Evolutionary Coupling Methods in de novo Protein Structure Prediction

Stefan Seemayer

München 2016

Dissertation zur Erlangung des Doktorgrades
der Fakultät für Chemie und Pharmazie
der Ludwig-Maximilians-Universität München

Evolutionary Coupling Methods in de novo Protein Structure Prediction

Stefan Seemayer
aus München

2016

Erklärung

Diese Dissertation wurde im Sinne von § 7 der Promotionsordnung vom 28. November 2011 von Herrn Dr. Johannes Söding betreut.

Eidesstattliche Versicherung

Diese Dissertation wurde eigenständig und ohne unerlaubte Hilfe erarbeitet.

München, den 9. September 2016

Stefan Seemayer

Dissertation eingereicht am:	20.09.2016
1. Gutachter:	Dr. Johannes Söding
2. Gutachter:	Prof. Dr. Julien Gagneur
Mündliche Prüfung am:	22.05.2017

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

An understanding of protein tertiary structure is important for both basic and translational research, for example to understand molecular mechanisms, engineer new or optimized catalysts, or formulate new cures. Protein tertiary structures are typically determined experimentally, a time-consuming process with average costs in the hundred thousands of US dollars for determining a single protein structure. Consequently, there is much interest in using computational methods for driving down the cost of obtaining new structures.

While great successes have been made in transferring structural information from already structurally solved homologous proteins, the sensitivity improvements of methods for detecting homologous proteins have plateaued in recent years and homology-based protein structure prediction is ultimately limited by the availability of a suitable template that must be determined experimentally. *De novo* protein structure prediction could theoretically use physical models to determine the native conformation of a protein without prior structural information but in practice, such approaches are limited by the computational costs of evaluating expensive energy functions for many different points in an enormous search space.

An old idea in protein bioinformatics is to use the compensatory mutations observed due to the evolutionary pressure of maintaining a protein fold to predict which residue pairs in a protein structures are interacting in the folded structure. If such interactions can be reliably predicted, they can be used to constrain the search space of *de novo* protein structure prediction sufficiently so that the lowest-energy conformation can be found. Through recent improvements in the accuracy of such residue-residue interaction predictors, protein domain structures of typical size could be predicted in a blinded experiment for the first time in 2011. However, the new class of methods is still limited in its applicability in that methods are sensitive to false-positive predictions of interactions and can only provide reliable predictions with low false-positive rates for protein families that have a high number of homologous sequences.

This work aims to improve residue-residue contact predictions by improving the underlying mathematical models in a Bayesian framework. By explicitly modelling noise effects inherent in the underlying data and including priors to reflect the nature of residue-residue interactions, an attempt is made to reduce random and systematic errors inherent in contact prediction to make protein *de novo* structure prediction widely applicable.

Chapter 1

General Introduction to Protein Structure Prediction

1.1 Protein Structure Determination

Since the first protein 3D structure was solved by X-ray diffraction in 1958 [1], structural biology has grown into a field that produced many breakthroughs in basic and applied research and many Nobel prizes were awarded for the solution of protein structures through a variety of techniques. Using X-ray crystallography [2], Nuclear Magnetic Resonance Spectroscopy [3], Cryo-Electron Microscopy [4], and Mass Spectrometry [5], a large number of protein structures (currently more than $111,000$ [6]) have been solved and the files describing the coordinates of their atoms are deposited into the RSCB Protein Data Bank (PDB) [7] together with annotations such as associated publications, experimental details or the presence and atom coordinates of other molecules.

X-ray crystallography can be considered the workhorse of structural biology as 90% of PDB entries have been determined using this method, with NMR spectroscopy at 9% of structures and electron microscopy at 0.7% [6]. While X-ray crystallography has been widely used in the past, finding the correct experimental conditions to arrive at a pure and regular crystal remains challenging. This is especially the case for membrane proteins where hydrophobic surfaces and flexibility are commonplace [8] but the high-throughput screening of crystallization conditions [9] and the development of lipidic cubic phases [10] have lead to higher success rates in the crystallographic study of membrane proteins. On the other hand, Cryo-EM has become viable for solving near-atomic structures through the development of better electron detectors and image processing [11]. Though this maturation of cryo-EM technology, it can be expected that many new high-resolution cryo-EM structures will enter the PDB, especially of large symmetric protein complexes whose orientation can be efficiently classified by the image processing methods.

While the rate at which new protein structures are solved has steadily increased, the even bigger advances in high-throughput sequencing techniques have lead to a situation where the number of protein sequences grows much faster than the number of solved protein

structures. The resultant *sequence-structure gap* [12] is expected to widen as time passes.

In order to boost knowledge about previously unknown folds, *structural genomics* projects attempted to apply high-throughput approaches to protein structure determination and were successful at providing many new novel protein structures and reducing the average cost per solved structure. While successful at their goal, the projects were criticized for not being more efficient than leading traditional structural biology labs and publications on novel structures arising from structural genomics projects yielded a significantly smaller impact than publications on novel structures from traditional groups [13].

1.2 Approaches of Protein Structure Prediction

With the sequence-structure gap ever widening, structural bioinformatics attempts to bridge the gap through the computational prediction of protein structure using a combination of physical and statistical techniques. The major approaches to protein structure prediction and their benefits and challenges will be discussed in this section.

1.2.1 Homology Modelling

Since protein structure is more strongly conserved than protein sequence [14], there will be several protein sequences folding into the same structure, suggesting that protein folds and their corresponding sequences can be grouped into discrete clusters [15, 16] with overlaps between folds indicating the existence of common motifs [17].

The conservation of protein structures enables the application of a knowledge-based strategy for predicting the structure of unknown proteins: Using a sequence search method, the protein of interest can be assigned to one of the clusters of protein folds and thus predicted to be folding to the same structure as other sequences in that cluster. If structural information is already available for one member of the cluster, this information can be directly transferred by mapping equivalent residue positions [18], for example by generating distance restraints between residues from the template structure [19]. For proteins where suitable templates were available, the best-performing homology modelling methods were able to generate predictions with $C\alpha$-root mean square distances $< 3\mathring{A}$ from the native structure in the independent CASP11 evaluation [20].

Since homology modelling is only possible if a template structure can be found for the target protein, the success of homology modelling is dependant on both the availability of a template structure and the sensitivity and alignment quality of the sequence search method used to detect cluster membership. From alignment length and sequence identity, a *safe homology modeling zone* was defined for alignments with a sufficiently large overlap and agreement [21] with low-identity alignments falling into a *twilight zone* where it is unclear if a detected sequence homology would also lead to structural homology. More sensitive sequence search methods such as HHblits [22] were developed to push the limits of homology detection through the inclusion of more evolutionary information by comparing profile hidden markov models of query and template multiple sequence alignments.

Homology modelling leads to a multiplier effect for protein structure prediction: Every previously unknown protein fold that has a structure solved for one member of the cluster will allow the prediction of protein structures for all other members of that protein fold that can be detected using sequence search methods. However, many protein folds are challenging to determine due to experimental limitations so that homology modelling cannot be applied. For such protein folds, *de novo* protein structure prediction can provide an alternative solution.

1.2.2 De novo Protein Structure Prediction

De novo protein structure prediction is the discipline of predicting protein structures without prior information on the structure of that protein or related proteins but only general knowledge about intramolecular forces and protein folding. Typically, de novo protein structure predictors are based on force-field models used to rank the free energy of a conformation combined with efficient search strategies of exploring conformational space. Since the native protein structure generally corresponds to the lowest-energy conformation [23], given an accurate force-field model, finding the lowest-energy conformation will correspond to finding the native conformation.

Due to the high degree of conformational flexibility in proteins and the computational cost of computing all-against-all forces on atoms, the search space of all possible conformations cannot be explored exhaustively for proteins of typical lengths. For this reason, de novo methods need to reply on simplifications to decrease computational complexity in the choice of atom representation, forcefield model and amount of conformational freedom.

In order to assign an energy to a conformation under consideration, force-field models sum up Lennard-Jones and electrostatic terms between atoms in the molecule, combined with terms for hydrogen bonding, ϕ/ψ angle preferences, and torsion angles to make the molecule more protein-like [24]. Since the amount of interactions to be calculated increases quadratically with the amount of atoms for which interactions are considered, a logical simplification is to reduce the amount of interacting objects by implicitly treating solvent water molecules through an additional term in the force field model [25] instead of calculating interaction terms between solvent molecules and the protein or other solvents.

Combined with a reduction of complexity in the force field model, conformational space is also reduced by relying on discrete libraries of side chain orientations called rotamers [26]. For the protein main chain, conformational space can be reduced by using local conformations of small peptide fragments that were sampled from other crystal structures [27] or treating secondary structure elements as rigid fragments to be assembled [28].

After the coarse-grained simulations arising from the discussed simplifications, many methods follow up with a second fine-grained refinement step to arrive at a final model. By careful tuning and extensive use of computing power, de novo methods have made excellent (down to a $C\alpha$-RMSD of $1.4\mathring{A}$ over 90 residues) predictions for a 112-residue target protein [29] but since the size of the conformational search space grows exponentially with the protein sequence, finding a lowest-energy conformation for longer proteins becomes prohibitively expensive without relying on external help.

1.2.3 Contact Maps for De Novo Structure Prediction

Residue-Residue Contact Maps are an alternative way of encoding protein structure that can be can be trivially generated from a set of $C\alpha$ atom coordinates. They are defined as $L \times L$ matrices of binary variables $C(i, j)$:

$$C(i,j) = \begin{cases} 1, & = D_\beta(i,j) < t \\ 0, & = \text{else} \end{cases} \tag{1.1}$$

where $D_\beta(i, j)$ is the euclidean distance between $C\beta$ atoms ($C\alpha$ for glycine) for residues (i, j) and t is a distance threshold (typically chosen as $t = 8\mathring{A}$). Figure 1.1a shows an example of a residue-residue contact map generated from a small protein domain.

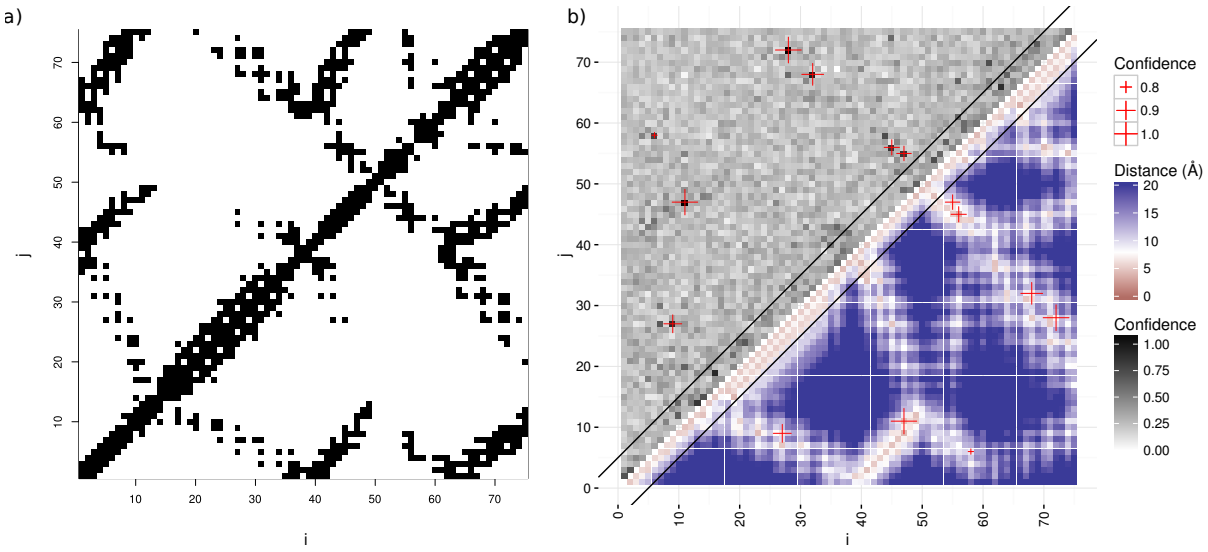


Figure 1.1: **a.** Residue-residue contact map of the human von Willebrand factor A3 domain (PDB code `1atz`), chain A. A black square is drawn at (i, j) if the $C - \beta$ atoms of residues i and j are closer than 8 Å.**b.** Predicted residue-residue contact map and true distance map for 1atzA. The top-$\frac{L}{10}$ most confident predictions are highlighted as red crosses.

While only a 2D representation, residue-residue contact maps retain the full 3D structural information of a protein so that $C\alpha$ atom coordinates can be reconstructed reliably using residue-residue contact maps as input. Perhaps surprisingly, contact maps are still informative when only part of their information is available: Partial contact maps generated from true 3D structures were shown to identify the true 3D structure out of a library of decoy structures [30, 31] and to constrain the search space of de novo protein structure prediction sufficiently to fold proteins to RMSD values of < 6.5 Å [32].

An explanation for why partial contact maps remain useful can be found in Figure 1.2: By picking only one residue-residue interaction for every tenth residue in the protein, the

orientation of secondary structure elements is still sufficiently constrained and the relative orientation of secondary structure elements constrains the overall structure sufficiently to quickly find the lowest-energy state. An attractive goal for protein structure prediction is therefore to reliably determine a few of the residue-residue contacts of a protein through another means than the full experimental determination of the protein structure since this would enable the full structure calculation from the limited set of restraints.

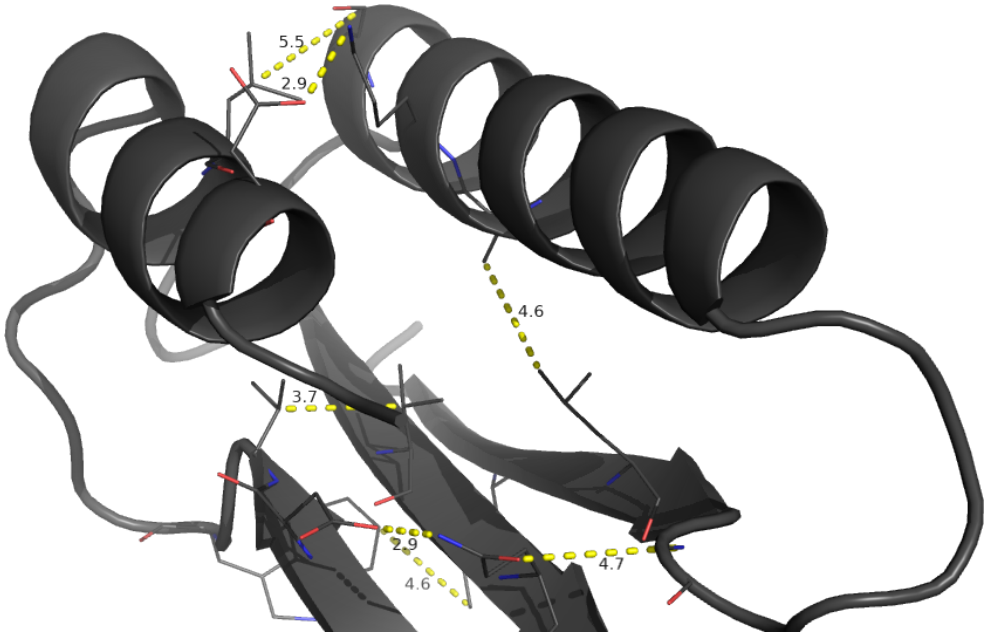


Figure 1.2: Top-$\frac{L}{10}$ residue-residue contact predictions mapped onto the X-ray crystal structure for 1atzA. The long-range residue-residue contacts constrain the orientation of secondary structure elements relative to one another, greatly reducing the combinatorial complexity of conformational sampling.

Experimental Determination of Residue-Residue Interactions

Several experimental techniques exist for deriving information residue-residue interaction, most importantly chemical cross-linking and nuclear overhauser effect spectroscopy.

Chemical Cross-Linking [33] uses specially designed and commercially available cross-linking reagents that covalently bind to two proximal functional groups in the protein (most frequently, the amino groups found at the protein N terminus and in lysine side chains). By varying the length of the *spacer* section of the cross-linking reagent, functional groups that are at different distances to another can be linked together. The cross-linking products are analyzed in a mass spectrometer, the residues participating in the cross-link identified and thus residue-residue distance restraints are determined.

Nuclear Overhauser Effect Spectroscopy (NOESY) [34] can be used to infer distance information using the Nuclear Overhauser Effect (NOE) that occurs when protons in a

dipole-dipole interaction exchange magnetic spins (either directly or indirectly through neighboring nuclei) in nuclear magnetic range (NMR) spectroscopy. By exploiting distance-dependent variations in the strength of the measured NOE, distance information between atoms can be derived after successful peak assignment in a 2D NMR experiment. The distance information derived from NOESY experiments is used as the primary information source when calculating protein structures from NMR data.

Computational Prediction of Residue-Residue Interactions

While experimental methods for determining residue-residue interactions can be applied successfully to predict unknown protein structures, progress has been made to predict residue-residue interactions computationally without reliance on experimental data. Instead of a binary contact map, residue-residue contact prediction calculates confidence scores that give a degree of belief for each residue combination (i, j) to be interacting. Figure 1.1b shows a high-quality residue-residue contact prediction with the most confident predictions highlighted compared with the distance map of true $C\beta/C\alpha$ coordinates.

Typically interpreted in a machine learning framework, different features of protein sequences such as conservation, sequence context, secondary structure and hydrophobicity were found to correlate with the propensity of forming residue-residue interactions, with the most descriptive feature being covariation between residue positions in a multiple sequence alignment of homologous protein sequences as a result of compensatory mutations. Chapter 2 will give an introduction to how previous contact prediction approaches made use of the residue-residue covariation signal and discuss subtleties that have to be taken into account when using it as input data.

Since the quality of residue-residue contact predictions controls the quality of the final protein structure prediction, special effort has to be made to arrive at the best-possible contact predictions. The aim of this thesis is therefore to improve the models underlying residue-residue contact prediction in a bayesian framework to make the best possible use of available data and expert knowledge about residue-residue interactions.

Part I

Precise Residue-Residue Contact Prediction

Chapter 2

Introduction to Residue-Residue Contact Prediction

Residue-residue contact prediction uses features calculated on multiple sequence alignments to predict with residue positions of a protein family are interacting. While there has been agreement for over 30 years [30] that contact predictions could be of use for protein structure prediction, the contact prediction methods developed until the late 2000s were not accurate enough for a blind 3D structure prediction as too many sources of noise caused many false-positive predictions. It took until 2008 for the major noise sources of entropic and phylogenetic effects [35] and until 2009 for indirect couplings [36] to be eliminated. In 2011, it was finally shown that these improvements could be combined with a NMR-based structure prediction program to fold de-novo protein structures using only sequence data [37], causing a spike in interest in further development of even more accurate contact prediction methods.

This chapter will introduce the basic concepts, techniques and error sources of residue-residue contact prediction and present important previous methods.

2.1 Co-evolution as a Signal

The most important signal used in all modern residue-residue contact prediction methods is evolutionary coupling. Since proteins evolve under evolutionary pressure to maintain their function (and thus, their structure), there is selection pressure on the residue-residue interactions in the protein that are important for constraining the fold. In most cases, a high sequence conservation at these residue positions will be expected over the evolutionary history of a protein family, but another way that interactions in a fold can be preserved is to compensate for the mutation of one interaction partner by also mutating the other interaction partner.

Consider a salt bridge interaction between a positively charged arginine residue and a negatively charged aspartate residue as seen in Figure 2.1a. If the arginine residue is exchanged into a lysine residue having a slightly smaller side chain, the interacting aspar-

tate residue can grow into a slightly longer glutamate residue to maintain the interaction distance. Alternatively, should the charge of the arginine change into a aspartate or glutamate, the interacting aspartate can mutate into an arginine or lysine. Similar sets of compensatory mutations can be found for hydrophobic interactions.

Since multiple sequence alignments can be seen as the evolutionary history of the extant homologous sequences of a protein family, they also reflect compensatory mutations that have occurred as co-occurring amino acid pairs in interacting MSA columns as shown in Figure 2.1b. *Evolutionary coupling* residue-residue contact prediction methods use this preference of amino acid pairs to occur together in the MSA more often than would be expected by random chance as their main input signal. A simple contact prediction method could therefore be constructed by just measuring the correlation (or alternatively, mutual information) between all column pairs in the multiple sequence alignment and predicting the n most highly correlated column pairs as contacting residues.

2.2 Error Sources for Contact Prediction

While some successes were achieved with using correlation or mutual information measures for contact prediction, strong noise sources caused many false-positive predictions making early contact prediction methods too unreliable for use in de novo protein structure prediction. So far, several sources of error have been characterized and attempts have been made to overcome these sources of error to varying levels of success. Figure 2.2 summarizes the most important error sources that have been characterized.

Sample Size Effects The amount of data available for residue-residue contact prediction is limited to the number of homologous sequence information that can be detected for the protein family under study. For insufficient data scenarios, the many different possible amino acid combinations at different position combinations will cause true interaction signals to be almost indistinguishable from random statistical noise. These effects are normally combatted through better statistical modelling, combined with the hope for more data to become available through high-throughput sequencing experiments.

Entropic Effects Since covariation-based measures need sequence variation as a signal for predicting residue-residue contact, the most highly conserved interactions in a protein are paradoxically more difficult to detect since they display little variance [38, 39, 35]. Unless entropy is accounted for in the contact prediction, this would lead to an underestimation of true coupling scores in highly conserved columns while columns with high variance being predicted to contain more contacts on average.

Phylogenetic Effects Almost all residue-residue contact prediction use mathematical formulations that assume statistical independence between individual MSA sequences. Since the sequences have evolved from a common ancestry, however, phylogenetically more closely related sequences in the multiple sequence will still have statistical dependencies

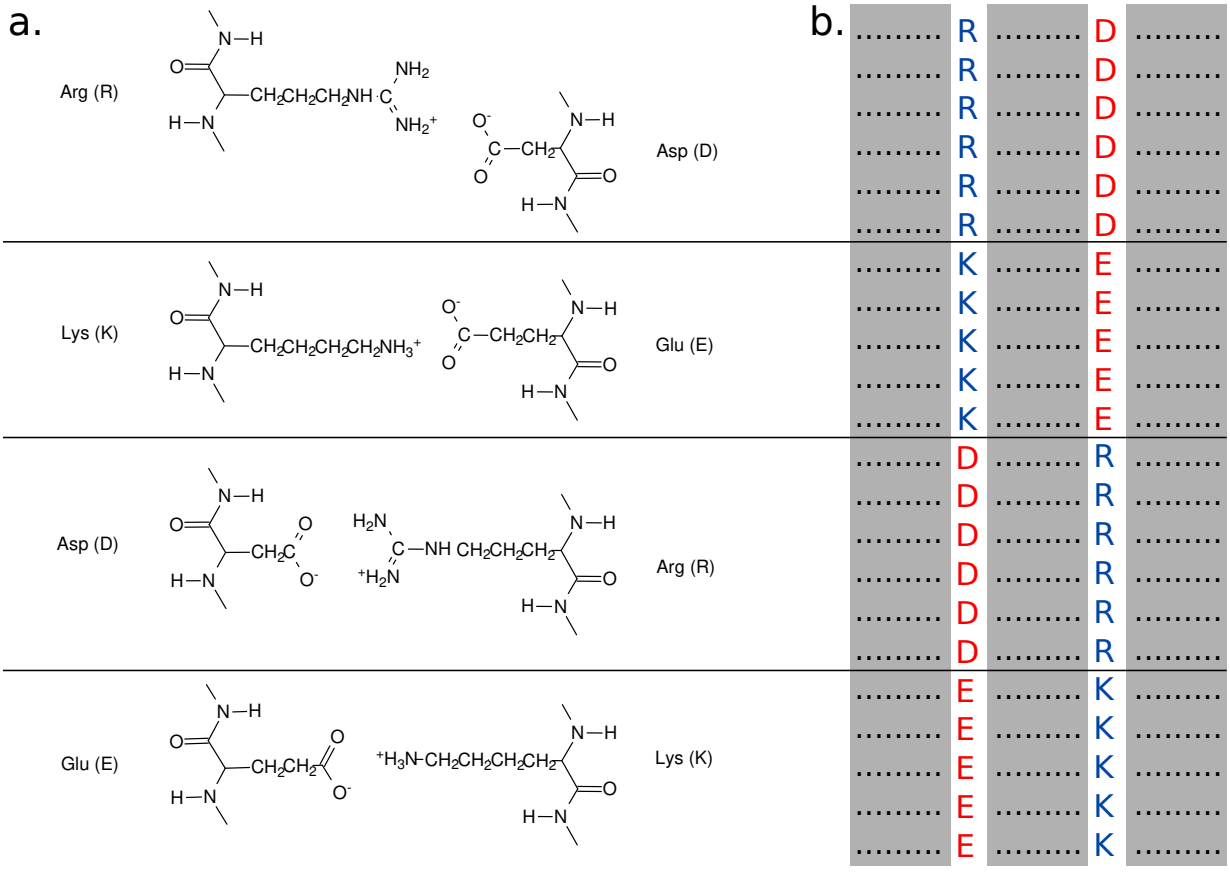


Figure 2.1: Acceptable substitutions for a Arg-Asp salt bridge. Both size and charge changes can be compensated by a size and charge change in the interaction partner. **a.** Lewis structures of interacting side chain residues showing the size and charge compensations. **b.** Multiple sequence alignment representation of the two interacting columns. Compensatory mutations can be seen as co-occurrence of interacting residue pairs.

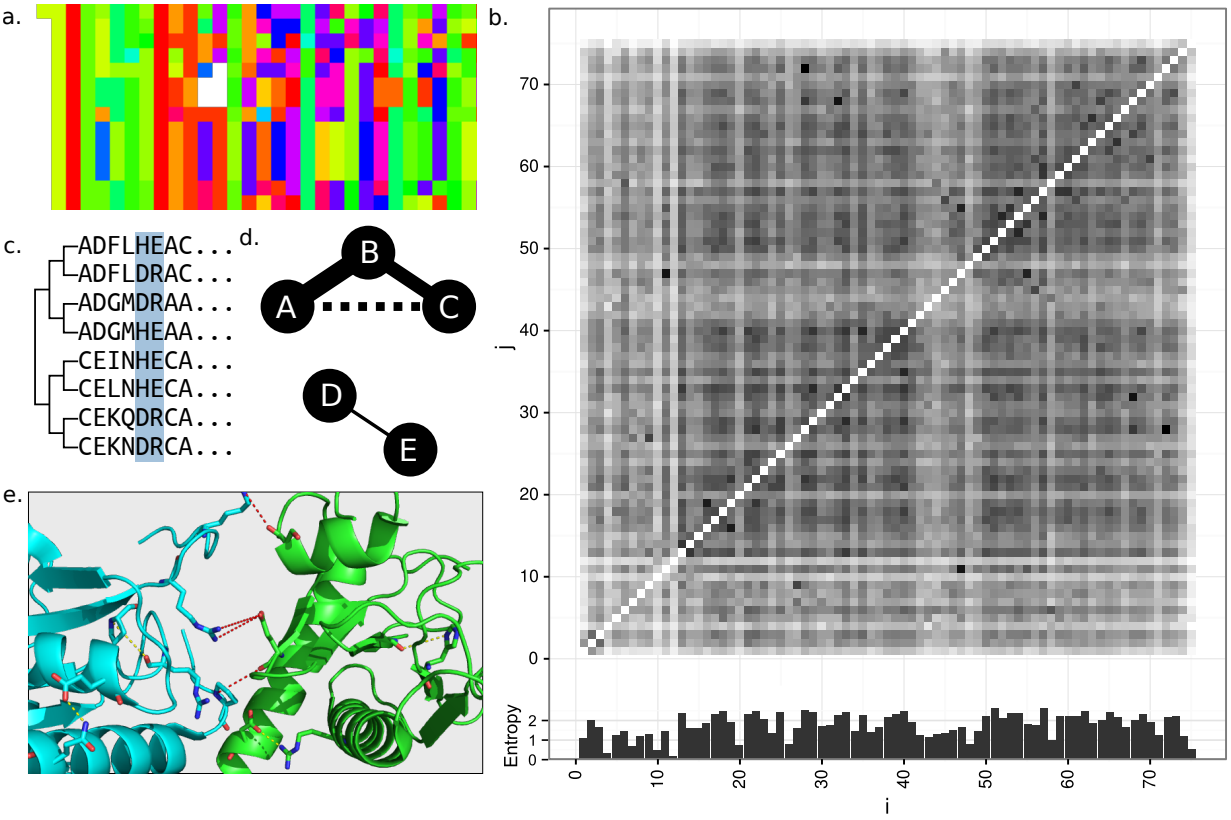


Figure 2.2: Noise Sources in Contact Prediction **a.** Sampling Size Effects. Residue-residue contacts learned from protein MSAs with a low number of homologous sequences in relation to the number of residue position will have high amounts of statistical noise, leading to inaccurate predictions. **b.** Entropic Effects. Overall coupling values are determined by the amount of variation that can be measured in a MSA position, leading to striped brightness patterns in the predicted contact maps. **c.** Phylogenetic Effects. Common ancestry of the extant sequences observed in the MSA can cause spurious correlation of residue positions in closely related sequences. **d.** Transitive Effects. Pairwise correlation between all MSA positions can lead to spatially distant residues `AC` being assigned a high coupling score if their constituent residues are part of other high-scoring, true interactions `AB`, `BC`. Such high coupling scores can become higher than true physical interactions `DE` in other parts of the protein and thus lead to false predictions. **e.** Intermolecular Couplings. On homomeric proteins, intermolecular couplings (red) have to be disentangled from intramolecular couplings (yellow) for de novo structure prediction to succeed (representative contacts mapped on PDB code 3O0M)

between their sequences as they have not had sufficient time to diverge [40]. This statistical dependence of sequences will look like a covariation signal to residue-residue contact prediction methods and thus lead to false-positive predictions.

Transitive Effects Correlation-based residue-residue contact prediction methods will assign high coupling scores to residue positions that belong to other, highly-coupled true residue-residue interactions even if they do not form a physical interaction themselves [40]. As shown in Figure 2.2b, such transitive coupling can lead to high-scoring false-positive interactions that can mask true-positive lower-scoring interactions in other parts of the protein. Transitive effects are typically eliminated by learning couplings using a global statistical model under the maximum entropy assumption [36, 41, 42].

Intermolecular Interactions If a protein family forms a multimer to maintain its structure of function, there will not only be selection pressure on maintaining the fold of the monomer but also on maintaining the interaction surfaces. In the case of homomultimers, the covariation from interactions stabilizing the fold have to be distinguished from multimeric interactions for structure predictions to succeed but can help in precisely identifying the structure of the protein-protein interaction once the structure of both interaction partners has been solved [43, 44, 45].

From the sources of errors discussed above, it becomes clear that better contact prediction accuracy can be achieved by effectively eliminating these noise sources. The following sections will discuss previous milestones.

2.2.1 Correcting for Entropic Effects

One of the biggest improvements to contact prediction accuracy comes from correcting entropic effects using the heuristic known as Average Product Correction (APC) [35]. Dunn *et al.* proposed that the coupling signal observed in a predicted contact map $C(i, j)$ consists of the coupling due to structural and functional constraints $C_{sf}(i, j)$ and a background term $C_b(i, j)$ that is related to per-position entropy and phylogenetic coupling. Since most position pairs in a contact map are expected to be non-contacting, they propose that the background term C_b can be estimated using per-column and per-row average coupling values $\bar{C}(i)$ and the average coupling value over the whole predicted contact map $\bar{C}$:

$$C_b(i, j) = \frac{\bar{C}(i)\bar{C}(j)}{\bar{C}}, \quad C_{sf}(i, j) = C(i) - C_b(i, j) \tag{2.1}$$

While APC has been originally proposed for mutual information-based contact prediction methods, it has been used to eliminate entropic effects in the majority of modern maximum entropy and pseudo-likelihood contact prediction methods since it also significantly boosts prediction accuracy there. Chapter 4 will discuss some of the characteristics

of the APC heuristic more deeply and suggest alternatives more suitable for future method development.

2.2.2 Correcting for Sequence Redundancy

Since the protein sequence space of a protein family is typically not sampled uniformly but biased in many different ways, it can be expected that some protein sequences and variants appear more frequently in a multiple sequence alignment than others, overemphasizing the covariations measured in that branch of the evolutionary tree. Most contact prediction methods correct for this effect by introducing weights that reduce the influence of sequences that have a high sequence identity to many other sequences in the alignment. A common strategy is to weight sequence i by the reciprocal of the number of sequences with a sequence identity of at least 90% to sequence i:

$$w_i = \frac{1}{\sum_{n=1}^{N} I(ID(i,n) \geq 90\%)}$$

Section 3.6 will discuss experiments with alternative weighting schemes to better account for redundancy in MSAs.

2.2.3 Correcting for Transitive Effects

The second major improvement of residue-residue contact prediction was the elimination of transitive effects to arrive at direct couplings. Inferring direct coupling parameters is a problem that has come up in both the statistical physics (known here as the *Invere Ising* problem) and probability theory (known as *partial correlation*) communities.

In the simplest case [37], direct couplings can be obtained by computing the covariance matrix of all position pairs in the multiple sequence alignment $\boldsymbol{\Omega}$ and its inverse $\boldsymbol{P} = \boldsymbol{\Omega}^{-1}$. The inverse covariance matrix (also known as the *precision matrix*) then can be used to obtain the partial correlation ρ:

$$\rho_{i,j} = -\frac{P_{i,j}}{\sqrt{P_{ii}P_{jj}}}$$

Since the covariation matrix is not necessarily invertible, a *shrinking* procedure [42] may be used to simplify convergence by increasing the diagonal elements of the covariance matrix. Since covariance is typically estimated in a low-data scenario, a regularization term introduced to the inversion objective function additionally helps direct coupling values for potentially interacting pairs with low amounts of evidence to zero, reducing noise.

2.3 Markov Random Fields

After the initial successes with inverse matrix approaches, shrinking heuristics and regularization, a more user-friendly and bayesian framework for the elimination of transitive

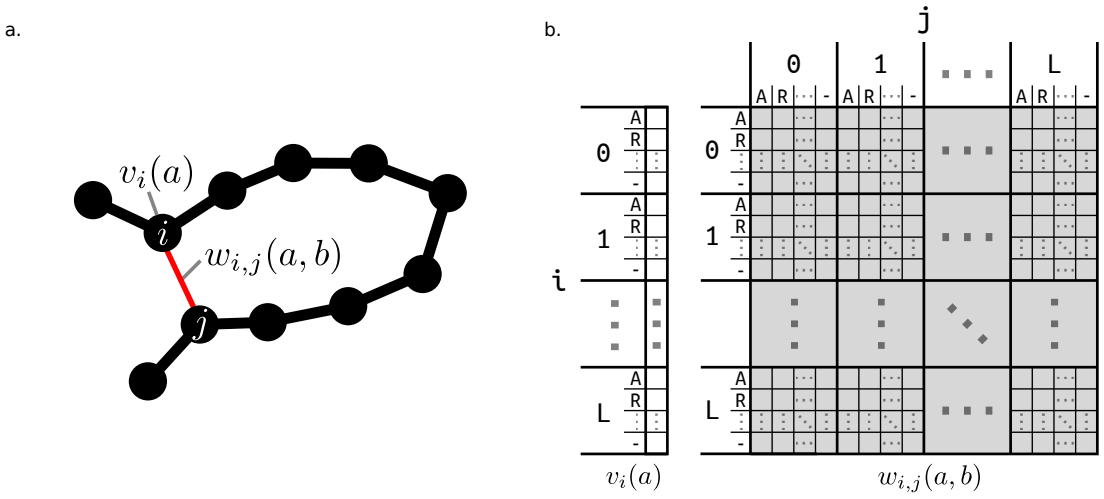


Figure 2.3: Markov Random Fields for Contact Prediction. **a.** Visual representation of graph structure. Each vertex corresponds to a column in the multiple sequence alignment while every edge corresponds to covariation between MSA columns. **b.** Visualization of the $L \times 21$ $v_i(a)$ and the $L \times L \times 21 \times 21$ $w_{i,j}(a,b)$ matrices used to parametrize the model.

effects was found by directly learning the parameters of a global model of all residue positions using maximum-pseudo-likelihood approaches [41]. A Markov Random Field (MRF) is an undirected graphical model allowing cycles that encodes all amino acid preferences and residue-residue couplings into a joint probability distribution of observing a protein sequence $\vec{x} = (x_1, \ldots, x_L)$ given single-emission potentials $v_i(a)$ and pairwise emission potentials $w_{i,j}(a,b)$:

$$P(\vec{x}|\boldsymbol{v}, \boldsymbol{w}) = \frac{1}{Z} \prod_{i=1}^{L} \exp\left[v_i(x_i)\right] \prod_{\substack{i,j=1 \\ i \neq j}}^{L} \exp\left[w_{i,j}(x_i, x_j)\right] \tag{2.2}$$

where Z is a normalization term also known as the *partition function* that sums over all possible sequence assignments to ensure the probability is normalized:

$$Z = \sum_{x' \in \{1 \ldots 20\}^L} P(x'|\boldsymbol{v}, \boldsymbol{w}) \tag{2.3}$$

Since the normalization term can only be computed in exponential time, the MRF likelihood is typically replaced by a *pseudo-likelihood* that has been proven to converge to the same optimum as the number of sequences increases [46]. In a pseudo-likelihood, the normalization is localized for a column i and will only sum over the 20 possible assignments for that column while keeping the other positions fixed to the sequence context:

$$P_{pseudo}(x|\boldsymbol{v},\boldsymbol{w}) = \prod_{i=1}^{L}\left(\frac{1}{Z_i}\exp\left[v_i(x_i)\right]\prod_{\substack{j=1\\i\neq j}}^{L}\exp\left[w_{i,j}(x_i,x_j)\right]\right) \tag{2.4}$$

$$Z_i = \sum_{a=1}^{20}\exp\left[v_i(a)\right]\prod_{\substack{j=1\\i\neq j}}^{L}\exp\left[w_{i,j}(a,x_j)\right] \tag{2.5}$$

Because of the logarithmic parametrization, a coupling score of 1 can be understood as the coupling being e^1 times more likely than would be expected if the corresponding positions were independent. This parametrization leads to a gauge invariance since multiple parametrizations can express the same probability distribution. The indeterminacy can be fixed by including a regularization prior that will favor the solution with all parameters closest to zero and can also be used to include prior knowledge. A common example is to use an L_2 prior with regularization parameters λ_s, λ_p to push single and pairwise emission parameters with low amounts of evidence towards zero and also reduce overfitting:

$$R(\boldsymbol{v},\boldsymbol{w}) = \lambda_s\left\|\boldsymbol{v}\right\|_2^2 + \lambda_p\left\|\boldsymbol{w}\right\|_2^2 \tag{2.6}$$

Note that the aforementioned L_2 regularization is equivalent to the negative logarithm of two gaussian priors centered around 0 with scale parameter $\lambda_s^{-1}, \lambda_p^{-1}$. Instead of simply maximizing the pseudo-likelihood, we can now maximize the pseudo-log-likelihood minus regularization term to arrive at a maximum a posteriori (MAP) estimate:

$$\hat{\boldsymbol{v}},\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{v},\boldsymbol{w}}\mathrm{pll}(\boldsymbol{X}|\boldsymbol{v},\boldsymbol{w}) - R(\boldsymbol{v},\boldsymbol{w}) \tag{2.7}$$

2.4 Machine Learning-based Predictors

When looking at the top-performing contact prediction methods published so far, the most precise methods [47] such as MULTICOM and pConsC3 combine many input features into complicated machine learning apparatus to maximize the prediction score. While such approaches yield impressive performance metrics, they are slow to train and are the result of heuristic experimentation with input feature construction and learning strategy. Furthermore, the trained predictors are too complex for human comprehension and thus have to be used as a "black box".

Since many of the top-performing predictors use a MRF-based contact predictor as their main input feature and MRF models are still humanly comprehensible and are inviting for better bayesian modelling, the focus of this work is on the improvement of MRF-based contact prediction methods that is also expected to lead to improvements in the accuracy of the complex black box predictors.

2.5 Predicting 3D Structures from Contact Maps

The second part to de novo protein structure prediction is calculating protein 3D structures from predicted residue-residue contact maps. While structure calculation is not part of the thesis project, understanding some of the characteristics of structure calculation is important for producing meaningful predictions. The following section will therefore give a brief overview over common approaches.

2.5.1 Residue-Residue Contacts as Distance Restraints

Since the restraints generated by contact prediction methods are similar to the residue-residue distance constraints that can be determined from NMR nuclear overhauser effect (NOE) experiments, an obvious approach for solving protein 3D structure using predicted residue-residue contacts is to use the existing tools for folding protein structures from experimental data. The EVfold method [37] had some initial successes with the Crystallography and NMR System (CNS) [48] and CNS is still employed in newer pipelines such as CONFOLD [49] since it can still run well on a moderately powerful desktop computer. On the other hand, the powerful and computationally expensive Rosetta AbInitioRelax [50] is employed in approaches such as PConsFold [51] and further improved for the residue-residue contact prediction scenario by RASREC [52].

2.5.2 Specialized Programs for Residue-Residue Contacts

Since existing forcefield-based approaches are typically very computationally expensive, some interest has gone into developing custom protein structure prediction methods for solving protein structures using predicted contact maps. Instead of using all-atom representations and a forcefield model, only $C\beta$ atom coordinates are initially shuffled to fulfill the restraints of predicted contact maps and backbone geometry. After such an initial model has been solved, the overall fold of the protein is already determined, making the inclusion of side chains a matter of comparatively smaller conformational changes.

The FT-COMAR [53, 54] method made initial successes in predicting protein structures from noisy data by a heuristic that generates an initial random solution using a distance geometric algorithm combined with background knowledge of protein residue-residue distances and then iteratively refining the coordinates using different movement steps. Using contact maps derived from crystal structures that had noise artificially added, FT-COMAR can recover structures with $C\alpha$-RMSD ≤ 4 Å while using contact maps with 75% of the information masked out. On contact maps predicted using the CORNET method [55] taking the top-$\frac{L}{5}$ contacts, however, FT-COMAR only achieves $C\alpha$-RMSDs of ~ 17 Å.

The newer GDFuzz3D [56] method was developed to calculate all-atom 3D structures from predicted contact maps in a three-step procedure. First, the predicted contact map is interpreted as an adjacency map of the graph of residue contacts and a natural-numbered graph distance map is derived from the graph structure and then scaled to real-valued distances using statistics derived from PDB structure contacts. Next, the 2D distance

map is transformed into an euclidean distance map in 3D space and corresponding 3D coordinates using the multidimensional scaling algorithm [57]. Finally, an all-atom model is constructed from the coarse $C\alpha$ 3D model and optimized using MODELLER with $C\alpha - C\alpha$ distance restraints ≤ 4 Å and predicted secondary structure from SSpro4 [58]. On CASP10 single-domain targets outperforms FT-COMAR with average TM-scores of 0.41 (compared to 0.31 for FT-COMAR) and average RMSD values of 11.06 Å (compared to 14.88 Å for FT-COMAR). On the set of 150 protein families from the PSICOV data set, FT-COMAR (average TM-score 0.49) slightly outperforms the EVfold protocol (average TM-score 0.47) and is outperformed by the computationally much more expensive PconsFold (average TM-score 0.55).

2.5.3 Choosing Good Restraints for 3D Structure Prediction

The results of the aforementioned 3D structure prediction approaches repeatedly show that while 3D structures can be calculated from only a few number of contacts, all methods are very vulnerable to being derailed by false-positive contacts. A good contact prediction method should therefore generate restraints with a high specificity. Secondly, especially contacts between residues that are distant in sequence are important for constraining the correct fold since local structural phenomena can already be partially determined through reliable secondary structure prediction. A contact prediction method that achieves a high accuracy on few long-distance interactions is therefore clearly more informative than a method than can only generate many redundant local interactions.

2.6 Accuracy and Applicability of Contact Prediction

2.6.1 Evaluating Contact Prediction Accuracy

Since residue-residue contact prediction methods ultimately have to improve the accuracy of ab initio protein structure prediction, a reasonable strategy for evaluating might be to predict protein structures using different residue-residue contact predictions and then directly evaluating the TM-score or $C\alpha$ RMSDs of the models. Since the CASP10 experiment, predicted contacts have been shown to assist protein structure prediction [59] and since the recent CASP11 experiment, contact prediction models are employed in the challenging free modelling category [60], thus proving their usefulness, and full ab initio protein structure prediction pipelines evaluate their model quality using TM-score, $C\alpha$-RMSD and other measures [37, 51, 56, 49]. However, the structure calculation pipelines add a considerable amount of complexity of evaluation and so the precision and recall of individual residue-residue contacts is often used as a simpler proxy for structural modelling performance when developing a pure residue-residue contact predictor.

Using a test set of protein domains with experimentally well-determined coordinates and accompanying MSAs of homologous protein sequences, contacts are predicted for each MSA and each contact prediction method under comparison. For each of the protein

domains for which a contact prediction was found, contacts too close in sequence (e.g. $|i-j| \leq 6$) are masked out since they are trivial to predict and more distant contacts are grouped into sequence separation bins. As discussed in Section 2.5.3, especially contacts between distant sequence regions are of importance since they are informative for folding protein structures.

Using the filtered contact lists, the top-ranking n contacts are compared to the true 3D structures by assigning distances smaller than a threshold (typically, 8 Å are used) to be true positives (TP) and contacts further away than that threshold to be false positive (TP). The contacts that exist in structure but were not detected by the contact prediction method are denoted as false negatives (FN) while the residue pairs that are spatially distant and were not classified to be contacting are denoted as true negatives (TN). Using these assignments, several performance criteria typical for evaluating binary classifiers can be calculated:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.8}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.9}$$

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{2.11}$$

To compare different methods against another, a typical plot is the mean precision for the top-ranked n contacts averaged over all protein domains in the test set against the number of accepted contacts n, either as an absolute number or normalized by protein length. A popular single-figure performance criterion is the top-$\frac{L}{5}$ precision, or the mean precision of contact predictions when choosing the top-ranked $n = \frac{L}{5}$ contacts for each protein domain in the test set but other numbers such as the area under the ROC curve or the area under the precision-recall curve are also used.

Since the performance on the discussed evaluation criteria is crucially dependent on the set of protein domains used, methods can only be compared against each other fairly if they had the same MSA data available. A de facto standard set of protein MSAs is the set of 150 protein domains derived from Pfam that was originally published with the PSICOV method [42] but has to be considered unrealistically easy because of its high sequence coverage compared to real-life applications of predicting unknown protein structures. The CASP RR category [61, 62, 47] is an independent evaluation of residue-residue contact prediction during the CASP experiment. Participating methods are tasked to predict contacts for the targets considered to be hard homology modeling targets and the evaluation can therefore be considered more realistic although only performed on a smaller number of proteins.

2.6.2 Sequence Coverage and Applicability

Balakrishnan and Kamisetty argue that for reliable residue-residue contact predictions to be made, a protein family of L residues will need to have at least $5L$ homologous sequences aligned in its multiple sequence alignment for contact predictions to be reliable enough to be used in ab initio structure prediction [41, 63]. As seen in Figure 2.4, only a small minority of protein families annotated in the Pfam database currently meet this threshold and such families occur in so many branches of life that they are likely to have been extensively studied already and it is consequently also likely that an experimental structure has already been solved.

Since the sequence-structure gap is expected to continue to grow [12] due to the faster improvements in high-throughput sequencing technology compared to the increase in rate of protein 3D structures being solved, it can be expected that both the applicability as the importance of contact prediction will rise as time passes. Still, for rarer protein families that are only specialized into a small subtree of the tree of life, even an extensive sequencing of all extant species might not yield enough sequence coverage to make the family a suitable target for 3D structure prediction.

Through improving residue-residue contact prediction and 3D structure calculation methods, the influence of statistical noise in low-data prediction scenarios can be further reduced, allowing to extract more information out of the same amount of available data.

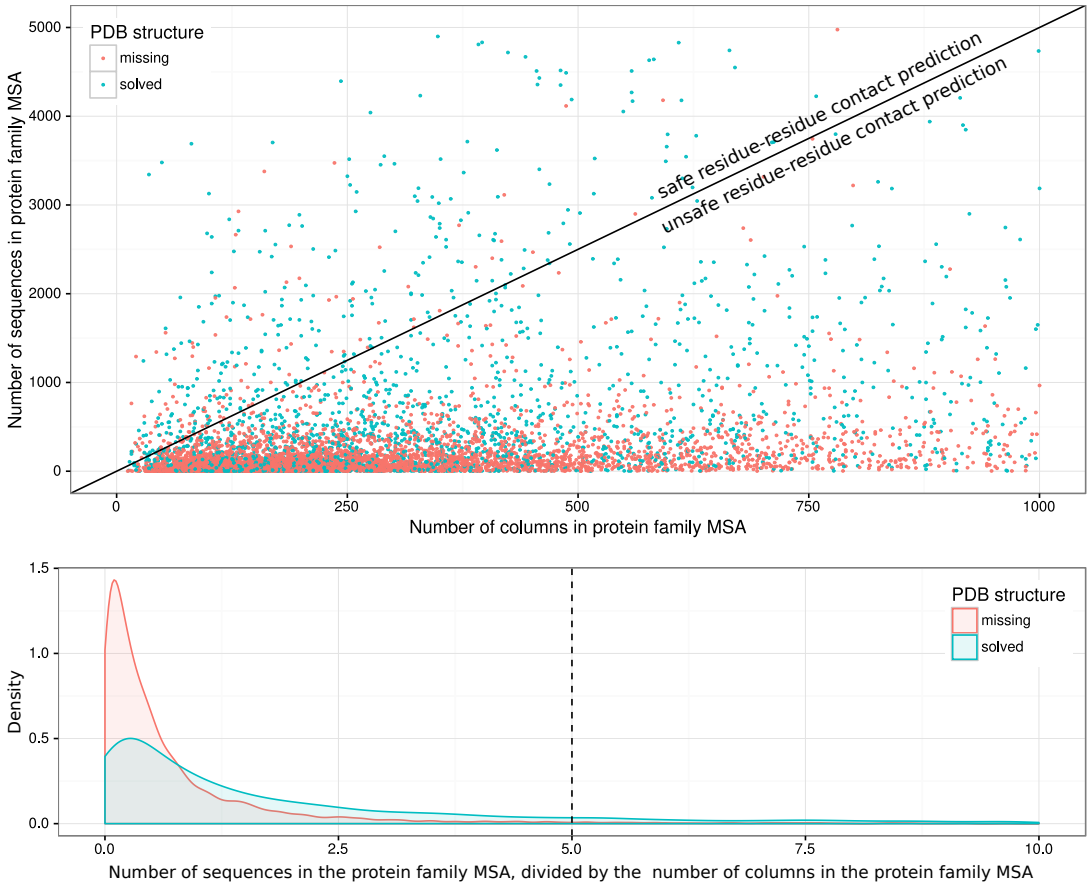


Figure 2.4: Sequence coverage of Pfam families relative to family size. For a protein family with L residues or columns in the multiple sequence alignment, there should at least be $5L$ homologous sequences so that the amount of information outweighs the statistical noise inherent in contact prediction. The $5L$ criterion is shown as the straight line in the plot with only protein families above the line currently being suitable targets for residue-residue contact prediction. As can be seen from the high number of points at the bottom of the plot, most Pfam families are currently not suited for contact prediction.

Chapter 3

Improving Residue-Residue Contact Prediction

3.1 Evaluation Dataset

To evaluate the predictive performance of contact prediction methods, two datasets of protein MSAs combined with high-resolution crystal structures were used:

Initially, the dataset of 150 Pfam families and corresponding crystal structures initially published with the PSICOV method [42] was used for most analyses since it has become a de-facto standard evaluation set in the contact prediction community because of its ease of use and availability. However, the protein families in the dataset are biased towards protein families with high amounts of homologous sequences in their MSAs.

In order to have a more realistic evaluation scenario with a focus on more difficult targets, Dr. Jessica Andreani built a more representative dataset using classes 1 (mainly α), 2 (mainly β) and 3 ($\alpha + \beta$) from the CATH database of protein domains [64], version 4. Using `pdbfilter` from the HH-suite scripts [22], domains were filtered for redundancy and for each fold, the domain with the highest-resolution crystal structure was kept as a representative. Alignments were enriched using HHblits using parameters to maximize the detection of homologous sequences (E-Value cutoff 0.1, 5 iterations, all filtering disabled with `-all`). After this procedure, each protein domain in the set is non-redundant to other protein domains on both the fold and sequence level and no bias is placed on a particular CATH class or the number of homologous sequence available for a domain. In total, 1231 domain MSAs and corresponding crystal structures were produced.

Since the difficulty of contact prediction is mostly dependent on the amount of sequence data available for every position in the protein family, the measure of *MSA diversity D* was defined in dependence of the number of sequences N and number of columns L in the MSA as a criterion for describing the difficulty of predicting contacts for a protein family:

$$D = \frac{\sqrt{N}}{L} \tag{3.1}$$

with $D \in [0, 0.1)$ considered very hard, $D \in [0.1, 0.3)$ considered hard, $D \in [0.3, 0.5)$ considered moderate difficulty, and $D \in [0.5, \infty)$ considered easy. The square root was chosen instead of a linear dependency on N since the square root term shows a better proportionality to the precision of contact predictions. Wherever possible, evaluation was performed separately for each of these bins to give special attention to improvements in the very hard and hard cases.

3.2 Explaining Phylogenetic Effects in Inverse Covariance Approaches

Since existing contact prediction methods make the wrong assumption that sequences in a multiple sequence alignment are drawn independently or only correct for the phylogenetic interdependence of sequences using sequence weighting or average coupling corrections, a reasonable approach to improving contact prediction accuracy is to include phylogenetic interdependence explicitly in the contact prediction.

A simple way to include phylogenetic information into the model is to add additional variables to each sequence that encode information about where in the phylogenetic tree that sequence is located. As shown in Figure 3.1, if having an amino acid at a sequence position can be explained by ancestry, couplings between sequence and phylogeny will be learned instead of trying to explain the appearance of the amino acid by residue-residue covariation.

3.2.1 Encoding Ancestry in Multiple Sequence Alignments

A phylogenetic tree was reconstructed using the FastTree 2 Maximum-Likelihood tree reconstruction method [65]. Starting from the root node of the phylogenetic tree, the first T (with T a configurable parameter to be optimized) nodes of the tree as determined by a breadth-first traversal were selected to be the roots of *sequence clusters* and all sequences having that node as a direct or indirect ancestor denoted to be members of that cluster. As seen in Figure 3.2, the evolutionary time covered by the multiple sequence alignment is crucial in selecting ancestral nodes that can form informative sequence clusters. If clusters too high up in the evolutionary tree are chosen and the extant sequences have fully diverged away from them, no phylogenetic coupling is expected to be observable for these clusters. Since it is expected that annotating too many clusters will only lead to non-informative coupling scores, large T were chosen to ensure that the informative levels of evolutionary distance are covered by the cluster annotations.

For the sake of simplicity, a binary encoding like in the PSICOV [42] and protein sectors [66] works was used to encode both amino acids and sequence cluster membership. Consequently, every sequence in the MSA corresponds to an $T + 20 \times L$ vector of binary variables.

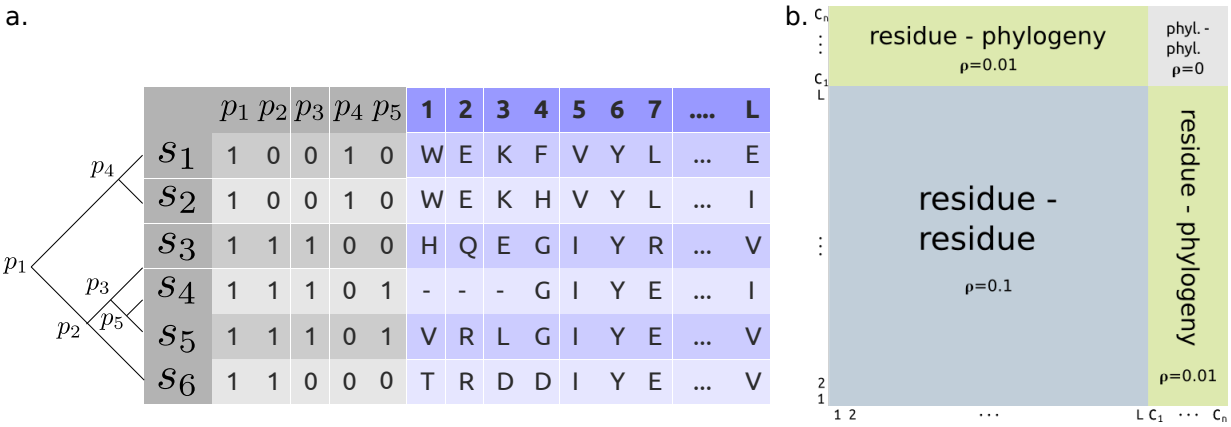


Figure 3.1: Correcting phylogenetic noise by subtree membership annotations. **a.** An example multiple sequence alignment with a phylogeny explaining common ancestry between sequences and phylogenetic subtree annotations $p_1 \dots p_5$. Most co-occurrences between amino acids can be explained by insufficient evolutionary time having passed for the sequences to diverge away from a common ancestor. **b.** Decomposition of the covariation matrix. By including additional phylogeny membership columns into the multiple sequence alignment, residue-phylogeny and phylogeny-phylogeny interactions can be encoded by the model so that residue-residue interactions that are due to phylogeny can be "explained away" through appropriate choices of regularization coefficients ρ.

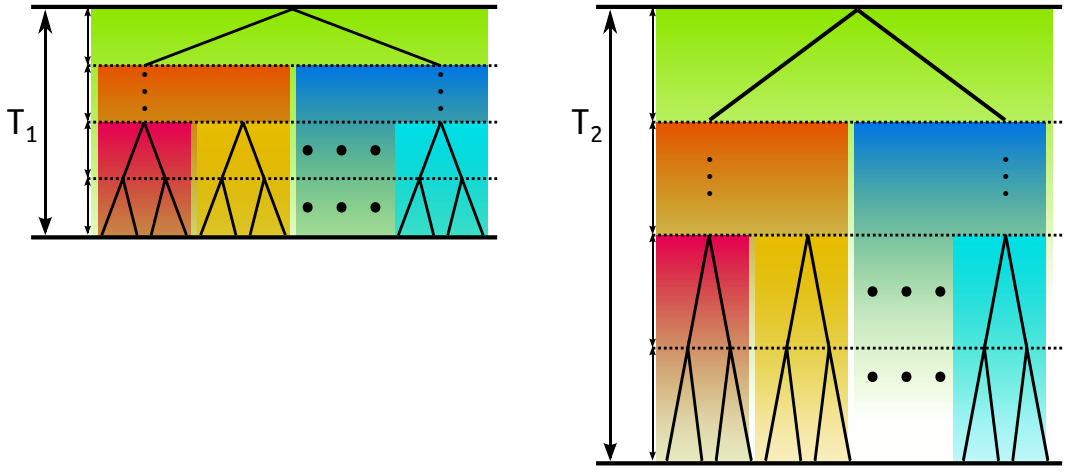


Figure 3.2: Time-dependent divergence away from common ancestors. While the sequence of each clade in the phylogenetic tree depends on the ancestral sequences above it, the dependency decreases exponentially with evolutionary time as sequences diverge. Depending on the total evolutionary time covered by the phylogenetic tree, the extant sequences might still show dependence on the root sequence of the phylogenetic tree if only a short amount of time (T_1, left) has passed or the extant sequences are correlated in clusters of more recent divergence in cases of longer evolutionary time (T_2, right).

3.2.2 Learning Sequence-Phylogeny Couplings

Similar to the PSICOV method, the $N \times (T + 20 \times 20)$ feature matrix is used to calculate a covariance matrix that can be inverted using a regularization term [67, 68] to arrive at direct interaction terms. As can be seen in Figure 3.1b, the matrix can be decomposed into sequence-sequence, sequence-phylogeny and phylogeny-phylogeny interactions and sequences that are observed due to a common ancestry will be explained as terms in the sequence-phylogeny sector while true residue-residue interactions can be found in the sequence-sequence sector.

Using just the true residue-residue interactions in the sequence-sequence sector, an L_1 norm is computed for each of the 20×20 submatrices corresponding to the amino acid combinations observable at a pair of residue positions. The resultant summed score matrix is processed using APC and then used to rank interacting residues.

3.2.3 Results of Phylogenetic Inverse Covariance

Figure 3.3 compare the contact prediction precision of phylogenetic inverse covariance with other inverse covariance approaches on the PSICOV dataset. While the phylogenetic inverse covariance leads to improvement over a non-phylogenetic inverse covariance before APC is applied, the non-phylogeny-corrected inverse covariance benefits more from APC and outperforms a phylogenetic inverse covariance without APC. Applying APC to the phylogenetic inverse covariance leads to a slight decrease in precision. The loss in precision after applying APC can be explained by assumptions APC makes over the distributions of the prediction scores. Chapter 4 will characterize these assumptions further and propose a replacement to APC.

3.3 $L_{2,1}$ Regularization of Pairwise Potential Matrices

The L_2 regularization seen in Equation 2.6 treats the coupling values of all pairwise residue combinations as independent from another. While this is a reasonable modeling assumption to simplify the MRF learning, it is not accurate for the contact prediction scenario.

When considering any pair (i, j) of residue positions within the protein fold, they can either be evolutionarily constrained to maintain a fold or none such constraint exist. If there is no evolutionary constraint, the correct solution for the MRF for this pair is to set $w_{i,j}(a, b) = 0$ for all (a, b) belonging to this residue combination. On the other hand, if the two positions are interacting and there is evidence for an interaction between a combination of amino acids (a, b), it is likely that other amino acid combinations (a', b') would also interact at the same position. In both cases of interacting and non-interacting residue positions, it is therefore reasonable to make the strength of the regularization on one pair of amino acids also depend on the strength of the coupling of other amino acid pairs at the same position.

The $L_{2,1}$ prior used in *group lasso* regression [69] is a regularization term that groups together related variables:

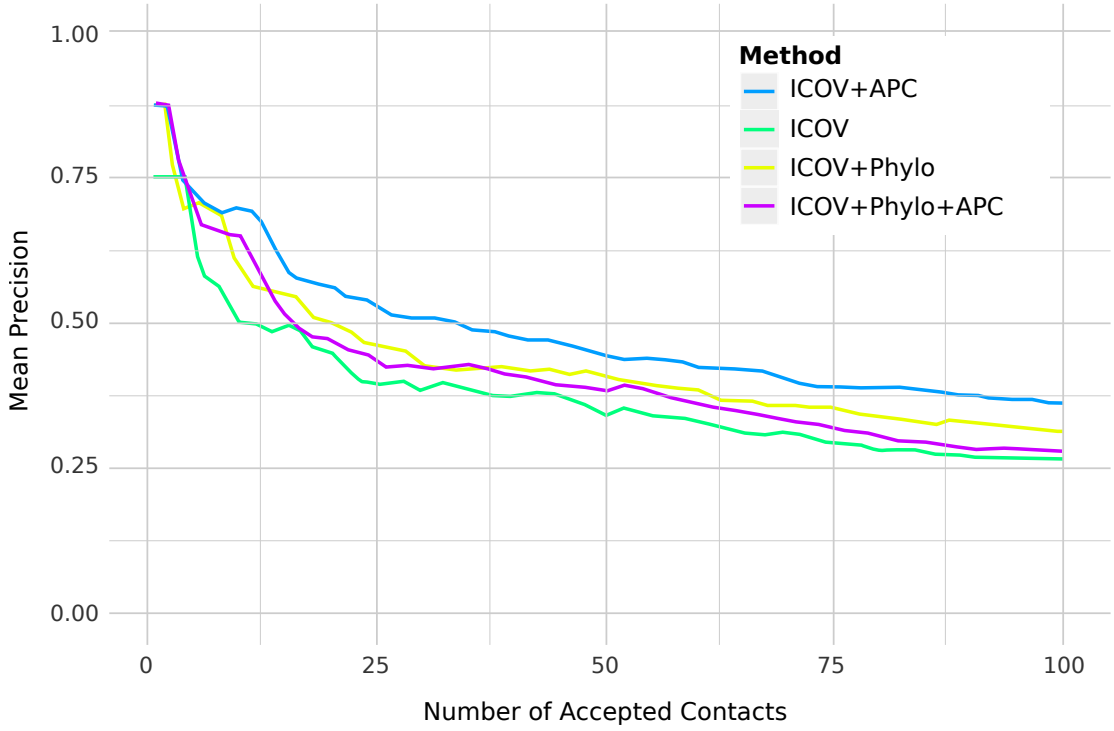


Figure 3.3: Evaluation of Phylogenetic Inverse Covariance on the PSICOV dataset. While phylogenetic inverse covariance boosts precision over inverse covariance, it cannot benefit from a precision boost by Average Product Correction.

$$R_{L_{2,1}}(\boldsymbol{w}) = \lambda_{2,1} \sum_{\substack{i,j=1 \\ i \neq j}}^{L} \|w_{i,j}\|_2 = \lambda_{2,1} \sum_{\substack{i,j=1 \\ i \neq j}}^{L} \sqrt{\sum_{a,b=1}^{20} w_{i,j}(a,b)^2} \tag{3.2}$$

Consider a simple two-dimensional example with $R = \sqrt{x_1^2 + x_2^2}$. As can be seen in Figure 3.7a, if $x_2 = 0$, the gradient for x_1 will push x_1 to zero with the same force as an L_1 regularization term. If both x_1 and x_2 are nonzero, however, the strength of the regularization term will be divided between both variables. In the case of contact prediction, grouping together the 20×20 entries for each (i, j) submatrix of $w_{i,j}(a, b)$ will share the regularization pressure between all amino acid combinations found at a residue position pair while position pairs with insufficient evidence have all of their (a, b) coupling values strongly pushed towards zero.

The $L_{2,1}$ prior was integrated as a new regularization term using the custom optimization framework discussed in Section 3.5 to account for non-differentiable regions in the objective function and evaluated for contact prediction accuracy improvements.

3.3.1 Results for $L_{2,1}$ prior

Figure 3.4 compares the contact prediction precision for protein MSAs of the PSICOV data set. While slight improvements to the precision of $L_{2,1}$-regularized contact predictions compared to L_2-regularized contact predictions can be seen before APC was applied, the APC-corrected L_2-regularized contact predictions are more precise than both the APC-corrected and non-APC-corrected $L_{2,1}$-regularized contact predictions.

Since APC estimates per-column background couplings from the pre-correction coupling matrices, a possible explanation for the unexpected decrease in predictive precision could be that the $L_{2,1}$ regularization shifts the mean background coupling in a way that is incompatible with APC. See Chapter 4 for a more detailed discussion of this effect.

3.4 Contacts-per-Position Prior

The Average Product Correction (APC) enforces the same average number of predictions (or average level of coupling) for each column or row in the predicted coupling matrix. Such a correction makes biological sense since only a limited number of other residues can interact with any given residue. Instead of enforcing this criterion as a post-correction step, however, it was investigated whether including a constraint on the number of contacts per position during the model learning process would improve prediction accuracy.

Using the bayesian framework of regularization terms as priors, a new term R_{CPP} (with new parameters $\lambda_{CPP}, \beta, \gamma$ to be optimized) was added that limits the number of contacts per position by using a smoothed step function to count the number of contacts:

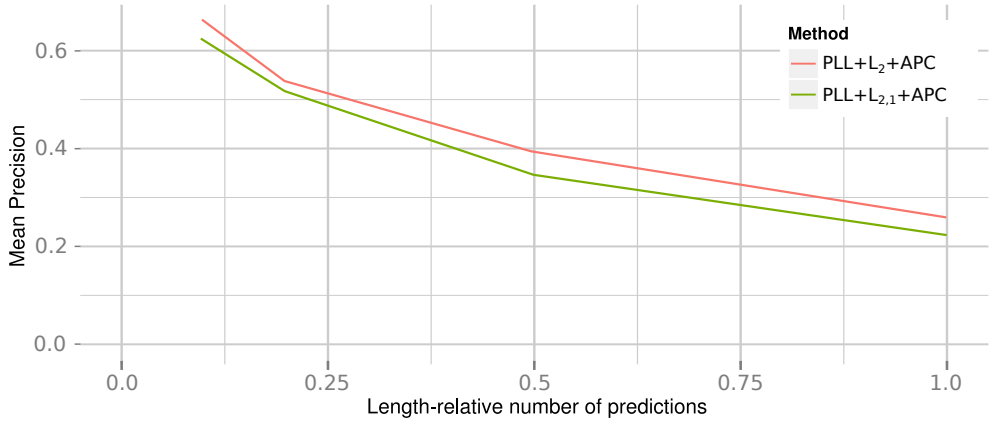


Figure 3.4: Evaluation of the $L_{2,1}$ regularization. After Average Product Correction, the grouped variable regularization performs systematically worse than the conventional L_2 regularization.

$$R_{CPP}(\boldsymbol{w}) = \lambda_{CPP} \sum_{i=1}^{L} \left(\sum_{j=1}^{L} \|w_{i,j}\|_2^{\gamma} \right)^{\frac{2\beta}{\gamma}} \tag{3.3}$$

The regularization term R_{CPP} is an $L_{p,q}$ norm [70] (with $p = 2, q = \gamma$) and works on two levels: First, all 20×20 amino acid pairs for a pair of residue positions (i, j) are grouped together into a L_2 term $\|w_{i,j}\|_2$. Next, all residues j of a column i are grouped together like in a group lasso $L_{2,1}$ term [71], but with individual summands exponentiated by $\gamma \geq 1$ so that high values are exaggerated and low values even closer to zero. In addition to the classic $L_{p,q}$ term, the R_{CPP} term allows exponentially scaling the regularization term by β for increased flexibility.

By modifying the λ_{CPP}, β and γ parameters, the strength of the regularization can be adjusted. Figure 3.5 shows the effect of varying λ_{CPP} on both stripe reduction and enforcing sparsity. Note that the Contacts-per-Position prior includes both the $L_{2,1}$ regularization and L_2 regularization as special cases: L_2 regularization corresponds to setting $\gamma = 2, \beta = 1$ and $L_{2,1}$ regularization corresponds to $\gamma = 1, \beta = \frac{1}{2}$.

For the protein family MSAs from the PSICOV data set, grid search (with $(\lambda_{CPP}, \beta, \gamma) \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}\} \times \{0.25, 0.5, 1, 2, 4\} \times \{0.25, 0.5, 1, 2, 4\}$) was used to predict residue-residue contacts using the R_{CPP} regularization term.

Since Equation3.3 is non-differentiable for $\|w_{i,j}\| = 0$, special care has to be taken when optimizing the model under regularization. A custom optimization method was developed for this purpose that will be discussed in Section 3.5.

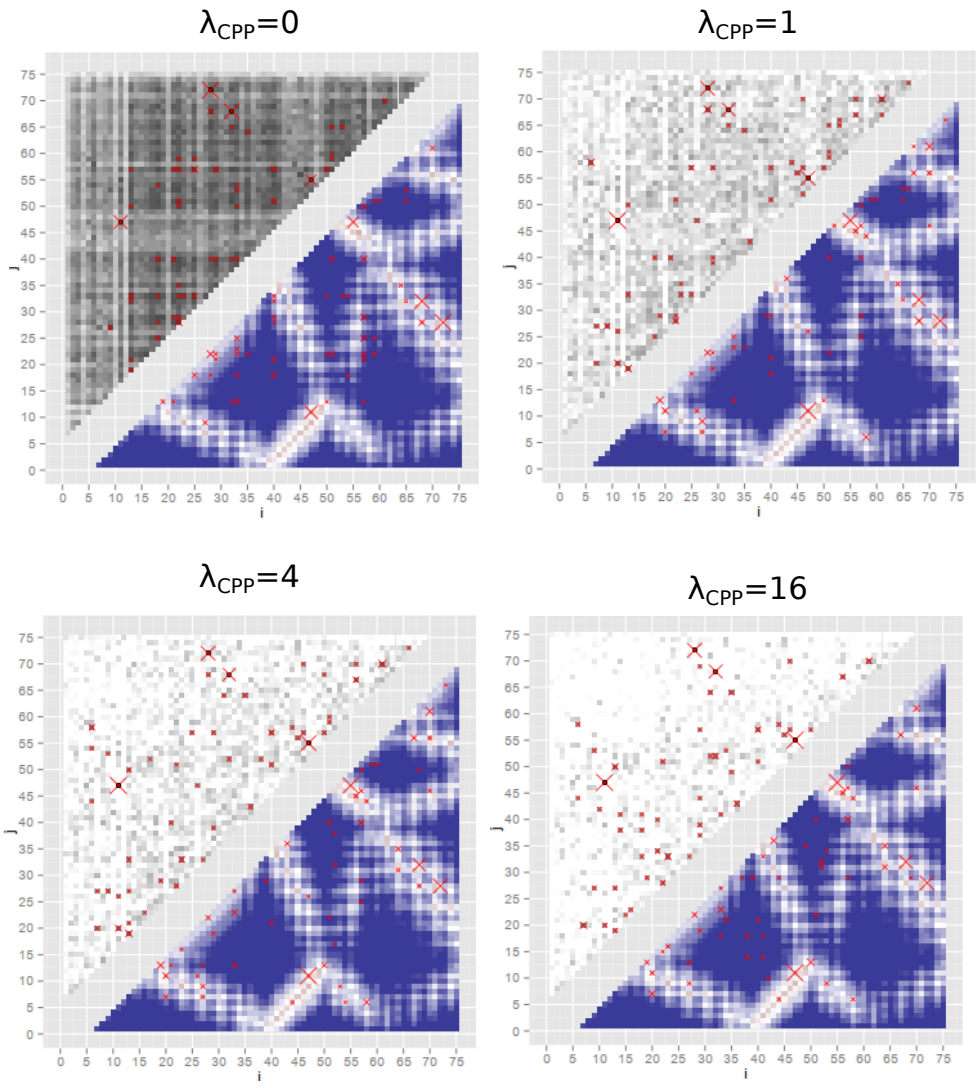


Figure 3.5: Effect of λ_{CPP} on the predicted contact maps. As the strength of the contacts-per-position prior increases, the striping pattern in the resultant contact maps is reduced but mean coupling also is more sharply centered around 0. The 50 most confident predictions are highlighted as red crosses. As the strength of regularization increases, the decrease in striping leads to an increase in precision.

3.4.1 Results for the Contacts-per-Position Prior

Before applying APC correction to the output matrices, contact predictions using a contact-per-position prior with $\gamma = 2, \beta = 2$ showed higher precision values than non-APC-corrected L_2 prior contact predictions ($\frac{L}{10}$ precisions of 46.0% vs. 40.8% on a non-redundant set of SCOP domain MSAs). Applying APC to the predictions further boosts performance both in the R_{CPP} and L_2 regularization cases, however, with the highest-precision R_{CPP} parametrization equalling an L_2 regularization. Figure 3.6 shows the results of the contacts-per-position prior evaluation after APC.

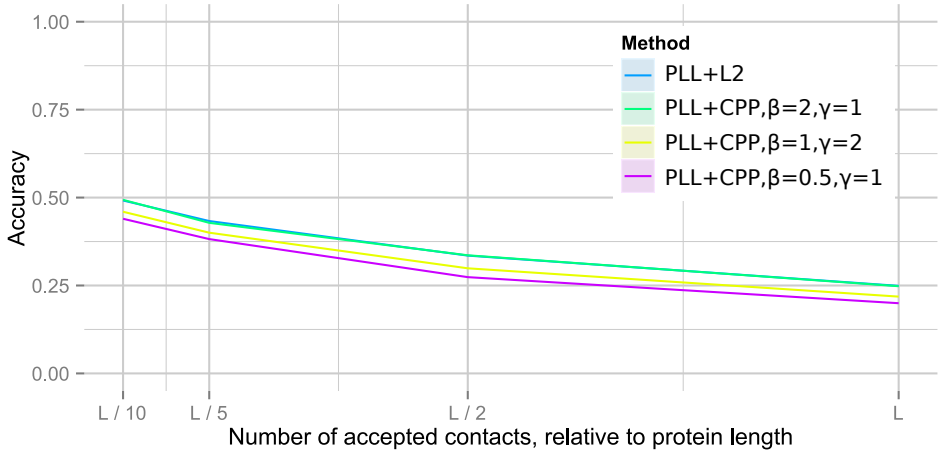


Figure 3.6: Evaluation of the Contacts-Per-Position Prior. The best-performing parametrization of the contacts-per-position prior is identical to an L_2 norm.

3.5 Optimizing with Non-Differentiable Points

For both the $L_{2,1}$ regularization discussed in Section 3.3 and the contacts-per-position prior discussed in Section 3.4, the regularization term is non-differentiable for cases where $\sum_{a,b=1}^{20} w_{i,j}(a,b)^2 = 0$ due to that term appearing in a denominator for the gradients. Since regularization gradients strongly pull parameters to zero for both regularization terms discussed here, it is therefore likely that a conventional optimization method will run in to a non-differentiable point, causing gradient calculation and optimization methods based on quadratic approximations to fail. To combat this optimization failure, a custom optimization strategy was developed based on the Orthantwise Quasi-Newton algorithm for L_1-regularized objective functions [72] and the implementation thereof contained in the `liblbfgs` library by Naoaki Okazaki.

While the Orthantwise Quasi-Newton algorithm only needs to deal with non-differentiable spots for an L_1 regularization that occur when any parameter is exactly zero, the grouped regularization optimizer has to handle cases where whole groups of variables become zero.

To do this, gradient evaluation and line search steps are split into blocks of grouped variables that can be evaluated and for which parameters can be updated individually. A consequence of this is that there is not a single step length derived for all parameters in the model per line search step but different variable groups can advance with different line search steps.

Consider a likelihood function $l(\vec{\Theta})$ under a regularization term $r(\vec{\Theta})$ for variables $\theta_{g,k}$ grouped into G groups of K_g variables each with regularization coefficient c_g. The grouped regularizer is defined as:

$$r(\vec{\Theta}) = \sum_{g=1}^{G} c_g \sqrt{\sum_{k=1}^{K_g} \theta_{g,k}^2} = \sum_{g=1}^{G} c_g \left\| \vec{\theta}_g \right\|_2 \tag{3.4}$$

The gradient for the regularization term can be shown to have a fixed magnitude c_g for each group of variables independent of the magnitude of variables in that group:

$$\frac{\partial r(\vec{\Theta})}{\partial \theta_{g,k}} = c_g \frac{\theta_{g,k}}{\left\| \vec{\theta}_g \right\|_2} \tag{3.5}$$

Gradient evaluation for the regularization gradient was directly implemented inside of the optimizer to handle the required special cases during optimization and a *pseudo-gradient* $\diamond_{g,k}$ of the likelihood function was introduced to handle non-differentiable spots in the regularization gradient:

$$\diamond_{g,k} = \begin{cases} \frac{\partial f(\vec{\Theta})}{\partial \theta_{g,k}}, & \vec{\theta}_g \neq 0, \\ \frac{\partial f(\vec{\Theta})}{\partial \theta_{g,k}} \times (1 - \frac{c_g}{\left\| \nabla_{\vec{\theta}_g} l(\vec{\Theta}) \right\|_2}), & \vec{\theta}_g = 0 \wedge \left\| \nabla_{\vec{\theta}_g} l(\vec{\Theta}) \right\|_2 > c_g, \\ 0, & \vec{\theta}_g = 0 \wedge \left\| \nabla_{\vec{\theta}_g} l(\vec{\Theta}) \right\|_2 \leq c_g, \end{cases} \tag{3.6}$$

A group of variables is considered to be "trapped" if all of the variables belonging to the group are currently zero (i.e. the group of variables is located in a non-differentiable spot). If a group of variables is trapped, the gradient for all variables will be forced to zero (case 3 of Equation 3.6) unless the gradient of the unregularized function is sufficiently strong to "escape" the trapped location with the magnitude of the regularization gradient for group g pulling towards zero with magnitude c_g (as shown in Equation 3.5) and the likelihood gradient pointing out of the zero point. A group of variables can therefore escape if the magnitude of its likelihood gradient is larger than the magnitude of the regularization gradient c_g (case 2). If the group is not currently trapped, the regularization term and gradients will be defined for that variable block and can be calculated to determine the search direction as normal (case 1).

Additionally, the line search strategy is modified to ensure that the optimization has a chance of hitting the non-differentiable spots even if a strong gradient passes by the point at zero: As illustrated in Figure 3.7b, for each of the variable blocks, a perpendicular is drawn from the zero point of that block to the search direction line. If line search results in a step length that crosses the foot of the perpendicular, the parameters for that variable block are only advanced until they hit the foot. For the next iteration, the gradient at the foot of the perpendicular can either point towards the zero point (indicating that there was not enough evidence for the grouped variables to be nonzero) or the optimization can continue normally, crossing the foot of the perpendicular.

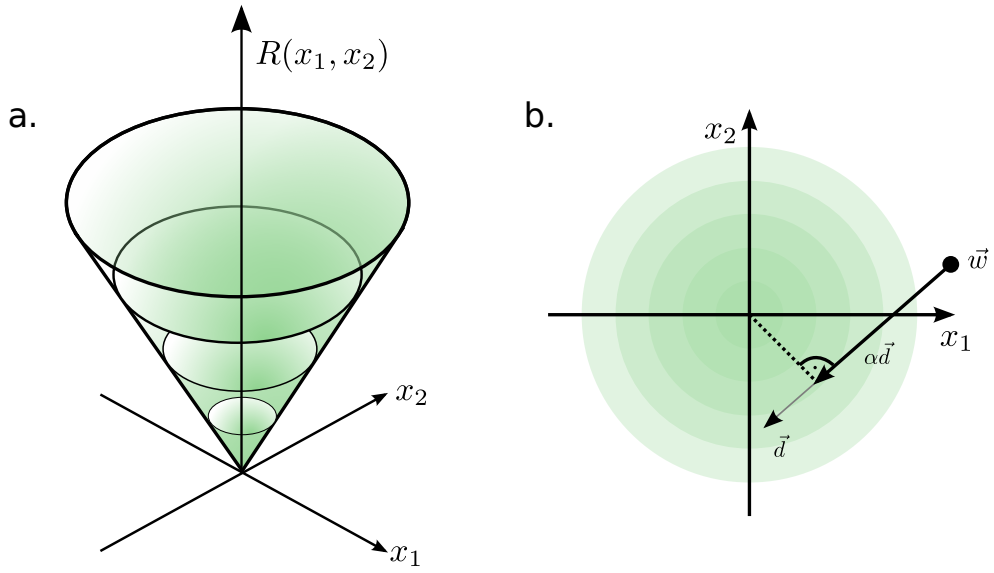


Figure 3.7: Minimization into non-differentiable points shown on a two-dimensional function $R(x_1, x_2) = \sqrt{x_1^2 + x_2^2}$ with a non-differentiable point at $(0, 0)$ **a.** Perspective view of R showing the overall shape of the function **b.** Contour plot of R showing the early termination of line search. If line search for a variable block is crossing over the foot of the perpendicular of the zero point of the variable block to the line search direction, variables are only updated until the foot of the perpendicular is reached. Subsequent iterations can allow the optimization to converge into the zero point or move away, depending on the gradient at that point.

3.6 Improved Sequence Weighting

To reduce the effect of redundant sequences in the multiple sequence alignments for contact prediction, redundant sequences are typically removed using an identity threshold and then re-weighted to reduce the influence of remaining redundancy. Usually, a simple weighting strategy is used that clusters sequences according to an identity threshold and then splits a weight between all sequences in the cluster:

$$w_n^S = \frac{1}{c_n},\ c_n = \sum_{m=1}^{N} I(\mathrm{ID}(n,m) \geq 90\%) \qquad \mathrm{ID}(n,m) = \frac{1}{L}\sum_{i=1}^{L} I(x_i^n = x_i^m) \tag{3.7}$$

3.6.1 Alternative Weighting Schemes

Henikoff weights [73] use a per-column diversity measure and have been applied successfully in many sequence analysis methods (see e.g. [74]). Each sequence position is inversely scored depending on how many unique amino acids occur in the same column and how often the current amino acid also appears in other sequences:

$$w_n^H = \sum_{i=1}^{L} \frac{1}{c_i N_i(x_i^n)},\ c_i = |\{x_i^n | n \in \{1 \ldots N\}\}| \tag{3.8}$$

$$N_i(a) = \sum_{n=1}^{N} I(x_i^n = a)$$

Since contact prediction primarily deals with pairwise interactions, the Henikoff weighting scheme was extended to use a pairwise diversity measure in simple analogy to the single-column measure:

$$w_n^{H2} = \sum_{\substack{i,j=1 \\ i \neq j}}^{L} \frac{1}{c_{i,j} N_{i,j}(x_i^n, x_j^n)},\ c_{i,j} = \left|\{(x_i^n, x_j^n) | n \in \{1 \ldots N\}\}\right| \tag{3.9}$$

$$\tag{3.10}$$

3.6.2 Sequence Weighting Results

The sequence weighting methods were evaluated for their usefulness in accurate contact prediction using the alignments of the PSICOV dataset. Figure 3.8 shows that the pairwise Henikoff weights give an increase of 1.8 percent points in $\frac{L}{10}$ precision compared to the commonly used simple weights, or 2.85 percent points compared to using no weighting. The only slight improvements compared to not using weights can be explained by the fact that input alignments were constructed to minimize redundancy. Still, the results show that introducing suitable sequence weighting methods into the contact prediction procedure can make it resistant to sampling biases introduced by redundant input alignments.

3.7 Learning Triplet Interactions

The residue-residue contact prediction methods published so far specialize on predicting two-residue interactions as they can be simply understood as the presence or absence of

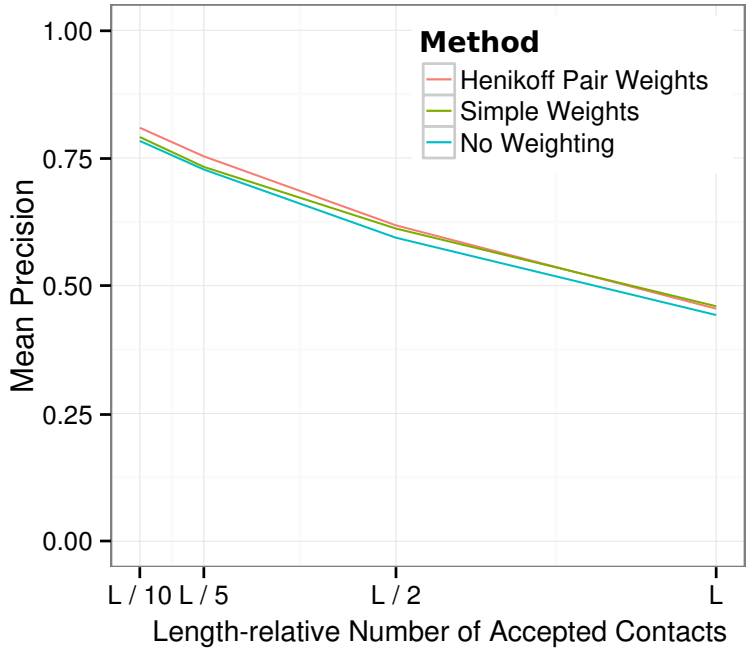


Figure 3.8: Alternative Sequence Weighting Evaluation. On the PSICOV data set of protein domain MSAs, Henikoff pair weights achieve $\frac{L}{10}$ precisions of 80.7% compared to the $\frac{L}{10}$ precisions of the simple weighting scheme of 78.9% and 77.8% of not using weights.

edges in a residue interaction graph. For the highly position-specific interactions of salt bridges or disulfide bonds this is a reasonable modeling approach, but there is evidence that especially the hydrophobic residues in the core of a protein form different, more unspecific interaction patterns so that a covariance signal might be weaker but present in many residue positions [75].

Since hydrophobic interactions in the core of a protein fold are especially important for maintaining the structure, a reasonable extension for residue-residue contact prediction is to extend the MRF likelihood function by terms to account for several-residue interactions, with the simplest extension being a triplet interaction term.

The following section will discuss a heuristic approach for extending the MRF model to include the triplet terms most likely to be true triplet interactions and thus increase model expressiveness.

3.7.1 Triplet Selection

If all possible triplet scores were included in the MRF model likelihood, the algorithmic complexity of the model would necessarily rise from $O(NL^2 20^2)$ for considering pairwise interactions to $O(NL^3 20^3)$ for triplet interactions. Since only very few triplets are expected to display sufficient covariance to be detected by the MRF model in relation to the many possible triplets, it should be sufficient to heuristically pick a list of T triplets and only optimize triplet coupling potentials for these pre-picked triplets, reducing algorithmic

complexity to $O(NL^2 + T)$ and making optimization feasible:

$$\mathrm{pll}(X|\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) = \sum_{n=1}^{N}\left(\sum_{i=1}^{L}\frac{1}{Z_i^n}\left[u_i(x_i^n) + \sum_{\substack{j=1\\j\neq i}}^{L} v_{i,j}(x_i^n, x_j^n) + \sum_{(i',j',k')\in T_i} w_{i',j',k'}(x_{i'}^n, x_{j'}^n, x_{k'}^n)\right]\right) \tag{3.11}$$

Since triplet interactions require a covariance signal to be detected, it is expected that the residues making up a triplet interaction will also show a high degree of pairwise covariation. Since pairwise interactions can be computed quickly, pairwise couplings are first estimated for all pairs in the protein domain and then used to select candidate triplets.

Different heuristics of triplet selection were evaluated independently since it was unclear *a priori* which would perform the best. First, two hypotheses of how triplet interaction candidates are detected seemed plausible. For a given combination of three positions (i, j, k), there could either be strong evidence only in one correspondent combination of amino acids (a, b, c) or several different amino acid combinations contributing partial signals. If specific amino acid combinations contribute to the triplet signal, the correct strategy would be to select triplets T as sextuplets (i, j, k, a, b, c) where for the case of partial contributions, triplets (i, j, k) should be selected and all amino acids at that position combination should be considered. For this reason, both sextuplet ranking score S_6 and triplet ranking score S_3 from pairwise coupling signals $v'_{i,j}(a, b)$ were defined:

$$S_6(i,j,k,a,b,c) = v'_{i,j}(a,b) + v'_{j,k}(b,c) + v'_{k,i}(c,a) \tag{3.12}$$

$$S_3(i,j,k) = \sum_{a,b,c=1}^{20} v'_{i,j}(a,b) + v'_{j,k}(b,c) + v'_{k,i}(c,a) \tag{3.13}$$

Furthermore, it is known from pairwise coupling predictions that both positive and negative coupling scores are indicative of spatial proximity (with positive couplings corresponding to positive selection for an interaction and negative couplings corresponding to negative selection of that interaction). In order to use the maximum amount of information, all non-zero interactions should consequently be considered for ranking triplet candidates. Again, several coupling signal transformations were chosen:

$$v'^{1}_{i,j}(a,b) = |v_{i,j}(a,b)| \tag{3.14}$$

$$v'^{2}_{i,j}(a,b) = (v_{i,j}(a,b))^2 \tag{3.15}$$

With $v_{i,j}(a, b)$ being the untransformed raw coupling potentials. All combinations of coupling transformations and tuple ranking scores were used to select triplets T and will be compared in the results section.

The parallelized fast C code for learning pseudo-likelihood MRF models was extended to additionally optimize triplet interaction terms from a user-provided list of triplets.

3.7.2 Detecting Interacting Triplets

The triplet couplings learned by the maximum-pseudo-likelihood estimator were evaluated for recognizing spatially proximal three-residue interactions by using it for ranking $\{i, j, k\}$ residue sets using the triplet coupling values. For each of the residue combinations, either the most highly coupled $TS(i,j,k)^{max} = \max_{a,b,c} w'_{i,j,k}(a,b,c)$ value ("single evidence") in the set of selected triplets T was used to rank the residue combination, or evidence for all couplings were summed $TS(i,j,k)^{\Sigma} = \sum_{a,b,c=1}^{20} w'_{i,j,k}$, with the same coupling transformations for $v'_{i,j}(a,b)$ discussed above applied to the triplet couplings $w'_{i,j,k}(a,b)$.

Figure 3.9 compares the triplet couplings after pseudo-likelihood optimization with simply using the S_6 score discussed above for ranking triplet interactions. The S_6 score appears to be already highly informative in picking spatially close triplet pairs with the triplet scores after optimization performing slightly worse than the pre-optimization S_6 scores in high-diversity multiple sequence alignments. In the low-diversity case, all evaluated methods are equally bad at detecting spatially close triplet interactions with mean pairwise distances $> 12\mathring{A}$. While triplet couplings are not informative by themselves, combining them into a model that also takes pairwise interactions into account might still help improve predictive performance.

3.7.3 Combining Pair and Triplet Interactions

Pseudo-likelihood maximization methods produce coupling information for both pairwise and triplet terms. While only a small number of triplet interactions are expected to exist in a protein MSA and it is expected that only a small fraction will be detectable from a coevolution-based method, it is also expected that these detected triplets will show high specificity. A reasonable idea is therefore to combine the information of triplet couplings with the pairwise couplings to predict pairwise contact probabilities in a bayesian framework using bayes factors (a bayesian interpretation of likelihood-ratio tests):

$$BF(i,j) = \frac{P(C_{i,j}=1|\|w_{i,j,k}\|_\alpha, \|v_{i,j}\|_2, T)}{P(C_{i,j}=0|\|w_{i,j,k}\|_\alpha, \|v_{i,j}\|_2, T)} = \tag{3.16}$$

$$= \frac{P(\|w_{i,j,k}\|_\alpha, \|v_{i,j}\|_2|C_{i,j}=1,T)}{P(\|w_{i,j,k}\|_\alpha, \|v_{i,j}\|_2|C_{i,j}=0,T)} \frac{P(C_{i,j}=1)}{P(C_{i,j}=0)} \approx \tag{3.17}$$

$$\approx \frac{P(\|w_{i,j,k}\|_\alpha|C_{i,j}=1,T)}{P(\|w_{i,j,k}\|_\alpha|C_{i,j}=0,T)} \frac{P(\|v_{i,j}\|_2|C_{i,j}=1,T)}{P(\|v_{i,j}\|_2|C_{i,j}=0,T)} \frac{P(C_{i,j}=1)}{P(C_{i,j}=0)} \tag{3.18}$$

The bayes factor can be decomposed into terms involving the triplet couplings (first fraction) and pairwise couplings (second fraction) plus a prior (third fraction). These one-dimensional probability distributions can be fit separately for $C_{i,j} = 1$ (i.e. they are part of a physically contacting residue pair) and $C_{i,j} = 0$ using mixture density models.

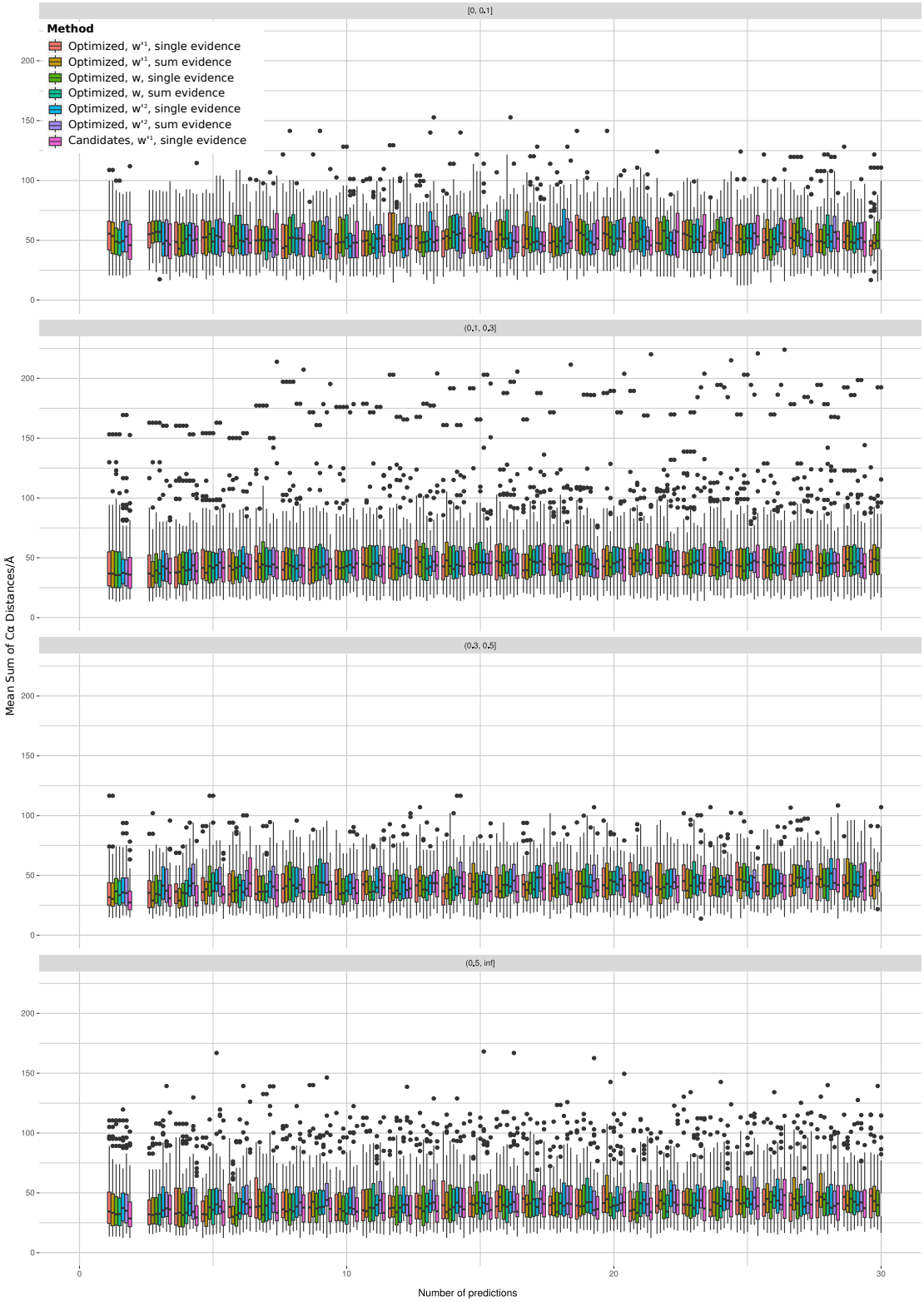


Figure 3.9: Triplet Couplings for Tri-Residue Contact Prediction. For different triplet scoring methods, the mean sum of residue-residue Cα distances $d_{ij} + d_{jk} + d_{ki}$ is plotted against the ranking of the corresponding triplet prediction.

Triplet coupling scores were fit as the mixture of three exponential distributions, with mixing weights π_d:

$$P(\|w_{ijk}\|_\alpha = x|C_{ij}, T) = \sum_{d=1}^{3} \pi_d \lambda_d \exp\left[-\lambda_d x\right], \ \sum_{d=1} \pi_d = 1 \tag{3.19}$$

Figure 3.10 shows the binned triplet score counts for contacting and non-contacting triplets together with their fits and the corresponding bayes factor term. As the triplet coupling score increases, the bayes factor increases correspondingly. The dip for coupling scores between 0.05 and 0.17 can be explained by the limited-data learning scenario of triplet interactions.

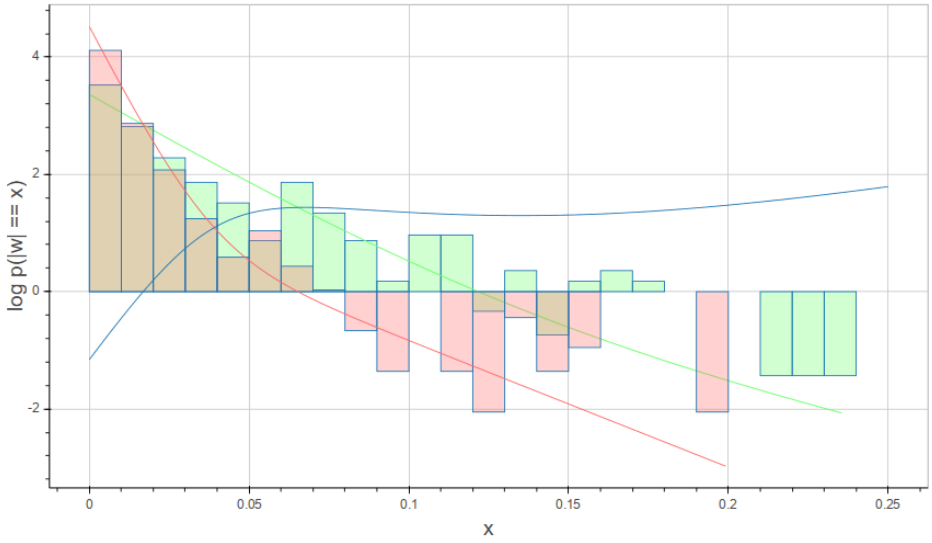


Figure 3.10: Fitting of Triplet Score Distributions of Couplings Learned from Non-Redundant Protein Domain MSAs. The bars represent log-counts for triplet scores where triplets involve contacting residues (green) and triplets that are spatially distant (red). The lines show the fits of Equation 3.19 and the blue line the Bayes Factor component.

Pairwise coupling scores were fit as the mixture of three exponential distributions for the contacting case and as a mixture of an exponential distribution and a log-normal distribution for the non-contacting case:

$$P(\|v_{ij}\|_2 = x|C_{ij}, T) = \begin{cases} \sum_{d=1}^{3} \pi_d \lambda_d \exp\left[-\lambda_d x\right], & C_{ij} = 0 \\ \pi_4 \lambda_4 \exp\left[-\lambda_4 x\right] + \pi_5 \frac{1}{x\sigma\sqrt{2\pi}} \exp\left[-\frac{(\ln x - \mu)^2}{2\sigma^2}\right], & C_{ij} = 1 \end{cases} \tag{3.20}$$

Figure 3.11 shows the binned pair score counts for contacting and non-contacting residues together with the fitted distribution and the corresponding bayes factor term. Again, the bayes factor increases as the pairwise coupling score increases.

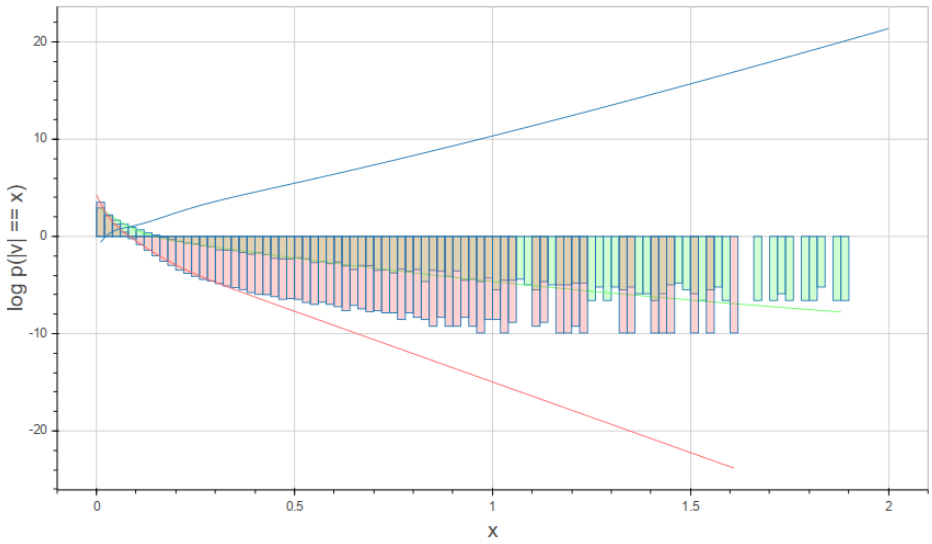


Figure 3.11: Fitting of Pair Score Distributions of Couplings Learned from Non-Redundant Protein Domain MSAs. The bars represent log-counts for summed pair scores of contacting residues (green) and non-contacting residues (red). Fits of Equation 3.20 are shown as lines with the Bayes Factor shown in blue.

By combining the probability fits with the prior for observing a contacting residue pair (third fraction of the bayes factor), the bayes factor can be used to calculate posterior probabilities for a pair of residues to be interacting given the triplet and pair coupling information:

$$P(C_{i,j} = 1 | \|w_{i,j,k}\|_\alpha, \|v_{i,j}\|_2, T) = \frac{1}{1 + BF(i,j)} \tag{3.21}$$

A similar bayes factor was also developed to calculate posterior values for the only-pairwise situation of traditional pseudo-likelihood models and will be compared with triplet interaction posteriors and traditional APC-corrected summed score matrices.

3.7.4 Triplet Couplings for Pairwise Constraints

Figure 3.12 shows the prediction performance of triplet contact prediction methods to enhance pairwise contact prediction on real-world protein domain MSA data. Since the posterior probability estimate using only pairwise terms uses the same information as the conventional L_2 norm sum score, it is not surprising that pair posterior and pseudo-likelihood methods exhibit identical performance. Unfortunately, the triplet posterior term shows inferior prediction performance compared to all other methods with even simply using the pairwise coupling potentials of a pseudo-likelihood model that also includes triplet

terms being more informative than computing a posterior probability. It can therefore be concluded that the additional coupling scores for triplet interactions cannot help a more accurate pairwise prediction.

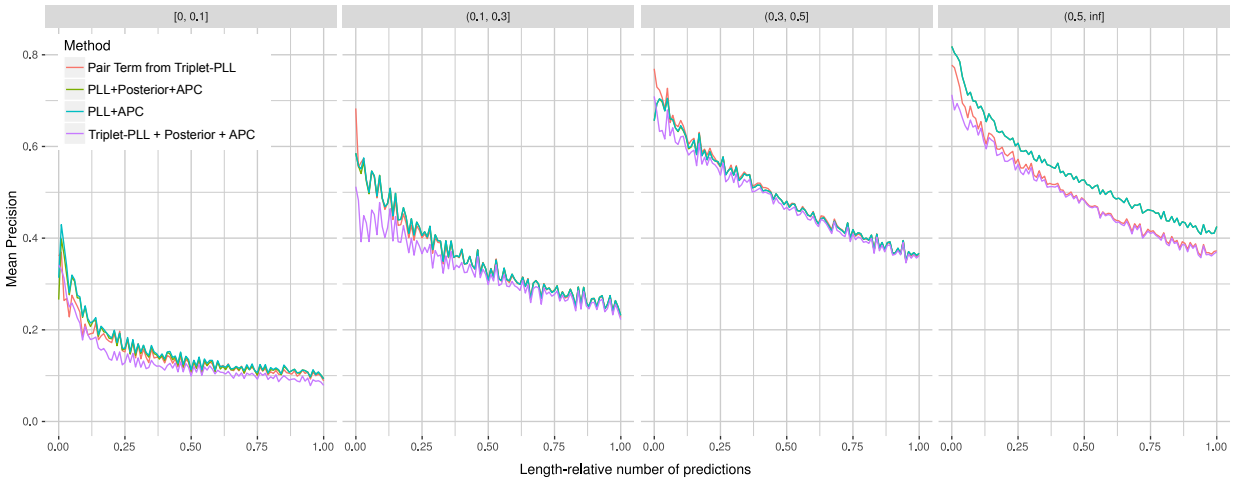


Figure 3.12: Results of Triplet Evaluation for Pairwise Constraints. Using the posterior formulation of Equation 3.21, the utility of triplet coupling potentials for detecting pairwise residue-residue interactions is examined for four different MSA diversity classes. In all cases, triplet posterior scores perform the worst at predicting residue-residue contacts, with the difference in performance especially high for alignments with many homologs.

3.8 Full Likelihood Approaches

While it has been shown that the pseudo-likelihood estimate is consistent with the full likelihood estimate for large amounts of data, there is no clear indication whether pseudo-likelihood performs as well as full likelihood in the low-data scenarios typical of contact prediction. While the partition function of the full likelihood cannot typically be computed due to its exponential complexity, it is possible to use monte carlo methods to get an approximate gradient for optimization by drawing protein sequences according to the current estimate of the probability distribution.

This section will discuss two sampling-based methods to approximate the full likelihood gradients and then go a step further by including phylogenetic interdependencies in the sampling process to integrate phylogenetic noise correction into a full likelihood framework.

3.8.1 Intractability of Partition Function

As mentioned in the introduction to this part, the partition function for a markov random field is:

$$Z = \sum_{x' \in \{1...20\}^L} p(x'|\boldsymbol{v}, \boldsymbol{w}) \tag{3.22}$$

With $p(x'|\boldsymbol{v}, \boldsymbol{w})$ an unnormalized probability term. Since the sum goes over all possible amino acid sequences of length L, this means that 20^L terms would need to be summed up to compute the partition function exactly. For all realistic protein lengths, however, this impossible — the most powerful supercomputer at the time of writing (Tianhe-2 can perform 33.86×10^{15} floating-point operations per second) would take a little more than 5 years to compute this sum even for a 19-residue protein assuming that each summand only requires one floating point operation:

$$\frac{20^{19}}{33.86 \times 10^{15} \times 60 \times 60 \times 24 \times 365} \approx 5.038$$

As the full likelihood function cannot be computed exhaustively, a strategy often used in bayesian inference is to instead employ sampling methods. The following sections will discuss how sampling was applied to the MRF model to optimize the full likelihood function.

3.8.2 Hybrid Monte Carlo Method

The Markov Chain Monte Carlo (MCMC) class of algorithms provide the facilities to generate samples from a probability distribution by generating a markov chain of samples whose equilibrium distribution is equal to the underlying probability distribution. By relying on the sampling approach, the computationally expensive partition function is

no longer needed, making optimization of the full likelihood feasible and sample mean and variance of the equilibrium distribution can be used for quadratic approximations in the optimum of the MRF full likelihood. Since the sample mean of the markov chain is equivalent to the sample mean of the full likelihood MRF probability, these mean values are expected to accurately recover residue-residue coupling. Section 12.2 gives a more detailed overview over important Monte Carlo algorithms.

Based on the fast GPU implementation of MRF gradients discussed in Chapter 7, the Hybrid Monte Carlo algorithm explained in Section 12.2.1 was implemented as a GPU code to sample from the full likelihood MRF probability. While it is possible to determine unnormalized likelihood gradients and unnormalized likelihood values, the differences in gradient magnitudes frequently leads to cases where the generated candidates are frequently rejected and HMC parameters need to be manually adjusted for each new protein under study. Since another interpretation of the likelihood gradient more suitable for optimization was found in contrastive divergence, the HMC approach was abandoned.

3.8.3 Reinterpreting the Full Likelihood Gradient

Alternatively to the HMC strategy of sampling from an unnormalized probability distribution, another approach is to sample sequences from the current model probability distribution and use these samples to calculate full likelihood gradients. The gradients of the Markov Random Field Likelihood can be rearranged into the following forms (see Appendix A for the full derivation):

$$\frac{\partial L(X|\boldsymbol{v},\boldsymbol{w})}{\partial w_{i,j}(a,b)} = N_{i,j}(a,b) - NP(x_i = a, x_j = b|\boldsymbol{v},\boldsymbol{w}) \tag{3.23}$$

$$\frac{\partial L(X|\boldsymbol{v},\boldsymbol{w})}{\partial v_i(a)} = N_i(a) - NP(x_i = a|\boldsymbol{v},\boldsymbol{w}) \tag{3.24}$$

with $N_{i,j}(a,b)$ and $N_i(a)$ being pairwise and single amino acid counts that are fixed for a given input MSA, respectively. The probability terms correspond to the marginals of the full MRF probability distribution that would contain the partition function term Z to ensure the probability distribution is correctly normalized.

Since the marginal probability distributions $P(x_i = a, x_j = b|\boldsymbol{v},\boldsymbol{w})$ and $P(x_i = a|\boldsymbol{v},\boldsymbol{w})$ consist of a term that is constant for a given input alignment and a term that depends on the expected number of amino acids and amino acid pairs according to the model probability, the expected counts can also be estimated by pairwise and single frequencies of a sampled data set drawn from the correct model distribution.

By the use of Gibbs sampling, unnormalized probability terms of a candidate sequence can be compared to a baseline sequence and the candidate sequence accepted or rejected without the need of normalizing the probability distributions. If the Gibbs sampling process is repeated, a set of new sequences can be drawn independently from the model distribution to form a new multiple sequence alignment that can be used as empirical probabilities in the above equations.

The next sections will show how Gibbs sampling can be used to optimize the full likelihood from sampled gradients. Chapter 13 will show how the marginal probability distributions was used to sample synthetic multiple sequence alignments for sequence analysis method debugging and protein engineering applications.

3.8.4 Contrastive Divergence

While originally only applied to training products of expert models, the *Contrastive Divergence* (CD) method [76] can applied for maximizing general log-likelihoods without the expensive computation of a normalization term. Instead, Gibbs sampling is used to generate new samples from the input data by evolving just one step of a markov chain. Hinton showed empirically that by re-evolving the markov chain from the input data for each gradient evaluation, gradients can be computed that will point towards the true parameters whereas if the parameter optimum is reached, the mean direction of sampled gradients will zero out so that the model converges after sufficient numbers of iterations.

The MRF probability can be rearranged into the following conditional probability for observing an amino acid a at position i given the model parameters and all other sequence positions as context:

$$\log P(x_i = a|\vec{v}, \vec{w}, (x_{1\ldots L\setminus\{x_i\}})) \propto v_i(a) + \sum_{\substack{j=1 \\ j\neq i}}^{L} w_{i,j}(a, x_j) \tag{3.25}$$

This conditional probability can be used in a Gibbs sampler to evolve new sequences: A random sequence position is picked with uniform probability and the amino acid at that position is replaced by a randomly selected amino acid using the conditional probabilities in Equation 3.25. This process is repeated for all sequences in the input alignment to derive a new sampled alignment. By alternating between parameter updates using the sampled gradients and the sampling of new MSAs, the maximum-likelihood solution for the MRF parameters can be reached.

The *Persistent Contrastive Divergence* (PCD) algorithm [77] is a variation on the CD algorithm: Instead of restarting the MSA sampling procedure from the original input MSA in every iteration, the markov chains are never restarted and are just updated using new model parameters in every iterations. On several data sets, Tieleman et al. showed convincingly that PCD shows superior convergence properties compared to CD.

PCD was implemented for the full likelihood optimization of contact prediction MRFs with the results shown in Figure 3.13 — both APC-corrected and non-APC-corrected predictions perform worse than the corresponding pseudo-likelihood predictions.

To elucidate whether the discrepancy in prediction score arises from sampling errors, synthetic protein MSAs generated from the methods outlined in Chapter 13 using coupling potentials learned from real-world protein MSAs were used to compare the real-world coupling potentials to coupling potentials recovered during sampling. Figure 3.14 shows that the recovered coupling potentials for both PLL and PCD methods are centered more

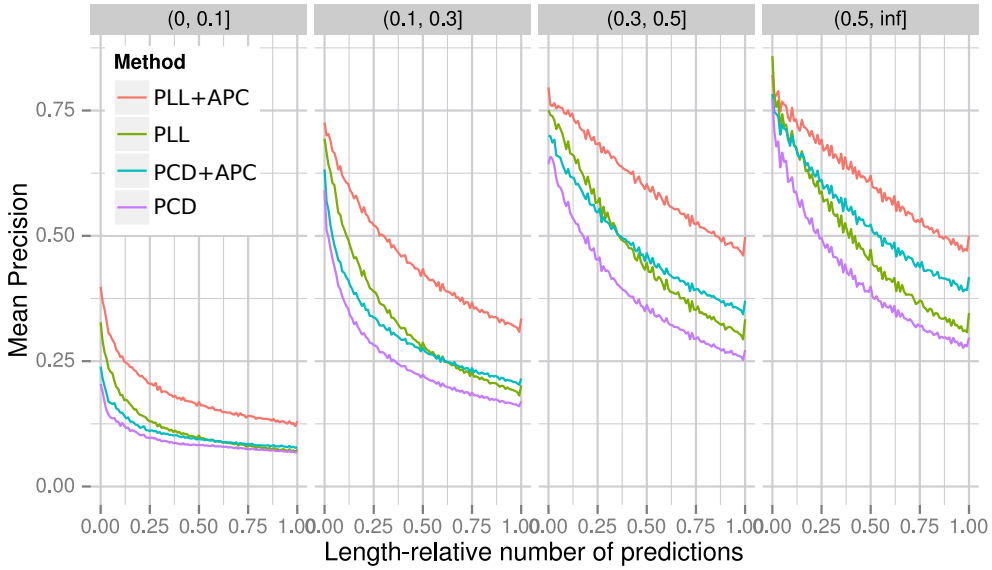


Figure 3.13: Results of Persistent Contrastive Divergence Evaluation. Contrastive divergence-based methods perform worse than pseudo-likelihood approaches.

around zero than what is described by true coupling parameters, a tendency that persisted for various choices of regularization parameters λ_{pair}.

Since a centering on zero should not necessarily have a negative effect on the ranking of residue-residue interactions, a second evaluation was performed to compare the root-mean-square-difference of score ranks determined by PLL and PCD methods from the true ranks of the parameters the synthetic MSAs were generated from. As seen in Figure 3.15, different choices of target function and regularization coefficient mostly affect pairwise coupling potentials $w_{i,j}$ while single emission potentials $v_i(a)$ and summed-score matrices $\|\vec{w}_{ij}\|$ show little difference in the distribution of rank differences. The discrepancy on real-world protein MSAs must therefore result from variance in the covariance patterns introduced by sampling protein MSAs since both PCD on real-world MSAs and PLL on sampled MSAs have included at least one step of sampling to arrive at coupling parameters. PCD could not outperform a sampling-free pseudo-likelihood method since the noise introduced through sampling outweighed the possible reduction of statistical noise through a full-likelihood treatment as compared to a pseudo-likelihood method.

3.8.5 Phylogenetic Contrastive Divergence

Since Contrastive Divergence requires the sampling of a protein MSA in every iteration of the maximum-likelihood optimization, an extension to the sampling strategy is to include phylogenetic interdependencies into the sampling process by evolving markov chains according to a reconstructed phylogeny instead of generating independent markov chains. By including the same phylogenetic interdependencies in the sampling as would be expected in the input MSA, the phylogenetic interdependencies no longer need to be expressed us-

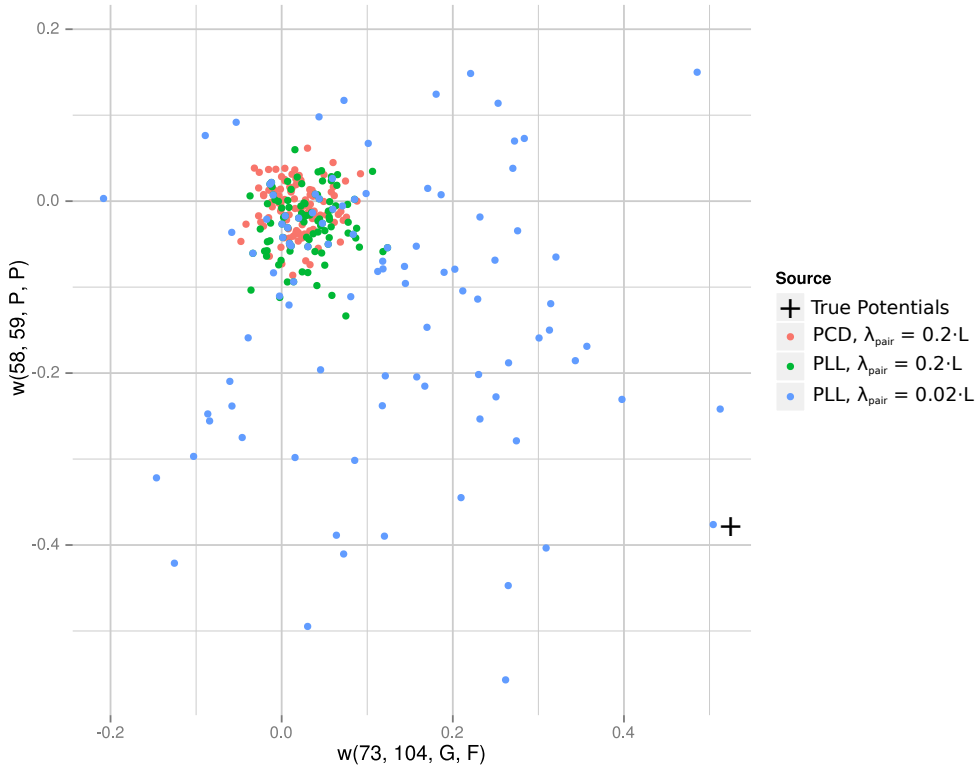


Figure 3.14: Recovery of True Coupling Potentials using PCD and PLL. Using coupling potentials derived from real-world protein alignments, synthetic MSAs were derived for which coupling parameters were recovered using pseudo-likelihood and persistent contrastive divergence optimizers. The scatterplot shows a 2D slice out of all coupling parameters with the dimensions selected to be the two largest-magnitude dimensions for the true coupling potentials.

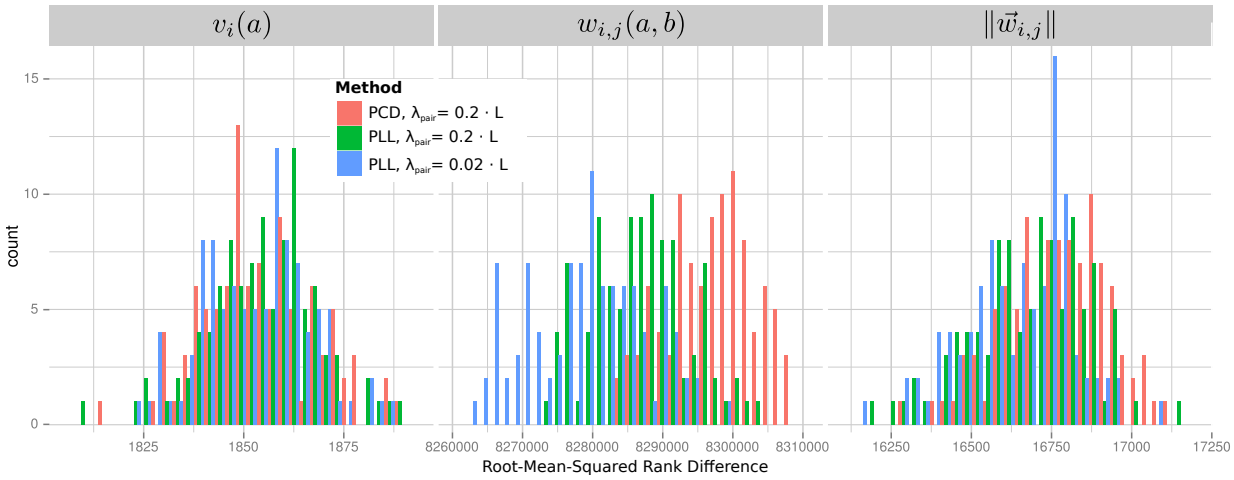


Figure 3.15: Recovery of Coupling Ranks using PCD and PLL. For a set of protein MSAs synthetically generated from known coupling parameters, the difference in the $v_i(a)$ potential ranks, $w_{ij}(a,b)$ potential ranks and $\|\vec{w}_{ij}\|$ sum score ranks was compared to the true ranks observed in the known coupling parameters with a histogram of root-mean-square-deviations of ranks plotted. Only for $w_{i,j}(a,b)$ values, a difference of ranks can be observed for the different coupling recovery methods and regularization parameters.

ing the pairwise coupling terms, increasing contact prediction accuracy. While persistent contrastive divergence showed sub-par prediction accuracy to simpler pseudo-likelihood models, the explicit treatment of phylogenetic noise could outweigh the newly introduced statistical noise and lead to an overall improvement.

Phylogeny was reconstructed using the FastTree approximate maximum-likelihood phylogenetic tree reconstruction method [65] and ancestral sequences were reconstructed using the CodeML method in the PHYLIP toolkit [78].

As seen in Figure 3.16, the coupling values produced by the phylogenetic contrastive divergence methods were significantly worse at predicting residue-residue contacts than approaches not including phylogenetic sampling. Since ancestral sequence and tree topology constrain the sampling of the empirical gradients, it is unclear whether the sampled multiple sequence alignments cover the sequence space sufficiently for the stochastic optimization to converge towards an optimum. More work would have to go into investigating whether the generated MSAs are comparable to the data distribution although initial investigations show that per-column amino acid frequencies converge to comparable values as in the input MSA.

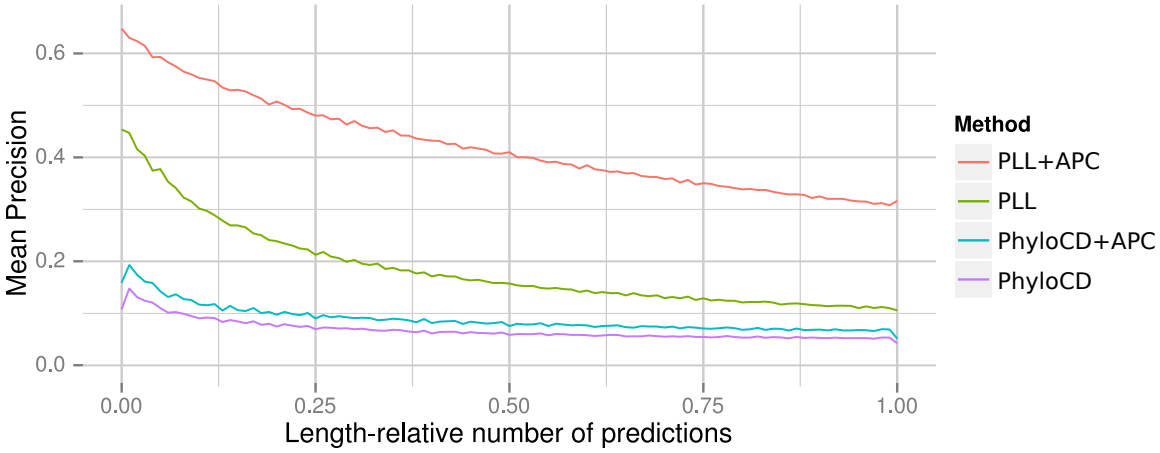


Figure 3.16: Results of Phylogenetic Contrastive Divergence Evaluation. In its current state, phylogenetic contrastive divergence is unable to recover the true coupling potentials detectable from a MSA.

Chapter 4

Replacing the Average Product Correction

For many of the approaches discussed in the previous chapter, an initial improvement of contact prediction precision could be achieved when comparing a baseline MRF model without APC against the improved method without APC, but the baseline model with APC would outperform the improved model without APC and applying APC to the improved model would reduce its performance. The Average Product Correction can therefore be seen as a barrier to further improvement in contact prediction methods and needs to be understood in more depth. This chapter summarizes the discoveries made in disentangling APC and details an attempt at replacing it with a more robust correction.

4.1 Underlying Assumptions of APC

Since the attempts at improving contact prediction outlined in the previous chapter were all turning out to be unsuccessful, experiments to check the sanity of the different components of contact prediction were made of which one experiment yielded unexpected results.

A *cheating regularization* term consisting of an L_2 regularization with pair-dependent $\lambda_p(i,j)$ was introduced to help contact prediction by masking out all pairs (i,j) from the predicted contact map whose physical $C\beta/C\alpha$ distance is larger than the median distances measured for the protein family under consideration as shown in Figure 4.1. By giving distant pairs a high regularization factor $C \times \lambda_p(i,j)$ (with $C \in \{1,2,4,8\}$) and a low regularization factor $\lambda_p(i,j)$ to other pairs, it should become much less likely to make completely false predictions and precision would be expected to go up. Surprisingly, as shown in Figure 4.2, increasing the masking factor C has the opposite effect on precision once APC is applied.

Looking back at the definition of the APC in Section 2.2.1, the two major underlying assumptions are that the total coupling observed in a predicted contact matrix is the sum of structural and functional coupling terms plus a background term controlled by entropic and phylogenetic effects, and that the background term can be estimated by computing

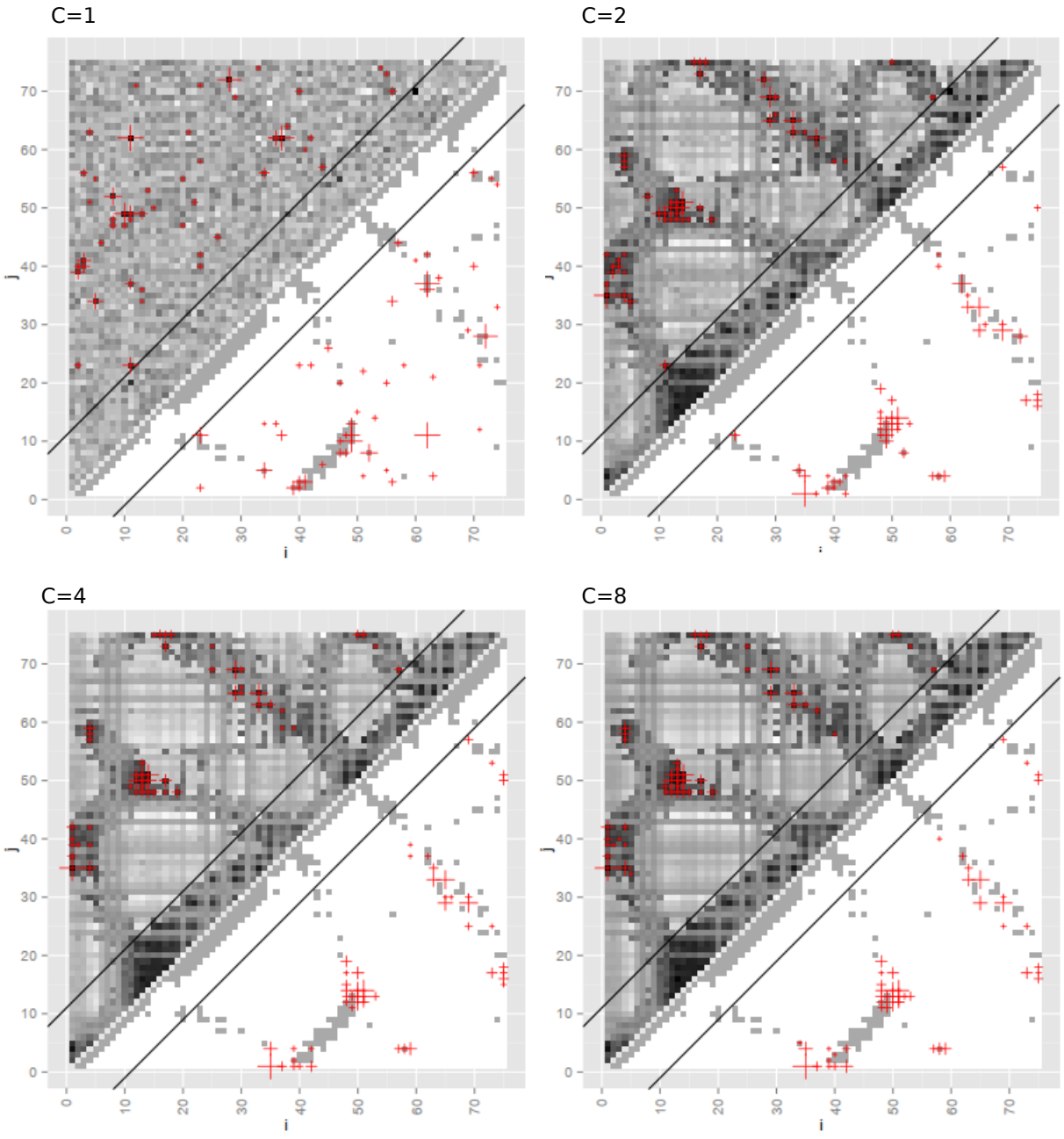


Figure 4.1: Contact prediction matrices under cheating regularization. As the masking factor is increased, the distant residue pairs are more strongly disfavored.

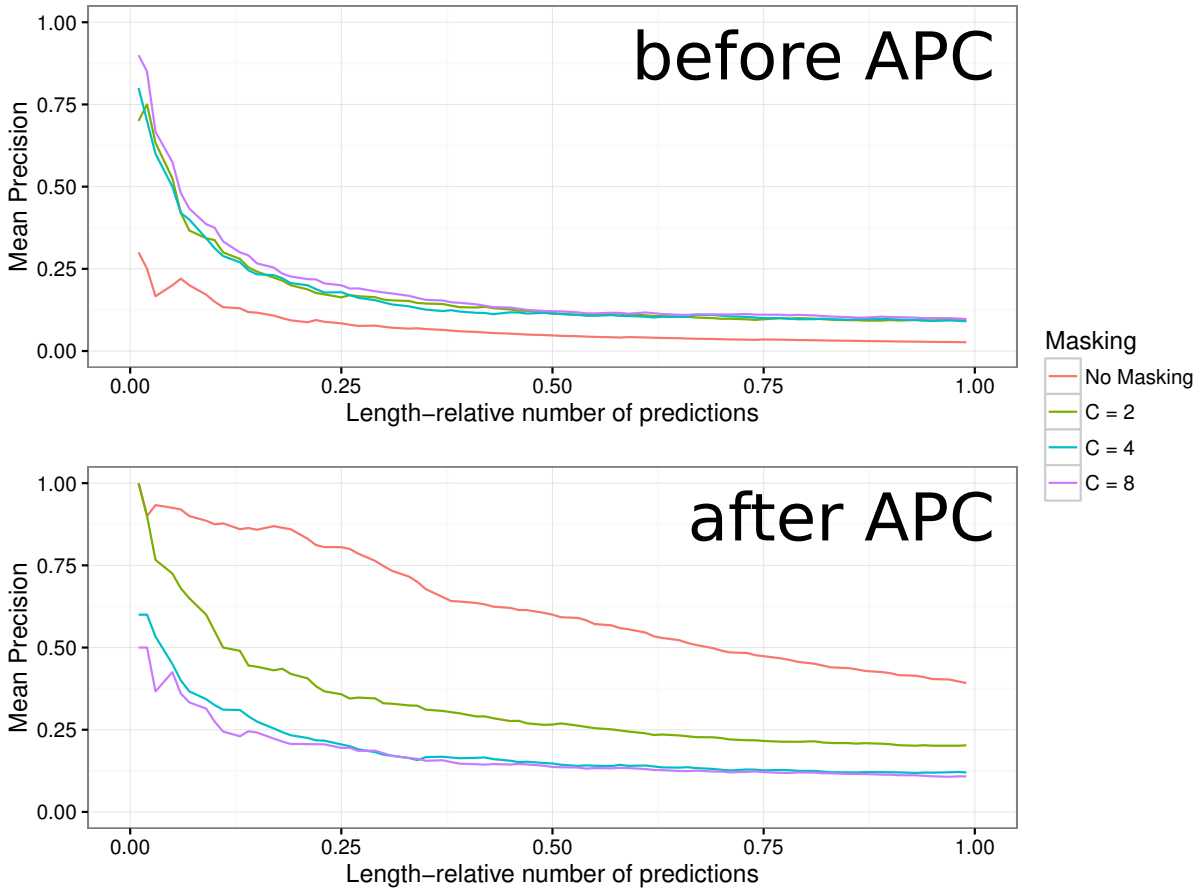


Figure 4.2: Evaluation of "cheating" regularization. While the masking increases precision as expected when not using APC (top panel), an increasing masking factor decreases precision after APC correction (bottom panel).

mean statistics over the total coupling matrix since most cells of the total coupling matrix show the background distribution.

The second assumption becomes problematic when modifying the underlying probability distribution to improve contact prediction, however, since the inclusion of any modification to prior or optimization strategy will affect all of the matrix and thus shift the background distribution assumed by the matrix. This means that when attempting to improve contact prediction through an altered probabilistic model, APC has to be replaced for improvements to become visible.

4.2 Eigenvalue Interpretation of APC

For the largest eigenvalue λ_0 of the pre-APC coupling matrix and its corresponding eigenvector $\boldsymbol{v}_0$, the background coupling C_b of Equation 2.1 is an approximation of the outer product of the first eigenvector scaled by the first eigenvalue:

$$C_b \approx \boldsymbol{v}_0 \boldsymbol{v}_0^T \times \lambda_0 \tag{4.1}$$

The identity was confirmed empirically using contact prediction data. Because of this identity, APC can be reinterpreted as simply removing the highest degree of variation within a coupling matrix. While not discussed in the literature around APC, this result greatly simplifies understanding of APC and is further indication that measuring background model statistics from the same input matrix that should be corrected can lead to wrong corrections when the underlying model is changed.

4.3 Entropy Correction

Since the problems with APC described above come from using the contact matrix to calculate a background model, a replacement for APC was developed that would use statistics from the input multiple sequence alignment instead and thus eliminate any dependencies from changes in the probabilistic modeling.

As shown in Figure 2.2b, the main noise component in contact prediction comes from entropic effects since the strength of covariation that can be measured at a pair of residue positions is dependent on the product of entropies $h(i)$ of participating columns i. A scatterplot shown in Figure 4.3 shows that the APC correction term C_b in realistic contact maps correlates with the geometric mean of per-column entropies for the corresponding alignments. Consequently, the *Entropy Correction* (EC) was defined as the original coupling matrix minus the geometric mean of per-column entropies, scaled by factor α:

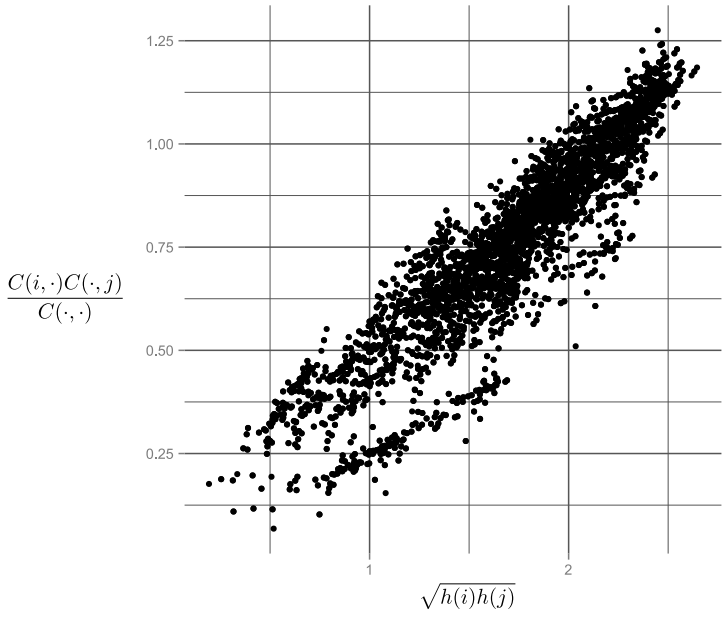


Figure 4.3: Entropy Correction correlates well with APC term ($\rho = 0.91$). For the majority of points in the contact maps under comparison, the geometric mean of per-column entropies correlates well with the correction term from Average Product Correction.

$$C^{EC}(i,j) = C(i,j) - \alpha E(i,j) \tag{4.2}$$

$$E(i,j) = \sqrt{h(i)h(j)} \tag{4.3}$$

$$h(i) = \sum_{a=1}^{20} P(x_i = a) \log P(x_i = a) \tag{4.4}$$

The correction term in Equation 4.2 no longer depends on statistics of the coupling matrix. However, the additional parameter α is introduced that will have to be determined.

4.3.1 Finding the Correction Magnitude

To find a suitable correction magnitude α, synthetic multiple sequence alignments of varying numbers of sequence, topology and evolutionary distance were produced using the procedure outlined in Chapter 13. Using structural knowledge associated with the multiple sequence alignments, a correction magnitude α^* was chosen to maximize the area-under-ROC-curve for the resultant contact predictions. The resulting α^* values are plotted against evolutionary distance and sequence count in Figure 4.4 to determine how the quantities are related to another.

While some correlation between the number of sequences, the evolutionary distance and the ideal correction magnitude can be seen, there is considerable variance in the α^* values

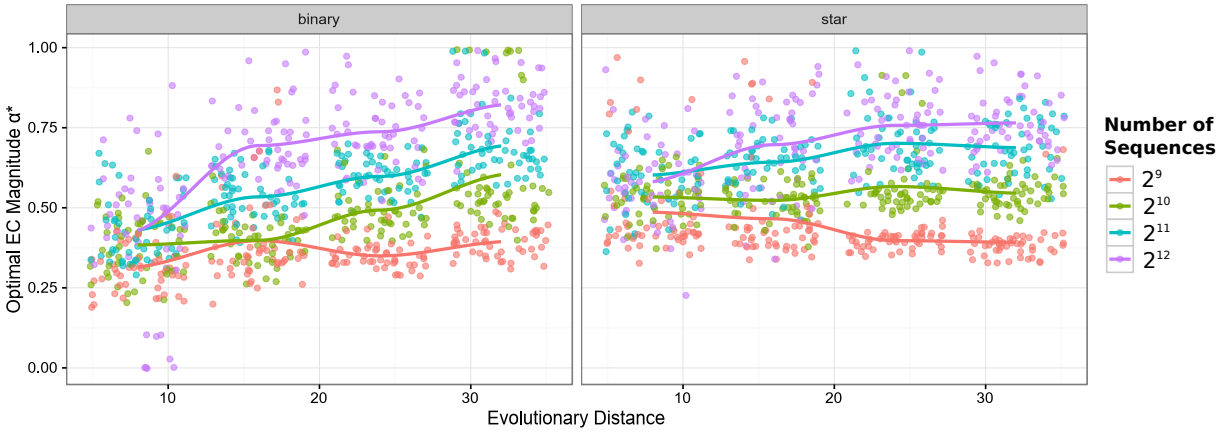


Figure 4.4: Optimal Entropy Correction Magnitudes for Synthetic Alignments.

and no functional form becomes apparent for the relationship even on simple synthetic data. In order to investigate the effectiveness of entropy correction without spending more time on calibration of the correction magnitude, several heuristic strategies were chosen.

Maximum-AUC Entropy Correction (MAUC-EC) Maximum-AUC EC uses the α^* derived from structural information to perform the best-possible correction term. While this strategy can be considered cheating since it relies on structural information, it can serve as an upper bound to the performance of entropy correction.

Renormalizing Entropy Correction (NORM-EC) The coupling score matrix to be corrected is linearly rescaled so that the lowest matrix element corresponds to 0 and the 90% percentile corresponds to 1 in the rescaled matrix with values from the 90% to 100% percentiles having values larger than 1. Since it is expected that 90% of the matrix will be comprised of background coupling signals, this operation corresponds to scaling the range of background couplings into the range $[0, 1]$. The Entropy Correction matrix is also rescaled into the interval $[0, 1]$ and subtracted from the rescaled coupling matrix.

Minimum-Frobenius-Norm Entropy Correction (F-EC) The correction magnitude α_F is chosen to minimize the frobenius norm of the entropy-corrected coupling matrix (cf. Equation 4.2) and thus minimize the amount of coupling remaining after correction.

$$\alpha_F = \arg\min_{\alpha} \|C - \alpha E\|_2 \tag{4.5}$$

By deriving for α, an analytical term can be determined for α_F:

$$\alpha_F = \frac{\sum_{i,j} C(i,j)E(i,j)}{\sum_{i,j} E^2(i,j)} \tag{4.6}$$

4.3.2 Discussion of Entropy Correction

The different entropy correction strategies were applied to the real-world set of protein domain MSAs described in Section 3.1. Figure 4.5 compares the different entropy correction strategies in their prediction performance to uncorrected pseudo-likelihood contact prediction matrices and APC-corrected matrices. Apart from the renormalizing EC that appears to fail at correcting entropic effects in the coupling matrix, both Minimum-Frobenius-Norm-EC and Maximum-AUC-EC are successful in correcting the entropic effects from the coupling matrix, with a slim lead in the performance metrics by APC that can be explained by the APC method additionally correcting for phylogenetic effects by enforcing a mean coupling level in the output matrix.

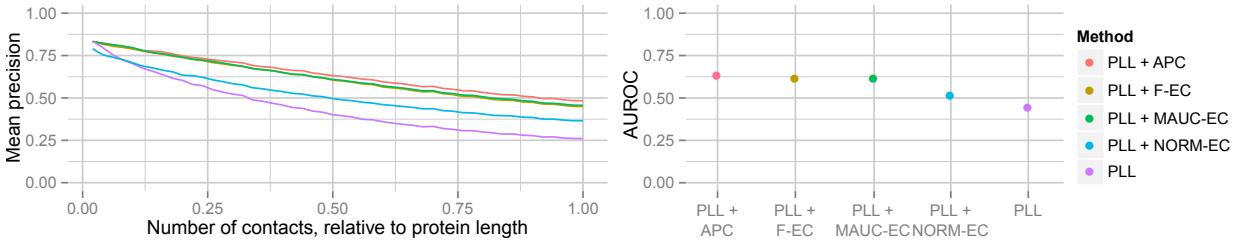


Figure 4.5: Evaluation of Entropy Correction. Entropy-corrected contact predictions achieve almost the same precision as APC-corrected contact predictions. The remaining discrepancy in precision scores can be explained by the uncorrected phylogenetic noise.

Since the original goal of entropy correction was to correct for entropic effects *using only the input MSA to compute statistics* in order to be more tolerant to changes in the contact prediction models, the discussed strategies can only be considered as partially successful since they either rely on structural information for Maximum-AUC-EC or still compute summary statistics on the contact map in the case of Minimum-Frobenius-Norm-EC. Still, the Minimum-Frobenius-Norm EC can be considered more stable than APC since it only uses a single value α_F to correct a whole coupling matrix while APC uses per-column mean coupling values for each column to determine correction magnitude. Through further investigation, it might however still be possible to find a relationship between input MSA parameters and the optimal correction magnitude α^*.

Chapter 5

Conclusion

While currently published contact prediction methods are already very successful at predicting contacts for well-sequenced protein families, the underlying statistical models carry many simplifying assumptions and have heuristics stacked on top of them that greatly contribute to the predictive performance.

The reasonable assumption that replacing assumptions with a more explicit bayesian model of protein interactions would improve predictive performance turned out to be false. This phenomenon can be explained by that heuristics such as the Average Product Correction make assumptions about the score distributions of the generated contact predictions. When a different modelling of protein interactions changes these distributions, the heuristics fail and produce worse predictions that cancel out any progress made by the improvement in modelling. Further improvement in contact prediction is thus blocked as long as these heuristics are used.

In order to unblock further progress in contact prediction methods, the heuristics have to be replaced by well-understood probabilistic models. An attempt was made in replacing the Average Product Correction with a correction solely dependent on the per-column entropies in the input alignment, with the remaining phylogenetic noise corrected by the APC left uncorrected. If a reliable calibration of the magnitude of entropy correction can be found that does not depend on statistics on the prediction matrices, the entropy correction could serve as a suitable replacement.

Part II

Accelerating Evolutionary Coupling Methods

Chapter 6

Introduction to High Performance Computing

With evolutionary coupling methods based on MRFs turning out to become the new state of the art in residue-residue contact prediction and existing implementations being implemented in slow and proprietary programming languages, we saw a need for a high-quality free implementation of these methods as a starting point for further improvements.

High Performance Computing (HPC) is the section of computer science focusing on optimizing algorithms to be executed as efficiently as possible on a given machine architecture, exploiting parallelism to further decrease computation times. We used our previous experience in HPC to create a high-speed implementation of a MRF pseudo-likelihood contact predictor called CCMpred. This chapter will go into important considerations for making algorithms run efficiently on CPU and GPU hardware architectures.

6.1 Hardware Architectures

In order to optimize algorithms effectively, it is vital to be aware of the execution properties of the underlying hardware architecture. The following section will give an introduction to the CPU and GPU architectures used in this work.

6.1.1 CPU Architectures

For the purpose of HPC, a computer can be simplified to a series of central processing unit (CPU) cores used for calculations and controlling program flow that operate on a hierarchy of memories of increasing size and decreasing access speeds.

Modern computers typically have 4 to 32 processor cores that can execute roughly 1.5×10^9 to 4.0×10^9 operations per second independently from another and theoretically achieving on the order of 1.6×10^{10} floating point operations per second and processor core. In practice, the computational efficiency is highly dependent on the ordering of operations

and how quickly the required data can be loaded from memory. For this matter, it is important to consider the memory hierarchy present in typical computers.

Memory type	Time to first data $/s$	Bandwidth $/bytes\,s^{-1}$	Capacity $/bytes$
L1 Cache[79]	10^{-9}	n/a	10^5
L2 Cache[79]	10^{-8}	n/a	10^6
L3 Cache[79]	10^{-7}	n/a	10^9
DDR4 RAM[80]	10^{-6}	10^{10}	10^{10}
SSD[81]	10^{-5}	10^9	10^{12}
HDD[82, 83]	10^{-3}	10^8	10^{13}

Table 6.1: Approximate memory access timings, bandwith and capacity for different memory types. As we move further away from the CPU, capacity, access latencies and capacity increase while bandwidth decreases.

As seen in Table 6.1, the memory capacity increases as we move away from the CPU, but so do access times. If an algorithm's data can be predictably read from high-capacity storage into RAM and caches, fetches of subsequent data will hit the caches and main memory instead of causing additional reads from storage, increasing performance. The most important task for writing efficient programs for CPU architectures is therefore to lay out memory so that it can be accessed as sequentially as possible.

6.1.2 CUDA GPU Architecture

Additional considerations need to be made when calculating on general-purpose graphics processing units (GPGPUs). According to the NVIDIA Cuda programming model [84], a graphics card consists of several hundreds to thousands of processor cores organized in streaming multiprocessors (SM), plus its own hierarchy of memories.

Compared to CPU cores, GPU cores typically have a lower clock rate (on the order of 10^9 operations per second) and cannot execute instructions independently. Instead, groups of 32 processors are grouped together in a *warp* of threads executing the same instruction in lock step. For this reason, it is important to divide problems up into several thousands of subtasks to fully occupy all GPU cores and minimize branching within a warp so that groups of threads do not have to wait for others to complete.

Since a GPGPU carries its own limited amount of memory, there are both overheads in transferring data to and from the GPU. Consequently, a GPGPU computation will only be sensible if the problem fits into the GPU memory and a sufficient amount of computation in relation to data transfer can be performed on the GPU before data needs to be sent back to the computer memory.

Similar to the CPU programming model, there is a memory hierarchy for GPU programming: a large (several gigabytes) amount of slow *global memory* is available to all cores while the faster memories are only available to subgroups of threads (*shared memory*, typically 48 kilobytes per group of up to 1024 threads) or only available to a single thread

(*registers*). Since memory is accessed in parallel by several threads, memory access should ideally be *coalesced*, i.e. all threads of a warp accessing a contiguous and aligned segment of 128 bytes in global memory.

6.2 Parallel Programming

Before 2005, processor were increasing their speeds through *Dennard scaling*: The power use (and consequently, heat generation) of a processor stays constant within a certain area and CPU frequency. If we want to increase the speed of a CPU, we therefore had to shrink the transistors to complete more computation in the same size of chip. Since 2005, however, processor manufacturers have begun hitting physical lower limits for transistor size: small transistors increase the risk of current leakage, causing interference. Since we cannot shrink transistors further, we cannot get higher clock rates. Instead, CPU manufacturers are now increasing processor power by producing processors where several cores can complete computations in parallel independently from one another but still sharing memory.

In this section, we will discuss strategies for dividing up problems to be computed in parallel, first by exploiting all of the available arithmetic units of a single processor core using SIMD instructions and then going into multi-core parallelization strategies.

6.2.1 SIMD Parallel Programming

The idea of *single instruction, multiple data* (SIMD) programming is that while we cannot increase the speed at which instructions are processed further, we can increase the amount of work being done per instruction. Say we have a large in-memory array on which we want to perform computations. Instead of processing one array element after the other, we can use several arithmetic units of a processor core by letting one processor core process several subsequent array elements according to special instructions (as specified in the SSE, SSE2, SSE3, AVX and AVX2 standards). This lets us perform computations on 4 to 16 array elements in a single processor core in a single clock cycle. The instructions can be placed into the code explicitly as *SIMD intrinsics*[85] or if it is clear to the compiler that an array operation will be processed sequentially over all array elements, can be automatically inserted by the compiler.

6.2.2 Shared Memory Parallel Programming

Now that we have maximized the amount of computation that can be done with a single processor core, the next step of parallelization is to employ several processor cores in parallel. In the *shared memory* parallel programming model, all processor cores share access to a common main memory so that for efficient computation, the main responsibility of the programmer is to divide up the work in a way that processors can work as independently from one another as possible without having to wait too long for synchronization.

A very convenient yet powerful technique for parallelizing shared-memory programs is to use the OpenMP[86] programming interface that is supported by major C and C++ compilers. In the simplest use case, an annotation to a *for* loop can be made saying that the individual loop calls can be executed in parallel. From this information, the compiler can infer which variables should be kept local to each of the independent threads and which should be shared between threads. Additional annotations for critical regions that should be executed sequentially and reduction operations are also possible. After compilation, the program will automatically use as many threads as there are cores available in the system or as many as have been specified by an environment variable. By compiling the same code while ignoring the OpenMP annotations, it is also possible to have a sequential version of the algorithm use the exact same code.

6.2.3 CUDA Parallel Programming

Finally, the last parallel programming model explained here is the *CUDA* (formerly *Compute Unified Device Architecture*) model of parallel programming on general-purpose graphics processing units (GPGPUs). As processor threads execute in groups of 32 called *warps*, CUDA shares similarities to SIMD programming and is sometimes also called SIMT (single instruction, multiple thread) programming.

In CUDA, work is described in a *kernel* whose execution is divided up in a virtual three-dimensional coordinate system called the *grid*. The grid is divided into *blocks* that the GPU can schedule on a *streaming multiprocessor* as it sees fit, without possibility of programmer control. This is to ensure efficient parallelization and that newer-generation GPUs can increase performance by simply increasing the number of streaming multiprocessors. Within a block, work can be further divided into *threads* that are assigned to the individual GPU cores. Within a block, some kilobytes *shared memory* can be requested that can be accessed by all threads belonging to the block to exchange information during computation. Alternatively, every thread always has access to the significantly slower *global memory* of the GPU which is several gigabytes big.

Since a GPU has its own memory chips, the programmer has to manage memory allocation and deallocation on both the CPU (using `malloc` and `free` commands) and the GPU (using `cudaMalloc` and `cudaFree` commands). Before data can be used on the GPU, it has to be copied to the device using the `cudaMemcpy` command which can also be used to copy results back to the CPU memory once computation is finished. Since memory bandwidth between GPU and CPU is limited, a CUDA implementation of an algorithm will only be more efficient if a sufficient amount of computation can be performed on the GPU before communication with the host computer becomes necessary again.

Because of the hardware limitations of a GPU, the amount of registers, shared memory and threads allocated to a single block will determine how many of the cores of a streaming multiprocessor can be used to execute a kernel since all cores have to share resources from a common pool. The ratio of computing to available cores is called the *occupancy* and it's important to allocate resources so that occupancy can be maximized.

Chapter 7

Accelerating Evolutionary Coupling Methods

With the different computing architectures and parallel programming models explained, we can now apply them to the use case of accelerating pseudo-likelihood evolutionary coupling methods. First, we will outline steps to make the sequential computation as efficient as possible by re-using intermediate results in the computations and optimizing the order in which variables are stored in memory. After this, we will go into detail how computations can be divided up to be done in parallel.

7.1 Complexity of Required Computation Steps

A pseudo-likelihood maximization program consists of two components — a numerical optimization method such as *conjugate gradients* to maximize a function given its parameters and gradients, and the likelihood function that computes the function value and gradients for each step of the optimization procedure. If the protein family under consideration has N sequences with L positions each and we want to optimize for I iterations, the complexity of the conjugate gradient method will be in $O(SL^2)$ since the number of parameters in the model grows quadratically in the number of columns in the alignment. The actual pseudo-likelihood and gradient computation will have a complexity in $O(SNL^2)$ as will become apparent in a moment.

Since efficient algorithms for numerical optimization already exist, we will focus our attention on optimizing the pseudo-likelihood and gradient computation which also takes up the majority of run-time.

7.2 Decomposing Computations for Reuse

The pseudo-likelihood of coupling parameters $\boldsymbol{v}, \boldsymbol{w}$ to generate alignment $\boldsymbol{X}$ is defined as such:

$$\begin{aligned}
\mathrm{pll}(\boldsymbol{v}, \boldsymbol{w}|\boldsymbol{X}) &= \log \prod_{n=1}^{N} \prod_{i=1}^{L} p(X_i = x_i^n | (x_1^n, \ldots, x_{i-1}^n, x_{i+1}^n, \ldots x_L^n, \boldsymbol{v}, \boldsymbol{w})) \\
&= \sum_{n=1}^{N} \sum_{i=1}^{L} \log \frac{\exp\left[v_i(x_i^n) + \sum_{\substack{j=1 \\ j\neq i}}^{L} w_{i,j}(x_i^n, x_j^n)\right]}{\sum_{c=1}^{21} \exp\left[v_i(c) + \sum_{\substack{j=1 \\ j\neq i}}^{L} w_{i,j}(c, x_j^n)\right]} \\
&= \sum_{n=1}^{N} \sum_{i=1}^{L} \left[v_i(x_i^n) + \sum_{\substack{j=1 \\ j\neq i}}^{L} w_{i,j}(x_i^n, x_j^n) - \log Z_i^n \right]
\end{aligned} \tag{7.1}$$

With the *partition function* normalization term Z_i^n:

$$Z_i^n = \sum_{c=1}^{20} \exp\left[v_i(c) + \sum_{\substack{j=1 \\ j\neq i}}^{L} w_{i,j}(c, x_j^n) \right] \tag{7.2}$$

By calculating the derivative of equation 7.1 of $w_{i,j}(a, b)$, we can obtain the gradient for pairwise emission parameters (see Appendix A for the full derivation):

$$\begin{aligned}
\frac{\partial pll(\boldsymbol{v}, \boldsymbol{w}|\boldsymbol{X})}{\partial w_{i,j}(a, b)} &= \sum_{n=1}^{N} \left\{ I\left(x_j^n = b\right) \cdot \left(I\left(x_i^n = a\right) - \frac{\exp\left[v_i\left(a\right) + \sum_{\substack{k=1 \\ k\neq i}}^{L} w_{i,k}\left(a, x_k^n\right) \right]}{\sum_{c=1}^{20} \exp\left(v_i\left(c\right) + \sum_{\substack{k=1 \\ k\neq i}}^{L} w_{i,k}\left(c, x_k^n\right) \right)} \right) \right\} \\
&= \sum_{n=1}^{N} \left[I(x_j^n = b) \left[I(x_i^n = a) - p(X_i = a | (x_1^n, \ldots, x_{i-1}^n, x_{i+1}^n, \ldots x_L^n), \boldsymbol{v}, \boldsymbol{w}) \right] \right]
\end{aligned} \tag{7.3}$$

Similarly, we can derive for $v_i(a)$ to obtain the gradient for single emission parameters:

$$\frac{\partial pll(\boldsymbol{v},\boldsymbol{w}|\boldsymbol{X})}{\partial v_i(a)} = \sum_{n=1}^{N}\left\{ I\left(x_i^n = a\right) - \frac{\exp\left[v_i\left(a\right) + \sum\limits_{\substack{k=1\\k\neq i}}^{L} w_{i,k}\left(a, x_k^n\right)\right]}{\sum\limits_{c=1}^{20}\exp\left(v_i\left(c\right) + \sum\limits_{\substack{k=1\\k\neq i}}^{L} w_{i,k}\left(c, x_k^n\right)\right)}\right\}$$

$$= \sum_{n=1}^{N}\left[I(x_i^n = a) - p(X_i = a|(x_1^n,\ldots,x_{i-1}^n,x_{i+1}^n,\ldots x_L^n),\boldsymbol{v},\boldsymbol{w})\right] \tag{7.4}$$

From looking at equations 7.1, 7.2, 7.3 and 7.4, we can see that certain terms appear in several places for these computations. We can therefore save computation time by identifying and pre-computing these terms ahead of actual pseudo-likelihood and gradient computation:

$$\text{sumPot}(n,a,i) = v_i(a) + \sum_{\substack{j=1\\j\neq i}}^{L} w_{i,j}(a, x_j^n) \tag{7.5}$$

$$\text{Z}(n,i) = \sum_{a=1}^{21}\exp\left[\text{sumPot}(n,a,i)\right] \tag{7.6}$$

$$\text{pCond}(n,a,i) = \frac{\exp\left[\text{sumPot}(n,a,i)\right]}{\text{Z}(n,i)} \tag{7.7}$$

We can now rewrite equations 7.1 to 7.4 in a simplified way using these precomputed terms:

$$\text{pll}(\boldsymbol{v},\boldsymbol{w}|\boldsymbol{X}) = \sum_{n=1}^{N}\sum_{i=1}^{L}\log\text{pCond}(n, x_i^n, i) \tag{7.8}$$

$$\frac{\partial pll(\boldsymbol{v},\boldsymbol{w}|\boldsymbol{X})}{\partial v_i(a)} = N(x_i = a) - N\,\text{pCond}(n,a,i) \tag{7.9}$$

$$\frac{\partial pll(\boldsymbol{v},\boldsymbol{w}|\boldsymbol{X})}{\partial w_{i,j}(a,b)} = N(x_i = a \wedge x_j = b) - \sum_{n=1}^{N} I(x_j^n = b)\,\text{pCond}(n,a,i) \tag{7.10}$$

Where $N_i(a)$ corresponds to the number of times amino acid a appears in column i and $N_{i,j}(a,b)$ corresponds to the number of times amino acids a and b co-occur in columns i and j (matrices that can be pre-computed once before optimization begins). With all the precomputed terms identified, we can formulate algorithm 7.1 for pre-computing values and then calculating log-likelihood and gradients.

Algorithm 7.1 Pseudo-log-likelihood computation with precomputed values.

for $i, j \leftarrow 1 \dots L, i \neq j, a, b \leftarrow 1 \dots 20$ **do**
  // Initialize pairwise gradients with minuend of equation 7.10
  $g_{i,j}(a, b) \leftarrow N_{i,j}(a, b)$
end for
$\text{pll} \leftarrow 0$
for $n \leftarrow 1 \dots N$ **do**
  for $i \leftarrow 1 \dots L$ **do**
    $\text{Z}(n, i) \leftarrow 0$
    for $a \leftarrow 1 \dots 20$ **do**
      // Calculate sumPot in $O(NL^2)$
      // Note random access depending on x_j^n
      $\text{sumPot}(n, a, i) \leftarrow v_i(a) + \sum_{\substack{j=1 \\ j \neq i}}^{L} w_{i,j}(a, x_j^n)$
      $\text{Z}(n, i) \leftarrow \text{Z}(n, i) + \exp \text{sumPot}(n, a, i)$
    end for
    for $a \leftarrow 1 \dots 20$ **do**
      $\text{pCond}(n, a, i) \leftarrow \frac{\exp \text{sumPot}(n,a,i)}{\text{Z}(n,i)}$
      $g_i(a) \leftarrow N(x_i = a) - N \, \text{pCond}(n, a, i)$
      // Compute subtrahend of equation 7.10 in $O(NL^2)$
      // Note random access depending on x_j^n
      for $j \leftarrow 1 \dots L \wedge j \neq i$ **do**
        $g_{i,j}(a, x_j^n) \leftarrow g_{i,j}(a, x_j^n) - \text{pCond}(n, a, i)$
      end for
    end for
    $\text{pll} \leftarrow \text{pll} + \log \text{pCond}(n, x_i^n, i)$
  end for
end for

7.3 Efficient Memory Access

While the naive algorithm 7.1 will compute the pseudo-log-likelihood and gradients correctly, it currently will have to frequently jump to different memory locations to do its job. From the introduction of this section, we know that random memory accesses require an additional seek in memory every time we jump to a non-prefetched memory location, giving us significant slowdowns. The next step in optimization is therefore to make sure that memory can be accessed as linearly as possible by rearranging memory layout and the order of access. For the sake of simplicity, we will only consider the most important accesses to pairwise coupling parameters $w_{i,j}(a, b)$ and pairwise coupling gradients $g_{i,j}(a, b)$ here but memory access patterns for all other variables was also optimized.

7.3.1 Variable Re-ordering

A simple strategy for ensuring that memory access to a multidimensional array is as linear as possible is to use the same order of array indices as the order of **for** statements so that the innermost **for** loop goes over the quickest-changing dimension in the array as seen in algorithm 7.2 and figure 7.1a. On the other hand, when reading in a transposed manner (see algorithm 7.3 and figure 7.1b), most cell accesses will result in jumping in memory to another row, incurring new seek times if the array is bigger than what fits into the processor caches.

Algorithm 7.2 Efficient linear memory access on a 2D array

```
k ← 0
for i ← 0..3 do
    for j ← 0..3 do
        A(i, j) ← k
        k ← k + 1
    end for
end for
```

Algorithm 7.3 Inefficient transposed memory access on a 2D array

```
k ← 0
for j ← 0..3 do
    for i ← 0..3 do
        A(i, j) ← k
        k ← k + 1
    end for
end for
```

In the case of our pseudo-likelihood algorithm as seen in algorithm 7.1, the in-memory index ordering for $w_{i,j}(a, b)$ and $g_{i,j}(a, b)$ should consequently be (i, a, j, b). However, as

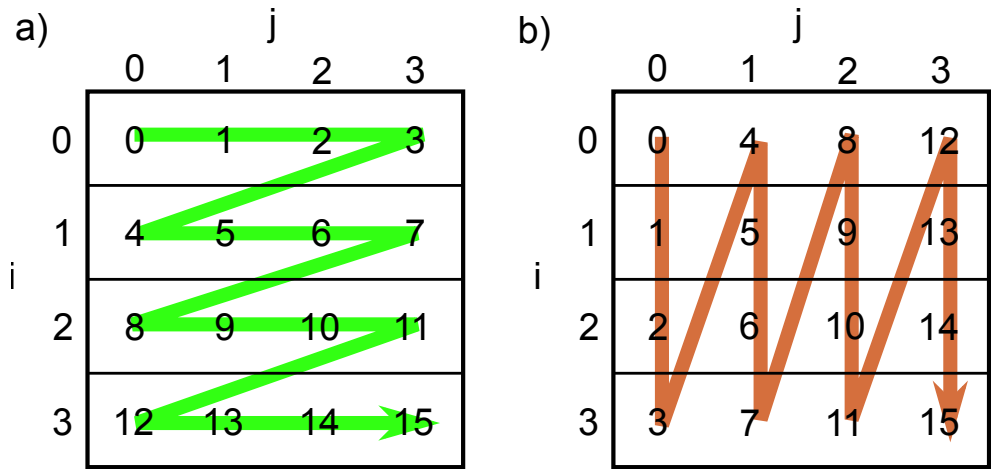


Figure 7.1: Efficient and inefficient memory access on a 2D array $A(i, j)$ with memory accesses in row-major order. **a.** Optimally efficient linear memory access as seen in algorithm 7.2. **b.** Inefficient transposed access as seen in algorithm 7.3

we can see in lines 14 and 23 of algorithm 7.1, there is an additional complication in that the memory addresses we have to access depend on the sequence data at x_j^n. Since we cannot control which amino acid we observe, we have to compensate for the random seeks introduced by the data by reordering our operations so that (n, j) stays constant for as long as possible so that we only make random jumps rarely and then linearly process data starting from the jump destination. Since $b = x_j^n$ for all our accesses, our optimal memory layout therefore becomes (x_j^n, j, a, i).

7.3.2 Exploiting Symmetry

As seen above, accessing a matrix in transposed coordinates is inefficient. However, for the purposes of our markov random field model, we can exploit the symmetry in the data: $w_{i,j}(a, b) = w_{j,i}(b, a)$. This means that as long as we are willing to pay the additional memory overhead of keeping both the necessary triangle matrix for $w_{i,j}(a, b)$ for $i < j$ and also the transposed triangle matrix for $j > i$ in memory, we can always replace inefficient transposed memory accesses by their efficient linear counterparts. The necessary re-symmetrization step after each evaluation step has unfavorable memory access characteristics but since it has a much lower complexity of $L^2 20^2$ compared to gradient calculation, the speedups gained by this optimization greatly outweigh the slowdown incurred by this additional step.

7.3.3 The Final Sequential Algorithm

Putting all considerations above together, we arrive at algorithm 7.4, the memory access-optimized version of a pseudo-log-likelihood function and gradient calculation. Note that

we now write $v(a,i)$, $w(b,j,a,i)$, $g(a,i)$ and $g(b,j,a,i)$ to denote the re-ordered indices in the corresponding potential and gradient arrays.

Algorithm 7.4 The final memory access-optimized pseudo-log-likelihood function

$g(a,i) = N_i(a)\ \forall i \in \{1 \dots L\}, a \in \{1 \dots 20\}$
$g(b,j,a,i) \leftarrow N_{i,j}(a,b)\ \forall i \neq j \in \{1 \dots L\}, a,b \in \{1 \dots 20\}$
$Z(n,i) \leftarrow 0\ \forall n \in \{1 \dots N\}, i \in \{1 \dots L\}$
pll $\leftarrow 0$
for $n \leftarrow 1 \dots N$ **do**
  for $a \leftarrow 1 \dots 20, i \leftarrow 1 \dots L$ **do**
    sumPot$(n,a,i) \leftarrow v(a,i)$
  end for
  for $j \leftarrow 1 \dots L$ **do**
    // Note: random memory jump here depending on x_j^n
    for $a \leftarrow 1 \dots 20, i \leftarrow 1 \dots L \wedge i \neq j$ **do**
      sumPot$(n,a,i) \leftarrow$ sumPot$(n,a,i) + w(x_j^n, j, a, i)$
    end for
  end for
  for $a \leftarrow 1 \dots 20, i \leftarrow 1 \dots L$ **do**
    $Z(n,i) \leftarrow Z(n,i) + \exp$ sumPot(n,a,i)
  end for
  for $a \leftarrow 1 \dots 20, i \leftarrow 1 \dots L$ **do**
    pCond$(n,a,i) \leftarrow \frac{\exp \text{sumPot}(n,a,i)}{Z(n,i)}$
    $g(a,i) \leftarrow g(a,i) -$ pCond(n,a,i)
  end for
  for $i \leftarrow 1 \dots L$ **do**
    $pll \leftarrow pll + \log$ pCond(n, x_i^n)
  end for
  for $j \leftarrow 1 \dots L$ **do**
    // Note: random memory jump here depending on x_j^n
    for $a \leftarrow 1 \dots 20, i \leftarrow 1 \dots L \wedge i \neq j$ **do**
      $g(x_j^n, j, a, i) \leftarrow g(x_j^n, j, a, i) +$ pCond(n,a,i)
    end for
  end for
end for

7.3.4 Parallelization with SIMD intrinsics

With the sequential algorithm optimized, it was now time to begin parallelizing the program using SIMD intrinsics. However, by inspecting the assembly generated by the GNU and Intel C compilers, it turned out that the code reorganizations necessary for ensuring memory access allowed the compilers to automatically detect the linear operations over

memory vectors and SSE and AVX instructions were inserted automatically. We also experimented with manually inserting SIMD intrinsics at appropriate locations but achieved similar runtimes as the automatically-generated assembly code.

7.4 Learning Evolutionary Couplings on Many Processor Cores

Having optimized our algorithm to make the most out of a single processor core, the next step is to decompose the computation into steps that can be performed in parallel on several processor cores in order to further speed up computation.

First, we need to identify what variables which part of the algorithm will have to read from and write to and whether there are any interdependencies. The algorithm will only have to read from the MSA x_i^n and the coupling potentials $v_i(a)$ and $w_{i,j}(a,b)$ so because these values stay constant within one iteration of computation, we will not have to worry about synchronization. Furthermore, while we write to the sumPot(n,a,i), $Z(n,i)$ and pCond(n,a,i) variables, any fixed sequence index n and will never need to write to the variables for another sequence index. Finally, we sum up pseudo-likelihood values and gradient values for all sequence and column indices.

Taking all these interdependencies into account, a simple parallelization strategy becomes apparent: We can launch separate parallel computations for each sequence index n by parallelizing the outer loop in algorithm 7.4. All writes to the precomputed terms will only affect their own sequence index n so we only need to make sure that accesses to the shared pll, $g(a,i)$ and $g(b,j,a,i)$ variables are sequential. The OpenMP standard supports the *pragma omp atomic* setting for this which will only minimally impact performance.

7.5 Learning Evolutionary Couplings on the GPU

Since pseudo-likelihood maximization breaks down to calculating various vector operations on very large matrices without too much synchronization necessary, it is a suitable candidate for computing pseudo-log-likelihood values and gradients on the GPU. The following section will outline the steps taken to make PLL maximization execute well on current GPUs.

7.5.1 Conjugate Gradients on the GPU

Initial experiments showed that when running PLL and gradient calculations on the GPU and transferring parameters and gradients between CPU and GPU for every iteration to do the numerical optimization on the CPU, the majority of runtime of the algorithm would spent on data transfer. We therefore realized that the only way of achieving highly performant PLL maximization using the GPU would be to transfer all data to the GPU once, then do all of the work (PLL and gradient calculation plus numerical optimization)

there and only copy back results after convergence. Because of this requirement, the amount of memory used on the GPU became crucial for success.

In PLL maximization, memory usage increases quadratically in the number of MSA columns. We can calculate the number of variables in the parameter space P from the number of columns L in the MSA as such:

$$P(L) = 20^2L^2 + 20L \tag{7.11}$$

For the purposes of numerical optimization, we will require several times this amount of memory since in addition to the memory required for storing the current model parameters, we will at least need to allocate memory for storing the current gradient estimates and most algorithms will need additional variables to ensure more efficient convergence. Assuming all parameters are stored as 4-byte single-precision floating point numbers, we will therefore need $4CP(L)$ bytes of memory in total, where C is the total number of times the parameters need to be stored in memory. For a simple gradient descent, $C = 2$. In the case of GPU computation, we needed to find an optimization algorithm with good convergence properties that is still memory-efficient enough to fit into the GPU memory. For a typical problem of $L = 300$, the problem size is $P(300) = 1.0 \times 10^8$ variables already. Assuming we store values as single-precision floating point values taking four bytes per value, the highest factor C we can fit into a 2 GB graphics card would be $C = 5$.

We initially planned on using the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm since it provides excellent optimization properties by estimating the curvature of the target function by generating a local estimate of the Hessian matrix to efficiently converge into the maximum of the function and is a typical choice for fitting log-linear models[87]. Unfortunately, the L-BFGS algorithm needs to maintain a history of the parameter and gradient values from previous iterations, giving it a $C = 13$ or higher for 10 iterations of history. Since this memory requirement is too high for our purposes, we decided to use the lower-C Conjugate Gradients algorithm instead.

The Conjugate Gradient algorithm is an efficient optimization algorithm that in addition to storing parameter and gradient stores a previous search direction vector, resulting in $C = 3$. Using the previous search direction, it can be guaranteed that the next chosen search direction will be orthogonal to all previously chosen search directions, leading to efficient convergence. In order to make use of conjugate gradients on the GPU, we implemented a CPU and GPU-supporting conjugate gradients library and released it as the open source library `libconjugrad`.

7.5.2 Reordering Memory for Streaming Multiprocessors

Similar to the optimization for CPUs, a crucial part of efficient optimization on the GPU was also to decide on an efficient memory layout. Instead of optimizing for linear memory access, the goal for efficient GPU programming is to optimize for coalesced memory access. Lucky for us, the same memory layout as discussed in Section 7.3 will also ensure coalesced

access as long as the variable positions are correctly aligned to 128 byte boundaries. By introducing some padding variables, this can easily be ensured.

7.5.3 Efficient Pre-Computation

All pre-computed values in equations 7.5, 7.6 and 7.7 can be efficiently pre-computed using a single kernel. Since we only read from potential variables $\boldsymbol{v}$ and $\boldsymbol{w}$ and only write to fixed array positions for one fixed (n, i), it makes sense to parallelize according to cells in the multiple sequence alignment. We execute a block per sequence index n in the MSA and within that block launch a separate thread for every position i. Figure 7.2 outlines the parallelization and computation strategy to calculate precomputed values.

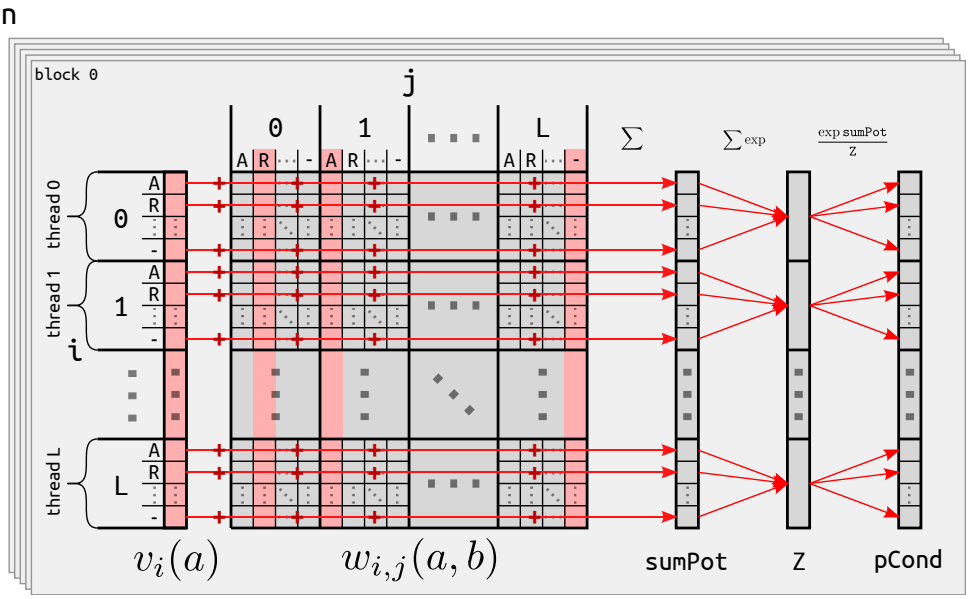


Figure 7.2: Pre-computation of frequently used values $\mathrm{sumPot}(n, a, i)$, $\mathrm{Z}(n, i)$ and $\mathrm{pCond}(n, a, i)$ by summing up single-emission potentials with pairwise emission potentials based on the amino acid at the current sequence position.

7.5.4 Efficient Gradient and Pseudo-Log-Likelihood Computation

With the pre-computation completed, calculating the actual gradient and pseudo-log-likelihood values becomes relatively easy. As seen in Equation 7.8, pseudo-log-likelihood computation can be completed by simply summing up $\log \mathrm{pCond}(n, a, i)$ for all n, x_i^n, i. We do this using a standard reduction kernel to maximize efficiency. For gradient computation, the minuend of Equations 7.9 and 7.10 can again be pre-calculated once before optimization begins. For the subtrahend, we need to repeatedly decrement appropriate

$g_{i,j}(a, x_j^n)$ by $\text{pCond}(n, a, i)$ depending on n, j and x_j^n. To maximize the usage of GPU cores and ensure we do not encounter write conflicts, a thread block is launched for every column j that decrements $g_{i,j}(a, x_j^n)$ by $\text{pCond}(n, a, i)$ for all n, a, i. Figure 7.3 illustrates the parallelization strategy for computing gradient subtrahends.

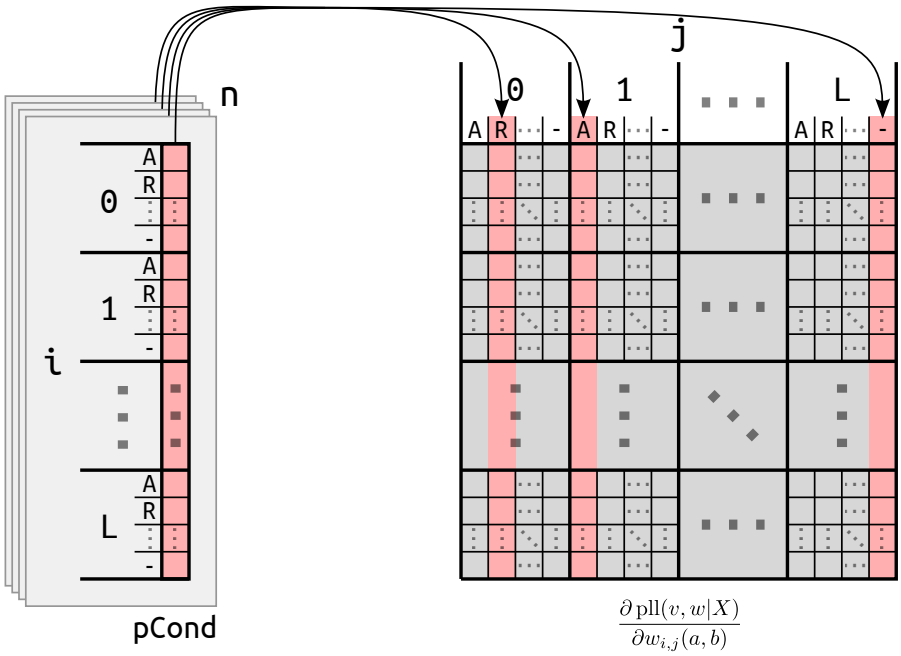


Figure 7.3: GPU computation of gradients. Conditional probability values $\text{pCond}(n, a, i)$ are atomically added to the pairwise gradient matrix $w_{i,j}(a, b)$ to compute the subtrahend of the pairwise gradient. The minuend corresponds to the pairwise counts and can be pre-computed once before optimization begins.

7.5.5 Performance Profiling

Using the NVIDIA visual profiler, we were able to identify the computation steps taking up the most time in order to focus our optimization efforts. Table 7.1 shows the different kernels together with their runtimes for a typically-sized example alignment. We can see that as expected, the kernels `d_compute_edge_gradients` and `d_compute_pc` together take up more than 90% of runtime. Looking into these algorithms further, we see that their runtime is dominated by the on-GPU memory bandwidth for accessing global memory. This indicates that our optimizations have resulted in very good GPU core utilization.

Time (%)	Time	Calls	Avg	Name	Shared memory / block (bytes)
6.85	14.67s	48	305.64ms	d_compute_edge_gradients	N (max 6144)
29.26	6.94s	48	144.59ms	d_compute_pc	$L \times 4$
1.87	442.66ms	208	2.13ms	vecdot_inter	$nthreads \times 4$
1.83	432.97ms	1	432.97ms	d_edge_gradients_histogram_weighted	$nthreads \times 21 \times 4$
1.07	252.70ms	1	252.70ms	d_initialize_w	–
0.88	209.71ms	48	4.37ms	d_add_transposed_g2	1056×4
0.72	171.42ms	48	3.57ms	d_compute_reg_inter_edges	$nthreads \times 4$
0.69	164.79ms	47	3.51ms	update_x	–
0.58	136.66ms	39	3.50ms	update_s	–
0.43	101.11ms	48	2.11ms	[CUDA memcpy DtoD]	–
0.34	80.81ms	305	264.95μs	[CUDA memcpy DtoH]	–
0.32	75.36ms	2	37.68ms	[CUDA memcpy HtoD]	–
0.12	28.32ms	48	589.99μs	d_compute_node_gradients	–
0.01	2.50ms	1	2.50ms	initialize_s	–
0.01	2.47ms	48	51.44μs	d_reset_edge_gradients	–
0.01	2.10ms	3	700.22μs	[CUDA memset]	–
0.01	1.93ms	48	40.16μs	d_compute_fx_inter	$nthreads \times 4$

Table 7.1: Overview over the kernels with highest runtimes for a synthetic alignment with $N = 3000$ and $L = 300$. The majority of runtime is spent in computing pairwise emission gradients and precomputed values.

Chapter 8

Conclusion

Algorithms can be restructured to execute several orders of magnitude quicker on a single processor core and make use out of today's many-core computers. While the particularities of a particular algorithm have to be considered when optimizing, a set of best practices such as identifying performance bottlenecks and linearizing memory access exist that provide a good starting point.

By applying the techniques for high performance computing discussed in this part, we were able to speed up residue-residue contact prediction by 35 to 112 times while maintaining the same predictive accuracy as the best competing methods. We released this software as the open-source software CCMpred. The software package allow interested researches to easily and quickly run their own contact predictions and provide a solid foundation for building improved contact prediction methods — both for ourselves and other researchers in the community. In fact, at the time of writing, CCMpred has already become a component for several contact prediction methods and is used to provide quick contact prediction response times in web servers.

A next logical step for even faster contact predictions would be to divide computation up between several computers, e.g. using the Message Passing Interface (MPI) API for communicating between computers. However, since the whole pairwise emission potential matrix needs to be accessed by each of the iterations of summing over sequences in each of the pre-computation steps run in every iteration of the optimization, applying the same parallelization strategy to a distributed memory system would lead to a large communication overhead. For this reason, further work would have to go into finding an efficient parallel algorithm for this use case before multi-computer parallelism will lead to faster overall computation times. Until then, the minute-scale run times on GPUs should be sufficient for typical domain sizes and more throughput can be achieved by predicting contacts for several protein families in parallel.

Part III

Interactive Evolutionary Coupling

Chapter 9

Introduction to Scientific Visualization in Modern Web Browsers

9.1 Modern Web Technologies

From the early 2000s until now, development of web pages have been revolutionized by the joint work of various standards organizations.

While the HTML5 standard jointly worked on by the *Web Hypertext Application Technology Working Group* (WHATWG) introduced new elements to be included in web pages such as the `<canvas>` element for drawing graphics, the HTML5 standard is typically used as a stand-in for the combination of several innovations in different standards worked on by different working groups. In addition, major web browsers greatly improved their performance and standards compliance. The other major development of the modern web is the introduction of the EcmaScript6 standard (developed by the Ecma International standards organization) for the JavaScript programming language and the significant performance improvements of introducing *just-in-time* (JIT) compilation to the V8 (powering Google Chrome) and SpiderMonkey (powering Mozilla Firefox) JavaScript engines.

Together with these two major improvements, new standards such as CSS3 (developed by the world wide web consortium) offer more powerful options for laying out documents, WebGL and WebCL (developed by the *Khronos Foundation*) allow access to the graphics card from the web browser and the introduction of Scalable Vector Graphics as a way to flexibly create and control graphical elements programmatically by modifying the document structure of the webpage offer a powerful environment for processing and displaying various types of information.

Since all of the aforementioned standards were jointly developed by all major browser and hardware manufacturers, they form a standardized platform without the need for the vendor-specific extensions to standards and proprietary plugins of the late 1990s and early 2000s. The websites created on these new standards run on a variety of computers and web

browsers, are accessible to people with disabilities and can also be accessed from mobile devices. In addition, they greatly reduce the complexity for web developers in supporting many edge cases and differences in features of different browser versions. Consequently, the new web platform forms a very attractive foundation for developing interactive experiences and has lead to an explosion in innovation in the open source world.

9.2 JavaScript for Visualization

Built on top of the new programming APIs, low-level wrapper libraries were created by the open source community to conveniently access all of the newly available features. For the purposes of data visualization, the *D3.js* (*Data-driven Documents*) library by Mike Bostock and the *Three.js* library by Ricardo Cabello build the foundation for the majority of modern web-based visualization.

D3.js applies the idea of Wilkinson's *Grammar of Graphics* [88] of decomposing charts into their commonly used components to the web by allowing developers to map data values onto the properties of elements in an HTML page. Since Scalable Vector Graphics are handled by modern browsers as substructures of the document tree of a HTML page, plots can be generated simply by mapping the data values of a data set onto e.g. the X and Y coordinates of point or line elements in a SVG graphic.

Three.js is a similarly low-level library for accessing the WebGL programming interface by specifying shapes through polygons, adding light sources and offloading the calculation of the final image to the graphics processing unit of the computer.

Since the two libraries mentioned here only provide the most basic primitives for creating graphics on web pages, many higher-level visualization libraries have been built on top of them such as *NVD3.js* for drawing commonly used plot types using D3.js.

9.3 Bioinformatics in the Browser

Following the developments of the new web, the bioinformatics community adopted the new visualization and web development techniques in their own web server developments. An important development is the BioJS javascript framework originally introduced by John Gómez [89]. BioJS provides a standard and registry for bioinformatics data visualization components that can be mixed together to create bioinformatic user interfaces without reinventing the wheel on commonly used components such as multiple sequence alignment viewers, sequence feature viewers or graph viewers. Even complex components such as 3D molecular viewers have been implemented in pure JavaScript, first by cross-compiling previously existing viewers into JavaScript (JSMol being a JavaScript version of JMol [90]) and later on from-scratch developments of molecular viewers using Three.js (WebGLMol (`https://github.com/biochem-fan/GLmol`) and PV [91]).

9.4 Performance Considerations

Even though JavaScript performance has drastically improved with new just in time-compiling virtual machines, it is still important to keep performance in mind when developing interactive visualizations in the browser. Re-layouting of documents especially in tables is still a computationally expensive operation so it is important to minimize the amount of re-layouting necessary by adding and removing elements in bulk and using fixed object sizes. When drawing in an SVG element, there are still noticeable performance problems when rendering complex graphics. It is therefore advisable to limit visualizations to drawing fewer than 10000 elements and using binned representations to reduce the amount of necessary drawing.

Chapter 10

Interactive Evolutionary Couplings

The interactive visualization techniques and technologies discussed above were applied to contact prediction by custom development of a series of visualization components based on HTML5 and JavaScript and running within a custom-made contact prediction webserver to provide an environment for understanding contact predictions and enriching them with expert knowledge about protein folding and the protein family under study. The following chapter will discuss the design of the individual components.

10.1 Contact Map Viewer

The most common way of visualizing residue-residue contact predictions is to map summed coupling scores to greyscale values in a 2D map of (i, j) positions. Since the resultant plot is symmetric across the main diagonal, the true contact map of the protein family under study is often plotted in one triangle matrix so that it can be directly compared with the predicted contact map in the other triangle matrix.

Since web-based visualizations allow for interactivity, the coupling map is enhanced by allowing user-selectable representation for each of the triangle matrices. The representations available are the common summed score matrix and APC-corrected summed score matrices, together with binary contact maps and shaded distance maps (if a PDB structure was provided).

To simplify comparing of predictions, highlighting an (i, j) position pair with the mouse will also highlight the corresponding (j, i) position in the other triangle matrix. The contact map viewer also serves as a selector panel for all subsequently discussed visualization components — clicking on an (i, j) position pair forwards that position pair to all other visualization components that will provide more details on the selected interaction.

10.2 Information in Pairwise Coupling Matrices

While contact prediction methods typically only consider summed score matrices, the coupling potentials learned for each of the alignment position combinations contain valuable information what amino acids interact at any given position pair. To visualize these interaction potentials, a custom plot was developed based on D3.js showing both the magnitude of individual coupling strengths and physical properties of the amino acids involved. Since sequence logos are a successful technique for visualizing sequence profiles by mapping frequency to letter size, the initial idea for the pairwise visualization was to map pairwise frequencies to letter sizes on a 2D matrix, creating 2D sequence logos.

When the user clicks on a residue position combination (i, j) on the contact map view, the coupling potentials are displayed in the coupling matrix viewer. The 20×20 possible amino acid combinations are shown in a 20×20 grid, with amino acids of similar physical properties (positive, negative, hydrophobic, aromatic, small, etc.) color-coded and positioned next to each other. For each of the 20×20 amino acid combinations, a circle split into two color-coded halves and labeled by the interacting amino acids is drawn with the area of the circle corresponding to the magnitude of the coupling parameters. Negative couplings are shown as upside-down circle since a negative coupling contains valuable information by signifying an interaction that needs to be avoided to stabilize the protein.

As seen in Figures 10.1c and 10.1d, common interactions stabilizing protein folds have a characteristic fingerprint in the coupling matrix viewer that can be easily recognized by the color-coded amino acids. For example, since hydrophobic amino acids are grouped in the beginning of both plot axes, a hydrophobic interaction will be visible as large dark grey circles in the bottom-left square of the coupling matrix viewer while salt bridges consisting of a positively-charged (blue) and a negatively-charged (red) amino acids will be shown as red-and-blue circles in off-diagonal matrix elements together with some same-charge amino acids showing negative couplings as upside-down circles.

By looking more closely at the coupling patterns observable in the coupling matrix viewer, it becomes apparent that not just the type of interaction, but also size preferences for the amino acids and consequently distance preferences for the interactions can be learned from the coupled amino acids. From this insight, another Master's thesis (and now doctoral thesis) project was started to predict interaction distances for creating better distance restraints.

10.3 Couplings on 3D Structures

To allow users to validate the contact predictions, it is helpful to map predicted contacts onto the corresponding 3D structure to validate the constraints and measure the true distances. For this, the WebGLMol (`https://github.com/biochem-fan/GLmol`) protein viewer based on Three.js has been extended to draw cylinders when a (i, j) position pair was selected on the contact map viewer. Figure 10.1b shows the protein structure viewer displaying a salt bridge interaction.

10.4 Web Server Development

All visualization components described above were combined with a simple bioinformatics workflow system to create a CCMpred webserver using the Django Python web application framework [92] and the Celery task worker system [93].

Starting from a simple submission page, the user can provide a single protein sequence or multiple sequence alignment that will be enriched with homologous protein sequences using HHblits [22] and residue-residue contacts predicted using CCMpred. Once the prediction is completed, the user gets redirected to the results page shown in Figure 10.2 where they can interactively explore the results of the contact prediction and optionally upload a 3D structure to compare predicted contacts to the structural data.

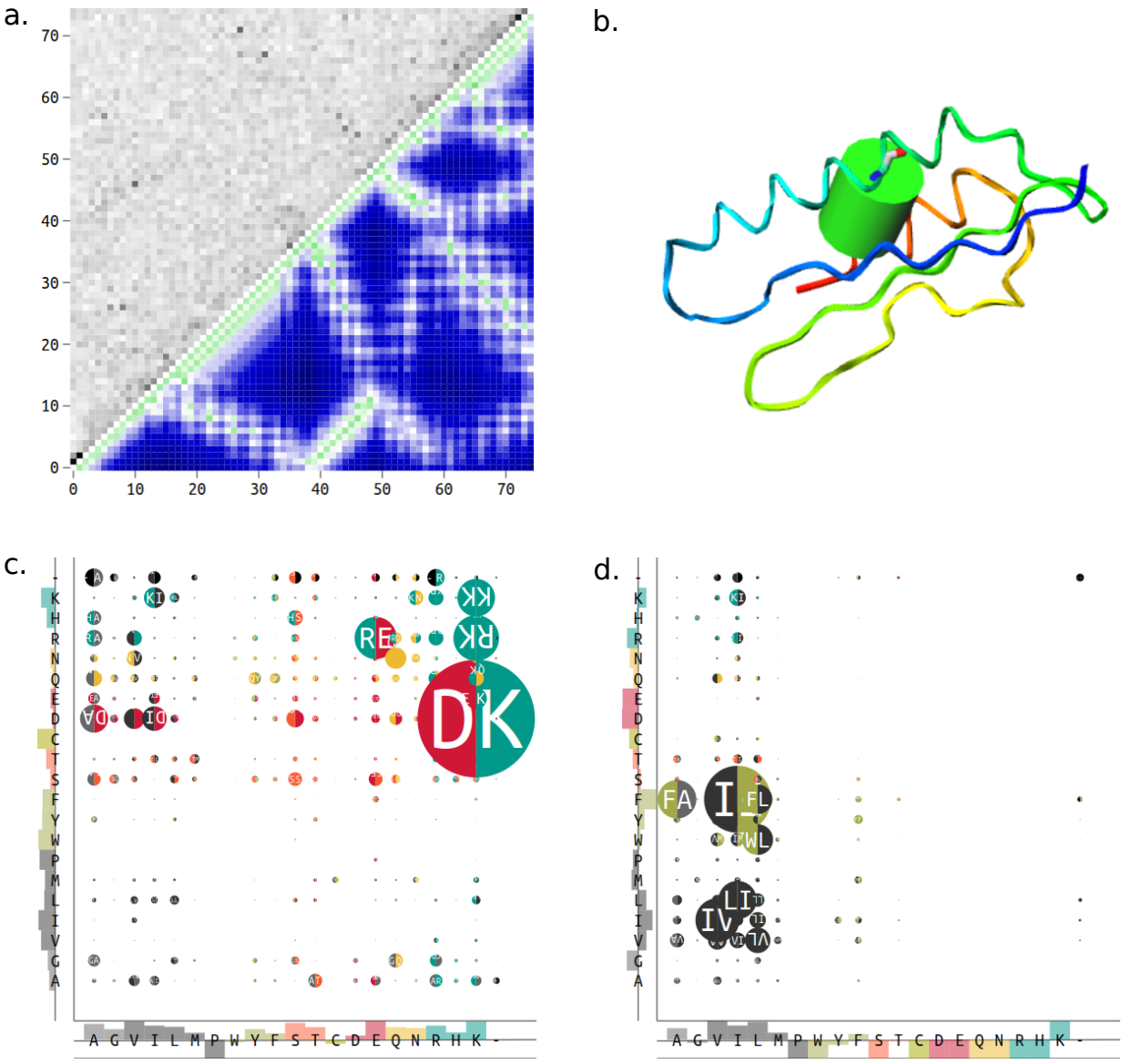


Figure 10.1: Interactive Visualization Components for the CCMpred webserver. **a.** The configurable contact map viewer can show summed score matrices, APC-corrected score matrices, physical contact maps and physical distance maps. Shown here is the APC-corrected predicted contact map and a distance map from the PDB structure. **b.** WebGLMol protein structure viewer with a custom addition of showing selected contacts as cylinders. **c.** Coupling matrix viewer showing a salt bridge interaction. **d.** Coupling matrix viewer showing a hydrophobic interaction.

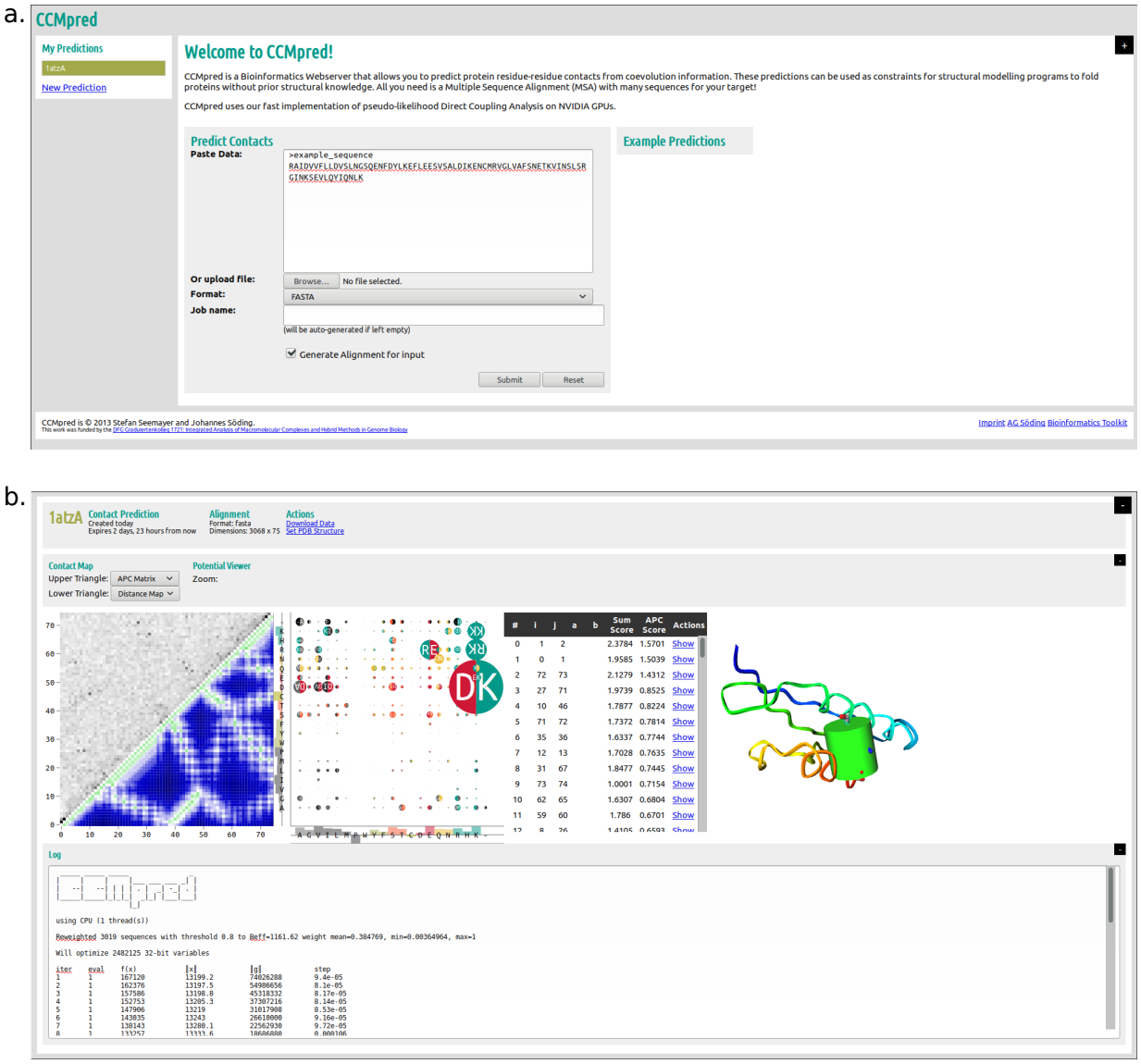


Figure 10.2: CCMpred webserver user interface. **a.** Job submission page. The user can paste or upload a protein sequence or multiple sequence alignment and the specified sequence will automatically enriched with homologous sequences before the actual contact prediction begins. **b.** An example prediction page. When a user clicks on an (i, j) pair in the contact map viewer, the corresponding $w_{i,j}(a, b)$ coupling matrix is loaded into the coupling matrix viewer and the contact highlighted on a user-supplied 3D structure (if available). The log viewer shows the CCMpred run protocolling the pseudo-likelihood optimization and convergence.

Chapter 11

Conclusion

Modern interactive visualization techniques and minimizing the time passed between the user requesting information and the response appearing allow structural biologists and protein engineering researchers interested in applying evolutionary coupling methods in their studies of a protein family to focus on the biological questions they want to answer instead of concentrating on how a tool is correctly used. By following these principles, the CCMpred webserver provides an intuitive and user-friendly tool to explore the topic of evolutionary coupling and its application in protein folding and has been successfully used in collaborations with structural biology groups. Furthermore, the visualizations convey an intuitive understanding of the coupling potentials learned from multiple sequence alignments and are useful in debugging and improving contact prediction methods further.

The visualization components developed in this work could be useful for developers of other websites so a valuable further step would be to package up individual visualization components as BioJS components.

Part IV

Generating Protein Sequences from Couplings

Chapter 12

Introduction to Computational Protein Design

Computational protein design is the part of protein engineering that uses computational methods to predict protein sequences that will fold into a desired fold and exhibit a desired activity [94]. Depending on the goal of the study, protein design can be employed to optimize or change ligand specificity [95], catalyze reactions [96], prefer one of several alternative conformations [97], or, most typically, to maximize the thermostability [98].

The task of finding an optimal protein sequence can be seen as a complex optimization problem where the search space is made up of the space of possible amino acid sequences plus the possible conformations of their side chains. While the search space is already enormous when considering only these two factors, flexible-backbone models have been used more recently to further expand the dimensionality of the search space.

By using empirical and knowledge-based force field models that were further optimized for protein design tasks, a free energy "score" can be attached to a given configuration of sequence and conformation. While the search space cannot be fully enumerated for reasonably-sized proteins, discretizing side-chain conformations [99] combined with Monte Carlo-based [100] or genetic algorithms [101] for traversing the search space efficiently make it possible to stochastically explore meaningful parts of the search space and at least find local optima. For small proteins, *Dead End Elimination* (DEE) [102] can be used to prune impossible areas of the search space to arrive at a global optimum.

Through Monte Carlo optimizations of a library of 108 protein structures, Kuhlmann and Baker showed that 51% of lowest free energy-model residues in the protein core were identical to the residues appearing at these positions in nature [103], indicating that the stochastic exploration of sequence space by nature has already done a remarkable job at optimizing protein stability. However, by introducing artificial selection pressures that would not be observed in nature, protein design can help optimize proteins to even better fit the needs of humanity.

12.1 Covariation in Protein Design

Covariation has been previously used in protein design applications in the framework of *protein sectors* by Rama Ranganthan and colleagues. Using covariance statistics as a clustering similarity measure, proteins can be dissected into allosterically interacting groups of residues that are mostly biochemically independent from other sectors [104, 66]. An artificial protein MSA was sampled from a Monte Carlo algorithm to match the covariance properties of a MSA of real-world WW domain and proteins from the artificial MSA expressed. While mean sequence identity between artificial and natural sequences was low at 36% as expected when sampling using per-column amino acid frequencies, the artificial MSA sequences taking covariation into account folded into the native structure in 28% of experiments when taking covariation into account while natural sequences folded into the native structure 67% of times and protein sequences taking only single-column frequencies into account would not fold into the native structure [105].

Another application of covariation in a protein context is the work of Ollikainen and Kortemme [106]. Using 40 diverse protein domain MSAs and corresponding structures, ensembles of alternative backbone and side-chain conformations were generated and then low-energy sequences were sampled using the Monte Carlo-based Rosetta toolkit. APC-corrected mutual information covariance scores were calculated for both the natural MSAs and low-energy sequences obtained from protein design. Comparing the covariance scores of natural and designed sequences show significant overlap between covariances that is highest when using intermediate levels of backbone flexibility in the protein design process. On the amino acid level, similar amino acid pairs are found to be covarying in natural and designed sequences, except for interactions not modelled by the energy function underlying the protein design such as cation-pi interactions.

12.2 Markov Chain Monte Carlo Algorithms

The Markov Chain Monte Carlo (MCMC) class of algorithms allow the efficient traversal of high-dimensional probability distributions with local minima through stochastic processes and consequently enjoy widespread application in protein design approaches. By following along a markov chain that has the same equilibrium distribution as the underlying probability distribution, MCMC algorithms can be used to generate samples from an underlying distribution or empirically compute integrals for the distribution without the computationally expensive calculation of a partition function.

12.2.1 Metropolis-Hastings and Hybrid Monte Carlo Algorithms

Within the class of MCMC algorithms, the metropolis-hastings algorithm is a popular choice because it can construct markov chains following a probability distribution with little requirements: It is sufficient to compute a term $p(x|\Theta)$ that is proportional to the true probability distribution $P(x|\Theta)$ but does not need to be normalized [107]. The metropolis-

hastings algorithm first constructs a candidate sample x' by modifying an existing sample x and then accepts the candidate with a probability of α. For $\alpha > 1$, the candidate is always accepted. If the candidate is rejected, the existing sample is emitted as the new sample instead:

$$\alpha = \frac{P(x'|\Theta)}{P(x|\Theta)} = \frac{p(x'|\Theta)}{p(x|\Theta)} \tag{12.1}$$

Using this simple acceptance criterion, the markov chain of emitted samples has been shown to follow the same equilibrium distribution as the underlying probability distribution $P(x|\Theta)$. However, the candidate construction strategy is important to achieve the acceptance rates necessary for efficient optimization.

For probability distributions where a gradient can be computed, the Hybrid Monte Carlo (HMC) or Hamiltonian Monte Carlo method can generate samples with a high acceptance probability [108]. Model parameters Θ are reinterpreted as the position parameters $\vec{q}$ of a virtual particle with uniform mass moving across a probability landscape defined by the model probability distribution and carrying a momentum $\vec{p}$. Together, $\vec{p}$ and $\vec{q}$ form a hamiltonian that can be integrated over time to generate new proposed candidates.

12.2.2 Gibbs Sampling

Another important MCMC algorithm is the Gibbs sampling algorithm. If a probability distribution can be rearranged to calculate marginal probabilities for all Ω possible values of one random variable given all other dimensions $P(x_i|x_1, \ldots, x_L \setminus \{x_i\}, \mathbf{\Theta})$, new samples for all random variables can be efficiently determined using an iterative scheme of S replacements of 100% acceptance probability. When picking a sufficiently high S, the samples produced by the Gibbs sampling algorithm can be considered independent from another. The whole procedure is outlined in Algorithm 12.1.

Algorithm 12.1 Gibbs Sampling

// Input: X^1
for $s \leftarrow 1 \ldots S$ **do**
  // Randomly pick a position to mutate with uniform probability
  $i \leftarrow \text{randomInteger}(\{1 \ldots L\}, P(i) = \frac{1}{L})$
  for $j \leftarrow 1 \ldots L$ **do**
    // Randomly pick a new value at position i from conditional probability
    $$x_j^{s+1} \leftarrow \begin{cases} x_j^s, & i \neq j \\ \text{randomInteger}(\{1 \ldots \Omega\}, P(a) = P(x_i|x_1^s, \ldots x_L^s \setminus \{x_i^s\}, \mathbf{\Theta})), & i = j \end{cases}$$
  end for
end for
// X^{S+1} now contains a newly sampled sequence

Chapter 13

Sampled Protein Sequences from Couplings

The prior works combining protein design and residue-residue covariation that were mentioned in the last chapter show a strong connection between the results of protein design methods and the covariation occurring in nature and measurable by even more old-fashioned contact prediction methods. The results even show that covariation information can sufficiently constrain protein sequences so they can fold into a native structure. Instead of generating sequences using a protein design-centered sampling method, a reasonable experiment would consequently be to draw protein sequences directly from the probability distribution underlying a MRF-based contact prediction, thus simplifying the protein design process by avoiding the dependency on carefully tuned forcefield, backbone flexibility, and sequence search methods. Additionally, such a model can be used to generate synthetic sequence data to further deeper understanding of sequence analysis methods.

This chapter will explain the techniques used for implementing such a covariation-based protein sequence sampling method and will show how they can be used for protein design and sequence analysis applications.

13.1 Gibbs Sampling Sequences from MRFs

The marginal probability distribution for observing an amino acid in a single alignment position given the others as derived in Equation 3.25 can be used in a Gibbs sampling algorithm to efficiently evolve new sequences from a starting sequence. As opposed to the contrastive divergence optimization scenario encountered previously, the Gibbs sampler is continued for higher numbers of substitutions S in proportion to a user-specified phylogenetic distance.

In order to more realistically evolve sequences from common ancestries, the Gibbs sampling procedure was modified to evolve several branching markov chains guided by a phylogenetic tree. For every edge pointing away from an ancestral sequence, Gibbs sampling is started using the ancestral sequence as a starting state and choosing a number

of substitutions S proportional to the length of the current branch. The resultant sampled sequence is annotated to the descendant clade in the tree and will become the ancestral sequence for all of its descendants.

Artificial sequences that were derived from phylogenies were compared to ideal phylogenetic models and models where sequences are sampled independently by choosing different tree topologies. While possible to provide phylogenetic trees that were reconstructed using a phylogenetic tree reconstruction program, perfect binary trees and trees that only consist of one common root node of which all extant sequences are direct descendants (the "star" topology) were studied to better understand the effects of common ancestry on the evolutionary coupling signals.

13.2 Debugging Evolutionary Coupling Methods with Synthetic Sequences

13.2.1 Characterizing Phylogenetic Noise

By varying the evolutionary distance between clades and the evolutionary distance between the extant sequences and their common ancestors, phylogenetic inderdependencies and thus phylogenetic noise can be encoded into the sampled MSA. As seen in Figure 13.1, more sequences in the resultant MSA make the true couplings stand out from the background coupling. Figure 13.2 characterizes the magnitude of phylogenetic noise for different numbers of sequences, evolutionary distances and tree topologies.

Both when increasing the number of sampled sequences and increasing the evolutionary distance of sampling, the amount of variation in the alignment and thus the observed covariation increases, leading to an increase in mean coupling strength. Since additional interdependencies between extant sequences exist for the binary tree phylogeny, the overall variation and the couplings observed in the sampled MSAs are about 30% lower than couplings in MSAs sampled from star-shaped phylogenetic trees that show less interdependency. While both the mean coupling between physically interacting and non-interacting residues increases, the foreground coupling becomes more clearly discernible from the background covariation as evolutionary distance increases.

Looking at only the coupling scores observed for non-contacting residue pairs, the level of background coupling signal stays constant for the binary tree phylogeny when increasing the evolutionary distance covered but increases when sampling from a star-shaped phylogeny. As explained previously in Section 2.2 and Figure 3.2, recent common ancestry leads to sequences that have not fully diverged away into independence. When sampling using a binary tree, extant sequences might have lost their correlation to the ancestral sequence at the root of the binary tree but might still be correlated with the roots of subtrees found further down in the phylogenetic tree.

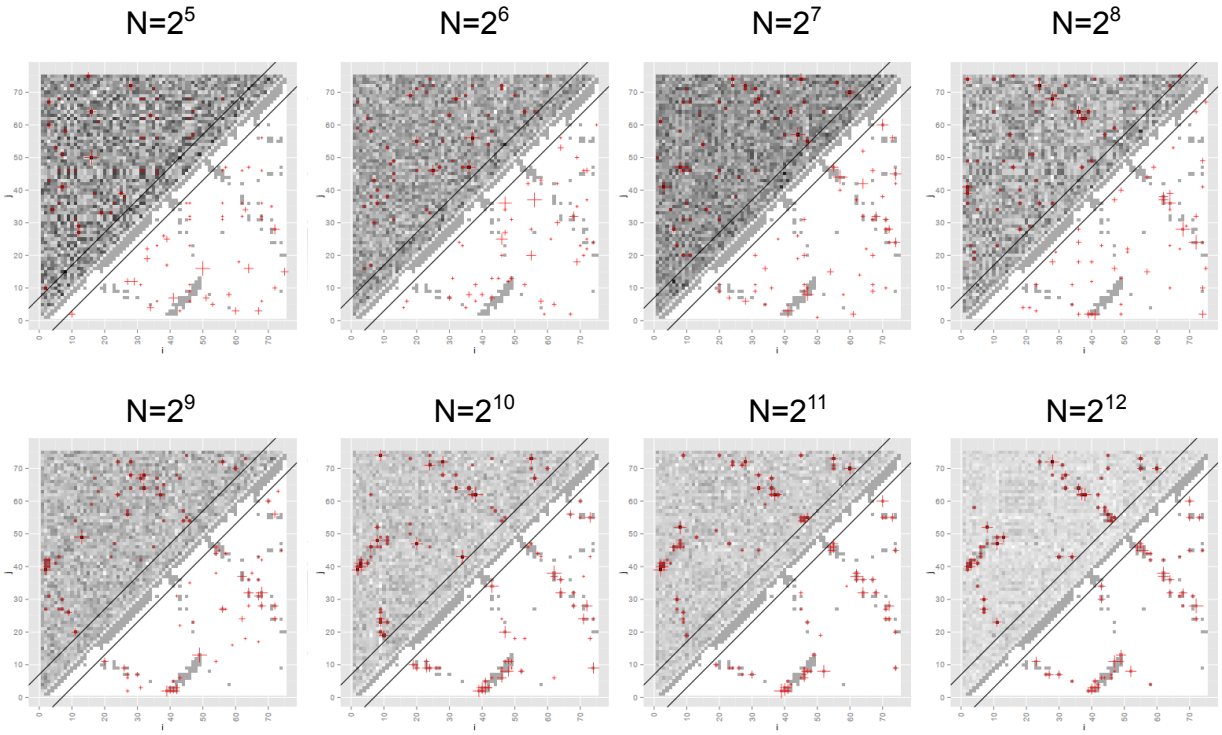


Figure 13.1: Effect of Number of Sequences on Synthetic Alignments. As the number of sequences in the multiple sequence alignment increases, the true residue-residue interactions become more distinguished from the background couplings.

13.2.2 Separating Noise Effects Corrected by APC

Since the sampling strategy allows the creation of protein MSAs with sequences drawn independently from another, the amount of phylogenetic noise present in the contact prediction of a protein family can be disentangled by generating synthetic alignments with and without phylogenetic interdependencies and transforming the contact predictions using APC (which should correct for both entropic and phylogenetic effects) and entropy correction (which should correct only for entropic effects). By comparing the prediction accuracies of both corrections against each other and an uncorrected contact prediction, the magnitude of the different noise sources with respect to their effect on prediction accuracy can be quantified.

However, the sampled alignents do not support the inclusion of gaps and the per-column entropy values were significantly different from the ones calculated for biological alignments, causing entropy corrections to fail to produce reasonable corrections for synthetic alignments. For the lack of time, a further investigation of the shift of entropy terms will have to be done in future work.

13.3 Predicting the Effect of Mutations

Since the joint probability of the MRF can be used to assign probabilities of any protein sequence to be drawn from the MRF probability distribution, the MRF distribution can be used as a statistical potential using the inverse Boltzmann distribution. The Boltzmann distribution $P(X)$ can be solved for $E(X)$, setting $kT = 1$:

$$P(X) = \frac{1}{Z} e^{-\frac{E(X)}{kT}} \tag{13.1}$$

$$\Rightarrow E(X) = -kT \ln P(X) - kT \ln Z \tag{13.2}$$

$$= -\ln P(X) - \ln Z \tag{13.3}$$

The statistical potential formulation can be used to compare the change in statistical potential energy $E(X_0)$ to an alternative state $E(X_1)$:

$$\Delta E = E(X_1) - E(X_0) = \ln P(X_0) - \ln P(X_1) \tag{13.4}$$

Putting this result into the MRF contact prediction framework shows that the unnormalized probability terms can be used to compare the difference in model free energy without computing a normalization term.

An implementation of this strategy was included in the `CCMpred` toolkit and can be used to scan a native protein sequence for ΔE values of all single-, pairwise-, or multi-amino-acid substitutions.

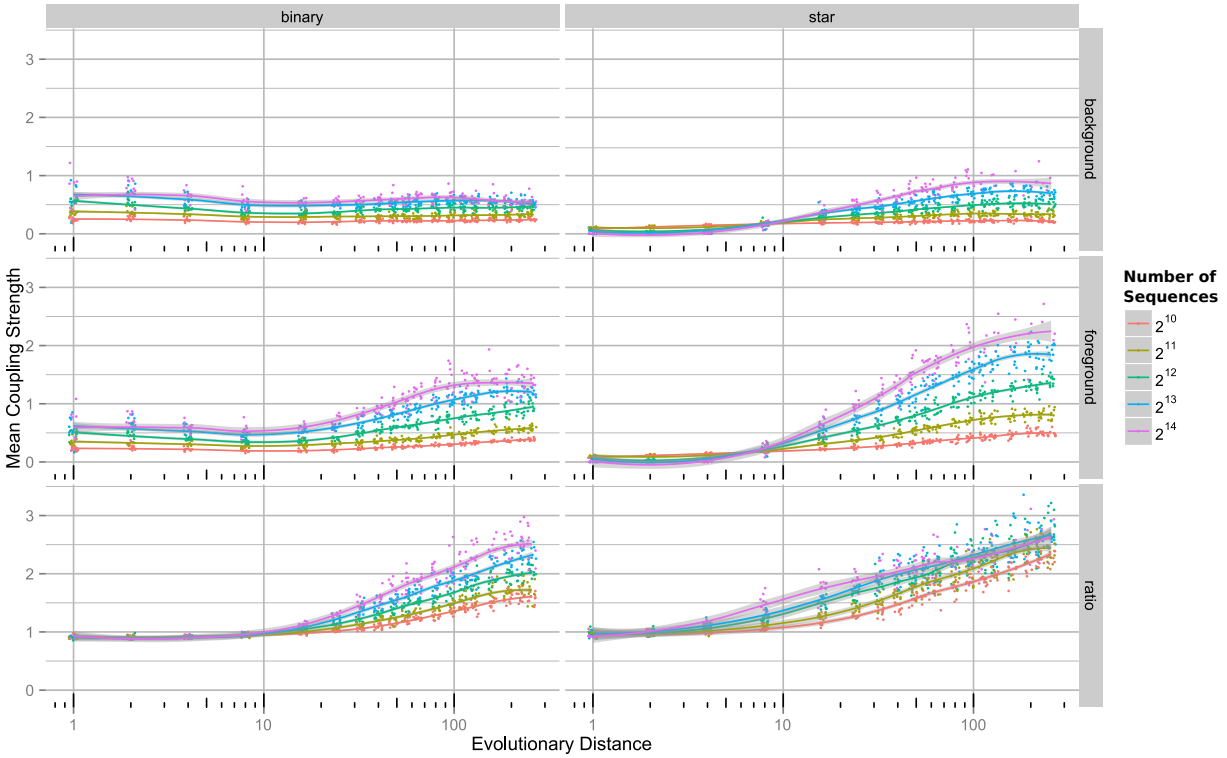


Figure 13.2: Signal-to-Noise Ratio of Contact Prediction determined on Artificial Sequences. For a set of 100 protein families with known physical contacts, artificial multiple sequence alignments were drawn for varying numbers of sequences and evolutionary distances, using five alignments for each family, sequence count and evolutionary distance. As either the number of sequences or the evolutionary distance covered increases, both the mean coupling in non-contacting residues (background) as the mean coupling in contacting residues (foreground) increases, but the foreground coupling increases more quickly than the background coupling. For binary tree topologies, the amount of background coupling is higher for low evolutionary distances due to the phylogenetic noise introduced by common ancestry.

Chapter 14

Conclusion

Evolutionary couplings are a powerful tool for understanding the residue-residue interactions that are important for a protein family under study. The generative model of protein sequences can be used to generate new protein sequence alignments that can even simulate phylogenetic interdependence, or to predict the effect of mutations using the inverse Boltzmann distribution. However, the models learned from protein multiple sequence alignments can only reflect compensatory mutations that have been previously observed, and any interaction that makes biochemical sense but has not been seen before will have their compatibility underestimated.

Since evolutionary coupling methods tap a previously unused source of information for protein design approaches, they can provide an orthogonal source of information to the existing techniques based on structural modeling and could be integrated as an additional energy term in protein design to move the search towards residue-residue interactions that are more likely to be correct. For a proper validation, high-quality experimental data of changes in Gibbs free energy or changes in melting temperature upon single- and multi-site mutations could be used to examine the predictive performance of these energy terms.

Appendix A

Derivation of the Markov Random Field Likelihood Gradients

A.1 Pseudo-Likelihood

The pseudo-log-likelihood of the MRF for a multiple sequence alignment $\boldsymbol{X}$ of N sequences and L columns with coupling parameters $\boldsymbol{v}, \boldsymbol{w}$ is defined as:

$$\begin{aligned}
\mathrm{pll}(\boldsymbol{v}, \boldsymbol{w}|\boldsymbol{X}) &= \log \prod_{n=1}^{N} \prod_{i=1}^{L} p(X_i = x_i^n | (x_1^n, \ldots, x_{i-1}^n, x_{i+1}^n, \ldots x_L^n, \boldsymbol{v}, \boldsymbol{w})) \\
&= \sum_{n=1}^{N} \sum_{i=1}^{L} \log \frac{\exp\left[v_i(x_i^n) + \sum_{\substack{j=1 \\ j \neq i}}^{L} w_{i,j}(x_i^n, x_j^n)\right]}{\sum_{c=1}^{21} \exp\left[v_i(c) + \sum_{\substack{j=1 \\ j \neq i}}^{L} w_{i,j}(c, x_j^n)\right]} \\
&= \sum_{n=1}^{N} \sum_{i=1}^{L} \left[v_i(x_i^n) + \sum_{\substack{j=1 \\ j \neq i}}^{L} w_{i,j}(x_i^n, x_j^n) - \log Z_i^n \right]
\end{aligned} \tag{A.1}$$

With the *partition function* normalization term Z_i^n:

$$Z_i^n = \sum_{c=1}^{20} \exp\left[v_i(c) + \sum_{\substack{j=1 \\ j \neq i}}^{L} w_{i,j}(c, x_j^n) \right] \tag{A.2}$$

The derivative for single-column emission potentials is:

$$\frac{\partial \operatorname{pll}(\boldsymbol{v}, \boldsymbol{w}|\boldsymbol{X})}{\partial v_i(a)} = \sum_{n=1}^{N}\sum_{i'=1}^{L}\left[I(i'=i, x_{i'}^n = a) - \frac{\sum_{c=1}^{20}\left(\exp\left[v_{i'}(c) + \sum_{\substack{j'=1\\ j'\neq i'}}^{L} w_{i'j'}(c, x_{j'}^n)\right] I(i'=i, c=a)\right)}{\sum_{c=1}^{20}\exp\left[v_{i'}(c) + \sum_{\substack{j'=1\\ j'\neq i'}}^{L} w_{i'j'}(c, x_{j'}^n)\right]}\right] = \tag{A.3}$$

$$= \sum_{n=1}^{N}\left[I(x_i^n = a) - \frac{\exp\left[v_i(a) + \sum_{\substack{j'=1\\ j'\neq i}}^{L} w_{ij'}(a, x_{j'}^n)\right]}{\sum_{c=1}^{20}\exp\left[v_i(c) + \sum_{\substack{j'=1\\ j'\neq i}}^{L} w_{ij'}(c, x_{j'}^n)\right]}\right] \tag{A.4}$$

$$= N(x_i = a) - \sum_{n=1}^{N}\frac{\exp\left[v_i(a) + \sum_{\substack{j'=1\\ j'\neq i}}^{L} w_{ij'}(a, x_{j'}^n)\right]}{\sum_{c=1}^{20}\exp\left[v_i(c) + \sum_{\substack{j'=1\\ j'\neq i}}^{L} w_{ij'}(c, x_{j'}^n)\right]} \tag{A.5}$$

When deriving the pairwise emission potentials, it is important to note the symmetry of the pairwise emission potentials: $w_{ij}(a,b) = w_{ji}(b,a)$. The derivative for pairwise emission potentials is:

$$\frac{\partial \operatorname{pll}(\boldsymbol{v}, \boldsymbol{w}|\boldsymbol{X})}{\partial w_{ij}(a,b)} = N(x_i^n = a, x_j^n = b) - \sum_{n=1}^{N}\frac{\exp\left[v_i(a) + \sum_{\substack{j'=1\\ j'\neq i}}^{L} w_{ij'}(a, x_{j'}^n)\right]}{\sum_{c=1}^{20}\exp\left[v_i(c) + \sum_{\substack{j'=1\\ j'\neq i}}^{L} w_{ij'}(c, x_{j'}^n)\right]} + \tag{A.6}$$

$$+ N(x_i^n = b, x_j^n = a) - \sum_{n=1}^{N}\frac{\exp\left[v_j(b) + \sum_{\substack{i'=1\\ i'\neq j}}^{L} w_{ji'}(b, x_{i'}^n)\right]}{\sum_{c=1}^{20}\exp\left[v_j(c) + \sum_{\substack{i'=1\\ i'\neq j}}^{L} w_{ji'}(c, x_{i'}^n)\right]} \tag{A.7}$$

A.2 Full Likelihood

The log-likelihood of the MRF for a multiple sequence alignment $\boldsymbol{X}$ of N sequences and L columns with coupling parameters $\boldsymbol{v}, \boldsymbol{w}$ is defined as:

$$\operatorname{ll}(\boldsymbol{v}, \boldsymbol{w}|\boldsymbol{X}) = \sum_{n=1}^{N}\left[\sum_{i=1}^{L} v_i(x_i^n) + \sum_{\substack{i,j=1\\ i\neq j}}^{L} w_{ij}(x_i^n, x_j^n) - \log Z\right] \tag{A.8}$$

With the partition function Z:

$$Z = \sum_{x' \in \{1...20\}^L} \exp\left[\sum_{i=1}^{L} v_i(x'_i) + \sum_{\substack{i,j=1 \\ i \neq j}}^{L} w_{ij}(x'_i, x'_j)\right] \tag{A.9}$$

The single-column emission potential gradient is:

$$\frac{\partial \, \mathrm{ll}(\boldsymbol{v}, \boldsymbol{w} | \boldsymbol{X})}{\partial v_i(a)} = \sum_{n=1}^{N} \left[I(x_i^n = a) - \frac{\sum_{\substack{x' \in \{1...20\}^L \\ x'_i = a}} \exp\left[v_i(a) + \sum_{\substack{j'=1 \\ j' \neq i}}^{L} w_{ij'}(a, x'_{j'})\right]}{\sum_{x' \in \{1...20\}^L} \exp\left[\sum_{i'=1}^{L} v_{i'}(x'_{i'}) + \sum_{\substack{i',j'=1 \\ i' \neq j'}}^{L} w_{i',j'}(x'_{i'}, x'_{j'})\right]} \right] = \tag{A.10}$$

$$= N(x_i = a) - NP(x_i = a | \boldsymbol{v}, \boldsymbol{w}) \tag{A.11}$$

The pairwise emission potential gradient is:

$$\frac{\partial \, \mathrm{ll}(\boldsymbol{v}, \boldsymbol{w} | \boldsymbol{X})}{\partial w_{ij}(a, b)} = \sum_{n=1}^{N} \Bigg[I(x_i^n = a, x_j^n = b) - \frac{\sum_{\substack{x' \in \{1...20\}^L \\ x'_i = a, x'_j = b}} \exp\left[v_i(a) + \sum_{\substack{j'=1 \\ j' \neq i}}^{L} w_{ij'}(a, x'_{j'})\right]}{\sum_{x' \in \{1...20\}^L} \exp\left[v_i(a) + \sum_{\substack{j'=1 \\ j' \neq i}}^{L} w_{ij'}(a, x'_{j'})\right]} + \tag{A.12}$$

$$+ I(x_j^n = a, x_i^n = b) - \frac{\sum_{\substack{x' \in \{1...20\}^L \\ x'_j = a, x'_i = b}} \exp\left[v_i(a) + \sum_{\substack{j'=1 \\ j' \neq i}}^{L} w_{ij'}(a, x'_{j'})\right]}{\sum_{x' \in \{1...20\}^L} \exp\left[v_i(a) + \sum_{\substack{j'=1 \\ j' \neq i}}^{L} w_{ij'}(a, x'_{j'})\right]} \Bigg] = \tag{A.13}$$

$$= N(x_i = a, x_j = b) - NP(x_i = a, x_j = b | \boldsymbol{v}, \boldsymbol{w}) + \tag{A.14}$$

$$+ N(x_j = a, x_i = b) - NP(x_j = a, x_i = b | \boldsymbol{v}, \boldsymbol{w}) \tag{A.15}$$

Bibliography

[1] Kendrew, JC, Bodo, G, Dintzis, HM, Parrish, RG, Wyckoff, H, and Phillips, DC. A three-dimensional model of the myoglobin molecule obtained by x-ray analysis. *Nature*, 181(4610):662–666, 1958. ISSN 0028-0836. doi:10.1038/181662a0.

[2] Wlodawer, A, Minor, W, Dauter, Z, and Jaskolski, M. Protein crystallography for non-crystallographers, or how to get the best (but not more) from published macromolecular structures. *FEBS Journal*, 275(1):1–21, 2008. ISSN 1742464X. doi: 10.1111/j.1742-4658.2007.06178.x.

[3] Kwan, AH, Mobli, M, Gooley, PR, King, GF, and MacKay, JP. Macromolecular NMR spectroscopy for the non-spectroscopist. *FEBS Journal*, 278(5):687–703, 2011. ISSN 1742464X. doi:10.1111/j.1742-4658.2011.08004.x.

[4] Milne, JLS, Borgnia, MJ, Bartesaghi, A, Tran, EEH, Earl, LA, Schauder, DM, Lengyel, J, Pierson, J, Patwardhan, A, and Subramaniam, S. Cryo-electron microscopy - A primer for the non-microscopist. *FEBS Journal*, 280(1):28–45, 2013. ISSN 1742464X. doi:10.1111/febs.12078.

[5] Smith, DL and Zhang, ZQ. Probing Noncovalent Structural Features of Proteins by Mass-Spectrometry. *Mass Spectrometry Reviews*, 13(5-6):411–429, 1994. ISSN 1098-2787. doi:Doi10.1002/Mas.1280130503.

[6] PDB Current Holdings Breakdown.
URL `http://www.rcsb.org/pdb/statistics/holdings.do`

[7] Berman, HM. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, jan 2000. ISSN 13624962.

[8] Carpenter, EP, Beis, K, Cameron, AD, and Iwata, S. Overcoming the challenges of membrane protein crystallography. *Current Opinion in Structural Biology*, 18(5):581–586, 2008. ISSN 0959440X. doi:10.1016/j.sbi.2008.07.001.

[9] Eshaghi, S, Hedrén, M, Nasser, MIA, Hammarberg, T, Thornell, A, and Nordlund, P. An efficient strategy for high-throughput expression screening of recombinant integral membrane proteins. *Protein science : a publication of the Protein Society*, 14(3):676–83, 2005. ISSN 0961-8368. doi:10.1110/ps.041127005.

[10] Landau, EM and Rosenbusch, JP. Lipidic cubic phases: a novel concept for the crystallization of membrane proteins. *Proceedings of the National Academy of Sciences of the United States of America*, 93(25):14532–14535, 1996. ISSN 00278424. doi:10.1073/pnas.93.25.14532.

[11] Bai, XC, McMullan, G, and Scheres, SHW. How cryo-EM is revolutionizing structural biology. *Trends in Biochemical Sciences*, 40(1):49–57, 2014. ISSN 13624326. doi: 10.1016/j.tibs.2014.10.005.

[12] Rost, B and Sander, C. Bridging the Protein Sequence-Structure Gap by Structure Predictions. *Annual Review of Biophysics and Biomolecular Structure*, 25(1):113–136, jun 1996. ISSN 1056-8700. doi:10.1146/annurev.bb.25.060196.000553.

[13] Chandonia, JM and Brenner, SE. The impact of structural genomics: expectations and outcomes. *Science (New York, N.Y.)*, 311(5759):347–351, 2006. ISSN 0036-8075. doi:10.1126/science.1121018.

[14] Chothia, C and Lesk, AM. The relation between the divergence of sequence and structure in proteins. *The EMBO journal*, 5(4):823–6, 1986. ISSN 0261-4189. doi: 060fehlt.

[15] Hubbard, TJP, Murzin, AG, Brenner, SE, and Chothia, C. SCOP: a Structural Classification of Proteins database. *Nucleic Acids Research*, 25(1):236–239, jan 1997. ISSN 0305-1048. doi:10.1093/nar/25.1.236.

[16] Orengo, C, Michie, A, Jones, S, Jones, D, Swindells, M, and Thornton, J. CATH - a hierarchic classification of protein domain structures. *Structure*, 17(March):1093–1109, 1997. ISSN 09692126. doi:10.1016/S0969-2126(97)00260-8.

[17] Harrison, A, Pearl, F, Mott, R, Thornton, J, and Orengo, C. Quantifying the similarities within fold space. *Journal of Molecular Biology*, 323(5):909–926, 2002. ISSN 00222836. doi:10.1016/S0022-2836(02)00992-0.

[18] Martí-Renom, MA, Stuart, AC, Fiser, A, Sánchez, R, Melo, F, and Šali, A. Comparative Protein Structure Modeling of Genes and Genomes. *Annual Review of Biophysics and Biomolecular Structure*, 29(1):291–325, jun 2000. ISSN 1056-8700. doi:10.1146/annurev.biophys.29.1.291.

[19] Webb, B and Sali, A. Comparative Protein Structure Modeling Using MODELLER. *Current Protocols in Bioinformatics*, (June):1–37, 2002. ISSN 1934-340X. doi:10.1002/0471250953.bi0506s47.

[20] Modi, V, Xu, Q, Adhikari, S, and Dunbrack, RL. Assessment of template-based modeling of protein structure in CASP11. *Proteins*, (April):1–21, 2016. ISSN 1097-0134. doi:10.1002/prot.25049.

[21] Rost, B. Twilight zone of protein sequence alignments. *Protein Engineering Design and Selection*, 12(2):85–94, feb 1999. ISSN 1741-0126. doi:10.1093/protein/12.2.85.

[22] Remmert, M, Biegert, A, Hauser, A, and Söding, J. HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature Methods*, 9(2):173–175, dec 2012. ISSN 1548-7091. doi:10.1038/nmeth.1818.

[23] Kim, P and Baldwin, R. Intermediates In The Folding Reactions Of Small Proteins. *Annual Review of Biochemistry*, 59(1):631–660, 1990. ISSN 00664154. doi:10.1146/annurev.biochem.59.1.631.

[24] O'Meara, MJ, Leaver-Fay, A, Tyka, MD, Stein, A, Houlihan, K, Dimaio, F, Bradley, P, Kortemme, T, Baker, D, Snoeyink, J, and Kuhlman, B. Combined covalent-electrostatic model of hydrogen bonding improves structure prediction with Rosetta. *Journal of Chemical Theory and Computation*, 11(2):609–622, 2015. ISSN 15499626. doi:10.1021/ct500864r.

[25] Lazaridis, T and Karplus, M. Effective energy function for proteins in solution. *Proteins: Structure, Function and Genetics*, 35(2):133–152, 1999. ISSN 08873585. doi:10.1002/(SICI)1097-0134(19990501)35:2⟨133::AID-PROT1⟩3.0.CO;2-N.

[26] Dunbrack, RL. Rotamer libraries in the 21st century. *Current Opinion in Structural Biology*, 12(4):431–440, 2002. ISSN 0959440X. doi:10.1016/S0959-440X(02)00344-5.

[27] Simons, KT, Kooperberg, C, Huang, E, and Baker, D. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *Journal of molecular biology*, 268(1):209–225, 1997. ISSN 0022-2836. doi:10.1006/jmbi.1997.0959.

[28] Jones, DT. Predicting novel protein folds by using FRAGFOLD. *Proteins: Structure, Function and Genetics*, 45(SUPPL. 5):127–132, 2001. ISSN 08873585. doi:10.1002/prot.1171.

[29] Qian, B, Raman, S, Das, R, Bradley, P, McCoy, AJ, Read, RJ, and Baker, D. High-resolution structure prediction and the crystallographic phase problem. *Nature*, 450(7167):259–264, 2007. ISSN 0028-0836. doi:10.1038/nature06249.

[30] Havel, TF, Crippen, GM, and Irwin, D. Effects of Distance Constraints on Macromolecular Conformation . 11 . Simulation of Experimental Results and Theoretical Predictions. *Biopolymers*, 18:73–81, 1979. ISSN 0006-3525. doi:10.1002/bip.1979.360180108.

[31] Tress, ML and Valencia, A. Predicted residue-residue contacts can help the scoring of 3D models. *Proteins*, 78(8):1980–91, jun 2010. ISSN 1097-0134. doi:10.1002/prot.22714.

[32] Li, W, Zhang, Y, and Skolnick, J. Application of sparse NMR restraints to large-scale protein structure prediction. *Biophysical journal*, 87(2):1241–1248, 2004. ISSN 00063495. doi:10.1529/biophysj.104.044750.

[33] Sinz, A. Chemical cross-linking and mass spectrometry for mapping three-dimensional structures of proteins and protein complexes. *Journal of Mass Spectrometry*, 38(12):1225–1237, dec 2003. ISSN 1076-5174. doi:10.1002/jms.559.

[34] Jeener, J, Meier, BH, Bachmann, P, and Ernst, RR. Investigation of exchange processes by two-dimensional NMR spectroscopy. *Journal of Chemical Physics*, 71(11):4546–4553, 1979. ISSN 00219606 (ISSN). doi:10.1063/1.438208.

[35] Dunn, SD, Wahl, LM, and Gloor, GB. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. *Bioinformatics*, 24(3):333–40, feb 2008. ISSN 1367-4811. doi:10.1093/bioinformatics/btm604.

[36] Weigt, M, White, RA, Szurmant, H, Hoch, JA, and Hwa, T. Identification of direct residue contacts in protein-protein interaction by message passing. *PNAS*, 106(1):67–72, jan 2009. ISSN 1091-6490. doi:10.1073/pnas.0805923106.

[37] Marks, DS, Colwell, LJ, Sheridan, R, Hopf, TA, Pagnani, A, Zecchina, R, and Sander, C. Protein 3D structure computed from evolutionary sequence variation. *PloS one*, 6(12):e28766, jan 2011. ISSN 1932-6203. doi:10.1371/journal.pone.0028766.

[38] Fodor, AA and Aldrich, RW. Influence of conservation on calculations of amino acid covariance in multiple sequence alignments. *Proteins*, 56(2):211–21, aug 2004. ISSN 1097-0134. doi:10.1002/prot.20098.

[39] Martin, LC, Gloor, GB, Dunn, SD, and Wahl, LM. Using information theory to search for co-evolving residues in proteins. *Bioinformatics (Oxford, England)*, 21(22):4116–24, nov 2005. ISSN 1367-4803. doi:10.1093/bioinformatics/bti671.

[40] Lapedes, AS, Giraud, BG, Liu, L, and Stormo, GD. Correlated Mutations in Models of Protein Sequences : Phylogenetic and Structural Effects. Technical Report 1999, Institute of Mathematical Statistics, 1999.

[41] Balakrishnan, S, Kamisetty, H, Carbonell, JG, Lee, SI, and Langmead, CJ. Learning generative models for protein fold families. *Proteins*, 79(4):1061–78, apr 2011. ISSN 1097-0134. doi:10.1002/prot.22934.

[42] Jones, DT, Buchan, DW, Cozzetto, D, and Pontil, M. PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. *Bioinformatics*, 28(2):184–90, jan 2012. ISSN 1367-4811. doi:10.1093/bioinformatics/btr638.

[43] Morcos, F, Pagnani, A, Lunt, B, Bertolino, A, Marks, DS, Sander, C, Zecchina, R, Onuchic, JN, Hwa, T, and Weigt, M. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *PNAS*, 108(49):E1293–301, dec 2011. ISSN 1091-6490. doi:10.1073/pnas.1111471108.

[44] Hopf, TA, Colwell, LJ, Sheridan, R, Rost, B, Sander, C, and Marks, DS. Three-Dimensional Structures of Membrane Proteins from Genomic Sequencing. *Cell*, 149(7):1607–1621, may 2012. ISSN 00928674. doi:10.1016/j.cell.2012.04.012.

[45] Dos Santos, RN, Morcos, F, Jana, B, Andricopulo, AD, and Onuchic, JN. Dimeric interactions and complex formation using direct coevolutionary couplings. *Scientific reports*, 5:13652, jan 2015. ISSN 2045-2322.

[46] Gidas, B. Consistency of Maximum Likelihood and Pseudo-Likelihood Estimators for Gibbs Distributions. In Fleming, W and Lions, PL (editors), *Stochastic Differential Systems, Stochastic Control Theory and Applications*, pages 129–145. Springer New York, New York, NY, 1988. ISBN 978-1-4613-8762-6. doi:10.1007/978-1-4613-8762-6_10.

[47] Monastyrskyy, B, D'Andrea, D, Fidelis, K, Tramontano, A, and Kryshtafovych, A. New encouraging developments in contact prediction: Assessment of the CASP11 results. *Proteins*, oct 2015. ISSN 1097-0134. doi:10.1002/prot.24943.

[48] Brunger, AT. Version 1.2 of the Crystallography and NMR system. *Nature protocols*, 2(11):2728–33, 2007. ISSN 1750-2799. doi:10.1038/nprot.2007.406.

[49] Adhikari, B, Bhattacharya, D, Cao, R, and Cheng, J. CONFOLD: Residue-residue contact-guided ab initio protein folding. *Proteins*, 83(8):1436–49, aug 2015. ISSN 1097-0134.

[50] Rohl, CA, Strauss, CEM, Misura, KMS, and Baker, D. Protein Structure Prediction Using Rosetta. *Methods in Enzymology*, 383(2003):66–93, 2004. ISSN 00766879. doi:10.1016/S0076-6879(04)83004-0.

[51] Michel, M, Hayat, S, Skwark, MJ, Sander, C, Marks, DS, and Elofsson, A. PconsFold: Improved contact predictions improve protein models. *Bioinformatics*, 30(17):482–488, 2014. ISSN 14602059. doi:10.1093/bioinformatics/btu458.

[52] Braun, T, Koehler Leman, J, and Lange, OF. Combining Evolutionary Information and an Iterative Sampling Strategy for Accurate Protein Structure Prediction. *PLoS computational biology*, 11(12):e1004661, dec 2015. ISSN 1553-7358. doi:10.1371/journal.pcbi.1004661.

[53] Margara, L, Vassura, M, Di Lena, P, Medri, F, Fariselli, P, and Casadio, R. Reconstruction of the protein structures from contact maps. 5(3):1–18, 2006.

[54] Vassura, M, Margara, L, Di Lena, P, Medri, F, Fariselli, P, and Casadio, R. Fault Tolerance for Large Scale Protein 3D Reconstruction from Contact Maps. In Giancarlo, R and Hannenhalli, S (editors), *Algorithms in Bioinformatics: 7th International Workshop, WABI 2007*, chapter Fault Tole, pages 25–37. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-74126-8. doi:10.1007/978-3-540-74126-8_4.

[55] Fariselli, P, Olmea, O, Valencia, A, and Casadio, R. Prediction of contact maps with neural networks and correlated mutations. *Protein engineering*, 14(11):835–43, nov 2001. ISSN 0269-2139.

[56] Pietal, MJ, Bujnicki, JM, and Kozlowski, LP. GDFuzz3D: A method for protein 3D structure reconstruction from contact maps, based on a non-Euclidean distance function. *Bioinformatics*, 31(21):3499–3505, 2014. ISSN 14602059. doi:10.1093/bioinformatics/btv390.

[57] Kruskal, JB. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964. ISSN 00333123. doi:10.1007/BF02289565.

[58] Cheng, J, Randall, AZ, Sweredoski, MJ, and Baldi, P. SCRATCH: A protein structure and structural feature prediction server. *Nucleic Acids Research*, 33(SUPPL. 2):72–76, 2005. ISSN 03051048. doi:10.1093/nar/gki396.

[59] Taylor, TJ, Bai, H, Tai, CH, and Lee, B. Assessment of CASP10 contact-assisted predictions. *Proteins: Structure, Function and Bioinformatics*, 82(SUPPL.2):84–97, 2014. ISSN 08873585. doi:10.1002/prot.24367.

[60] Kinch, LN, Li, W, Monastyrskyy, B, Kryshtafovych, A, and Grishin, NV. Evaluation of free modeling targets in CASP11 and ROLL. *Proteins: Structure, Function and Bioinformatics*, (September), 2016. ISSN 10970134. doi:10.1002/prot.24973.

[61] Monastyrskyy, B, Fidelis, K, Tramontano, A, and Kryshtafovych, A. Evaluation of residue-residue contact predictions in CASP9. *Proteins: Structure, Function, and Bioinformatics*, 79(S10):119–125, 2011. ISSN 08873585. doi:10.1002/prot.23160.

[62] Monastyrskyy, B, D'Andrea, D, Fidelis, K, Tramontano, A, and Kryshtafovych, A. Evaluation of residue-residue contact prediction in CASP10. *Proteins: Structure, Function, and Bioinformatics*, 82(0 2):138–153, feb 2014. ISSN 08873585. doi:10.1002/prot.24340.

[63] Kamisetty, H, Ovchinnikov, S, and Baker, D. Assessing the utility of coevolution-based residue-residue contact predictions in a sequence- and structure-rich era. *PNAS*, 110(39):15674–15679, sep 2013. doi:10.1073/pnas.1314045110.

[64] Sillitoe, I, Lewis, TE, Cuff, A, Das, S, Ashford, P, Dawson, NL, Furnham, N, Laskowski, RA, Lee, D, Lees, JG, Lehtinen, S, Studer, RA, Thornton, J, and Orengo,

CA. CATH: Comprehensive structural and functional annotations for genome sequences. *Nucleic Acids Research*, 43(D1):D376–D381, 2015. ISSN 13624962. doi: 10.1093/nar/gku947.

[65] Price, MN, Dehal, PS, and Arkin, AP. FastTree 2 - Approximately maximum-likelihood trees for large alignments. *PLoS ONE*, 5(3), 2010. ISSN 19326203. doi: 10.1371/journal.pone.0009490.

[66] Halabi, N, Rivoire, O, Leibler, S, and Ranganathan, R. Protein sectors: evolutionary units of three-dimensional structure. *Cell*, 138(4):774–786, 2009. doi:10.1016/j.cell.2009.07.038.

[67] Friedman, J, Hastie, T, and Tibshirani, R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics (Oxford, England)*, 9(3):432–41, jul 2008. ISSN 1468-4357. doi:10.1093/biostatistics/kxm045.

[68] Witten, DM, Friedman, JM, and Simon, N. New Insights and Faster Computations for the Graphical Lasso. *Journal of Computational and Graphical Statistics*, 20(4):892–900, dec 2011. ISSN 1061-8600. doi:10.1198/jcgs.2011.11051a.

[69] Meier, L, Van De Geer, S, and Bühlmann, P. The group lasso for logistic regression. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 70(1):53–71, 2008. ISSN 13697412. doi:10.1111/j.1467-9868.2007.00627.x.

[70] Kowalski, M. Sparse regression using mixed norms. *Applied and Computational Harmonic Analysis*, 27(3):303–324, nov 2009. ISSN 10635203. doi:10.1016/j.acha.2009.05.006.

[71] Yuan, M and Lin, Y. Model selection and estimation in regression with grouped variables. pages 49–67, 2006.

[72] Andrew, G and Gao, J. Scalable Training of L1 -Regularized Log-Linear Models. In *Proceedings of the 24th International Converence on Machine Learning*. Corvallis, OR, 2007.

[73] Henikoff, S and Henikoff, JG. Position-based sequence weights. *Journal of molecular biology*, 243(4):574–8, nov 1994. ISSN 0022-2836.

[74] Altschul, SF, Madden, TL, Schäffer, AA, Zhang, J, Zhang, Z, Miller, W, and Lipman, DJ. Gapped BLAST and PS I-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, 1997. doi:10.1093/nar/25.17.3389.

[75] Gloor, GB, Martin, LC, Wahl, LM, and Dunn, SD. Mutual information in protein multiple sequence alignments reveals two classes of coevolving positions. *Biochemistry*, 44(19):7156–7165, 2005. ISSN 00062960. doi:10.1021/bi050293e.

[76] Hinton, GE. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002. ISSN 0899-7667. doi:10.1162/089976602760128018.

[77] Tieleman, T. Training restricted Boltzmann machines using approximations to the likelihood gradient. *ICML; Vol. 307*, page 7, 2008. doi:10.1145/1390156.1390290.

[78] Felsenstein, J. PHYLIP (Phylogeny Inference Package), 2005.
URL `http://evolution.genetics.washington.edu/phylip/`

[79] Drepper, U. Memory part 2: CPU caches, 2007.
URL `https://lwn.net/Articles/252125/`

[80] JEDEC Solid State Technology Association. DDR4 SDRAM Standard, 2013.
URL `https://www.jedec.org/standards-documents/docs/jesd79-4a`

[81] Samsung. Samsung SSD 850 Pro Data Sheet, Rev. 2.0, 2015.
URL `http://www.samsung.com/global/business/semiconductor/minisite/SSD/downloads/document/Samsung{_}SSD{_}850{_}PRO{_}Data{_}Sheet{_}rev{_}2{_}0.pdf`

[82] Kozierok, CM. Hard Disk Seek Times, 2001.
URL `http://pcguide.com/ref/hdd/perf/perf/spec/posAccess-c.html`

[83] Western Digital WD6002 Data Sheet, 2016.
URL `http://www.wdc.com/wdproducts/library/SpecSheet/ENG/2879-800066.pdf`

[84] NVIDIA. CUDA Toolkit 7.5 Documentation, 2015.
URL `http://docs.nvidia.com/cuda/index.html`

[85] Intel. Intrinsics Guide, 2016.
URL `https://software.intel.com/sites/landingpage/IntrinsicsGuide/{#}`

[86] OpenMP Architecture Review Board. OpenMP.
URL `http://openmp.org/`

[87] Nocedal, J and Wright, SJ. *Numerical optimization*. Springer, 2 edition, feb 2006. ISBN 9780387303031.

[88] Wilkinson, L. *The Grammar of Graphics*. Statistics and Computing. Springer-Verlag, New York, 2005. ISBN 0-387-24544-8. doi:10.1007/0-387-28695-0.

[89] Gomez, J, Garcia, LJ, Salazar, GA, Villaveces, J, Gore, S, Garcia, A, Martin, MJ, Launay, G, Alcantara, R, Del-Toro, N, Dumousseau, M, Orchard, S, Velankar, S, Hermjakob, H, Zong, C, Ping, P, Corpas, M, and Jimenez, RC. BioJS: an open source JavaScript framework for biological data visualization. *Bioinformatics*, 29(8):1103–1104, apr 2013. ISSN 1367-4803. doi:10.1093/bioinformatics/btt100.

[90] Hanson, RM, Prilusky, J, Renjian, Z, Nakane, T, and Sussman, JL. JSmol and the next-generation web-based representation of 3D molecular structure as applied to proteopedia. *Israel Journal of Chemistry*, 53(3-4):207–216, 2013. ISSN 00212148. doi:10.1002/ijch.201300024.

[91] Biasini, M. pv 1.8.1, 2015. doi:10.5281/zenodo.20980.

[92] Django, 2016.
URL `https://djangoproject.com`

[93] Celery: Distributed Task Queue, 2013.
URL `http://www.celeryproject.org/`

[94] Pokala, N and Handel, TM. Review: Protein Design—Where We Were, Where We Are, Where We're Going. *Journal of Structural Biology*, 134(2-3):269–281, 2001. ISSN 10478477. doi:10.1006/jsbi.2001.4349.

[95] Looger, LL, Dwyer, MA, Smith, JJ, and Hellinga, HW. Computational design of receptor and sensor proteins with novel functions. *Nature*, 423(6936):185–190, may 2003. ISSN 0028-0836. doi:10.1038/nature01556.

[96] Lerner, Ra, Benkovic, SJ, and Schultz, PG. At the crossroads of chemistry and immunology: catalytic antibodies. *Science (New York, N.Y.)*, 252(5006):659–667, 1991. ISSN 0036-8075. doi:10.1126/science.2024118.

[97] Shimaoka, M, Shifman, JM, Jing, H, Takagi, J, Mayo, SL, and Springer, TA. Computational design of an integrin I domain stabilized in the open high affinity conformation. *Nature Structural Biology*, 7(8):674–678, 2000. ISSN 1072-8368. doi:10.1038/77978.

[98] Malakauskas, SM and Mayo, SL. Design, structure and stability of a hyperthermophilic protein variant. *Nature structural biology*, 5(6):470–475, 1998. ISSN 1072-8368. doi:10.1038/nsb0698-470.

[99] Dunbrack, RL and Cohen, FE. Bayesian statistical analysis of protein side-chain rotamer preferences. *Protein science : a publication of the Protein Society*, 6(8):1661–81, 1997. ISSN 0961-8368. doi:10.1002/pro.5560060807.

[100] Lee, C and Levitt, M. Accurate prediction of the stability and activity effects of site-directed mutagenesis on a protein core., 1991. doi:10.1038/352448a0.

[101] Tuffery, P, Etchebest, C, Hazout, S, and Lavery, R. A new approach to the rapid determination of protein side chain conformations. *Journal of biomolecular structure & dynamics*, 8(6):1267–89, 1991. ISSN 0739-1102. doi:10.1080/07391102.1991.10507882.

[102] Desmet, J, De Maeyer, M, Hazes, B, and Lasters, I. The dead-end elimination theorem and its use in protein side-chain positioning. *Nature*, 356(6369):539–542, 1992. ISSN 0028-0836. doi:10.1038/356539a0.

[103] Kuhlman, B and Baker, D. Native protein sequences are close to optimal for their structures. *Proceedings of the National Academy of Sciences*, 97(19):10383–10388, 2000. ISSN 0027-8424, 1091-6490. doi:10.1073/pnas.97.19.10383.

[104] Lockless, SW and Ranganathan, R. Evolutionarily conserved pathways of energetic connectivity in protein families. *Science (New York, N.Y.)*, 286(5438):295–9, oct 1999. ISSN 0036-8075.

[105] Socolich, M, Lockless, SW, Russ, WP, Lee, H, Gardner, KH, and Ranganathan, R. Evolutionary information for specifying a protein fold. *Nature*, 437(7058):512–8, sep 2005. ISSN 1476-4687. doi:10.1038/nature03991.

[106] Ollikainen, N and Kortemme, T. Computational Protein Design Quantifies Structural Constraints on Amino Acid Covariation. *PLoS Computational Biology*, 9(11), 2013. ISSN 1553734X. doi:10.1371/journal.pcbi.1003313.

[107] Hastings, WK. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika Vol*, 57:97–109, 1970.

[108] Duane, S, Kennedy, A, Pendleton, BJ, and Roweth, D. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987. ISSN 03702693. doi:10.1016/0370-2693(87)91197-X.

Acknowledgements

Doctoral research is a long and challenging process that I would never have been able to complete on my own. For this reason, I want to take the time to thank the people that have helped me make it to the end.

First, I want to thank Dr. Johannes Söding, for his mentorship and support, for sharing his brilliance, for the opportunity to work on such an exciting project and for being able to work in the welcoming environment and with the wonderful group of people that he has gathered. In the past four years, I've learned more about science, life and myself than in any years before that and I thank Johannes wholeheartedly for creating the opportunity and environment for that growth. My thanks also go to the whole Söding lab, for the friendships and support, the many helpful discussions, feedbacks and inspirations, and making the doctoral studies an enjoyable experience. I will fondly remember our many traditions and hope to carry them with me wherever my steps will take me.

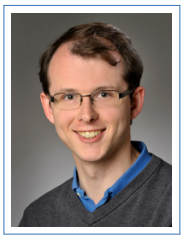
I thank the DFG Graduiertenkolleg 1721 for funding the first half of my doctoral studies and the Max Planck Institute for Biophysical Chemistry for funding the second half. I thank the members of my examination board for their time and insight.

I also want to thank my family for always being there to support me and giving me a sense of belonging. I thank my parents Walter and Luise for their love, for being my role models, giving me both great freedom and an unconditional safety net and teaching me the values that allowed me to succeed. I thank my sister Sofie for being a great friend and teaching me new perspectives.

Finally, I want to thank Adriana, for her love, friendship and support, for being a partner in life and crime, for reminding me to stop, take a breath and enjoy the beauty around me and making me realize that everything is fine and absolutely anything is possible.

Stefan Seemayer

B2, 10
68159 Mannheim
0160 / 2064130
stefan@seemayer.de
stefan.seemayer.de



Personal Information

Date of Birth March 23rd, 1987

Place of Birth Munich, Germany

Experience

since 07.2012 **Doctoral Student**, *Gene Center (Faculty for Chemistry and Pharmacy, LMU Munich), Max-Planck-Institute for Biophysical Chemistry (Göttingen), Söding Lab*.

Estimated completion date: July 2016

- Development of methods for *de novo* protein structure prediction and protein design based on bayesian statistics and insights from structural biology
- Design and implementation of bioinformatics programs, webservers and interactive visualizations
- Bioinformatic support of structural biology projects (data analysis, predictions, etc.)
- Establishing of modern software engineering standards in the research group
- Tutoring of students for their bachelor's and master's theses

04.2012 - 06.2012 **Web Developer**, *Scandio GmbH, Munich, Germany*.

Back- and frontend development for the chambers of industry and commerce (IHK) for Munich and Nuremberg

06.2010 - 12.2011 **Student Assistant**, *Institute of Informatics, Prof. Burkhard Rost*, TU Munich.

- Development of a synchronization and post-processing system for scientific databases (incremental and atomic updates, version control, cleanup and clustering)
- Teaching of the practical course "*The Bioinformatics Lab*" in the area web servers, virtualization and cloud computing

2005 - 2010 **Freelance Web Developer**.

Development of internet and intranet pages as a part-time job. Design, development and maintenance of the website of Dannon Germany GmbH (Haar b. München), www.danone.de from 2007-2009

Education

since 07.2012 **Doctorate Bioinformatics**, *LMU Munich, MPI for Biophysical Chemistry Göttingen*, Estimated completion date: July 2016.

De novo protein structure prediction from evolutionary data, scientific computing, high performance computing, statistical modelling, interactive visualization

06.2010 – 12.2011 **Master of Science Bioinformatics**, *TU/LMU Munich*, final mark 1.7.

Machine learning, protein structure and function prediction, high performance computing

10.2006 – 06.2010 **Bachelor of Science Bioinformatics**, *TU/LMU Munich*, final mark 1.9.

2006 **Abitur**, *Gymnasium Geretsried*, final mark 1.5.

Advanced subjects: English and chemistry. Examination subjects: English, chemistry, math, economics and law

Publications

Theses

Doctoral thesis: Graphical Models for *de novo* Protein Structure Prediction (in preparation), *Gene Center (LMU Munich) and MPI for Biophysical Chemistry (Göttingen)*, Advisor: Dr. Johannes Söding.

Improvement of methods for predicting spatial contacts in proteins through statistical modelling of biological phenomena. Development of software and visualizations.

Master's thesis: Significant Similarities of Small Molecules in 3D, *TU Munich*, Advisor: Prof. Stefan Kramer.

Development of a statistical similarity measure for drug-like molecules using geometric and physical properties.

Bachelor's thesis: Molecular Paths - a Novel Structural Descriptor for Small Molecule Similarity, *TU Munich*, Advisor: Prof. Stefan Kramer.

Workshops

2014-2015 **Protein Structure and Bioinformatics Databases**, LMU Munich, GAU Göttingen, Lecturer: Dr. Johannes Söding.

Preparation and teaching of the practical part for structural biology doctoral students and molecular biology master students (held on three occasions)

Other Interests

Languages German (native), English (fluent), French (basic)

Hobbies Scuba diving (PADI Advanced Open Water Diver), guitar and Ukulele, computer game development (team leader in a volunteer project for 4 years; participation in competitions), organization of cooking and gaming evenings