4.5\ 10^{-4}$	9	0.076	$4.49\ 10^{-4}$	x	x	x	**7**	0.071	$-7.32\ 10^{-4}$
5.6	10^{-3}	13	0.93	$5.2\ 10^{-4}$	13	0.927	$5.22\ 10^{-4}$	15	0.926	$5.33\ 10^{-6}$	**10**	0.929	$7.34\ 10^{-4}$
	10^{-1}	11	0.93	$8.4\ 10^{-5}$	11	0.926	$8.35\ 10^{-5}$	x	x	x	**8**	0.928	$7.32\ 10^{-4}$
	10^{+1}	8	0.92	$-2.4\ 10^{-4}$	8	0.924	$-2.37\ 10^{-4}$	7	0.926	$7.64\ 10^{-6}$	**6**	0.926	$8.51\ 10^{-6}$
	10^{+3}	11	0.92	$-3.2\ 10^{-4}$	11	0.924	$-3.24\ 10^{-4}$	x	x	x	**7**	0.926	$3.25\ 10^{-6}$

Table 3.1: Performance on artificial test problems taken from Moré and Thuente [123] with varying parameters and initial step sizes. Note: these functions do not have a unique position satisfying the convergence criteria. As in the original publication we use these functions only as a benchmark for finding fast one acceptable position, i.e. criteria (3.2) and (3.3) are fulfilled. Additionally, we verify that our (f2c'd) reimplementation of the line search of Moré and Thuente indeed yields the same results as their reference implementation. m = number of steps each process took to find an acceptable point, x_0 = initial step size, ϕ' = derivative at the found position, MTlit = results published in Moré and Thuente [123] (reference implementation, double precision, IPX Sparcstation), MT = results of our (f2c'd) reimplementation of the line search of Moré and Thuente, WN = line search of Wright and Nocedal, CLS = (proposed) consensus line search, x = not able to find a suitable position within 20 iterations. Bold letters: CLS is superior to MT and WN.

to their highly non-polynomial numerical behavior combined with the start positions: The strictly polynomial interpolation and the update strategy tends to under/overestimate the steps. This, however, does not hold in general for molecular potential energy functions.

3.1.4 Results

As we describe in Section 2.3, the line search is only a part of a larger optimization package. Hence, to test the performance of our method we combined our line search approach with the search direction determining methods presented in Section 2.3.2.3:

(a) L-BFGS method [132] based on the Strang recurrences with two different parameter settings (number m of stored vector pairs).

(b) Improved version of L-BFGS proposed by Al-Baali [2] with two different parameter settings, where we used only the newest vector pair as update for the initial estimation[2].

(c) Shifted limited-memory variable metric method of Vlček and Lukšan [180] in the recommended variant 2 with the ratio of shift parameters as correction parameter.[3]

(d) Shanno derived method proposed in Watowitch et al. [187].

The combination of different parameter settings and optimization algorithms leads to total of six different optimization schemes that have been combined with the three considered line search schemes, our consensus line search as well as the two alternative approaches.

All minimization algorithms used the standard convergence criteria, i.e. convergence of a minimization run was assumed if either

$$\mathrm{RMS}(\nabla E(\mathbf{x})) < \epsilon$$

or

$$\frac{|E(\mathbf{x}_i) - E(\mathbf{x}_{i-1})|}{\max\{1, |E(\mathbf{x}_i)|\}} < \sqrt{\epsilon_{\mathrm{machine}}},$$

where we set $\epsilon = 10^{-5}$, $\epsilon_{\mathrm{machine}}$ is the double precision machine epsilon (on our machine $\sqrt{\epsilon_{\mathrm{machine}}} \approx 1.49 \cdot 10^{-8}$), and RMS denotes the root mean square. All line searches used the standard parameters that were shown to be optimal in the case of the L-BFGS method by Baysal, Meirovitch, and Navon [16], i.e. $\alpha = 10^{-4}$ in (2.3) and $\beta = 0.25$ in (2.4).

The tests that have been carried out simulated two different application scenarios: The optimization of experimentally derived molecular structures and the reconstruction of strongly perturbed small molecules.

[2]Baysal, Meirovitch, and Navon [16] reported that m is typically $3 \leq m \leq 7$, where taking $m > 7$ does not improve the performance. Thus, we tested both L-BFGS methods with the standard choice $m = 5$ and a choice out of this range, we tested $m = 8$.

[3]To the best of our knowledge, these methods have not yet been used on molecular potential energy functions. We set the number of columns of the iteratively updated matrix U to 10.

3.1.4.1 Optimization of Experimentally Derived Structures

The first scenario has been designed to reduce the number of test cases where different algorithms converge into different minima, the so-called multiple minima problem of molecular potential energy functions. Since the total energy of a molecule depends on atom-atom interactions, the number of possible low-energy configurations can grow exponentially with the number of atoms and has been estimated by Hoare [78] to be on the order of $\mathcal{O}(\exp(n^2))$ for an n-atom molecule. Thus, we restricted our test runs to molecules with rather small number of atoms in order to obtain comparable results. We searched the Protein Data Base (PDB) [18] for NMR derived structures with at most about 300 atoms that include all hydrogen positions. As the objective function, we chose the AMBER force field [189] and the Merck Molecular Force Field (MMFF94) [63, 64, 65, 66, 67]. To avoid artificial minima, we deactivated all cutoffs for nonbonded interactions and used the distance dependent dielectric constant. All other options were set to their default values.

Finally, we performed test runs with: (1) Chignolin[80] (PDB-ID: 1uao, 138 atoms), (2) solution structure of HP7[8] (PDB-ID: 2evq, 197 atoms), (3) structure of Methionine-Enkephalin in fast tumbling DMPC/DHPC bicelles[113] (PDB-ID: 1plw, 75 atoms), (4) solution structure of CM15 in DPC micelles[145] (PDB-ID: 2jmy, 282 atoms), and (5) Trp-Cage Miniprotein Construct TC5b[127] (PDB-ID: 1l2y, 304 atoms). Off the multiple models contained in the PDB files, we always chose the first one.

Table 3.2 presents the results of our test runs, i.e. the energy value (f) at the found minimum in kJ/mol, the number of function/gradient evaluations (eval), and the total time for each test run in seconds. Considering the total number of evaluations, our line search algorithm shows the best performance in 53 out of 60 test cases. Unfortunately, not all of these cases are comparable due to the multiple minima problem of molecular potential energy functions. To cope with this problem, we consider the results of two minimizers to be comparable if the all-atoms root mean square deviation of the respective structures is below a rather restrictive value of 0.5 Å. Otherwise, we assume that the minimizers converged to different minima. If two or more results in a box of Table 3.2 are not comparable, we present the values in this box in small italic letters. In summary, Table 3.2 contains 52 comparable test cases.

Table 3.2 shows that our consensus line search reduced the total number of function/-gradient evaluations down to 47.7% (down to 85% on average). Our algorithm led to the lowest number of evaluations in 46 of 52 comparable test examples. Note that the total time obviously correlates with the number of function/gradient evaluations and, hence, in these cases our algorithm also led to the lowest total time. The improvements in total time of our method are not affected by our decision to include all possible interpolation schemes. It is interesting to note that the Shanno derived method [187] (CG deriv.) showed competitive performance using the proposed line search.

The method of Wright and Nocedal [133] performed better than the approach of Moré and Thuente [123] in some cases while not being competitive on the artificial test problems in Table 3.1 as mentioned before. Considering the start points, the test examples are constructed with the aim that all polynomial interpolation schemes significantly under- or overestimate the real function in the first steps. In the case of our molecular test runs, the line search algorithms are provided with well scaled initial steps by the search direction determining algorithm and this problem vanishes.

			1uao			2evq			1plw			2jmy			1l2y	
MMFF94		*f*	eval	time	*f*	eval	time	*f*	eval	time	*f*	eval	time	*f*	eval	time
L-BFGS	MT	340.6	2096	31.93	72.6	2372	72.97	334.4	2009	9.31	1083.1	5022	315.00	*335.5*	*1751*	*127.75*
$m = 5$	WN	340.6	2013	30.63	72.7	2328	71.32	334.5	2018	9.41	1083.6	4753	298.24	*358.0*	*2065*	*155.50*
	CLS	340.6	1594	**24.26**	72.7	1978	**60.86**	334.4	1613	**7.48**	1083.4	4336	**271.78**	*358.0*	*1824*	*132.90*
L-BFGS	MT	340.6	1726	26.27	72.6	2231	68.46	334.4	2042	9.71	1083.0	5185	322.94	335.5	1696	124.55
$m = 8$	WN	340.6	1803	27.37	72.6	2288	70.21	334.5	1973	9.12	1083.3	4553	283.48	322.2	1457	105.89
	CLS	340.6	1686	**25.66**	72.7	2062	**63.33**	334.4	1945	**9.04**	1083.3	3910	**245.60**	335.5	1436	**105.00**
L-BFGS	MT	340.6	1665	25.39	73.3	2458	75.40	334.4	1916	8.97	1083.0	5308	338.27	335.5	1769	128.19
$m = 5$	WN	340.6	1743	26.46	72.7	2260	69.69	334.5	1951	8.99	1083.5	4779	298.07	335.5	1649	121.34
imp.	CLS	340.6	1609	**24.88**	72.7	1907	**58.75**	334.4	1852	**8.54**	1083.1	3846	**240.62**	335.5	1497	**108.46**
L-BFGS	MT	340.6	1675	25.97	72.6	2200	67.51	334.4	1904	8.97	1083.0	5190	325.20	*357.9*	*2034*	*148.59*
$m = 8$	WN	340.6	1773	27.01	72.7	2243	69.84	334.4	2010	9.31	1083.5	4698	293.01	*322.2*	*1643*	*119.81*
imp.	CLS	340.6	1684	26.03	72.7	2084	**64.39**	334.5	1808	**8.47**	1083.1	3881	**253.91**	*335.5*	*1483*	***108.78***
SLVMM	MT	340.6	2396	37.12	72.6	3273	100.36	*334.4*	*3136*	*14.66*	1083.1	6179	387.33	335.5	2271	164.19
VAR2	WN	340.6	3123	47.89	72.7	3602	110.10	*334.4*	*3680*	*17.15*	1083.2	7201	451.29	335.6	2569	187.81
	CLS	340.6	2228	**34.01**	72.7	2744	**84.19**	*234.5*	*3130*	***14.55***	1083.2	6076	**385.57**	322.3	1824	**132.55**
CG	MT	340.6	2966	46.11	72.7	3342	102.77	*206.8*	*2777*	*12.81*	1083.6	8030	500.90	335.5	2032	147.91
deriv.	WN	340.6	1879	28.82	72.6	2167	66.46	*334.6*	*3045*	*14.05*	1083.3	7951	495.01	322.3	1729	126.08
	CLS	340.7	1909	29.04	72.7	1907	**59.10**	*223.8*	*2300*	***10.62***	1083.3	3593	**228.43**	335.6	1425	**104.61**
AMBER																
L-BFGS	MT	-116.6	2504	3.95	-161.4	2418	6.97	-32.6	1737	1.01	*49.2*	*3180*	*17.17*	-100.5	2619	16.20
$m = 5$	WN	-116.6	2435	3.85	-161.2	1959	5.65	-32.6	1678	0.97	*49.5*	*3128*	*16.86*	-100.3	2134	13.25
	CLS	-123.6	2055	**3.27**	-161.1	1543	**4.47**	-32.6	1502	**0.88**	*42.5*	*2561*	***13.57***	-100.3	1834	**11.42**
L-BFGS	MT	-116.7	2371	3.77	*-161.8*	*3174*	*9.19*	-32.6	1695	0.99	49.2	3413	18.47	-100.5	2408	15.02
$m = 8$	WN	-116.6	2572	4.08	*-161.2*	*1851*	*5.35*	-32.6	1636	0.95	49.3	3123	16.87	-100.5	2297	14.31
	CLS	-116.6	2135	**3.40**	*-161.2*	*1765*	***5.09***	-32.6	1463	**0.90**	49.3	2685	**14.54**	-100.4	1955	**12.21**
L-BFGS	MT	-116.6	2515	3.98	*-161.8*	*3386*	*9.80*	-32.6	1699	0.99	49.2	3316	17.94	-100.5	2604	16.22
$m = 5$	WN	-116.6	2284	3.61	*-161.2*	*1808*	*5.27*	-32.6	1664	1.02	49.3	3136	16.95	-100.3	2150	13.38
imp.	CLS	-116.6	2419	3.84	*-161.2*	*1610*	***4.67***	-32.6	1581	**0.93**	49.2	2738	**14.85**	-100.4	2037	**12.68**
L-BFGS	MT	-116.7	2196	3.52	-161.8	3111	9.01	-32.6	1606	0.94	49.2	3534	19.16	-100.5	2415	15.01
$m = 8$	WN	-116.6	2514	4.01	-161.2	1956	5.65	-32.6	1618	0.99	49.2	3240	17.55	-100.4	2058	12.83
imp.	CLS	-116.6	2166	**3.46**	-161.2	1606	**4.70**	-32.6	1458	**0.86**	49.2	2827	**15.33**	-100.4	2105	13.13
SLVMM	MT	-116.7	3350	5.55	-161.2	2524	7.59	-35.8	2858	1.74	49.2	4303	23.78	-100.5	3407	21.65
VAR2	WN	-123.6	2934	4.78	-161.2	2990	8.79	-35.8	3311	2.00	49.3	5256	28.83	-100.3	3382	21.39
	CLS	-116.6	3448	5.60	-161.2	2416	**7.12**	-32.6	2444	**1.51**	49.5	3864	**21.42**	-100.3	2875	**18.25**
CG	MT	-123.6	2499	3.91	-161.1	2913	8.39	-32.6	2025	1.16	*49.5*	*4999*	*26.93*	-100.4	2892	17.85
deriv.	WN	-123.6	2520	3.92	-160.8	2771	7.96	-32.6	1862	1.08	*51.7*	*4320*	*23.22*	-100.5	2795	17.30
	CLS	-123.6	1468	**2.25**	-161.2	1846	**5.32**	-32.6	1996	1.18	*42.6*	*3201*	***17.21***	-100.5	2276	**14.13**

Table 3.2: Performance of different minimization test runs. f = value of the potential function at minimum in kJ/mol, eval = total number of function and gradient evaluations, time = total time in seconds, MT = line search of Moré and Thuente, WN = line search of Wright and Nocedal, CLS = (proposed) consensus line search. Small italic letters: not comparable results, see text. Bold letters: CLS is superior to MT and WN.

3.1.4.2 Reconstruction of Strongly Perturbed Molecules

For the reconstruction of strongly perturbed molecules, the second test scenario, we chose the test suite for the Merck Molecular Force Field (MMFF94) [63, 64, 65, 66, 67]. We randomized these structures by adding a random displacement up to 2 Å to every atom and optimized them using the Merck Molecular Force Field. We used electrostatics with constant dielectric since the 760 molecules of this test suite have originally been optimized (by a non-local monte carlo like approach using this setting). Since we employed a pure local minimization method, we cannot expect to reconstruct all molecules due to the large number of local minima. Nevertheless, all algorithms could reconstruct 542 molecules with deviation less than 0.5 Å. Moreover, for 654 test cases all algorithms converged to the same local minimum and, hence, these cases can be used for evaluating performance. The results are shown in Table 3.3, which presents the total number of function/gradient evaluations and the total time in seconds. Again, we used the six standard optimization techniques combined with our line search, the line search of More and Thuente, and the method of Write and Nocedal. In all cases our line search produced the lowest run times and the smallest number of function evaluations.

	MT		WN		CLS	
	eval	time	eval	time	eval	time
L-BFGS, $m = 5$	193,835	69.66	209,524	72.83	186,539	**65.20**
L-BFGS, $m = 8$	184,148	66.40	195,726	68.99	179,038	**63.68**
L-BFGS $m = 5$, imp.	186,925	66.37	200,564	70.58	183,492	**64.44**
L-BFGS $m = 8$, imp.	178,506	63.61	188,181	66.92	174,680	**60.91**
SLVMM VAR2	259,687	94.98	302,600	108.17	256,513	**92.05**
CG deriv.	364,323	131.88	429,100	147.96	273,418	**94.52**

Table 3.3: Performance of structure reconstruction runs. eval = total number of function and gradient evaluations to optimize all 654 comparable test examples, time = total time in seconds for all reconstructions, MT = line search of Moré and Thuente, WN = line search of Wright and Nocedal, CLS = (proposed) consensus line search. Bold letters: CLS is superior to MT and WN.

3.1.5 Conclusion

Our consensus approach allows the line search procedure to adapt locally to qualitatively different kinds of behavior of the objective function. This is particularly useful in computational chemistry where the molecular potential energies exhibit both polynomial and rational/exponential behavior. To guarantee high efficiency of the method the choice of the most reliable interpolant is made without additional function evaluations, thus, incurring only negligible computational overhead. We have tested our method in different scenarios: established, synthetic test problems from the line search literature, the optimization of experimentally derived structures, and the reconstruction of strongly perturbed molecules. Our test runs have shown that the consensus approach can significantly reduce the number of function/gradient evaluations and consequently the total run time. While the method of Moré and Thuente [123], in particular, is very powerful and lives up to its good reputation, our consensus approach performs even better in most cases. In addition, in the few cases where it did not yield the lowest run time the algorithm never performed significantly worse than the fastest choice. Replacing a standard line search method with the new algorithm reduced the total number

of function/gradient evaluations down to 47.7% (down to 85% on average). Our results, thus, show that the proposed algorithm is a promising alternative to standard line search algorithms.

In the next section, we incorporate our line search method into our novel approach for gradient-based local optimization of flexible ligands and, finally, in the subsequent sections, into our docking approach for flexible ligand-receptor docking.

3.2 Gradient-Based Local Optimization of Flexible Ligands

Ligand-receptor docking is a key task in the drug development process. Although there exist many different algorithms and programs, the problem is far from being solved. In general, the complexity of the task rises with increasing flexibility of the molecules [103]. The aim is to find an energetically favorable conformation of ligand and receptor in complex. The binding free energy is approximated by a force-field or knowledge based potential energy function. Hence, the task of finding the best conformation is an optimization problem: Find the best position on the energy landscape.

The conformational sampling necessitates the knowledge of the ligand's and receptor's atoms and some representation thereof. The obvious way to represent the spatial arrangement of a ligand molecule while sampling the conformational space is using the Cartesian coordinates of all its atoms as well as a set of bonds that defines their interconnectivity. The drawback of this method is the high number of degrees of freedom (DOFs), which frustrates many methods in molecular modeling when applied to larger structures, e.g. proteins.

Alternatively, a molecule may be represented by its translation, orientation, and a set of torsional angles, bond angles, and bond lengths resulting in a similar number of DOFs. This representation has the advantage that nonrelevant degrees of freedom can be frozen. For several applications in molecular modeling, like docking, it is reasonable to constrain molecular flexibility by fixing bond angles and bond lengths and restricting torsional flexibility to rotation around single bonds that connect rigid entities, like ring structures [15, 47, 124]. In this section, we use this compact representation where the number of degrees of freedom is significantly reduced while it is possible to rapidly convert from the compact to the Cartesian representation. However, the efficient local optimization techniques described in Section 2.3 require the computation of derivatives. Furthermore, they are sensitive to singularities, like a loss of DOFs, or to nonminimal parametrizations. Therefore, the main challenge with respect to the compact molecular representation is to find an orientational parametrization that is well suited for gradient optimization.

In Cartesian space, the orientation of a body can be represented in different ways. Using the well-known Euler angle representation resulting in a straightforward calculation of the gradient has the disadvantage of the so called gimbal lock phenomenon [188], that is the loss of DOFs. Generally speaking, this means that in the course of an optimization, a molecule can reach positions in which not all possible further alterations of its orientation are achievable by variations of the three Euler angles. Thus, the unit quaternion, which avoids gimbal lock singularities, has become a quasi standard for orientations [95]. However, because the unit quaternion space is only a subset of $\mathbb{R}^4$ with three DOFs, direct optimization of the four interdependent unit quaternion values is awkward [159].

In this section, we present a novel method [54] for energy minimization of molecular

complexes with special attention to gradient-based optimization of the molecules orientation. We use the exponential mapping [62], which transforms a vector in $\mathbb{R}^3$ to the unit quaternion space. Although this representation is not free of singularities, they can be avoided with only negligible additional computational cost allowing for the application of most gradient-based local optimization algorithms. We demonstrate that an L-BFGS approach is, in our modified version, optimally suited and highly efficient for the given task. Furthermore, we show that negligible straightforward modifications of a continuous but not differentiable function are sufficient to apply our optimization algorithm successfully. To compare the gradient-based optimization of the exponential map representation to the method of Solis and Wets [164] that operates directly on the four quaternion values, we minimized the energy of seven ligands with increasing complexity in the presence of their corresponding receptors. As an example for energy functions without a continuous derivative, we used the piecewise linear Gehlhaar scoring function [56], which was repeatedly employed for molecular docking [179, 81, 170]. Our novel method shows better performance in all test cases. Especially the energetically more favorable conformations – found already after a few iterations – render the new approach a valuable tool for molecular optimization.

3.2.1 Molecular Representation

The conformation and position of a molecule in space is uniquely defined by the Cartesian coordinates of its atoms. Often the complete molecular flexibility is abandoned in favor of a reduced set of parameters that is required for representing a molecule. Like many other applications [124, 88, 173] we use translation, orientation, and a set of flexible torsional angles that connect rigid compounds. Thus, we need three real values for the translation (t_x, t_y, t_z), one real value for each flexible torsional angle $(\phi_1, \ldots, \phi_n)$, and a unit quaternion composed of four real values (q_1, q_2, q_3, q_4) for the molecules orientation. A parameter vector $x = (t_x, t_y, t_z, q_1, q_2, q_3, q_4, \phi_1, \ldots, \phi_n)$ is converted into a molecular conformation by a series of transformations. In the first step, all flexible torsional angles are processed. Because in our case such a rotatable bond is guaranteed not to be part of a ring, it divides the molecule in two substructures. The part containing fewer atoms is rotated while the other one remains stationary, see Figure 3.8. This procedure is applied to all flexible torsional angles. In the next step, the whole molecule is rotated. To this end, the origin is defined by the average position of all atoms that were not rotated in the first step thus defining a form of molecular centroid. In other implementations the rotation origin is intuitively placed onto the geometric center of the ligand, but this method complicates the computation of the gradient. In the last step, the molecule is moved according to the three translational parameters.

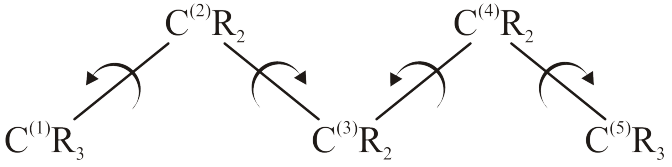

Figure 3.8: Example for rotatable bonds and molecular centroid with R being arbitrary heavy atoms. If bond $C^{(1)}$ - $C^{(2)}$ is rotated, only atoms R connected to $C^{(1)}$ are moved. If bond $C^{(2)}$ - $C^{(3)}$ is rotated, $C^{(1)}$ and atoms R connected to $C^{(1)}$ and $C^{(2)}$ are moved. Due to symmetry, the same holds true for the other two bonds with other indices. This means $C^{(2)}$, $C^{(3)}$, and $C^{(4)}$ are never moved and, hence, define the molecular centroid.

3.2.2 Gehlhaar Scoring Function

A scoring function in ligand-receptor docking is expected to meet multiple requirements. In the first place the scoring function should allow to differentiate native binding poses from decoy structures. Secondly, the score should approximate the binding free energy. Furthermore, it ought to be efficiently computable. We chose the Gehlhaar function [56] mainly for the following reasons: ease of implementation, sufficient correlation of the root mean square deviation (RMSD), less frustrated energy landscape compared to other scoring functions, and finally as a test case for an inherently not continuously differentiable function. It must be noted that the Gehlhaar score cannot be used to estimate the binding free energy. The formula for the score E is composed of one bonded term for the torsional potential E_{tor} and one nonbonded term E_{pair} for a kind of van der Waals interaction

$$E = E_{\text{tor}} + E_{\text{pair}}.$$

For the computation of E_{pair}, the Gehlhaar scoring function distinguishes only four atom types: nonpolar, hydrogen-bond-donor, hydrogen-bond-acceptor, and both-acceptor-and--donor. The interaction between any of these atom types results in two types of non-bonded interaction, namely steric and hydrogen bond contributions, see Table 3.4.

Atom type	Donor	Acceptor	Both	Nonpolar
Donor	Steric	HB	HB	Steric
Acceptor	HB	Steric	HB	Steric
Both	HB	HB	HB	Steric
Nonpolar	Steric	Steric	Steric	Steric

Table 3.4: Atom Types for Nonbonded Interactions

	A	B	C	D	E	F
Steric	3.4 Å	3.6 Å	4.5 Å	5.5 Å	-0.4	20.0
HB	2.3 Å	2.6 Å	3.1 Å	3.4 Å	-2.0	20.0

Table 3.5: Parameter Set for Nonbonded Steric and Hydrogen-Bonding Potentials.

Both interaction types are calculated by an interval piecewise linear function f of the pairwise atom distance d_{ij} of atoms i and j, with each type having different function parameters, see Table 3.5 and Figure 3.9,

$$E_{\text{pair}} = \sum_{i \neq j} f(d_{ij}).$$

This function is obviously not continuously differentiable. Thus, we added a quadratic transition function in an interval of 0.02 Å length at each junction of the original linear segments, see Figure 2. These functions are uniquely defined by their interpolation conditions.

The term for the torsional energy E_{tor} is similar to that of other scoring functions, but restricted to sp^3 – sp^3 and sp^2 – sp^3 bonds:

$$E_{\text{tor}} = A \cdot (1 + \cos(n \cdot \phi - \phi_0))$$

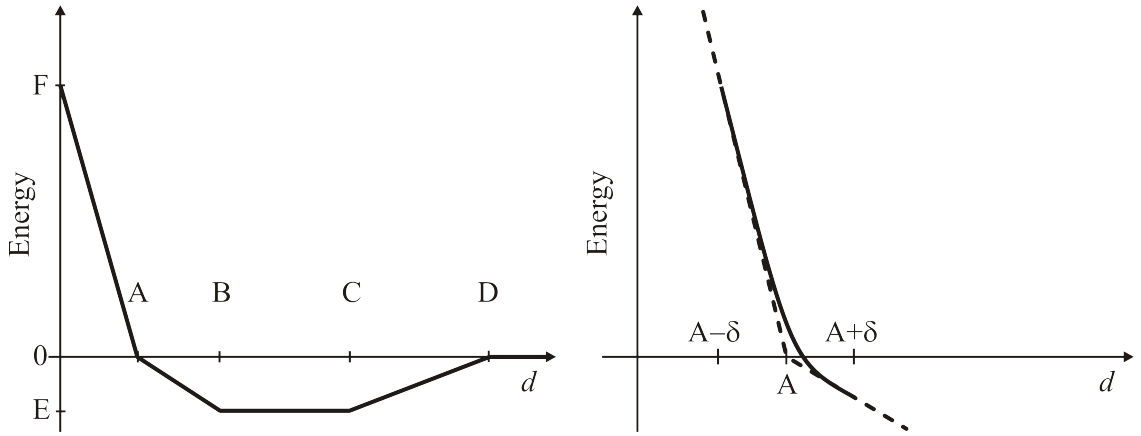

Figure 3.9: The left picture shows the original piecewise linear pairwise potential function used for nonbonded interactions. The right picture illustrates the modifications (solid line) applied to the original function (dashed line) to produce a continuously differentiable function.

with $A = 3.0$, $n = 3$, $\phi_0 = \pi$ for sp^3 – sp^3 bonds, and $A = 1.5$, $n = 6$, $\phi_0 = 0$ for sp^2 – sp^3 bonds. The original Gehlhaar function provides a separate energy term for the internal nonbonded interaction of the ligand by assigning a penalty of 10^4 if two ligand atoms that do not share a bond come closer than 2.35 Å. This kind of energy calculation is entirely unsuited for the computation of a gradient, because it is highly noncontinuous. To circumvent this problem, we use the same term for internal ligand-ligand interactions as for ligand-receptor interactions.

3.2.3 Gradient Computation and Exponential Mapping

The application of a gradient-based optimizer requires the derivatives of the underlying energy or scoring function E with respect to the parameter vector $\mathbf{x}$. The Gehlhaar function [56] consists of a pairwise term E_{pair} and a torsional term E_{tor}. Thus, the gradient $\mathbf{g}$ is given by

$$\mathbf{g} := \frac{\partial E}{\partial \mathbf{x}} = \frac{\partial E_{\text{pair}}}{\partial \mathbf{x}} + \frac{\partial E_{\text{tor}}}{\partial \mathbf{x}}.$$

The gradient of E_{tor} can be easily computed and affects only torsional parameters $\phi_1, \ldots, \phi_n$

$$\frac{\partial E_{\text{tor}}}{\partial \phi} = \frac{\partial}{\partial \phi}\left(A \cdot (1 + \cos(n \cdot \phi - \phi_0))\right) = -n \cdot A \cdot \sin(n \cdot \phi - \phi_0).$$

To calculate the derivatives for the pairwise interactions E_{pair}, we first compute the gradient $\mathbf{g}_i$ for each atom i. This is the sum of all derivatives of pairwise interactions that an atom participates in with $\mathbf{v}_i$ being the position vector of atom i and $\mathbf{v}_j$ being the position vector of the interacting atom j

$$\mathbf{g}_i = \sum_{i \neq j} f'(d_{ij}) \frac{\mathbf{v}_i - \mathbf{v}_j}{\|\mathbf{v}_i - \mathbf{v}_j\|}.$$

Mapping the gradient $\mathbf{g}_i$ of an atom i with position $\mathbf{v}_i$ to an arbitrary parameter r requires the derivative of $\mathbf{v}_i$ with respect to r. $\partial \mathbf{v}_i$ represents the tangential movement of atom i when

r varies by an infinitesimal amount and can now be used to calculate the derivative of E_{pair} with respect to r

$$\frac{\partial E_{\text{pair}}}{\partial r} = \sum_i \left(\frac{\partial \mathbf{v}_i}{\partial r}\right)^T \mathbf{g}_i. \tag{3.6}$$

In the following, we will use Equation (3.6) to calculate the derivatives of E with respect to specific parameters.

3.2.3.1 Translational Gradient

Calculating $\partial \mathbf{v}_i$ with respect to a translational parameter t is straight forward because any change in t translates $\mathbf{v}_i$ linearly. Thus, for any translational parameter t, Equation (3.6) can be reduced to

$$\begin{aligned}
\frac{\partial E_{\text{pair}}}{\partial t_x} &= (1,0,0) \cdot \sum_i \mathbf{g}_i, \\
\frac{\partial E_{\text{pair}}}{\partial t_y} &= (0,1,0) \cdot \sum_i \mathbf{g}_i, \\
\frac{\partial E_{\text{pair}}}{\partial t_z} &= (0,0,1) \cdot \sum_i \mathbf{g}_i.
\end{aligned}$$

3.2.3.2 Torsional Gradient

The rapid computation of the torsional gradient has been the subject of numerous scientific studies [1]. If atoms k and j are connected by a rotatable bond and atom i is moved by rotating this bond, see Figure (3.10a), the derivative of $\mathbf{v}_i$ with respect to a torsional parameter ϕ_{jk} can be calculated by

$$\frac{\partial \mathbf{v}_i}{\partial \phi_{jk}} = (\mathbf{v}_k - \mathbf{v}_j) \times (\mathbf{v}_i - \mathbf{v}_j). \tag{3.7}$$

Inserting (3.7) in (3.6) yields

$$\frac{\partial E_{\text{pair}}}{\partial \phi_{jk}} = \sum_i \left((\mathbf{v}_k - \mathbf{v}_j) \times (\mathbf{v}_i - \mathbf{v}_j)\right)^T \mathbf{g}_i.$$

3.2.3.3 Orientational Gradient

The most challenging part is the computation of the orientational gradient, because, up to now, there is no minimal representation that does not inherit some kind of singularity, e.g. loss of DOFs. Representing the orientation (three DOFs) by a unit quaternion does not include such a singularity, but the independent optimization of its four values is undesirable [159]. This is caused by the unit quaternions representing only a subset of the entire four-dimensional quaternion space. To alleviate this problem, we use exponential mapping [62] to map a point $\mathbf{p} = (p_1, p_2, p_3)$ from parameter space $\mathbb{R}^3$ to q in the unit quaternion space S^3:

$$q = (q_1, q_2, q_3, q_4) = \begin{cases} (0,0,0,1) & \text{if } \mathbf{p} = (0,0,0) \\ \left(\sin(\frac{1}{2}\|\mathbf{p}\|)\frac{\mathbf{p}}{\|\mathbf{p}\|}, \cos(\frac{1}{2}\|\mathbf{p}\|)\right) & \text{otherwise} \end{cases}.$$

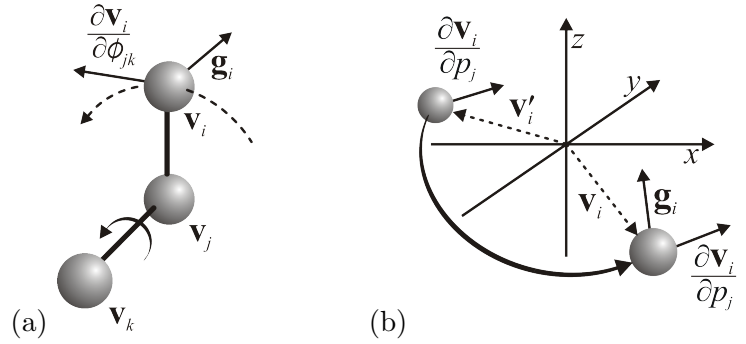

Figure 3.10: Mapping of nonbonded gradient to (a) torsional and (b) orientational parameter. Please note that in Figure (b), $\frac{\partial \mathbf{v}_i}{\partial p_j}$ is first calculated using $\mathbf{v}'_i$ and then translated to $\mathbf{v}_i$.

This enables us to compute the derivative $\mathbf{T}_j$ of the corresponding rotation matrix $\mathbf{R}$,

$$\mathbf{T}_j = \frac{\partial \mathbf{R}}{\partial p_j},$$

for each of the three orientation parameters p_j [62] that can now be used to calculate the gradient $\frac{\partial E_{\text{pair}}}{\partial p_j}$. For a detailed description of the computation of $\mathbf{T}_j$, please refer to the publication by Grassia [62]. For each evaluation of the objective function, the orientational parameter $\mathbf{p}$ is mapped to a unit quaternion q, which is then converted to a rotation matrix $\mathbf{R}$ that defines the molecular orientation

$$\mathbf{R} = \begin{pmatrix} q_1^2 + q_2^2 - q_3^2 - q_4^2 & 2(q_2q_3 - q_1q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_4 + q_2q_3) & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2(q_3q_4 - q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_1q_2 + q_3q_4) & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{pmatrix}.$$

Let $\mathbf{v}'_i$ be the position of an arbitrary atom i and $\mathbf{v}_i$ the position of the atom after the rotation by $\mathbf{R}$, see Figure 3.10b),

$$\mathbf{v}_i = \mathbf{R}\mathbf{v}'_i.$$

Then we have

$$\frac{\partial \mathbf{v}_i}{\partial p_j} = \mathbf{T}_j \mathbf{v}'_i.$$

Again inserting into Equation (3.6) yields

$$\frac{\partial E_{\text{pair}}}{\partial p_j} = \sum_i (\mathbf{T}_j \mathbf{v}'_i)^T \mathbf{g}_i, \; j = 1, 2, 3.$$

As mentioned earlier, no method for a minimal parametrization of the orientation is free of singularities. This also holds for exponential mapping, where singularities arise if the length of the orientational parameter vector $\mathbf{p}$ approaches 2π. All parameter vectors $\mathbf{p}$ with $\|\mathbf{p}\| = n \cdot 2\pi$, $n \in \mathbb{Z}_{>0}$ are mapped to the quaternion $q = (0, 0, 0, -1)$. For these parameter vectors, all gradients $\frac{\partial \mathbf{v}_i}{\partial p_j}$ point into the same direction, reducing the number of DOFs to one. Fortunately, all possible orientations can be denoted by parameters within a shell of π around the origin in $\mathbb{R}^3$. Thus, we only need to take care that the optimization algorithm stays within this shell.

3.2.4 Gradient-Based Local Optimization

In Section 2.3.2, we presented current up-to-date local optimization methods. In this study, we chose the limited memory variant of the BFGS quasi-Newton method (L-BFGS) based on the Strang recurrences described in Section 2.3.2.3. This line search based approach can be easily extended to fulfill our rotational condition. In fact, our line search in Section 3.1 is a two stage bracketing approach and, thus, ideally suited to introduce an upper bound on the step size. If the bracketing interval is newly determined, the update procedure just has to ensure to stay within the safe region. Let $\mathbf{x}_k^{\text{ori}}$ be the orientational parameters of the current position $\mathbf{x}_k$ on E and $\mathbf{d}_k^{\text{ori}}$ the corresponding orientational parameters of the current search direction $\mathbf{d}_k$, then the approximative line search problem

$$\lambda_k := \arg\min_{\lambda>0} E(\mathbf{x}_k + \lambda \mathbf{d}_k)$$

is restricted to

$$\|\mathbf{x}_k^{\text{ori}} + \lambda_k \mathbf{d}_k^{\text{ori}}\| \leq \frac{3}{2}\pi.$$

This ensures that we always stay in a region far away from the singularity 2π. After the line search has finished, in a usual quasi-Newton method the current iteration would terminate yielding $\hat{\mathbf{x}}_k := \mathbf{x}_k + \lambda_k \mathbf{d}_k$ as the next iterate $\mathbf{x}_{k+1}$. In our case, however, we have to avoid the orientational singularity. Our modified line search ensures that we are in a region where the orientational derivatives are usable but possibly tending to the singularity. Thus, we reparameterize the orientational part $\hat{\mathbf{x}}_k^{\text{ori}}$ of $\hat{\mathbf{x}}_k$ if

$$\|\hat{\mathbf{x}}_k^{\text{ori}}\| \geq \pi$$

via replacing $\hat{\mathbf{x}}_k^{\text{ori}}$ by[4]

$$\left(1 - \frac{2\pi}{\|\hat{\mathbf{x}}_k^{\text{ori}}\|}\right) \hat{\mathbf{x}}_k^{\text{ori}}.$$

Because this is an equivalent orientation [62] (but with better derivatives), this replacement is a reparametrization from a geometric point of view. From a mathematical point of view, this is a well-defined jump on E. In theory, the Euler angle representation could also be dynamically reparameterized to avoid the gimbal lock phenomenon. However, in our case the reparametrization means only to scale the orientational parameters, whereas the similar operation on Euler angles implies a sequence of inverse trigonometric functions [163] to determine the new parameters. Finally, the current iteration finishes by returning $\hat{\mathbf{x}}_k$ as the next iterate $\mathbf{x}_k$.

3.2.5 Solis and Wets Optimization Method

We compared our method to the approach of Solis and Wets [164], which is widely used for molecular docking purposes. This local search method is a stochastic heuristic for continuous parameter spaces. Its primal purpose is the optimization of functions that do not provide gradient information, e.g. the AUTODOCK scoring function [124]. For our comparison, we closely followed the version of AUTODOCK 3.1 with the only alterations being due to adjustments to the BALL [101, 102, 76] environment. The basic algorithm starts with a random search step

[4] We choose π because the derivatives in this region are excellent, while all possible orientations are still representable.

and generally follows this direction with random movements as long as the objective function keeps improving. Continued improvements lead to an expansion of the random search steps, whereas continued failing narrows the search. The algorithm iterates until either a maximum number of function evaluations is reached or convergence is established by the random step width falling below a certain threshold value.

3.2.6 Comparison of Optimization Methods

The entire code that was used in this work for the modeling of molecules, scoring functions, etc. was generated using the BALL library [101, 102, 76].

To test our method on different molecules with varying complexity, we selected seven different ligand-receptor complexes from the Protein Data Base (PDB) [18]. The set of ligands consisted of simple as well as of more complex molecules, see Table 3.6.

				Our method		Solis and Wets	
PDB ID	Ref.	$n_{\text{rb}}/n_{\text{ha}}$	E_{initial}	E_{final}	n_{ev}	E_{final}	n_{ev}
1FDS	[23]	0/20	232.5	-50.4	9.80	-12.35	39.41
1FMO	[126]	2/19	295.7	-56.62	21.89	0.46	46.63
2MCP	[136]	3/11	199.2	-30.8	16.77	-11.76	29.06
1DWD	[14]	8/37	714.13	-68.86	34.06	88.48	48.40
1HPV	[99]	9/35	627.3	-75.13	35.97	107.57	69.86
2R04	[13]	10/25	770.66	-19.8	62.50	230.70	51.37
1HTF	[87]	12/41	693.27	-65.93	38.89	117.36	58.17

Table 3.6: Comparison of our method to Solis and Wets in terms of average initial E_{initial} and final score E_{final} and average number of function evaluations n_{ev}; the number of rotatable bonds = n_{rb}; number of heavy atoms = n_{ha}.

To produce random starting positions, we uniformly randomized each ligands orientation and conformation as well as its translation within a cube of edge length 6 Å focused on the geometric center of the ligands reported binding site. Then, we used both methods to optimize each prepared ligand (start conformation) 500 times. We recorded for each minimization the best score and the number of function evaluations required to attain a score of 1.0 worse than the best score. This threshold value was selected to account for the inherently different stopping criteria of both methods. For the method of Solis and Wets we partially retained the stopping criterion of the autodock implementation, that is the falling below a defined step width while we dismissed the maximum number of iterations. This means that, other than in AUTODOCK, the approach of Solis and Wets is always allowed to explore the scoring function to the local minimum. Our method used the following standard stopping criterion [192], i.e., convergence is assumed if in iteration k

$$\|\mathbf{g}(\mathbf{x}_k)\| < \varepsilon_{\mathbf{g}}(1 + |E(\mathbf{x}_k)|)$$

holds[5] for $\varepsilon_{\mathbf{g}} = 10^{-6}$. However, the compact representation of the molecules has the side

[5]Note that this criterion is numerically motivated and considers typical terms of molecular potential energy functions. It takes the combined calculation of function value and gradient into account. If E evaluates to a value different from zero, it is very likely that numerical instabilities prevent gradient values close to zero even if we are close to a local optimum.

effect, especially with increasing flexibility, that small changes of some parameters cause large alterations of the energy value. This effect is still intensified for the gradient and might lead to many iterations around the position of interest without substantially improving the molecular structure. Hence, we used a second stopping criterion, derived from the idea behind the criterion of the method of Solis and Wets combined with an energy criterion that cannot be satisfied if we are too far away from the minimum: Convergence is also assumed if the steps fall below a relative step size and the energy values do not significantly alter any more, i.e. if

$$\|\mathbf{x}_k - \mathbf{x}_{k-1}\| < \sqrt{\varepsilon_E}\frac{(1 + \|\mathbf{x}_k\|)}{100}$$

and

$$|E(\mathbf{x}_k) - E(\mathbf{x}_{k-1})| < \varepsilon_E(1 + |E(\mathbf{x}_k)|)$$

hold for $\varepsilon_E = 10^{-8}$.

3.2.7 Results

Table 3.6 shows the average Gehlhaar-score E_{final} of 500 minimization runs together with the average number of evaluations n_{ev} required to reach a function value at most 1.0 worse than the final score. The number of rotatable bonds corresponds roughly to the complexity of the optimization problem while the average energy before optimization indicates that generally the ligand has multiple van der Waals clashes at the random initial position. The results show that, on average, the score of our method is well below 0 for all ligands. This means that it generally resolves all van der Waals clashes and moves the molecule in a way that it is able to form multiple interactions. Even for more complex ligands, representing more difficult optimization problems, the average score does not deteriorate and seems to be roughly corresponding to the number of heavy atoms. As expected, more complex ligands require more function evaluations to reach the local minimum. In contrast to that, the method of Solis and Wets is able to resolve van der Waals clashes only for simple ligands with both average score and average number of function evaluations being considerably worse compared to our method. As ligands get more complex, the approach of Solis and Wets fails to resolve van der Waals clashes and the scores decline considerably.

There seems to be one outlier, 2R04, for which the results are worse then expected. This is caused by the particular morphology of the binding pocket, which forms a longish tube inside the receptor and is located relatively near to the receptor surface. Thus, the likewise elongated ligand can be trapped with one part being situated in the binding pocket and the other outside the receptor while the center is penetrating the protein producing multiple van der Waals clashes. Figure 3.11 illustrates the performance difference of both methods and the non-deterministic character of the approach of Solis and Wets for 1DWD. In this case, all minimization runs started from the same initial position. Our method always converged to a score of -108.49 (solid line) while the best result out of 100 Solis and Wets minimizations was -69.50 (dashed line). On average, the approach of Solis and Wets reached a value of 76.74 (the dotted line shows a typical minimization). The method of Solis and Wets required 149 function evaluations to produce its best results, a value that was reached by our method with only 17 function evaluations.

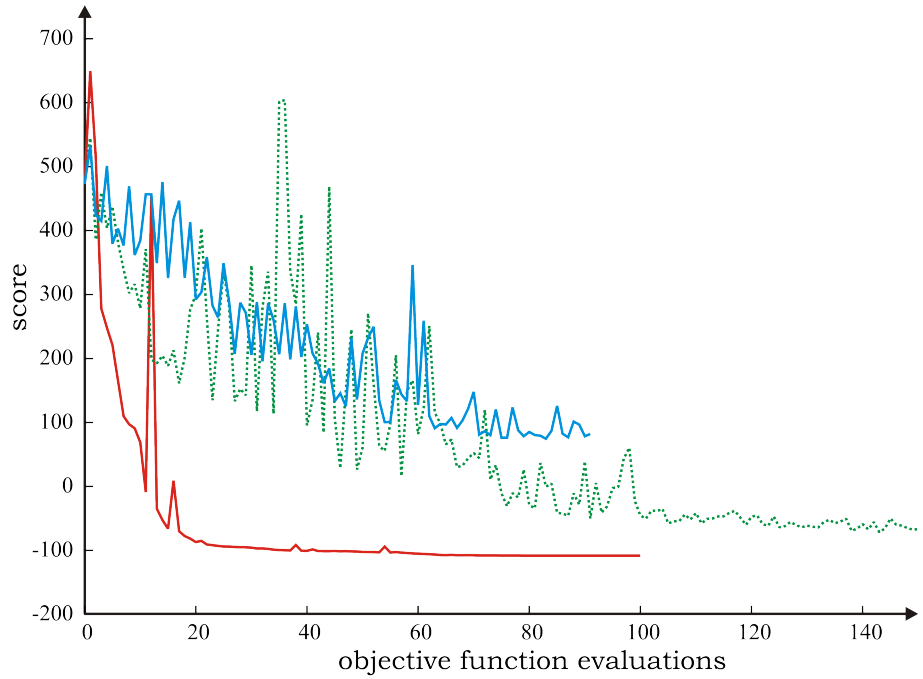

Figure 3.11: Comparison of one deterministic minimization of our method (red solid line) to two different minimizations of Solis and Wets from the same initial position (PDB ID 1DWD). The green dashed line is the best result of the approach of Solis and Wets out of 100 minimizations.

3.2.8 Conclusion

When performing docking calculations, the atom positions of the ligand are often described using a compact representation, which allows for reducing the DOFs, e.g., by fixing bond lengths. In some cases, the energy or scoring function is continuous and differentiable, which is a prerequisite for the usage of gradient-based algorithms for structure optimization. Despite the advantages of these algorithms, stochastic methods like the approach by Solis and Wets are employed, because the computation of the orientational gradient is difficult. Here, we demonstrated how to solve this issue by using the exponential map to transform a vector in $\mathbb{R}^3$ to the unit quaternion space and by avoiding the arising singularities. Thus, gradient-based optimization of molecules represented by translation, orientation, and torsional angles is possible employing any continuously differentiable scoring function.

However, our approach is not confined to continuously differentiable objective functions. For energy or scoring functions, which are continuous and not differentiable at a finite number of points, the gradient may be computed by introducing a patch function at the junction of two adjacent continuously differentiable segments of the original objective function. In general, a cubic patch function should meet the requirements to adequately interpolate between the two segments. An example for such a piecewise linear potential function is the Gehlhaar scoring function, which was repeatedly employed in protein-ligand docking studies [56, 179, 81, 170]. For this function, a quadratic patch function was sufficient for a smooth transition between adjacent linear portions.

Our approach outperformed the widely employed stochastic Solis and Wets algorithm even

for relatively simple optimization problems with few DOFs. The difference in performance became even more substantial with increasing complexity of the optimization problem. For molecules with many rotatable bonds, the approach of Solis and Wets is not able to determine search directions, which are as promising as the directions calculated by the gradient-based approach. The repeated failing to improve the score leads to permanently decreasing step sizes and finally to the abortion of the optimization possibly without having located a minimum in the energy hypersurface.

In contrast, the internal search direction calculation of our approach is very efficient due to the use of the exponential map and the adoption of the L-BFGS method. The only noticeable additional costs concern the calculation of the gradient of the scoring function. However, in the case of our tested function, the evaluation of the energy value can be extended to compute the gradient efficiently. These additional operations appear to be negligible in comparison to the high number of exploratory energy evaluations of the approach of Solis and Wets. Finally, because of the use of derivative information the proposed method reaches usually significantly deeper energy values in fewer steps than the Solis and Wets algorithm.

3.3 Extension to Ligand-Receptor Docking

In the previous section, we presented details of our method for the local optimization of flexible ligands using our compact representation. Here, we extend this part to a full docking approach for ligand-receptor docking where the ligand is fully flexible with respect to torsional angles and the receptor is still kept rigid. We integrate our approach into a Lamarckian genetic algorithm and combine this method with a multi-deme genetic algorithm [55]. Using the Astex diverse set, we show that our approach leads to substantially higher docking precision and shorter running times in comparison to genetic algorithms using conventional non-gradient local optimization. This difference in performance rises with increasing complexity (higher flexibility) of the ligands. In summary, the results indicate that our method is superior to other Lamarckian genetic algorithms, which employ the commonly used stochastic algorithm of Solis and Wets for local optimization.

3.3.1 Implemented Search Heuristics

In this section, we describe briefly our approach and the algorithms and techniques to which we compared our new method and give the parameter settings we tested. Details can be found in Fuhrmann [53].

A genetic algorithm [79] (GA) closely follows the principles of Darwinian evolutionary theory, in particular, natural selection and reproduction. It applies a set of genetic operations to a population of solution candidates (individuals) of an optimization problem, iteratively producing better results. The algorithm starts by creating an initial random population of fixed size. Afterwards, the objective function is evaluated for each individual to calculate its fitness score. Individuals with the worst scores are discarded, while the remaining solution candidates may create progeny. Mutations modify existing individuals, new individuals are produced by mating. However, mutations are not applied to several top ranked individuals, which is called *elitism*. When the population has grown again to the fixed size, this procedure of discarding, creating progeny, mutation, and mating is iterated until a convergence criterion is met. Genetic algorithms are widely used for ligand-receptor docking [135, 90, 28, 168].

Variants of the genetic algorithm are used in well-known docking suites like AUTODOCK [124, 82] and FITTED [31].

Two general modifications are known to influence the behavior of a genetic algorithm. In the first variant, the so called Lamarckian genetic algorithm (LGA) [72], a dedicated local search procedure is applied to improve the fitness of existing individuals. This variant has been popularized for ligand-receptor docking by AUTODOCK [124]. In the second variant, the so called multi-deme genetic algorithm (MDGA) [26], two or more populations evolve independently, while there is a limited migration between them, increasing the genetic diversity of a population and in turn facilitating breaking out of local minima. This approach has been employed successfully for the superimposition of flexible molecules [89] and flexible ligand-receptor docking in the program GOLD [88]. Finally, combining both modifications leads to the multi-deme Lamarckian genetic algorithm (MDLGA), which forms the basis of our hybrid method.

3.3.1.1 The Lamarckian Genetic Algorithm

In order to compare our novel approach to a genetic algorithm similar to the one used in the well-known docking suite AUTODOCK [124, 82], we implemented a Lamarckian genetic algorithm (LGA) employing the local search method of Solis and Wets [164], see the previous section, to which we compared our local search procedure. Our implementation closely follows the version of AUTODOCK 3.1, making adaptations where necessary due to using the C++ environment of BALL [101, 102, 76].

In each iteration the fitness of all individuals was assessed and only the top 50% of the individuals were allowed to create progeny to replenish the genetic pool, while the other 50% of the population were discarded. Selection of the survivors for mating was based on the rank-order of the individuals. The probability for random mutation was the same for all individuals except for the fittest individual, which was protected from mutation. Finally, a randomly chosen individual, which was not subject to local optimization in previous iterations, was optimized using the method of Solis and Wets.

3.3.1.2 The Multi-Deme Lamarckian Genetic Algorithm

To ensure that our approach is not only superior because we use a multi-deme genetic algorithm variant in contrast to the above mentioned standard variant, we also expanded the ordinary LGA to a multi-deme Lamarckian genetic algorithm (MDLGA) using the same local search method of Solis and Wets [164]. As for the ordinary LGA, we adapted the population size for best results. The MDLGA uses five island populations, which evolve similar to the single populations in the LGA. In addition, starting after 20 iterations we allow for migration to occur every 4 iterations. For the migration operations, the populations are rank-ordered according to the fitness of their best individuals. Migration is allowed only if the least fittest population has converged, that is if the difference between the scores of the fittest and the least fittest individuals is below a threshold value of 0.1. Migration is then performed by replacing the least fittest individual with the information of the fittest individual from the fittest population.

3.3.1.3 The New Hybrid Method

Our new hybrid method follows the tradition of evolutionary algorithms for molecular docking purposes. It consists of a MDLGA and our numerical gradient-based local optimization approach, which is called by the MDLGA for (re)optimization of the calculated conformations. Figure 3.12 shows the basic schedule of our hybrid search heuristic.

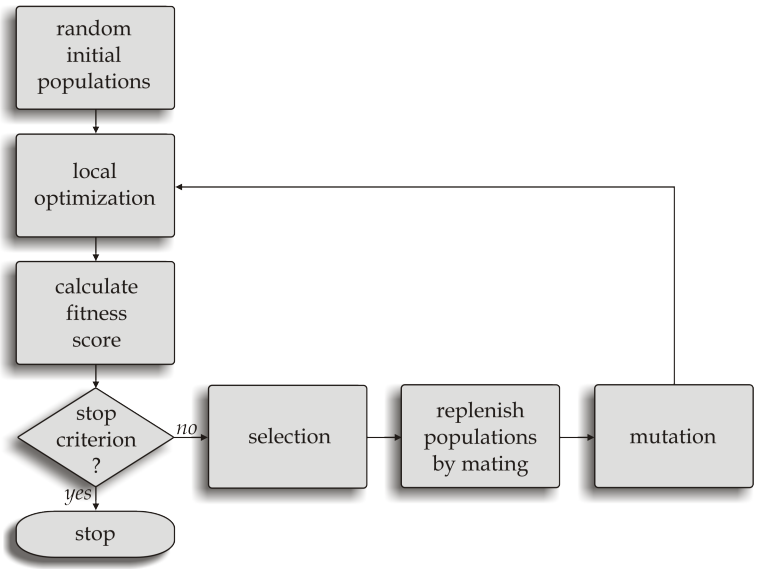

Figure 3.12: Flowchart of our hybrid search heuristic. The algorithm is based on a multi-deme Lamarckian genetic algorithm and employs our novel gradient-based optimization method for the local optimization of individuals.

3.3.2 Comparison of Search Heuristics

To compare the performance of our new hybrid method to the non-gradient-based search heuristics, we chose the Astex [73] diverse set for flexible ligand-receptor docking. This test set consists of 85 high resolution structures of protein-ligand complexes with all ligands featuring drug-like properties.

We implemented four different MDLGAs with gradient based optimization: smallOne and largeOne optimize one individual per iteration and population, whereas smallAll and largeAll optimize all individuals. Furthermore, smallOne and smallAll have smaller populations than largeOne and largeAll. The maximum number of iterations for our local search was set to 50. Table 3.7 lists the parameters used in this study for all tested genetic algorithm.

For each optimization method and for each ligand of the Astex data set, we performed 300 docking runs. For this purpose, we defined a translation box with an edge length of 10 Å centered on the reported binding site. Before each docking run, the ligands position and orientation was randomized. To compare the running time, we applied the same stopping criterion for all methods: The algorithm stopped when the best score had not improved for a given number of function evaluations. This maximum number of function evaluations per run was calculated as a linear function of the number of rotatable bonds in the ligand being docked and was between 3000 and 5000.

search heuristic	number of populations	initial population	population size	survivors	elitism	mutation rate	individuals optimized
LGA	1	100	200	100	1	0.05	1
MDLGA	5	20	40	20	1	0.05	1
smallOne	5	5	10	5	1	0.05	1
smallAll	5	5	10	5	1	0.05	5
largeOne	5	10	20	10	1	0.05	1
largeAll	5	10	20	10	1	0.05	5

Table 3.7: Parameters for the genetic algorithms implemented in this study. LGA: Lamarckian genetic algorithm using the Solis and Wets minimizer; MDLGA: LGA using multiple populations; smallOne/largeOne: MDLGA with small/large population size using gradient based local search optimizing a single individual only; smallAll/largeAll: same as smallOne/largeOne but optimizing all individuals. Note that the values for multi-deme algorithms are given per population, e.g. the MDLGA features an overall number of individuals of $5 \times 40 = 200$.

For the evaluation, we recorded the best score and the total number of scoring function evaluations for each docking run. For each complex, we also recorded the energetically best binding pose found in all our docking runs (including docking results from preliminary studies). We call this position the target binding pose in the following and define a *hit* as a ligand position featuring a root mean square deviation (RMSD) smaller than 2 Å to the corresponding target binding pose. In addition, we divided the complexes into sets

$$S_{l,u} := \{\text{complex where: } l \leq \text{number of rotatable bonds of ligand} \leq u\}$$

containing ligands with a similar degree of difficulty. In this study, we used three sets $S_{0,3}$, $S_{4,7}$, $S_{8,11}$.

To compare the search methods in terms of their ability to find the presumed global optimum, we calculated the average number of hits, the average mean score, and the average best score for each algorithm a and each set $S_{l,u}$. The average number of hits was computed by

$$\frac{\sum_{j \in S_{l,u}} H_a(j)}{|S_{l,u}|},$$

where $H_a(j)$ is the number of hits for complex j divided by the total number of docking runs k_{max} (in our case $k_{max} = 300$). The average mean score was computed by

$$\frac{\sum_{j \in S_{l,u}} \sum_{k=1}^{k_{max}} E_a(j,k)}{k_{max}|S_{l,u}|},$$

where $E_a(j,k)$ is the score for complex j and docking run k. Finally, the average best score was calculated by

$$\frac{\sum_{j \in S_{l,u}} \min_{k=1,\ldots,k_{max}} \{E_a(j,k)\}}{|S_{l,u}|}.$$

To assess the reliability of the results, we defined a saturation measure. If h is the number of hits, then we define saturation as the number of hits in the h top ranked results divided by h. For example, if a meta-heuristic produced ten hits in one docking experiment and out of the ten top-ranked results, three were hits, the algorithm achieved a saturation of 0.3.

For a more in-depth comparison of the search heuristics, we also computed the normalized number of hits $\hat{H}_a$ and the normalized ratio $\hat{R}_a$ between H_a and the number of function evaluations F_a for each algorithm a as follows: Let $T(i)$ be the set of complexes with $i \in \mathbb{Z}$ rotatable bonds of the ligand. Then, $\hat{H}_a$ is calculated for each set by

$$\hat{H}_a(i) := \frac{\sum_{j \in T(i)} H_a(j)}{\max_{\alpha \in A}\{\sum_{j \in T(i)} H_\alpha(j)\}}.$$

Let $H_a(j)$ be the total number of hits and let $F_a(j)$ be the total number of function evaluations performed by algorithm a in all docking runs with complex j, then the ratio between $H_a(j)$ and $F_a(j)$ is given by

$$R_a(i) := \frac{\sum_{j \in T(i)} H_a(j)}{\sum_{j \in T(i)} F_a(j)}.$$

Finally, the normalized $\hat{R}_a$ is simply given by:

$$\hat{R}_a(i) := \frac{R_a(i)}{\max_{\alpha \in A}\{R_\alpha(i)\}}.$$

The statistical significance of differences between the search heuristics was assessed using the Mann-Whitney test as implemented in R, a system for statistical computation and graphics [169].

3.3.3 Results

The results for the new gradient-based search heuristics and for the non-gradient-based heuristics are summarized in Table 3.8. We divided the test set into three classes, namely ligands of low (0-3 flexible torsional angles, $n = 40$), medium (4-7 flexible torsional angles, $n = 37$) and high (more than 7 torsional angles, $n = 8$) complexity. In general, the gradient-based methods have a higher chance to find the target binding pose than the non-gradient-based methods, that is they produce more hits than LGA and MDLGA. This difference is even more pronounced for ligands of high complexity for which the LGA fails almost completely. A similar trend is observed for the average mean scores, where LGA and MDLGA give worse results than the gradient-based search heuristics. Statistical analysis of the differences in the number of hits confirmed this observation. For ligands with less than 8 rotatable bonds, the search algorithms fall into four distinct "groups" ($p < 0.05$). LGA and MDLGA each form a group of their own, while the other two groups contain smallOne/largeOne and smallAll/largeAll, respectively. However, for ligands with more than 7 rotatable bonds ($n = 8$), the distinction between these groups becomes blurred except for LGA, for which results still differs significantly from all other heuristics.

The average best scores for ligands with few rotatable bonds are comparable for all methods and this finding seems to hold for ligands of medium complexity, too. For ligands of high complexity, the performance of the non-gradient search methods deteriorates severely. In contrast, the gradient-based methods, with the exception of largeOne, deliver consistent results. Additionally, the gradient-based heuristics require fewer function evaluations than the non-gradient-based methods before meeting the stop criterion. However, this gap decreases as ligands get more complex and the gradient-based methods perform more local search steps. Statistical analysis of the number of function evaluations for few rotatable bonds shows the

#rot. bonds (# ligands)	name	ø hits [%]	ø mean score	ø best score	ø function eval.	ø saturation
0–3 ($n = 40$)	LGA	21.8	-77.7	-99.7	8424	0.87
	MDLGA	46.6	-89.1	**-99.8**	14360	0.87
	smallOne	60.0	-92.6	-99.7	**5657**	0.93
	largeOne	63.9	-92.7	-99.7	6178	0.89
	smallAll	78.7	-96.6	-99.6	6720	**0.94**
	largeAll	**84.8**	**-97.0**	-99.6	7369	**0.94**
4–7 ($n = 37$)	LGA	4.9	-84.8	-116.6	11792	0.61
	MDLGA	15.4	-98.4	-122.3	20275	0.55
	smallOne	26.7	-105.2	**-123.6**	**9620**	0.70
	largeOne	25.1	-101.9	-123.1	10105	0.63
	smallAll	39.7	**-110.2**	-123.5	11698	**0.72**
	largeAll	**47.2**	-109.5	-123.3	12497	0.71
8–11 ($n = 8$)	LGA	0.1	-89.0	-124.7	15055	–
	MDLGA	3.8	-106.9	-147.7	26048	0.42
	smallOne	8.2	-123.8	**-156.4**	13533	0.44
	largeOne	5.1	-112.9	-153.6	**13498**	0.43
	smallAll	13.5	-130.1	-156.1	15728	0.43
	largeAll	**21.6**	-130.6	**-156.4**	18354	**0.52**

Table 3.8: Comparison of meta-heuristics in terms of average number of hits, average mean and best score, average number of function evaluations as well as saturation. The results are partitioned for small, medium and large number of flexible torsional angles. Bold letters indicate best results for each column and category.

same groupings as observed for the number of hits ($p < 0.05$): (1) LGA, (2) MDLGA, (3) smallOne/largeOne, (4) smallAll/largeAll. For ligands with more than three rotatable bonds, it is not possible to classify LGA into a single group anymore. However, the differences between the groups MDLGA, smallOne/largeOne, and smallAll/largeAll are still significant.

As for the saturation, the gradient-based methods optimizing all individuals have a higher chance to rank hits higher than the other search heuristics. For the gradient-based methods optimizing just a single individual per population and iteration, the one featuring a smaller population seems to produce slightly better results. Figure 3.13a displays the chance to find the target binding pose relative to the best performing search heuristic. Obviously, the gradient-based methods are always well above their competitors, and again, the difference becomes larger as the complexity of the ligands increases. Nonetheless, the MDLGA produces a considerably higher number of hits than the LGA and seems to perform quite well.

When evaluating the number of hits per number of function evaluations, the difference between the gradient-based and the non-gradient-based genetic algorithms is even more pronounced, while the differences between the search heuristics using the same approach for local optimization are less marked, see Figure 3.13b. Here, LGA and MDLGA show a similar profile. Thus, the higher number of hits found by the MDLGA is counter-balanced by a higher number of function evaluations and hence a longer run-time of the search algorithm.

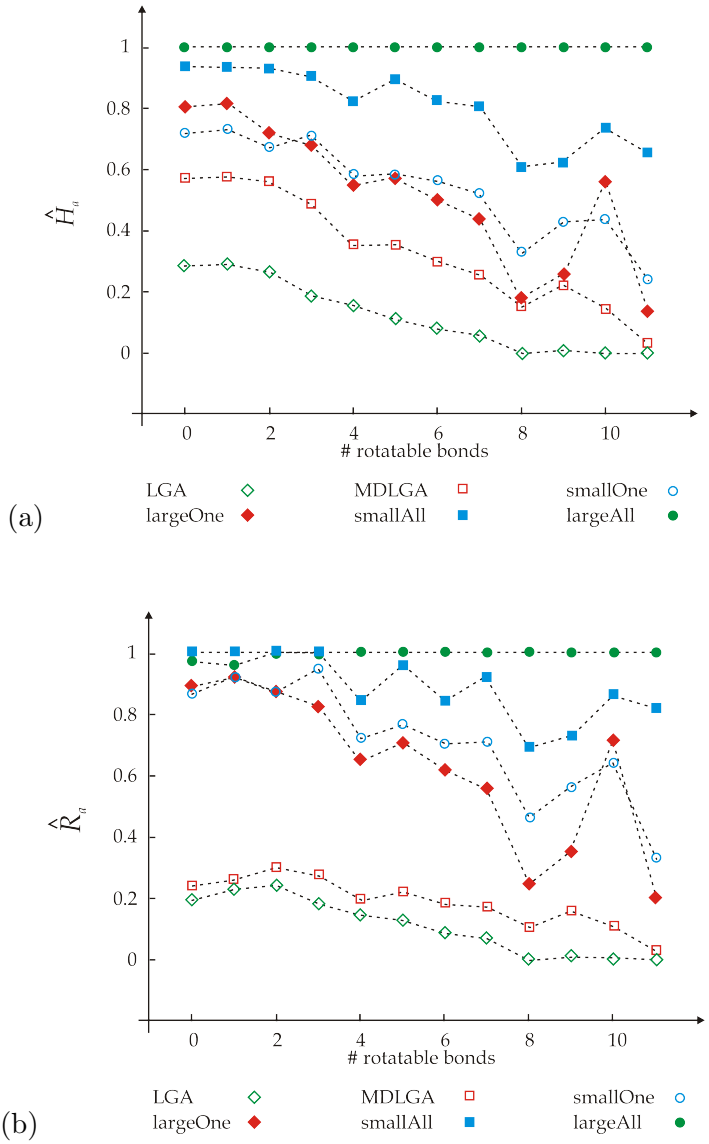

Figure 3.13: Comparison of the various genetic algorithms with respect to (a) the normalized number of hits and (b) the normalized number of hits per number of function evaluations. The best performing method was used for normalizing the values of all other methods.

3.3.4 Conclusion

Our results strongly indicate that population based search heuristics benefit from incorporating a gradient based search method. Regardless of the complexity of the ligand, the gradient-based methods deliver better results with fewer iterations. For ligands of high complexity, the performance of non-gradient procedures breaks down, while gradient-based methods are still feasible.

The advantage of the new hybrid method can be exemplified by the direct comparison between the MDLGA and smallOne, which both perform local optimization of a single individual. Although MDLGA features a much larger total population of 200 individuals than smallOne (50 individuals), the gradient-based search heuristic produces more hits while

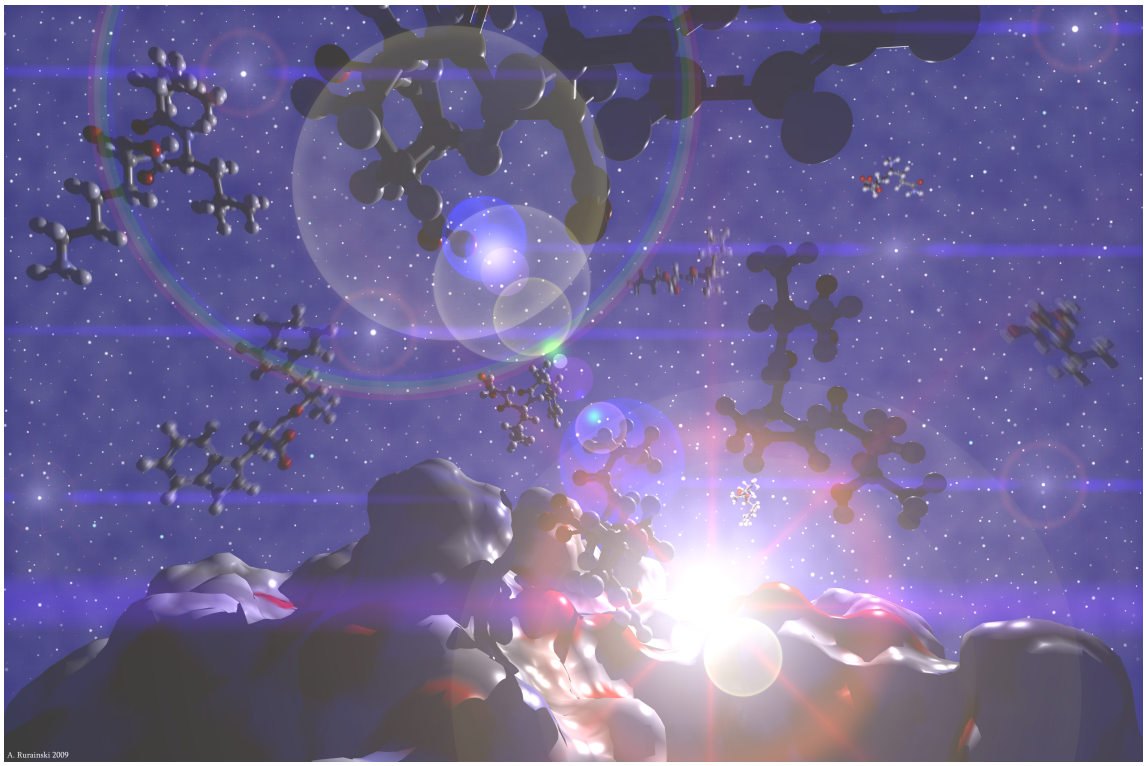

Figure 3.14: Different genetic individuals during the docking process. This picture is the graphical abstract of our publication [55] in the Journal of Computational Chemistry.

performing much fewer iterations (approximately 50%) than MDLGA – irrespective of the number of rotatable bonds.

A simple stochastic calculation may help to underline the improvement. The probability p_r to find at least a single hit after r docking runs is given by

$$p_r = 1 - \prod_{i=0}^{r-1} \frac{r_{\text{total}} - h - i}{r_{\text{total}} - i}, \tag{3.8}$$

where r_{total} is the total number of docking runs and h is the number of hits. Thus, if we want to achieve a 99% chance to find the target binding pose at least once, we have to find $\min\{r \mid p_r \geq 0.99\}$. The enhancement of using the gradient-based approach may be shown by inserting the appropriate figures into Equation (3.8). For example, for ligands of high complexity, we have $r_{\text{total}} = \sharp\text{ligands} \cdot k_{max} = 2400$ and $h = 90$ for MDLGA, whereas $h = 518$ for largeAll. We easily see that we must perform at least 118 docking experiments with MDLGA, but only 19 with largeAll. If we take into account that largeAll also requires fewer function evaluations, we gain an almost tenfold speedup.

The improvement in performance indicates that a higher number of degrees of freedom may be studied during docking. Thus, our method may facilitate docking of highly flexible ligands or to incorporate protein flexibility in the docking algorithm (e.g. in cross docking).

We also tried to answer the question of how many individuals should be locally optimized. Both heuristics that optimized all individuals (smallAll, largeAll) performed slightly better than those performing local search for only one individual per population and itera-

tion (smallOne, largeOne). This finding is supported by the fact that smallOne performed better than largeOne: smallOne dedicates a larger fraction of its function evaluations to the gradient-based local search procedure than largeOne. Despite the advances in molecular docking presented in this work, there is no guarantee that any however sophisticated search algorithm will identify the native binding pose correctly. The performance of finding a true hit depends directly on the quality of the scoring or energy function. Especially when using an unsuited scoring function, the probability of finding a hit is small and even if the native binding pose is reconstructed by chance, it is very improbable that this result is top-ranked.

3.4 Receptor Flexibility in Ligand-Receptor Docking

In the previous section the receptor was rigid, which is a widely used assumption in ligand-receptor docking. However, this is a strong assumption. Not only side chains may fold down during the docking process. Different backbone conformations of the same protein in the PDB [18] show that the backbone plays an important role in some cases. The assumption of a rigid backbone does not hold in general.

In this section, we present our approach to incorporate receptor flexibility into our docking method. In general, our gradient-based local search method is independent of the used meta-heuristic. For this study, we replaced the genetic algorithm by a differential evolution approach [166]. The intention was a comparison of the docking performance between genetic algorithms and differential evolution. The results are beyond the scope of this work and can be found in Fuhrmann [53]. In this study of Schackmann [156], see also Rurainski et al. [152], we chose the Human Serum Albumin (HSA), which is the most abundant transport protein of the human blood plasma. It is known for its promiscuity to bind different ligand species and it is one of the most extensively studied proteins. Thus, many high-resolution structure entries are available in the PDB providing a good basis for intense investigations of conformational changes. Finally, the fact that HSA undergoes tremendous backbone movements upon binding to fatty acids, facilitating enormous topological and structural changes over the whole protein, renders this molecule to an ideal candidate for our study.

3.4.1 Differential Evolution

Since we do not focus on meta-heuristics in this work, we only sketch the main ideas behind differential evolution (DE). Details can be found in, e.g., Rurainski et al. [152], Schackmann [156], and Fuhrmann [53].

DE is strongly related to genetic algorithms (GA) and differs mainly in two points. First, in a GA a certain proportion of the population is discarded, where DE replaces only existing individuals by better ones. Second, the procedure of creating new individuals by existing ones is more complex than the mating process in a GA, see Figure 3.15. Two individuals are selected and the difference between them is computed. Multiplied by a weighting factor the result is then added to a third individual. The obtained vector is called *trial vector*. Finally, DE chooses another individual, the so called *base vector*, and performs a crossover operation by randomly blending elements of the base and trial vector. This outcome replaces the base vector only if it features a better score. All individuals appearing in this DE approach are optimized by our local search procedure of Section 3.2.

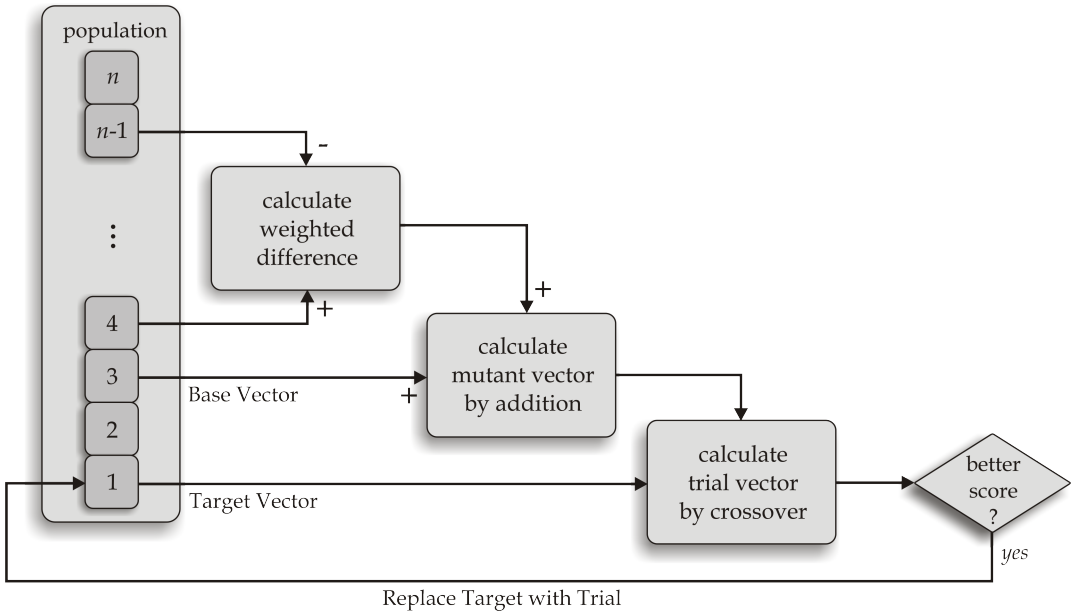

Figure 3.15: Basic illustration of the differential evolution optimization procedure introduced by Storn [166]. Start with the random selection of four individuals. Addition of the weighted difference of two members to the base vector delivers a new mutant vector. After crossover of mutant and target vector, the resulting trial vector replaces the target vector if it achieves a better score.

3.4.2 Side Chain Flexibility

A widely used approach is the application of a rotamer library. Such libraries contain discrete sets of the most frequently occuring torsional angles in preselected sets of crystal structures for each rotamer. Thus, sampling becomes a combinatorial task: Find the best scoring conformation among the discrete set of angles. Unfortunately, in preliminary studies we identified proteins whose crystal structure could not be reproduced using, e.g., Dunbrack's backbone-dependent rotamer library [46]. Furthermore, the idea behind our gradient-based docking approach is to consider all possible conformations and not only discrete ones. A natural idea is then to use the same approach for side chains as we employed for the ligand's flexibility, i.e. we have one parameter per rotatable bond in our optimization problem.

In our definition, a bond always subdivides a molecule into two atom sets (S_1 and S_2) and the bond is only accepted as rotatable if S_1 and S_2 contain at least two heavy atoms. This restriction is due to the fact that hydrogen positions are not explicitly contributing to the scoring function and sets containing only one heavy atom are therefore rotationally invariant with respect to the bond. In contrast to ligand movements, where the part with fewer atoms is rotated in practice, our implementation for side chains ensures that the backbone structure is preserved. Table 3.9 shows how many degrees of freedom (number of rotatable bonds) the essential amino acids possess [156].

3.4.3 Backbone Flexibility

The main idea of our approach is to use two extreme backbone conformations of the same protein, obtained from the PDB [18], and to interpolate the residue positions. To be more

Name	DOFs	Name	DOFs	Name	DOFs	Name	DOFs
Ala	0	Gln	3	Leu	2	Ser	1
Arg	5	Glu	3	Lys	4	Thr	1
Asp	2	Gly	0	Met	3	Trp	2
Asn	2	His	2	Phe	2	Tyr	2
Cys	1	Ile	3	Pro	0	Val	1

Table 3.9: List of flexible torsion angles of essential amino acids. The degrees of freedom (DOFs) are equal to the number of rotatable bonds and the number of variables introduced in the optimization problem for the given residue.

exact, we need a smooth path starting from one position ending at the other position for each residue. Other approaches [115] use linear interpolation. This is certainly the best way for the translational part of the parameters. However, the orientational part suffers from unphysical states and unfavorable conformations during the "morphing process". To avoid this unrealistic behavior, our novel approach uses spherical linear extrapolation (SLERP) [163] for the rotational component, where we interpolate between the two empirical states via moves on the four-dimensional quaternion space. In summary, we obtain one additional parameter $t \in [0, 1]$ for each residue, which shifts the backbone atoms from the first position ($t = 0$) to the end position ($t = 1$) on a continuously differentiable path. These parameters are finally added to the optimization problem.

3.4.3.1 Backbone Transformation

First, we have to chose one of the given extreme conformations as start position. Afterwards, we have to determine an optimum transformation (translation and orientation) from this position to the other one using the backbone atoms. Finally, we have to create a smooth path from the identity (start position) to the determined optimum transformation (end position).

The optimum transformation for the translational part is straightforward. A small calculation shows that two point sets can only have minimum root mean square deviation (RMSD) if they have identical geometric centers. Thus, we have only to ensure that the atoms of both conformations have the same geometric centers after the transformation.

The rotational component is the challenging part. We follow the work of Coutsias et al. [33] for using quaternions to calculate the RMSD between two point sets. This study is strongly related to Kabsch's algorithm [91, 92] for computing the optimum rotation matrix $\mathbf{R}$ incorporating singular value decomposition of a spatial correlation matrix between a pair of point sets. Despite its popularity, this parametrization of the orientational part of the overall transformation suffers from singularities of the Euler angle parametrization and can violate the chirality of a solid object, whereas our constrained quaternion parametrization does not.

Let $\mathbf{v}_i$ be the positions of backbone atoms of conformation V at $t = 0$ and $\mathbf{w}_i$ those of conformation W. By the approach of Coutsias et al. [33] we obtain a unit quaternion q_{opt}, which represents the optimum rotation to map $\mathbf{v}_i$ onto $\mathbf{w}_i$, $i = 1, \ldots, N$. Thus, q_{opt} is the final quaternion, whereas the identity quaternion $q_0 = (0, 0, 0, 1)^T$ is the start quaternion. Using spherical linear extrapolation we obtain a smooth quaternion path between q_0 and q_{opt}

with parameter t

$$q(t) := \frac{\sin((1-t)\varrho)}{\sin(\varrho)} q_0 + \frac{\sin(t\varrho)}{\sin(\varrho)} q_{\text{opt}},$$

where ϱ is computed as the angle subtended by the arc, so that $\cos(\varrho) = q_0^T q_{\text{opt}}$. Let $q_x(t)$, $q_y(t)$, $q_z(t)$, and $q_w(t)$ be the components of $q(t)$, i.e. $q(t) = (q_x(t), q_y(t), q_z(t), q_w(t))^T$, then the corresponding rotation matrix $\mathbf{R}(t)$ is given by

$$\mathbf{R}(t) := \begin{pmatrix} 1-2((q_y(t))^2+(q_z(t))^2) & 2(q_x(t)q_y(t)-q_z(t)q_w(t)) & 2(q_z(t)q_x(t)+q_y(t)q_w(t)) \\ 2(q_x(t)q_y(t)+q_z(t)q_w(t)) & 1-2((q_z(t))^2+(q_x(t))^2) & 2(q_y(t)q_z(t)-q_x(t)q_w(t)) \\ 2(q_z(t)q_x(t)-q_y(t)q_w(t)) & 2(q_y(t)q_z(t)+q_x(t)q_w(t)) & 1-2((q_x(t))^2+(q_y(t))^2) \end{pmatrix},$$

yielding the smooth path $\mathbf{v}_i(t)$ of each $\mathbf{v}_i$

$$\mathbf{v}_i(t) := \mathbf{R}(t)(\mathbf{v}_i - \theta_V) + \theta_V + t(\theta_W - \theta_V), \tag{3.9}$$

where θ_V and θ_W are the geometric centers of V and W

$$\theta_V = \frac{1}{N}\sum_{i=1}^{N} \mathbf{v}_i, \qquad \theta_W = \frac{1}{N}\sum_{i=1}^{N} \mathbf{w}_i.$$

3.4.3.2 Gradient of Backbone Transformation

For the incorporation of the obtained smooth path into our gradient-based local search method, we use the chain rule to calculate the derivative for the parameter t. The formula for the translational part is obvious

$$\begin{aligned} \frac{d}{dt}\mathbf{v}_i(t) &= \frac{d}{dt}(\mathbf{R}(t)(\mathbf{v}_i - \theta_V) + \theta_V + t(\theta_W - \theta_V)) \\ &= \frac{d}{dt}(\mathbf{R}(t)(\mathbf{v}_i - \theta_V)) + (\theta_W - \theta_V). \end{aligned}$$

For the orientational part we first compute the gradient $\nabla q(t)$ considering its components $q_x(t)$, $q_y(t)$, $q_z(t)$, and $q_w(t)$

$$\nabla q(t) = \frac{1}{\sqrt{1-\phi^2}} \begin{pmatrix} \cos(t\cdot\phi)\cdot q_x(t) \\ \cos(t\cdot\phi)\cdot q_y(t) \\ \cos(t\cdot\phi)\cdot q_z(t) \\ \phi\cdot(\cos(t\cdot\phi))\cdot q_w(t) - \cos((1-t)\cdot\phi), \end{pmatrix} \tag{3.10}$$

with $\phi = \cos^{-1}(q_w(t))$. $\mathbf{R}(t)$ can be seen as mapping that maps $q(t)$ into $\mathbb{R}^9$. Thus the resulting Jacobian is given by

$$\mathbf{J}_{\mathbf{R}}(q(t)) = \begin{pmatrix} 0 & -4q_y(t) & -4q_z(t) & 0 \\ 2q_y(t) & 2q_x(t) & -2q_w(t) & -2q_z(t) \\ 2q_z(t) & 2q_w(t) & 2q_x(t) & 2q_y(t) \\ 2q_y(t) & 2q_x(t) & 2q_w(t) & 2q_z(t) \\ -4q_x(t) & 0 & -4q_z(t) & 0 \\ -2q_w(t) & 2q_z(t) & 2q_y(t) & -2q_x(t) \\ 2q_z(t) & -2q_w(t) & 2q_y(t) & -2q_y(t) \\ 2q_w(t) & 2q_z(t) & 2q_y(t) & 2q_x(t) \\ -4q_x(t) & -4q_y(t) & 0 & 0 \end{pmatrix}. \tag{3.11}$$

Using the Projection

$$\mathbf{P} = \begin{pmatrix} s_x & s_y & s_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & s_x & s_y & s_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & s_x & s_y & s_z \end{pmatrix}, \tag{3.12}$$

where $\mathbf{v}_i - \theta_V = (s_x, s_y, s_z)$, finally leads to the Jacobian matrix of $\mathbf{v}_i(t)$:

$$\mathbf{J}_{\mathbf{v}_i}(t) = \mathbf{P} \cdot \mathbf{J}_{\mathbf{R}}(q(t)) \cdot \nabla q(t) + (\theta_W - \theta_V). \tag{3.13}$$

The overall gradient entry g_t of parameter t is the sum of all atom wise contributions. Thus, we finally calculate

$$g_t = \sum_{i=1}^{N} \mathbf{J}_{\mathbf{v}_i}(t)^T \mathbf{f}_i, \tag{3.14}$$

with N being the number of heavy atoms moved by the transformation and $\mathbf{f}_i$ being the force vector acting on atom i at position $\mathbf{v}_i(t)$.

3.4.4 Results

Our approach has been implemented in C++ using the BALL framework [101, 102, 76]. In order to find good parameters for our final test runs, we first carried out several docking runs with reduced flexibility. In the first scenario only the ligand was flexible as in Section 3.3, in the second scenario additionally the side chains were allowed to move. The resulting parameters should reasonably balance the probability of finding low RMSD hits (saturation) and a feasible runtime for a single docking study. In particular, the size of the genetic pool was varied during testing as well as the number of allowed iterations for our gradient-based optimization method. We found that the approach reached its allowed iteration boundary for at most 20% of all observed optimizations in test scenarios. As a result of our tests, we came up with the specification of limiting the pool size to 100 individuals and the iteration boundary to 100 optimization steps. The known drawback of our model is that with more flexibility the runtimes increase. Our approach that only keeps the ligand flexible took only a few minutes on a Dual-Core AMD Opteron Processor 2222 for 300 docking runs. Our new approach took about 60 hours to produce 50 results for the smallest ligand (B3I) and about 300 hours to produce the same amount of docked results for the most complex ligand (IDB). One reason for these runtimes is the immense number of pair interactions to be calculated for the Gehlhaar scoring function [56]. Another reason is that for the above presented flexible setup for IDB, our algorithm had to scope with 89 degrees of freedom. Nevertheless, our approach was able to reproduce the reference position with an RMSD of 2.07 Å.

In order to overcome the known limitations of the Gehlhaar function to accurately estimate the binding-free energy of a complex, we rescored the results using X-Score [184, 185, 186, 183]. We used the Merck Molecular Force Field (MMFF94) [63, 64, 65, 66, 67] and performed structure optimizations to provide X-Score with valid hydrogen geometries. However, the application of the semi-empirical consensus X-Score for reranking did not improve the results. Geometry optimization of the hydrogen atoms with MMFF94 showed slightly improved results. Nevertheless, in conclusion, we have to admit that neither the Gehlhaar scoring nor X-Score are able to assess the structural situation sufficiently. In future work, the implementation of other scoring functions is needed to guide the optimization process to a higher

rate of low RMSD hits. Nonetheless, the application has generally shown to be able to cope successfully with the tremendous amount of degrees of freedom (more than 79) in each of our validation setups, even if the scores are not reflecting this. Table 3.10 shows our results.

Ligand	#RBs	Hits (≤ 3 Å)	DOFs	Best RMSD	$\emptyset$ RMSD	TopScore RMSD
B3I	1	0	79	3.11	5.4	3.92
AZQ	3	3	81	2.2	6.08	6.75
WRS	4	0	82	3.43	6.24	6.33
P1Z	5	2	83	2.09	6.04	2.09
IDB	11	1	89	2.07	9.1	8.77

Table 3.10: Fully flexible docking runs. The receptor was 1BM0 (PDB ID) with 34 flexible residues in the setup. The genetic pool always consisted of 100 individuals. 100 docking runs were performed. We give the name of the docked ligand as PDB ID, the number of flexible torsional angles inside the ligand (RBs), the number of calculated structures that differ from the reference structures by an RMSD of less than 3 Å, the number of variables (degrees of freedom, DOFs), the best RMSD, the mean RMSD, and the RMSD of the best scored structure.

3.4.5 Conclusion

Our results show that our method to support limited backbone flexibility is a promising approach for ligand-receptor docking. However, according to our investigations the main problem seems to be the scoring function and no longer the method to find favorable conformations. The development of better scoring functions is not subject of this thesis, but will be the next step to improve our results in the future. Such a simple function like the Gehlhaar scoring function [56] seems not to be capable for penalizing unrealistic conformations, when side chains are flexible. In many resulting structures, we found that side chains obviously interact nearly perfectly with each other but the remaining space between them was too small for the ligand. To be more precise, the ligand could not be positioned between the side chains because the scoring function favored perfect interactions of the side chains. In these situations, a simple weighting of ligand-receptor interactions and receptor-receptor interactions may alleviate these problems. Future investigations will provide more insight.

3.5 Backbone Flexibility in Ligand-Receptor Docking

The study in the previous section achieves backbone flexibility by interpolating between (known) extreme conformations. Here, we allow a loop region to be fully flexible, see Rurainski et al. [151] and Roth [148]. We show that the Gō-Scheraga ring closure equations combined with our interval arithmetic based approach to calculate guaranteed all solutions of the occurring equations is able to identify promising loop/ligand complexes in our test runs. In fact, we were able to dock ligands into a highly flexible loop region, where other approaches failed in preliminary studies. In an outer loop a Monte Carlo approach randomizes "free" parameters, i.e. parameters that do not mathematically depend on each other, e.g., torsion angles at both endpoints of the loop. Our approach then calculates all closed loop conformations, i.e. it determines all different values for the remaining dependent parameters in a way

that the loop is closed. Afterwards, the method introduced in Section 3.4 with the Gehlhaar scoring function [56] is used to dock the fully flexible ligand with the receptor, where side chains are flexible, but the backbone is now kept rigid.

In the study of Roth [148], see also Rurainski et al. [151], we chose the human 17β-hydroxysteroid dehydrogenase type 1 (17β-HSD 1), which was first described over 50 years ago by Engler and Langer in 1958 [109, 108]. Much research has been done in the last decades resulting in the determination of function, amino acid sequence and overall structure of this protein. Playing a central role in the development and proliferation of breast cancer and other estrogen-dependent diseases, 17β-HDS 1 presents itself as a promising target for drug development [74]. While several putative inhibitors of this enzyme have been studied during the last decades, up to now no clinical trial has been performed and no final drug has been released yet [143].

3.5.1 Gō-Scheraga Ring Closure Equations

For the loop closure in polypeptides, Gō and Scheraga [58] published an approach in 1970. Essentially, they presented how the problem of finding the parameters of a closed loop given both endpoints can be reduced to an equation in one variable. All other unknown parameters can be calculated using the solution. In order to solve the central equation they used the best available method when they published their work: Newton's method. Since this is only a local search method, they may have sampled the search space by providing different equally distributed start positions. Thus, there is no guarantee to find all solutions and, most likely, many function and derivative evaluations had to be performed. Our interval arithmetic based approach presented in the next section guarantees to identify all solutions of this equation. In our test runs, all results were obtained nearly instantly. In this section, we give the ideas behind the loop closure equations and state the resulting formulas.

The original work of Gō and Scheraga [58] had to be modified due to a change in IUPAC conventions concerning torsion angles in 1970, resulting in dihedral angles being assigned values between -180° and 180° rather than between 0° and 360°, to provide information about the angle's direction. In addition, the original approach assumes fixed coordinates of backbone atoms: they provide the ideal theoretical positions of the backbone atoms in the corresponding coordinate system, see Tables 3.11 and 3.12. However, it might not be possible to obtain exact loop closure in reality using these values due to slightly perturbed coordinates in the PDB file. Thus, we calculated these values from the respective coordinate of the backbone atoms in the original conformation of the loop. The obtained coordinates differ slightly from the proposed ones.

atom	x	y	z
C_i^α	0.000	0.000	0.000
C_i'	1.530	0.000	0.000
N_{i+1}	2.067	1.206	0.000
C_{i+1}^α	3.519	1.436	0.000

Table 3.11: Cartesian coordinates of atoms in local coordinate system $2i$ in Å.

Gō and Scheraga determine torsion angles that are needed to connect the two known endpoints of the loop by mathematical equations. Keeping bond lengths and bond angles

atom	x	y	z
N_i	0.000	0.000	0.000
C_i^{α}	1.470	0.000	0.000
C_i'	1.980	1.443	0.000

Table 3.12: Cartesian coordinates of atoms in local coordinate system $2i-1$ in Å.

fixed for n torsion angles results in $n-6$ independent and six dependent dihedrals, the latter ones to be calculated based on the values assigned to the former ones, due to six degrees of freedom necessary for the calculations, as we show below. Five dependent torsion angles can be computed using the value of the remaining dependent one, which has to be calculated by solving the central equation and is assumed to be known for the moment. Free torsion angles (ω_i) of the loop are described by local coordinate systems as shown in Figure 3.16, also indicating how bond lengths (p_i) and bond angles (θ_i) are defined. Additionally, the backbone atoms are given with their individual names. A given point in space can be represented by

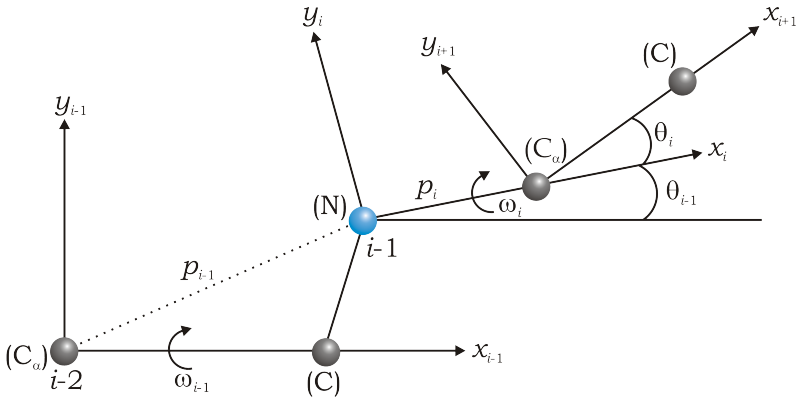

Figure 3.16: Definition of local coordinate systems in the Gō-Scheraga equations.

position vectors $\mathbf{r}_i$ and $\mathbf{r}_{i-1}$ with respect to the ith and $(i-1)$th coordinate system, where we use the original notation of Gō and Scheraga. Thus, their relation can be described by

$$\mathbf{r}_{i-1} = \mathbf{T}_{i-1}\mathbf{R}_i\mathbf{r}_i + \mathbf{p}_{i-1},$$

$$\mathbf{T}_{i-1} = \begin{pmatrix} \cos\theta_{i-1} & -\sin\theta_{i-1} & 0 \\ \sin\theta_{i-1} & \cos\theta_{i-1} & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

and

$$\mathbf{R}_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\omega_i & -\sin\omega_i \\ 0 & \sin\omega_i & \cos\omega_i \end{pmatrix}. \tag{3.15}$$

Hence, we can express the endpoint of the loop in the coordinate system of the starting point and vice versa by successive coordinate transformations. Six torsion angles as free variables correspond to three needed amino acid residues, each including two dihedrals, namely ϕ and ψ, to perform loop closure. Gō and Scheraga use three vectors to describe the transformation

between the loop's endpoints: $\mathbf{s}$ points from the first endpoint to the second, $\mathbf{u}$ is the direction of the x-axis of the second endpoint's coordinate system to the first endpoint's coordinate system, and $\mathbf{v}$ is the corresponding direction of the y-axis. Using unit vectors we have

$$\begin{aligned}\mathbf{u}^T\mathbf{u} &= 1,\\ \mathbf{v}^T\mathbf{v} &= 1,\\ \mathbf{u}^T\mathbf{v} &= 0.\end{aligned}$$

Let $\mathbf{e}_1$ and $\mathbf{e}_2$ be unit vectors in x and y direction, then we can calculate the three vectors $\mathbf{s}_n$, $\mathbf{u}_n$ and $\mathbf{v}_n$, which express the relative position and orientation of the nth coordinate system with respect to coordinate system 0, by

$$\begin{aligned}\mathbf{s}_n &= \mathbf{p}_0 + \mathbf{T}_0\mathbf{R}_1\mathbf{p}_1 + \mathbf{T}_0\mathbf{R}_1\mathbf{T}_1\mathbf{R}_2\mathbf{p}_2 + \cdots + \mathbf{T}_0\mathbf{R}_1\mathbf{T}_1\mathbf{R}_2\cdots\mathbf{T}_{n-2}\mathbf{R}_{n-1}\mathbf{p}_{n-1}, \qquad (3.16)\\ \mathbf{u}_n &= \mathbf{T}_0\mathbf{R}_1\mathbf{T}_1\mathbf{R}_2\cdots\mathbf{T}_{n-2}\mathbf{R}_{n-1}\mathbf{T}_{n-1}\mathbf{R}_n\mathbf{e}_1,\\ \mathbf{v}_n &= \mathbf{T}_0\mathbf{R}_1\mathbf{T}_1\mathbf{R}_2\cdots\mathbf{T}_{n-2}\mathbf{R}_{n-1}\mathbf{T}_{n-1}\mathbf{R}_n\mathbf{e}_2.\end{aligned}$$

If the $(n-6)$ independent variables are arbitrarily chosen as ω_1 to ω_{n-6}, Equations (3.17) to (3.19) have to be solved for the dependent ones.

$$\begin{aligned}\mathbf{s} &= \mathbf{p}_{n-6} +\\ &\quad \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{p}_{n-5} +\\ &\quad \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{T}_{n-5}\mathbf{R}_{n-4}\mathbf{p}_{n-4} +\\ &\quad \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{T}_{n-5}\mathbf{R}_{n-4}\mathbf{T}_{n-4}\mathbf{R}_{n-3}\mathbf{p}_{n-3} +\\ &\quad \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{T}_{n-5}\mathbf{R}_{n-4}\mathbf{T}_{n-4}\mathbf{R}_{n-3}\mathbf{T}_{n-3}\mathbf{R}_{n-2}\mathbf{p}_{n-2} +\\ &\quad \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{T}_{n-5}\mathbf{R}_{n-4}\mathbf{T}_{n-4}\mathbf{R}_{n-3}\mathbf{T}_{n-3}\mathbf{R}_{n-2}\mathbf{T}_{n-2}\mathbf{R}_{n-1}\mathbf{p}_{n-1} \qquad (3.17)\\ \mathbf{u} &= \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{T}_{n-5}\mathbf{R}_{n-4}\mathbf{T}_{n-4}\mathbf{R}_{n-3}\mathbf{T}_{n-3}\mathbf{R}_{n-2}\mathbf{T}_{n-2}\mathbf{R}_{n-1}\mathbf{T}_{n-1}\mathbf{R}_n\mathbf{e}_1 \qquad (3.18)\\ \mathbf{v} &= \mathbf{T}_{n-6}\mathbf{R}_{n-5}\mathbf{T}_{n-5}\mathbf{R}_{n-4}\mathbf{T}_{n-4}\mathbf{R}_{n-3}\mathbf{T}_{n-3}\mathbf{R}_{n-2}\mathbf{T}_{n-2}\mathbf{R}_{n-1}\mathbf{T}_{n-1}\mathbf{R}_n\mathbf{e}_2 \qquad (3.19)\end{aligned}$$

Hence, we can express $\mathbf{s}$, $\mathbf{u}$, and $\mathbf{v}$ in terms of the $n-6$ independent variables, where $\mathbf{s}$ is the position vector of the origin of the nth coordinate system with respect to the $(n-6)$th one.

$$\begin{aligned}\mathbf{s} &= \mathbf{R}_{n-6}^{-1}\mathbf{T}_{n-7}^{-1}\cdots\mathbf{R}_1^{-1}\mathbf{T}_0^{-1}\cdot\\ &\quad \cdot\left(\mathbf{s}_n - \mathbf{p}_0 - \mathbf{T}_0\mathbf{R}_1\mathbf{p}_1 - \cdots - \mathbf{T}_0\mathbf{R}_1\cdots\mathbf{T}_{n-8}\mathbf{R}_{n-7}\mathbf{p}_{n-7}\right), \qquad (3.20)\\ \mathbf{u} &= \mathbf{R}_{n-6}^{-1}\mathbf{T}_{n-7}^{-1}\cdots\mathbf{R}_1^{-1}\mathbf{T}_0^{-1}\mathbf{u}_n,\\ \mathbf{v} &= \mathbf{R}_{n-6}^{-1}\mathbf{T}_{n-7}^{-1}\cdots\mathbf{R}_1^{-1}\mathbf{T}_0^{-1}\mathbf{v}_n.\end{aligned}$$

These equations apply to torsion angles of arbitrary chain molecules. If we consider a polypeptide chain with different torsion angles ϕ and ψ, we have to change Equation (3.15) into

$$\mathbf{R}_{2i+1} = \mathbf{R}_{\phi_{i+1}+\pi} = \begin{pmatrix}1 & 0 & 0\\ 0 & \cos(\phi_{i+1}+\pi) & -\sin(\phi_{i+1}+\pi)\\ 0 & \sin(\phi_{i+1}+\pi) & \cos(\phi_{i+1}+\pi)\end{pmatrix}$$

and

$$\mathbf{R}_{2i} = \mathbf{R}_{\psi_i+2\pi} = \begin{pmatrix}1 & 0 & 0\\ 0 & \cos(\psi_i+2\pi) & -\sin(\psi_i+2\pi)\\ 0 & \sin(\psi_i+2\pi) & \cos(\psi_i+2\pi)\end{pmatrix}.$$

In the following we assume the independent torsional angles to be given and $\mathbf{s}$ being calculated by using Equations (3.16) and (3.20), respectively. Thus, a part of the loop is already given by the independent torsional angles, i.e. we have a new "endpoint", and we are faced with the task to determine the remaining six dependent parameters to "close" the loop. For notational simplicity, we call these remaining variables ω_1 to ω_6 from now on.

To derive the central equation, we assume the torsional angle $\omega_1 \equiv \phi_1$ to be given for the moment. For brevity, we introduce vectors $\mathbf{q}_0$ to $\mathbf{q}_2$, where

$$\mathbf{q}_i = \mathbf{T}_i \mathbf{p}_{i-1} + \mathbf{p}_i \equiv \begin{pmatrix} \rho_i \\ \sigma_i \\ 0 \end{pmatrix}. \tag{3.21}$$

Following Gō and Scheraga, the vector $\mathbf{r}$ is needed for the calculation of the torsion angles ω_2 to ω_6, where $\omega_2 \equiv \psi_1$, $\omega_3 \equiv \phi_2$, $\omega_4 \equiv \psi_2$, $\omega_5 \equiv \phi_3$, and $\omega_6 \equiv \psi_3$

$$\mathbf{r} = \mathbf{T}_1^{-1} \mathbf{R}_{\phi_1+\pi}^{-1} \mathbf{T}_0^{-1} (\mathbf{s} - \mathbf{q_0}) \equiv \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

According to [58], necessary and sufficient conditions for the existence of a value for ω_2 are

$$\|\mathbf{q}_1\| - \|\mathbf{q}_2\| \leq \|\mathbf{r}\| \leq \|\mathbf{q}_1\| + \|\mathbf{q}_2\| \tag{3.22}$$

and

$$\frac{\rho_1(\|\mathbf{r}\|^2 - \|\mathbf{q}_1\|^2 - \|\mathbf{q}_2\|^2) + \sigma_1\sqrt{((\|\mathbf{q}_1\| + \|\mathbf{q}_2\|)^2 - \|\mathbf{r}\|^2)(\|\mathbf{r}\|_2 - (\|\mathbf{q}_1\| - \|\mathbf{q}_2\|)^2)}}{2\|\mathbf{q}_1\|^2} \leq x \leq$$
$$\frac{\rho_1(\|\mathbf{r}\|^2 + \|\mathbf{q}_1\|^2 - \|\mathbf{q}_2\|^2) + \sigma_1\sqrt{((\|\mathbf{q}_1\| + \|\mathbf{q}_2\|)^2 - \|\mathbf{r}\|^2)(\|\mathbf{r}\|_2 - (\|\mathbf{q}_1\| - \|\mathbf{q}_2\|)^2)}}{2\|\mathbf{q}_1\|^2}. \tag{3.23}$$

If these inequalities are fulfilled, we have

$$\begin{aligned} \cos\omega_2 &= \frac{y\gamma \pm z\sqrt{\|\mathbf{r}\|^2 - x^2 - \gamma^2}}{\|\mathbf{r}\|^2 - x^2}, \\ \sin\omega_2 &= \frac{z\gamma \mp z\sqrt{\|\mathbf{r}\|^2 - x^2 - \gamma^2}}{\|\mathbf{r}\|^2 - x^2}, \end{aligned} \tag{3.24}$$

where

$$\gamma = \frac{\|\mathbf{r}\|^2 - \|\mathbf{q}_1\|^2 - \|\mathbf{q}_2\|^2 - 2\rho_1 x}{2\sigma_1}.$$

The cosine of ω_4 can be calculated by

$$\cos\omega_4 = \frac{\rho_2\cos\theta_3 - (x - \rho_1)\cos\theta_2 - (\gamma - \sigma_1)\sin\theta_2}{\sigma_2\sin\theta_3}, \tag{3.25}$$

which gives only a valid result, if

$$-1 \leq \frac{\rho_2\cos\theta_3 - (x - \rho_1)\cos\theta_2 - (\gamma - \sigma_1)\sin\theta_2}{\sigma_2\sin\theta_3} \leq 1 \tag{3.26}$$

holds, since otherwise the arc cosine would not be defined. If that holds true, we have

$$\sin\omega_4 = \pm\sqrt{1 - \frac{(\rho_2\cos\theta_3 - (x-\rho_1)\cos\theta_2 - (\gamma-\sigma_1)\sin\theta_2)^2}{(\sigma_2\sin\theta_3)^2}}, \tag{3.27}$$

where the signs are *in*dependent of the signs in Equation (3.24). Gō and Scheraga use then a vector

$$\mathbf{q}_3 = \begin{pmatrix} x-\rho_1 \\ \gamma-\sigma_1 \\ \pm\sqrt{\|\mathbf{r}\|^2 - x^2 - \gamma^2} \end{pmatrix} \equiv \begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix}$$

whose z-component's sign is *inter*dependent with those in Equation (3.24). This means that if the upper (lower) sign is taken in one equation, the upper (lower) has to be chosen in the other two equations. Then, ω_3 can be determined by

$$\begin{aligned} \cos\omega_3 &= \frac{(\rho_2\sin\theta_3 + \sigma_2\cos\omega_4\cos\theta_3)\cdot(-q_x\sin\theta_2 + q_y\cos\theta_2) + \sigma_2 q_z\sin\omega_4}{(-q_x\sin\theta_2 + q_y\cos\theta_2)^2 + q_z^2}, \\ \sin\omega_3 &= \frac{(\rho_2\sin\theta_3 + \sigma_2\cos\omega_4\cos\theta_3)q_z - (-q_x\sin\theta_2 + q_y\cos\theta_2)\sigma_2\sin\omega_4}{(-q_x\sin\theta_2 + q_y\cos\theta_2)^2 + q_z^2}. \end{aligned} \tag{3.28}$$

Because of the signs in Equations (3.24) and (3.27), there exist two solutions for ω_2 in general and analogously for ω_4. Since both groups of signs are independent as mentioned above, this yields four sets of solutions for ω_2 and ω_4, for given $\mathbf{s}$ and ω_1, affecting as well the values of ω_3, ω_5, and ω_6. Using Equations (3.17) and (3.21) we have

$$\mathbf{s} = \mathbf{q}_0 + \mathbf{T}_0\mathbf{R}_{\phi_1+\pi}\mathbf{T}_1\mathbf{R}_{\psi_1+2\pi}\mathbf{q}_1 + \mathbf{T}_0\mathbf{R}_{\phi_1+\pi}\mathbf{T}_1\mathbf{R}_{\psi_1+2\pi}\mathbf{T}_2\mathbf{R}_{\phi_2+\pi}\mathbf{T}_3\mathbf{R}_{\psi_2+2\pi}\mathbf{q}_2, \tag{3.29}$$

which can be solved utilizing the values of ω_2 to ω_4 in terms of ω_1. The result allows for the analytical determination of

$$f(\phi_1) = \mathbf{u}^T\mathbf{T}_0\mathbf{R}_{\phi_1+\pi}\mathbf{T}_1\mathbf{R}_{\psi_1+2\pi}\mathbf{T}_2\mathbf{R}_{\phi_2+\pi}\mathbf{T}_3\mathbf{R}_{\psi_2+2\pi}\mathbf{T}_4\mathbf{e}_1 - \cos\theta_5, \tag{3.30}$$

where $\mathbf{e}_1$ is the unit vector in x-direction. The roots of f are valid values for ω_1, which can in turn be used to calculate values of ω_2 to ω_4. Thus, the central equation is

$$f(\phi_1) = 0 \tag{3.31}$$

and finding all roots of f is the central task in the next section. Finally, using all obtained values so far ω_5 and ω_6 can be determined by

$$\mathbf{T}_4^{-1}\mathbf{R}_{\psi_2+2\pi}^{-1}\mathbf{T}_3^{-1}\mathbf{R}_{\phi_2+\pi}^{-1}\mathbf{T}_2^{-1}\mathbf{R}_{\psi_1+2\pi}^{-1}\mathbf{T}_1^{-1}\mathbf{R}_{\phi_1+\pi}^{-1}\mathbf{T}_0^{-1}\mathbf{u} = \begin{pmatrix} \cos\theta_5 \\ \sin\theta_5\cos\omega_5 \\ \sin\theta_5\sin\omega_5 \end{pmatrix}$$

and

$$\mathbf{T}_5^{-1}\mathbf{R}_{\phi_3+\pi}^{-1}\mathbf{T}_4^{-1}\mathbf{R}_{\psi_2+2\pi}^{-1}\mathbf{T}_3^{-1}\mathbf{R}_{\phi_2+\pi}^{-1}\mathbf{T}_2^{-1}\mathbf{R}_{\psi_1+2\pi}^{-1}\mathbf{T}_1^{-1}\mathbf{R}_{\phi_1+\pi}^{-1}\mathbf{T}_0^{-1}\mathbf{v} = \begin{pmatrix} 0 \\ \cos\omega_6 \\ \sin\omega_6 \end{pmatrix}.$$

3.5.2 Determination of ω_1

We used Maple^{TM} to express each of the above mentioned Equations (3.24), (3.25), (3.27), and (3.28) as a function of ω_1. The resulting terms took several pages. Maple^{TM} is a commercial computer algebra system developed and maintained by Waterloo Maple Inc.. It has the capability to produce C source code, which can be used in our C++ implementation. In theory, the equation

$$f(\phi_1) = 0$$

can be solved by a nested intervals approach using interval arithmetic as we described in Section 2.5.1, see also Figure 3.17. The initial interval is $[-\pi, \pi]$, i.e. the whole range of

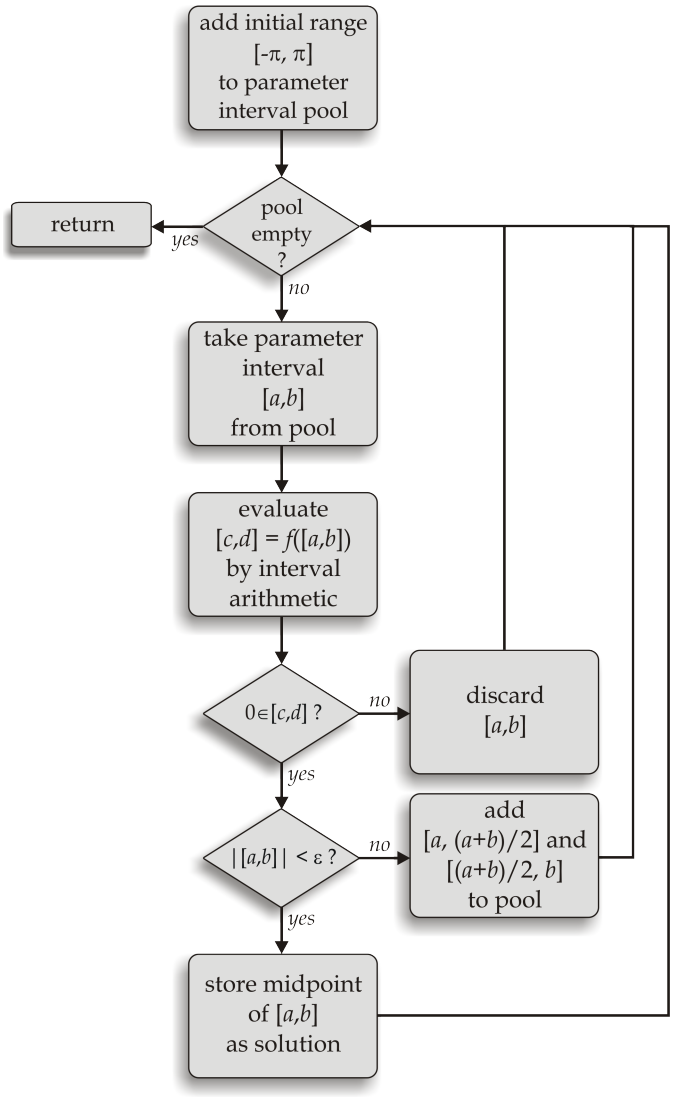

Figure 3.17: General scheme of nested intervals. The obtained solutions are closer than $\frac{\varepsilon}{2}$ to a root.

ϕ_1. However, the obtained bounds using the produced code and replacing all arithmetic operations by interval operations are not tight enough for determining all roots within a few iterations. Thus, parts of the code have been manually replaced by specialized interval func-

tions, which provide tighter bounds. For example, all occurrences of "`x*x`" have been replaced by "`sqr(x)`", which computes the interval square function and, e.g., prevents negative lower bounds, see Section 2.5.1.

Another observation is that not only real solutions of Equation (3.31) exist but also complex ones. However, we are only interested in the discrete real solutions. Considering the above derivation we see that all numbers occurring during the calculation have to be real for practical reasons. Thus, we can discard an interval if during the computation complex numbers would occur on the whole interval, i.e. inequalities (3.22), (3.23), or (3.26) do not hold. If only a part of the interval to be used for such a range check violates the corresponding inequality, we have to intersect this interval with the allowed range during the computation in order to proceed without complex interval arithmetic, still guaranteeing to find all solutions. For example, if interval calculations yield

$$\frac{\rho_2 \cos\theta_3 - (x - \rho_1)\cos\theta_2 - (\gamma - \sigma_1)\sin\theta_2}{\sigma_2 \sin\theta_3} = \left[-2, \frac{1}{2}\right] \equiv \xi,$$

then inequality (3.26) is violated by a part of ξ. We have to intersect ξ with the allowed range and to proceed with

$$\xi' = \left[-1, \frac{1}{2}\right].$$

The resulting procedure is able to identify efficiently all real solutions of Equation (3.31). This has to be done for all four combinations of signs in Equations (3.24) and (3.27) to obtain all possible loop conformations given the $n - 6$ independent variables.

Note that other interval approaches exist like the interval Newton method, see e.g. Hansen [68], which use interval derivatives and are, thus, superior to our derivative free approach in theory. However, beside the fact that the derivative is much more complex (yielding more imprecise bounds) the aforementioned approach cannot deal with complex numbers. Applying the interval Newton method would require to avoid complex interval arithmetic with range checking and interval intersection as we have shown above, which are entirely unsuited for this algorithm. It might be possible to adapt the method for our problem, but it is not necessary in practice. The limiting factor for the total run time is not the solution process for ω_1 (within milliseconds, finishes nearly instantly) but the differential evolution algorithm with scoring function evaluations afterwards.

3.5.3 Results

Our approach has been implemented in C++ using the BALL framework [101, 102, 76]. In this study, we applied our method to human 17β-hydroxysteroid dehydrogenase type 1 (17β-HSD 1) for which the protein data bank holds several structure files that have been determined at varying solutions, including different ligands and cofactors, some additionally containing single residue mutations. The loop region of 17β-HSD 1 is flexible. Thus, many structures do not incorporate this region. However, our approach needs at least one original loop conformation to calculate the initial vectors and matrices in order to guarantee that a closed loop is mathematically possible as described in Section 3.5.1. Hence, the docking experiments were performed on PDB structures 1FDT conformation A, 1FDT conformation B and 1I5R. The structure of 1I5R has been chosen due to its similarity to 1QYV, which does not contain the product of the enzymatic reaction but no loop structure as well. 1I5R

1FDT, conf. A		1FDT, conf. B		1I5R	
2 times 850 runs		1900 runs	3000 runs	1900 runs	3000 runs
2d 18h 53m	3d 8h 3m	4d 12h 39m	16d 20h 29m	4d 19h 39m	14d 23h 31m
4.721/min	5.650/min	3.431/min	8.089/min	3.652/min	7.190/min
Pro187–Pro200		His189–Ser199		Pro187–Pro200	

Table 3.13: Runtimes and mean values of the performed docking experiments.

was mapped to 1FDT in order to obtain comparable results. While all docking runs were performed on the original loop of 17β-HSD 1, different loop lengths have been chosen, to allow for additional flexibility, especially in the start and end regions of the considered sequence. The complete loop region, showing the short (blue) and extended part (red) is:

Gly186 Pro187 Val188 His189 Thr190 Ala191 Phe192 Met193
Glu194 Lys195 Val196 Leu197 Gly198 Ser199 Pro200 Glu201

Our docking experiments have shown that the resulting energies of the extended loops were slightly lower than the energies of the shorter loops, resulting from the possibility to move more freely due to additional degrees of freedom.

Estradiol was used as ligand in the docking runs, the cofactor has not been considered in this study [148]. The runtimes (AMD Opteron 3GHz) and the average number of docking runs are given in Table 3.13. Apart from a few cases where the ligand is considerably moved from its original position or it is flipped upside down, the majority of the calculated complexes shows only small changes in ligand position and orientation, the respective loop conformation being the main contributor to the resulting energy of the complex. The stability of the ligand position in combination with the realistic loop conformations calculated during optimization renders our current implementation a promising basis for fully flexible ligand-receptor docking. Figure 3.18 shows the best results in terms of the energy values, i.e. the shown complexes are the results with the lowest energy in the corresponding docking runs. They substantiate our good results against the background that, as mentioned in the previous section, the Gehlhaar scoring function does not assess the structural situation sufficiently. Thus, it is surprising that these excellent complexes are the best scoring ones.

3.5.4 Conclusion

In preliminary studies, different approaches to perform flexible docking with 17β-hydroxysteroid dehydrogenase type 1 were tried out. Precalculated torsion angles, as provided by, e.g., Dunbrack's backbone-dependent rotamer library [46], were found not to reflect the situation of this special enzyme. Furthermore, self-organizing molecular field analysis (SOMFA) [146], a 3D-QSAR approach, did not improve the results of our docking algorithm of Section 3.3, where originally the backbone is kept rigid. Our results show that the Gō-Scheraga equations with our interval arithmetic based solution approach combined with our gradient-based optimization method is able to overcome these problems with 17β-HSD 1 and to dock ligands into highly flexible loop regions. Although we applied our approach only to this specific enzyme in our study, our method is not restricted to the loop region of any protein nor to 17β-HSD 1. It works on any polypeptide chain, provided the studied part does not contain

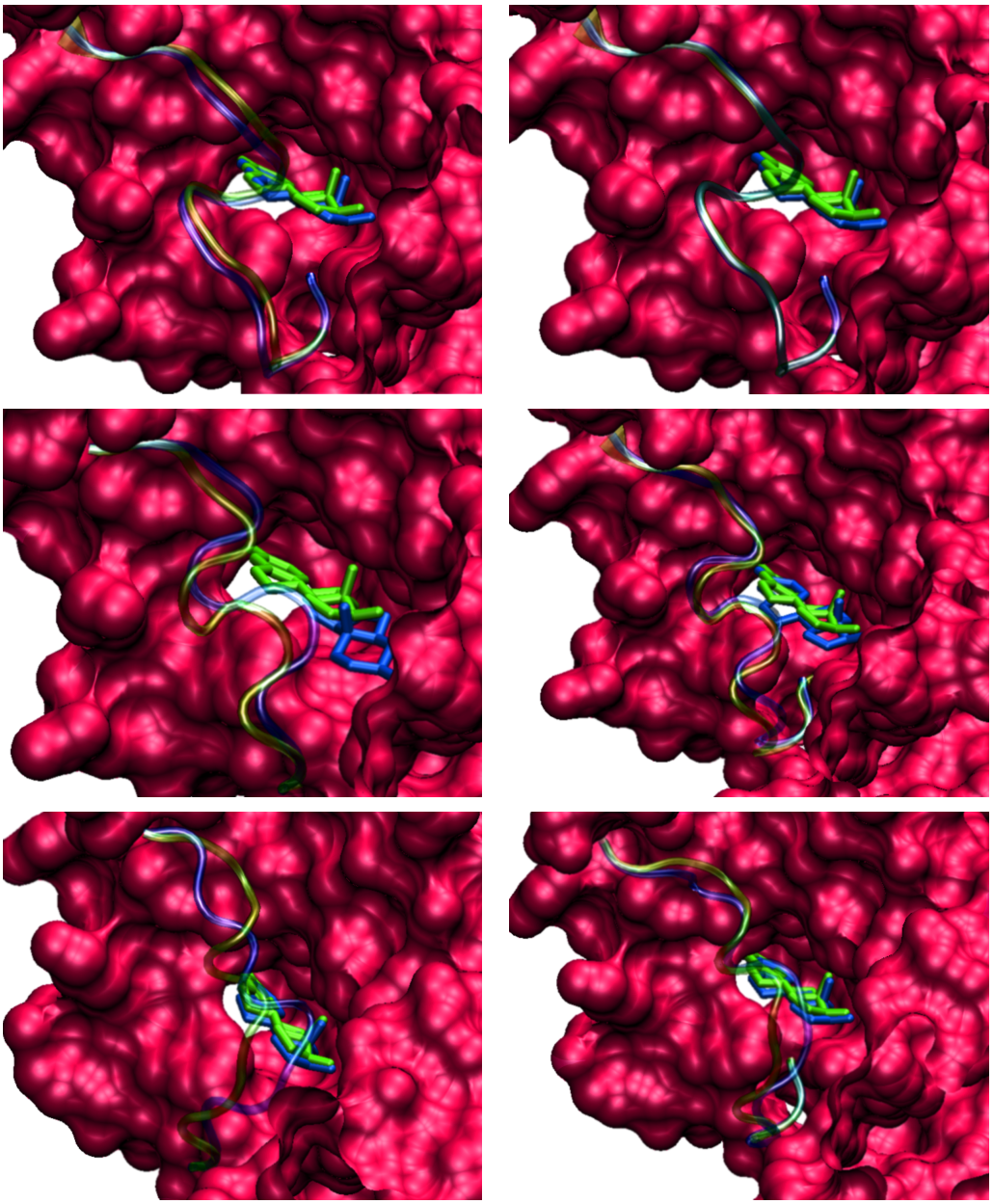

Figure 3.18: Best resulting complexes of our docking runs in terms of energy values. Original complexes are colored green, calculated complexes are colored blue. From upper left to lower right: 1FDT conf. A pass 1, 1FDT conf. A pass 2, 1FDT conf. B 1900 runs, 1FDT conf. B 3000 runs, 1I5R 1900 runs, 1I5R 3000 runs. The figures are based on [148]. Please note that in case of 1FDT conf. A pass 2 the calculated loop region matches perfectly the crystallized one.

gaps. Hence, it would be possible in theory to apply our method to a complete protein sequence and let it fold, based on its original conformation. In the future we will investigate the performance of our approach on other hard docking tasks, where especially loop flexibility is the main reason for the failure of other algorithms.

Chapter 4

Discrete Optimization

In this chapter, we focus on discrete problems in bioinformatics and how to solve them. We start in Section 4.1 with the bond order assignment problem for molecular structures. Bond order information can often not be directly inferred from the available experimental data. Even important molecular databases, like the Cambridge Structural Database [6] and the Protein Data Bank (PDB) [18, 17], are known to contain erroneous data for connectivity and bond order information [106] or to omit them entirely. For nucleic acids and proteins bond orders can easily obtained due to their building block nature, but this does not hold for other kinds of molecules like ligands. Furthermore, it is not practicable to assign bond orders manually for, e.g., virtual screening purposes, where thousands of molecules are to be considered. Hence, automated bond order assingment is often a fundamental task for the work with molecules. We present our novel linear 0-1-programming formulation for the efficient computation of all optimal and suboptimal bond order assignments based on the penalty table of Wang et al. [182]. We show that our approach outperforms not only the original method of Wang et al. [182], but also commonly used software for determining bond orders on our test set considering all results. The test set consists of 761 thoroughly prepared drug like molecules that were originally used for the validation of the Merck Molecular Force Field (MMFF94).

In the following section, we address the task of feature subset selection based on our second order mutual information criterion. In machine learning, the problem of supervised classification is concerned with using labeled examples to induce a model that classifies objects into a finite set of known classes. Avoiding irrelevant or redundant features is important because they may have a negative effect on the accuracy of the classifier. Copious classification tasks occur in bioinformatics. The examples stretch from distinguishing cancer tissues from normal tissues [7] to splice site prediction [38]. In this work, we present our filter method for feature subset selection, where our criterion is mathematically well motivated and, in contrast to other methods, exactly solved by quadratic 0-1-programming. In the validation runs, our method could achieve in 18 out of 21 test scenarios the best classification accuracies.

In the last section, we present a novel linear 0-1-programming based branch-and-cut approach for the detection of deregulated subgraphs within regulatory networks using expression profiles. The vision implicated by the proposed connectivity model is to identify – besides the most deregulated components – the root node that may represent a key player in the pathogenic process. This key player may be responsible for the observed differences between the investigated conditions and may serve as a potential target for therapy purposes. To this

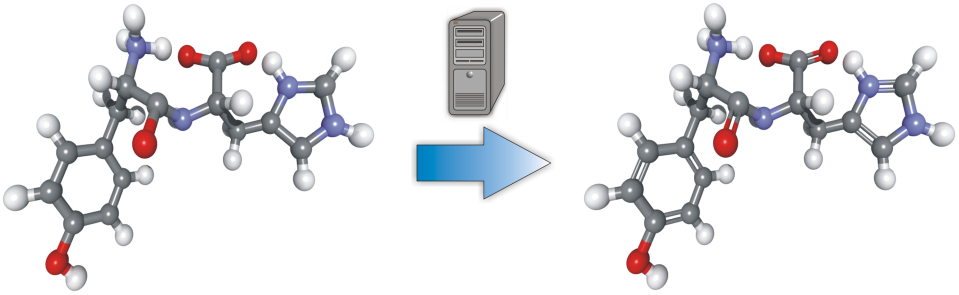

Figure 4.1: Bond order assignment. Connectivity information is symbolized by single bonds on the left.

end, we identify subgraphs with an explicit root node, i.e. all other vertices are reachable from this special node in the directed subgraph. To demonstrate the capabilities of our algorithm, we analyzed expression profiles from nonmalignant primary mammary epithelial cells derived from BRCA1 mutation carriers and epithelial cells without BRCA1 mutation. Our results suggest that oxidative stress plays an important role in epithelial cells with BRCA1 mutations that may contribute to the later development of breast cancer.

4.1 Bond Order Assignment by Linear 0-1-Programming

Copious applications in bioinformatics and computational biology process molecular structures. Thus, not only reliable atomic coordinates are essential for the corresponding algorithms, but also correct bond order information. For example, we made the acquaintance of such applications in the previous chapter. Bonds do not only define the connectivity of atoms in a molecule, but also specify structural aspects like rotatability of individual groups. Unfortunately, bond order information can often not be directly inferred from the available experimental data and are inextricably linked with the atoms' charges. Even important molecular databases, like the Cambridge Structural Database [6] and the Protein Data Bank (PDB) [18, 17], are known to contain erroneous data for connectivity and bond order information [106] or to omit them entirely. For nucleic acids and proteins bond orders can easily obtained due to their building block nature, e.g. our biochemical algorithms library (BALL) [101, 102, 76] contains routines and a database for assigning the information of interest. However, this does not hold for other kinds of molecules like ligands. Very different strategies have been developed and applied to derive bond order information, most of them relying on the correctness of the atom coordinates.

Historically the first approach has been to overcome these problems by hand curation which, certainly, does not scale well to large numbers of molecules, but provides most probably the highest reliability. Early automated methods rely on the correctness of atomic coordinates and focus on reference bond lengths and valence angles [10]. Other algorithms [75] consider additionally functional group detection and further molecular features like hybridization states and charges [175, 196]. Main disadvantages are not only their heuristic nature but also the dependence on the correctness of atomic coordinates, which is not given, e.g., in a molecular modeling environment with manually drawn molecules. Labute [106] proposed recently to

represent the addressed task as a maximum weighted matching problem for nonbipartite graphs and used an exact approach to solve it. Froeyen and Herdewijn [52] presented an integer linear program for generating valid Lewis structures (electron dot structures) with minimal formal charge on each atom.

The recent work of Wang et al. [182] deals with the problem in an elegant way: a chemically motivated, expert generated penalty function is used to score bond order assignments. Regrettably, in the provided source code the penalty function is hard coded, but in principle this approach is a good choice when the bond order assignment process has to be tuned for special kinds of molecules. The scoring function can be adapted to, e.g., highly charged atoms in a series of experimental derived molecules while the algorithm to find the solution is not affected. Wang et al. [182] optimized this scoring function heuristically. This procedure has two drawbacks: the score of the resulting assignments is not guaranteed to be optimal and the algorithm provides only one solution while there can be more than one assignment with optimal score. Recently [41], we proposed two approaches that solve the problem to global optimality. Here, we present our novel linear 0-1-programming formulation for efficient computation of all optimal and suboptimal bond order assignments and show that our approach does not only outperform the original heuristic approach of Wang et al. but also commonly used software for determining bond orders on our test set. It consists of 761 thoroughly prepared drug like molecules that were originally used for the validation of the Merck Molecular Force Field (MMFF94), see also [40].

4.1.1 The Scoring Function of Wang et al.

The idea behind the scoring function of Wang et al. [182] is to penalize unlikely bond orders, prevent impossible orders, and support frequent assignments. To this end, a score is assigned to every atom type depending on the orders of its bonds to its neighbors. In other words, its atomic valence is defined as the sum over all bond orders *bo* of all bonds connected to the atom under consideration

$$av = \sum_{i=1}^{con} bo_i,$$

where *con* denotes the number of bonded atoms. Finding the most probable consistent bond order assignment for a given molecule is then addressed by minimizing the total penalty score

$$tps = \sum_{j=1}^{n} aps_j,$$

where n denotes the number of atoms and aps_j is the penalty score that is assigned to atom j. The penalty scores are stored in a penalty table that uses a rule-based atom type classification, see Table 4.1.

4.1.2 Linear 0-1-Program

While in Wang et al. [182] minimization proceeds in a heuristic and greedy manner with the aforementioned drawbacks, our new approach solves the problem to global optimality with the techniques mentioned in Section 2.7.1. Let P be the penalty table. We use the following notations:

- A is the set of all atoms of the molecule under consideration.

No.	Atom type	av0	av1	av2	av3	av4	av5	av6	av7
1–5	H, F, Cl, Br, I	64	0	64					
6	C in C≡N–R				0	1	32		
7	C(X1)				1	0	32		
8	C in COO^-					32	0	32	
9	C			64	32	0	32	64	
10	Si					0			
11	N(X1) in N=N=R			0	0				
12	N(X1)			3	0	32			
13	N(X2) in N=N=R				1	0			
14	N(X2)			4	0	2			
15	N(X3) in nitro				64	32	0	32	
16	N(X3) in pyridine-1-oxide, etc.				1	0			
17	N(X3)			32	0	1	2		
18	N(X4)			64	0	64			
19	O(X1) in pyridine-1-oxide, etc.		0	1					
20	O(X1)		1	0	64				
21	O(X2)		32	0	64				
22	P(X1)			2	0	32			
23	P(X2)			4	0	2			
24	P(X3)			32	0	1	2		
25	P(X4) is bonded to two O(X1) or S(X1)						32	0	32
26	P(X4) is bonded to three O(X1) or S(X1)							32	0
27	P(X4)				64	1	0	32	
28	S(X1) in pyridine-1-thiol anion, etc.		0	1					
29	S(X1)		2	0	64				
30	S(X2)		32	0	32	1			
31	S(X3)				1	0	2	2	
32	S(X4) is bonded to two O(X1) or S(X1)							0	32
33	S(X4) is bonded to three O(X1) or S(X1)							32	0
34	S(X4) is bonded to four O(X1) or S(X1)							32	0
35	S(X4)					4	2	0	

Table 4.1: Atomic penalty scores for different atom types according to Wang et al. [182]. In the atom type column, X1–X4 denotes that the number of bonded atoms is 1–4, av0–av7 stands for atomic valences 0–7. Empty table entries symbolize forbidden valences.

- $B(a)$ is the set of bonds of atom $a \in A$ and B denotes the set of all bonds of the molecule.
- $V(a) \subset \mathbb{Z}_{\geq 0}$ contains the possible valences of atom $a \in A$ according to penalty table P.
- $P(a, v)$ is the entry of P for atom $a \in A$ and valence $v \in V(a)$.

Our approach uses two different classes of variables. For each bond $b \in B$ we introduce variables $x_{b,1}, \ldots, x_{b,\mu} \in \{0, 1\}$, where μ is the maximum bond order considered[1]. In other words, $x_{b,k}$ equals 1 iff the order of bond b is k. For all atoms $a \in A$ corresponding possible valences $v \in V(a)$, we introduce choice variables $y_{a,v} \in \{0, 1\}$. Each $y_{a,v}$ symbolizes whether the corresponding penalty $P(a, v)$ is chosen or not, i.e. penalty $P(a, v)$ contributes to the score iff $y_{a,v} = 1$. Thus, the objective function of our score minimization problem can be formulated as a linear function in $\mathbf{y}$ with penalty prefactors

$$\min_{\mathbf{y}} \sum_{a \in A} \sum_{v \in V(a)} P(a, v) \cdot y_{a,v}.$$

To ensure that each atom is assigned exactly one valence state, we add the additional constraints

$$\sum_{v \in V(a)} y_{a,v} = 1$$

for all $a \in A$. In addition, we have to ensure that the sum of its bond orders equals its chosen valence. These constraints can be formulated as

$$\sum_{v \in V(a)} y_{a,v} \cdot v = \sum_{b \in B(a)} \sum_{k=1}^{\mu} x_{b,k} \cdot k$$

for all $a \in A$, because both sides evaluate to the chosen valence. The uniqueness of an order assignment is guaranteed by constraints

$$\sum_{k=1}^{\mu} x_{b,k} = 1$$

for all $b \in B$. The advantage of binary variables in contrast to integer variables is that after obtaining a single solution S for the bond order assignment problem, we can prohibit this concrete solution by adding the constraint

$$\sum_{b \in B} x_{b,S(b)} \leq |B| - 1, \tag{4.1}$$

where $S(b)$ denotes the order of bond b in the solution S to be forbidden. Solving the total system again will yield another, perhaps also optimal solution or one of the "best suboptimal" solutions. In summary, the score minimization problem can be formulated as the following

[1] In the following, we will set μ to 3, allowing single, double, and triple bonds.

linear 0-1-program

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{y}} \quad & \sum_{a \in A} \sum_{v \in V(a)} P(a,v) \cdot y_{a,v} \\
\text{s.t.} \quad & \sum_{v \in V(a)} y_{a,v} \cdot v = \sum_{b \in B(a)} \sum_{k=1}^{\mu} x_{b,k} \cdot k && \forall a \in A, \\
& \sum_{v \in V(a)} y_{a,v} = 1 && \forall a \in A, \\
& \sum_{k=1}^{\mu} x_{b,k} = 1 && \forall b \in B, \\
& y_{a,v} \in \{0,1\} && \forall a \in A,\ \forall v \in V(a), \\
& x_{b,k} \in \{0,1\} && \forall k = 1, \ldots, \mu, \forall b \in B.
\end{aligned}
$$

As mentioned in Section 2.7.1, several strategies can be chosen to solve our linear 0-1-program to provable global optimality. We employed CPLEX in this work, a commercial linear and quadratic integer solver. It is interesting to note that the penalties in Table 4.1 (almost) all can be expressed as powers of two and as such led to short computation times. Still, the problem itself is NP complete [137]. However, in many test cases the solution of the relaxed linear program, i.e. the above program without the 0-1-constraints, has been integral and, hence, a solution of the original problem (obtained without any branching). In other cases, the solution of the linear program has been almost integral, leading to only a few branching steps. Thus, our approach is well suited for obtaining one optimal bond order assignment. By adding constraints (4.1), we obtain all optimal and suboptimal solutions. It is interesting to note that CPLEX provides a so-called "repair heuristic" and techniques to obtain a feasible solution for the new problem based on the solution of the old problem. In other words, adding a constraint (4.1) does not require to solve the derived problem from scratch, but CPLEX is able to perform a warm start and only a few iterates lead to a globally optimal solution anew. Thus, all optimal and suboptimal solutions can be enumerated efficiently.

4.1.3 Results

The entire code that was used in this work for the modeling of atoms, bonds, and molecules was generated using the C++ library BALL [101, 102, 76]. We employed CPLEX[2], version 11.1.1, as solver in this work. We chose to compare our approach to the implementation of Wang et al. [182], called Antechamber, as well as to widely used software like OpenBabel[3], version 2.2.0, and I-interpret[4] [196], version 1.0. OpenBabel is an open source software originally intended for the conversion between different molecular file formats and has to, hence, due to the lack of bond order information in some file standards, add and assign bonds and bond orders. I-interpret relies, like OpenBabel, on correct bond lengths, hybridization states, etc. and has been recently developed. Regrettably, we were not able to initialize the bond order assignment process of OpenBabel and I-interpret without deleting all bond information from the tested molecules, i.e. both algorithms had to detect where bonds are and

[2] `http://www-01.ibm.com/software/integration/optimization/cplex/`
[3] `http://openbabel.org`
[4] `http://www.sioc-ccbg.ac.cn/software/I-interpret/`

where not. Fortunately, both products were able on our test set, the MMFF94 validation suite [63, 64, 65, 66, 67], to reconstruct (most) bonds. To be exact, OpenBabel failed to detect the connectivity information of 1 molecule, I-interpret went wrong on 6 of 761 molecules of the MMFF94 validation suite. Both numbers are, fortunately, substantially small in contrast to the 761 molecules tested in our validation runs. Thus, the comparability between the different algorithms is ensured. The MMFF94 validation suite consists of 761 thoroughly prepared drug like molecules with well defined distances, bond angles, etc.. Hence, the comparison of our approach to OpenBabel and I-interpret is legitimate, since both methods strongly rely on correct bond angles and lengths. In our test runs, we used the penalty Table 4.1 as defined in Wang et al. [182]. Certainly, it is solver dependent which solution is returned if more than one global optimum solution exist. Therefore, we calculated all optimal solutions and compared which coincides with the original bond orders. Table 4.2 shows our results. It must be noted that Antechamber returned non-optimal solutions in 6 cases and is not able to produce more than one solution. Additionally, OpenBabel and I-interpret can calculate only one bond order assignment due to their approach.

Method	Reference is		No Solution	Connectivity
	1st solution	optimal		not determined
Antechamber	282 (37.05%)	282 (37.05%)	18 (2.36%)	–
OpenBabel	509 (66.87%)	509 (66.87%)	0	1 (0.13%)
I-interpret	546 (71.75%)	546 (71.75%)	0	6 (0.79%)
Our method	409 (53.75%)	**600 (78.84%)**	4 (0.53%)	–

Table 4.2: Performance of the original Antechamber implementation, OpenBabel, I-interpret, and our linear 0-1-programming formulation on the MMFF94 validation suite. The second column denotes the number of molecules for which the algorithm returns the original bond order assignment as first solution. The third column denotes the number of cases, where the reference bond order assignment was within the solutions with minimal total penalty score. The fourth column denotes the number of molecules for which no solution was found. The last column gives the number of molecules for which the original connectivity information could not be achieved, see text.

As shown in Table 4.2, I-interpret is a very promising tool for assigning bond orders. It could even slightly outperform the widely used OpenBabel. Both methods are clearly superior considering only the first solutions obtained by our approach. However, the test set contains many (kekulized) ring structures where a unique bond order assignment does not exist. Thus, we have to compare the results to all optimal solutions and find that the successive enumeration leads to a success rate of 78.84%. Our method is able to significantly reproduce more molecules of the MMFF94 validation suite than the other approaches. Table 4.3 gives information about how many correct bond order assignments occurred with the number of their solution. For example, in 409 cases the first calculated solution led to correct assignments. If the first solution has not been the reference assignment, in 142 cases the second computed solution succeeded, etc.. In summary, we only needed to calculate up to 8 solutions in order to obtain a success rate of 78.84%.

Obviously, the quality of the penalty table, e.g. the definition of the atom classes, their allowed valence states, and the choice of the valence state's penalties have a significant influence on the performance of our method. Table 4.2 shows that the current penalty table does not cover all molecules in the MMFF94 validation suite – for four molecules some of the required

Solution returned	1st	2nd	3rd	4th	5th	6th	8th
No. of correct bond order ass.	409	142	21	19	3	4	2

Table 4.3: Number of correct bond order assignments with the returned solutions, e.g. in 142 cases the second solution has been the reference assignment.

atom classes are missing. It is important to note that the difference to the Antechamber bailing out rate is a result of the heuristic nature of the optimization proposed in Wang et al. [182].

4.1.4 Conclusion

Our method improves considerably upon earlier approximate solution schemes (Antechamber) by guaranteeing optimality. Our results are surprising in comparison to OpenBabel and I-interpret. Both approaches strongly rely on bond lengths and angles and, thus, should be optimally suited for our test set, the MMFF94 validation suite.

Due to our work, the limiting factor is no longer the algorithm to find an assignment but the scoring function behind. Wang et al. have introduced atom types and penalty scores for each type. New atom types and adapted scores may further improve the results [39]. However, the development of such scoring functions is beyond the scope of this work. Most probably, the best results will be achieved with specialized scores for certain classes of molecules.

4.2 Optimal Feature Selection

Discriminant analysis is widely used in bioinformatics, such as distinguishing cancer tissues from normal tissues [7] or one cancer subtype vs another [5], predicting protein fold or superfamily from its sequence [85, 43], etc.. Avoiding irrelevant or redundant features is important because they may have a negative effect on the accuracy of the classifier. Many features may introduce noise without carrying information about the class labels. Thus, crucial for these classification tasks is feature selection. Instead of using all available features, only a subset is employed for the classification. Feature selection is accompanied by several advantages: (1) reduction of overfitting of the used learning methods and, hence, improvement of the classification accuracy, (2) the obtained features are more interpretable that can help identifying and monitoring the target diseases or function types, and, finally, (3) dimension reduction decreases the computational costs for the classification algorithms.

Feature selection methods can be classified into two types, wrappers and filters [35, 100]. In wrapper type methods, feature selection is "wrapped" around a learning method. Often, a set with very small number of non-redundant features can be obtained [100], which gives high accuracy, because the characteristics of the features match well with the characteristics of the learning method. However, wrapper methods have one major drawback: computational costs. They typically require extensive calculations to select the best features.

The second class of algorithms, filters, is classifier independent, as filters are not dedicated to a specific type of classification method. Essentially, they are a kind of data preprocessing which selects features based on intrinsic characteristics determining their discriminant capabilities with regard to the class labels. The main advantage over wrappers is that they do not need to use the classification algorithm for their selection and are usually computationally

cheaper. Here, we present our filter method that uses second order information [153], see Section 4.2.1, while other methods strongly rely only on first order information, see Sections 4.2.2 to 4.2.4. Furthermore, our criterion is mathematically well motivated and, in contrast to other methods, exactly solved by quadratic 0-1-programming, see also Section 2.7.2, and not approximated.

4.2.1 Second Order Mutual Information Criterion

The "best" feature subset often means a minimal classification error. In an unsupervised situation where the classifiers are not specified, minimal errors usually require the maximal statistical dependency of the class labels on the features and vice versa. This is usually characterized in terms of correlation or mutual information, which is a widely used measure for the dependency between random variables. It is defined as

$$I(\mathbf{a};\mathbf{b}) = H(\mathbf{a}) + H(\mathbf{b}) - H(\mathbf{a},\mathbf{b}),$$

where $H(\mathbf{a})$ is the entropy of $\mathbf{a}$. In terms of mutual information we can express the problem of finding a feature subset $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ with m features, which have maximal statistical dependency on the class labels $\mathbf{y}$ by

$$\begin{aligned} &\max_S I(S;\mathbf{y}) \\ &\text{s. t. } |S| = m. \end{aligned}$$

Obviously, for $m = 1$ the solution is the feature j that maximizes $I(\mathbf{x}_j;\mathbf{y})$. We have

$$\begin{aligned} I(S;\mathbf{y}) =& \frac{1}{m}\sum_{i=1}^{m} I(\mathbf{x}_i;\mathbf{y}) + \frac{1}{(m-1)\binom{m}{1}}\sum_{j\neq i} I(\mathbf{x}_i;\mathbf{y}|\mathbf{x}_j) \\ &+ \frac{1}{(m-2)\binom{m}{2}}\sum_{\substack{j<k,\\ i\neq j,\\ i\neq k}} I(\mathbf{x}_i;\mathbf{y}|\mathbf{x}_j,\mathbf{x}_k) + \cdots + \frac{1}{m}\sum_{i=1}^{m} I(\mathbf{x}_i;\mathbf{y}|S\backslash\{\mathbf{x}_i\}). \end{aligned}$$

All approaches we present in the subsequent sections approximate $I(S,\mathbf{y})$ only by first order information, i.e. $I(\mathbf{x}_i;\mathbf{y})$, all remaining terms are neglected. Here, we present our approach to use second order information.

Theorem 4.2.1. *It holds*

$$I(S;\mathbf{y}) = \frac{1}{\binom{m}{2}}\sum_{i<j}^{m} I((\mathbf{x}_i,\mathbf{x}_j);\mathbf{y}) + \sum_{r=2}^{m-1}\frac{1}{(m-r)\binom{m}{r}}\sum_{\substack{k_1<k_2<\ldots k_r,\\ k_{r+1}\neq k_i, i=1,\ldots,r}} I(\mathbf{x}_{k_{r+1}};\mathbf{y}|\mathbf{x}_{k_1},\ldots,\mathbf{x}_{k_r}).$$

Proof. We have

$$\begin{aligned} I(S;\mathbf{y}) &= I(\mathbf{x}_1;\mathbf{y}) + I((\mathbf{x}_1,\mathbf{x}_2);\mathbf{y}) - I(\mathbf{x}_1;\mathbf{y}) + \cdots + I(S;\mathbf{y}) - I((\mathbf{x}_1,\ldots,\mathbf{x}_{m-1});\mathbf{y}) \\ &= I(\mathbf{x}_1;\mathbf{y}) + I(\mathbf{x}_2;\mathbf{y}|\mathbf{x}_1) + I(\mathbf{x}_3;\mathbf{y}|\mathbf{x}_1\mathbf{x}_2) + \cdots + I(\mathbf{x}_m;\mathbf{y}|\mathbf{x}_1,\ldots,\mathbf{x}_{m-1}). \end{aligned}$$

Reordering and counting the terms $I(\mathbf{x}_i;\mathbf{y})$ yield

$$\begin{aligned} I(S;\mathbf{y}) &= I((\mathbf{x}_1,\mathbf{x}_2);\mathbf{y}) + I((\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3);\mathbf{y}) - I((\mathbf{x}_1,\mathbf{x}_2);\mathbf{y}) + \cdots \\ &\quad \cdots + I(S;\mathbf{y}) - I((\mathbf{x}_1,\ldots,\mathbf{x}_{m-1});\mathbf{y}) \\ &= I((\mathbf{x}_1,\mathbf{x}_2);\mathbf{y}) + I(\mathbf{x}_3;\mathbf{y}|\mathbf{x}_1,\mathbf{x}_2) + \cdots + I(\mathbf{x}_m;\mathbf{y}|\mathbf{x}_1,\ldots,\mathbf{x}_{m-1}). \end{aligned}$$

The choice of the sequence of random variables is arbitrary. Thus, we can average over all possible permutations of $\mathbf{x}_1,\ldots,\mathbf{x}_m$. We have for k variables $\binom{m}{k}$ different possibilities because we do not consider the order of the random variables. We can choose among all remaining $(m-k)$ random variables. Thus, the number of possibilities of the term $I(\mathbf{x}_{k_{r+1}};\mathbf{y}|\mathbf{x}_{k_1},\ldots,\mathbf{x}_{k_r})$ is given by $(m-r)\binom{m}{r}$. □

Based on the above observation we propose to maximize

$$\begin{aligned} &\max_S \sum_{\substack{\mathbf{x}_i,\mathbf{x}_j \in S \\ i\neq j}} I((\mathbf{x}_i,\mathbf{x}_j);\mathbf{y}) \\ &\text{s. t. } |S| = m, \end{aligned} \tag{4.2}$$

where we use second order information. We solve (4.2) to provable global optimality by modeling the problem as quadratic 0-1-program and with the techniques described in Section 2.7.2. We introduce binary 0-1 decision variables x_i for each feature $\mathbf{x}_i$ meaning feature $\mathbf{x}_i$ is selected iff $x_i = 1$. Thus, with

$$\mathbf{A} = \begin{pmatrix} 0 & \frac{1}{2}I((\mathbf{x}_1,\mathbf{x_2});\mathbf{y}) & \cdots & \frac{1}{2}I((\mathbf{x}_1,\mathbf{x}_n);\mathbf{y}) \\ \frac{1}{2}I((\mathbf{x}_1,\mathbf{x_2});\mathbf{y}) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{1}{2}I((\mathbf{x}_{n-1},\mathbf{x}_n);\mathbf{y}) \\ \frac{1}{2}I((\mathbf{x}_1,\mathbf{x}_n);\mathbf{y}) & \cdots & \frac{1}{2}I((\mathbf{x}_{n-1},\mathbf{x}_n);\mathbf{y}) & 0 \end{pmatrix}$$

problem (4.2) is equivalent to the quadratic 0-1-program

$$\begin{aligned} &\min_{\mathbf{x}} \; -\mathbf{x}^T\mathbf{A}\mathbf{x} \\ &\text{s. t. } \sum_{i=1}^{n} x_i = m, \\ &\qquad x_i \in \{0,1\}. \end{aligned} \tag{4.3}$$

The matrix $-\mathbf{A}$ is obviously not positive semidefinite. Thus, we have to perform the semidefinite preprocessing described in Section 2.7.2. Afterwards, we use CPLEX[5], version 11.1.1, as solver for the convexified quadratic 0-1-program. It is interesting to note that if we sample m, we do not need to solve (4.3) for every m from scratch. The (convexified) objective function does not change when m has another value. Furthermore, CPLEX provides a so-called "repair heuristic" and techniques to obtain a feasible solution for the new problem based on the solution of the old problem. In other words, if m is changed in (4.3), the derived problem does not have to be solved from scratch. CPLEX is able to perform a warm start.

[5] `http://www-01.ibm.com/software/integration/optimization/cplex/`

4.2.2 Max-Relevance and Min-Redundancy Criterion

The approach of Peng et al. [140] is to maximize the relevance while minimizing the redundancy. Let S be the feature set with m features $\mathbf{x}_i$ (to be determined) and $\mathbf{y}$ the class labels. Maximum relevance is according to Peng et al. [140] a first order criterion

$$\begin{aligned} &\max_S \frac{1}{m} \sum_{\mathbf{x}_i \in S} I(\mathbf{x}_i; \mathbf{y}) \\ &\text{s. t. } |S| = m. \end{aligned}$$

Since it is likely that features selected according to this criterion could have rich redundancy, i.e. the dependency among these features could be large, they further give the minimal redundancy criterion[6]

$$\begin{aligned} &\min_S \frac{1}{m \cdot (m-1)} \sum_{\mathbf{x}_i \in S} \sum_{\substack{\mathbf{x}_j \in S \\ \mathbf{x}_i \neq \mathbf{x}_j}} I(\mathbf{x}_i; \mathbf{x}_j) \\ &\text{s. t. } |S| = m, \end{aligned} \tag{4.4}$$

and combine both to their max-relevance and min-redundancy criterion

$$\begin{aligned} &\max_S \frac{1}{m} \sum_{\mathbf{x}_i \in S} I(\mathbf{x}_i; \mathbf{y}) - \frac{1}{m \cdot (m-1)} \sum_{\mathbf{x}_i \in S} \sum_{\substack{\mathbf{x}_j \in S \\ \mathbf{x}_i \neq \mathbf{x}_j}} I(\mathbf{x}_i; \mathbf{x}_j) \\ &\text{s. t. } |S| = m. \end{aligned} \tag{4.5}$$

They give an incremental search method to find near optimal features. Let X be the set of all features. Based on the set S_{m-1} of $m-1$ already selected features, they choose the mth feature by determining

$$\max_{\mathbf{x}_j \in X \setminus S_{m-1}} \left[I(\mathbf{x}_j; \mathbf{y}) - \frac{1}{m-1} \sum_{\mathbf{x}_i \in S_{m-1}} I(\mathbf{x}_j; \mathbf{x}_i) \right].$$

4.2.3 Globally Optimal Max-Relevance-Min-Redundancy

Peng et al. [140] give only an incremental approach for finding near optimal feature sets. However, we can solve the optimization problem (4.5) exactly by quadratic 0-1-programming similar to our approach in Section 4.2.1. We introduce binary 0-1 decision variables x_i for every feature $\mathbf{x}_i$ meaning feature $\mathbf{x}_i$ is selected iff $x_i = 1$. Thus, with

$$\mathbf{A} = \begin{pmatrix} I(\mathbf{x}_1; \mathbf{y}) & -\frac{1}{2}\frac{1}{m-1} I(\mathbf{x}_1; \mathbf{x_2}) & \cdots & -\frac{1}{2}\frac{1}{m-1} I(\mathbf{x}_1; \mathbf{x}_n) \\ -\frac{1}{2}\frac{1}{m-1} I(\mathbf{x}_1; \mathbf{x_2}) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -\frac{1}{2}\frac{1}{m-1} I(\mathbf{x}_{n-1}; \mathbf{x}_n) \\ -\frac{1}{2}\frac{1}{m-1} I(\mathbf{x}_1; \mathbf{x}_n) & \cdots & -\frac{1}{2}\frac{1}{m-1} I(\mathbf{x}_{n-1}; \mathbf{x}_n) & I(\mathbf{x}_n; \mathbf{y}) \end{pmatrix}$$

[6] Please note that we corrected an obvious mistake in [140], because minimizing the entropy of a feature is useless.

problem (4.5) is equivalent to the quadratic 0-1-program

$$\begin{aligned} \min_{\mathbf{x}} \; & -\mathbf{x}^T \mathbf{A} \mathbf{x} \\ \text{s. t.} \; & \sum_{i=1}^{n} x_i = m, \\ & x_i \in \{0,1\}, \end{aligned}$$

which can be solved by the semidefinite convexification approach in Section 2.7.2 followed by a branch-and-cut method. Please note that the computational advantages of our method do not hold in this case if m is sampled. Changing m means modifying $\mathbf{A}$ and, thus, the semidefinite program solver has to be called once m is changed. Finally, CPLEX cannot perform a warm start but has to solve each problem from scratch.

4.2.4 The Conditional Mutual Information Method

Fleuret [51] proposes a feature selection method based on the conditional mutual information

$$\begin{aligned} I(X;Y|Z) &= I((X,Z);Y) - I(Z;Y) \\ &= H(X,Z) + H(Y) - H(X,Y,Z) - H(Z) - H(Y) + H(Y,Z) \\ &= H(X,Z) - H(X,Y,Z) - H(Z) + H(Y,Z). \end{aligned}$$

The main goal of his feature selection is to select a small subset of features that carries as much information as possible. His approach deals with the tradeoff between individual power and independence by comparing each new feature with the ones already picked. He assumes a feature $\mathbf{x}'$ is good only if $I(\mathbf{y};\mathbf{x}'|\mathbf{x})$ is large for every $\mathbf{x}$ already picked. In other words, $\mathbf{x}'$ is good only if it carries information about $\mathbf{y}$, and if this information has not been caught by any of the $\mathbf{x}$ already picked. Let X be the set of all features. Based on the set S_{m-1} of $m-1$ already selected features $\mathbf{x}_i$ he proposed to choose the mth feature by

$$\max_{\mathbf{x}_i \in X \setminus S_{m-1}} \left[\min_{\mathbf{x}_j \in S_{m-1}} I(\mathbf{y};\mathbf{x}_i|\mathbf{x}_j) \right], \tag{4.6}$$

where the first feature is chosen by

$$\max_{\mathbf{x}_i \in X} I(\mathbf{y};\mathbf{x}_i).$$

In his work [51], he focuses on binary features and shows how this can be exploited for a very efficient implementation. However, on his homepage we could find an erratum that admits that in his implementation not (4.6) is used but

$$\max_{\mathbf{x}_i \in X \setminus S_{m-1}} \left[\min \left\{ I(\mathbf{y};\mathbf{x}_i), \min_{\mathbf{x}_j \in S_{m-1}} I(\mathbf{y};\mathbf{x}_i|\mathbf{x}_j) \right\} \right]. \tag{4.7}$$

For our test runs we implemented (4.6) as well as (4.7).

4.2.5 Class Prediction Methods

All presented feature selection methods do not convolve with specific classifiers. In this section, we give a short survey of the classifiers we tested and our improvements for parameter optimization.

4.2.5.1 Naïve-Bayes Classifier

The Naïve Bayes (NB) [119] is one of the earliest classifiers. The idea is to apply Bayes rule under the assumption that the features are independent of each other, given the target classes. Let $\mathbf{s} = (s_1, \ldots, s_m)$ be a sample of m features, then the posterior probability that $\mathbf{s}$ belongs to class c_k is

$$p(c_k \mid \mathbf{s}) \propto \prod_{i=1}^{m} p(s_i \mid c_k), \tag{4.8}$$

where $p(s_i \mid c_k)$ is the conditional probability table (or densities) learned from examples in the training process. In order to assign a sample $\mathbf{s}$ to a class, this classifier evaluates Formula (4.8) for all possible class labels and returns the label with the highest probability. Although there exist many examples where the independence assumption fails, NB shows classification results comparable to other more sophisticated classifiers on many real data sets [119].

4.2.5.2 Support Vector Machine

Support vector machines (SVMs) have been developed by Cortes and Vapnik [32] for binary classification. Basically, SVMs look for the optimal separating hyperplane between the two classes by maximizing the so-called *margin* between the closest points of each class, see Figure 4.2. The points on the boundaries are called *support vectors*, and the middle of the margin is the optimal separating hyperplane. If both classes are not separable, soft margin techniques are applied, i.e. data points on the "wrong" side of the discriminant margin are allowed by additional variables that measure the degree of misclassification. The influence of allowed misclassification can be adjusted by a parameter C. The training of a SVM consists of finding the optimal hyperplane. Data points are classified afterwards by evaluating on which side the points are located. We used `libsvm` [27], which also supports multi-class classification using a one-against-one technique by fitting all binary subclassifiers and finding the correct class by a voting mechanism.

This original optimal hyperplane algorithm is a linear classifier. In 1992, Boser et al. [22] suggested a way to create non-linear classifiers by applying the kernel trick to maximum-margin hyperplanes. In the resulting algorithm, every occurring dot product $\mathbf{x}_i^T \mathbf{x}_j$ is replaced by a non-linear kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$. This allows the algorithm to find the optimal separating hyperplane in a transformed feature space. The embedding function may be highly non-linear and the transformed space high dimensional. Hence, the classifier may be non-linear in the original space. In this work, we used the linear kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$

and the Gaussian radial basis function (RBF)

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \cdot \|\mathbf{x}_i - \mathbf{x}_j\|^2\right).$$

Before training and classifying with a SVM it is not known a-priori which values of the above mentioned parameters C and γ are the best for the classification problem and, hence, some kind of model selection (parameter search) has to be done. To avoid overfitting, our parameter optimization approach is a cross-validation procedure. The authors of `libsvm` recommend a grid search on C and γ. They found that trying exponential growing sequences of C and γ is a practical method to identify good parameters, e.g. $C = 2^{-5}, 2^{-3}, \ldots, 2^{15}$,

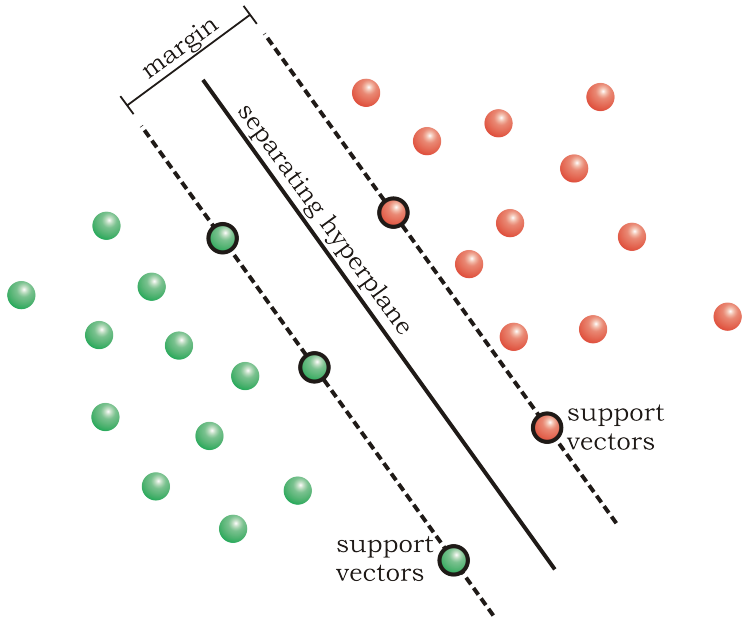

Figure 4.2: Classification by separating hyperplane. Depicted is the linear separable case.

$\gamma = 2^{-15}, 2^{-13}, \ldots, 2^3$. As they recommend, we first use a coarse grid and after identifying a "better" region a finer grid search on that region is performed. However, each cross-validation run is expensive. Thus, we improved the grid search approach by a concave overestimater approach.

Certainly, we want to maximize the accuracy or – equivalently – to minimize (-accuracy). The concave overestimator for the accuracy is equivalent to the convex underestimator for (-accuracy). The semidefinite underestimater approach is a technique for the (heuristic) global minimization of an unknown function f. A quadratic function

$$q(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} + c$$

is constructed, which systematically underestimates known positions, i.e.

$$q(\mathbf{x}_i) \leq f(\mathbf{x}_i) \tag{4.9}$$

$i = 1, \ldots, m$ for all m positions where $f(\mathbf{x}_i)$ is already known, see Figure 4.3. The idea is to force $\mathbf{A}$ to be positive definite and, hence, q is a convex function. The minimum of q is then used as new trial position. The aim is to choose q in a way that q is a very tight underestimator. To be more precise it is chosen in a way that

$$\sum_{i=1}^{m} (f(\mathbf{x}_i) - q(\mathbf{x}_i))$$

is minimal, subject to Inequalities (4.9) and

$$\mathbf{A} \succ 0,$$

which is achieved by solving a semidefinite program, i.e.

$$\mathbf{A}' \succeq 0$$

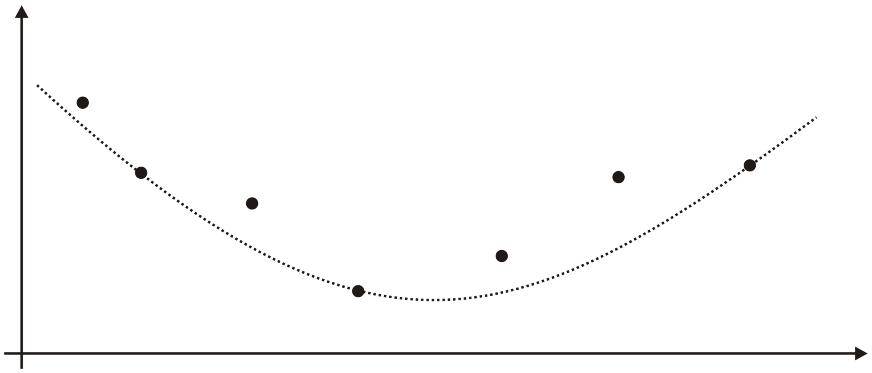

Figure 4.3: Quadratic underestimater (dashed line) of given function values (black circles).

and using afterwards

$$\mathbf{A} = \mathbf{A}' + \varepsilon\mathbf{I},$$

where $\varepsilon > 0$ is a lower bound of the smallest eigenvalue of $\mathbf{A}$. Our SDP formulation is quite similar to that of Paschalidis et al. [139] and Marcia et al. [112] but we do not set explicit bounds on $\mathbf{b}$ and c, and we force $\mathbf{A}$ to be positive definite. We introduce vectors $\mathbf{b}^+, \mathbf{b}^- \geq 0$, scalars $c^+, c^- \geq 0$ satisfying $\mathbf{b} = \mathbf{b}^+ - \mathbf{b}^-$ and $c = c^+ - c^-$, and slack variables s_i with

$$s_i = f(\mathbf{x}_i) - q(\mathbf{x}_i),$$

$i = 1, \ldots, m$. Since

$$\mathbf{x}_i^T\mathbf{A}\mathbf{x}_i = \left\langle \mathbf{x}_i\mathbf{x}_i^T, \mathbf{A} \right\rangle = \left\langle \mathbf{x}_i\mathbf{x}_i^T, \mathbf{A}' + \varepsilon\mathbf{I} \right\rangle = \left\langle \mathbf{x}_i\mathbf{x}_i^T, \mathbf{A}' \right\rangle + \varepsilon\mathbf{x}_i^T\mathbf{x}_i,$$

where $\langle \mathbf{C}, \mathbf{D} \rangle = \mathrm{vec}(\mathbf{C})^T\mathrm{vec}(\mathbf{D})$ and $\mathrm{vec}(\mathbf{C})$ is the "vectorization" of a matrix $\mathbf{C}$, we have constraints

$$\begin{aligned} s_i &= f(\mathbf{x}_i) - \frac{1}{2}\left\langle \mathbf{x}_i\mathbf{x}_i^T, \mathbf{A}' \right\rangle - (\mathbf{b}^+ - \mathbf{b}^-)^T\mathbf{x}_i - (c^+ - c^-) - \frac{1}{2}\varepsilon\mathbf{x}_i^T\mathbf{x}_i, \\ s_i &\geq 0, \end{aligned}$$

$i = 1, \ldots, m$ and we want to minimize

$$\sum_{i=1}^{m} s_i.$$

Note that in our case $\mathbf{b}^+ = (b_C^+, b_\gamma^+)^T$, $\mathbf{b}^- = (b_C^-, b_\gamma^-)^T$, and $\mathbf{x}_i = (x_{i,C}, x_{i,\gamma})^T$ since we only want to optimize two variables C and γ (in case of the Gaussian radial basis kernel, only C in case of the linear kernel). Using the block diagonal matrices

$$\mathbf{X} = \begin{pmatrix} \mathbf{A}' & & & & & & & & \\ & b_c^+ & & & & & & 0 & \\ & & b_\gamma^+ & & & & & & \\ & & & b_C^- & & & & & \\ & & & & b_\gamma^- & & & & \\ & & & & & c^+ & & & \\ & & & & & & c^- & & \\ & & & & & & & s_1 & \\ & & & & & & & & \ddots \\ & 0 & & & & & & & \\ & & & & & & & & s_m \end{pmatrix},$$

$$\mathbf{C} = \begin{pmatrix} 0 & & & & & & & & & \\ & 0 & & & & & & & 0 & \\ & & 0 & & & & & & & \\ & & & 0 & & & & & & \\ & & & & 0 & & & & & \\ & & & & & 0 & & & & \\ & & & & & & 0 & & & \\ & & & & & & & -1 & & \\ & 0 & & & & & & & \ddots & \\ & & & & & & & & & -1 \end{pmatrix},$$

$$\mathbf{A}_i = \begin{pmatrix} \frac{1}{2}\mathbf{x}_i\mathbf{x}_i^T & & & & & & & & & \\ & x_{i,C} & & & & & & & 0 & \\ & & x_{i,\gamma} & & & & & & & \\ & & & -x_{i,C} & & & & & & \\ & & & & -x_{i,\gamma} & & & & & \\ & & & & & 1 & & & & \\ & & & & & & -1 & & & \\ & & & & & & & 0 & & \\ & 0 & & & & & & & 1_i & \\ & & & & & & & & & 0 \end{pmatrix},$$

$i = 1, \ldots, m$, where 1_i denotes that all entries corresponding to $s_1, \ldots, s_m$ in $\mathbf{X}$ are zero except for s_i where the entry is one, and the values

$$b_i = f(\mathbf{x}_i) - \frac{1}{2}\varepsilon\mathbf{x}_i^T\mathbf{x}_i,$$

we have a semidefinite program in primal standard form

$$\begin{aligned} \min_{\mathbf{X}\in\mathbb{S}^n} \ & \langle \mathbf{C}, \mathbf{X} \rangle \\ \text{s. t. } & \langle \mathbf{A}_i, \mathbf{X} \rangle = b_i, \quad i = 1, \ldots, m, \\ & \mathbf{X} \succeq 0. \end{aligned}$$

In summary, after a grid search in the current region we used this approach to calculate a further trial position (minimum of q). While not substantially increasing the total run time (the semidefinite programs are solved very fast by the software library CSDP [21]), our approach may be able to save cross-validation runs.

4.2.6 Results

We used `libsvm` [27] for the support vector machines, CSDP [21] as semidefinite program solver, and CPLEX[7] as solver for the convexified quadratic 0-1-program. The remaining routines and methods have been implemented from scratch in C and C++. Since most of the necessary calculations are independent, our implementation has been parallelized where possible. The parameter optimization process for the support vector machines has been parallelized using Intel® Threading Building Blocks[8], which is a library that offers a rich and

[7] `http://www-01.ibm.com/software/integration/optimization/cplex/`
[8] `http://www.threadingbuildingblocks.org`

complete approach to expressing parallelism in a C++ program. Thus, the cross-validation runs for parameter optimization used all 4 cores of our Intel Core2 Quad, 2.83GHz. The task of generating the matrices in (4.3) means essentially counting the number of occurrences for the probabilities. This can be parallelized in short threads, but there are many of them. Thus, we decided to employ GPU computing and used NVIDIA's Compute Unified Device Architecture (CUDA)[9]. This way, the generation time of a 700×700 test matrix could be reduced from about 15 minutes (serial, not optimized, CPU) to 14 seconds on our NVIDIA GeForce 9800 GT with identical results. Furthermore, we implemented the Naïve Bayes classifier in a highly parallel variant using CUDA.

We compared the different methods on publically available data sets from the UCI machine learning repository[10]. Some data sets provide separate trainings and test data. If this has been the case, we used them as in the repository. For data sets without test data, we evaluated the approaches by 10-fold cross-validation. If data has not been categorical, we preprocessed them with a binning procedure with bin size (number of samples)$^{\frac{1}{3}}$. Thus, the mutual information could be estimated by counting on the categorical data. We used the following data sets:

- Soybean (large) [118] with 35 categorical attributes and 307 instances. This data set contains trainings and test data.
- Molecular Biology (Splice-junction Gene Sequences) [134] with 60 categorical attributes and 3190 instances.
- Breast Cancer Wisconsin (Original) [111] with 10 integer attributes and 699 instances.
- Synthetic Control Chart Time Series Data Set [141] with 60 real attributes and 600 instances.
- MONK's Problems [171] with 6 categorical attributes and 432 instances. This data set consists of three different classification problems, each with trainings and test data, and was the basis of a first international comparison of learning algorithms.

In summary, we evaluated and compared the feature subset selection approaches on 21 test scenarios: 7 data sets with 3 classifiers each (SVM with linear kernel, SVM with RBF kernel, and NB classifier). Figures A.1 to A.21 in Appendix A show our classification results.

Our method could achieve in 8 scenarios a better classification accuracy than all other methods. In the remaining 13 scenarios, the best accuracy could be accomplished in 10 cases together with other approaches. In summary, the results show that our method led to a total of 18 cases out of 21 scenarios where the best classification accuracy could be achieved. Another observation is that the approach of Peng et al. [140] lives up to its good reputation. It is interesting to note that the results of their iterative calculation process often coincides with our globally optimal solutions. Where this has not been the case, i.e. synthetic and breast cancer data sets, we cannot uniquely decide which results are better and which worse.

4.2.7 Conclusion

We developed a mathematically motivated second order criterion for the feature subset selection task, which we solved to global optimality by semidefinite convexification and a branch-and-cut procedure. Our approach could achieve the highest classification accuracy in 18 of

[9] `http://www.nvidia.com/object/cuda_home.html`
[10] `http://archive.ics.uci.edu/ml/`

21 test scenarios, which could not be accomplished by other tested methods, rendering our method an alternative to well established filter methods. The additional computational expenditure of our approach has been compensated by a highly parallel implementation, which uses multi-core CPUs as well as GPU computing to achieve acceptable run times (maximum a few seconds for one fixed subset size). However, the major drawback of our approach are the quadratic storage needs for the corresponding matrices. Nevertheless, our good results show that using more memory might be more worth in comparison to the enormous computational costs of wrapper approaches. Future investigation, also on other data sets, will show whether an extended criterion, e.g. reduction of redundancy by a weighted criterion, can further improve our results.

Moreover, we solved optimally the maximal relevance minimal redundancy criterion of Peng et al. [140]. Certainly, their criterion is powerful, in particular when considering that we have the guarantee of optimality. However, the results of their incremental approach are very close to the optimal solutions. On average, our computationally more expensive globally optimal solutions seem not to be worth being preferred to the original incremental approach of Peng et al. [140]. Both methods, however, could not achieve the highest classification accuracy in such a substantial number of cases as our mathematically motivated second order method in our validation runs.

4.3 Determining Deregulated Subgraphs in Regulatory Networks

In the last decade, microarray-based gene expression profiles have become a central data resource to study deregulated molecular processes caused by diseases. Initially, microarray studies focused on single differentially expressed genes. Later, Gene Set Analysis (GSA) and related approaches were taking into account that genes do not act individually but in a coordinated fashion [121, 44, 4]. The disadvantage of this type of methods is that they can only reveal the enrichment of genes in predefined gene sets, e.g., canonical biological pathways. In recent years, the research focus has shifted toward analysis methods that integrate topological data mirroring the biological dependencies and interactions between the involved genes or proteins. In general, these graph-based approaches use scoring functions that assign scores or weights to the nodes or/and edges and make strong efforts to identify high-scoring pathways or subgraphs. A seminal work in this area is the paper of Ideker et al. who proposed a method for the detection of active subgraphs by devising a scoring function and a heuristic approach for detecting these subgraphs [83]. Other groups reported similar methods, which are all based on scoring networks given experimental data [144, 25, 125]. In 2008, Ulitsky and coworkers presented an algorithm for detecting disease-specific deregulated pathways by using clinical expression profiles [174]. However, the abovementioned approaches focused on protein-protein interaction (PPI) networks (undirected graphs) and used heuristics to find the subgraphs. Dittrich et al. devised the first approach to solve the maximal-scoring subgraph problem optimally by Integer Linear Programming (ILP) in the context of undirected PPI networks [45]. Considering regulatory networks, Keller et al. [96] recently proposed a dynamic programming algorithm to identify deregulated paths of a certain length relying on standard Gene Set Enrichment Analysis (GSEA) [121, 107, 97].

Here, we do not consider single deregulated paths, but subnetworks or subgraphs and present a novel branch-and-cut based approach for the determination of deregulated sub-

graphs in regulatory networks. Given a regulatory network and node scores indicating the deregulation of the corresponding genes, our approach identifies the heaviest connected subnetwork of size k, i.e. the most deregulated subnetwork with the highest sum of node scores. Here, we define a subgraph G as connected if it contains at least one root node v_r from which all other nodes in G are reachable, i.e., for each node v in G, a path from v_r to v consisting only of nodes in G exists. The vision implicated by the proposed connectivity model is to identify – besides the most deregulated components – the root node that may represent a key player in the pathogenic process. This key player may be responsible for the observed differences between the investigated conditions and may serve as a potential target for therapy purposes.

We applied our algorithm to gene expression profiles of nonmalignant mammary epithelial cells from BRCA1 mutation carriers and non BRCA1 mutation carriers [24] to explore the effect of the mutations on the regulatory processes and to gain new insights how these mutations may contribute to the development of breast cancer.

4.3.1 Linear 0-1-Program

The problem of finding a connected subgraph of size k that maximizes the sum of the scores is formulated as an Integer Linear Program (ILP) and then solved by a branch-and-cut approach. Here, we define a subgraph G as connected if it contains at least one root node v_r from which all other nodes in G are reachable, i.e., for each node v in G, a path from v_r to v consisting only of nodes in G exists. We assign a score (absolute value of the corresponding real data if available) to every node in the network. Since not all nodes or gene identifier of the network are also available on the microarray chip, we cannot assign a calculated score to every node of the regulatory network. Missing scores are set to zero.

Our ILP formulation uses two kinds of variables for each node i: x_i and y_i. The variables $x_i \in \{0, 1\}$ determine whether their corresponding nodes are contained in the subgraph ($x_i = 1$) or not ($x_i = 0$). Each variable $y_i \in \{0, 1\}$ indicates that its corresponding node i is the root node ($y_i = 1$) or not ($y_i = 0$). Let s_i be the score of node i then the optimization problem can be formulated as

$$\max_{\mathbf{x}} \sum_i s_i x_i.$$

The constraint that the subgraph has a predefined size of k nodes, is given by

$$\sum_i x_i = k.$$

We ensure that we obtain one root node by

$$\sum_i y_i = 1.$$

The inequalities

$$y_i \leq x_i \quad \text{for all } i$$

guarantee that a designated root node is also chosen. All remaining constraints concern the connectivity of the desired subgraph. Let $\mathrm{In}(i)$ be the set of indices of the predecessors of

node i, i.e. there exists an in-edge into node i, then we ensure that a chosen node has either a predecessor or it is the designated root node by

$$x_i - y_i - \sum_{j \in \mathrm{In}(i)} x_j \leq 0 \quad \text{for all } i.$$

Unfortunately, this kind of constraints is also fulfilled in cycles since every node in a cycle has a predecessor. Hence, a subgraph generated by the above constraints alone may have disconnected cycles. Let $\mathcal{C}$ be the indices of a cycle and analogously $\mathrm{In}(\mathcal{C})$ the set of indices of nodes which share an in-edge into this cycle, then the extension of the above constraint to the cycle $\mathcal{C}$ is given by

$$x_i - \sum_{j \in \mathcal{C}} y_j - \sum_{j \in \mathrm{In}(\mathcal{C})} x_j \leq 0 \quad \text{for all } i \in \mathcal{C}. \tag{4.10}$$

In theory, the complete description of our optimization problem as given above requires a constraint for every cycle, resulting in millions of inequalities of type (4.10) for the considered problem instances. In practice, branch-and-cut algorithms start with a basic set of constraints, solve the relaxed underlying LP problem, and check if the result violates constraints. If so, the violated constraints are added and the solver is restarted. As our set of basic cycle constraints, we only consider cycles with two or three nodes. In order to identify violated constraints, we implemented an efficient algorithm that searches in given LP solutions for cycles that do not satisfy inequalities of type (4.10). These inequalities will be added to the constraint set. This procedure is iterated until either we obtain an optimal subgraph, i.e an integer solution without violated constraints, or we have a non-integral solution, but we cannot identify further violated constraints. In the latter case, we perform a branching step. In this study, we used the branch-and-cut framework of CPLEX[11], version 11.110, with the "traditional mixed integer search method". This commercial library provides the possibility to branch using automatically detected favorable strategies. We used CPLEX's default settings. For a detailed survey of branch-and-cut algorithms, the interested reader is referred to Nemhauser [128] and Schrijver [160].

Our reference implementation is a single thread application, i.e. we could further speed up the solution process by parallelization techniques. However, all calculations finished within a few minutes on an Intel Xeon CPU, 2.5GHz. Thus, we did not incorporate advanced programming methods.

4.3.2 Microarray Data and Scored List Generation

To evaluate our method, we generated scored gene lists from different microarray expression profiles. For each of these data sets, we performed a normalization of the arrays if necessary. If log transformed data was given, we computed for each transcript the fold difference of the mean of the sample data versus the mean of the reference data. Since we can only map NCBI Gene IDs to the network nodes, the microarray transcript IDs are converted to this identifier type. The resulting list contains for each gene on the microarray a score that mirrors the deregulation of the gene under the considered conditions.

[11] `http://www-01.ibm.com/software/integration/optimization/cplex/`

4.3.3 Statistical Methods and Gene Sets for Validation

For testing the significance of a computed subgraph of size k and root node v_r, we carried out 1000 permutation tests where we permuted the scores of the network nodes and computed the best subgraph of size k with root v_r. The p-value was calculated as the number of permutations reaching an equal or better score than our original subgraph rooted in v_r divided by the total number of permutations.

To compare our method to the results of standard gene set analysis methods, we analyzed the input lists sorted by their score with standard unweighted gene set enrichment analysis (GSEA) using GeneTrail [11, 97]. Amongst other functional categories already provided by GeneTrail, we also analyzed the curated gene set "c2.all.v2.5.symbols.gmt" from the Molecular Signatures Database (MSigDB) [167], which contains additional gene sets from online pathway databases, publications in PubMed, and knowledge of domain experts. Furthermore, we performed an over-representation analysis (ORA) with GeneTrail of the nodes/genes of the deregulated subgraph as test set and the genes of the regulatory graph as reference set.

4.3.4 Results

The input of our algorithm consists of a regulatory network and a list of genes that are scored according to their deregulation. In this work, the underlying regulatory network was taken from the KEGG database [93]. Since KEGG pathways also contain nodes for protein families, we transformed the original KEGG pathways by splitting the nodes of protein families into their components.

The second necessary input for our algorithm is a list of scored genes. These scores can be derived, e.g., from expression experiments. In brief, if we want to compare the differences in expression of two conditions, we compute for each transcript on the microarray a score that mirrors the difference between the considered states. In general, we can use any measure that is also applied for finding differentially expressed genes as, e.g., the fold change. In an additional step, the transcript IDs are converted to gene identifiers. The resulting list contains for each gene on the microarray a score that mirrors the deregulation of the gene under the considered conditions, i.e., the higher the expression difference between the two considered states, the larger the score of a gene.

Before the computation, the genes of the list have to be mapped to the network nodes. Since not all nodes or gene identifier of the network are also available on the microarray, we cannot assign a calculated score to every node of the regulatory network. Missing scores are assumed to be zero. In our tests, about an eighth of all nodes had a zero score.

Given this input, our ILP-based algorithm computes the most deregulated connected subnetwork of size k, i.e. the subgraph with the highest sum of node scores. It is rooted in a special node v_r from which all other nodes in the computed subnetwork are reachable. The results can be visualized in the Biological Network Analyzer (BiNA) [105], which is a Java application suited for the visualization of metabolic and regulatory networks.

We downloaded and analyzed the GSE13671 data set from GEO. The GSE13671 set contains expression data from nonmalignant primary mammary epithelial cells with and without BRCA1 mutations and was published in a study of Burga et al. [24]. We computed the fold difference for the mean of the BRCA1 mutation carriers against the mean of non mutation carriers given the normalized and log transformed expression values. The Affymetrix chip IDs were mapped to NCBI Gene IDs and the resulting list containing genes and correspond-

ing expression values served as input for our algorithm. To explore the stability of the core components in this case, we computed the most deregulated subgraphs for different subgraph sizes ranging from 10 to 25 nodes as in the first example.

Figure 4.4 shows the best subgraph for 25 nodes and, additionally, the remaining nodes of the union graph as isolated vertices. Figure 4.4 also reveals that the complete union graph is very compact (only 34 vertices for the most deregulated subgraphs consisting of 10-25 nodes), which means that the most deregulated part of the network seems to be stable. The core components occurring in all of these subgraphs are the path EGLN3 (PHD3) $\rightarrow$ EPAS1 (HIF-2α) $\rightarrow$ VEGF $\rightarrow$ KDR (VEGFR2) with the designated root node EGLN3 and, more downstream located, the subgraph rooted in MAPK13 consisting of the genes TP53, DDIT3, RRM2, and GADD45B. It is interesting to note that the root node is very stable, i.e. independent of the size of the subgraph, EGLN3 is always the designated root node.

When performing an ORA for the genes of the subgraph of size 25 as test set and the genes of the regulatory network as reference set, we find many pathways significantly enriched that are associated with cancer, e.g., the KEGG pathways: "VEGF signaling pathway", "MAPK signaling pathway", "Focal adhesion", "ErbB signaling pathway", and the "p53 signaling pathway". These pathways have in common that they influence crucial cell processes as proliferation, differentiation, cell motility, and survival. Furthermore, we can confirm the results of Burga et al. [24], since the genes of the detected subgraph are also enriched in the EGF pathway (MSigDB), as well as in the GO terms cell cycle and cell cycle arrest. Interestingly, we also find pathways or categories significantly enriched that are associated with hypoxia and oxidative stress, as e.g. "Hypoxia review", "Hypoxia normal up", and "Oxstress breastca up" from MSigDB.

To compare the results of our algorithm to a standard gene set enrichment analysis, we subjected the input list containing the genes sorted by the absolute values of their fold differences to the GSEA variant implemented in GeneTrail. The analysis revealed many significantly deregulated pathways (p-value < 0.05, FDR adjusted), amongst others the KEGG pathways "cell cycle", "DNA replication", and "missmatch repair". When regarding the MSigDB gene sets, we find the breast cancer related categories "BRCA ER neg", "BRCA ER pos", "Breast cancer estrogen signaling", and "Breast ductal carcinoma genes", as well as the hypoxia related category "Hypoxia reg up" significantly deregulated. Interestingly, in this analysis neither the p53 signaling pathway nor the EGF signaling pathway was significantly deregulated.

Since a complete discussion about our results from a biological point of view is beyond the scope of this work, the interested reader may be referred to Backes et al. [12].

4.3.5 Conclusion

Our approach is able to identify deregulated connected subgraphs in a directed regulatory network. The optimization approach can be combined with every node-based scoring function that is suitable to measure the deregulation of the corresponding genes or proteins. In this study, we used the regulatory network from KEGG. However, we can apply the method to any type of biological network, with slight modifications even to protein-protein interaction networks or a combination of regulatory and protein-protein interaction networks.

The identification of patterns of pathway deregulation is a crucial task in differential network analysis. Moreover, the determination of the initiators of the observed differences between the investigated conditions is a major challenge. With our connectivity model we

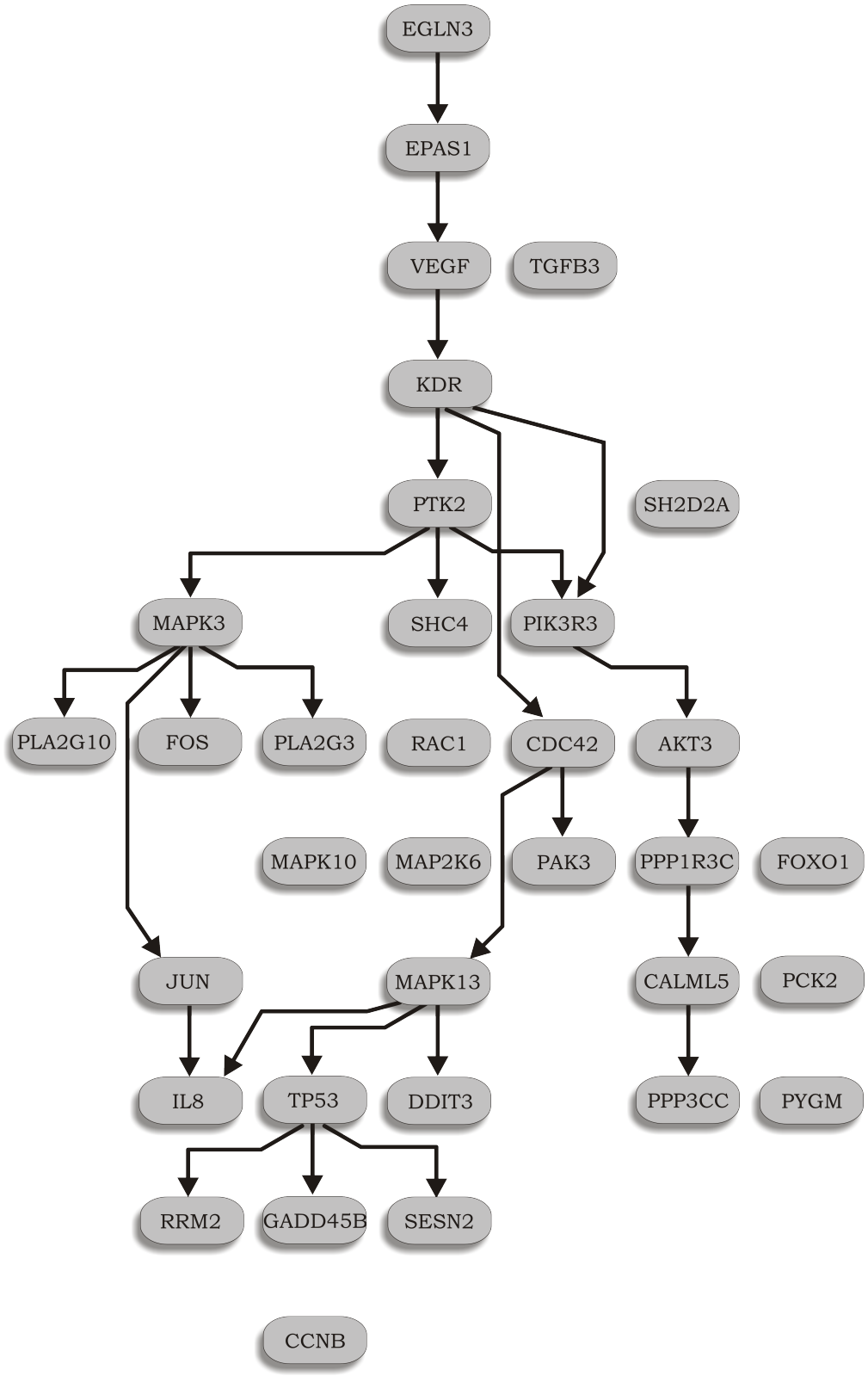

Figure 4.4: The most deregulated subgraph for BRCA1 mutation carriers against non-mutation carriers for a network size of 25. The unconnected nodes are part of the consensus network of the deregulated subgraphs of size 10–25.

do not only identify the most deregulated subgraph, but also a root node, which may be one of the key players for the deregulation. We applied our method to expression profiles of nonmalignant primary mammary epithelial cells (PMECs) isolated from BRCA1 mutation carriers and women without BRCA1 mutations. BRCA1 germline mutations are associated with a predisposition for developing breast cancer.

In [12], we show that the nonmalignant mammary epithelial cells with BRCA1 mutations exhibit many properties that are known from breast cancer. Our study indicates that the cells are in a stressful state potentially originated from the processes involved in the regulation of long-term oxidative stress. Moreover, it seems that it is a very thin line between a cancerous outcome and non-cancerous phenotype for BRCA1 mutated mammary epithelial cells considering the accumulated deregulation affecting multiple signaling pathways visible in our computed subgraphs. Although a GSEA also reported hypoxia as significant finding, this category was one of some hundred significant categories. Thus, we had missed this information most probably, while our approach led definitely to this result.

With our approach the most deregulated part of a network can be visualized and experts can directly grasp the processes involved in the deregulation. Although the interpretation is not always straightforward, our approach is at least a very powerful complement to the standard gene set and single gene analysis methods for microarray data. Furthermore, we showed that the application of our algorithm to already published data can yield new insights. As expression data and network data are still growing, methods as our ILP-based algorithm will be valuable to detect deregulated subgraphs in different conditions and help contribute to a better understanding of diseases.

Chapter 5

Summary and Outlook

Many computational tasks can be formulated as optimization problems. Optimization is also a core task in bioinformatics. In this work, we addressed optimization problems that are central in bioinformatics.

We improved current computational approaches for local optimization in the context of molecular potential energy functions. We presented a novel and efficient algorithm to find the next local optimum into a given direction on a molecular potential energy function. This task arises as a subproblem of many local optimization methods. We presented a new consensus like approach that can be easily adapted to the local behavior of objective functions, while it does not require additional function evaluations, imposing only negligible computational overhead. Replacing a standard line search method with the new algorithm reduced the number of function/gradient evaluations in our test runs down to 47.7% (down to 85% on average).

Furthermore, we dealt with the parametrization problem in the context of molecular representations. Our approach to parameterize a ligand uses a compact representation to reduce the number of variables. However, due to the well-known singularity problem of orientational parameters, the optimization process in ligand-receptor docking may get stuck at non-optimum positions. We showed that our method avoids this problem by computationally efficient reparametrization and enables gradient-based optimization of molecular complexes using the compact molecular representation. We gave details on the incorporation of our orientational parametrization into the local optimization procedure and showed, especially, that our line search method is well suited for this task. Our approach was clearly superior to the stochastic method of Solis and Wets in our test runs. This fact became even more substantial when we considered ligands of increasing complexity.

Then, we extended this local optimization procedure to a ligand-receptor docking approach. We presented a so-called Lamarckian genetic algorithm, a heuristic for providing the local optimization procedure with different start positions. We showed that this combined approach is clearly superior to other approaches employing a stochastic local optimization method. The new algorithm features shorter run times and gives substantially better results, especially with increasing complexity of the ligands. In our validation runs, we gained an up to tenfold speedup in comparison to other tested methods. Thus, it may be used to dock ligands with many rotatable bonds with high efficiency.

While the ligand was fully flexible[1] in the aforementioned docking method, the receptor

[1] In terms of the compact representation.

was kept rigid. We then further incorporated side chain flexibility of the receptor with the same parametrization as we introduced for the ligand. Additionally, we achieved limited backbone movement by interpolating between two extremal conformations using spherical linear extrapolation and gave details on how the resulting gradient can be calculated. This approach enables application of the whole range of gradient-based optimization methods to flexible ligand-receptor docking when two or more extremal backbone conformations are known. In our study, we chose human serum albumin (HSA), which is the most abundant transport protein of the human blood plasma. It is known for its promiscuity to bind different ligand species, and it is one of the most extensively studied proteins. The fact that HSA undergoes tremendous backbone movements upon binding to fatty acids, facilitating enormous topological and structural changes over the whole protein, rendered this molecule to an ideal candidate for our study. According to our results, this approach is very promising for flexible ligand-receptor docking.

However, the aforementioned approach needs known extremal backbone conformations for the interpolation. In general, we cannot assume that all necessary conformations can be obtained by interpolation between already known structures. Hence, we extended our method and allowed a loop region to be fully flexible. We dealt with the modification of backbone torsion angles for flexible docking using global optimization techniques. The special challenge was to calculate "real" conformations, i.e. the obtained loops must start and end at given positions. In 1970, Gō and Scheraga published equations whose solutions represent the loop closure in polypeptides. They addressed the problem of solving these equations with Newton's method. Since this is only a local search method, they may have sampled the search space by providing different equally distributed start positions. Thus, there is no guarantee to find all solutions and, most probably, many function and derivative evaluations had to be performed. We presented a new method to find all possible conformations using interval arithmetic. In our test runs, all results were obtained nearly instantly. For this study, we chose the human 17β-hydroxysteroid dehydrogenase type 1 (17β-HSD 1). In preliminary studies, different approaches to perform flexible docking with 17β-HSD 1 failed because of the high flexibility of the loop region. This algorithm reliably found alternative conformations and was able to identify promising loop/ligand complexes of the studied example in our test runs.

The last part of this work dealt with global optimization. We described the bond order assignment problem for molecular structures. Bond order information can often not be directly inferred from the available experimental data. Even important molecular databases, like the Cambridge Structural Database [6] and the Protein Data Bank (PDB) [18, 17], are known to contain erroneous data for connectivity and bond order information [106] or to omit them entirely. For nucleic acids and proteins bond orders can easily be obtained due to their building block nature, but this does not hold for other kinds of molecules like ligands. Furthermore, it is not practicable to assign bond orders manually for, e.g., virtual screening purposes, where thousands of molecules are to be considered. Hence, automated bond order assingment is often a fundamental task for the work with molecules. In the past, very different strategies were applied to derive bond order information, most of them relying on the correctness of the atom coordinates. We extended an ansatz proposed by Wang et al. that assigns heuristic molecular penalty scores solely based on connectivity information and tries to heuristically approximate its optimum. This procedure has two drawbacks: the scores of the resulting assignments are not guaranteed to be optimal and the algorithm provides only one solution while there can be more than one assignment with optimal score. In this work, we presented our novel linear 0-1-programming formulation for the very efficient computation

of all optimal and suboptimal bond order assignments. Our approach did not only outperform the original heuristic approach of Wang et al., but also commonly used software for determining bond orders on our test set considering all optimal results. This set consisted of 761 thoroughly prepared drug like molecules that were originally used for the validation of the Merck Molecular Force Field (MMFF94).

Furthermore, we addressed the task of selecting a subset of a given set of features based on mutual information. In machine learning, the problem of supervised classification is to induce a model that classifies objects into a finite set of known classes using labeled examples. Copious classification tasks occur in bioinformatics, such as distinguishing cancer tissues from normal tissues [7] or one cancer subtype vs another [5], predicting protein fold or superfamily from its sequence [85, 43], etc.. Avoiding irrelevant or redundant features is important because they may have a negative effect on the accuracy of the classifier. Instead of using all available features, only a subset is employed for classification tasks mainly with the following aims: (1) reduction of overfitting of the used learning methods and, hence, improvement of the classification accuracy, (2) the obtained features are more interpretable that can help identifying and monitoring the target diseases or function types, and, finally, (3) dimension reduction decreases the computational costs for the classification algorithms. The prevalent methods are filter approaches and wrapper type methods. The last-named methods are computationally very expensive in comparison to filters. In this work, we presented our filter method that uses second order information while other methods strongly rely only on first order information. Additionally, our criterion is mathematically well motivated and, in contrast to other methods, exactly solved by quadratic 0-1-programming. In the validation runs, our method achieved in 18 out of 21 test scenarios the best classification accuracies.

Finally, we presented our novel branch-and-cut approach for the determination of deregulated subgraphs in regulatory networks using expression profiles. Our method assesses the subnetworks by the sum of their participating vertex scores. Each score indicates the deregulation of the corresponding gene. The vision implicated by the proposed connectivity model is to identify – besides the most deregulated components – the root node that may represent a key player in the pathogenic process. This key player may be responsible for the observed differences between the investigated conditions and may serve as a potential target for therapy purposes. To demonstrate the capabilities of our algorithm, we analyzed expression profiles from nonmalignant primary mammary epithelial cells derived from BRCA1 mutation carriers and epithelial cells without BRCA1 mutation. Our results suggest that oxidative stress plays an important role in epithelial cells with BRCA1 mutations that may contribute to the later development of breast cancer. It is important to note that the application of our algorithm to already published data could yield new insights. As expression data and network data are still growing, methods as our algorithm will be valuable to detect deregulated subgraphs in different conditions and help contribute to a better understanding of diseases.

We do not claim that the methods and approaches presented in this work render the addressed problems negligible. We intend to further improve these techniques. Here, we give a short outlook for future plans:

- Our line search approach strongly depends on suitable interpolants reflecting the local behavior of the objective function. Future investigations will show whether other interpolation schemes are even better suited for molecular potential energy functions.

- Our general docking approach may be improved by using other, maybe better suited heuristics to provide our local optimization techniques with starting positions.

- Our approach for loop modeling is a monte-carlo like procedure, i.e. the loop is (randomly) modeled and kept rigid when the ligand is docked. Future versions may incorporate directly the loop parameters into the docking algorithm without an outer monte carlo loop.
- Our bond order assignment approach can be improved by defining other scoring functions better suited for special classes of molecules.
- Our mathematically motivated criterion for the selection of a subset of features may be changed to, e.g., incorporate also the reduction of redundancy.
- The determination of deregulated subgraphs in regulatory networks strongly depends on good cuts (inequalities) that can be identified during the optimization process. Investigations will show whether we can identify other classes of inequalities helpful to solve the problems more efficiently.

There is plenty of room for improvements but the presented techniques are a good basis for further development. Our results show that we could achieve substantial improvements on many bioinformatical tasks.

Appendix A

Feature Subset Selection Validation Runs

Here, we present the results of our feature subset selection validation runs, see Section 4.2, where we use the abbreviations: SOC = our second order condition, see Section 4.2.1, glob. opt. MRMR = our globally optimal solution of the criterion in Peng et al. [140], see Section 4.2.3, MRMR = approach of Peng et al., see Section 4.2.2, CMIM (ori.) = method proposed by Fleuret in [51], and CMIM (err.) = method of Fleuret in his erratum (as used in his implementation).

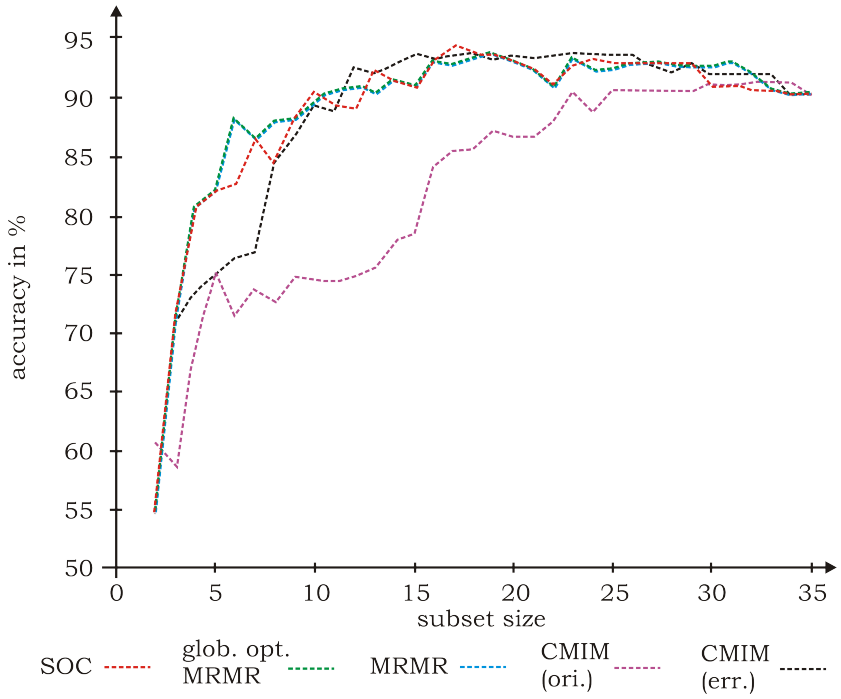

Figure A.1: Classification results of Soybean (large) data set. Classifier: SVM (linear kernel).

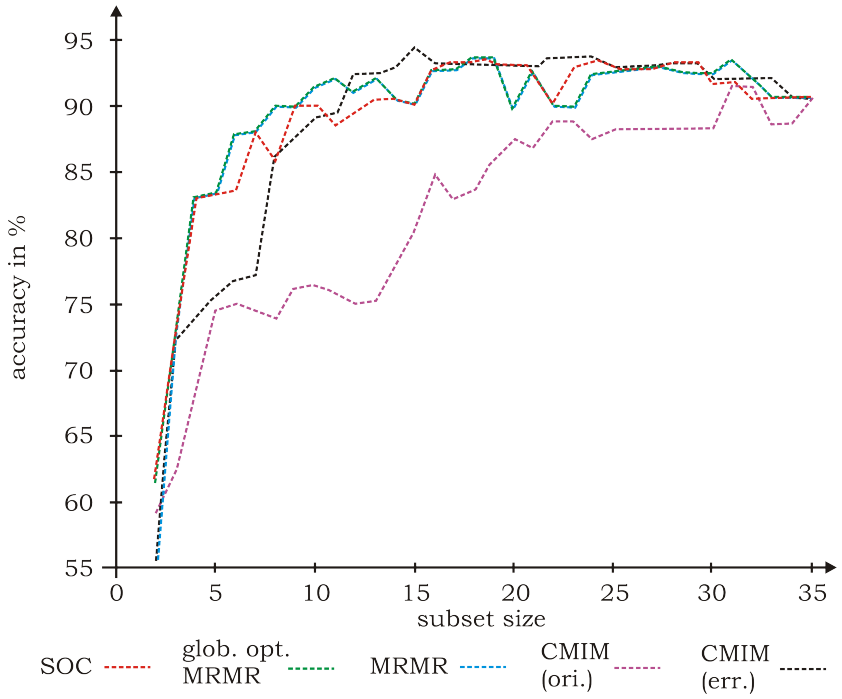

Figure A.2: Classification results of Soybean (large) data set. Classifier: SVM (RBF kernel).

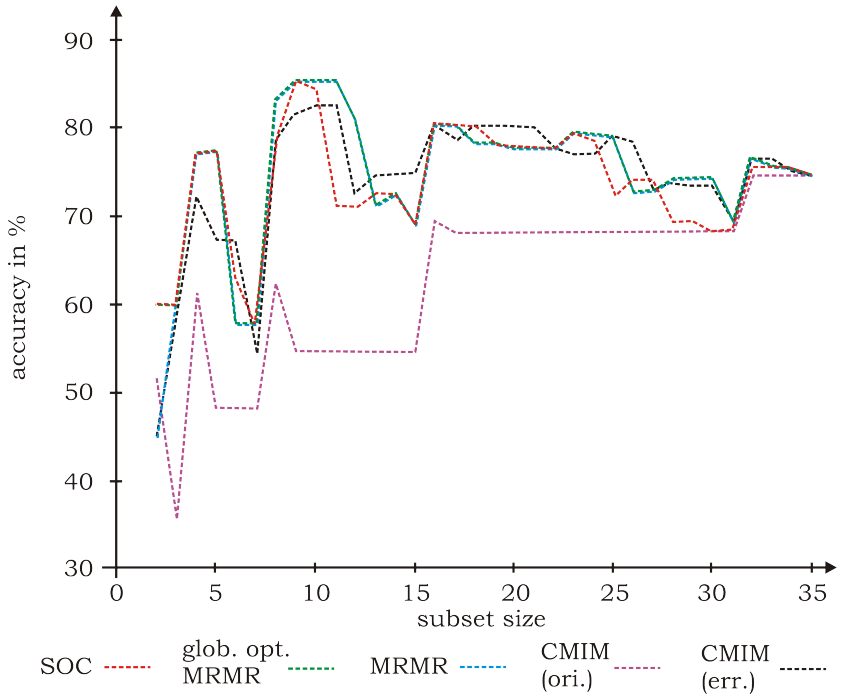

Figure A.3: Classification results of Soybean (large) data set. Classifier: NB.

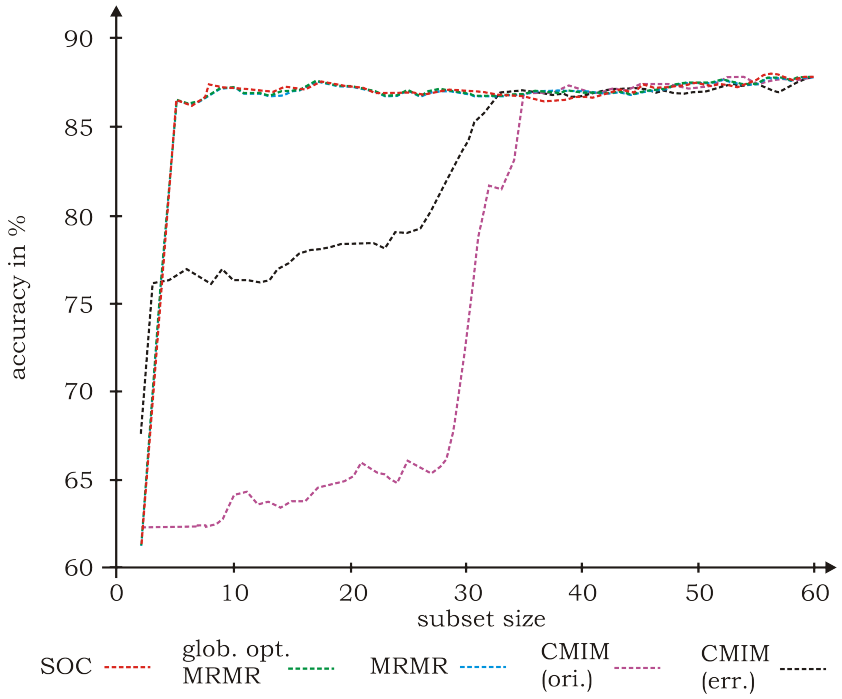

Figure A.4: Classification results of Splice-junction Gene Sequences data set. Classifier: SVM (linear kernel).

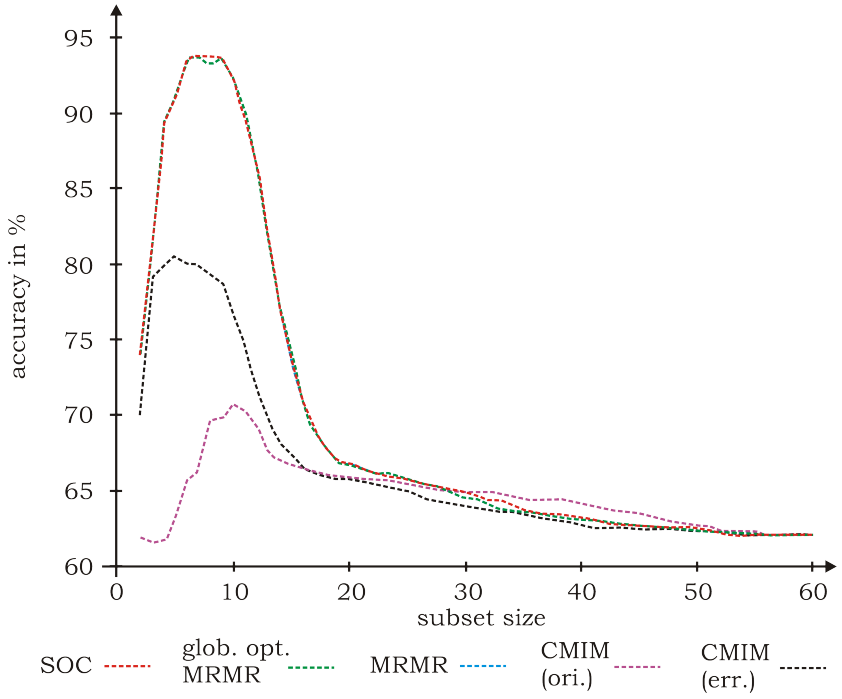

Figure A.5: Classification results of Splice-junction Gene Sequences data set. Classifier: SVM (RBF kernel).

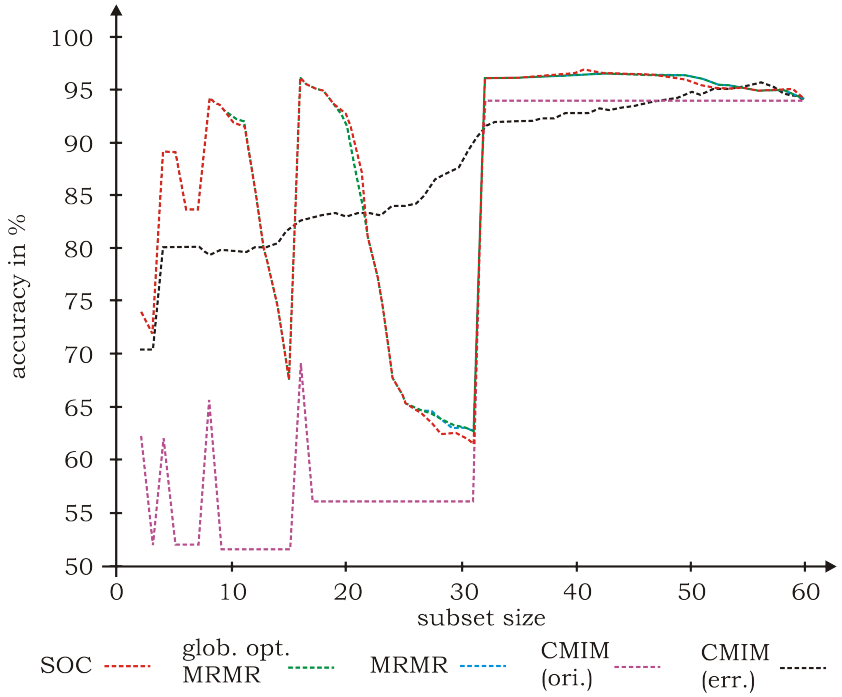

Figure A.6: Classification results of Splice-junction Gene Sequences data set. Classifier: NB.

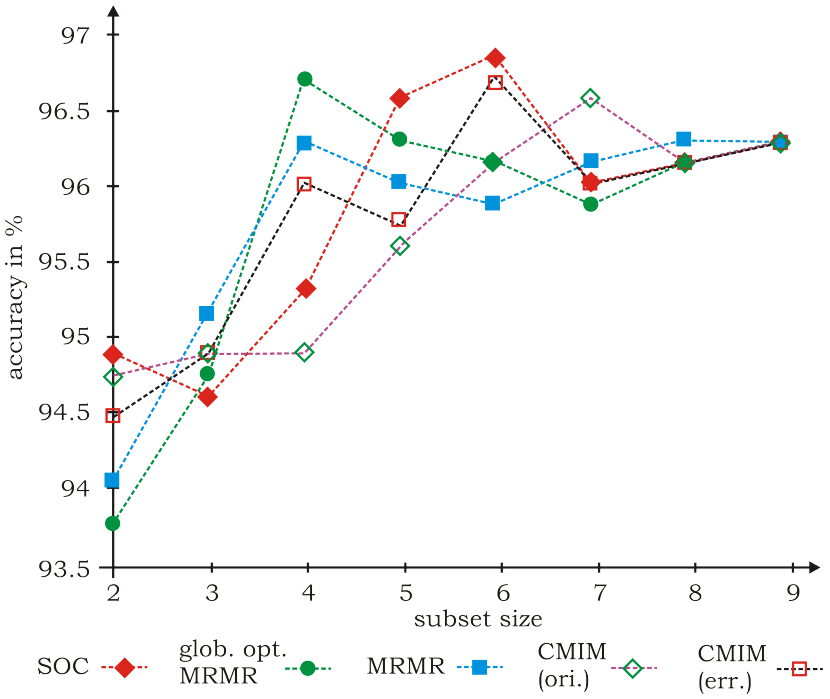

Figure A.7: Classification results of breast cancer data set. Classifier: SVM (linear kernel).

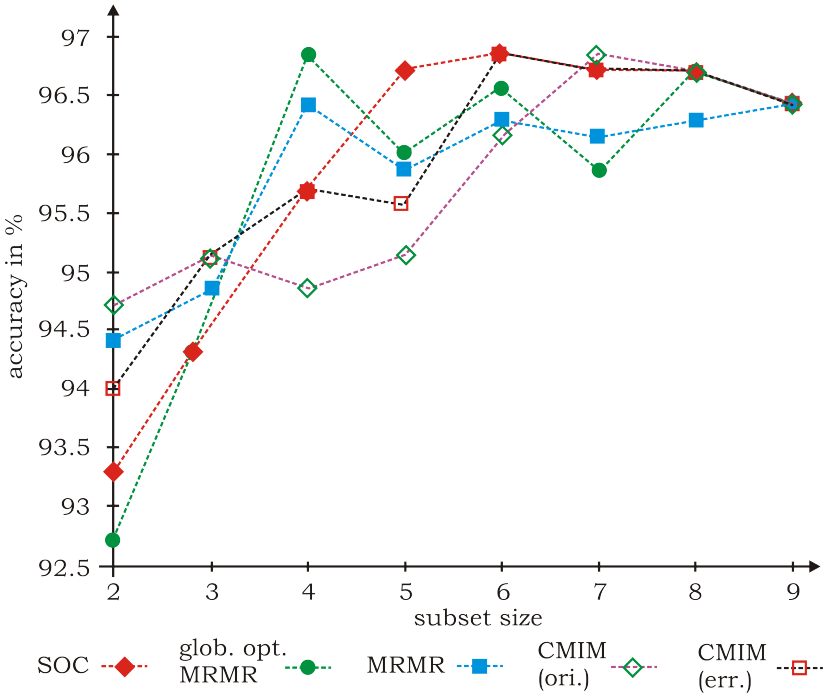

Figure A.8: Classification results of breast cancer data set. Classifier: SVM (RBF kernel).

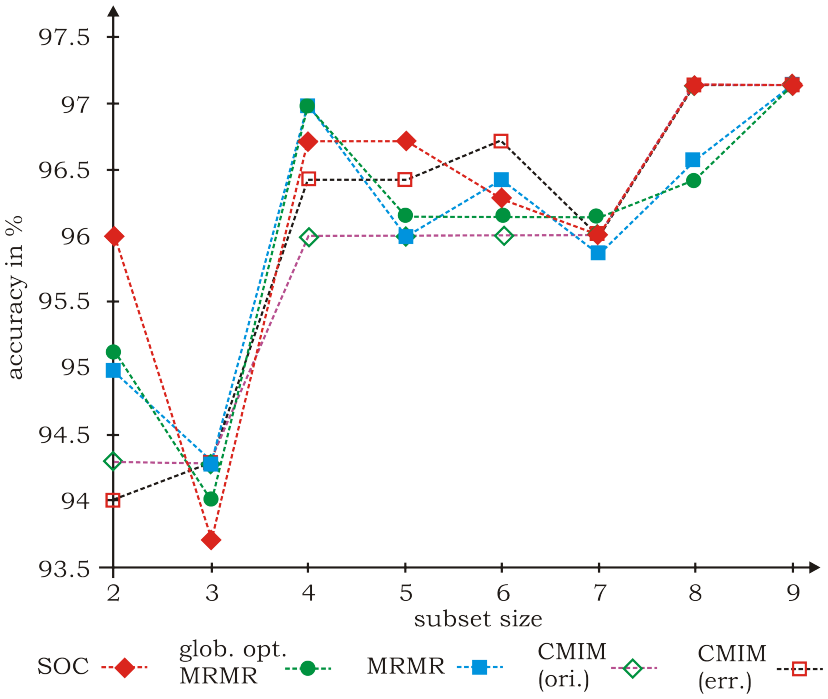

Figure A.9: Classification results of breast cancer data set. Classifier: NB.

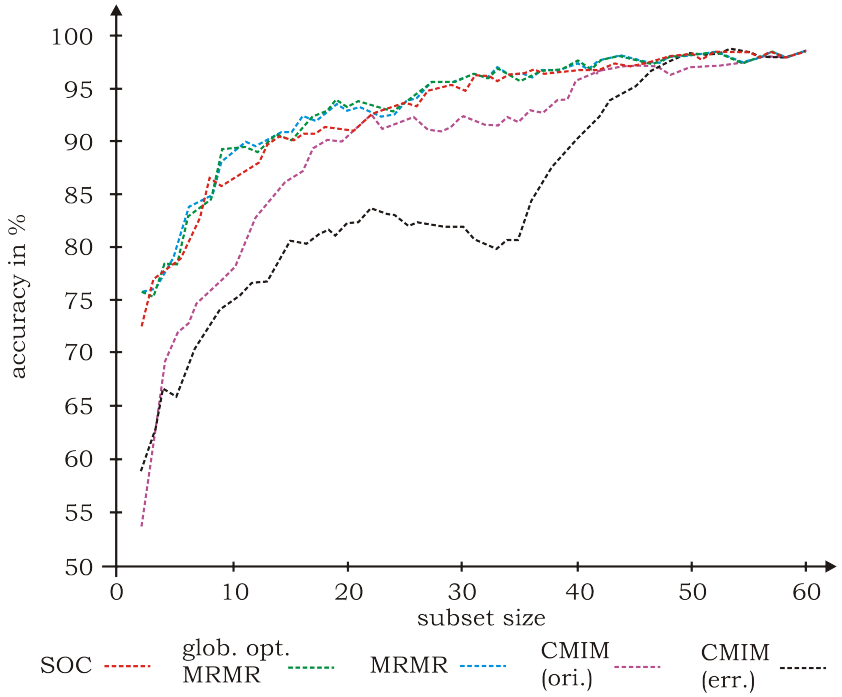

Figure A.10: Classification results of synthetic data set. Classifier: SVM (linear kernel).

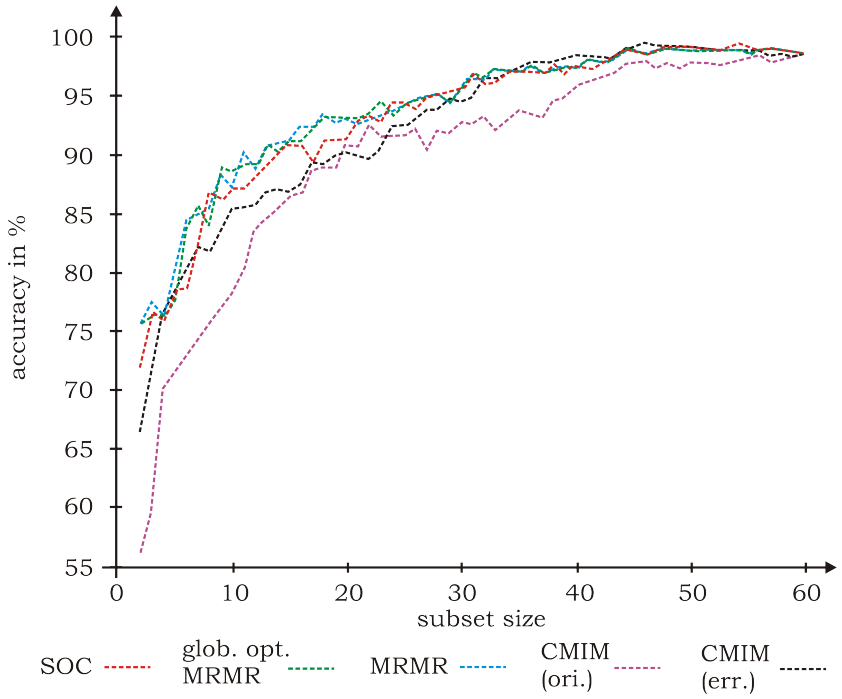

Figure A.11: Classification results of synthetic data set. Classifier: SVM (RBF kernel).

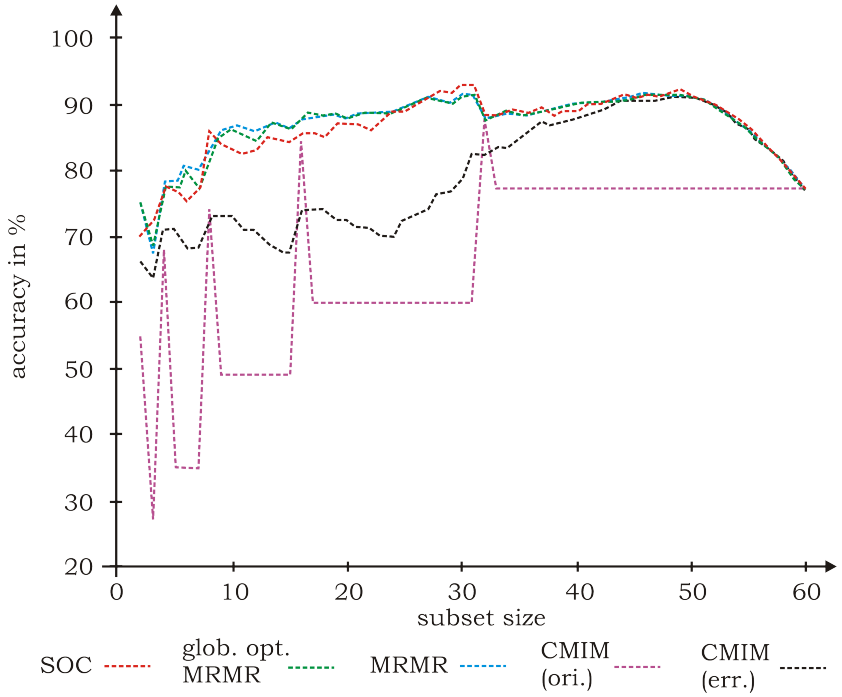

Figure A.12: Classification results of synthetic data set. Classifier: NB.

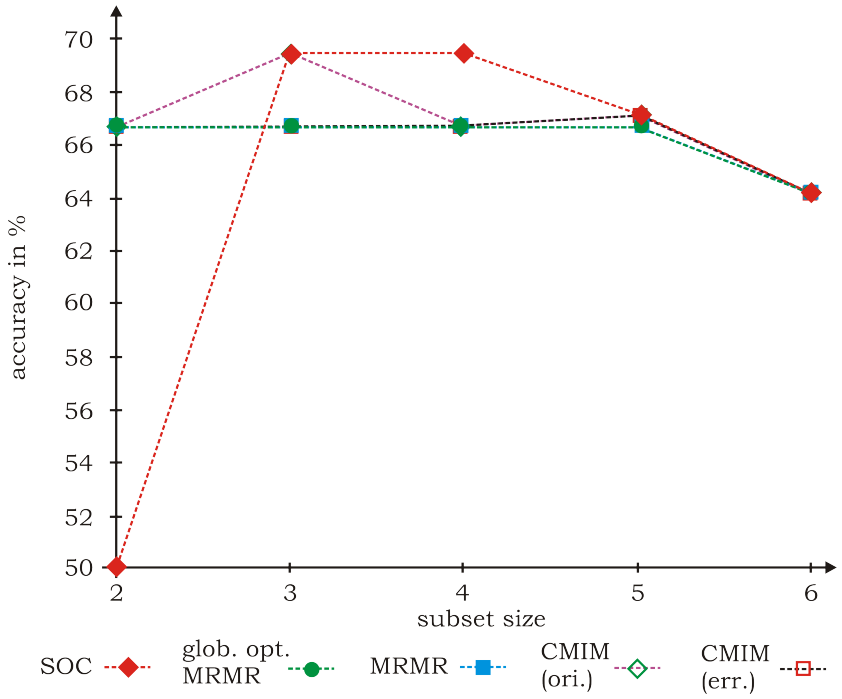

Figure A.13: Classification results of MONKS 1 data set. Classifier: SVM (linear kernel).

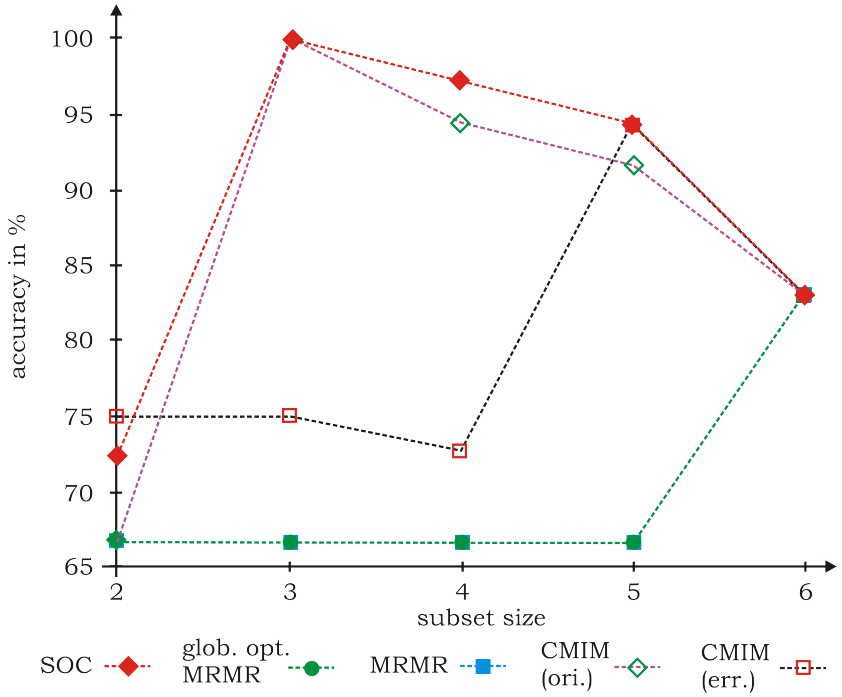

Figure A.14: Classification results of MONKS 1 data set. Classifier: SVM (RBF kernel).

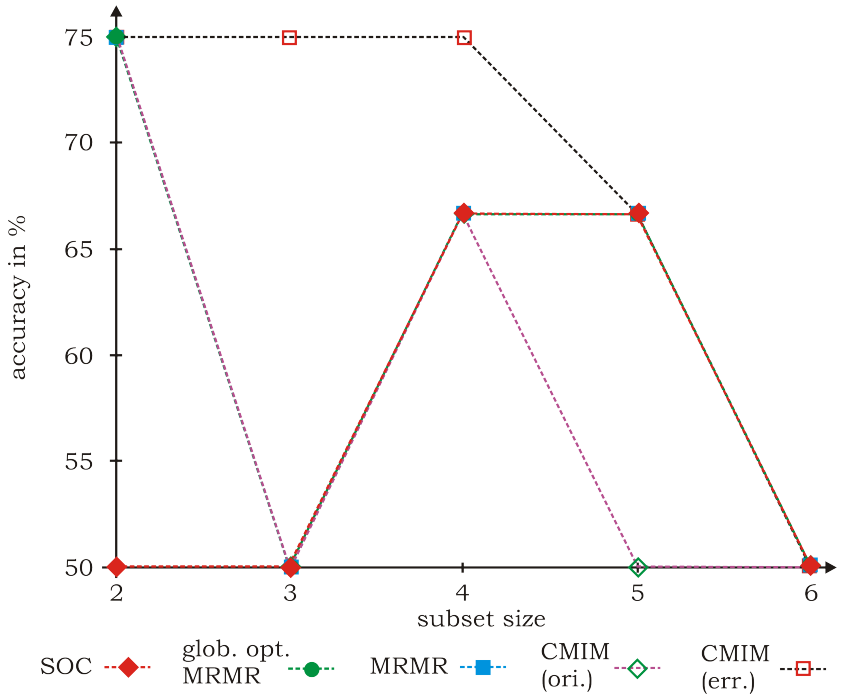

Figure A.15: Classification results of MONKS 1 data set. Classifier: NB.

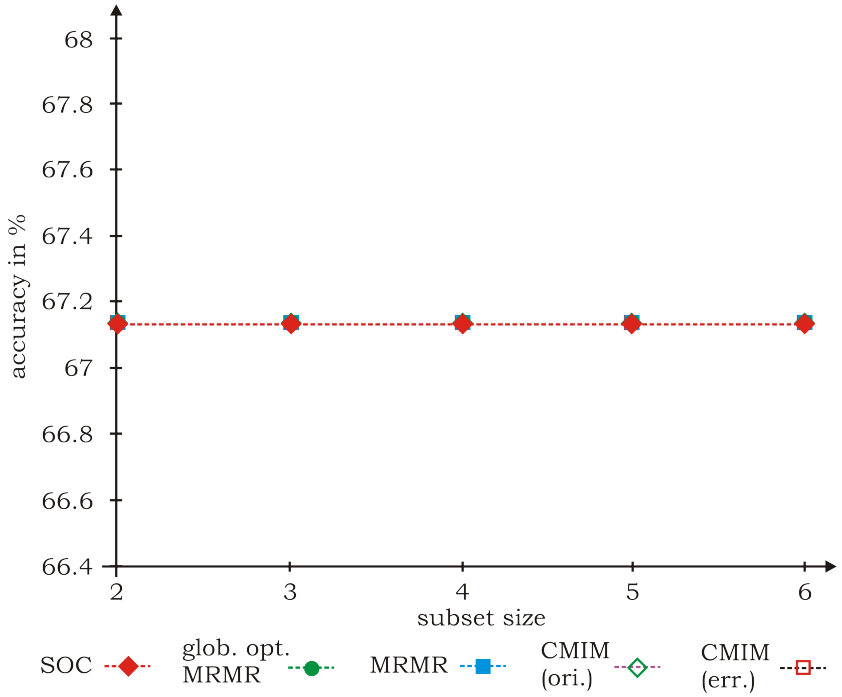

Figure A.16: Classification results of MONKS 2 data set. Classifier: SVM (linear kernel).

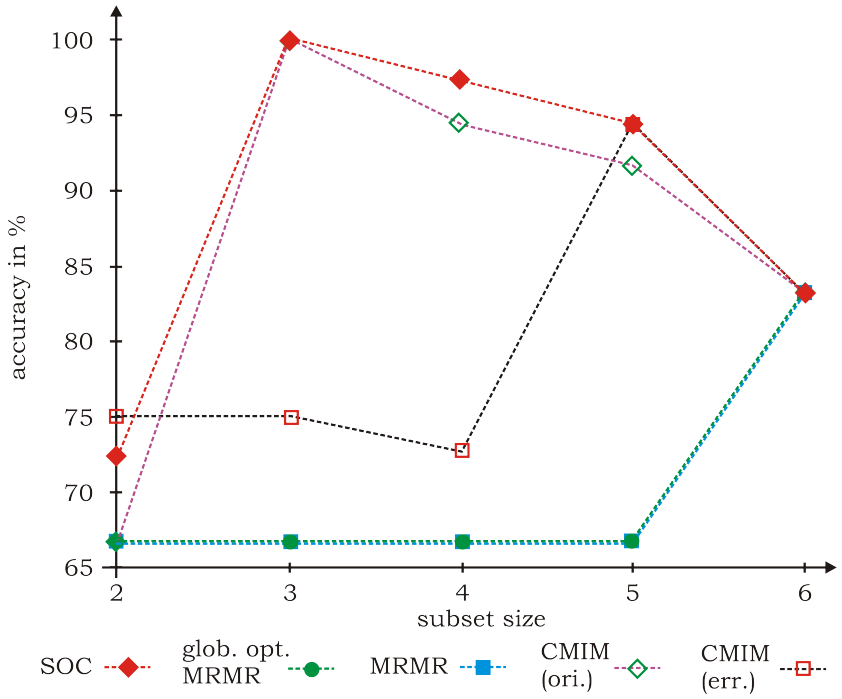

Figure A.17: Classification results of MONKS 2 data set. Classifier: SVM (RBF kernel).

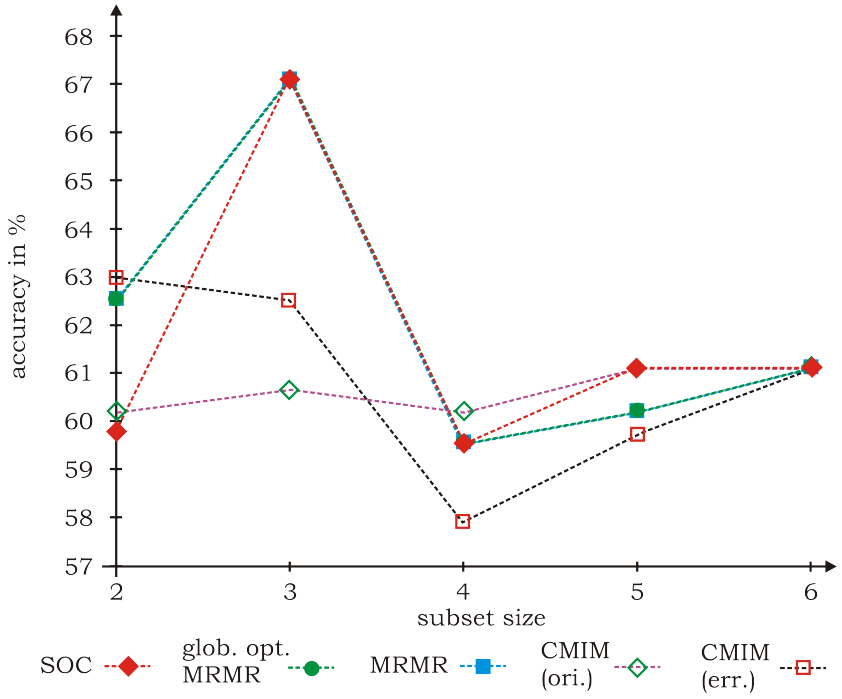

Figure A.18: Classification results of MONKS 2 data set. Classifier: NB.

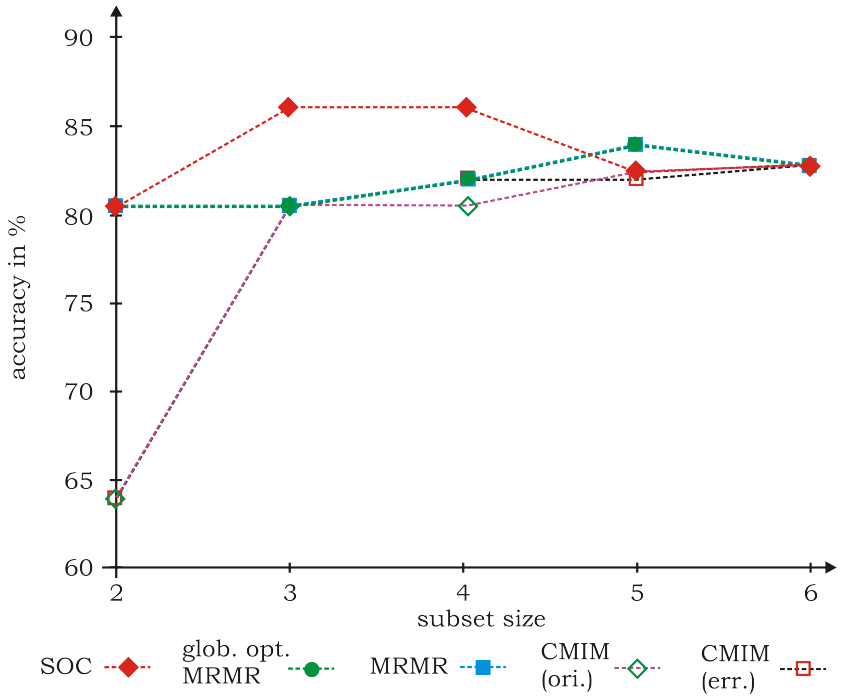

Figure A.19: Classification results of MONKS 3 data set. Classifier: SVM (linear kernel).

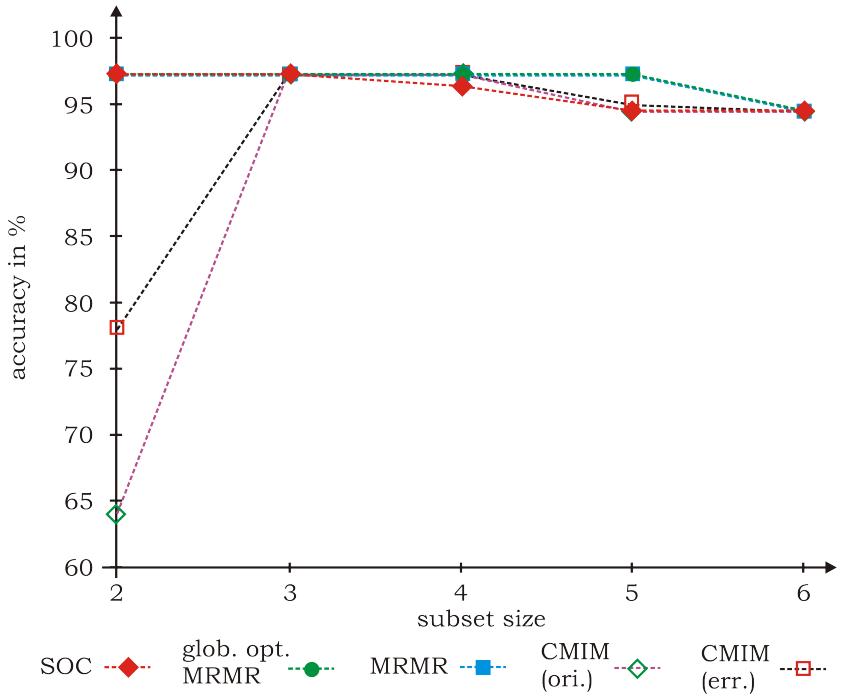

Figure A.20: Classification results of MONKS 3 data set. Classifier: SVM (RBF kernel).

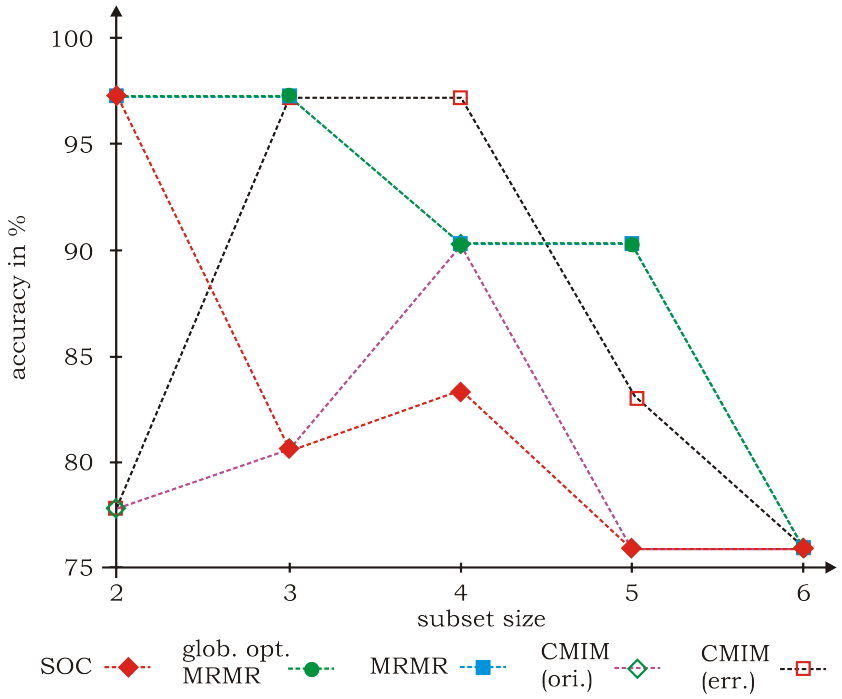

Figure A.21: Classification results of MONKS 3 data set. Classifier: NB.

Bibliography

[1] H. Abe, W. Braun, T. Noguti, and N. Go. Rapid calculation of first and second derivatives of conformational energy with respect to dihedral angles for proteins. General recurrent equations. *Computers and Chemistry*, 8(4):239–247, 1984.

[2] M. Al-Baali. Improved hessian approximations for the limited memory BFGS method. *Numerical Algorithms*, 22(1):99–112, October 1999.

[3] M. Al-Baali and R. Fletcher. An efficient line search for nonlinear least squares. *Journal of Optimization Theory and Applications*, 48(3):359–377, March 1986.

[4] F. Al-Shahrour, L. Arbiza, H. Dopazo, J. Huerta-Cepas, P. Mínguez, D. Montaner, and J. Dopazo. From genes to functional classes in the study of biological systems. *BMC Bioinformatics*, 8:114, 2007.

[5] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M. Staudt. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

[6] F. H. Allen. The cambridge structural database: a quarter of a million crystal structures and rising. *Acta Crystallographica B*, 58:380–388, 2002.

[7] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences, USA*, 96(12):6745–6750, June 1999.

[8] N. H. Andersen, K. A. Olsen, R. M. Fesinmeyer, Xu Tan, F. M. Hudson, L. A. Eidenschink, and S. R. Farazi. Minimization and optimization of designed β-hairpin folds. *Journal of the American Chemical Society*, 128(18):6101–6110, April 2006.

[9] B. M. Averick, R. G. Carter, and J. J. Moré. The minpack-2 test problem collection. Technical Report ANL/MCS-P153-0692, Argonne National Laboratory, Argonne, USA, 1992.

[10] J. C. Baber and E. E. Hodgkin. Automated assignment of chemical connectivity to organic molecules in the cambridge structural database. *Journal of Chemical Information and Computer Science*, 32:401–406, 1992.

[11] C. Backes, A. Keller, J. Kuentzer, B. Kneissl, N. Comtesse, Y. A. Elnakady, R. Mller, E. Meese, and H.-P. Lenhof. GeneTrail–advanced gene set enrichment analysis. *Nucleic Acids Res.*, 35:W186–192, July 2007.

[12] C. Backes, A. Rurainski, A. Gerasch, G. Klau, J. Küntzer, D. Eggle, M. Hein, A. Keller, H. Burtscher, M. Kaufmann, E. Meese, and H.-P. Lenhof. An integer linear programming approach for finding deregulated subgraphs in regulatory networks using expression profiles. Manuscript in preparation.

[13] I. Badger, I. Minor, M. A. Oliveira, T. I. Smith, and M. G. Rossmann. Structural analysis of antiviral agents that interact with the capsid of human rhinoviruses. *Proteins*, 6:1–19, 1989.

[14] D. W. Banner and P. Hadvary. Crystallographic analysis at 3.0-A resolution of the binding to human thrombin of four active site-directed inhibitors. *Journal of Biological Chemistry*, 266:20085–20093, 1991.

[15] C. A. Baxter, C. W. Murray, D. E. Clark, D. R. Westhead, and M. D. Eldridge. Flexible docking using tabu search and an empirical estimate of binding affinity. *Proteins*, 33:367–382, 1998.

[16] C. Baysal, H. Meirovitch, and M. Navon. Performance of efficient minimization algorithms as applied to models of peptides and proteins. *Journal of Computational Chemistry*, 20(3):354–364, February 1999.

[17] H. M. Berman, K. Henrick, and H. Nakamura. Announcing the worldwide Protein Data Bank. *Nature Structural Biology*, 10:980, 2003.

[18] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[19] A. Billionnet and S. Elloumi. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. *Mathematical Programming A*, 109:55–68, 2007.

[20] P. Bjørstad and J. Nocedal. Analysis of a new algorithm for one-dimensional minimization. *Computing*, 22(1):93–100, March 1979.

[21] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11(1):613–623, 1999.

[22] B. E. Boser, I. M. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, 1992.

[23] R. Breton, D. Housset, C. Mazza, and J. C. Fontecilla-Camps. The structure of a complex of human 17beta-hydroxysteroid dehydrogenase with estradiol and NADP+ identifies two principal targets for the design of inhibitors. *Structure*, 4:905–915, 1996.

[24] L. N. Burga, N. M. Tung, S. L. Troyan, M. Bostina, P. A. Konstantinopoulos, H. Fountzilas, D. Spentzos, A. Miron, Y. A. Yassin, B. T. Lee, and G. M. Wulf. Altered proliferation and differentiation properties of primary mammary epithelial cells from BRCA1 mutation carriers. *Cancer Research*, 69(4):1273–1278, February 2009.

[25] L. Cabusora, E. Sutton, A. Fulmer, and C. V. Forst. Differential network expression during drug and stress response. *Bioinformatics*, 21(12):2898–2905, June 2005.

[26] E. Cantú-Paz. *Efficient and accurate parallel genetic algorithms*. Kluwer Academic Publishers: Boston, MA, 2001.

[27] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[28] P. S. Charifson, J. J. Corkery, M. A. Murcko, and W. P. Walters. Consensus scoring: A method for obtaining improved hit rates from docking databases of three-dimensional structures into proteins. *Journal of Medicinal Chemistry*, 42(25):5100–5109, 1999.

[29] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. MPS-SIAM Series on Optimization 1. SIAM, 2000.

[30] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley and Sons, New York, 1997.

[31] C. R. Corbeil, P. Englebienne, and N. Moitessier. Docking ligands into flexible and solvated macromolecules. 1. development and validation of FITTED 1.0. *Journal of Chemical Information and Modeling*, 47(2):435–449, 2007.

[32] C. Cortes and V. Vapnik. Support-vector network. *Machine Learning*, 20:1–25, 1995.

[33] E. A. Coutsias, C. Seok, and K. A. Dill. Using quaternions to calculate rmsd. *Journal of Computational Chemistry*, 25:1849–1857, 2004.

[34] G. B. Dantzig. *Lineare Programmierung und Erweiterungen*. Springer, Berlin, 1966.

[35] S. Das. Filters, wrappers and a boosting-based hybrid for feature selection. *Proceedings of the International Conference on Machine Learning 2001*, pages 74–81, 2001.

[36] W. C. Davidon. Conic approximations and collinear scalings for optimizers. *SIAM Journal on Numerical Analysis*, 17(2):268–281, 1980.

[37] L. H. de Figueiredo and J. Stolfi. Affine arithmetic: concepts and applications. *Numerical Algorithms*, 37:147–158, 2004.

[38] S. Degroeve, B. De Baets, Y. Van de Peer, and P. Rouzé. Feature subset selection for splice site prediction. *Bioinformatics*, 18:S75–S83, 2002.

[39] A. K. Dehof. *Dissertation*. PhD thesis, Saarland University, Center for Bioinformatics.

[40] A. K. Dehof, A. Rurainski, Q. B. A. Bui, S. Böcker, H.-P. Lenhof, and A. Hildebrandt. Automated bond order assignment as an optimization problem. Manuscript in preparation.

[41] A. K. Dehof, A. Rurainski, H.-P. Lenhof, and A. Hildebrandt. Automated bond order assignment as an optimization problem. *German Conference on Bioinformatics*, 2009.

[42] M. Deza and M. Laurent. *Cut Polyhedra and Metrics*. Springer, Berlin, 1997.

[43] C. Ding and I. Dubchak. Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, 17:349–358, 2001.

[44] I. Dinu, J. D. Potter, T. Mueller, Q. Liu, A. J. Adewale, G. S. Jhangri, G. Einecke, K. S. Famulski, P. Halloran, and Y. Yasui. Improving gene set analysis of microarray data by SAM-GS. *BMC Bioinformatics*, 8:242, 2007.

[45] M. T. Dittrich, G. W. Klau, A. Rosenwald, T. Dandekar, and T. Müller. Identifying functional modules in protein-protein interaction networks: an integrated exact approach. *Bioinformatics*, 24(13):i223–231, July 2008.

[46] R. L. Dunbrack. Rotamer libraries in the 21st century. *Current Opinion in Structural Biology*, 12:431–440, 2002.

[47] T. J. A. Ewing and I. D. Kuntz. Critical evaluation of search algorithms for automated molecular docking and database screening. *Journal of Computational Chemistry*, 18:1175–1189, 1997.

[48] S. I. Feldman, D. M. Gay, M. W. Maimone, and N. L. Schryer. A Fortran-to-C converter. Technical Report 149, AT&T Bell Laboratories, Murray Hill, NJ, 1990.

[49] R. Fletcher. *Practical methods of optimization; (2nd ed.)*. Wiley-Interscience, New York, NY, USA, 1987.

[50] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *Computer Journal*, 7(2):149–154, 1964.

[51] F. Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

[52] M. Froeyen and P. Herdewijn. Correct bond order assignment in a molecular framework using integer linear programming with application to molecules where only non-hydrogen atom coordinates are available. *Journal of Chemical Information and Modeling*, 45(5):1267–1274, 2005.

[53] J. Fuhrmann. *Gradient Based Optimization in Ligand-Receptor Docking*. PhD thesis, Saarland University, Center for Bioinformatics, 2009.

[54] J. Fuhrmann, A. Rurainski, H.-P. Lenhof, and D. Neumann. A new method for the gradient-based optimization of molecular complexes. *Journal of Computational Chemistry*, 30(9):1371–1378, July 2009.

[55] J. Fuhrmann, A. Rurainski, H.-P. Lenhof, and D. Neumann. A new Lamarckian genetic algorithm for flexible ligand-receptor docking. *Journal of Computational Chemistry*, 2010.

[56] D. K. Gehlhaar, G. M. Verkhivker, P. A. Rejto, C. J. Sherman, D. R. Fogel, L. J. Fogel, and S. T. Freer. Molecular recognition of the inhibitor AG-1343 by HIV-1 protease: conformationally flexible docking by evolutionary programming. *Chemistry and Biology*, 2(5):317–324, May 1995.

[57] J. C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal on Optimization*, 2(1):21–42, 1992.

[58] N. Gō and H. A. Scheraga. Ring closure and local conformational deformations of chain molecules. *Macromolecules*, 3(2):178–187, 1970.

[59] N. Gould and Ph. L. Toint. Preprocessing for quadratic programming. *Mathematical Programming B*, 100(1):95–132, 2004.

[60] N. I. M. Gould and Ph. L. Toint. An iterative working-set method for large-scale non-convex quadratic programming. Technical Report RAL-TR-2001-026, Computational Science and Engeneering Department, Rutherford Appleton Laboratory, Atlas Centre, Rutherford Appleton Laboratory, Oxfordshire OX 11 0QX, July 2001.

[61] N. I. M. Gould and Ph. L. Toint. Numerical methods for large scale non-convex quadratic programming. In A. H. Siddiqi and M. Kocvara, editors, *Trends in Industrial and Applied Mathematics*, pages 149–179. Kluwer, Dordrecht (NL), 2002.

[62] F. S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

[63] T. A. Halgren. Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94. *Journal of Computational Chemistry*, 17(5):490–519, April 1996.

[64] T. A. Halgren. Merck molecular force field. II. MMFF94 van der waals and electrostatic parameters for intermolecular interactions. *Journal of Computational Chemistry*, 17(5):520–552, April 1996.

[65] T. A. Halgren. Merck molecular force field. III. Molecular geometries and vibrational frequencies for MMFF94. *Journal of Computational Chemistry*, 17(5):553–586, April 1996.

[66] T. A. Halgren. Merck molecular force field. IV. Conformational energies and geometries for MMFF94. *Journal of Computational Chemistry*, 17(5):587–615, April 1996.

[67] T. A. Halgren. Merck molecular force field. V. Extension of MMFF94 using experimental data, additional computational data, and empirical rules. *Journal of Computational Chemistry*, 17(5):616–641, April 1996.

[68] E. R. Hansen. A globally convergent interval method for computing and bounding real roots. *BIT Numerical Mathematics*, 18(4):415–424, December 1978.

[69] E. R. Hansen. Interval forms of newton methods. *Computing*, 20:153–163, 1978.

[70] E. R. Hansen. Global optimization using interval analysis: The one-dimensional case. *Journal of Optimization Theory and Applications*, 29(3):331–344, November 1979.

[71] P. Hansen, B. Jaumard, M. Ruiz, and J. Xiong. Global minimization of indefinite quadratic functions subject to box constraints. *Nav. Res. Logist.*, 40:373–392, 1993.

[72] W. E. Hart. *Adaptive global optimization with local search.* PhD thesis, Computer Science and Engeneering Department, University of California, San Diego, 1994.

[73] M. J. Hartshorn, M. L. Verdonk, G. Chessari, S. C. Brewerton, W. T. M. Mooij, P. N. Mortenson, and C. W. Murray. Diverse, high-quality test set for the validation of protein-ligand docking performance. *Journal of Medicinal Chemistry*, 50:726–741, 2007.

[74] B. E. Henderson, R. Ross, and L. Bernstein. Estrogens as a cause of human cancer: the richard and hinda rosenthal foundation award lecture. *Cancer Research*, 48:246–253, January 1988.

[75] M. Hendlich, F. Rippmann, and G. Barnickel. BALI: Automatic assignment of bond and atom types for protein ligands in the Brookhaven Protein Databank. *Journal of Chemical Information and Computer Sciences*, 37:774–778, 1997.

[76] A. Hildebrandt, A. K. Dehof, A. Rurainski, A. Bertsch, M. Schumann, N. C. Toussaint, A. Moll, D. Stöckel, S. Nickels, H.-P. Lenhof, and O. Kohlbacher. BALL – Biochemical Algorithms Library 1.3. Manuscript in preparation.

[77] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms.* Springer, Berlin, New-York, 1993.

[78] M. R. Hoare. Structure and dynamics of simple microclusters. *Advances in Chemical Physics*, 40:49–135, 1979.

[79] J. H. Holland. *Adaption in natural and artificial systems.* MIT Press, Cambridge, MA, 1975.

[80] S. Honda, K. Yamasaki, Y. Sawada, and H. Morii. 10 residue folded peptide designed by segment statistics. *Structure*, 12(8):1507–1518, August 2004.

[81] T. J. Hou, J. M. Wang, and X. J. Xu. A comparison of three heuristic algorithms for molecular docking. *Chinese Chemical Letters*, 10:615–618, 1999.

[82] R. Huey, G. M. Morris, A. J. Olson, and D. S. Goodsell. A semiempirical free energy force field with charge-based desolvation. *Journal of Computational Chemistry*, 28(6):1145–1152, 2007.

[83] T. Ideker, O. Ozier, B. Schwikowski, and A. F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 18 Suppl 1:S233–240, 2002.

[84] Jr. J. E. Dennis and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations (Classics in Applied Mathematics, 16).* Society for Industrial & Applied Mathematics (SIAM), 1996.

[85] T. Jaakkola, M. Diekhans, and D. Haussler. Using the fisher kernel method to detect remote protein homologies. *ISMB'99*, pages 149–158.

[86] F. Jarre and J. Stoer. *Optimierung*. Springer-Verlag Berlin Heidelberg New York, 2004.

[87] H. Jhoti, O. M. Singh, M. P. Weir, R. Cooke, P. Murray-Rust, and A. Wonacott. X-ray crystallographic studies of a series of penicillin-derived asymmetric inhibitors of HIV-1 protease. *Biochemistry*, 33:8417–8427, 1994.

[88] G. Jones, P. Willett, R. C. Glen, and A. R. Leach. Development and validation of a genetic algorithm for flexible docking. *Journal of Molecular Biology*, 267:727–748, 1997.

[89] G. J. Jones, P. Willett, and R. C. Glen. A genetic algorithm for flexible molecular overlay and pharmacophore elucidation. *Journal of Computer-Aided Molecular Design*, 9:532–549, 1995.

[90] R. S. Judson, E. P. Jaeger, and A. M. Treasurywala. A genetic algorithm based method for docking flexible molecules. *Journal of Molecular Structure: THEOCHEM*, 308:191–206, 1994.

[91] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica*, A32:922–923, 1976.

[92] W. Kabsch. A discussion of the solutio for the best rotation to relate two sets of vectors. *Acta Crystallographica*, A34:827–828, 1978.

[93] M. Kanehisa and F. Goto. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28:27–30, 2000.

[94] M. Kanehisa, F. Goto, M. Hattori, K. F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa. From genomics to chemical genomics: new developments in kegg. *Nucleic Acids Research*, 34:D354–357, 2006.

[95] C. F. F. Karney. Quaternions in molecular modeling. *Journal of Molecular Graphics and Modelling*, 25(5):595–604, 2007.

[96] A. Keller, C. Backes, A. Gerasch, M. Kaufmann, O. Kohlbacher, E. Meese, and H.-P. Lenhof. A novel algorithm for detecting differentially regulated paths based on gene set enrichment analysis. *Bioinformatics*, page btp510, August 2009.

[97] A. Keller, C. Backes, and H.-P. Lenhof. Computation of significance scores of unweighted gene set enrichment analyses. *BMC Bioinformatics*, 8(290), 2007.

[98] L. G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[99] E. E. Kim, C. T. Baker, M. D. Dwver, M. A. Murcko, B. G. Rao, R. D.Tung, and M. A. Navia. Crystal structure of HIV-1 protease in complex with Vx-478, a potent and orally bioavailable inhibitor of the enzyme. *Journal of the American Chemical Society*, 117:1181–1182, 1995.

[100] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, pages 273–324, 1997.

[101] O. Kohlbacher and H.-P. Lenhof. Rapid software prototyping in computational molecular biology. In *Proceedings of the German Conference on Bioinformatics (GCB'99)*, 1999.

[102] O. Kohlbacher and H.-P. Lenhof. BALL - rapid software prototyping in computational molecular biology. *Bioinformatics*, 16:815–824, 2000.

[103] M. Kontoyianni, L. M. McClellan, and G. S. Sokol. Evaluation of Docking Performance: Comparative Data on Docking Algorithms. *Journal of Medicinal Chemistry*, 47(3):558–565, 2003.

[104] B. Korte and J. Vygen. *Combinatorial Optimization – Theory and Algorithms*. Springer-Verlag, Berlin, Heidelberg, New York, 2008.

[105] J. Kuentzer, T. Blum, A. Gerasch, C. Backes, A. Hildebrandt, M. Kaufmann, O. Kohlbacher, and H.-P. Lenhof. BN++ - a biological information system. *Journal of Integrative Bioinformatics*, 3(2), 2006.

[106] P. Labute. On the perception of molecules from 3D atomic coordinates. *Journal of Chemical Information and Modeling*, 45:215–221, 2005.

[107] J. Lamb, S. Ramaswamy, H. L. Ford, B. Contreras, R. V. Martinez, F. S. Kittrell, C. A. Zahnow, N. Patterson, T. R. Golub, and M. E. Ewen. A mechanism of cyclin d1 action encoded in the patterns of gene expression in human cancer. *Cell*, 114(3):323–334, August 2003.

[108] L. J. Langer, J. A. Alexander, and L. L. Engel. Human placental estradiol-17 beta dehydrogenase. II. kinetics and substrate specificities. *Journal of Biological Chemistry*, 234:2609–2614, October 1959.

[109] L. J. Langer and L. L. Engel. Human placental estradiol-17 beta dehydrogenase. I. concentration, characterization and assay. *Journal of Biological Chemistry*, 233:583–588, September 1958.

[110] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[111] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5):1–18, 1990.

[112] R. F. Marcia, J. C. Mitchell, and S. J. Wright. Global optimization in protein docking using clustering, underestimation and semidefinite programming. *Optimization Methods and Software*, 22:803–811, 2007.

[113] I. Marcotte, F. Separovic, M. Auger, and S. M. Gagné. A multidimensional ^{1}h nmr investigation of the conformation of methionine-enkephalin in fast-tumbling bicelles. *Biophysical Journal*, 86(3):1587–1600, March 2004.

[114] M. Cs. Markót, J. Fernández, L. G. Casado, and T. Csendes. New interval methods for constrained global optimization. *Mathematical Programming: Series A and B*, 106(2):287–318, 2006.

[115] E. Martz. Morpher. http://www.umass.edu/microbio/chime/morpher/morphmtd.htm, 1998–2002.

[116] H. Matthies and G. Strang. The solution of nonlinear finite element equations. *International Journal for Numerical Methods in Engineering*, 14:1613–1626, 1979.

[117] Z. Michalewicz and D. B. Fogel. *How to solve it: Modern Heuristics*. Springer Berlin, Heidelberg, New York, 2004.

[118] R. S. Michalski and R. L. Chilausky. Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *International Journal of Policy Analysis and Information Systems*, 4(2), 1980.

[119] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[120] R. E. Moore. *Interval Analysis*. Prentice-Hall, 1966.

[121] V. K. Mootha, C. M. Lindgren, K. F. Eriksson, A. Subramanian, S. Sihag, J. Lehar, P. Puigserver, E. Carlsson, M. Ridderstrale, E. Laurila, N. Houstis, M. J. Daly, N. Patterson, J. P. Mesirov, T. R. Golub, P. Tamayo, B. Spiegelman, E. S. Lander, J. N. Hirschhorn, D. Altshuler, and L. C. Groop. PGC-1alpha-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. *Nature Genetics*, 34:267–73, 2003.

[122] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. Algorithm 566: Fortran subroutines for testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, 7(1):136–140, March 1981.

[123] J. J. Moré and D. J. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software*, 20:286–307, 1994.

[124] G. M. Morris, D. S. Goodsell, R. S. Halliday, R. Huey, W. E. Hart, R. K. Belew, and A. J. Olson. Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry*, 19:1639–1662, 1998.

[125] S. Nacu, R. Critchley-Thorne, P. Lee, and S. Holmes. Gene expression network analysis and applications to immunology. *Bioinformatics*, 23(7):850–858, April 2007.

[126] N. Narayana, S. Cox, S. Shaltiel, S. S. Taylor, and N. Xuong. Crystal structure of a polyhistidine-tagged recombinant catalytic subunit of cAMP-dependent protein kinase complexed with the peptide inhibitor PKI(5-24) and adenosine. *Biochemistry*, 36:4438–4448, 1997.

[127] J. W. Neidigh, R. M. Fesinmeyer, and N. H. Andersen. Designing a 20-residue protein. *Nature Structural Biology*, 9:425–430, April 2002.

[128] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley and Sons, New York, 1988.

[129] A. Neumeier. *Interval methods for systems of equations*. Cambridge University Press, 1990.

[130] K. Nickel. *Topics in interval analysis*, chapter Triplex-Algol and its application. 1969.

[131] K. Nickel. On the newton method in interval analysis. Mathematics research center report, University of Wisconsin, December 1971.

[132] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782, 1980.

[133] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, 2006.

[134] M. O. Noordewier, G. G. Towell, and J. W. Shavlik. Training knowledge-based neural networks to recognize genes in dna sequences. *Advances in Neural Information Processing Systems*, 3:530, 1991.

[135] C. M. Oshiro, I. D. Kuntz, and J. S. Dixon. Flexible ligand docking using a genetic algorithm. *Journal of Computer-Aided Molecular Design*, 9(2):113–130, 1995.

[136] E. A. Padlan, G. H. Cohen, and D. R. Davies. On the specificity of antibody/antigen interactions: phosphocholine binding to McPC603 and the correlation of three-dimensional structure and sequence data. *Annales de l'Institut Pasteur. Immunologie*, 136C:271–276, 1985.

[137] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization – Algorithms and Complexity*. Dover Publications, 1998.

[138] P. M. Pardalos and S. A. Vavasis. Quadratic programming with one negative eigenvalue is np-hard. *Journal of Global Optimization*, 1:15–22, 1991.

[139] I. C. Paschalidis, Y. Shen, P. Vakili, and S. Vajda. SDU: A semi-definite programming-based underestimation method for stochastic global optimization in protein docking. *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 3675–3680, 2005.

[140] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[141] D. T. Pham and A. B. Chan. Control chart pattern recognition using a new type of self organizing neural network. *Proceedings of the Institution of Mechanical Engineers*, 212(1):115–127, 1998.

[142] A. Poljak, F. Rendl, and H. Wolkowicz. A recipe for semidefinite relaxation for (0,1)-quadratic programming. *Journal of Global Optimization*, 7:51–73, 1995.

[143] T. Puranen, M. Poutanen, D. Ghosh, P. Vihko, and R. Vihko. Characterization of stuctureal and functional properties of human 17 beta-hydroxysteroid dehydrogenase type 1 using recombinant enzymes and site-directed mutagenesis. *Molecular Endocrinology*, 11:77–86, January 1997.

[144] D. Rajagopalan and P. Agarwal. Inferring pathways from gene lists using a literature-derived network of biological relationships. *Bioinformatics*, 21(6):788–793, March 2005.

[145] M. Respondek, T. Madl, C. Göbl, R. Golser, and K. Zangger. Mapping the orientation of helices in micelle-bound peptides by paramagnetic relaxation waves. *Journal of the American Chemical Society*, 129(16):5228–5234, March 2007.

[146] D. D. Robinson, P. J. Winn, P. D. Lyne, and W. G. Richards. Self-organizing molecular field analysis: a tool for structure-activity studies. *Journal of Medicinal Chemistry*, 42:573–583, February 1999.

[147] R. T. Rockafellar. *Convex analysis*. Princeton University Press, Princeton, NJ, 1997.

[148] Stephan Roth. Modification of backbone torsion angles for flexible docking. Master's thesis, Saarland University, Center for Bioinformatics, 2009.

[149] A. Rurainski. Einführung in die lineare ganzzahlige Programmierung und Lösen mittels Branch-and-Cut-Verfahren. Published online, http://bioinf-www.bioinf.uni-sb.de/people/rurainki/ilp_bac.pdf, December 2004.

[150] A. Rurainski. Semidefinite Programmierung in der Bioinformatik. Diplomarbeit, Saarland University, Center for Bioinformatics, January 2004.

[151] A. Rurainski, J. Fuhrmann, S. Roth, D. Neumann, and H.-P. Lenhof. Protein-ligand docking with backbone flexibility. Manuscript in preparation.

[152] A. Rurainski, J. Fuhrmann, T. Schackmann, D. Neumann, and H.-P. Lenhof. Flexible protein-ligand docking with backbone movements. Manuscript in preparation.

[153] A. Rurainski, M. Hein, A. Keller, and H.-P. Lenhof. Optimal feature selection based on mutual information second order criterion. Manuscript in preparation.

[154] A. Rurainski, A. Hildebrandt, and H.-P. Lenhof. A consensus line search algorithm for molecular potential energy functions. *Journal of Computational Chemistry*, 30(9):1499–1509, July 2009.

[155] M. H. Rusin. A revised simplex method for quadratic programming. *SIAM Journal on Applied Mathematics*, 20(2):143–160, 1971.

[156] Thomas Schackmann. Incorporating backbone movements in flexible protein-ligand docking. Master's thesis, Saarland University, Center for Bioinformatics, 2009.

[157] T. Schlick and A. Fogelson. Tnpack– a truncated newton minimization package for large-scale problems: I. algorithm and usage. *ACM Transactions on Mathematical Software*, 18(1):56–70, March 1992.

[158] T. Schlick and A. Fogelson. Tnpack– a truncated newton minimization package for large-scale problems: Ii. implementation examples. *ACM Transactions on Mathematical Software*, 18(1):71–111, March 1992.

[159] J. Schmidt and H. Niemann. Using Quaternions for Parametrizing 3–D Rotations in Unconstrained Nonlinear Optimization. In T. Ertl, B. Girod, G. Greiner, H. Niemann, and H.-P. Seidel, editors, *Vision, Modeling, and Visualization 2001*, pages 399–406, Stuttgart, Germany, 2001. AKA/IOS Press, Berlin, Amsterdam.

[160] A. Schrijver. *Theory of linear and integer programming.* John Wiley and Sons, 1998.

[161] S. Sheng. Interpolation by conic model for unconstrained optimization. *Computing*, 54(1):83–98, March 1995.

[162] H. D. Sherali and C. H. Tuncbilek. A reformulation-convexification approach for solving nonconvex quadratic programming problems. *Journal of Global Optimization*, 7:1–31, 1995.

[163] K. Shoemake. Euler angle conversion. In *Graphics gems IV*, pages 222–229. Academic Press Professional, Inc., San Diego, CA, USA, 1994.

[164] F. J. Solis and R. J.-B. Wets. Minimization by random search techniques. *Mathematics of Operations Research*, 2:19–30, 1981.

[165] J. Stolfi and L. H. de Figueiredo. Self-validated numerical methods and applications, 1997.

[166] R. Storn and K. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[167] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, October 2005.

[168] J. S. Taylor and R. M. Burnett. DARWIN: A program for docking flexible molecules. *Proteins*, 41(2):173–191, 2000.

[169] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing: Vienna, Austria, 2009.

[170] R. Thomsen and M. H. Christensen. MolDock: a new technique for high-accuracy molecular docking. *Journal of Medicinal Chemistry*, 49:3315–3321, 2006.

[171] S.B. Thrun, J. Bala, E. Bloedorn, I. Bratko, B. Cestnik, J. Cheng, K. De Jong, S. Dzeroski, S. E. Fahlman, D. Fisher, R. Hamann, K. Kaufman, S. Keller, I. Kononenko, J. Kreuziger, R. S. Michalski, T. Mitchell, P. Pachowicz, Y. Reich, H. Vafaie, W. Van de Welde, W. Wenzel, J. Wnek, and J. Zhang. The monk's problems - a performance comparison of different learning algorithms. Technical report, Carnegie Mellon University, 1991.

[172] M. J. Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, August 2001.

[173] J.-Y. Trosset and H. A. Scheraga. PRODOCK: Software package for protein modeling and docking. *Journal of Computational Chemistry*, 20(4):412–427, 1999.

[174] I. Ulitsky, R. Karp, and R. Shamir. Detecting Disease-Specific dysregulated pathways via analysis of clinical expression profiles. In *Research in Computational Molecular Biology*, pages 347–359. Springer Berlin, Heidelberg, 2008.

[175] D. M. van Aalten, R. Bywater, J. B. Findlay, M. Hendrich, R. W. Hooft, and G. Vriend. PRODRG, a program for generating molecular topologies and unique molecular descriptors from coordinates of scmall molecules. *Journal of Computer-Aided Molecular Design*, 10(3):255–262, 1996.

[176] D. Vandenbussche and L. Nemhauser. A branch-and-cut algorithm for nonconvex quadratic programs with box constraints. *Mathematical Programming A*, 102:559–575, 2005.

[177] D. Vandenbussche and L. Nemhauser. A polyhedral study of nonconvex quadratic programs with box constraints. *Mathematical Programming A*, 102:531–557, 2005.

[178] S. A. Vavasis. Convex quadratic programming. In *Nonlinear Optimization: Complexity Issues*, pages 36–75. Oxford University Press, 1991.

[179] G. M. Verkhivker, D. Bouzida, D. K. Gehlhaar, P. A. Reijto, S. Arthurs, A. B. Colson, S. T.and Larson V. Freer, B. A. Luty, T. Marrone, and P. W. Rose. Deciphering common failures in molecular docking of ligand-protein complexes. *Journal of Computer-Aided Molecular Design*, 14:731–751, 2000.

[180] J. Vlček and L. Lukšan. Shifted limited-memory variable metric methods for large-scale unconstrained optimization. *Journal of Computational and Applied Mathematics*, 186(2):365–390, February 2006.

[181] A. Wächter and L. T. Biegler. Line search filter methods for nonlinear programming: Motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31, 2005.

[182] J. Wang, W. Wang, P. A. Kollmann, and D. A. Case. Automatic atom type and bond type perception in molecular mechanical calculations. *Journal of Molecular Graphics and Modelling*, 25(2):247–260, 2006.

[183] R. Wang, Y. Gao, and L. Lai. Calculating partition coefficient by atom-additive method. *Perspectives in Drug Discovery and Design*, 19:47–66, 2000.

[184] R. Wang, L. Lai, and S. Wang. Further development and validation of empirical scoring functions for structure-based binding affinity prediction. *Journal of Computer Aided Molecular Design*, 16:11–26, 2002.

[185] R. Wang, Y. Lu, and S. Wang. Comparative evaluation of 11 scoring functions for molecular docking. *Journal of Medicinal Chemistry*, 46:2287–2303, 2003.

[186] R. Wang and S. Wang. How does consensus scoring work for virtual library screening? an idealized computer experiment. *Journal of Chemical Information and Computer Science*, 41:1422–1426, 2001.

[187] S. J. Watowich, E. S. Meyer, R. Hagstrom, and R. Josephs. A stable, rapidly converging conjugate gradient method for energy minimization. *Journal of Computational Chemistry*, 9(6):650–661, September 1988.

[188] A. Watt and M. Watt. *Advanced Animation and Rendering Techniques–Theory and Practice*. ACM Press, 1992.

[189] P. K. Weiner and P. A. Kollman. Amber: Assisted model building with energy refinement. a general program for modeling molecules and their interactions. *Journal of Computational Chemistry*, 2(3):287–303, 1981.

[190] P. Wolfe. A simplex method for quadratic programming. *Econometrica*, 27(3):382–398, 1959.

[191] L. A. Wolsey. *Integer Programming*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons, New York, NY, 1998.

[192] D. Xie and T. Schlick. Efficient implementation of the truncated-Newton algorithm for large-scale chemistry applications. *SIAM Journal on Optimization*, 10(1):132–154, 1999.

[193] P. Xu. A hybrid global optimization method: The one-dimensional case. *Journal of Computational and Applied Mathematics*, 147:301–314, 2002.

[194] P. Xu. A hybrid global optimization method: The multi-dimensional case. *Journal of Computational and Applied Mathematics*, 155:423–446, 2003.

[195] P. Xu. Numerical solution for bounding feasible point sets. *Journal of Computational and Applied Mathematics*, 156:401–419, 2003.

[196] Y. Zhao, T. Cheng, and R. Wang. Automatic perception of organic molecules based on essential structural information. *Journal of Chemical Information and Modeling*, 47:1379–1385, 2007.

Publications

Papers in refereed journals

A. Rurainski, A. Hildebrandt, and H.-P. Lenhof. A consensus line search algorithm for molecular potential energy functions. *Journal of Computational Chemistry*, 30(9):1499–1509, July, 2009.

J. Fuhrmann, A. Rurainski, H.-P. Lenhof, and D. Neumann. A new method for the gradient-based optimization of molecular complexes. *Journal of Computational Chemistry*, 30(9):1371–1378, July, 2009.

J. Fuhrmann, A. Rurainski, H.-P. Lenhof, and D. Neumann. A new Lamarckian genetic algorithm for flexible ligand-receptor docking. *Journal of Computational Chemistry*, DOI: 10.1002/jcc.21478, 2010.

Papers in proceedings

A. K. Dehof, A. Rurainski, H.-P. Lenhof, and A. Hildebrandt. Automated bond order assignment as an optimization problem. *German Conference on Bioinformatics*, 2009.

Manuscripts in preparation

A. Rurainski, J. Fuhrmann, S. Roth, D. Neumann, and H.-P. Lenhof. Protein-ligand docking with backbone flexibility.

A. Rurainski, J. Fuhrmann, T. Schackmann, D. Neumann, and H.-P. Lenhof. Flexible protein-ligand docking with backbone movements.

A. Rurainski, M. Hein, A. Keller, and H.-P. Lenhof. Optimal feature selection based on mutual information second order criterion.

C. Backes, A. Rurainski, A. Gerasch, G. Klau, J. Küntzer, D. Eggle, M. Hein, A. Keller, H. Burtscher, M. Kaufmann, E. Meese, and H.-P. Lenhof. An integer linear programming approach for finding deregulated subgraphs in regulatory networks using expression profiles.

A.-K. Dehof, A. Rurainski, Q. B. A. Bui, S. Böcker, H.-P. Lenhof, and A. Hildebrandt. Automated Bond Order Assignment as an Optimization Problem.

A. Hildebrandt, A.-K. Dehof, A. Rurainski, A. Bertsch, M. Schumann, N. C. Toussaint, A. Moll, D. Stöckel, S. Nickels, H.-P. Lenhof, and O. Kohlbacher. BALL – Biochemical Algorithms Library 1.3.

Presentations at international conferences

J. Fuhrmann, A. Rurainski, H.-P. Lenhof, and D. Neumann. A new method for gradient based optimization of partially flexible molecules using derivatives of a fair quaternion parametrization. *European Bioperspectives 2008 – Book of Abstracts*, 322–323, October 2008, Hannover, Germany.

Submitted presentations at international conferences

A.-K. Dehof, D. Stoeckel, S. Nickels, S. Mueller, A. Rurainski, M. Schumann, H.-P. Lenhof, O. Kohlbacher, and A. Hildebrandt. The Biochemical Algorithms Library (BALL) - Rapid Application Development in Structural Bioinformatics. *Fourteenth International Conference on Research in Computational Molecular Biology (RECOMB)*, Lisbon 2010, Portugal.

Curriculum Vitae

Personal Data

Name	Alexander Rurainski
Address	Kaiserstrasse 235 66133 Saarbrücken Germany
Date of birth	August 2, 1977
Place of birth	Saarbrücken, Germany
Citizenship	German

Education

1984–1987	Grundschule am Stiefel, Rentrisch, Germany
1987–1996	Abitur Gymnasium Johanneum, Homburg, Germany
1996–1997	Military service
1997–2004	Diplom Center for Bioinformatics Saarland University, Saarbrücken, Germany
2004–2010	PhD student Center for Bioinformatics Saarland University, Saarbrücken, Germany

Teaching Experience

Winter 2000/2001	Mathematischer Vorkurs für Chemiker, Biologen und Pharmazeuten, Lineare Algebra I
Winter 2001/2002	Lineare Algebra I

Summer 2002	Lineare Algebra II
Winter 2002/2003	Algebra I
Summer 2004	Seminar Bioinformatik, Programmierung II
Winter 2004/2005	Bioinformatik I, Seminar Bioinformatik
Winter 2005/2006	Bioinformatik I, Seminar Bioinformatik
Summer 2006	Bioinformatik II, Seminar Bioinformatik
Winter 2006/2007	Bioinformatik I, Seminar Bionformatik
Summer 2007	Bioinformatik II, Seminar Bioinformatik
Winter 2007/2008	Bioinformatik I, Seminar Bioinformatik
Summer 2008	Seminar Bioinformatik
Winter 2008/2009	Bioinformatik I, Seminar Bioinformatik
Summer 2009	Bioinformatik II, Four seminars Bioinformatik
Winter 2009/2010	Bioinformatik I, Four seminars Bioinformatik

May 10, 2010