Rapid Determination of Protein Structures in Solution Using NMR Dipolar Couplings

Dissertation

zur Erlangung des Doktorgrades

der Mathematisch-Naturwissenschaftlichen Fakultäten

der Georg-August-Universität zu Göttingen

vorgelegt von

Young-Sang Jung

aus Pusan, Korea

Göttingen 2005

D7

Referent: Prof. Dr. Reiner Kree

Korreferent: Prof. Dr. Christian Griesigner

Tag der mündlichen Prüfung:

Acknowledgments

First of all, I would like to thank Prof. Dr. Griesinger and Dr. Zweckstetter for giving me opportunity to study in Max-Planck-Institute for Biophysical Chemistry, and their kindness, invaluable advice and continuous support made this thesis possible. I would like to thank Prof. Dr. Kree for his kindness and favor. He agreed to become the referee of my thesis and supported me to do my PhD degree in Physics Department of Göttingen University.

Secondly, I would like to thank Prof. Dr. Hegerfeldt, Prof. Dr. Pruschke, Prof. Parlitz and Junior-Porf. Dr. Meden for participating PhD disputation committee. I would like to thank again to the Prof. Dr. Hegerfeldt for helping me to organize the disputation committee and thank to the Faculty of Physics.

Thirdly, I would like to thank Dr. Junker and Dr. Montaville for the thorough proofreading and Nils for his a lot of help in many ways. In addition, I would like to express my thanks to my friends, Vinesh, Venkatesh, Dirk, Sigrun, Dr. Sanchez, Dr. Farjon, Fernando, Kerstin, Hai-Young, Min-Kyu, Jegannath, Volker, Peter, Monika, Hui, Jörg, Ping, Marco, Devanathan, Marcel, Adam, Carlos and all colleagues at Department of NMR based Structural Biology.

Finally, I would like to address my sincere gratitude to my brother, Woo-Sang Jung, and I would like to show my great thanks to my mother and father, who have been sacrificing everything to their children.

I would like to dedicate this thesis to my father and mother whom I am greatly indebted for their never-ending patience, understanding and love.

Abstract

Once NMR spectra are measured, two main steps follow to determine NMR structure. One is backbone resonance assignment and the other is structure calculation. The both are time-consuming steps. We present program MARS and ITAS to speed up NMR structure determination.

At first, MARS is robust for automatic backbone resonance assignment of ^{13}C/^{15}N labeled proteins. MARS simultaneously optimizes the local and global quality of assignments in order to minimize the propagation of initial assignment errors and to extract reliable assignments. It works with a wide variety of NMR experiments and is robust against missing chemical shift information. Furthermore, a new method was implemented into MARS, which uses sequential connectivity and experimental residual dipolar couplings (RDCs) simultaneously for NMR resonance assignment when structures are available. Assignment was significantly enhanced when experimental RDCs are additionally matched to back-calculated values from a known three-dimensional structure. The combination of sequential connectivity information with RDC-matching allows for more residues to be assigned reliably and backbone assignments to be more robust against missing data.

Secondly, ITAS simultaneously calculates protein structure and assigns the backbone resonances using unassigned chemical shifts and RDCs. Opposite to conventional approaches, where sequential resonance assignment has to be completed prior to structure calculation, partial assignments are used to obtain low-resolution models. These low-resolution models are used to improve the backbone resonance assignment and the improved assignment is again used for structure calculation. Within four to eight iteration steps consisting of automatic

assignment using MARS and structure calculation using RosettaNMR a nearly complete resonance assignment and medium accuracy structures of protein backbones are obtained.

The automation of resonance assignment allows for significant time savings for resonance assignment compared to manual assignment. Furthermore the ITAS automated structure calculation including automatic resonance assignment without any manual intervention avoids another time consuming step.

Organization and Outline of the Thesis

This thesis is composed of seven chapters. Chapters 3, 4, and 5 are the main chapters. They share the same structure (introduction, methods, results and discussion, and concluding remark) and can be read independently without needing continuous cross referring.

The thesis is organized as follows:

Chapter 1 introduces the basic theory of Nuclear Magnetic Resonance (NMR), and the general concept of the multidimensional NMR experiment.

Chapter 2 gives an overview of the NMR structure determination and related terms starting with the NMR experiments, NMR resonance assignment, structure calculation, and ending with automation of the structure calculation.

- In section 2.1, 3D triple-resonance experiments which are the most commonly measured 3D NMR experiments for NMR resonance assignment are explained. It focuses on magnetization transfer and chemical shift evolution to show how and which kind of chemical shift information can be extracted.
- In section 2.2, the terms, which are frequently used in this thesis for NMR resonance assignment, are explained. The order of terms follows the NMR resonance assignment procedure.
- Section 2.3 discusses distance, dihedral angle, and orientational restraints, which are used for structure calculation.

- Section 2.4 describes structure calculation comparing two structure calculation methods.
- Section 2.5 describes the concept of automatic structure calculation explaining the common parts to automatic structure calculation approaches and manual assignment.

Chapter 3 presents the new algorithm for automatic NMR resonance assignment and demonstrates the results of MARS.

- In the introduction, the previously published assignment algorithms are explained shortly and advantages and disadvantages are compared.
- In methods, MARS algorithm is precisely explained.
- In results and discussion, assignment results according to the category of the protein *i.e.* small proteins, partially and completely disordered proteins and big proteins are shown. Then assignment results when considered real assignment situations *i.e.* incomplete chemical shift data, larger sequential connectivity thresholds, missing pseudo-residues, and missing sequential connectivity due to the abnormally large chemical shift deviation between inter- and intra-chemical shifts are shown.
- In concluding remarks, the advantages of MARS are shortly summarized.

Chapter 4 introduces methods to incorporate the algorithm, presented in chapter 3, with the RDCs and known protein structures for enhancing NMR resonance assignment.

- In introduction, the previously published methods, structure and RDCs assisted assignment methods, which cannot use sequential connectivity information simultaneously and don't give indication of the reliability of the assignments are introduced.

- In the methods, it is shown how to implement RDC values into the MARS algorithm and how to get the alignment tensor which is required to calculate RDC values from the structures.
- In results and discussion, enhanced assignments with sequential connectivity information incorporated with only RDC based assignment. It also shows the dependency of RDC-enhanced assignment on the number of types of RDC and dependency of assignment on missing pseudo-residues comparing RDC-enhanced assignment, which make use of both sequential connectivity information and RDC values, and only sequential connectivity based assignment.
- In concluding remarks, the advantages of RDC-enhanced assignment for large proteins and the importance of using structures for assignment are stressed.

Chapter 5 presents ITAS, new the method for simultaneous NMR resonance assignment and protein structure calculation.

- In the introduction, published the automatic structure calculation approaches are introduced. There are two parts. In the first part, it introduces conventional softwares, which require backbone resonance assignment prior to structure calculation, comparing the methods. In the second part, it introduces newly suggested methods, which do not require a prior backbone resonance assignment, comparing their methods.
- In methods, it describes overall procedure of iterative assignment and structure calculation; and in figure 5.1, it shows the overview of the procedure. Then it precisely explains each step *e.g.* automatic resonance assignment using MARS, automatic analysis of assignments, structure calculation by RosettaNMR, and structure refinement by RosettaNMR.
- In results and discussion, it shows assignment percentages and rmsd values between calculated structures and native structures which have different size

starting small protein (56-residue protein) to medium-sized protein (153-residue protein). In the figure 5.2, it shows simultaneous improvements of assignment and structure quality; and it describes structure validation.

- In the concluding remarks, the features of the 'iterative assignment and structure-calculation' approach are explored. It is stressed that the medium-resolution structure is valuable as initial structure for determining 3D high-resolution structures, when additionally inter-atom distance information is available.

Chapter 6 gives the general conclusion of the thesis.

Table of Contents

Page

Appendix

List of Figures

List of Tables

List of Abbreviations

CSA	Chemical shift anisotropy
CBCA(CO)NH	NMR experiment observing peptide ^{15}N, ^{1}H^{N}, ^{13}C$^{\alpha}$ and ^{13}C$^{\beta}$
CBCANH	NMR experiment observing peptide ^{15}N, ^{1}H^{N}, ^{13}C$^{\alpha}$ and ^{13}C$^{\beta}$
COSY	COrrelated SpectroscopY
DG	Distance geometry
rMD	Restrained Molecular Dynamics
FID	Free induction decay
HCACO	NMR experiment observing correlations between peptide H, ^{13}C$^{\alpha}$ and CO
HCCH-COSY	COSY experiment using H-C-C-H magnetization transfer pathway
HCCH-TOCSY	TOCSY experiment using H-C-C-H magnetization transfer pathway
HNCA	NMR experiment observing peptide ^{15}N, ^{1}H^{N} and ^{13}C$^{\alpha}$
HN(CA)CO	NMR experiment observing peptide ^{15}N, ^{1}H^{N} and ^{13}C'
HNCO	NMR experiment observing peptide ^{15}N, ^{1}H^{N} and previous CO
HN(CO)CA	NMR experiment observing peptide ^{15}N, ^{1}H^{N} and previous ^{13}C$^{\alpha}$
HSQC	Heteronuclear single-quantum correlation
NMR	Nuclear Magnetic Resonance
NOE	Nuclear Overhauser Enhancement
NOESY	NOE SpectroscopY

PDB	Protein data bank
PR	Pseudo residue
PAS	Principal axis system
RDC	Residual dipolar coupling
RF	Radio frequency
SA	Simulated annealing
SVD	Singular Value Decomposition

Chapter 1

Background

1.1 Introduction to NMR spectroscopy

Nuclear **M**agnetic **R**esonance (NMR) spectroscopy is one of techniques capable of determining the structures of biological macromolecules like proteins and nucleic acids at atomic resolution. In addition, it is possible to study time dependent phenomena with NMR, such as intramolecular dynamics in macromolecules, reaction kinetics, molecular recognition or protein folding.

The basic phenomenon of NMR was discovered in 1945: The energy levels of atomic nuclei are split up by a magnetic field. Transitions between these energy levels can be induced by exciting the sample with electromagnetic radiation whose frequency is equivalent to the energy difference between the two levels. Since 1960 the field of NMR has seen an explosive growth which started with the development of pulsed Fourier-transform NMR and multidimensional NMR spectroscopy and still continues today.

The limitations of NMR spectroscopy result from the low inherent sensitivity of the technique and from the high complexity and information content of NMR spectra. These problems are partially alleviated by new developments: The sensitivity and resolution of NMR are increased by progress in spectrometer technology. Progress in the theoretical and practical capabilities of NMR lead to a increasingly efficient utilization of the information content of NMR spectra. Parallel developments in the biochemical methods (recombinant protein expression) allow the simple and fast preparation of protein samples. Heteronuclei like ^{15}N, ^{13}C and ^{2}H can be incorporated in proteins by uniformly or selective isotopic labeling. Spectra from these samples can be dramatically simplified. Additionally, new information about structure and dynamics of macromolecules can be determined with these methods.

1.2 Basic Theory of NMR

1.2.1 The Hamiltonians

The nuclear spin Hamiltonian can be written as a sum of internal and external parts:

$$H = H_{int} + H_{ext}. \tag{1.1}$$

With this separation, the effects intrinsic to the spin system are included in the H_{int} Hamiltonian while H_{ext} contains terms due to the experimental setup. The H_{int} can be further subdivided into the basic interactions resulting from the environment of the nucleus:

$$H_{int} = H_{CS} + H_J + H_D + H_Q \tag{1.2}$$

where H_{CS} is the chemical shielding (or chemical shift), H_J is the indirect spin-spin coupling (or J coupling), H_D is the direct dipole-dipole coupling (or dipolar coupling), and H_Q is the quadrupolar coupling. Each of these interactions is intrinsic to the spin system and primarily depends upon the chemical environment of the nucleus. Effects that are a result of actions performed on the spin system are included in the external Hamiltonian, and they can be separated into Zeeman and radio frequency (RF) contributions:

$$H_{ext} = H_z + H_{rf}. \tag{1.3}$$

It is through the Hamiltonian of equation (1.3) that the experimental is able to interact with the spins, and this has been the focus of much of the field of NMR[2]. With a thorough knowledge of the information that is intrinsically available from the internal Hamiltonian of equation (1.2), we can tailor our H_{ext} to extract the desired information. Each of the components of the internal and external Hamiltonians will be described in more detail below. The basic NMR interaction Hamiltonians can be described as the product of vectors $\vec{\mathbf{I}}$ and $\vec{\mathbf{S}}$ with a second rank Cartesian tensors ($\hat{\mathbf{A}}$) which are 3x3 matrices:

$$H = \vec{\mathbf{I}} \cdot \hat{\mathbf{A}} \cdot \vec{\mathbf{S}} = \begin{bmatrix} I_x & I_y & I_z \end{bmatrix} \begin{bmatrix} A_{xx} & A_{xy} & A_{xz} \\ A_{yx} & A_{yy} & A_{yz} \\ A_{zx} & A_{zy} & A_{zz} \end{bmatrix} \begin{bmatrix} S_x \\ S_y \\ S_z \end{bmatrix} \tag{1.4}$$

For example, coupling of the spin I to an external magnetic field can be represented as:

$$H_{0,I} = \vec{\mathbf{I}} \cdot \hat{\mathbf{Z}} \cdot \vec{\mathbf{B}}_0 \tag{1.5}$$

where $\hat{\mathbf{Z}} = \gamma_I \hat{\mathbf{1}}$ and $\vec{\mathbf{B}}_0 = (B_x, B_y, B_z)$.

These second rank Cartesian tensors are represented in the molecular axis system; they can be made diagonal in their principal axis system (PAS) to yield three principal components (A_{11}, A_{22}, A_{33}). Often times in NMR, frame transformations are performed in and out of the PAS to facilitate calculations. This is depicted in Figure 1.1:

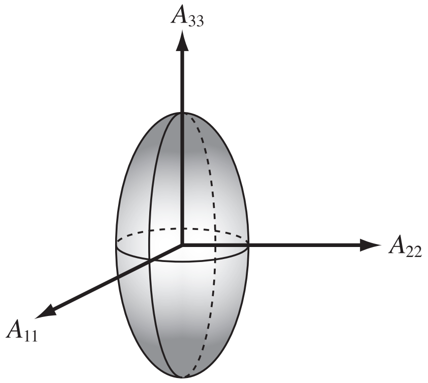

Figure 1.1: Ellipsoid representing a second rank interaction tensor in the principal axis system

$$\hat{\mathbf{H}}_{PAS} = \begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{bmatrix}$$

Table 1.1: Interaction Hamiltonians

Interaction	Hamiltonian
Chemical Shift	$H_{CS} = \gamma\vec{\mathbf{I}} \cdot \hat{\sigma} \cdot \vec{\mathbf{B}}_0$
Dipole-Dipole	$H_D = \frac{\gamma_i\gamma_j\hbar}{r_{ij}^3}\left[\vec{\mathbf{I}}_i \cdot \mathbf{I}_j - \frac{3(\vec{\mathbf{I}}_i\cdot\vec{\mathbf{r}}_{ij})(\vec{\mathbf{I}}_j\cdot\vec{\mathbf{r}}_{ij})}{r_{ij}^2}\right]$
	$H_D = \vec{\mathbf{I}}_i \cdot \hat{\mathbf{D}} \cdot \vec{\mathbf{I}}_j$
J-coupling	$H_j = \vec{\mathbf{I}}_i \cdot \hat{\mathbf{J}} \cdot \vec{\mathbf{I}}_j$

1.2.2 Zeeman Effect

The largest interaction in magnetic resonance is that of the spin with the large external magnetic field. It is the field which creates the $2I+1$ non-degenerated spin angular momentum energy levels characterized by the spin angular momentum quantum number I. When written as a second rank Cartesian tensor, the Zeeman Hamiltonian has the form:

$$H_z = -\vec{\mathbf{I}} \cdot \hat{\mathbf{Z}} \cdot \vec{\mathbf{B}} \tag{1.6}$$

equation 1.6 is simplified when the magnetic field is only applied in the $\hat{z}$ direction:

$$H_z = -\gamma B_z I_z \tag{1.7}$$

where γ is the gyromagnetic ratio, B_z is the magnetic field strength, and I_z is a spin angular momentum operator with eigenvalues $m = -I, (-I+1), ..., I$. The Zeeman Hamiltonian is often written in the form:

$$H_z = \omega_0 I_z \tag{1.8}$$

where ω_0 is the Larmor frequency and is given by $\omega_0 = -\gamma B_z$.

1.2.3 Radio Frequency

The other external interaction is represented by the RF Hamiltonian which occurs due to an applied RF field of frequency ω and strength $\omega_I = -\gamma B_z$:

$$H_{rf} = 2\omega_I \cos(\omega t + \pi) I_x. \tag{1.9}$$

This Hamiltonian describes the application of RF pulses.

1.2.4 Rotating Frame

In an effort to simplify the calculation and interpretation of NMR signals, we often perform the rotating frame transformation to remove the large Zeeman term from the analysis. In the rotating frame transformation, equation 1.9 becomes:

$$H_{rf} = \omega_I (\cos \pi I_x + \sin \pi I_y). \tag{1.10}$$

In this manner, the frequency of the applied field does not oscillate but instead lies in the x-y plane at an angle π from the x axis. This results in the replacement of the Larmor frequency with an offset frequency $\Delta\omega = \omega_0 - \omega$ in the Zeeman Hamiltonian:

$$H_Z = \Delta\omega I_Z. \tag{1.11}$$

This frame transformation allows us to focus on the smaller perturbations that represent the interesting aspects of NMR.

1.2.5 Chemical Shift

The field experienced at the nucleus generally is not exactly the applied B_z; instead, the nucleus is shielded by the surrounding bonding electrons, and the field it experiences varies accordingly. This chemical shielding Hamiltonian can be written as:

$$H_{CS} = \gamma \vec{\mathbf{I}} \cdot \hat{\sigma} \cdot \vec{\mathbf{B}}_0 \tag{1.12}$$

where $\hat{\sigma}$ represents a second rank tensor describing the chemical shielding. The following relationship is noteworthy to remember

$$\hat{\delta} = (\hat{\sigma}_{ref} - \hat{\sigma}) \tag{1.13}$$

where $\hat{\delta}$ represents the chemical shift which is commonly what is reported experimentally, and $\hat{\sigma}_{ref}$ is the absolute shielding of a reference compound (such as tetramethylsilane). The chemical shielding in the PAS can be separated into an isotropic:

$$\begin{aligned} H_{CS}^{iso} &= \gamma B_z \frac{1}{3}(\sigma_{11} + \sigma_{22} + \sigma_{33}) I_z \\ &= -\omega_0 \sigma_{iso} I_z \end{aligned} \tag{1.14}$$

and an anisotropic part:

$$\begin{aligned} H_{CS}^{iso} &= -\frac{1}{3}[\sigma_{33} + \frac{1}{2}(\sigma_{11} + \sigma_{22})](3\cos^2\beta - 1) I_z \\ &= -\omega_0[(\sigma_{11} + \sigma_{22}) \sin^2\beta \ \cos 2\alpha] I_z \end{aligned} \tag{1.15}$$

If we define $\Delta\sigma = \sigma_{33} - \sigma^{iso}$ as the chemical shift anisotropy (CSA) and $\eta = \frac{\sigma_{22} - \sigma_{11}}{\Delta\sigma}$ as the asymmetry of the chemical shift, then the anisotropic part becomes:

$$H_{CS}^{iso} = -\frac{1}{2}\omega_0 \Delta\sigma[(3\cos^2\beta - 1) + \eta \sin^2\beta \ \cos 2\alpha] I_z \tag{1.16}$$

where α and β relate the principal axis system of the chemical shielding tensor to the chemical shielding Hamiltonian is:

$$H_{CS}^{iso} = -\omega_0 \sigma_{iso} I_z - \frac{1}{2}\omega_0 \Delta\sigma[(3\cos^2\beta - 1) + \eta \sin^2\beta \ \cos 2\alpha] I_z \tag{1.17}$$

Quantities which are convenient for comparing the chemical shift anisotropy tensors as defined by Jameson are the span (Ω, which is always positive) and skew (κ, ranging from -1 to +1):

$$\Omega = (\sigma_{33} - \sigma_{11}), \text{ where } \sigma_{33} \geq \sigma_{22} \geq \sigma_{11} \tag{1.18}$$

$$\kappa = (\sigma_{iso} - \sigma_{22})/(\sigma_{33} - \sigma 11), \text{ where } \sigma_{33} \geq \sigma_{22} \geq \sigma_{11} \tag{1.19}$$

1.2.6 J Coupling

Indirect spin-spin coupling, also called the J coupling, is the interaction between nuclei mediated through the bond electrons in the molecule. The J coupling can also be expressed as a second rank Cartesian tensor:

$$H_J = \vec{\mathbf{I}}_i \cdot \hat{\mathbf{J}} \cdot \vec{\mathbf{I}}_j. \tag{1.20}$$

Although most people are familiar with the isotropic part of the J coupling observed in solution state NMR, an anisotropic part also exists which is not usually seen. Using the familiar ladder operators:

$$I_{\pm} = I_x \pm iI_y, \tag{1.21}$$

we simplify equation (1.20) to:

$$H_J = J_{zz} I_{i,z} I_{j,z} + \frac{1}{4}(J_{ij,xx} + J_{ij,yy})(I_{i,+}I_{j,-} + I_{i,-}I_{j,+}) \tag{1.22}$$

where only those terms that commute with I_z are observable. In equation(1.22) the J coupling can be separated into the isotropic:

$$H_J^{iso} = J_{zz} I_{i,z} I_{j,z} \tag{1.23}$$

and the anisotropic part:

$$H_J^{aniso} = \frac{1}{4}(J_{ij,xx} + J_{ij,yy})(I_{i,+}I_{j,-} + I_{i,-}I_{j,+}) \tag{1.24}$$

Thus, even if the anisotropic part is not negligible, it will be difficult to separate it from the direct dipolar couplings experimentally.

1.2.7 Dipolar Coupling

The direct dipole-dipole interaction, also called dipolar coupling, is the interaction of two spins through space. Unlike the J coupling or the chemical shift, the dipolar coupling has no isotropic part; therefore, in liquid state NMR where the samples are isotropically tumbling, the dipolar coupling is not observed. The dipolar coupling interaction can be expressed as a second-rank Cartesian tensor that is both symmetric and traceless:

$$H_D = \vec{\mathbf{I}}_i \cdot \hat{\mathbf{D}} \cdot \vec{\mathbf{I}}_j. \tag{1.25}$$

Again, it is more convenient to write the interaction in the laboratory frame; this frame is rotated from the principal axis system and is axially symmetric about the internuclear vector. The second rank tensor $\hat{\mathbf{D}}$ can be rewritten as:

$$D_{\alpha\beta} = \frac{\mu_0 \hbar \gamma_i \gamma_j}{8\pi^2 r_{ij}^3} \left[\delta_{\alpha\beta} - 3 e_\alpha e_\beta\right] \tag{1.26}$$

where α and β are the laboratory frame axes x, y and z, $\delta_{\alpha\beta}$ is the Kronecker delta function (1 if $\alpha = \beta$, 0 if $\alpha \neq \beta$) and $e_{\alpha,\beta}$ is the α, β component of the unit vector along the internuclear vector, $\vec{r}_{ij}$. Using spherical coordinates and the ladder operators of equation (1.21), equation (1.25) can be rewritten as:

$$H_D = \frac{\mu_0 \hbar \gamma_i \gamma_j}{8\pi^2 r_{ij}^3}(A + B + C + D + E + F) \tag{1.27}$$

with

$$\begin{aligned}
A &= (1 - 3\cos^2\theta_{ij}) I_{i,z} I_{j,z} && (1.28)\\
B &= -\frac{1}{4}(1 - 3\cos^2\theta_{ij})(I_{i,+}I_{j,-} + I_{i,-}I_{j,+}) && (1.29)\\
C &= -\frac{3}{2}\sin\theta_{ij}\cos^2\theta_{ij} e^{-i\phi_{ij}}(I_{i,+}I_{j,z} + I_{i,z}I_{j,+}) && (1.30)\\
D &= C^* = -\frac{3}{2}\sin\theta_{ij}\cos^2\theta_{ij} e^{+i\phi_{ij}}(I_{i,-}I_{j,z} + I_{i,z}I_{j,-}) && (1.31)\\
E &= -\frac{3}{4}\sin\theta_{ij}^2 e^{-i2\phi_{ij}} I_{i,+}I_{j,+} && (1.32)\\
F &= E^* = -\frac{3}{4}\sin\theta_{ij}^2 e^{+i2\phi_{ij}} I_{i,-}I_{j,-} && (1.33)
\end{aligned}$$

Keeping olny those terms in the Hamiltonian that commute with I_z, we are left with the ‘secular’ terms:

$$H_D = 2D_{ij,zz}[I_{i,z}I_{j,z} - \frac{1}{4}(I_{i,+}I_{j,-} + I_{i,-}I_{j,+})] \tag{1.34}$$

where

$$D_{ij,zz} = \frac{\mu_0 \hbar \gamma_i \gamma_j}{4\pi r_{ij}^3}(1 - 3\cos^2\theta_{ij}). \tag{1.35}$$

1.2.8 Calculating Observables

Now that the relevant interaction Hammiltonians have been described in detail for our NMR experiments, a brief review is provided on how to use the Hamiltonians to calculate an NMR signal. Using the density matrix method, we begin by describing our equilibrium density operator which is determined by the populations of states given by the Boltzmann distribution:

$$p_i \propto e^{-\frac{E_i}{kT}} \tag{1.36}$$

where E_i is the energy of the state i. The dominant energy contribution to our system is the Zeeman energy, thus we have:

$$\rho_{eq} = p_i \propto e^{-\frac{\omega_0 I_z}{kT}} \tag{1.37}$$

for the equilibrium density operator. Since the Zeeman energy is small compared to $\frac{1}{kT}$, we can expand the exponential as a Taylor series and truncate it as follows:

$$\rho_{eq} = 1 - \frac{\omega_0 I_z}{kT}. \tag{1.38}$$

The costant term does not evolve; therefore they can be dropped, leaving the reduced density operator:

$$\rho_{eq} = I_z. \tag{1.39}$$

Under the influence of Hermitian Hamiltonian, the time evolution of the density operator can be described by the Liouville-von-Neumann equation:

$$\frac{d\rho}{dt} = i[\rho, H]. \tag{1.40}$$

This equation can be solved for a time-independent Hamiltonian to yield:

$$\rho(t) = e^{-iHt}\rho(0)e^{iHt}. \tag{1.41}$$

If the time evolution of the system can be divided up into several time intervals, each governed by a time-independent Hamiltonian, the evolution can be expressed by:

$$\rho(t) = e^{-iH_n t_n} e^{-iH_{n-1}t_{n-1}} ... e^{-iH_1 t_1} \rho(0)\ e^{-iH_1 t_1} ... e^{-iH_{n-1}t_{n-1}} e^{-iH_n t_n}. \tag{1.42}$$

Using equation (1.42) and the Hamiltonians given in the previous sections, we can now calculate the density operator at a given time. In order to generate the detected signal from the calculated evolution, $\rho(t)$, we employ the operator $I_+ = I_x + iI_y$ which reflects what is detected by the NMR spectrometer. The signal, $S(t)$, is then calculated by:

$$S(t) = Tr(\rho I_+). \tag{1.43}$$

For example, the signal calculated from the NMR experiment which is simply a $(\frac{\pi}{2})_y$ RF-pulse is:

$$S(t) = Tr(e^{-iHt} e^{-i\frac{\pi}{2}I_y} \rho_{eq}\ e^{i\frac{\pi}{2}I_y e^{-iHt}} I_+) \tag{1.44}$$

Here the RF Hamiltonian is expressed in terms of the pulse angle $\theta = \omega_1 \tau = \frac{\pi}{2}$ and spin operator I_y. Immediately following the pulse, the density operator is:

$$\rho(0) = e^{-i\frac{\pi}{2}I_y} I_z e^{i\frac{\pi}{2}I_y} = I_x \tag{1.45}$$

The signal simplifies to:

$$S(t) = Tr(I_z e^{iHt} I_+ e^{-iHt}). \tag{1.46}$$

For an actual calculation, we must choose a basis set; in this case the most convenient are the eigenstates of the Hamiltonian $|i\rangle$. The signal is then:

$$\begin{aligned}
S(t) &= \sum_i \langle i|I_z e^{iHt} I_+ e^{-iHt}|i\rangle && (1.47)\\
S(t) &= \sum_{i,j} \langle i|I_z e^{iHt}|j\rangle\langle j|I_+ e^{-iHt}|i\rangle && (1.48)\\
S(t) &= \sum_{i,j} e^{i(\omega_j - \omega_i)t}\langle i|I_z|j\rangle\langle j|I_+|i\rangle && (1.49)
\end{aligned}$$

where ω_i and ω_j are the eigenvalues of the Hamiltonian and $(\omega_j - \omega_i)$ is the transition frequency. The difference between the diagonal elements of the Hamiltonian matrix $(\omega_j - \omega_i)$ provides the observed transition frequencies, and the product $\langle i|I_z|j\rangle\langle j|I_+|i\rangle$ gives the relative amplitude of the signal at this frequency. Thus we have successfully calculated the NMR signal.

1.3 The General Concept of the Multidimensional NMR Experiment

All multidimensional experiments involve the same basic procedural building blocks and data processing methods with the common aim of revealing either obscured or hidden spectral information. Condensed to the bare essentials, a two-dimensional NMR experiment involves several time periods: preparation, evolution, mixing, and detection. Higher dimensional experiments use additional evolution and mixing periods.

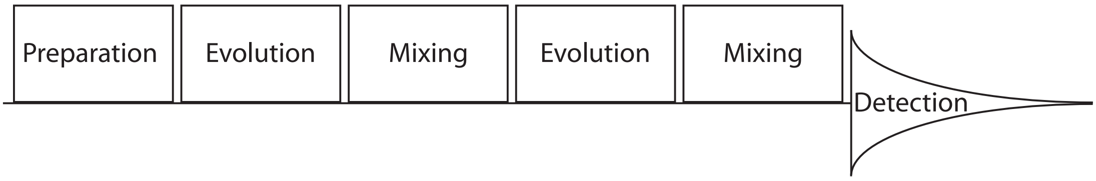

Figure 1.2: General scheme for multi-dimensional NMR spectroscopy.

1.3.1 Preparation

The nuclear spins are ‘prepared’ for the experiment by establishing some well-defined state. Since all multidimensional NMR methods require multiple separate NMR experiments, it is necessary to start all of the individual experiments from the same ‘place’. This ‘state’ can be thermal equilibrium, where all spins have their ‘natural’ magnetization governed by

Boltzmann statistics. Alternatively, this state may be one in which all the spins for one type of nucleus are randomized in orientation (saturated) while another type of nucleus is in thermal equilibrium. A wide variety of experiments can be considered that vary only in the preparation period. In most experiments, however, the preparation period consists only of a delay sufficient to give equilibrium magnetization for all nuclei. The final part of the preparation period usually involves one or more pulses that place magnetization(s) at perpendicular angle to the orientation of the magnetic field axis.

1.3.2 Evolution

Nuclear magnetic moments precess around the direction of a magnetic field, much like a top precesses within the gravitational field of the earth. Nuclei in different chemical environments precess at different rates. These differences in the nuclear precession rate allow us to probe how each type of nucleus will react to a well-defined environment. We can construct this environment out of magnetic field gradients, radio frequency (RF) fields, magnetic fields, and nuclear spin interactions such as J-couplings or through-space dipolar magnetic interactions. The magnetization induced by the last part of the preparation period is permitted to evolve over a fixed period of time (which we will call t_1) under a well-defined magnetic and RF environment.

1.3.3 Mixing

At the end of an evolution time we have the option to redistribute nuclear magnetization among the spins. This distribution may involve the use of pulses and/or time periods. The idea is to allow spin communication for a fixed period. The communication mechanism(s) present will determine the way we interpret the data.Two examples of mechanisms of spin communication are J-coupling and dipolar relaxation.

1.3.4 Detection

Finally, the NMR spectrum of these nuclei is recorded in the form of free induction decay (FID), which looks like dumping harmonic oscillation. The appearance of the spectrum will usually differ in intensity or phased from the ordinary spectrum, but the features are still similar. These phase and /or intensity variations can be investigated in a complete manner by systematically and regularly varying the evolution time(t_1) from zero to some upper limit, collecting a spectrum for each new value of the evolution time used in the experiment. These variations can reveal pertinent details about the chemical and magnetic environments of the nuclei present during the evolution time and can produce information that might otherwise be unobservable.

1.3.5 Summary

The preparation period establishes the condition of the spin system at the beginning of t_1. the preparation time can be set long enough to allow full thermal equilibrium or to produce a steady-sate condition resulting from rapid pulsing. It could involve saturation of one or more spins–either observed nuclei or heteronuclei. This central requirement is that the spin system can be brought to some well-defined state that is the same for all separated values of t_1. It usually ends with a pulse that generates transverse magnetization. This magnetization might arise from the sampling of z magnetization, from the conversion of zero- or double-quantum coherence into single-quantum xy coherence, or from a series of pulses and delays that generate polarization transfer. The magnetization thus induced does not necessarily have to belong to the same nucleus eventually observed.

During the evolution time the magnetization precesses in an environment that might include refocusing pulses to decouple J-couplings and/or refocus chemical shifts. Homonuclear or heteronuclear decoupling and pulsed-field gradients might be applied during all or part of this time. The interactions to be examined in the 2D NMR experiment must be permitted to be active during this period.

The mixing period that follows might be as short as a pulse or as long as many seconds, depending on the coherence or magnetization to be redistributed. For example, a single 90° pulse acting on coupled homonuclear spins can instantly convert magnetization precessing at one transition of the spin system into all other transitions of the same spin system. In this sense it mixes or divides coherences. On the other hand, the mixing period might be much longer if z magnetization is to be redistributed between different frequencies through chemical exchange or dipolar relaxation.

The detection period t_2 is used for the recording of the FID of the observe nucleus. t_2 always has the same duration, no matter what the value of t_1. t_2 can be thought of as a running time axis, 0 to t_{2max}, just as t_1 runs from zero to some maximum value.

These same general features apply to 3D and 4D NMR. These experiments are characterized by replacing a detection period with an evolution time. In the 3D experiment the time t_2 is now an evolution time that may be followed by further pulses and/or delays. t_3 becomes the detection time. The 4D experiment has t_3 as an evolution time and t_4 as the detection time. In general, a mixing period follows each evolution time. This period can involves pulses, spin-locks, delays, and so on.

Chapter 2

Related Issues

2.1 3D Triple-Resonance Experiments for Resonance Assignment

Three- and four-dimensional heteronuclear triple-resonance experiments correlate backbone ^{1}H^N, ^{15}N, ^{1}H$^\alpha$, ^{13}C$^\alpha$, and ^{13}C'(and side-chain ^{1}H$^\beta$and ^{13}C$^\beta$) spins using one-bond and two-bond scalar coupling interactions. The nomenclature established for triple-resonance experiments is more-or-less systematic. The spins that are frequency labeled during the indirect evolution periods or the acquisition period are listed using HN, N, HA, CA, CO, HB, and CB to represent the ^{1}H^N, ^{15}N, ^{1}H$^\alpha$, ^{13}C$^\alpha$, ^{13}C', ^{1}H$^\beta$, and ^{13}C$^\beta$spins, respectively. Spins through which coherence is transferred, but not frequency-labeled, are given in parentheses. For example, a triple-resonance experiment utilizing the following coherence transfers:

$$\begin{array}{ccccccc} {}^{1}\mathbf{H}^{N} \rightarrow & {}^{15}\mathbf{N} \rightarrow & {}^{13}\mathbf{CO} \rightarrow & {}^{13}\mathbf{C}^{\alpha} \rightarrow & {}^{13}\mathbf{CO} \rightarrow & {}^{15}\mathbf{N} \rightarrow & {}^{1}\mathbf{H}^{N} \\ & (t_1) & & (t_2) & & & (t_3) \end{array}$$

might be called a (HN)N(CO)CA(CO)(N)NH experiment. However, this unwieldy naming can be shortened by using the following conventions. First, the experiment is a so-called "out and back" pulse sequence in which the initially excited proton spin and the detected proton spin are identical. Omitting the back-transfer steps from the name yields the shorter form, HNN(CO)CA, without introducing an ambiguity because the ^{13}C$^\alpha$would never be the detected spin, and the presence of a back-transfer pathway to the ^{1}H^{N}spin thereby is implied. Second, the designation of the ^{1}H^{N}spin is redundant, because the transfer ^{1}H^N ↔ ^{15}N is the only available step. Thus, HN can be abbreviated as H without complication to yield the final name, HN(CO)CA, for this experiment. This abbreviated name equally well describes an experiment that rearranges the labeling periods as

$$\begin{array}{ccccccc} {}^{1}\mathbf{H}^{N} \rightarrow & {}^{15}\mathbf{N} \rightarrow & {}^{13}\mathbf{CO} \rightarrow & {}^{13}\mathbf{C}^{\alpha} \rightarrow & {}^{13}\mathbf{CO} \rightarrow & {}^{15}\mathbf{N} \rightarrow & {}^{1}\mathbf{H}^{N} \\ & & & (t_1) & & (t_2) & (t_3) \end{array}$$

The order in which the frequency labeling is performed is easily determined from the pulse sequence.

Triple resonance experiments are the method of choice for the sequential assignment of larger proteins ($>$ 150 amino acids). These experiments are called 'triple resonance' because three different nuclei (^{1}H, ^{13}C, ^{15}N) are correlated. The experiments are performed on doubly labeled (^{13}C, ^{15}N) proteins.

The most important advantage of the triple resonance spectra is their simplicity: They contain only a few signals on each frequency - often only one. The problem of spectral overlap is therefore remarkably reduced (this is the main reason, why proteins of more than 20 kDa can be assigned with triple resonance experiments). The correct choice of connectivities between amino acids is the main problem in the assignment of triple resonance spectra.

Another advantage of triple resonance spectra is their high sensitivity which is caused by an efficient transfer of magnetization. The magnetization is transferred via ^{1}J or ^{2}J couplings (*i.e.* directly via the covalent chemical bonds). Therefore, the transfer times are shorter and the losses due to relaxation are smaller than in homonuclear experiments.

The following sub-sections describes the most frequently used 3D triple-resonance experiments for sequence specific resonance assignment.

2.1.1 HNCA experiment

The HNCA experiment is the prototype of all triple resonance experiments. It correlates the ^{13}C$^{\alpha}$ resonances of an amino acid residue with the ^{1}H^{N} and ^{15}N resonances of the following residue. Starting at an ^{1}H^N, the magnetization is transferred to the directly attached ^{15}N (via $^1J_{H^NN}$) then to the ^{13}C$^\alpha$ (via $^1J_{C^\alpha N}$), following the chemical shift evolution of ^{13}C$^\alpha(t_1)$ as first spectral dimension.

The magnetization is transferred back to the same pathway. Therefore, the magnetization is transferred from ^{13}C$^\alpha$ to ^{15}N , which is measured as ^{15}N(t_2), the second spectral dimension. Then the magnetization is transferred to the ^{1}H^N which is measured as ^{1}H$^N(t_3)$, the third spectral dimension.

In each step magnetization is transferred via J couplings between the nuclei. The coupling which connects the ^{15}N atom with the ^{13}C$^\alpha$ carbon of the preceding amino acid ($^2J_{C^\alpha N} = 7$ Hz) is only marginally smaller than the coupling to the directly attached ^{13}C$^\alpha$ atom ($^1J_{C^\alpha N}$ = 11 Hz). Thus, the ^{15}N atom of a given amino acid is correlated with both ^{13}C$^\alpha$ − its own and the one of the preceding amino acid.

In this experiment, C$^\alpha(i)$, C$^\alpha(i-1)$, N(i), and H$^N(i)$ resonances are observed, where i is the $i - th$ residue in the amino acid chain (*e.g.* a protein or a peptide). Therefore, it is possible to assign the protein backbone resonances exclusively with an HNCA spectrum. But usually more triple resonance experiments are needed because the cross signal of the preceding amino acid has to be identified and degenerated resonance frequencies have to be resolved.

2.1.2 HN(CO)CA experiment

The HN(CO)CA experiment provides sequential correlations between the ^{1}H^N and ^{15}N chemical shifts of one amino acid residue and the ^{13}C$^\alpha$ chemical shift of the preceding residue by transferring coherence via the intervening ^{13}C' spin. In this experiment, C$^\alpha(i-1)$, N(i), and H$^N(i)$ resonances are observed. These chemical shifts provide the same sequential information, C$^\alpha(i-1)$, as the HNCA experiment; however, the HNCA experiment dose not always distinguish intra-residue and inter-residue connectivities because the $^1J_{C^\alpha N}$ and $^2J_{C^\alpha N}$ cou-

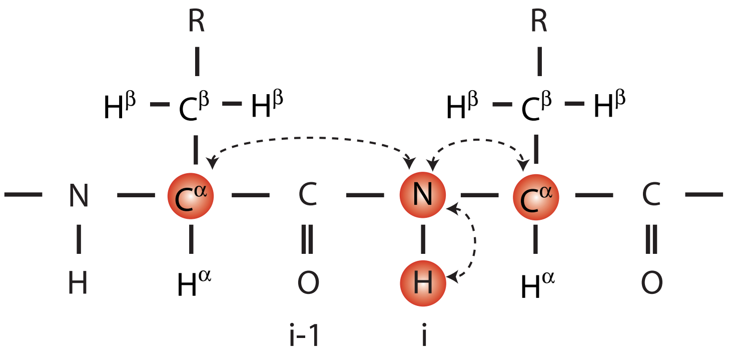

Figure 2.1: HNCA experiment: The magnetization is transferred from the ${}^{1}\mathrm{H}^{N}(i) \rightarrow {}^{15}\mathrm{N}(i) \rightarrow {}^{13}\mathrm{C}^{\alpha}(i)/{}^{13}\mathrm{C}^{\alpha}(i-1)$ and then comes back to ${}^{1}\mathrm{H}^{N}(i)$ along the same path. The frequencies of ${}^{13}\mathrm{C}^{\alpha}(i)$, ${}^{13}\mathrm{C}^{\alpha}(i-1)$, ${}^{15}\mathrm{N}(i)$ and ${}^{1}\mathrm{H}^{N}(i)$ (red) are observed.

pling constants can be of comparable magnitude, or the intra-residue and inter-residue ${}^{13}\mathrm{C}^{\alpha}$ chemical shifts may coincidentally be degenerated.

The HN(CO)CA experiment circumvents these problems by providing sequential correlations exclusively. In addition, the sensitivity of the HN(CO)CA experiment is larger than that of the HNCA for larger proteins, because the relay of magnetization via the one bond ${}^{1}J_{NC'}$ and ${}^{1}J_{C^{\alpha}C'}$ scalar coupling interactions is more efficient than transfer via the relatively small two-bond ${}^{2}J_{C^{\alpha}N}$ scalar coupling interaction. In the HN(CO)CA experiment, $\mathrm{C}^{\alpha}(i-1)$, $\mathrm{N}(i)$, and $\mathrm{H}^{N}(i)$ resonances are observed.

2.1.3 CBCANH experiment

The CBCANH experiment correlates the ${}^{13}\mathrm{C}^{\alpha}$ and ${}^{13}\mathrm{C}^{\beta}$ resonances with the ${}^{1}\mathrm{H}^{N}$ and ${}^{15}\mathrm{N}$ resonances of the same residue and the ${}^{1}\mathrm{H}^{N}$and ${}^{15}\mathrm{N}$ resonances of the neighboring residue via the ${}^{1}J_{C^{\alpha}N}$ and ${}^{2}J_{C^{\alpha}N}$ couplings, respectively. Thus, magnetization is transferred from $\mathrm{H}^{\alpha}/\mathrm{H}^{\beta}$ to directly bound ${}^{13}\mathrm{C}^{\alpha}/{}^{13}\mathrm{C}^{\beta}$, following by chemical shift evolution of ${}^{13}\mathrm{C}^{\alpha}(t_1)/{}^{13}\mathrm{C}^{\beta}(t_1)$. In the following step, the magnetization transfer ${}^{13}\mathrm{C}^{\beta} \rightarrow {}^{13}\mathrm{C}^{\alpha}$ and ${}^{13}\mathrm{C}^{\alpha} \rightarrow {}^{13}\mathrm{C}^{\alpha}$ is selected. The

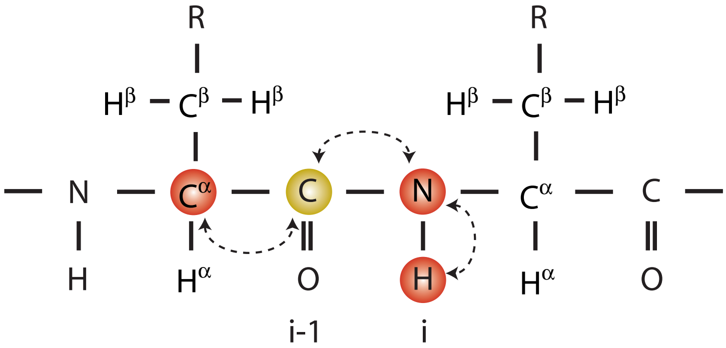

Figure 2.2: HN(CO)CA experiment: The magnetization is transferred from the $^{1}H^{N}(i)$ → $^{15}N(i)$ → $^{13}C'(i-1)$ → $^{13}C^{\alpha}(i-1)$ and then comes back to $^{1}H^{N}(i)$ along the same pathway. The $^{13}C'$ (yellow) acts only as relay nucleus, its frequency is not detected. The frequencies of $^{13}C^{\alpha}(i-1)$, $^{15}N(i)$ and $^{1}H^{N}(i)$ (red) are observed.

magnetization is transferred to ^{15}N from $^{13}C^{\alpha}$ of the same amino acid (via $^{1}J_{C^{\alpha}N}$) and of the next amino acid (via $^{2}J_{C^{\alpha}N}$), following chemical shift evolution of $^{15}N(t_2)$. Finally, after transfer from ^{15}N to H^{N}, the magnetization is detected during chemical shift evolution of $H^{N}(t_3)$.

In this experiment, $C^{\beta}(i)$, $C^{\beta}(i-1)$, $C^{\alpha}(i)$, $C^{\alpha}(i-1)$, $N(i)$, and $H^{N}(i)$ resonances are observed. For a medium-sized protein ($\sim$ 15 kDa), this experiment alone can provide virtually complete sequential assignment of the $^{1}H^{N}$, ^{15}N, $^{13}C^{\alpha}$, and $^{13}C^{\beta}$ resonances, because in addition to the sequential connectivities, the $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ chemical shifts provide information on the amino acid type.

2.1.4 CBCA(CO)NH experiment

The CBCA(CO)NH experiment correlates both the $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ resonances of an amino acid residue with the $^{1}H^{N}$ and ^{15}N resonances of the preceding residue. Thus, magnetization

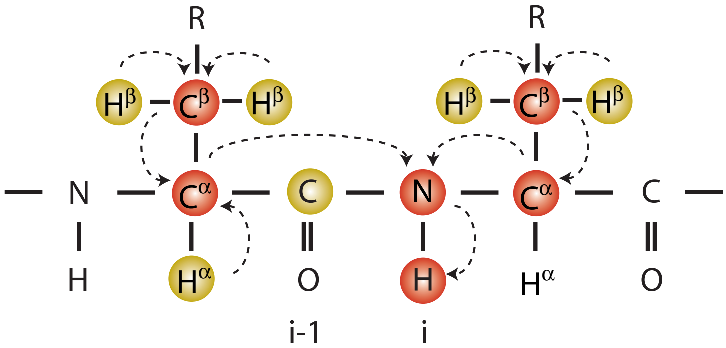

Figure 2.3: CBCANH experiment: The magnetization is transferred from the ^{1}H$^{\alpha}(i)$/^{1}H$^{\beta}(i)$ → ^{13}C$^{\alpha}(i)$/^{13}C$^{\beta}(i)$ → ^{13}C$^{\alpha}(i)$, at the same time ^{1}H$^{\alpha}(i-1)$/^{1}H$^{\beta}(i-1)$ → ^{13}C$^{\alpha}(i-1)$/^{13}C$^{\beta}(i-1)$ → ^{13}C$^{\alpha}(i-1)$. Then, the magnetization is transferred from ^{13}C$^{\alpha}(i)$/^{13}C$^{\alpha}(i-1)$ → ^{15}N(i) → ^{1}H$^{N}(i)$. The ^{1}H$^{\alpha}$, ^{1}H$^{\beta}$ and ^{13}C' (yellow) act only as relay nuclei, their frequency are not detected. The frequencies of ^{1}H$^{N}(i)$, ^{15}N(i), ^{13}C$^{\alpha}(i)$, ^{13}C$^{\beta}(i)$, ^{13}C$^{\alpha}(i-1)$ and ^{13}C$^{\beta}(i-1)$ (red) are observed.

is transferred from H$^{\alpha}$/H$^{\beta}$ to directly bound ^{13}C$^{\alpha}$/^{13}C$^{\beta}$ followed by chemical shift evolution of ^{13}C$^{\alpha}(t_1)$/^{13}C$^{\beta}(t_1)$, then from ^{13}C$^{\alpha}$/^{13}C$^{\beta}$ to ^{15}N, following chemical shift evolution of ^{15}N(t_2). Finally, after transferred from ^{15}N to H^{N}, the magnetization is detected during chemical shift evolution of H$^{N}(t_3)$.

In this experiment, C$^{\beta}(i-1)$, C$^{\alpha}(i-1)$, N(i), and H$^{N}(i)$ resonances are observed. With the same reason to HN(CO)CA experiment, this experiment is useful to circumvent the degeneracy between the intra-residue (C$^{\beta}(i)$ and C$^{\alpha}(i)$) and inter-residue (C$^{\beta}(i-1)$ and C$^{\alpha}(i-1)$) chemical shifts and to obtain more intense inter-residue chemical shift peaks.

2.1.5 HNCO experiment

The HNCO experiment is identical to the HNCA experiments except for the interchange of ^{13}C$^{\alpha}$ and ^{13}C' . Starting at an ^{1}H^{N}, the magnetization is transferred to the directly

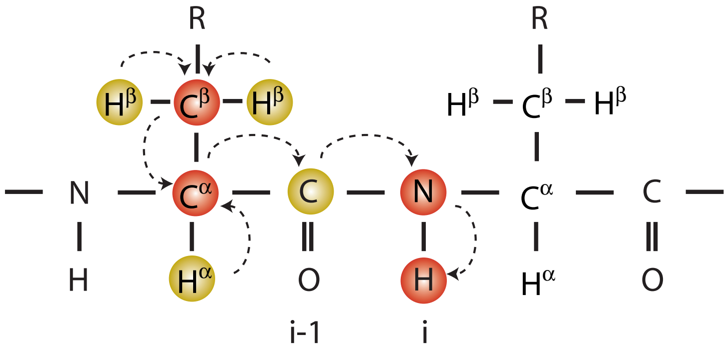

Figure 2.4: CBCA(CO)NH experiment: The magnetization is transferred from the ^{1}H$^\alpha(i-1)$/^{1}H$^\beta(i-1)$ → ^{13}C$^\alpha(i-1)$/^{13}C$^\beta(i-1)$ → ^{13}C$^\alpha(i-1)$ → ^{15}N(i) → ^{1}H$^N(i)$. The ^{1}H$^\alpha$, ^{1}H$^\beta$ and ^{13}C' (yellow) act only as relay nucleus, their frequency are not detected. The frequencies of ^{1}H$^N(i)$, ^{15}N(i), ^{13}C$^\alpha(i-1)$ and ^{13}C$^\beta(i-1)$ (red) are observed.

attached ^{15}N (via $^1J_{H^N N}$) then to the ^{13}C' (via $^1J_{NC'}$), following the chemical shift evolution of ^{13}C'(t_1). After that, the magnetization is transferred back to same way. Therefore, the magnetization is transferred from ^{13}C' to ^{15}N , which is measured as ^{15}N(t_2). Then the magnetization is transferred to the ^{1}H^N which is measured as ^{1}H$^N(t_3)$.

In this experiment, C$'(i-1)$, N(i), and H$^N(i)$ resonances are observed. The HNCO experiment is one of the most sensitive 3D NMR experiments. It can be used as reference spectrum for the 2D HN-HSQC spectrum allowing to distinguish the backbone ^{15}N chemical shifts from side chain ^{15}N chemical shifts in the 2D HN-HSQC spectrum.

2.1.6 HN(CA)CO experiment

The HN(CA)CO experiment provides intra-residue correlations between the amide ^{1}H^N , ^{15}N and ^{13}C' chemical shifts by using the one-bond ^{15}N$-^{13}$C$^\alpha$ and ^{13}C$^\alpha-$ ^{13}C' J couplings to transfer coherence. In addition, this experiment can also provide sequential connectivities

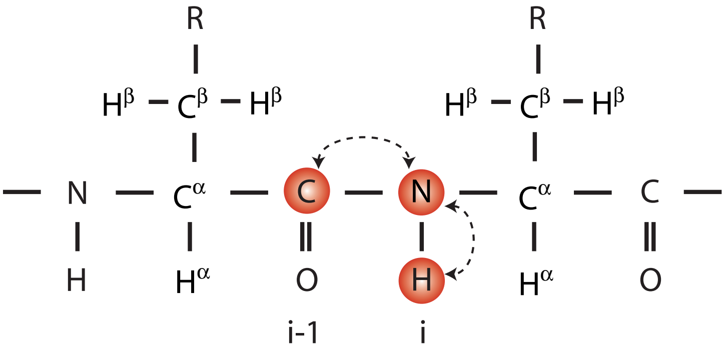

Figure 2.5: HNCO experiment: The magnetization is transferred from the ${}^{1}\text{H}^{N}(i)$ → ${}^{15}\text{N}(i)$ → ${}^{13}\text{C'}(i-1)$ and then comes back to ${}^{1}\text{H}^{N}(i)$ along the same path. The frequencies of ${}^{1}\text{H}^{N}(i)$, ${}^{15}\text{N}(i)$ and ${}^{13}\text{C'}(i-1)$ (red) are observed.

from the ^{15}N spins to the ^{13}C' of the preceding residue via the inter-residue two-bond ${}^{2}J_{C^{\alpha}N}$ coupling. The HN(CA)CO experiment is derived from the HNCA experiment. Starting on the ${}^{1}\text{H}^{N}$, magnetization is transferred via the ^{15}N to the ${}^{13}\text{C}^{\alpha}$ spins. The magnetization is transferred to ${}^{13}\text{C}^{\alpha}$ from ^{13}C' of the same amino acid (via ${}^{1}J_{C^{\alpha}C'}$) and of the next amino acid (via ${}^{2}J_{C^{\alpha}C'}$), followed by chemical shift evolution of ${}^{13}\text{C'}(t_1)$, then from ^{13}C' to the ^{15}N, following chemical shift evolution of ${}^{15}\text{N}(t_2)$. Finally, after transferred from ^{15}N to H^{N}, the magnetization is detected during chemical shift evolution of $\text{H}^{N}(t_3)$.

When used in conjunction with the HNCO pxperiment which gives the sequential correlations only, the HN(CA)CO experiment provides a method for sequentially assigning the ${}^{1}\text{H}^{N}$, ^{15}N , and ^{13}C' resonances.

2.1.7 Assignment strategy

From the combination of CBCA(CO)NH and CBCANH experiments backbone resonance assignments and the sequential connectivities can be obtained. These experiments will be

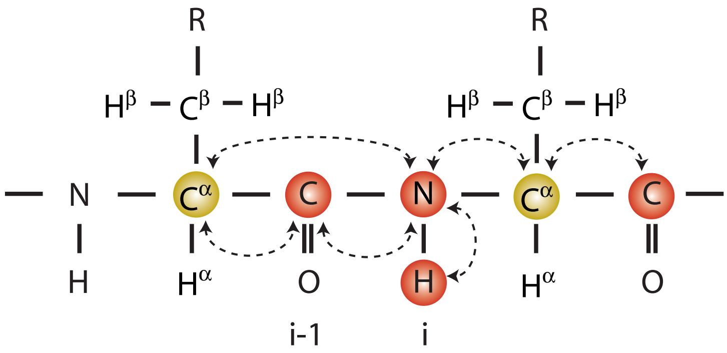

Figure 2.6: HN(CA)CO experiment: The magnetization is transferred from the ^{1}H$^N(i)$ $\rightarrow$ ^{15}N(i) $\rightarrow$ ^{13}C$^\alpha(i)$ $\rightarrow$ ^{13}C'($i-1$)/^{13}C'(i) and then comes back to ^{1}H$^N(i)$ along the same pathway. The ^{13}C$^\alpha$ (yellow) acts only as relay nucleus, its frequency is not detected. The frequencies of ^{1}H^N, ^{15}N and ^{13}C' (red) are observed.

sensitive enough for medium size proteins ($\sim$ 15 kD, 130 amino acids) and provide the ^{13}C$^\alpha$ and ^{13}C$^\beta$ chemical shifts to establish the sequential link between neighboring residues. Furthermore, when both the ^{13}C$^\alpha$ and ^{13}C$^\beta$ chemical shifts are provided at the same time, it gives important information about the amino acid type and secondary structure (*e.g.* $\alpha-$helix and $\beta-$strand).

However, for larger proteins, the CBCA(CO)NH and CBCANH experiments become less sensitive; therefore, some chemical shifts become hard to distinguish from noise. Then the more sensitive experiments, HNCA and HN(CO)CA, can be used to fill in the missing chemical shifts. If the ^{13}C$^\alpha$ and ^{13}C$^\beta$ chemical shifts obtained from these four experiments still leave ambiguities, the pair of HNCO and HN(CA)CO can be used to resolve the ambiguities.

However, since the HN(CA)CO experiment is quite insensitive, this approach will be useful only in combination with a deuterated protein. The set of six backbone experiments should allow the unambiguous assignment even for larger proteins ($\sim$ 30 kD).

2.2 Resonance Assignment Procedure

Once all the spectra required for resonance assignment, have been measured, several steps follow for the resonance assignment: peak picking, peak grouping, pseudo-residue linking, and pseudo-residue-segment mapping. The following sub-sections describe each step in detail.

2.2.1 Peak picking

After NMR experiment and processing the NMR spectra, the first step in resonance assignment is the peak picking. In the spectra, there will be real resonance peaks, which come from nuclei of amino acids, as well as noise and artifact peaks. The peak picking comprises extraction of real peaks from noise and artifact peaks, definition of exact peak positions in frequency dimensions, and integration of peak volumes. The positions (or coordinates) of the peaks are used for pseudo-residue linking, and the peak volumes are used to distinguish between intra- and inter-chemical shift peaks.

2.2.2 Peak grouping

After peak picking, the peaks have to be grouped into so called pseudo-residues to be useful for assignment. In a 3D spectrum, a peak has three coordinates in chemical shift 'space', whose unit is frequency (Hz). Conventionally, 'ppm' is used as unit of the chemical shift instead of 'Hz', since it is independent of the magnetic field.

The second and third chemical shift dimensions generally correspond to N and H^N chemical shifts, respectively: The first chemical shift dimension depends on the type of 3D NMR experiment. For example, in a HNCA experiment, the first chemical shift dimension corresponds to the $C^\alpha(i-1)$ and $C^\alpha(i)$ chemical shifts; in a HN(CO)CA experiment, $C^\alpha(i-1)$; in a HNCACB experiment, $C^\alpha(i-1)$, $C^\alpha(i)$, $C^\beta(i-1)$ and $C^\beta(i)$; in a CBCA(CO)NH experiment, $C^\alpha(i-1)$ and $C^\beta(i-1)$.

Therefore, although we measure different types of 3D NMR spectra, all the spectra can be associated with matching of the N and H^N chemical shift dimensions; and all peaks, which have the same N and H^N chemical shifts, can be grouped into a pseudo-residue. This pseudo-residue has inter-chemical shifts as well as intra-chemical shifts. These inter- and intra-chemical shifts are used for linking pseudo-residues in the next step.

2.2.3 Pseudo-residue linking

A PR (pseudo-residue: a group of peaks having the same H^N and N chemical shifts) consists of the intra- and neighbor inter-chemical shifts (e.g. PR(i) $\ni$ {C'($i-1$), $C^\alpha(i-1)$, $C^\beta(i-1)$, $H^\alpha(i-1)$, C'(i), $C^\alpha(i)$, $C^\beta(i)$, $H^\alpha(i)$, $\ldots$}). It is called *'pseudo-residue linking'* when a pseudo-residue is linked to another pseudo-residue with matching of inter-chemical shifts and intra-chemical shifts. This can be extended until there are no pseudo-residues left matching inter- and intra-chemical shifts.

For example, PR(i), which has $C^\alpha(i-1)$, $C^\beta(i-1)$, $C^\alpha(i)$ and $C^\beta(i)$ chemical shifts, can be linked to PR(j), which has $C^\alpha(j-1)$, $C^\beta(j-1)$, $C^\alpha(j)$ and $C^\beta(j)$, if the following conditions are satisfied, $C^\alpha(i-1) \simeq C^\alpha(j)$ and $C^\beta(i-1) \simeq C^\beta(j)$. The ' $\simeq$' is used instead of '=' to consider the experimental uncertainty. This uncertainty is generally, less than 0.5 ppm for C^α, C^β chemical shifts; 0.1 ppm for C'; 0.05 ppm for H^α.

2.2.4 Pseudo-residue-segment mapping

The final step is the mapping of spin-system segments onto the primary sequence using partial knowledge of their amino acid types. If we measure HNCA, HN(CO)CA, HNCACB and CBCA(CO)NH or HN(CO)CACB spectra, we can obtain the chemical shifts of $^1H^N$, ^{15}N, ^{13}C', $^{13}C^\alpha$ and $^{13}C^\beta$. These chemical shifts provide probabilities of amino acid types. While mapping, the best-fit position of the segment onto the primary protein sequence is searched by comparing the measured chemical shifts with the ‘expected chemical shifts (or predicted chemical shifts)’ from the protein sequence. The ‘expected chemical shifts’ can be obtained from the BMRB chemical shift database. Those chemical shifts are mean values from database, so that the standard deviations have to be considered.

When the standard deviations are include into the mean values of the chemical shifts, amino acid types could overlap. This ambiguity can usually be resolved, when more than 4 residues are sequentially connected. Because sequentially connected 5 amino acids are already quite unique against to the whole protein sequence, and although there is overlapping of amino acid types, it can be resolved when discrepancies or preferences of the other amino acid types are taken into account.

In principle, if the sequential connectivity is unique and the segment size is generally larger than 4 amino acids, then the segment can be mapped onto the protein sequence with certainty. However, the quality of the spectra frequently makes the assignment process difficult, because of the ambiguity of sequential connections, missing chemical shifts, additional artifact peaks, and isolated segments due to either missing chemical shifts or the occurrence of prolines, which are not observable due to lack of the $^1\text{H}^N$ atom.

2.3 Constraints for Structure Calculation

In protein NMR spectroscopy, structure-generation calculations are usually carried out using the following data as input: (1) distance constraints based on the analysis of multidimensional NOESY spectra; (2) dihedral angles constraints derived from experimental and/or statistical data, including NOESY, chemical shift and J coupling constant data; (3) residual dipolar couplings (RDCs). In some cases, disulfide and/or hydrogen bond distance constraints derived from other experimental data are also included.

2.3.1 Distance constraints

After the sequence specific assignment of NMR resonances has been done, the data which are relevant for the structure has to be extracted. By far the most important NMR-observable parameter used in determining protein structures is the NOE. The dipolar cross-relaxation constant is proportional to the inverse sixth power of the the distance, r_{ij} , between the two interacting protons, i and j. (Eq. 2.1)

$$NOE_{ij} \sim \frac{1}{r_{ij}^6} \ , \tag{2.1}$$

In the initial rate approximation, NOE cross-peak intensities are proprotional to the cross-relaxation rate constants. Thus, if one inter-proton distance, r_{ref}, is known (e.g., from covalent geometry), then another, unknown inter-proton distance, r_i is determined by the relationship (ignoring differential internal mobility):

$$r_i = r_{ref}\left(\frac{S_{ref}}{S_i}\right) \ , \tag{2.2}$$

in which S_{ref} and S_i are the integrated cross peakk intensities. It can be estimated in the 2D NOESY, 3D ^{15}N-NOESY-HSQC and 3D ^{13}C-NOESY-HSQC spectra.

In this procedure, all non-sequential signals which are visible in the NOESY spectra have to be assigned, and the ambiguity for the assignment significantly increases with the protein size. The number of NOEs easily exceeds 1000 in a medium-sized protein (100 amino acids). NOE assignment is one of the most time consuming and difficult part for the structure determination due to the ambiguity of the NOE assignment.

Generally, in the earlier stage, only unambiguously assigned NOEs are used for the structure calculation. If there are no violations between the distances which are estimated from NOE intensities and which are back-calculated from the structure, some amount of ambiguous NOEs can be included into the structure calculation; but if there are violations, then the violated NOE assignments have to be checked and eventually be re-assigned. Once there are no NOE violations, some more NOEs are included for the structure calculation.

After that, the newly introduced NOEs have to be evaluated. This iterative structure calculation and correcting NOE assignment makes structure determination tedious.

2.3.2 Dihedral angle constraints

In addition to inter-proton distances, the ϕ-dihedral angles of the protein backbone can be determined from a COSY spectrum or a HNCA-J spectrum (a variant of the HNCA spectrum, from which the coupling constants of the N-C$^\alpha$ bonds can be determined). Dihedral angles are connected with the coupling constants via the Karplus equation:

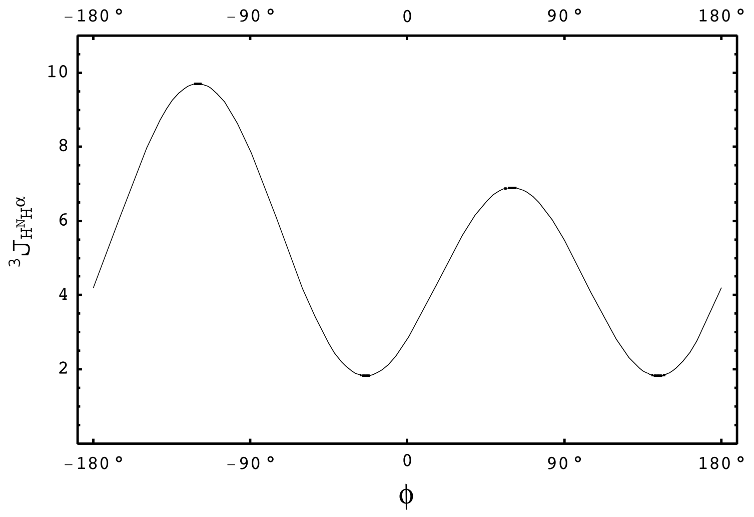

Figure 2.7: The Karplus curve describing the variation of $^3J_{H^NH^\alpha}$ with backbone dihedral angle ϕ . The dihedral angle between H^N and H$^\alpha$ is given by $\theta = \phi - 60°$. The curve shown was calculated using Eq. 2.3 with the constants $A = 6.4, B = -1.4,$ and $C = 1.9$.

$$^3J = A\cos^2\theta + B\cos\theta + C \ . \tag{2.3}$$

The constants A, B, and C depend on the particular nuclei involved in the covalent bonds. Historically, dihedral angle restraints for ϕ and χ_1 dihedral angles have been derived only from $^3J_{H^NH^\alpha}$ and $^3J_{H^\alpha H^\beta}$ coupling constants, respectively [21, 88, 110]. In addition, several experiments have recently been developed that allow measurement of $^{13}C - {}^{13}C$, $^{13}C - {}^{15}N$,

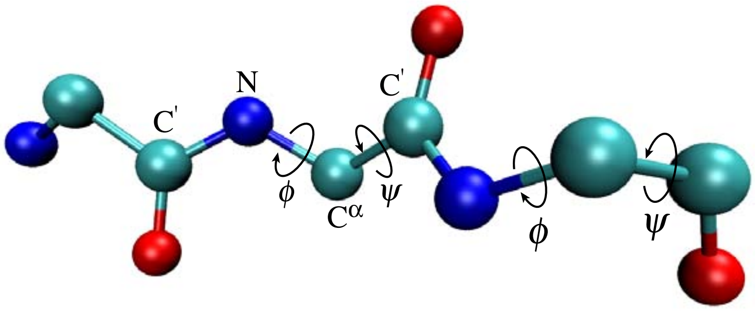

Figure 2.8: A backbone of a peptide chain, including oxygen. Blue, cyan and red indicate nitrogen, carbon and oxygen, respectively. The dihedral angle, ϕ, is the angle between a C'NC$^{\alpha}$-plane and a NC$^{\alpha}$C'-plane; and the dihedral angle, ψ, is angle between a NC$^{\alpha}$C'-plane and a C$^{\alpha}$C'N-plane. By convention, ϕ and ψ are both defined as 0° when the two peptide bonds flanking that C$^{\alpha}$ are in the same plane.

$^{1}H - {}^{15}N$ and $^{1}H - {}^{13}C$ three-bond coupling constants [75, 109].

2.3.3 Chemical-bond-vector orientation constraint

Residual dipolar couplings (RDCs) have recently re-emerged as a tool in NMR to study macromolecular structure and function in a solution environment. The relation between the internuclear vector and the dipolar coupling between two spins (atoms) can be found in chapter 2. For the purpose of deriving the resonance frequencies (*i.e.*, dipolar splittings) only the $\hat{z}$ component of the local field of one nuclear dipole at the position of the second nucleus is relevant (secular approximation):

$$D^{ij} = \frac{\mu_0 \hbar \gamma_i \gamma_j}{4\pi r_{ij}^3} \left\langle \frac{1 - 3\cos^2\theta_{ij}}{2} \right\rangle \tag{2.4}$$

, where the angular brackets refer to the time or ensemble average, which are equivalent for isotropic and liquidcrystalline solution, θ is the angle between the $\vec{r}_{ij}$ (internuclear vector) and B$_0$ (the magnetic field). RDCs are complementary to the more conventional use of NOEs

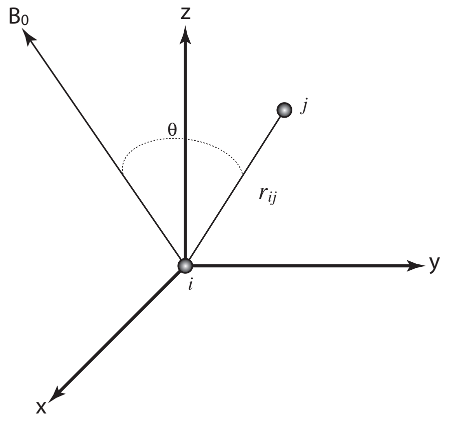

Figure 2.9: The definition of a molecular frame. r_{ij} is the distance between an atom i and j, B_0 is the strong static magnetic field.

to provide structural information. While NOEs are local-distance restraints, RDCs provide long-range orientational information. RDCs are now widely used in structure calculations [68]. RDCs are usually used in a refinement stage of structure calculations. The reasons are that the potential energy surface is very rough and initial inclusion of RDCs may trap the structure into a false minimum, leading to convergence problems.

2.4 Calculation of Tertiary Structure

The idea of computer-aided structure calculation is to convert distance- and torsion-angle-data (constraints) into a three dimensional structure. However, the experimentally determined distances and torsion angles by themselves are not sufficient to fully characterize a protein structure, as they are based on a limited number of proton-proton distances. Additional knowledge of empirical input data, such as bond lengths of all covalently attached atoms and bond angles, enables a reasonably exact structure determination when cooperated with experimental structure information.

For this purpose, a randomly folded starting structure is calculated from the empirical data and the known amino acid sequence. The computer program then tries to fold the starting structure in such a way, that the experimental restraints are satisfied by the calculated structures. In order to achieve this, each known parameter is assigned an energy potential, which will give minimal energy if the calculated distance or angle coincides with its input value. The computer program tries to calculate a structure having a possibly small $\mathbf{E}_{total}$ energy:

$$\mathbf{E}_{total} = \mathbf{E}_{chem} + w_{exp}\mathbf{E}_{exp} \tag{2.5}$$

$$\mathbf{E}_{exp} = \mathbf{E}_{NOE} + \mathbf{E}_{torsion} + \mathbf{E}_{H-bond} + \mathbf{E}_{RDC} + \ldots \tag{2.6}$$

$$\mathbf{E}_{chem} = \mathbf{E}_{bond} + \mathbf{E}_{angle} + \mathbf{E}_{dihedral} + \mathbf{E}_{vdw} + \mathbf{E}_{electric} \tag{2.7}$$

, in which $\mathbf{E}_{NOE}$, $\mathbf{E}_{torsion}$, $\mathbf{E}_{H-bond}$, and $\mathbf{E}_{RDC}$ are the energy of NOEs, torsion angles, hydrogen-bonds, and RDCs, respectively; $\mathbf{E}_{bond}$, $\mathbf{E}_{angle}$, $\mathbf{E}_{dihedral}$, $\mathbf{E}_{vdw}$, and $\mathbf{E}_{electric}$ are the energy of bonds, angles, dihedral angles, van-der-Waals, and electric potential, respectively.

Without the experimentally determined distance- and torsion angle-constraints from the NMR spectra, the protein molecule can adopt a huge number of conformations due to the free rotation around its chemical bonds (except for the peptide bond, the N$-$ C$^{\alpha}$ bond and the C$^{\alpha}$ $-$C' bond). All these possible conformations are summed up in the so-called conformational space. Therefore, it is important to identify as many constraints as possible from the NMR spectra to restrict the conformational space as much as possible, thus getting close to the true structure of the protein. In fact, the number of constraints employed is more important than the accuracy of proton-proton distances.

There are various computer programs, employing two different approaches for calculating a protein structure in solution:

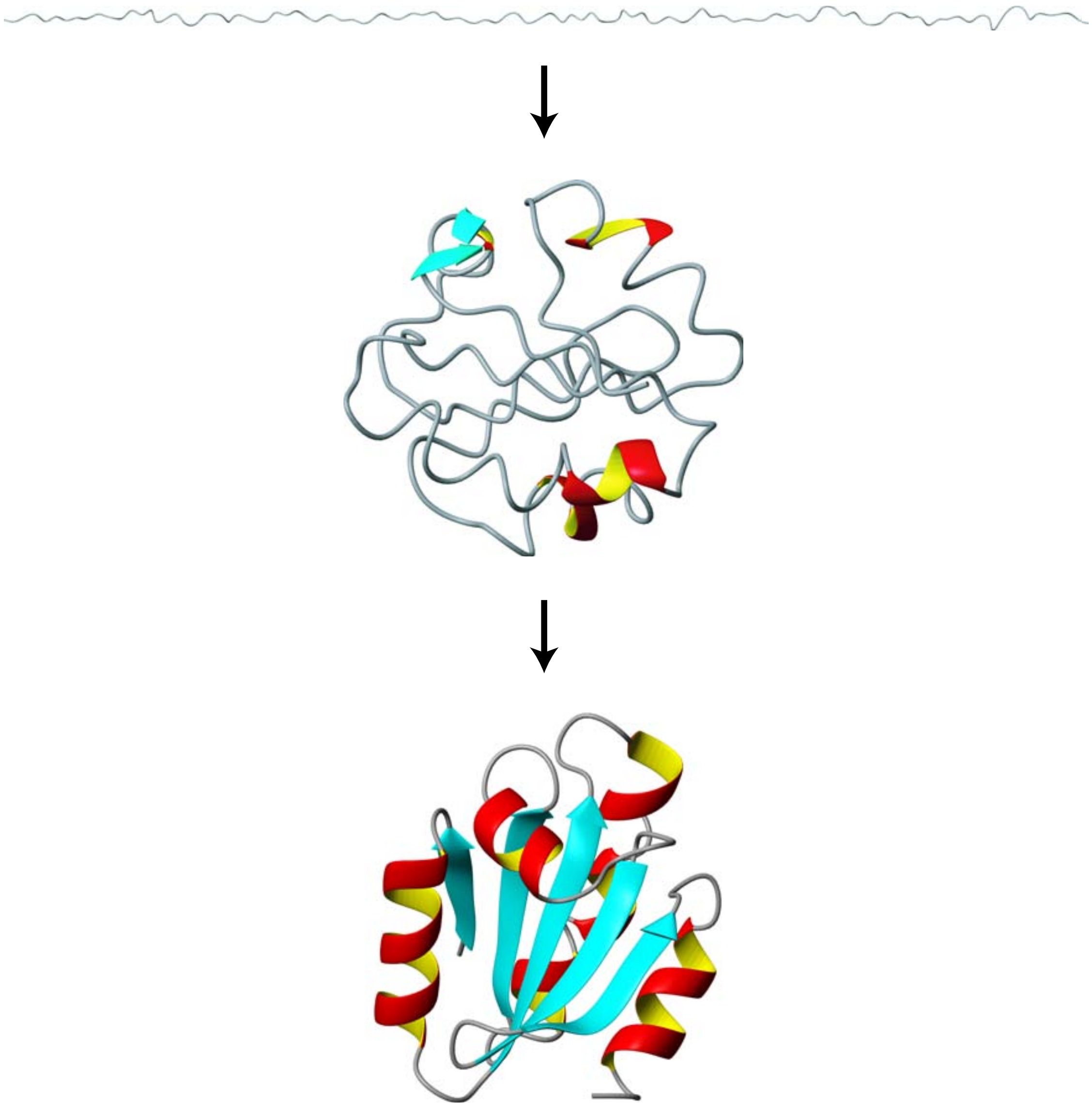

Figure 2.10: The first structure is the trace (N, C^{α}, and C') of a extended structure, the second is the trace of a distance geometry (DG) structure, and the third is the trace of a simulated annealing structure. The presented structures, ribbons, were made with MOLMOL [61]; the grey tube is loop, the red and yellow is α-helix, and the cyan is β-strand. In the beginning, the extended structure is made using the protein sequence, topology information of amino acids, and chemical properties of atoms and amino acids (*e.g.* bond lengths, angles, improper angles, masses, charges). The next step is to calculate the DG structure using given experimental constraints (*e.g.* NOEs, dihedral angles, and H-bonds). This structure mostly satisfies the given structural constraints, topology and chemical properties; however, it doesn't mean the DG structure is the minimum energy structure, moreover tremendous number of the DG structures can satisfy the structural constraints. The DG structure serves as good starting structure for simulated annealing. During simulated annealing, the structure is transformed to minimize the structural energy (Equation (2.5)).

- Distance geometry (DG): This method is based on a calculation of matrices of distance constraints for each pair of atoms from all available distance constraints, bond and torsion angles as well as van-der-Waals radii. This set of distances is then projected from the n-dimensional distance space into the three-dimensional space of a cartesian coordinate system, in which it determines the coordinates of all atoms of the proteins.

- Restrained Molecular Dynamics (rMD) / Simulated Annealing (SA): This is a molecular dynamics method, which takes place directly in the cartesian coordinate system. In this method, a starting structure is heated to a high temperature in a simulation (i.e. the atoms of the starting structure get a high thermal mobility). During many discrete cooling steps the starting structure can evolve towards the energetically favourable final structure under the influence of a force field (Eq. 2.5) derived from constraints.

Conventionally, a hybrid method is the chosen means of calculating structures [83]. Initial structures are generated by DG: in order to ease the computational burden, only a subset of the atoms may be included for large proteins, with the remainder added by reference to standard amino acid templates. The resulting structures have the correct global fold but poor local geometry, and are refined (annealed) using rMD. The annealing process removes many of the local violations of NMR restraints and covalent inconsistencies present in the DG structures.

The precision with which a structure can be calculated is directly related to the number of experimental restraint used to generate it. Structures of low resolution may be obtained with as few as five restraints per residue, whereas the most precise structures obtained from NOE constraints alone may have up to 15 restraints per residue.

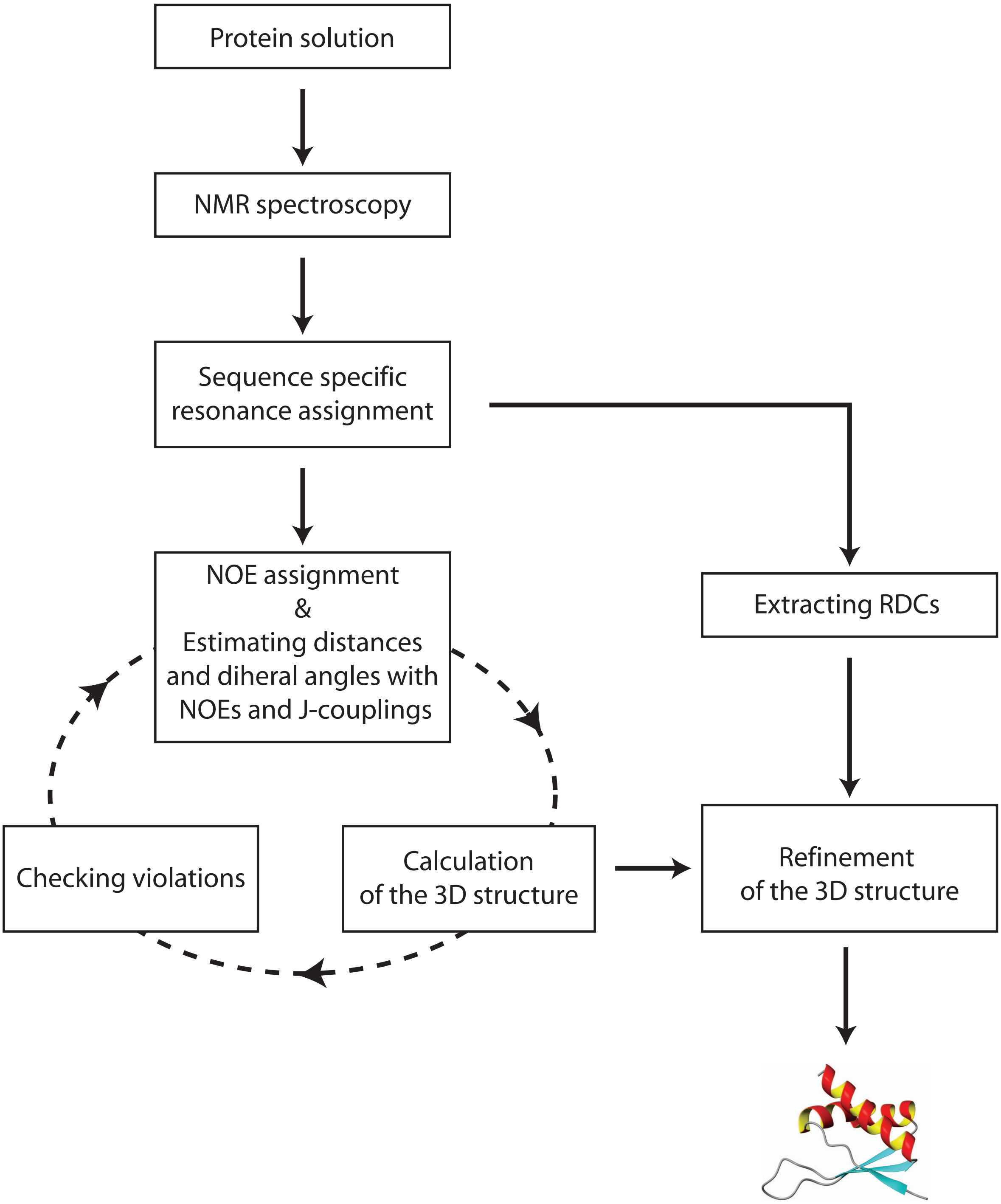
Protein solution
NMR spectroscopy
Sequence specific resonance assignment
NOE assignment
&
Estimating distances and diheral angles with NOEs and J-couplings
Extracting RDCs
Checking violations
Calculation of the 3D structure
Refinement of the 3D structure

Figure 2.11: Outline of the general strategy used to solve the three-dimensional structure of biological macromolecules in solution by NMR.

2.5 Automation of NMR Protein Structure Determination

One of the principal goals of automated structure determination is iterative analysis of multidimensional NOESY data, having the following steps in common: (i) Ambiguous proton-proton interactions from unassigned NOESY cross-peaks, together with unambiguously assigned proton-proton interactions, are incorporated into structure calculations and generate a new set of model structures. (ii) Ambiguous proton-proton interactions are iteratively trimmed using the resulting model structures if they are far apart in the inter-mediate model structures. The automated procedures follow the same general scheme but do not require manual intervention during the assignment/structure calculation cycles (In figure 2.11, the broken arrows indicate the cycle).

An automated approach starts without any prior knowledge of the structure, and then, in later cycles, uses the global fold of the structures generated in preceding cycles to assign and/or trim the ambiguous NOESY cross-peaks. Therefore, it is important to obtain a well-converged initial fold (rmsd < 3.0 Å, where the rmsd is the average backbone rmsd to the mean structure) for the rest of the cycles to achieve the correct structures. [9]

Chapter 3

Automatic Backbone Assignment of Proteins Using MARS

3.1 Introduction

The aim of the analysis of NMR spectra is to extract all available information about interatomic distances and torsion angles. The peaks, which are present in a NMR spectrum, come from resonances of spins of nuclei in frequency dimension; because atoms of a protein, which are located in different magnetic environments, give different resonances. In the initial stage of investigation by NMR spectroscopy each resonance must be associated with a specific nucleus in the investigated molecule. This process is called sequence-specific-resonance assignment.

Backbone resonance assignment is a prerequisite for structure determination of proteins by NMR [114]. Especially useful for backbone assignment are triple-resonance experiments on ^{13}C/^{15}N-labeled protein, such as HNCA, HN(CO)CA, HNCACB and CBCA(CO)NH or HN(CO)CACB. These experiments are the most sensitive triple-resonance experiments and they are also applicable to large deuterated proteins [14, 91]. They provide information on $^{1}\mathrm{H}^{N}{}_{i}$, $^{15}\mathrm{N}_{i}$, $^{13}\mathrm{C}^{\alpha}_{i}$, $^{13}\mathrm{C}^{\beta}_{i}$ chemical shifts of residue (i) and $^{13}\mathrm{C}^{\alpha}_{i-1}$, $^{13}\mathrm{C}^{\beta}_{i-1}$ chemical shifts of residue ($i-1$).

The chemical shifts are assembled into arrays called pseudo-residues, each of them associated with a single $^{1}\mathrm{H}^{N}$, ^{15}N root (a single resonance in a ^{15}N-^{1}H HSQC spectrum). Additional connectivity information, as obtained from experiments such as HNCO and HN(CA)CO, is also often included. In the assignment process these pseudo-residues are sequentially linked.

The connected segments are then mapped onto the known protein sequence based on the very sensitive relationship between amino acid type and $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ chemical shifts [77].

The assignment process is conceptually very simple and several algorithms have been developed in recent years to automate it. The different approaches can be grouped into two classes. The first group comprises numerical optimization algorithms that try to minimize a global pseudoenergy function or maximize a global 'goodness of fit'. These include simulated annealing [16, 10, 20, 71], threshold accepting [66], and neuronal networks [46]. The second class is based on best-first search strategies. Earlier implementations of the best-first approach were developed by Friedrichs *et al.* [33], Meadows *et al.* [73] and Olson and Markley [85]. The Montelione group expanded this strategy in their program AUTOASSIGN by propagating constraints from initial confident assignments towards later stages of the assignment process [121]. A similar approach is used by the program TATAPRO [8]. The program MAPPER by Güntert *et al.*[43] performs an exhaustive search to place connected segments onto the primary sequence and PACES performs an exhaustive search both for establishing sequential connectivity and for assignment [23].

Both strategies have their advantages and disadvantages. The problem of global optimization algorithms is that they can be trapped in local minima and assess only alternative complete assignments. Best-first strategies, on the other hand, are prone to propagation of errors made in the initial phases of the assignment process. Overall, good progress has been made in automation of backbone assignment for small to medium-sized proteins up to 20 kDa [77]. Especially for larger proteins, however, automation of resonance assignment is still difficult. Spectral overlap, chemical exchange or incomplete back-exchange of amide protons in deuterated proteins result in an incomplete set of resonances. These missing resonances severely deteriorate commonly used assignment algorithms. Therefore, for proteins above 20 kDa a significant fraction of manual assignment is still required.

Here we present MARS a program for robust automatic backbone assignment of $^{13}C/^{15}N$-labeled proteins. MARS simultaneously optimizes the local and global quality of assignment to minimize propagation of initial assignment errors and to extract reliable assignments. Using only $^{13}C^{\alpha}/^{13}C^{\beta}$ connectivity information, MARS allows automatic, error-free assignment of large proteins such as the 370 residue maltodextrin-binding protein (MBP). We demonstrate that MARS is highly robust against missing chemical shifts and reliably distinguishes correct from incorrect assignments. Additional information, such as residue specific information or known assignments from a homologues protein, can also be incorporated. MARS has been tested on 10 proteins ranging in size from the 71 residue Z domain of Staphylococcal protein A to 723 residue malate synthase G.

3.2 Methods

Resonance assignment of ^{13}C/^{15}N-labeled proteins is commonly performed using a five step analysis scheme: (1) pick and filter peaks, and reference resonances across different spectra; (2) group resonances into pseudo-residues (PRs); (3) identify the amino acid type of pseudo-residues; (4) find and link sequential pseudo-residues into segments; (5) map pseudo-residue segments onto the primary sequence. Step (1) and (2) are essential for manual assignment as well as for automatic approaches. Therefore, most NMR analysis software, like Felix [46], Aurelia [81], Xeasy [11], Sparky [60] and NMRView [55] provide tools for peak picking and referencing of multiple NMR spectra [11].

For assignment using MARS pseudo-residues should be generated using one of these programs. In principle, step (1) and (2) could also be performed automatically, however, the key to any successful assignment is reliable distinction between protein resonances and spectra noise. Therefore, in practice, 3D spectra, picked peaks and pseudo-residues are always inspected manually before starting the assignment process, as this can rapidly be done and the quality of picked peaks and pseudo-residues (or assignment strip) is crucial for successful assignment. The approach is further motivated by the fact that in most cases (especially for large proteins) assignment will be done semiautomatically, i.e. assignment results obtained by MARS will be refined visually on the screen.

Key features of MARS are:(1) simultaneous optimization of the local and global quality of assignment, (2) exhaustive search for fragment lengths comprising up to five PRs during linking and mapping, (3) best-first elements for both linking and mapping, (4) combination of the secondary structure prediction program PSIPRED [72] with statistical chemical shift distributions, which were corrected for neighboring residue effects [111], to improve identification of likely positions in the primary sequence and (5) assessment of the reliability of fragment mapping by performing multiple assignment runs with ‘noise-disturbed’ chemical shifts. The overall MARS strategy is outlined in Figure 3.1 and detailed below.

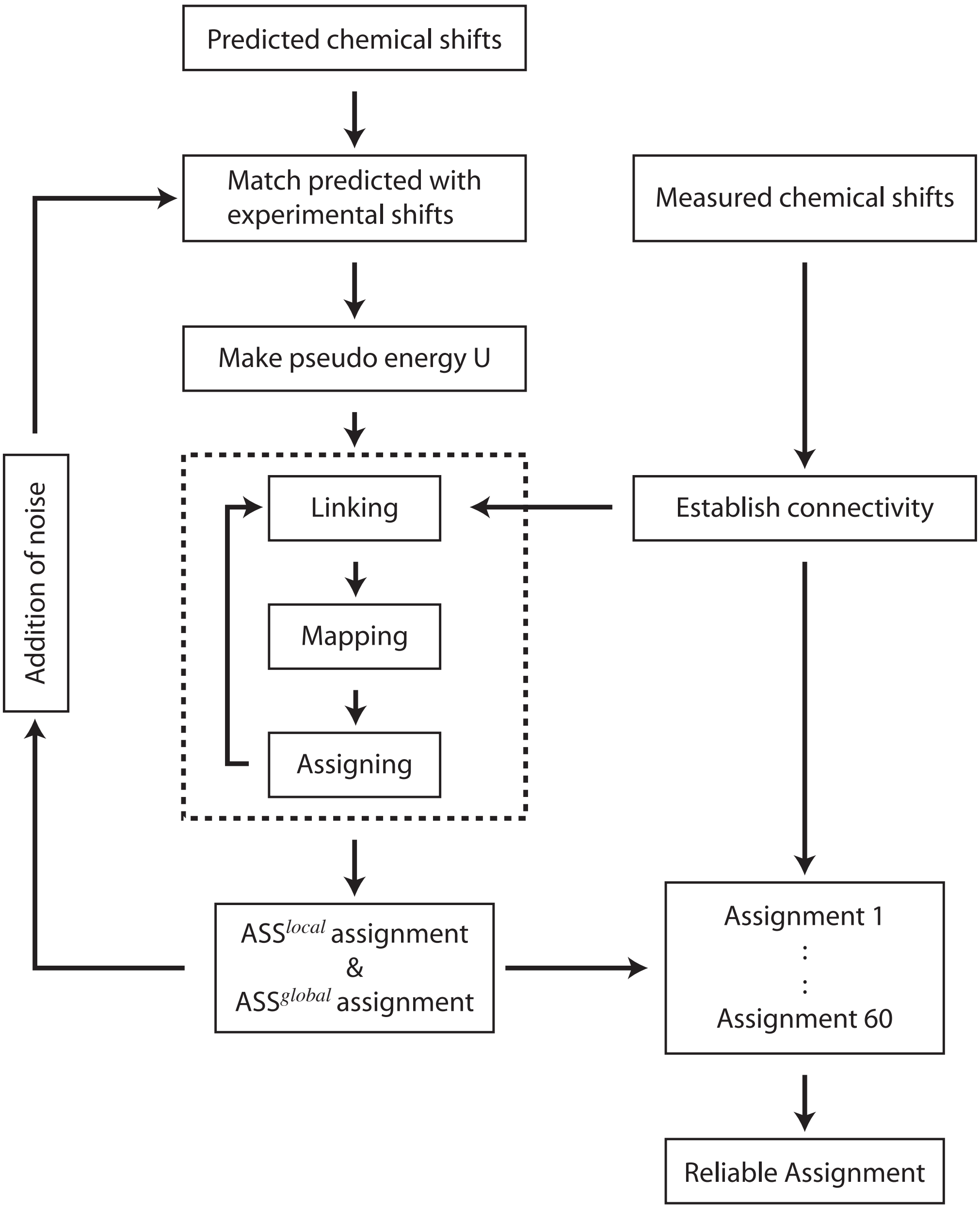

Figure 3.1: Overview of the MARS assignment procedure. See text for a definition of the two assignment solutions ASS^{local} and ASS^{global}.

3.2.1 Input Data

The input data for MARS consist of: 1) the primary sequence of the protein, (2) secondary structure prediction data (for example obtained from PSIPRED [72]), (3) an ASCII file that defines assignment parameters, such as the type of available information and chemical shift tolerances for establishing sequential connectivity, and (4) observed intra- and inter-residual chemical shifts grouped into pseudo-residues. A pseudo-residue (PR) comprises experimental chemical shifts that can be related to a single amino acid such as $\delta(\mathrm{H}^{N}{}_{i})$, $\delta(\mathrm{N}_i)$, $\delta(\mathrm{C'}_{i-1})$, $\delta(\mathrm{C}^{\alpha}_{i-1})$, $\delta(\mathrm{C}^{\alpha}_{i})$, $\delta(\mathrm{C}^{\beta}_{i-1})$ and $\delta(\mathrm{C}^{\beta}_{i})$ depending on the type of spectra available.

MARS does not perform peak picking, referencing of spectra or grouping of peaks into pseudo-residues. In our lab we use SPARKY [60] to perform these tasks. This allows visual control and refinement of pseudo-residues. When manually inspecting PRs, amide degeneracy can often be resolved, as peak shapes and the higher resolution in a 2D HSQC spectrum can be taken into account. If H^N/N overlap remains, multiple spin systems should be provided to MARS comprising the full set of possible combinations of peaks. In order to avoid an unreasonable high number of PRs in these cases, ambiguous peaks can also be partially discarded, as MARS does not favor pseudo-residues with more complete chemical shift information during the assignment process. The suspicious peaks can be reinserted when running MARS a second or third time, after an initial MARS run was performed, the assignment results were visually validated using SPARKY and verified assignments were fixed.

Besides $\mathrm{C}^{\alpha}/\mathrm{C}^{\beta}$ connectivity information, MARS can use sequential information from HNCO/HN(CA)CO and H^N-H^N NOESY spectra. Moreover, information about the amino acid type of a pseudo-residue can be included into MARS assignment. This information can come from a variety of sources, such as amino acid specific labeling [65, 87], backbone resonance experiments that select only signals from specific amino acids [30, 93] or amide peaks in a (H)C(CO)NH-TOCSY spectrum indicating methyl containing residues [35]. Information about the amino acid type of a pseudo-residue is most useful, when C^{α} and C^{β} chemical shift information is incomplete and for proteins above 40 kDa.

MARS not only allows restriction of possible amino acid types, the user can also fix a connectivity between two pseudo-residues. This is useful in an iterative approach, where a MARS assignment is refined manually on the screen, manually validated sequential connectivities are fixed and MARS is rerun with the reduced space of possible assignment solutions. Moreover, when assignment of a PR is known, *i.e.* the residue in the primary sequence of the protein that corresponds to the pseudo-residue has been identified, this assignment can be fixed.

3.2.2 Establishing Sequential Connectivity

In a first step, all possible sequential connectivities are detected. The approach taken in MARS is that initially each PR is assumed to be sequentially connected to every other PR and only connectivities not in agreement with experimental intra- and inter-residual chemical shifts are removed. Within the tolerance set for the individual nuclei, all matching shifts are equally accepted: there is no preference for the 'best match' to avoid a bias from insignificant chemical shift differences.

In addition, missing chemical shifts are not given a penalty, *i.e.* only when an atom type has chemical shift values for both pseudo-residues (in one case the intra-residual and in the other case the inter-residual chemical shift) and the difference between these two values is larger than the user-specified threshold the connectivity is deleted. This is especially important for assignment of proteins that miss chemical shifts for a substantial portion of residues.

Another important feature of MARS is that all pseudo-residues are used in all phases of the assignment procedure. PRs are not classified according to the number of chemical shifts they contain or the intensity of their corresponding NMR resonances. Therefore, PRs strongly affected by chemical exchange or by the presence of a paramagnetic ion can be fully utilized.

3.2.3 Matching of Experimental Chemical Shifts to the Protein Sequence

The second key step in assignment is to map segments that comprise sequentially linked pseudo-residues onto the primary sequence. Particularly useful in this respect is comparison of experimental C$^{\alpha}$ and C$^{\beta}$ chemical shifts with values that were obtained for each residue from a statistical analysis of chemical shifts deposited in the BMRB [29].

In MARS this process is further improved by using chemical shift distributions that are corrected for neighbor residue effects [111]. Besides the type of amino acid (and the type of neighbors in the primary sequence), however, chemical shifts very much depend on the type of secondary structure an amino acid is involved in. This is addressed in MARS by using the secondary structure prediction program PSIPRED [72] to identify regions in the protein sequence that are likely to be involved in regular secondary structure elements.

For each residue a theoretical chemical shift is calculated as the normalized sum of the random coil value and the value expected when this residue is involved in an α helix or a β strand. The probability of being in this secondary structure element, as identified by PSIPRED, is used as a weighting factor. Chemical shifts calculated in this way are of comparable quality as values predicted for proteins with known structure using the program SHIFTS [116]. If the protein under study is perdeuterated, MARS can be directed to adjust the calculated chemical shifts accordingly [107].

In order to map PR fragments onto the protein sequence, MARS calculates for all experimentally observed pseudo-residues the deviation of their experimental chemical shifts from predicted values according to

$$D(i,j) = \sum_{k=1}^{N_{cs}} \left\{ \frac{\delta(i)_k^{exp} - \delta(j)_k^{cal}}{\sigma_k} \right\}^2 \tag{3.1}$$

, where $\delta(i)_k^{exp}$ is the measured chemical shift of type k (e.g. $^{13}C^{\alpha}$ or $^{13}C^{\beta}$) of pseudo-residue i, $\delta(\mathrm{j})_{cal}^{k}$ is the predicted chemical shift of type k of residue j, N_{CS} is the number of chemical shift types and σ_k^2 is the variance of the statistical chemical shift distribution that is used for calculating $\delta(\mathrm{j})_{cal}^{k}$. Initially, the variances were obtained from standard deviations of the average random coil chemical shifts investigated by Wang and Jardetzky [111], then were optimized to improve assignment results. For $^{1}H^{N}$, ^{15}N, $^{13}C^{\alpha}$, $^{13}C^{\beta}$, ^{13}C' and $^{1}H^{\alpha}$ σ_k values of 0.82, 4.3, 1.2, 1.1, 1.7 and 0.82 ppm were used, respectively. In case a chemical shift of type k is missing, [$\delta(i)_k^{exp}$ - $\delta(\mathrm{j})_{cal}^{k}$l] is set to zero.

If calculation of chemical shifts from the protein sequence would be perfect, comparison with experimental values would be sufficient to complete assignment [41]. This, however, is not achievable with current prediction methods and additional connectivity information is required. In order to further increase the reliability of the mapping process, MARS does not rely directly on chemical shift deviations. Instead these values are converted into a pseudoenergy U(i, j) by ranking all residues j according to their chemical shift deviation (as calculated in equation (3.1)) with respect to pseudo-residue i. This makes MARS even more robust against unusual chemical shifts as not the exact fit of calculated to experimental chemical shifts is important, but the overall quality of the chemical shift fit.

3.2.4 Exhaustive Search for Establishing Sequential Connectivity and Mapping

At the start of a MARS assignment process all pseudo-residues are assigned randomly to the protein sequence. This information is stored as ASS^{local}. In order to refine ASS^{local}, MARS randomly selects a pseudo-residue. Starting from this PR it searches in the direction of the primary sequence (‘forward direction’) for all pseudo-residue segments of length five that can be assembled based on the available connectivity information. In the next step, all these N_{seg} segments are mapped onto all possible positions of the protein sequence. The probability that a fragment belongs to a specific position in the protein sequence is evaluated by calculating a summed pseudoenergy according to

$$U_i^m \sum_{k=i}^{i+n} U(k, j_i) \tag{3.2}$$

, where i is the number of the pseudo-residue that was randomly selected as the start of the segment, n is the length of the fragment (in this case $n = 5$), m is the fragment number ($m \in [1, N_{seq}]$) and j are the residue numbers to which pseudo-residues i to $i + n$ are tentatively assigned to (j is the starting position). Next, all are ranked.

The minimum identifies the best-fitting pseudo-residue segment, which starts with pseudo-residue i, and its corresponding position in the primary sequence. The information about this segment and the corresponding amino acid sequence is stored in SEG$_{for}$ and ASS$_{for}$, respectively. In order to validate this assignment, the same procedure is repeated but now starting from the last pseudo-residue of SEG$_{for}$ providing an additional assignment possibility (SEG$_{back}$/ASS$_{back}$). If SEG$_{for}$ = SEG$_{back}$, the assignment of the segment to the protein sequence is regarded as reliable and following approach is adopted to refine ASS^{local}. When SEG$_{for}$ = SEG$_{back}$ but ASS$_{for}$ $\neq$ ASS$_{local}$ the overall assignment is updated, *i.e.* ASS$_{for}$ $\rightarrow$ ASS^{local}.

In case of SEG$_{for}$ = SEG$_{back}$ and ASS$_{for}$ = ASS^{local}, this would have no effect. In order, however, to favor an assignment that is retained from previous assignment phases a penalty is given to all other assignments, which are possible for the PRs and residues that comprise SEG$_{for}$ and ASS$_{for}$. Thus, the total energy of the system is changed in such a way that the correct assignment is favored. When, on the other hand, SEG$_{for}$ $\neq$SEG$_{back}$, the suggested assignment solution is regarded as unreliable and ASS^{local} is kept unchanged. The whole optimization phase is repeated until all pseudo-residues have been used once as segment starting point.

So far, assignment has been optimized only with segments in which five PRs could be sequentially linked. The assignment is further refined in a second round, where the exhaustive search is restricted to segments in which four PRs are linked, then in a third and fourth round with tri- and dipeptide fragments. The procedure is conducted with decreasing fragment sizes based on the assumption that the longest matching segments have the greatest certainty of leading to correct assignments.

Finally, the whole phase comprising refinement of ASS^{local} by five, four, three and two PR segments is repeated four times. As each phase is based on pseudoenergies $\mathrm{U}(i,j)$ that were refined in the previous phase, the assignment procedure finally converges. All assignment results reported here comprised a total of five phases. The maximum segment length of five linked pseudo-residues is a compromise between the desired total execution time of a MARS assignment run and the ability to reliably place PR segments onto the protein sequence. When connectivity information from C$^{\alpha}$ and C$^{\beta}$ chemical shifts is available with an accuracy better than 0.5 ppm, MARS execution times for proteins as big as 370-residue maltose-binding protein are below 90 minutes on a single 1.7 GHz PC. At the same time, PR fragments with length five can in most cases be placed uniquely into the protein sequence when intra- and inter-residual C$^{\alpha}$ and C$^{\beta}$ chemical shifts are available.

3.2.5 Identification of Reliable Assignments

The algorithm described above results in a final optimized assignment ASS^{local}. This assignment is mainly driven by the local fit of fragments, comprising up to five pseudo-residues, to the protein sequence. In addition, however, pseudoenergy values $\mathrm{U}(i,j)$, which qualitatively describe the mapping of a single residue j to pseudo-residue i, have been changed during the process: This approach is similar to assignment algorithms where an energy function is optimized globally. Thus, a second assignment ASS^{global} can be extracted from $\mathrm{U}(i,j)$ at the end of the MARS assignment process. Each pseudo-residue i is assigned to that residue j for which $\mathrm{U}(i,j)$ is the minimum among all $\mathrm{U}(i,1)$, $\mathrm{U}(i,2)$, $\ldots$, $\mathrm{U}(i,N_{res})$ values. The two alternative assignment solutions, ASS^{global} and ASS^{local}, are compared and only consistent assignments are retained.

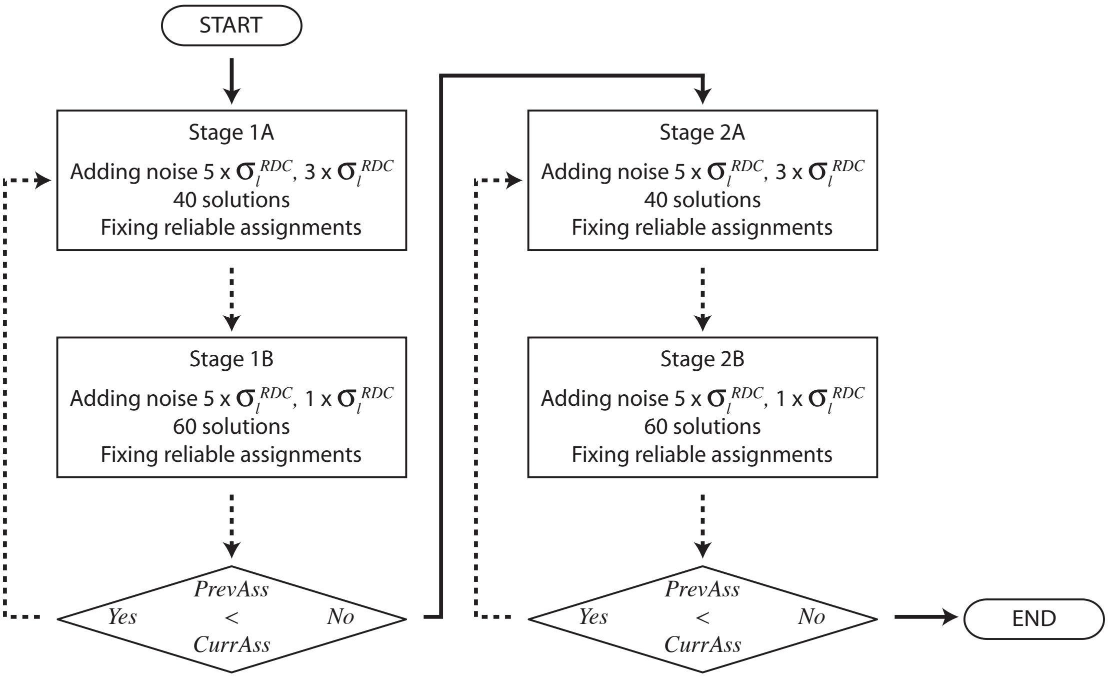
START
Stage 1A
Adding noise 5 x σ_l^{RDC}, 3 x σ_l^{RDC}
40 solutions
Fixing reliable assignments
Stage 1B
Adding noise 5 x σ_l^{RDC}, 1 x σ_l^{RDC}
60 solutions
Fixing reliable assignments
PrevAss < CurrAss
Yes
No
Stage 2A
Adding noise 5 x σ_l^{RDC}, 3 x σ_l^{RDC}
40 solutions
Fixing reliable assignments
Stage 2B
Adding noise 5 x σ_l^{RDC}, 1 x σ_l^{RDC}
60 solutions
Fixing reliable assignments
PrevAss < CurrAss
Yes
No
END

Figure 3.2: Empirically optimized scheme for avoiding errors due to inaccuracies in predicted chemical shifts when mapping pseudo-residue segments to the protein sequence. Stages 1A and 2A are identical except that the solution space is decreased when going from 1A to 2A due to assignments fixed in previous assignment stages. Stages 1B and 2B are also identical except that the amount of noise that is added to chemical shifts (which are calculated from the protein sequence) is decreased. σ_k is the standard deviation of the statistical chemical shift distribution that is used for calculating chemical shifts from the protein sequence. *PrevAss* and *CurrAss* is the number of assignments after stages A and B, respectively. Arrows indicate the program flow, i.e., if the number of assignments obtained from stage 1B (*CurrAss*) is larger than that from stage 1A (*PrevAss*) the program returns to stage 1A and reruns stage 1A but now with the reduced space of assignment solutions.

A major factor influencing the final assignment is the quality of chemical shifts predicted from the primary sequence as these values guide the mapping of PR segments to the protein sequence. To overcome this problem, MARS repeats the complete assignment process described above many times (Figure 3.2). For each assignment run predicted chemical shifts $\delta(j)_k^{cal}$ are modulated by addition of noise according to a Gaussian distribution. For the first 20 assignment runs, which generate a total of 40 assignment solutions (20 ASS^{global} and 20 ASS^{local} assignments), the width of this Gaussian is set to three times the standard deviation σ_k of the statistical chemical shift distributions.

By selecting assignments that are consistent across all 40 solutions, only the most reliable assignments are retained. These highly reliable assignments are fixed and the corresponding PRs and residues are excluded from future assignment runs. In subsequent assignment runs the amount of added noise is reduced according to an empirically optimized scheme (Figure 3.2). This gradually increases the number of consistent assignments. Thus, MARS uses best-first features both for establishing sequential connectivity (assignment is started with long connectivity segments) and for mapping PR segments onto the primary sequence (PR segments that are less affected by changes in calculated chemical shifts are mapped first).

3.2.6 Output Data

The output of MARS consists of different ASCII files: (1) 'assignment_AA.out', a file listing pseudo-residues assigned reliably to residues, i.e. the final assignment result, (2) 'assignment_AAs.out', an extended assignment including alternative assignment possibilities that show up with a 10 % probability, (3) 'assignment_PR.out', the most likely assignment for each pseudo-residue (this is useful in order to find out what is the most likely assignment for PRs that have not been assigned reliably to any residue), (4) 'connectivity.out', a summary of all possible sequential connectivities and (5) 'mars.log', which contains detailed information about predicted chemical shifts, number of reliable assignments, number of constraints for each pseudo-residue, matrices matching experimental and back-calculated chemical shifts and pseudoenergy matrices at each iteration step. In addition, chemical shift tables with updated assignments are stored ('sparky_all.out', 'sparky_CA.out', 'sparky_CA-1.out', 'sparky_CB.out', ...) that can directly be read into the analysis program SPARKY using the 'Read peak list' feature of SPARKY [60] and allow visual inspection of the assignment result.

Assigned pseudo-residues can be viewed as sequentially linked strips together with PRs that have not been assigned so far, alternative assignments can be evaluated on the screen using the information provided in the files 'assignment_AAs.out' and 'connectivity.out', and assignment suggestions for pseudo-residues that have not been assigned so far are provided in 'assignment_PR.out'. After validation on the screen safe assignments and sequential connectivities can be fixed and MARS can be rerun with the reduced space of possible assignment solutions.

3.2.7 Implementation

The core of MARS was written using the C programming language. This core is embedded into a shell script that uses the UNIX utility AWK for formatting of input and output files. This integrated approach has the advantage that improved programs for chemical shift prediction, chemical shifts from homologues proteins or chemical shifts from a previous assignment can easily be used.

3.2.8 Testing of MARS

MARS has been tested on 14 proteins ranging in size from the 71-residue Z domain of Staphylococcal protein A to 723-residue malate synthase G [3, 34, 36, 52, 69, 94, 101, 103, 105, 112]. Special focus was put on proteins that are challenging with respect to assignment either by their size or because chemical shifts are missing for a substantial portion of residues

(Table 3.1). MARS was tested primarily using only C$^\alpha$ and C$^\beta$ connectivity information as intra-residual carbonyl chemical shifts are most difficult to obtain experimentally due to the lower sensitivity of HN(CA)CO spectra. For selected proteins the effect of including C' connectivity information was evaluated and for ubiquitin the performance was tested using only C$^\alpha$ sequential connectivity. In addition, two threshold conditions for establishing connectivity were tested, namely 0.5, 0.5 and 0.25 ppm (condition I) and 0.2, 0.4 and 0.15 ppm (condition II) for C$^\alpha$, C$^\beta$ and C', respectively.

Table 3.1: Proteins and data quality used for testing MARS

Protein	BMRB code	# of residues	# of PRO/GLY	C^α_i / $C^\alpha_{i\text{-}1}$ (%) [a]	C^β_i / $C^\beta_{i\text{-}1}$ (%)[a]	C'_i / $C'_{i\text{-}1}$ (%)[a]	H^α_i / $H^\alpha_{i\text{-}1}$ (%)[a]
Malate synthase G	5471	723	31 / 51	95 / 95	94 / 94	94/95	--
Maltose binding protein	4354	370	21 / 29	96 / 96	95 / 96	--	--
Rous Sarcoma Virus capsid	4384	262	23 / 20	92 / 92	89 / 91	92 / 93	--
Human carbonic anhydrase I	4022	260	17 / 16	100 / 100	100 / 100	95 / 96	--
N-terminal domain of enzyme I (EIN)	4106	259	4 / 15	96 / 97	96 /97	--	--
E-cadherin domains II and III	4457	227	14 / 12	78 / 63	78 / 63	--	--
Human prion protein	4402	210	15 / 43	98 / 97	98 / 97	--	--
Superoxide dismutase	4341	192	8 / 14	64 / 64	62 / 63	48 / 61	--
Calmodulin/M13 complex	547	148	2 / 11	99 / 99	--	99 / 99	--
Profilin	4082	139	4 / 16	99 / 99	100 / 98	--	--
E. *coli* EmrE	4136	110	5 /12	86 / 84	57 / 60	73 / 77	--
Human ubiquitin	--	76	3 / 6	100 /100	100 / 100	--	--
Z domain	--	71	3 / 0	90 / 96	51 / 82	--	89 / 100
Tir110		110	12 / 15	100 / 100	100/ 100	100 / 100	--

[a] Percentage of available chemical shifts of a given type.

Chemical shifts were taken from the BMRB data base [29], with all HN and N chemical shifts entered as spin-systems and with the carbon chemical shifts of the preceding residue entered as inter-residue chemical shifts. To put MARS to a more rigorous test, we also started from raw peak lists obtained from automatic peak picking of NMR spectra recorded on Z domain of Staphylococcal protein A. These raw peak lists were taken from the distribution package of the AUTOASSIGN software [121]. Pseudoresidues for testing of MARS were generated from these peak lists by reading them into AUTOASSIGN and using the Create Ladders' feature. This produces the generic spin system objects (GS) that are equivalent to pseudo-residues in MARS. Overlapping GSs/PRs are thereby automatically separated [121].

In addition, MARS was applied to the assignment of the fully unfolded, soluble N-terminal 110-residues of intimin receptor Tir (Tir110). 3D HNCA, CBCA(CO)NH, HNCACB, HNCO and HNCACO experiments were collected on a Bruker DRX800 spectrometer and processed using NMRPipe [28]. Calibration of spectra, peak picking and grouping of peaks into pseudo-residues was done using SPARKY [60]. Pseudoresidues were saved to an ASCII file using the 'Save Assignment table' feature of SPARKY and read into MARS without further modification.

For proteins that lacked experimental data the robustness of MARS against missing chemical shifts was tested by random removal of entire pseudo-residues as well as deletion of certain chemical shifts within the pseudo-residues. In addition, it was evaluated how chemical shifts that are outside the connectivity threshold δ due to peak overlap or distortion (although in reality they are sequentially connected) affect automatic assignment by MARS. For this, random noise $d = N(0, \delta/2.5)$ was added to each inter-residual chemical shift, where $N(\mu, \sigma)$ represents a random variable of normal density with mean μ and standard deviation σ. In this way, about 2-3% of connectivities were affected (condition III). For the N-terminal domain of enzyme I of the phosphoenolpyruvate the percentage of wrong inter-residual chemical shifts was further increased up to 50%. This corresponds to $d = N(0, \delta/1.1)$.

In all tests assignment was performed by MARS without manual intervention and the results are reported in Table 3.2. Running times (not CPU times) on a 1.7 GHz Linux PC varied from about 30 seconds for ubiquitin to about 90 minutes in case of maltose-binding protein (only C$^{\alpha}$, C$^{\beta}$ connectivity with a common threshold of 0.5 ppm). For malate synthase G running times vary from two hours (C$^{\alpha}$, C$^{\beta}$ and C' connectivity with thresholds of 0.2, 0.4 and 0.15 ppm, respectively) to 13 hours (only C^{α} and C^{β} connectivity with thresholds of 0.2 and 0.4 ppm, respectively) and up to 150 hours when only C^{α} and C^{β} connectivity information is available with a resolution of 0.5 and 0.5 ppm, respectively.

3.3 Results and Discussion

3.3.1 Small Proteins

76-residue ubiquitin serves as a first basic test case. Using C^α/C^β connectivity information all 72 non-proline residues (excluding the N-terminus) could be assigned correctly and reliably for both threshold conditions. When only C^α chemical shift information was used, the total number of correct assignments dropped to 32 and 9 were identified as reliable. This rather strong decrease in reliable assignment is expected due to the higher degeneracy and the less precise determination of amino acid types in the absence of C^β chemical shifts. Only if fragments are sufficiently long or if they contain residues with very characteristic C^α chemical shifts, such as glycines, a mapping to the sequence is identified as reliable by MARS. However, none of the nine reliable assignments was wrong.

MARS was further tested on the 67 pseudo-residues of Z domain of Staphylococcal protein A as obtained from raw peak lists [121]. The number of pseudo-residues agrees with the expected number taking into account the three prolines and the N-terminal amino acid, i.e. no additional, spurious PRs are present. For 19% of Z domain's PRs the HN/N root frequencies partially overlap and 90% of all expected intra-residual C^α chemical shifts are present. However, C^β connectivity information is far from complete with only 51% of all expected intra-residual C^β chemical shifts available. Employing a common connectivity threshold of 0.5 ppm for both C^α and C^β MARS assigned 34 PRs reliably. In addition, the correct assignment was indicated for another 23 pseudo-residues, providing valuable starting points for manual assignment. Upon inclusion of H^α connectivity information the number of assignments was raised to 65 with no errors present.

Table 3.2: MARS assignment results for proteins of varying size and data completeness

Protein	# of residues with data [a]	Used chemical shifts	Condition I [b]		Condition II [c]		Condition III [d]	
			Assignment #		Assignment #		Assignment #	
			All [e]	Reliable / Errors [g]	All [e]	Reliable / Errors [g]	All [e]	Reliable / Errors [g]
Malate synthase G	654	C’, Cα, Cβ	652	639 / 0	652	639 / 0	651	623 / 0
		Cα, Cβ [f]	500	207 / 0	639	584 / 2	622	511 / 0
Maltose binding protein	335	Cα, Cβ	323	303 / 0	333	324 / 0	330	313 / 1
Rous Sarcoma Virus capsid	221	C’, Cα, Cβ	214	205 / 0	218	207 / 0	218	199 / 0
Human carbonic anhydrase I	243	C’, Cα, Cβ	242	235 / 0	242	237 / 0	242	225 / 0
N-terminal domain of enzyme I (EIN)	248	Cα, Cβ	246	232 / 0	246	246 / 0	248	245 / 0
E-cadherin domains II and III	167	Cα, Cβ	116	77 / 1	134	102 / 0	136	70 / 1
Human prion protein	190	Cα, Cβ	138	103 / 0	155	127 / 0	154	118 / 0
Superoxide dismutase	117	C’, Cα, Cβ	112	101 / 0	112	104 / 0	111	100 / 0
		Cα, Cβ	111	101 / 0	112	104 / 0	112	103 / 0
Calmodulin/M13	144	Cα, C’	97	37 / 0	144	142 / 0	136	119 / 0
Profilin	132	Cα, Cβ	130	132 / 2	132	132 / 0	132	123 / 0
E. *coli* EmrE	74	C’, Cα, Cβ	61	35 / 0	70	58 / 0	64	50 / 0
Human ubiquitin	72	Cα, Cβ	72	72 / 0	72	72 / 0	72	70 / 0
		Cα	32	9 / 0	58	18 / 0	58	9 / 0
Z domain	67	Cα,Cβ, Hα [h]	65	65 / 0	--	--	--	--
		Cα, Cβ [i]	57	34 / 0	--	--	--	--
Tir110 [j]	97	C’, Cα, Cβ	91	80 / 0	--	--	--	--

[a] Includes only those residues for which HN and N chemical shifts were reported.

[b] Condition I: 0.5, 0.5 and 0.25 ppm are used for establishing connectivity for Cα, Cβ and C’, respectively.

[c] Condition II: 0.2, 0.4 and 0.15 ppm are used for establishing connectivity for Cα, Cβ and C’, respectively.

[d] Condition III: Same as condition II but with simulated error.

[e] # of correct assignments in Ass^{global} ; Ass^{global} was obtained from a MARS run without addition of noise.

[f] The maximum length of pseudoresidue segments, which were searched exhaustively, was four (instead of five).

[g] Assignments that were identified as reliable but are incorrect, i.e. the number of errors.

[h] Experimental data. Connectivity thresholds of 0.5, 0.7 and 0.05 ppm were used for Cα, Cβ and H^{α}, respectively.

[i] Experimental data. Connectivity thresholds of 0.3 and 0.5 ppm were used for Cα and Cβ, respectively.

[j] Experimental data. Connectivity thresholds of 0.2, 0.5 and 0.25 ppm were used for Cα, Cβ and C’, respectively.

3.3.2 Partially and Completely Disordered Proteins

In case of the 210-residue full-length human prion protein, the N-terminal half (residues 1-125) is completely disordered. This results in a very narrow chemical shift dispersion, severe degeneracy and poses a significant challenge to sequential assignment. Using only C^{α}/C^{β} chemical shifts for establishing connectivity (with a common threshold of 0.5 ppm) MARS assigned 138 out of 190 available pseudo-residues correctly and 103 of these were identified as reliable. All assignments identified as reliable were correct, *i.e.* 103-residues were assigned by MARS without false positives. When the threshold was reduced to 0.2 and 0.4 ppm for C^{α} and C^{β}, respectively, the number of reliable and correct assignments increased to 127, i.e. an assignment score of 67%.

Similar, high quality results were obtained using experimental chemical shift lists that were prepared from triple-resonance spectra recorded on the completely unfolded, soluble N-terminal 110-residues of intimin receptor Tir. Using C^{α}, C^{β} and C' chemical shifts MARS assigned 80 out of 97 experimental pseudo-residues and indicated the correct assignment for a total of 91 PRs (Table 3.2). Based on the 80 reliable assignments and the assignment suggestions provided by MARS for the remaining PRs, the assignment could be quickly completed by visual inspection of assignment strips (pseudo-residues) using SPARKY.

3.3.3 Big Proteins

N-terminal domain of enzyme I of the phosphoenolpyruvate (EIN), human carbonic anhydrase I, rous sarcoma virus capsid, maltose-binding protein (MBP) and malate synthase G (MSG) are challenging for assignment due to their size of 259, 260, 262, 370 and 723-residues. With C^{α} and C^{β} chemical shifts at an accuracy of better than 0.2 and 0.4 ppm, respectively, 99% of EIN and 97% of MBP could be assigned reliably and for almost 100% the correct assignment was indicated. For 723-residue MSG 89% of pseudo-residues could be assigned, however, two of these were wrong. Inclusion of C' connectivity removed the two errors and increased the reliable assignment score to 98%.

Whereas in case of C^{α}, C^{β} and C' connectivity information the number of possible connectivities for each pseudo-residue is 1.02 on average (note that this is just an average value), it is raised to 4.48 when C' connectivity is not available. Therefore, it was necessary to reduce the maximum fragment length, which is searched exhaustively during the linking process, to four PRs and it still took several days to complete the assignment process on MSG. In case of condition III, 207 pseudo-residues of MSG were assigned reliably, out of a total of 500 correct ones, and not a single reliable assignment was wrong. The long duration of the assignment process for such difficult cases can significantly be shortened if MARS is run on several PCs in parallel on a Linux cluster.

3.3.4 Proteins with Incomplete Chemical Shift Data

EmrE, superoxide dismutase and E-cadherin are missing HN/N chemical shifts for a substantial portion of their residues. For superoxide dismutase only 61% (117 PRs) of expected pseudo-residues (183 PRs) were observed in triple-resonance NMR spectra as a result of paramagnetic relaxation of residues in the vicinity of an Fe3+ ion. In addition, about half of the available PRs are scattered throughout the length of the protein, separated by numerous small gaps. MARS was able to efficiently handle these difficult cases and assigned 101 out of 117 pseudo-residues reliably using only C^{α} and C^{β} connectivity information (threshold of 0.5 ppm for both). Including C' data or reducing the thresholds to 0.2 and 0.4 ppm for C^{α} and C^{β}, respectively, did not significantly affect the assignment score.

3.3.5 Required Chemical Shift Data and Thresholds for Establishing Connectivity

Most automatic assignment programs require sequential connectivity information for assignment, and it important to reduce the ambiguity of sequential connectivities for the successful assignment. Therefore the programs require a specific set or resolution of NMR spectra [122, 23]. MARS less depends on sequential connectivity information in terms of very low error rate in reliable assignments and, even, does not necessarily require the sequential connectivity information for the assignment. It makes MARS highly flexible. Whether only C^α or C^α, C^β, C' and H^α connectivity information is available, assignments identified by MARS as reliable will have a very low to zero error rate. When only C^α chemical shifts are available, reliable assignment is restricted to very small proteins with very complete data. With C^α and C^β information available for more than 80% of residues and with an accuracy better than 0.2 and 0.4 ppm, respectively, an assignment score of more than 95% is possible without errors.

For proteins above 40 kDa or less complete or more degenerate data it is highly useful to have access to additional C' connectivity information. Assignment is less susceptible to errors (see results on malate synthase G) and thresholds for establishing sequential connectivity have to be less tight. For example, for superoxide dismutase and malate synthase G similar results are obtained with thresholds of 0.5, 0.5, 0.25 ppm and 0.2, 0.4, 0.15 ppm for C^α, C^β and C', respectively. This is especially important, as overlap and weak resonances often require higher connectivity thresholds as anticipated on the basis of the digital resolution of the NMR spectra. In addition, the reduced degeneracy for establishing sequential connectivity significantly shortens execution times of MARS.

3.3.6 Robustness against Missing Data

When chemical shift information is close to complete and NMR spectra were recorded with a resolution better than 0.2, 0.4 and 0.15 ppm for C^{α}, C^{β} and C', respectively – as for ubiquitin, calmodulin or EIN – MARS allows automatic assignment of 99 to 100% of observed pseudo-residues. Such favorable situations, however, are rarely encountered in real applications. More important is, therefore, the reliability of the assignment procedure in case of incomplete chemical shift data. Table 3.2 shows that only for some selected test cases one or two reliable assignments were wrong. In all other situations assignments labeled as reliable by MARS were correct (*i.e.* zero error rate).

The robustness of MARS was further tested by randomly deleting a fraction of the observed pseudo-residues. Random deletion of pseudo-residues is particularly challenging as it introduces many gaps into the sequential connectivity path. Removing 10% of EIN's pseudo-residues decreased the reliable assignment from 95% to 78% (Figure 3.3). However, the assignment remains without error. When 20 or 30% of pseudo-residues are removed the number of reliable assignments is further reduced to 122 and 89 (out of a total of 204 and 178 remaining pseudo-residues of EIN, respectively). For MBP, on the other hand, the percentage of reliable assignments dropped to 30% when 30% of pseudo-residues were randomly deleted. This strong decrease is expected due to the large size of maltose-binding protein.

However, even in such a challenging situation the number of assignment errors is kept at a minimum. Both for EIN and MBP the number of errors is always less than three (zero for MBP, three for EIN at 30% randomly deleted pseudo-residues). In addition, for many proteins missing data are concentrated into a specific region of the protein sequence, such as for EmrE where NMR data for residues 32 to 76 are missing. This is less problematic than random deletion, as reliable assignment can be obtained efficiently for the remainder of the sequence.

The robustness of MARS against missing data was also tested by randomly deleting chemical shifts within pseudo-residues of EIN. Similar to the case where complete PRs are

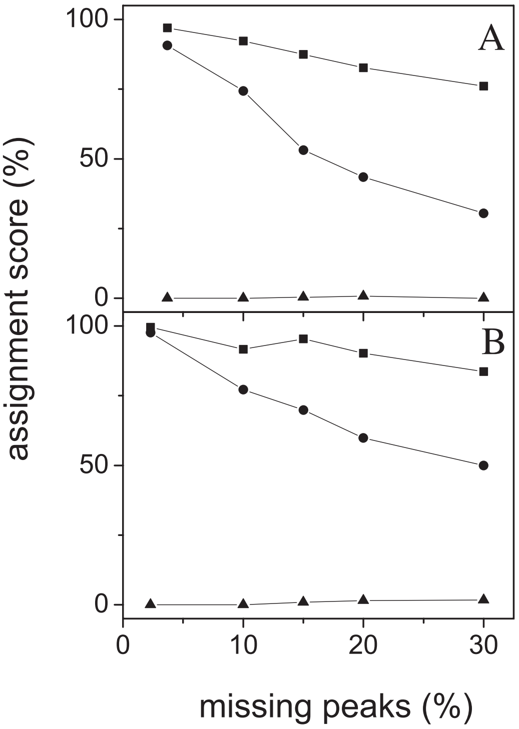

Figure 3.3: Dependence of MARS assignment on the percentage of missing pseudo-residues. Pseudoresidues were deleted randomly. ■ indicate the percentage of all assignments that were correct (not tested for reliability). ● show the percentage of residues that could be assigned reliably (relative to the total number of assignable residues) and ▲ indicate assignments that were identified as reliable but are wrong, *i.e.*, the error rate of MARS. $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ chemical shifts with a common threshold of 0.5 ppm for establishing sequential connectivity were used. (A) Results for the 370-residue maltose-binding protein. (B) Results for the 259-residue N-terminal domain of enzyme I. Note the very small to zero error rate.

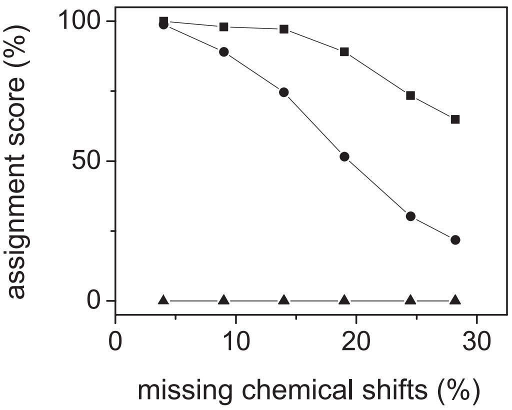

Figure 3.4: Dependence of MARS assignment on the percentage of missing chemical shifts within pseudo-residues for the 259-residue N-terminal domain of enzyme I. Chemical shifts were deleted randomly. ■, ● and ▲ indicate correct, reliable and wrong reliable assignments, respectively. $^{13}C^{\alpha}$ and $^{13}C^{\beta}$chemical shifts with thresholds of 0.2 and 0.4 ppm for establishing sequential connectivity were used.

deleted, the number of overall correct assignments remained almost unchanged up to 15% missing chemical shifts (Figure 3.4). For even more incomplete data the assignment score started dropping and ended up at 65% when 28% of chemical shifts were removed. At the same time the number of reliable assignments reduced more quickly with an assignment score of 52% for 19% missing chemical shifts. In agreement with the tests where complete pseudo-residues were removed, assignments termed reliable by MARS are indeed very reliable with zero errors even at 30% missing chemical shifts.

The low error rate of MARS is sometimes a trade-off with the completeness of assignment. For example, for ubiquitin (using only C^{α} chemical shifts with a threshold of 0.2 ppm) 58 assignments were correct, but only 18 were identified as reliable (Table 3.2). MARS, however, should be used together with analysis software that allows visual inspection, such as SPARKY, and the 58 correct assignments of ubiquitin provide a very valuable starting point to manually complete assignment. In addition, they can give hints on what additional information, such as selective labeling, is required.

3.3.7 Robustness against Chemical Shifts Outside the Connectivity Threshold

The connectivity information provided by inter- and intra-residual chemical shifts is an essential component of the assignment process. At the same time, however, peaks are often distorted or overlapped and corresponding chemical shifts fall outside the connectivity thresholds. The effect of chemical shift errors was tested by addition of noise to each inter-residual chemical shift, such that about 2 - 3% of connectivities were affected. For all tested proteins the overall assignment scores were virtually unchanged upon introduction of the distorted chemical shifts (Table 3.2).

In addition, the reliable assignments were only slightly affected. The strongest decreases in the number of reliable assignments were seen for E-cadherin and the calmodulin/M13 complex. For E-cadherin this can be attributed to the high number of missing chemical shifts and the fact that only C^{α} and C^{β} chemical shift information was available (Table 3.1). For superoxide dismutase, on the other hand, where even more pseudo-residues and C^{α} and C^{β} chemical shifts are missing, the assignment is almost unchanged due to the availability of C' chemical shifts (Table 3.2). This demonstrates that using slightly too tight connectivity thresholds is not problematic for MARS. For EIN we further took these tests to the extreme by strongly increasing the amount of added noise such that up to 45% of sequential connectivities were lost (Figure 3.5).

As long as less than 15% of inter-residual chemical shifts were outside the connectivity thresholds both the overall and the reliable assignment scores remained high. Only when even more chemical shifts were corrupted the number of assignments started to rapidly decrease. However, even when 45% of connectivities were lost (corresponding to 50% of chemical shifts outside the connectivity thresholds) only a single reliable assignment was wrong.

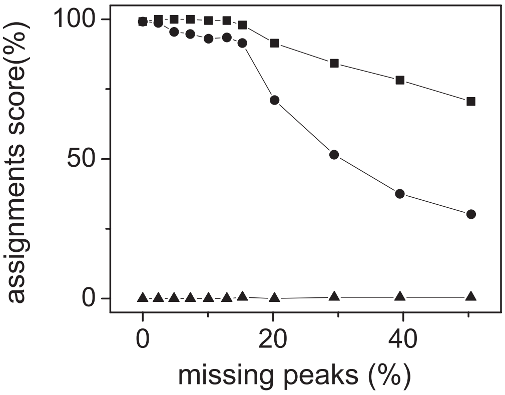

Figure 3.5: Dependence of MARS assignment on the percentage of chemical shifts falling outside the connectivity thresholds for the 259-residue N-terminal domain of enzyme I. Connectivity thresholds were 0.2 and 0.4 ppm for $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ respectively. ■, ● and ▲ indicate correct, reliable and wrong reliable assignments, respectively.

3.4 Concluding Remarks

We have introduced a software for backbone assignment of proteins that can be applied independent of the assignment complexity, that does not require tight thresholds for establishing sequential connectivity or detailed adjustment of these thresholds, that uses always all available data during the assignment process and that does not require a specific set of NMR experiments.

The key for any automatic assignment is that one can trust the answer the program returns. When the amount and quality of available information is poor, this will always result in a decrease in the number of assignments that will be regarded as reliable, independent of whether the assignment is performed manually or automatically. In these difficult cases MARS retains a good assignment score and, at the same time, assignments that are identified as reliable are almost always correct.

Compared to other currently available programs MARS is applicable to proteins above 15 kDa using only C^{α} and C^{β} chemical shift information with connectivity thresholds as high as 0.5 ppm and it is applicable to proteins with very high degeneracy such as partially or fully unfolded proteins. It offers improved assignment scores for proteins where data are missing for a substantial portion of residues and it has a good tolerance against erroneous chemical shifts.

MARS assignment results can be directly read into the program Sparky [60]. This allows visual validation of the assignment results. Thus, several cycles of automatic assignment using MARS and manual validation on the screen can be performed, in order to complete assignment even in difficult cases.

Chapter 4

Automatic Backbone Assignment of Proteins with Known Structure Using Residual Dipolar Couplings

4.1 Introduction

NMR spectroscopy is a powerful tool to study protein-ligand binding, protein-nucleic acid interactions and protein dynamics. A prerequisite for these studies is assignment of NMR spin resonances [114]. In recent years, good progress has been made in automating the assignment process for proteins up to 20 kDa [77] and we have introduced the program MARS that allows robust automatic backbone assignment also for unfolded and large proteins [56].

Most assignment approaches, such as MARS, rely on methods to connect NMR resonances related to single residues into segments and to map these segments onto the known protein sequence based on the very sensitive relationship between amino acid type and chemical shifts [42, 77, 100]. The accuracy of chemical shifts, calculated with current methods from the protein sequence or even from a known 3D structure, is, however, not sufficient, to assign error-free and unambiguously connectivity segments to the protein sequence. This is especially problematic for big proteins and proteins where a significant fraction of data is missing, independent of whether the assignment is performed manually or automatically. To avoid assignment errors in these cases, more conservative approaches have to be taken when connectivity segments are mapped onto the primary sequence. This generally results in a decrease in the number of residues that can be assigned reliably [56].

When a three-dimensional (3D) structure of the protein is known already, comparison of NMR parameters back-calculated from this structure with experimental values can potentially be used to improve the assignment process. So far, most studies have focused on incorporation of Nuclear Overhauser Effect (NOE) distance constraints into the assignment process: only assignments that are consistent with distances observed in the 3D structure are allowed [12, 16, 90].

Recently, it was shown that also residual dipolar couplings (RDCs) are very useful for resonance assignment. If no 3D structure is available, RDCs can be used to reduce chemical shift degeneracies in sequential connectivity experiments [127]. In case of small proteins, they even allow simultaneous resonance assignment and structure determination [102]. On the other hand, calculation of RDCs from a known 3D structure is straightforward and has been used previously for validation of protein structures [13, 89]. Therefore, an assignment method for proteins can be envisioned where dipolar couplings calculated from a known 3D structure are compared to experimental values.

Initially, such an approach was described for RNA [2]. Hus *et al.* extended this strategy recently to proteins [50]. Prestegard and coworkers, on the other hand, employed a manual approach where they assigned five peaks of the human ADP ribosylation factor 1, which could not be assigned using triple-resonance experiments, by matching predicted $^1\mathrm{D}_{NH}$ couplings with experimental values [5].

None of these approaches, however, allows simultaneous use of sequential connectivity information and RDCs, or provides an indication on how reliable an assignment obtained by RDC matching is.

Here we show that RDCs can be routinely included into backbone assignment of proteins with known structure using the program MARS. In case of small proteins, MARS allows RDC-based assignment of more than 90% of backbone resonances without the need for sequential connectivity information. For bigger proteins, we demonstrate that assignment can significantly be enhanced by combining RDC matching with sequential connectivity information and that inaccuracies in the 3D structure do not result in an increased number of assignment errors.

4.2 Methods

The assignment algorithm employed in MARS has been described in detail in the chapter 3. For RDC-enhanced assignment, this algorithm is extended as described below.

4.2.1 Input and Output Data

When dipolar couplings are to be used for assignment, a PDB file of a known 3D structure or homology model of the protein has to be supplied as input to MARS and the resolution of the structure has to be indicated. In addition, pseudo-residues comprise experimental chemical shifts and RDCs. One-bond RDCs are commonly measured from triple-resonance experiments, such as HNCO, [14, 13, 89] and it is therefore straightforward to add these RDCs to pseudo-residues.

Besides the standard output provided by MARS, an alignment tensor is returned that has been optimized during the assignment process together with RDCs back-calculated from the 3D structure.

4.2.2 Matching of Experimental RDCs to Back-calculated Values

When the 3D structure of the protein is unknown, mapping of single pseudo-residues (PR) or of segments connecting several pseudo-residues relies on comparison of experimental chemical shifts with values calculated from the protein sequence. This could potentially be improved when chemical shifts are calculated from the 3D structure. However, chemical shifts, which were calculated from the protein sequence with the use of correction factors for neighbor residue effects and secondary structure prediction information [56], are of comparable quality as values predicted for proteins with known structure using the program SHIFTS [117, 116]. Therefore, the assignment performance of MARS was not improved when using chemical shifts calculated with SHIFTS.

In order to include RDCs into the process of mapping PR segments onto the protein sequence, MARS calculates for all experimentally observed pseudo-residues the deviation of their experimental RDCs and chemical shifts from predicted values according to

$$D(i,j) = w \sum_{k=1}^{N_{CS}} \left\{ \frac{\delta(i)_k^{\exp} - \delta(i)_k^{cal}}{\sigma_k^{CS}} \right\}^2 + \sum_{k=1}^{N_{RDC}} \left\{ \frac{RDC(i)_l^{\exp} - RDC(i)_l^{cal}}{\sigma_l^{RDC}} \right\}^2 \qquad (4.1)$$

, where $\delta(i)_k^{exp}$ is the measured chemical shift of type k (e.g. ^{13}C$^{\alpha}$ or ^{13}C$^{\beta}$) of pseudo-residue i, $\delta(j)_k^{cal}$ is the predicted chemical shift of type k of residue j, N_{CS} is the number of chemical shift types and σ_k^{CS} is the standard deviation of the statistical chemical shift distribution that is used for calculating $\delta(j)_k^{cal}$. For ^{1}H^N, ^{15}N, ^{13}C$^{\alpha}$, ^{13}C$^{\beta}$, ^{13}C' and 1H$^{\alpha}$, σ_k^{CS} values of 0.82, 4.3, 1.2, 1.1, 1.7 and 0.82 ppm were used, respectively [56].

Similarly, $RDC(i)_l^{exp}$ is the experimental RDC of type l (e.g. ^{1}D$_{NH}$ or ^{1}D$_{CaC'}$) of pseudo-residue i, $RDC(j)_l^{cal}$ is the back-calculated RDC of type l of residue j, N_{RDC} is the number of RDC types and σ_l^{RDC} is the value used for normalizing RDC deviations. w is a weighting factor that takes into account the different reliability of calculated chemical shifts and RDCs. As back-calculated RDCs are directly influenced by structural and dynamical deviations from the PDB coordinates, empirical optimization resulted in $w = 3.3$, thereby downscaling the contribution of RDCs.

The RDC normalization constant σ_l^{RDC} is adjusted according to the resolution, R_{struc}, of the 3D structure. Figure 4.1 compares the normalized root-mean-square-deviation between experimental RDCs and values back-calculated from known crystal structures using SVD (based on published assignments) for 31 crystal structures (Table 4.2.2). Based on the slope of the linear fit shown in Figure 4.1, c_{RDC},

$$\sigma_l^{RDC} = c_{RDC} R_{struc} D_a^{HN} \qquad (4.2)$$

, where D_a^{HN} is the magnitude of the alignment tensor required to take into account the overall alignment strength. As the correlation visible in Figure 4.1 is not very high, σ_l^{RDC} can also be set to a fixed value of 0.21 for R_{struc} ranging from 1.4 to 2.4 Å without strongly

Table 4.1: Proteins used for evaluation of the correlation between resolution of a crystal structure and the fit of residual dipolar couplings to this structure

protein	crystal structure	resolution (Å)	RDCs	normalized
	(PDB code)		(PDB code)	rmsd [a]
maltose-binding protein	1DMB	1.8	1EZP	0.24
maltose-binding protein	1OMP	1.8	1EZP	0.19
barrier-to-autointegration factor	1CI4	1.9	2EZX	0.23
B1 IgG-binding domain	1IGD	1.1	1P7E	0.13
B1 IgG-binding domain	1PGA	2.1	1P7E	0.20
B1 IgG-binding domain	1PGB	1.9	1P7E	0.23
N-terminal domain of enzyme I	1ZYM	2.5	3EZA	0.26
histidine-containing phosphocarrier protein	1POH	2.0	3EZA	0.21
histidine-containing phosphocarrier protein	1OPD	1.5	3EZA	0.20
ubiquitin	1UBQ	1.8	1D3Z	0.18
ubiquitin	1UBI	1.8	1D3Z	0.18
ubiquitin	1F9J	2.7	1D3Z	0.32
ubiquitin	1AAR	2.3	1D3Z	0.24
bovine pancreatic trypsin inhibitor	5PTI	1.0	[b]	0.18
bovine pancreatic trypsin Inhibitor	1QLQ	1.4	[b]	0.26
cyanovirin-N	1L5B	2.0	2EZM	0.31
cyanovirin-N	3EZM	1.5	2EZM	0.18
lysozyme	1FLQ	1.8	1E8L	0.21
lysozyme	1UIG	1.9	1E8L	0.22
lysozyme	1UIH	1.8	1E8L	0.21
lysozyme	1H87	1.7	1E8L	0.22
lysozyme	1H6M	1.6	1E8L	0.27
lysozyme	1GWD	1.8	1E8L	0.24
lysozyme	193L	1.3	1E8L	0.19
lysozyme	1IEE	1.5	1E8L	0.25
lysozyme	194L	1.4	1E8L	0.20
lysozyme	1AKI	1.5	1E8L	0.22
lysozyme	1AT5	1.8	1E8L	0.21
lysozyme	1DPX	1.7	1E8L	0.21
lysozyme	1F0W	1.9	1E8L	0.25
lysozyme	1KXW	2.0	1E8L	0.23

[a] 'Normalized rmsd' is the root-mean-square-deviation between experimental and back-calculated RDCs divided by the experimental alignment strength D_a^{HN}.

[b] RDCs were kindly provided by Ben Ramirez and Ad Bax.

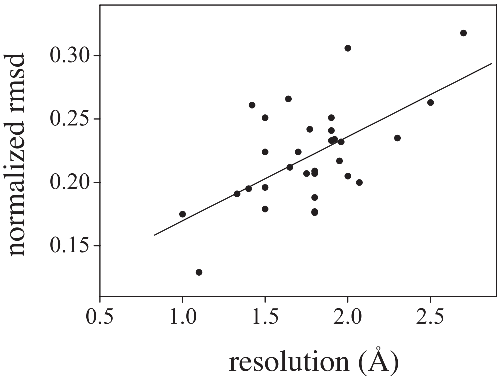

Figure 4.1: Correlation between resolution of a crystal structure and the fit of dipolar couplings to this structure. 'Normalized rmsd' is the root-mean-square-deviation between experimental and back-calculated RDCs divided by the experimental alignment strength D_a^{HN}. Back-calculation of RDCs was performed by SVD.

affecting the assignment result. RDCs are back-calculated from user-supplied PDB coordinates according to

$$RDC_{pq}^{cal} = \frac{-\mu_0 \gamma_p \gamma_q h}{8\pi^3 \left\langle r_{pq}^3 \right\rangle} \sum_{i,j} A_{ij} \cos \varphi_i^{pq} \cos \varphi_j^{pq} \qquad (4.3)$$

, where RDC_{pq}^{cal} is the dipolar coupling between a pair of spin-1/2 nuclei, p and q, separated by a distance r_{pq}, $\hat{\mathbf{A}}$ is a second-rank alignment tensor, γ_p and γ_q are the gyromagnetic ratios, h is Planck's constant, μ_0 is the magnetic permeability of vacuum, and π_i^{pq} is the angle between the p $-$ q internuclear vector and the ith molecular axis. As π_i^{pq} and r_{pq} can be derived from the 3D structure, the only unknown variable in Equation (4.3) is the alignment tensor $\hat{\mathbf{A}}$.

4.2.3 Alignment Tensor Determination

The magnitude and rhombicity of a molecular alignment tensor A can be obtained accurately without assignment from a histogram of experimental RDCs (Clore et al, 1998; Skrynnikov and Kay, 2000; Warren and Moore, 2001). In order to extract the orientation of the alignment tensor, four different methods are available in MARS: (1) shape and charge/shape-prediction of molecular alignment tensors [125, 124], (2) singular value decomposition [70] after an initial assignment step using only chemical shifts, (3) exhaustive back-calculation[126] and (4) a grid search that optimizes the fit of experimental chemical shifts and RDCs to values predicted from the 3D structure.

Shape- and charge/shape prediction is problematic for proteins with long, flexible loops or tails, but has the advantage that the only information necessary is the 3D structure [125, 124]. Exhaustive back-calculation is useful when the amino acid type of some resonances can be identified either by selective labeling or on the basis of the C^{α} and C^{β} chemical shift [126], as the actual size of the protein is not important, provided that experimental RDCs could be measured accurately. When sufficient chemical shift data are available (for example sequential connectivity information), it is straightforward to obtain the alignment tensor by a two-stage strategy which consists of an initial assignment run using only chemical shifts, followed by a best-fit of experimental RDCs to the 3D structure [70] based on this assignment.

Tests show that, even when the percentage of correct assignment is below 50%, the alignment tensor is very close to its correct orientation. As C^{α}/C^{β} chemical shifts depend very much on the type of secondary structure, exchange of assignments mainly takes place between residues located on the same type of regular secondary structure. If these secondary structure elements are close to collinear, such as two β-strands in a β-sheet, residues located in these strands can have similar RDCs and back-calculated alignment tensors are not severely affected by an interchanged assignment.

The most general method for extracting the orientation of the alignment tensor is a grid search in which the fit between experimental and predicted RDCs and chemical shifts is optimized. In this gird search 1116 uniformly distributed alignment tensor orientations are systematically sampled [31] and for each orientation the deviation $D(i,j)$ between experimental and back-calculated RDCs is determined (Equation (4.1)). All sampled orientations are ranked according to their corresponding $D(i,j)$ values and the lowest $D(i,j)$ value indicates the best estimate for the experimental alignment tensor. All assignment results reported here were obtained using this method.

After obtaining initial alignment tensor using any methods described above, a refinement step follows using SVD (See figure 4.2).

4.2.4 Assignment Schedule

The overall assignment schedule is slightly changed when RDCs are used in addition to chemical shifts. Although the methods described above allow determination of approximate alignment tensors, their accuracy is inferior to singular value decomposition based on a known assignment. Therefore, for RDC-enhanced assignment two complete MARS assignment runs are performed. In the first run, dipolar couplings are back-calculated from the 3D structure using the approximate alignment tensor. After this run a sufficient number of reliable assignments are generally available and based on these assignments MARS can perform a singular value decomposition. This results in an improved tensor that is used in a second assignment run to refine assignment (Figure 4.2). More assignment runs are generally not required. Due to this two-step procedure the final assignment score obtained by MARS is almost independent from the method that was chosen to get a first estimate of the alignment tensor.

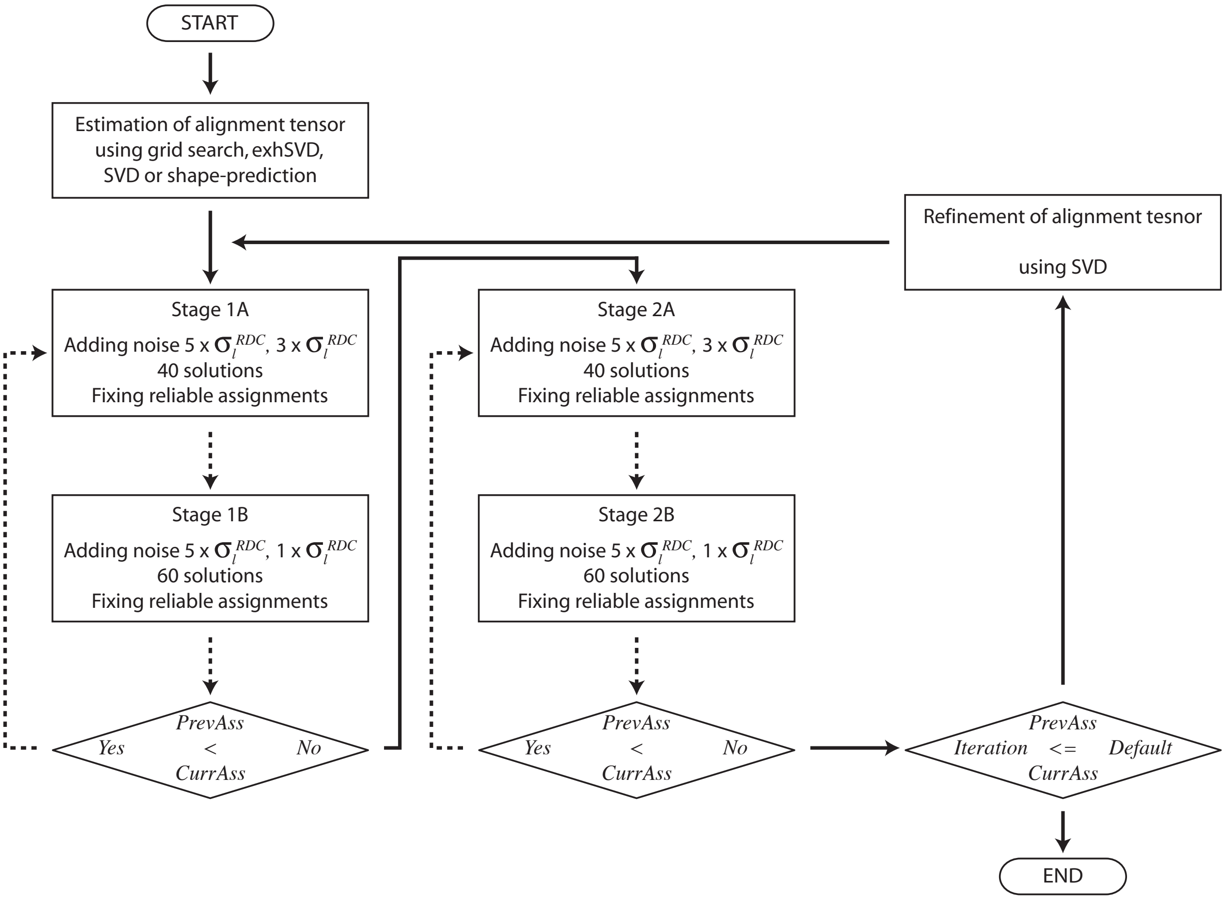
START
Estimation of alignment tensor using grid search, exhSVD, SVD or shape-prediction
Refinement of alignment tesnor using SVD
Stage 1A
Adding noise 5 x σ_l^{RDC}, 3 x σ_l^{RDC}
40 solutions
Fixing reliable assignments
Stage 2A
Adding noise 5 x σ_l^{RDC}, 3 x σ_l^{RDC}
40 solutions
Fixing reliable assignments
Stage 1B
Adding noise 5 x σ_l^{RDC}, 1 x σ_l^{RDC}
60 solutions
Fixing reliable assignments
Stage 2B
Adding noise 5 x σ_l^{RDC}, 1 x σ_l^{RDC}
60 solutions
Fixing reliable assignments
Yes PrevAss < CurrAss No
Yes PrevAss < CurrAss No
Iteration PrevAss <= CurrAss Default
END

Figure 4.2: Empirically optimized scheme for avoiding errors due to inaccuracies in calculated RDCs and chemical shifts when mapping pseudo-residue segments to the protein sequence. Opposite to the original scheme ([56]), two full assignment runs are performed and in the second run a refined alignment tensor, which has been obtained by SVD, is used. $5 * \sigma_l^{RDC}$ is the width of the Gaussian distribution function from which RDC noise is drawn. By default two iterations (2 <= Default) are performed. See text for a definition of σ_l^{RDC} .

4.2.5 Overcoming Structural and Dynamic Deviations from PDB Coordinates

RDCs strongly depend on the exact orientation of their corresponding internuclear vectors (Equation (4.3)) and slight errors in the structure can give rise to significant deviations in back-calculated dipolar couplings. Back-calculated RDCs, however, are used for mapping of pseudo-residue segments to the protein sequence and incorrect values can lead to wrong assignments. This problem is partially addressed by reducing the weight of RDCs compared to chemical shifts by a factor of 3.3 (Equation (4.1)).

To further improve the reliability of RDC-enhanced assignment, a similar approach as for chemical shifts is used ([56]): several assignment phases are performed where back-calculated RDCs are disturbed by addition of noise and only consistent assignments are retained. The addition of noise to RDCs and chemical shifts is done simultaneously and results in an empirically optimized assignment schedule outlined in Figure 4.2. Opposite to chemical shifts, however, the amount of noise added to back-calculated RDCs is kept fixed at five times σ_l^{RDC}. Such a large amount of variation in back-calculated RDCs is necessary, in order to avoid wrong assignments.

Often parts of proteins, such as flexible termini or loops, are unstructured and are not available in crystal structures or are prone to deviate from their conformation in solution. In order to identify potentially flexible parts of proteins, we estimated NMR S^2 order parameters of N-HN vectors of the protein backbone from the 3D structure [120]. Removal of back-calculated RDCs for residues with estimated S^2 order parameters smaller than 0.75 did, however, not improve MARS assignment results. Therefore, RDCs are back-calculated for all residues that are visible in a crystal structure and are used for enhanced mapping to the protein sequence.

4.2.6 Testing

RDC-enhanced assignment was applied to three proteins for which experimental chemical shifts and dipolar couplings have been reported and a high-resolution crystal structure is available: ubiquitin (76 aa; PDB codes: 1UBQ [1.8 Å] and 1AAR [2.3 Å]; RDCs: PDB code 1D3ZMR; chemical shifts from TALOS) [24, 25, 108], the N-terminal domain of enzyme I of the phosphoenolpyruvate (EIN) (259 aa; PDB code: 1ZYM [2.5 Å]; RDCs: PDB code 3EZAMR; chemical shifts: BMRB code 4106) [36, 37, 67] and two-domain maltose-binding protein (MBP) (370 aa; PDB code: 1DMB [1.8 Å]; RDCs: kindly provided by Lewis Kay; chemical shifts: BMRB code 4354) [34, 78, 96, 119]. Protons were added to crystal structures using MOLMOL [62]. In order to evaluate, how much information is required for successful assignment, we analyze different test cases, such as assignment without sequential connectivity information using only dipolar coupling/chemical shift matching, assignment with only C^α sequential connectivity information and assignment using C^α/C^β chemical shifts. In addition, the effect of including only one, two or three types of RDCs is tested.

4.3 Results and Discussion

4.3.1 RDC-enhanced Assignment without Sequential Connectivity Information

Table 4.2 shows the results of RDC-enhanced assignment for 76-residue protein ubiquitin. Initially, it was tested how inclusion of RDCs can enhance assignment when no connectivity information at all is available. A situation is assumed where only inter-residual C$^{\alpha}$, C$^{\beta}$ and C' chemical shifts could be measured, providing information about the amino-acid type of the preceeding residue. When no sequential connectivity information is available, MARS matches single pseudo-residues (comprising HN(i), N(i), C'(i-1), C$^{\alpha}$(i-1) and C$^{\beta}$(i-1) chemical shifts) to the primary sequence.

Alternatively, one could try to map all possible three-residue fragments (*i.e.* for a total of 72 pseudo-residues there would be about 360000 possible three-residue fragments that could be matched to each three-residue protein fragment). Tests, however, show that this significantly reduces the assignment quality. As calculation of chemical shifts from the protein sequence (or from the 3D structure) gives only approximate values, the total percentage of correct assignment in the absence of RDCs was only 36.1%. Moreover, only 19.5% (out of the total of 72 assignable residues) were labeled as reliable by MARS and about 40% of these were wrong. This highlights that mapping of single pseudo-residues (comprising only chemical shifts) to the protein sequence is not sufficient and that additional information for identification of reliable assignments is required. Comparison of RDCs back-calculated from a known 3D structure with experimental values provides such information.

Including only ^{1}D$_{NH}$ couplings into the assignment process increased the overall assignment score to 47.2%, the reliable assignment to 16.7% and out of these only one assignment was wrong. The situation was further improved when two or three types of RDCs were used. Without trying to distinguish between correct and incorrect assignments, *i.e.* without

Table 4.2: RDC-enhanced assignment of ubiquitin for varying amount of data

RDCs [a]	chemical shifts for linking [b]	chemical shifts for matching [c]	assignment score (%) [d]					
			1UBQ			1AAR		
			Total [e] correct	Reliable	Wrong reliable [f]	Total [e] correct	Reliable	Wrong reliable [f]
without sequential connectivity information								
--	--	C'_{i-1}, C^{α}_{i-1}, C^{β}_{i-1}	36.1	19.5	5.6	36.1	19.5	5.6
$^{1}D_{NH}$	--	C'_{i-1}, C^{α}_{i-1}, C^{β}_{i-1}	47.2	16.7	1.4	38.9	9.7	1.4
$^{1}D_{NH}$, $^{1}D_{CaC'}$	--	C'_{i-1}, C^{α}_{i-1}, C^{β}_{i-1}	76.4	44.4	0.0	68.1	33.4	2.8
$^{1}D_{NH}$, $^{1}D_{CaC'}$, $^{1}D_{NC'}$	--	C'_{i-1}, C^{α}_{i-1}, C^{β}_{i-1}	91.7	55.6	0.0	83.3	50.0	0.0
with sequential connectivity information								
--	C^{α}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}	80.6	25.0	0.0	80.6	25.0	0.0
$^{1}D_{NH}$	C^{α}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}	93.1	51.4	0.0	97.2	37.5	0.0
$^{1}D_{NH}$, $^{1}D_{CaC'}$	C^{α}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}	100.0	90.3	0.0	100.0	73.6	0.0
$^{1}D_{NH}$, $^{1}D_{CaC'}$, $^{1}D_{NC'}$	C^{α}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}	100.0	100.0	0.0	100.0	100.0	0.0

[a] RDCs observed in nearly neutral bicelles were used.

[b] Chemical shifts used for establishing sequential connectivity. The connectivity threshold was 0.2 ppm.

[c] Chemical shifts used for mapping pseudoresidue segments to the protein sequence. In addition to the mentioned values, HN and N chemical shifts were used.

[d] Relative to the number of assignable residues, 72 in case of ubiquitin.

[e] # of correct assignments in Ass^{global} ; Ass^{global} was obtained from a MARS run without addition of noise.

[f] Assignments that were identified as reliable but are incorrect, i.e. the number of errors.

applying the MARS criteria for reliability, 66 residues (91.7%) of ubiquitin were assigned correctly. This is in agreement with recent results by Hus et al. [50]. Opposite to their approach, however, MARS allows a clear distinction between reliable assignments and those that are prone to errors. Application of the MARS reliability criteria identifies 55.6% of residues (out of the total of 72 assignable residues) as reliably assigned, with not a single error present.

The results discussed so far were obtained with a 1.8 Å crystal structure of ubiquitin (PDB code: 1UBQ). Such high-resolution structures might not always be available. At the same time, structural noise is a major factor influencing the accuracy of back-calculated alignment tensors and RDCs, whereas the experimental accuracy of RDC data measured with current methods is usually sufficient [123]. In order to test the robustness of RDC-enhanced assignment against structural deviations from the PDB coordinates, assignment of ubiquitin was also performed using a 2.3 Å crystal structure (PDB code: 1AAR). When only $^1D_{NH}$ couplings were used, the number of reliable assignments was reduced compared to assignment based solely on chemical shifts (Table 4.2). This is due to the fact that reliability is now tested using both chemical shifts and RDCs, thereby removing some (previously reliable and correct) assignments that do not have very characteristic dipolar couplings.

More important, however, is that the error rate was reduced to a single wrong assignment when $^1D_{NH}$ RDCs were introduced. Using two or three types of RDCs, assignment of ubiquitin was significantly enhanced, similar to the results obtained for the 1.8 Å structure. Due to the lower quality of the 1AAR structure, however, the improvement achieved by inclusion of RDCs was not as strong. In particular, a lower number of assignments were identified as reliable, whereas the total number of correct assignments was only slightly affected. Nevertheless, 100% of ubiquitin could be assigned reliably, when C^{α} connectivity information was combined with RDC-matching of $^1D_{NH}$, $^1D_{CaC'}$ and $^1D_{NC'}$ couplings (see below).

4.3.2 RDC-enhanced Assignment with Sequential Connectivity Information

For some applications, such as titration studies, reliable assignment scores of 50% might be sufficient or some wrong assignments are not problematic. Complete and error-free assignment, however, will often still be the major aim. In addition, assignment of small proteins such as ubiquitin is straightforward using C^{α}/C^{β} connectivity information obtained from triple-resonance experiments, even without usage of RDCs. For bigger proteins, on the other hand, mapping of pseudo-residues to the protein sequence using only chemical shifts is usually not sufficient to reliably assign 100% of the protein. Especially, when a substantial amount of data is missing due to chemical exchange or incomplete back-exchange of amide protons in deuterated proteins, the number of residues, which can be assigned reliably, significantly decreases [56]. Therefore, the area where RDC-enhanced assignment has its largest potential is for big, deuterated proteins in combination with standard sequential connectivity information.

Combination of a limited amount of connectivity information with RDC-matching was first tested on ubiquitin (Table 4.2). Using only C^{α} connectivity information with a threshold of 0.2 ppm for establishing sequential connectivity together with ${}^{1}D_{NH}$, ${}^{1}D_{CaC'}$ and ${}^{1}D_{NC'}$ couplings, 100% of residues were assigned reliably by MARS without any assignment error (for both the 1UBQ and 1AAR structure). On the other hand, without RDCs, *i.e.* using just chemical shifts for mapping pseudo-residue segments to the protein sequence, only 25% of residues could be assigned reliably. This indicates the great potential of combining RDCs back-calculated from a known structure with sequential connectivity information.

Table 4.3 shows results obtained from RDC-enhanced assignment for 370-residue maltose-binding protein (MBP). Using the complete set of C^{α}/C^{β} chemical shifts deposited in the BMRB [29] but not using any RDC-matching, 87.2% of assignable residues of MBP were assigned reliably. This number was increased to about 94% when at least one RDC type was included. No errors were introduced into assignment by RDC-matching. As the assignment score was already very high, inclusion of more than one RDC type did not further improve assignment significantly.

More pronounced was the effect when a substantial amount of data was missing. When 20% of MBP's pseudo-residues were removed randomly and no RDC-matching was employed, the reliable assignment was reduced to 44.2% and two assignment errors were present [56]. Enhancing the mapping process by comparison of ${}^{1}D_{NH}$, ${}^{1}D_{CaC'}$ and ${}^{1}D_{NC'}$ couplings back-calculated from MBP's 1.8 Å structure with experimental values, increased the reliable assignment to 62.6% (total correct assignment of 94.3%). In addition, no assignment errors were present any more.

Table 4.3: RDC-enhanced assignment of 370-residue maltose-binding protein for varying amount of data

RDCs [a]	chemical shifts for linking [b]	chemical shifts for mapping [c]	missing chemical shifts (%) [d]	assignment score (%) [e]		
				Total correct [f]	Reliable	Wrong reliable [g]
--	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	4	95.8	87.2	0.0
${}^{1}D_{NH}$	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	4	98.5	94.6	0.0
${}^{1}D_{NH}$, ${}^{1}D_{CaC'}$	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	4	98.5	93.4	0.0
${}^{1}D_{NH}$, ${}^{1}D_{CaC'}$, ${}^{1}D_{NC'}$	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	4	99.1	94.6	0.0
--	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	20	82.7	44.2	0.7
${}^{1}D_{NH}$	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	20	88.8	51.4	0.0
${}^{1}D_{NH}$, ${}^{1}D_{CaC'}$	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	20	95.0	58.0	0.4
${}^{1}D_{NH}$, ${}^{1}D_{CaC'}$, ${}^{1}D_{NC'}$	C^{α}, C^{β}	C'_{i-1}, C^{α}_{i-1}, C^{α}_{i}, C^{β}_{i-1}, C^{β}_{i}	20	94.3	62.6	0.0

[a] RDCs were measured for MBP dissolved in Pf1 bacteriophage.

[b] Chemical shifts used for establishing sequential connectivity. A common connectivity threshold of 0.5 ppm was used for C^{α} and C^{β}.

[c] Chemical shifts used for mapping pseudoresidue segments to the protein sequence. In addition, to the mentioned values HN and N chemical shifts were also used.

[d] Percentage of non-proline residues for which HN and N chemical shifts were not present.

[e] Relative to the number of assignable residues, i.e. those residues with HN and N chemical shifts.

[f] # of correct assignments in Ass^{global} ; Ass^{global} was obtained from a MARS run without addition of noise.

[g] Assignments that were identified as reliable but are incorrect, i.e. the number of errors.

4.3.3 Robustness against Missing Data

The robustness of RDC-enhanced MARS assignment was further tested by continuously increasing the randomly deleted fraction of observed pseudo-residues from 5 to 30% for MBP and the N-terminal domain of enzyme I (EIN). Similar to the situation when RDCs were not used, the assignment decreased with decreasing number of pseudo-residues and the reliable assignment was most strongly affected (Figure 4.3B). Whereas, however, without RDCs the percentage of reliable assignments dropped to 31% when 30% of MBP's pseudo-residues were randomly deleted [56], it remained at 49% upon inclusion of $^1\mathrm{D}_{NH}$, $^1\mathrm{D}_{CaC'}$ and $^1\mathrm{D}_{NC'}$ couplings. In addition, the total number of correct assignments was increased from 76% to 86%.

For MBP a very extensive set of RDCs was measured by optimized triple-resonance experiments [119]. For EIN, on the other hand, only $^1\mathrm{D}_{NH}$ RDCs for 60% of residues were available from two-dimensional HSQC spectra [37]. In addition, with a resolution of 2.5 Å and 10 residues not present in the PDB coordinates, the crystal structure available for EIN (PDB code: 1ZYM) is of much lower quality than that of MBP. In this case, RDC-enhanced and RDC-free assignment were virtually identical (Figure 4.3A). A slight improvement upon inclusion of RDCs, however, is obtained with respect to the error-rate. Whereas for RDC-free assignment two, three and three residues were assigned wrongly at 15, 20 and 30% deleted pseudo-residues, respectively, this was reduced to zero, zero and two residues for RDC-enhanced assignment. Such a small effect is actually not unexpected as the major use of RDCs is improved matching of PR-segments to the primary sequence.

When C^α and C^β chemical shift information is close to complete, as it is the case for EIN, segment placement is already quite robust and incorporation of just $^1\mathrm{D}_{NH}$ couplings for 60% of pseudo-residues does not have a major impact. Very often, however, not entire pseudo-residues are missing, but certain chemical shifts are not observable. This situation was simulated by randomly removing chemical shifts within pseudo-residues of EIN. When C^α and C^β chemical shifts are removed from pseudo-residues this strongly affects the ability

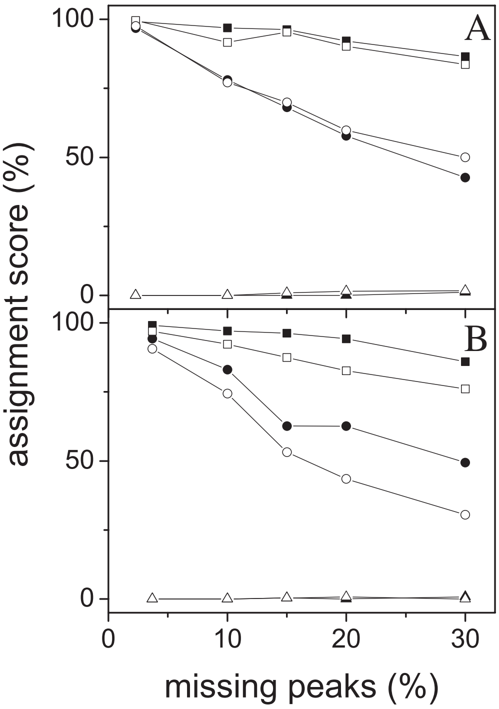

Figure 4.3: Dependence of RDC-enhanced assignment on the percentage of missing pseudo-residues. Pseudoresidues were randomly deleted. ■ indicate the percentage of all assignments that were correct (not tested for reliability). ● show the percentage of residues that could be assigned reliably (relative to the total number of assignable residues) and ▲ indicate assignments that were identified as reliable but are wrong, i.e., the error rate of MARS. Only $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ chemical shifts with a common threshold of 0.5 ppm for establishing sequential connectivity were used. Open symbols indicate the results without RDCs [56]. (A) Results for the 259-residue N-terminal domain of enzyme I using RDC-matching of $^1D_{NH}$ couplings. (B) Results for the 370-residue maltose-binding protein using RDC-matching of $^1D_{NH}$, $^1D_{CaC}$ and $^1D_{NC}$ couplings.

to correctly place PR-segments onto the primary sequence. In such situations even a small number of ${}^{1}D_{NH}$ RDCs can be useful as demonstrated in Figure 4.4. Although, the number

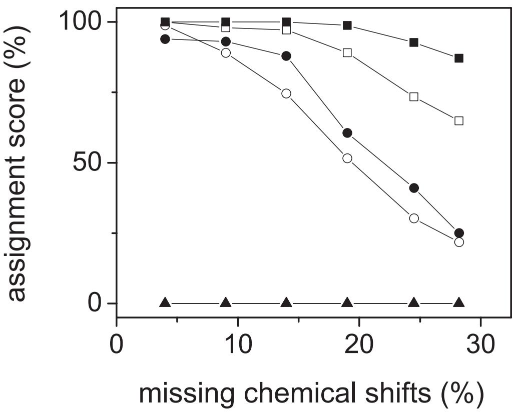

Figure 4.4: Dependence of RDC-enhanced assignment on the percentage of missing chemical shifts within pseudo-residues for the 259-residue N-terminal domain of enzyme I. Chemical shifts were deleted randomly. ■, ● and ▲ indicate correct, reliable and wrong reliable assignments, respectively. ${}^{13}C^{\alpha}$ and ${}^{13}C^{\beta}$ chemical shifts with thresholds of 0.2 and 0.4 ppm for establishing sequential connectivity were used. Open symbols indicate the results without RDCs [56]. There are zero errors for both RDC-enhanced and RDC-free assignment.

of reliable assignments was only increased by 6% on average, the total number of correct assignments was raised by 22% when 28% of C^{α}/C^{β} chemical shifts were missing. This means that with the help of ${}^{1}D_{NH}$ RDCs the correct assignment was proposed for 55 additional residues of EIN, providing a significantly improved starting point for manual refinement of the assignment (using for example the analysis software SPARKY). Therefore, even for sparse data comparison of RDCs back-calculated from a known 3D structure with experimental values is useful for assignment.

4.4 Concluding Remarks

We have introduced a reliable method for enhancing backbone resonance assignment of proteins with known structure using residual dipolar couplings. This method has been implemented into the automatic assignment program MARS. It is equally applicable to small or big proteins, when only $^1\mathrm{D}_{NH}$ couplings could be measured for a limited number of residues or when a complete set of dipolar couplings for five different inter-nuclear vectors is available.

RDC-enhanced assignment will be especially useful for large proteins where chemical shift data are often missing for a substantial portion of residues and chemical shift degeneracy is too high to allow unambiguous assignment. Similarly, if only a few reliable assignments could be obtained based on chemical shifts and sequential connectivity, RDC matching allows evaluation of remaining assignment possibilities. Safe assignments or connectivities (as established, for example, from manual inspection of assignment strips on the screen) can thereby be fixed.

Structure-enhanced assignment becomes increasingly important due to the rapid increase in the number of high-resolution 3D structures that are determined as part of the worldwide structural genomics effort. Residual dipolar couplings are in this respect particularly interesting, as they can be measured efficiently from two-dimensional ^{1}H-^{15}N HSQC or three-dimensional triple-resonance spectra.

Moreover, triple-resonance experiments can be used simultaneously for RDC measurement and to establish sequential connectivity [102, 127], thereby saving spectrometer time and money. At the same time, dipolar couplings will often be measured, in order to validate a crystal structure prior to its usage, for example, in binding studies [32, 54]. In these cases, inclusion of RDCs into the assignment process and therefore improved assignment will not require additional NMR samples or extra measurement time.

Chapter 5

Simultaneous Assignment and Structure Determination of Protein Backbones by Using NMR Dipolar Couplings

5.1 Introduction

The wealth of genomic data that has recently become available with completion of the sequencing of both the human and a variety of other genomes [1] has created a need for rapid and efficient determination of three-dimensional (3D) structures of the corresponding proteins. To date, most effort in this so-called structural genomics has focused on X-ray crystallography, but NMR spectroscopy also shows considerable potential [2, 3].

Approaches for automatic structure calculation I

Conventionally, NMR structure determination consists of two stages; a resonance assignment stage and structure calculation stage. A resonance-assignment stage usually relies on analysis of an extensive set of triple resonance J-connectivity data, and then followed by a structure calculation stage using distance-, angle-constraint information that rely on interpretation of NOE spectra and/or measurement of scalar and dipolar couplings. Each stage requires considerable amount of time, posing a problem for high-throughput desired by structural genomics.

For the first stage, sequence specific resonance assignment, there have been several efforts to automate and accelerate backbone resonance assignment, especially for ^{13}C/^{15}N-labeled proteins using triple-resonance spectra. The main problems with fully automatic approaches are that for a standard structure calculation based on NOE and scalar and dipolar couplings, the assignment has to be error-free and nearly complete. This, however, is rarely achievable by automatic approaches when dealing with real NMR spectra that show spectral overlap and missing resonances. Especially disastrous are errors in the assignment as they will generally result in incorrect protein structures. The automatic assignment approaches were described in detail in chapter 3.

For the second stage, structure calculation, several automated approaches for NOESY interpretation and structure calculation have been developed, including NOAH [80, 79], ARIA [84, 82], CANDID [47] and AutoStructure [49, 48]. The NOAH, ARIA, and CANDID programs utilize an iterative data interpretation approach.

NOAH creates an unambiguous constraint for each ambiguous proton-proton interaction, reassigning constraints that are internally inconsistent (self-correcting) in the course of the structure calculation. ARIA uses an ambiguous constraint strategy, involving multiple ambiguous distance constraints for each ambiguous NOESY peak. The program NOAH has been combined with the structure generation programs DYANA [44], XPLOR-NIH [64] and DIAMOD [118]. The program ARIA has been combined with the structure generation program CNS [18]. Initial structures are first built using ambiguous constraint strategies and then iteratively refined.

The program CANDID, combined with DYANA, also uses ambiguous constraint strategies but, in addition, employs network anchoring and constraint-combination methods, minimizing deleterious effects when this correctness assumption is not satisfied.

The program AutoStructure is aimed at iteratively identifying self-consistent NOE contact patterns without using any 3D structure model, and delineating secondary structures; including alignments between β-strands based upon a combined pattern analysis of secondary structure-specific NOE contacts, chemical shifts, scalar coupling constants, and slow amide proton exchange data. It automatically generates conformational constraints (*e.g.* distance, dihedral angle and hydrogen bond constraints) and submits parallel structure calculations. The resulting structure is then refined automatically by iterative cycles of self-consistent assignment of NOESY cross peaks and regeneration of the protein structure with the program DYANA.

A highly error-tolerant approach for automated structure analysis has recently been implemented within the XPLOR-NIH package [64]. The approach takes in a large list of NOE restraints created in a simplistic fashion from direct all-to-all matching of NOE peaks to resonance assignments and uses a probabilistic method to turn on and off NOE restraints as the simulated annealing progresses. The approach is very fault tolerant and robust but also computationally very intensive.

Recently, there was a new approach using only RDCs for structure calculation. It was shown by two independent methods that is possible to determine structures of protein backbones using only RDC restraints [102, 88]. Furthermore, one of the two, the molecular fragment approach [88], was made more robust by combination with the *ab initio* structure prediction program Rosetta [98]. The program Rosetta selects peptide fragments from proteins of known structure based on sequence similarity and consistency with chemical shift and RDC data, and then builds models from fragments by minimizing an energy function that favors hydrophobic burial, strand pairing, and satisfaction of RDC constraints. The method allows structures to be generated for proteins without collecting and assigning large constraint sets.

Approaches for automatic structure calculation II

In order to automate the whole process from NMR spectra to three-dimensional protein structures, mainly two approaches have been followed so far mainly. One is to calculate structures without resonance assignment [86, 63, 7, 6, 39, 40], and the other is to assign the protein backbone resonances and to calculate the structures simultaneously [102].

It is universally assumed that a protein structure determination by NMR requires the sequence-specific resonance assignments. Several attempts have been made to devise a strategy for NMR protein structure determination that circumvents the tedious chemical shift assignment step. The underlying idea of assignment-free NMR structure calculation methods is to exploit the fact that NOESY spectra provide distance information even in the absence of any chemical shift assignments. This proton-proton distance information can be exploited to calculate a spatial proton distribution. Since there is no association with the covalent structure at this point, the protons of the protein are treated as a gas of unconnected particles. Provided that the emerging proton distribution is sufficiently clear, a model can then be built into the proton density in a manner analogous to X-ray crystallography in which the structural model is constructed into the electron density.

The most recent approach to NMR structure determination without chemical shift assignment is the CLOUDS protocol [39, 40]. For the first time, the feasibility of the method has been demonstrated using experimental data rather than simulated data sets. The CLOUDS method relies on precise and abundant inter-proton distance constraints calculated via a relaxation matrix analysis of set of experimental NOESY cross peaks. It showed that assignment-free NMR structure calculation can successfully generate 3D protein structures from experimental data. Nevertheless, in the course of a *de novo* structure determination it may not be straightforward to produce a NOESY peak list of the completeness and quality used for these test calculations. In particular, it was assumed that the NOEs can be identified unambiguously, *i.e.* that it is known with certainty whether any two NOESY peaks involve the same proton or not.

Others showed that it is possible to perform resonance assignment and structure calculation for a 54 residue protein simultaneously, *i.e.* lifting the sequential nature of the two processes: RDCs are used to define local structural features ahead of assignment and the same couplings are used in combination with chemical shifts to connect these fragments in a sequence-specific way [102].

Here, we show new application for simultaneous automatic assignment and structure calculation, ITAS (ITerative Assignment and Structure), for rapid determination of protein folds: partial backbone resonance assignments obtained automatically by the program MARS [56, 57] are used to create low-resolution models with the program RosettaNMR [92], these low-resolution models are used to improve backbone resonance assignment and the improved assignment is again used for structure calculation. Starting from unassigned backbone chemical shifts and residual dipolar couplings nearly complete resonance assignment and medium-resolution structures of protein backbones are obtained within six steps of iteration.

In addition, we show that neither a small number of missing assignments nor some isolated wrong assignments significantly degrade the quality of these medium-resolution structures. The new strategy is demonstrated for proteins varying in length from 54 to 153 residues and covering various topologies, in order to automate the whole process from NMR spectra to three-dimensional protein structures.

5.2 Methods

We combined two programs MARS and RosettaNMR for iterative assignment and structure calculation. MARS carries out automatic sequential-specific-backbone-resonance assignment, and RosettaNMR takes care of structure calculation. For data analysis, data formatting and program execution, TCL/TK and AWK program are used. We named the '**IT**erative **A**ssignment and **S**tructure calculation program' as 'ITAS'. ITAS uses several additional programs; (i) PALES [125] and PSIPred [72] for MARS; (ii) PSIPred, PSIBLAST [4] and modified TALOS for RosettaNMR. MOLMOL [61] is used for adding hydrogen atoms because Rosetta-structures don't include the hydrogen atoms but PALES requires the hydrogen atoms for several types of RDCs (*e.g.* $\mathrm{D}_{\mathrm{NH}^N}$, $\mathrm{D}_{\mathrm{C}^\alpha,\mathrm{H}^\alpha}$).

5.2.1 Iteration Procedure-ITAS

ITAS automatically controls all necessary steps during the iterative assignment and structure calculation process (Figure 5.1). It starts from a list of unassigned chemical shifts and RDCs that are grouped according to their common $^1\mathrm{H}^N$, $^{15}\mathrm{N}$ chemical shift into so-called pseudo-residues. In the first step, assignment of backbone resonances is automatically performed by the program MARS using only chemical shift information (Figure 5.1 and subsection 5.2.2). No information about the tertiary structure is used yet. This will, in general, result in incomplete assignment. The degree of reliable assignment will depend on the size and complexity of the protein and on the number, type and quality of NMR spectra available (Table 5.1).

Experimental dipolar couplings and chemical shifts for pseudo-residues that were assigned by MARS are used in the second step for structure calculation using RosettaNMR. RosettaNMR combines experimental RDCs and chemical shifts with empirical statistics used by *ab initio* structure prediction methods. The three-dimensional structural models that are generated by RosettaNMR are then ranked according to favorable non-local interactions

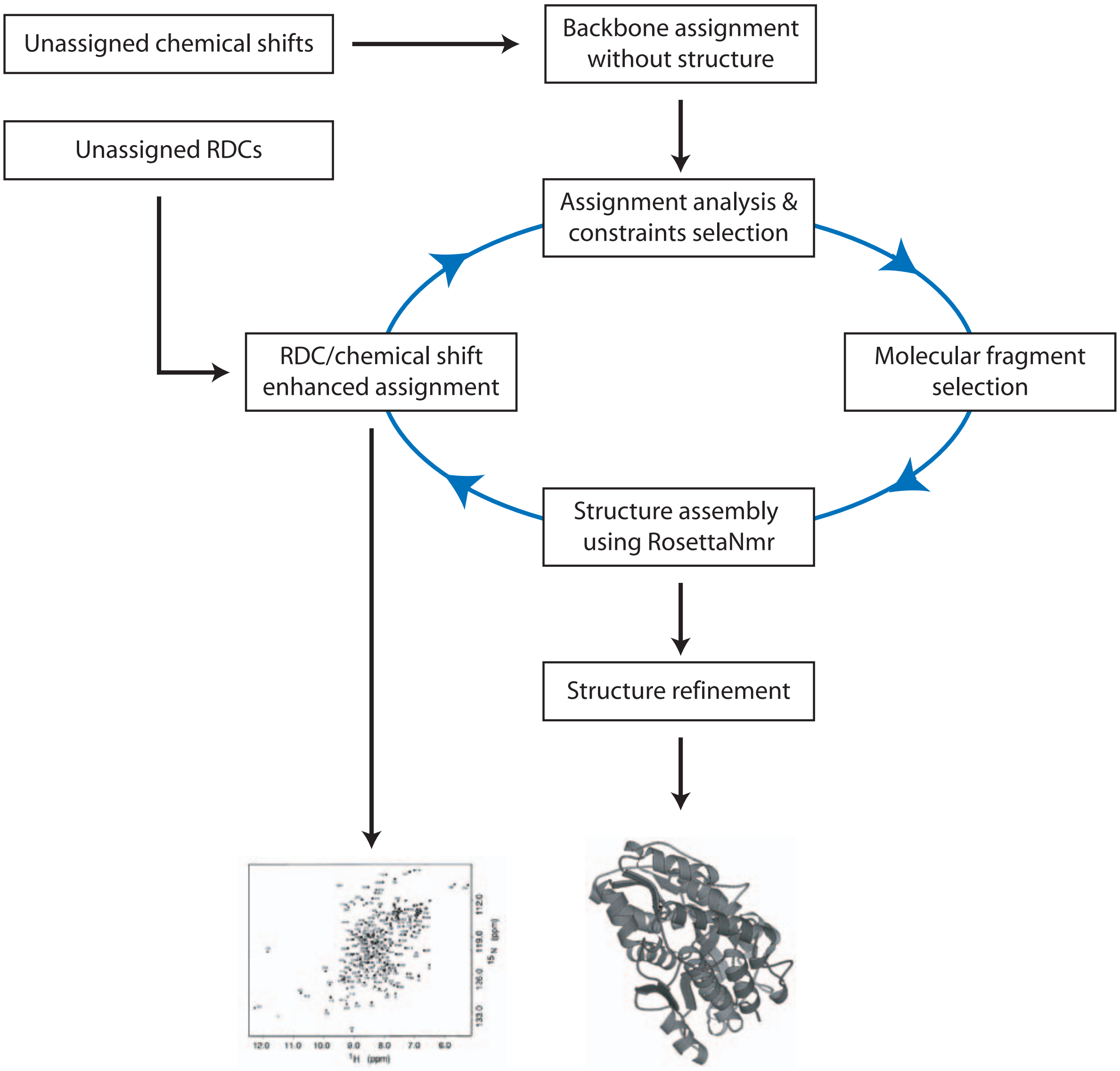
Unassigned chemical shifts
Backbone assignment without structure
Unassigned RDCs
Assignment analysis & constraints selection
RDC/chemical shift enhanced assignment
Molecular fragment selection
Structure assembly using RosettaNmr
Structure refinement
12.0 11.0 10.0 9.0 8.0 7.0 6.0
^{1}H (ppm)
112.0 119.0 126.0 133.0
^{15}N (ppm)

Figure 5.1: Overview of the ITAS fold determination procedure.

and agreement with experimental RDCs. These models are used in the next step to enhance backbone resonance assignment.

MARS is run again and this time the assignment no longer relies only on chemical shifts, but can be improved by comparison of experimental dipolar couplings with back-calculated dipolar couplings from the RosettaNMR-models. As more pseudo-residues can be assigned reliably, a larger number of chemical shifts and RDCs become available for a new round of structure calculation with RosettaNMR. This iterative procedure is continued until the number of backbone resonance assignments obtained from MARS is saturated. The structural models that are obtained with the highest number of experimental data, *i.e.* the highest number of assigned residues, during the iteration are finally subjected to a short energy minimization in which dihedral angles of single residues are perturbed. The following subsections describe each step of ITAS in more detail.

5.2.2 Automatic Resonance Assignment Using MARS

Automatic backbone resonance assignment is carried out by the program MARS. For the small proteins rubredoxin, the third Igg-binding domain of protein G, ubiquitin, the Z domain of Staphylococcal protein A, the RecA-binding protein DinI and the C-terminal KH domain of heterogeneous nuclear ribonucleoprotein only $^{13}C^{\alpha}$ connectivity information was employed, in order to demonstrate the efficiency of ITAS. Sequential connectivity was established based on a cutoff of 0.3 ppm. With 3D HNCA experiments this resolution is easily obtainable even for weak NMR resonances. For proteins above 10 kDa $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ connectivity information were used. In these cases connectivity cutoffs were set to 0.5 ppm for both $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ chemical shifts, in order to take into account the lower resolution usually obtainable from HNCACB and CBCA(CO)NH experiments.

MARS gives several outputs; 'assignment_AA.out', a file listing pseudo-residues assigned reliably to residues (2) 'assignment_AAs.out', an extended assignment including alternative assignment possibilities that show up with a 10 % probability, (3) ‘assignment_PR.out’, the most likely assignment for each pseudo-residue, (4) ‘connectivity.out’, ‘mars.log’, ‘sparky_all.out’, ‘sparky_CA.out’, ‘sparky_CA-1.out’, ‘sparky_CB.out’, . . . (See chapter 3).

In the beginning, MARS runs once without using a structure (See the step of ‘Backbone assignment without structure’ in Figure 5.1), then RosettaNMR runs with assigned RDCs; in this case, only one ‘assignment_AA.out’ result is used for structure calculation without further analysis of assignments. If there are no assignments in the ‘assignment_AA.out’, then ‘assignment_PR.out’ is used for structure calculation. After the first cycle, the structures generated by RosettaNMR are available for the assignment with RDC-enhanced assignment (See chapter 4). During each iteration without the first cycle, MARS runs 20 times with the 20 lowest energy structures. Each structure gives different assignment result because the back-calculated RDCs from the structures might be different due to the structure deviations. The subsection 5.2.3 is describing about the analysis of assignments.

5.2.3 Automatic Analysis of Assignments

Assignments are analyzed after the step of ‘RDC/chemical shift enhanced assignment’ (in Figure 5.1). The idea of multiple assignments with the 20 lowest energy structures is that in the earlier cycles, the structure qualities are not good enough to maximize the assignment percentage; on other hand, the possibility of introducing incorrect assignments due to the low quality structures has to be considered. To increase assignment percentage but discard unreliable assignments, we apply the following conditions.

1. The assignments in ‘assignment_AA.out’s of the 20 lowest energy structures are compared and only unambiguous assignments are taken, but ambiguous assignments, which are assigned more than once, are removed.

2. Compared to the number of assignments obtained previous iteration, the number of the assignment doesn't increase, generally at later cycle, then the assignments in 20 'assignment_PR.out's are taken for comparing. In this case, the most frequently occurred assignments are taken; and, same to the condition 1, ambiguous assignments are removed.

5.2.4 Structure Calculation by RosettaNMR

ITAS uses the program RosettaNMR [17, 92] to calculate structures. There are two steps for structure calculation. At the first step, generating fragment libraries, RosettaNMR generates fragment libraries that are consistent with chemical shifts and RDCs; at the second step, generating structures, the fragment libraries are used with the RosettaNMR *de novo* fragment insertion method that uses the constraint data in its scoring function to generate structures.

The fragment libraries consist of 200 nine− and three− residue fragments for every overlapping window in the protein sequence. The fragments originate selected from a non-redundant database of protein crystal structures of resolution better than 2.0 Å . The selection is performed on the basis of multiple sequence alignment and the fit between measured RDCs and chemical shifts and those back-calculated from the structure of each data base fragment [5, 6]. Then a compact structure is assembled from these fragments using the Rosetta Monte Carlo simulated annealing protocol.

All backbone atoms in the protein including H^N and H^α are explicitly included while each amino acid side chain is represented only by a single centroid. Simulations start with the protein chain in an extended conformation and then contiguous sets of backbone torsion angles are replaced with those of fragments chosen randomly from the library. 1000 low-resolution structural models are generated and ranked according to agreement with experimental RDCs and energetically favorable non-local interactions which include hydrophobic burial, pairing of β-strands, and overall compactness.

5.2.5 Structure refinement by RosettaNMR

The models generated in the final step are refined using RosettaNMR (vesrion 1.2) simulated annealing protocol. A standard protocol is used; initial and final temperature are set to be 5 K and 0.5 K, respectively. The refinements are carried out five times. In the beginning, the 20 lowest energy structure are selected out of 1000 structures to refine. Each structure, from the 20 lowest energy structures, generates the five refined structures, thus, it gives 100 (20×5) structures. For the second refinement, the 10 lowest energy structures are selected from the 100 structures, and then from each structure it generates 5 structures. In the same way to the second refinement, three additional refinements are carried out.

5.3 Results and Discussion

ITAS was tested on eight proteins of different size and topology starting from the small proteins protein G (β toplogy, 56 residues), Z domain (α topology, 71 residues) , Ubiquitin (α/β , 76 residues), DinI (α/β topology, 81 residues), KH domain (α/β topology, 89 residues), Proflin (α/β topology, 125 residues) to medium-sized Calmodulin (α/β, 148 residues) and Interleukin 1β (β, 153 residues). The details are described in table 5.1.

5.3.1 Small-Sized Proteins

For the third Igg-binding domain of protein G, the Ubiquitin, the RecA-binding protein DinI, and the C-terminal KH domain of heterogeneous nuclear ribonucleoprotein, only $^{13}C^{\alpha}$ connectivity information was employed, in order to demonstrate the efficiency of ITAS. Sequential connectivity was established based on a cutoff of 0.3 ppm. With 3D HNCA experiments this resolution is easily obtainable even for weak NMR resonances. The average of the assignment percentage of the small proteins was 17% when structure information is not used at initial iteration. Initial structures are poor, but already capture many features of the backbone structure [92]; therefore, they are useful for bootstrapping.

For protein G, only one pseudo-residue (2% assignment) was assigned in the first assignment. Therefore only 3 RDCs constraints were available for the first structure calculation. However, the rmsd between the native structure and the lowest energy structure generated by RosettaNMR was 1.5 Å . Therefore, when the structures were used for assignment, the assignment percentage was improved up to 92% . In the final structure calculation, a total of 145 RDCs were used for the structure generation and the lowest energy structure deviated by 0.7 Å from the native structure (Table 5.1). It couldn't show well how the assignment will be, if the low resolution structures are used for RDC-enhanced assignment (See chapter 4), but the result of the C-terminal KH domain of heterogeneous nuclear ribonucleoprotein shows simultaneous improvements of assignment and structure quality (Figure 5.2).

At the start of ITAS, using $^{13}C^{\alpha}$ chemical shifts with a tolerance of 0.3 ppm for establishing sequential connectivity, 8% of the C-terminal KH domain of heterogeneous nuclear ribonucleoprotein could be assigned. Thus, only 14 RDCs were available for the first structure calculation and the root mean- square-deviation (rmsd) between the high-resolution NMR structure and the 20 best-scoring structural models is on average around 14.5 Å (with a minimum value of 9.3 Å). Despite the large deviation from the native structure these initial models are already useful for improving the backbone assignment. Comparison of $^{1}D_{N,H}$, $^{1}D_{Ca,C'}$ and $^{1}D_{Ca,Ha}$ couplings that are back-calculated from these structural models with experimental values increases the assignment score to 66% (Figure 5.2). This big improvement is possible, as substructures of the 14.5 Å RosettaNMR models (especially secondary structure elements and partly their relative orientation) already agree well with the native structure.

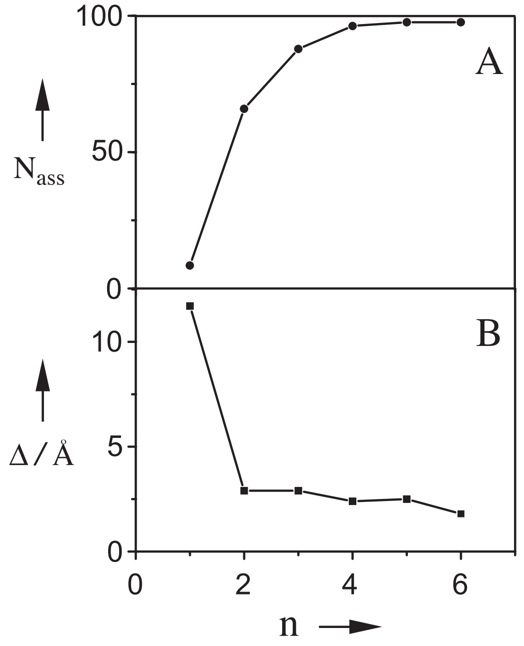

Figure 5.2: (A) Increase in the percentage of assigned residues, $\mathbf{N}_{ass}$, of KH domain during ITAS. (B) Decrease in the backbone root-mean-square-deviation, Δ, between the ITAS structure and the high-resolution NMR structure (PDB code: 1KHM). **n** is the iteration number.

At final structure calculation, total of 153 RDCs could be used for structure generation and the lowest-energy structure deviates by 1.8 Å from the high-resolution NMR structure (Table 5.1). Using all 157 experimental RDCs according to the published assignment results in a deviation of 1.6 Å . This demonstrates that the three wrong and two missing assignments, present at the end of the ITAS procedure, do not significantly deteriorate the quality of the backbone structure.

For the RecA-binding protein DinI, at the first assignment without structure information, 38% pseudo-residues was assigned. It is relatively better starting then other small proteins (Table 5.1), and finally the assignment percentage reached up to 97% including one wrong assignment. Although the assignment percentage is as high as other proteins, the deviation amounts to $5 - 6$ Å . Additional information is clearly needed to determine the high-resolution structure for this protein.

5.3.2 Medium-Sized Proteins

For proteins above 10 kDa ^{13}C$^{\alpha}$ and ^{13}C$^{\beta}$ connectivity information was used. In these cases, connectivity cutoffs were fixed at 0.5 ppm for both ^{13}C$^{\alpha}$ and ^{13}C$^{\beta}$ chemical shifts, in order to take into account the lower resolution usually obtainable from HNCACB and CBCA(CO)NH experiments.

Calmodulin poses a challenging test. It is all α-helical and comprises 148 residues in two structurally very similar domains that are connected by a flexible, seven residue linker. Using C$^{\alpha}$ and C$^{\beta}$ connectivity information, only 70 out of 144 assignable residues could be assigned. When calmodulin is subjected to ITAS 143 out of 144 residues were correctly assigned.

For the 153 residue protein interleukin 1β ^{1}D$_{NC'}$, in addition to ^{1}D$_{NH}$, ^{1}D$_{C^{\alpha}C'}$ and ^{1}D$_{C^{\alpha}H^{\alpha}}$ RDCs, were used for ITAS. Due to the large size of this protein four types of RDCs were required to obtain convergence during RosettaNMR structure calculations. Without structural information using only C$^{\alpha}$ and C$^{\beta}$ chemical shifts 60% of residues could be assigned

Table 5.1: Simultaneous assignment and structure determination for proteins varying in topological complexity and size.

Protein (PDB code)	length	chemical shifts [a]	assignment (%) (# of errors) [c]		RDC type [d]	# of RDCs	Residue range [e]	Backbone rmsd (Å) [f]	
			Start	Final				Itas	Full
interleukin 1β [g] (2I1B)	153	Cα/Cβ	60 (0)	98 (2)	1,2,3,4	553	2-140	2.1	1.9
calmodulin (1J7O)	148	Cα/Cβ	49 (0)	99 (0)	1,2,3	390	5-75	2.0	2.3
calmodulin (1J7P) [h]							82-146	1.4	2.4
profilin [g] (1ACF)	125	Cα/Cβ	82 (0)	98 (2)	1,2,3	341	2-125	1.5	1.3
KH domain (1KHM)	89	Cα	8 (0)	98 (3)	1,2,3	153	12-84	1.8	1.6
DinI (1GHH)	81	Cα	38 (0)	97 (1)	1,2,3,4	202	2-80	5.1	5.4
Ubiquitin (1D3Z)	76	Cα	22 (0)	96 (2)	1,2,4	185	2-72	2.0	1.4
Z domain [g] (2SPZ)	71	Cα/Cβ [b]	77 (0)	99 (2)	1,2,3	192	15-70	3.6	2.4
		Cα	11 (0)	78 (6)	1,2,3	153	15-70	5.0	2.4
protein G [i] (1PGB)	56	Cα	2 (0)	95 (0)	1,2,3	139	2-55	0.7	1.9

[a] Shift types used to establish sequential connectivity. Connectivity thresholds were 0.3 ppm for C^{α} and 0.5/0.5 ppm for C^{α}/C^{β}.

[b] Connectivity thresholds were 0.3 ppm for C^{α} and 0.5 ppm for C^{β}.

[c] Assignment scores were determined according to previously established methods. Start: without RDCs; Final: at the end of Itas.

[d] $^1D_{NH}$, $^1D_{C\alpha C'}$, $^1D_{C\alpha H\alpha}$, $^1D_{NC'}$ are indicated by 1, 2, 3, 4, respectively.

[e] Structured part of proteins used for RMSD calculations.

[f] For the lowest energy decoy relative to the native structure. 'Full' indicate structures obtained from published assignments.

[g] RDCs simulated using shape prediction.

[h] RDCs simulated for linker residues (77-81).

[i] Alignment tensor after first assignment run was obtained by shape prediction.

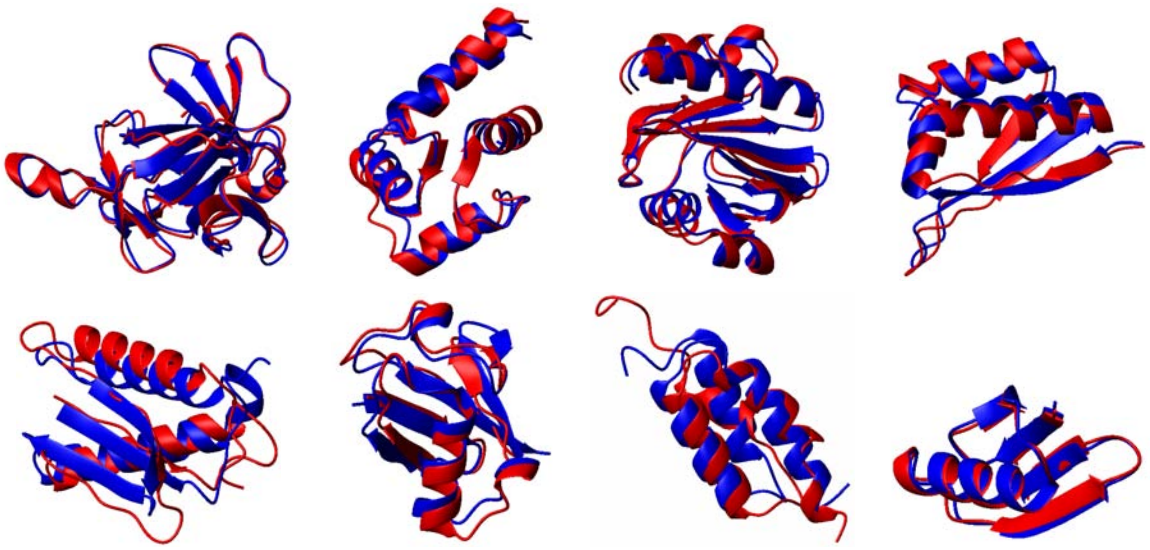

Figure 5.3: Comparisons of the ITAS structures (red) and NMR / X-ray structures (blue), determined automatically without prior assignment; at the first line, from left to right, interleukin (PDB code: 2I1B), calmodulin (PDB code: 1J7O), profilin (PDC codle: 1ACF), and KH domain (PDB code: 1KHM), at the second line, DinI (PDB code: 1GHH), Ubiquitin (PDB codle: 1D3Z), Z domain (PDB code: 2SPZ), and protein G (PDB code: 1PGB).

by MARS. During the iteration, the percentage of assigned residues was increased to a final value of 98%. The corresponding ITAS structure differs by 2.1 Å from the native structure (Figure 5.3).

5.3.3 Z Domain Protein

For all test proteins expect the Z domain of Staphylococcal protein A experimental chemical shifts and dipolar couplings were obtained from the BMRB and PDB or simulated using shape-prediction (RDCs for interleukin 1β ; see Table 5.1). Tests using these chemical shifts and RDCs are close to real applications, as they contain measurement errors, unusual chemical shifts and missing spin systems. Nevertheless, real applications are usually even more demanding as spectra can contain weak protein resonances next to noise peaks, resonances are overlapping and resonances from different spectra have to be assembled into pseudo-residues. Therefore, a larger number of C^{α}(i), C^{α}(i-1), C^{β}(i) and C^{β}(i-1) will be finally missing in the assembled pseudo-residues, an even larger number of pseudo-residues than in the BMRB will be missing completely and additional, incorrect pseudo-residues will be assembled from noise and overlapping peaks.

To put ITAS to a more rigorous test, we started from raw peak lists obtained from the automatic peak picking of NMR spectra recorded for the Z domain of Staphylococcal protein A. [122] . With ITAS, using only $^{13}C^{\alpha}$ chemical shift values to determine the sequential connectivity (cutoff of 0.3 ppm) and three types of RDCs from different internuclear vector types, the signal assignment increased from 15 to 78% and the final ITAS structure differs by 5.0 Å from the native structure. Using all 201 RDCs that were observed for the 71 residues of Z domain (corresponding to 100% assignment) RosettaNMR generates a structure that differs by 2.4 Å from the native one.

The lower quality of the ITAS structure is expected in this case, as the iterative procedure converged at a final assignment score of 78% and a significant lower number of RDCs was available for structure calculation. The six wrongly assigned residues, however, do not cause any problem as correcting the wrong assignments (thereby giving a total of 51 correct assignments) does not change the quality of the structure. This robustness against a small number of wrong assignments is achieved by selecting 25% of fragments, which are subsequently used for assembly of the ternary structure in RosettaNMR, without chemical shifts and RDCs, that is, these fragments are selected solely on the basis of agreement with multiple sequence alignment and sequence-based predicted secondary structure. Finally, the ITAS structure of the Z domain fold could be improved to 3.6 Å when both C^{α} and C^{β} connectivity information was employed. Only two assignment errors remain, the assignments of L58 and N65 are interchanged (both are located in helix 3 and do not have experimental

$^{13}C^{\alpha}$ or $^{13}C^{\beta}$ chemical shifts).

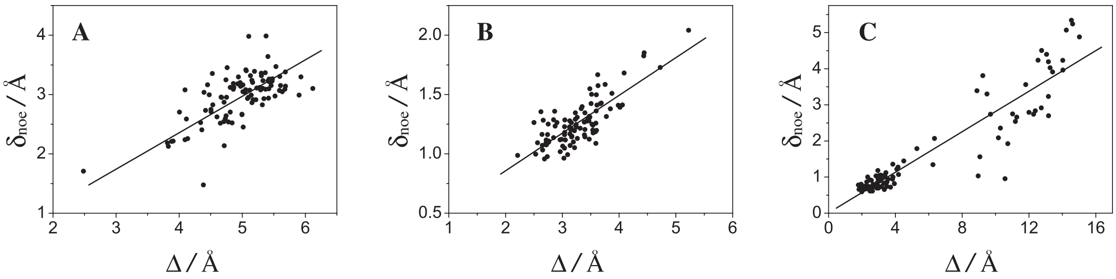

Figure 5.4: Structure validations using H^N-H^N NOEs of the 100 lowest energy structures obtained by ITAS. Δ indicates the backbone root-mean-square-deviation (rmsd) between a ITAS structure and the native structure and δ_{NOE} is the rmsd between experimental H^N-H^N and those back-calculated from ITAS structures. (A) DinI(PDB code: 1GHH, R = 0.75); (B) Ubiquitin (PDB code: 1G6J, $R = 0.79$); (C) KH domain (PDB code: 1KHM, $R = 0.93$). Here, R is linear regression coefficient.

5.3.4 Structure Validation

In our experience and the previous report [92], ITAS structures are identified as reliable if following two conditions are satisfied: (i) When more than 95% of backbone resonances are assigned at the end of the bootstrapping procedure. (ii) When the 10 lowest energy structures converge into same global fold. Addtionally, $^1H^N$- $^1H^N$NOEs, which are not used during ITAS, can be used for evaluation; the $^1H^N$- $^1H^N$NOEs are easily assignable, once the sequence specific backbone resonance assignment has been done. They can be measured with 2D NOESY and 3D ^{15}N-NOESY-HSQC spectra. The figure 5.4 shows the validation using $^1H^N$- $^1H^N$NOEs; and the average of linear regression coefficients of Ubiquitin, DinI and KH domain was 0.82.

5.4 Concluding Remarks

We introduced a method for simultaneous resonance-assignment/structure-determination. This method has been implemented into the ITAS, which integrates the programs, the MARS and the RosettaNMR for iterative resonance assignment and structure calculation. We demonstrated that protein fold can be achieved rapidly by ITAS without manual intervention starting from unassigned backbone chemical shifts and RDCs.

ITAS is applicable to small to medium-sized proteins. Medium-resolution models were generated and almost complete assignments were obtained with a few incorrect assignments. Opposite to the conventional structure determination, a few of incorrect assignments and missing assignments didn't spoil the structures, and the structures could be considered as reliable when the 10 lowest energy structures are converged into one conformation and the resonance assignment percentage is higher than 95%.

The medium resolution ITAS structure could serve as valuable initial structure for determining high-resolution 3D structures when additional NOEs are available.

Chapter 6

General Conclusion

This thesis presents automated approaches for 'sequence specific backbone resonance assignment' (Chapter 3 and 4) and 'simultaneous resonance-assignment/structure-determination' (Chapter 5). It introduces a new algorithm for the resonance assignment (Chapter 3), and shows how to incorporate residual dipolar couplings (RDCs) into conventional methods, which use either only RDC values or only sequential connectivity information for the resonance assignment (Chapter 4). Finally it introduces a new method for the simultaneous structure-determination and resonance-assignment for small and medium-sized proteins obtaining medium-resolution 3D structures. (Chapter 5).

The automation of the resonance assignment was achieved by developing the automated NMR resonance assignment computer program MARS. The automation of the simultaneous resonance-assignment and structure-determination was achieved by ITAS, adopting an iterative approach using MARS and RosettaNMR.

We demonstrate the robustness of MARS against missing pseudo-residues and missing chemical shifts in pseudo-residues, and the ability of resonance assignment for large proteins. The results mainly depend on the completeness and correctness of the input data (*e.g.* spectra quality, proper peak picking and peak grouping).

The MARS algorithm easily allows to incorporate RDC values with sequential connectivity information to enhance the assignment. Similarly, other structure information can be implemented to enhance the assignment (*e.g.* *J*-coupling constants, NOEs). It is becoming increasingly important to use known structures for resonance assignment and structure determination because advances in automation and genome sequence data will allow new protein structures to be produced faster than ever before.

In this research, RDC values were valuable for enhancing resonance assignment and rapid structure determination; especially, RDC-assisted resonance assignment played a key role for the bootstrapping procedure in ITAS.

The automation of resonance assignment allows for significant time savings for resonance assignment compared to manual assignment. Furthermore the ITAS automated structure calculation including automatic resonance assignment without any manual intervention avoids another time consuming step. The research described in this thesis contributes to the rapid protein structure determination by the automation of the resonance assignment and the structure calculation.

References

[1] Abbott, A. (2000) *Nature*, **408**, 130-132.

[2] Al-Hashimi, H.M., Gorin, A., Majumdar, A., Gosser, Y. and Patel, D.J. (2002) *J. Mol. Biol.*, **318**, 637-649.

[3] Alattia, J.R., Tong, F.K., Tong, K.I. and Ikura, M. (2000) *J. Biomol. NMR*, **16**, 181-182.

[4] Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J. (1997) *Nucleic Acids Res.*, **25**, 3389-3402

[5] Amor, J.C., Seidel, R.D., Tian, F., Kahn, R.A. and Prestegard, J.H. (2002) *J. Biomol. NMR*, **23**, 253-254.

[6] Atkinson, R.W., Saudek, V. (2002) *FEBS Lett.*, **510**, 1-4.

[7] Atkinson, R.W., Saudek, V. (1997) *J. Chem. Soc. Faraday Trans.*, **93**, 3319-3323.

[8] Atreya, H.S., Sahu, S.C., Chary, K.V.R. and Govil, G. (2000) *J. Biomol. NMR*, **17**, 125-136.

[9] Baran M.C., Huang Y.J., Moseley H.N. and Montelione G.T. (2004) *Chem Rev.*, **104**, 3541-3556.

[10] Bartels, C., Guntert, P., Billeter, M. and Wuthrich, K. (1997) *J. Comput. Chem.*, **18**, 139-149.

[11] Bartels, C., Xia, T.H., Billeter, M., Guntert, P. and Wuthrich, K. (1995) *J. Biomol. NMR*, **6**, 1-10.

[12] Bartels, C., Billeter, M., Guntert, P. and Wüthrich, K. (1996) *J. Biomol. NMR*, **7**, 207-213.

[13] Bax, A., Kontaxis, G. and Tjandra, N. 2001. Dipolar couplings in macromolecular structure determination.

[14] Bax, A. and Grzesiek, S. (1993) *Accounts Chem. Res.*, **26**, 131-138.

[15] Berardi, M.J., Sun, C.H., Zehr, M. , Abildgaard, F., Peng, J., Speck, N.A., Bushweller, J.H. (1999) *Struct. Folding Des.*, **7**, 1247-1256.

[16] Bernstein, R., Cieslar, C., Ross, A., Oschkinat, H., Freund, J. and Holak, T.A. (1993) *J. Biomol. NMR*, **3**, 245-251.

[17] Bowers, P.M., Strauss, C.E.M., Baker, D. (2000) *J. Biomol. NMR*, **18**, 311-318.

[18] Brünger, A.T., Adams, P.D., Clore, G.M., DeLano, W.L., Gros, P., Grosse-Kunstleve, R.W., Jiang, J.S., Kuszewski, J., Nilges, M., Pannu, N.S., Read, R.J., Rice, L.M., Simonson, T., Warren, G.L. (1998) *Acta Crystallogr., D: Biol. Crystallogr.*, **54**, 905-921.

[19] Brünger, A.T. (1992) *X-PLOR, Version 3.1: A system for X-ray crystallography and NMR*, Yale University Press, New Haven, CT.

[20] Buchler, N.E.G., Zuiderweg, E.R.P., Wang, H. and Goldstein, R.A. (1997) *J. Magn. Reson.*, **125**, 34-42.

[21] Bystrov, V.F. (1976) *Progr. Nucl. Magn. Reson.*, **10**, 41-82.

[22] Clore, G.M., Gronenborn, A.M. and Bax, A. (1998) *J. Magn. Reson.*, **133**, 216-221.

[23] Coggins, B.E. and Zhou, P. (2003) *J. Biomol. NMR*, **26**, 93-111.

[24] Cook, W.J., Jeffrey, L.C., Carson, M., Chen, Z.J. and Pickart, C.M. (1992) *J. Biol. Chem.*, **267**, 16467-16471.

[25] Cornilescu, G., Delaglio, F. and Bax, A. (1999) *J. of Biomol. NMR*, **13**, 289-302.

[26] Cornilescu, G., Marquardt, J.L., Ottiger, M. and Bax, A. (1998) *Journal of the American Chemical Society*, **120**, 6836-6837.

[27] Delaglio, F., Kontaxis, G., Bax, A. (2000) *J. Am. Chem. Soc.*, **122**, 2142-2143.

[28] Delaglio, F., Grzesiek, S., Vuister, G.W., Zhu, G., Pfeifer, J. and Bax, A. (1995) *J. Biomol. NMR*, **6**, 277-293.

[29] Doreleijers, J.F., Mading, S., Maziuk, D., Sojourner, K., Yin, L., Zhu, J., Markley, J.L., Ulrich, E.L. (2003) *J. Biomol. NMR*, **26**, 139-146.

[30] Dotsch, V., Oswald, R.E. and Wagner, G. (1996) *J. Magn. Reson. Ser. B*, **110**, 107-111.

[31] Eisenhaber, F., Lijnzaad, P., Argos, P., Sander, C. and Scharf, M. (1995) *Journal of Computational Chemistry*, **16**, 273-284.

[32] Evenas, J., Tugarinov, V., Skrynnikov, N.R., Goto, N.K., Muhandiram, R. and Kay, L.E. (2001) *J. Mol. Biol.*, **309**, 961-974.

[33] Friedrichs, M.S., Mueller, L. and Wittekind, M. (1994) *J. Biomol. NMR*, **4**, 703-726.

[34] Gardner, K.H., Zhang, X.C., Gehring, K. and Kay, L.E. (1998) *J. Am. Chem. Soc.*, **120**, 11738-11748.

[35] Gardner, K.H., Konrat, R., Rosen, M.K. and Kay, L.E. (1996) *J. Biomol. NMR*, **8**, 351-356.

[36] Garrett, D.S., Seok, Y.J., Liao, D.I., Peterkofsky, A., Gronenborn, A.M. and Clore, G.M. (1997) *Biochemistry*, **36**, 2517-2530.

[37] Garrett, D.S., Seok, Y.J., Peterkofsky, A., Gronenborn, A.M. and Clore, G.M. (1999) *Nat. Struct. Biol.*, **6**, 166-173.

[38] International Human Genome Sequencing Consortium (2001), *Nature*, **409**, 860-921.

[39] Grishaev, A., Llinás, M. (2002) *Proc. Natl Acad. Sci. USA*, **99**, 6707-6712.

[40] Grishaev, A., Llinás, M. (2002) *Proc. Natl Acad. Sci. USA*, **99**, 6713-6718.

[41] Gronwald, W., Willard, L., Jellard, T., Boyko, R.E., Rajarathnam, K., Wishart, D.S., Sonnichsen, F.D. and Sykes, B.D. (1998) *J. Biomol. NMR*, **12**, 395-405.

[42] Grzesiek, S. and Bax, A. (1993) *J. Biomol. NMR*, **3**, 185-204.

[43] Güntert, P., Salzmann, M., Braun, D. and Wuthrich, K. (2000) *J. Biomol. NMR*, **18**, 129-137.

[44] Güntert, P., Mumenthaler, C., Wüthrich, K. (1997) *J. Mol. Biol.*, **273**, 283-298.

[45] Hansen, M.R., Mueller, L. and Pardi, A. (1998) *Nat. Struct. Biol.*, **5**, 1065-1074.

[46] Hare, B.J. and Prestegard, J.H. (1994) *J. Biomol. NMR*, **4**, 35-46.

[47] Herrmann, T., Güntert, P., Wüthrich, K. (2002) *J. Mol. Biol.*, **319**, 209-227.

[48] Huang, Y.J., Swapna, G.V., Rajan, P.K., Ke, H., Xia, B., Shukla, K., Inouye, M., Montelione, G.T. (2003) *J. Mol. Biol.*, **327**, 521-536.

[49] Huang, Y.J. (2001) Rutgers University, New Brunswick, NJ.

[50] Hus, J.C., Prompers, J.J. and Bruschweiler, R. (2002) *J. Magn. Reson.*, **157**, 119-123.

[51] Hus, J.C., Marion, D., Blackledge, M. (2001) *J. Am. Chem. Soc.*, **123**, 1541-1542.

[52] Ikura, M., Kay, L.E., Krinks, M. and Bax, A. (1991) *Biochemistry*, **30**, 5498-5504.

[53] In Nuclear Magnetic Resonance of Biological Macromolecules, Pt B, pp. 127-174.

[54] Jain, N.U., Noble, S. and Prestegard, J.H. (2003) *J. Mol. Biol.*, **328**, 451-462.

[55] Johnson, B.A. and Blevins, R.A. (1994) *J. Biomol. NMR*, **4**, 603-614.

[56] Jung, Y.S. and Zweckstetter, M. (2004) *J. Biomol. NMR*, **30**, 11-23.

[57] Jung, Y.S. and Zweckstetter, M. (2004) *J. Biomol. NMR*, **30**, 25-35.

[58] Jung Y.S., Sharma M. and Zweckstetter M. (2004) *Angew Chem Int Ed Engl.*, **43**, 3479-3481.

[59] Karplus, M. (1959) *J. Phys. Chem.*, **30**, 11-15.

[60] Kneller, D.G. and Kuntz, I.D. (1993) *J. Cell. Biochem.*, 254-254.

[61] Koradi R, Billeter M., Wüthrich K. (1996) *J. Mol. Graph.*, **14**, 29-32.

[62] Koradi, R., Billeter, M. and Wuthrich, K. (1996) *J. Mol. Graph.*, **14**, 51-60.

[63] Kraulis, P.J. (1994) *J. Mol. Biol.* **243**, 696-718.

[64] Kuszewski, J., Schwieters, C.D., Garrett, D.S., Byrd, R.A., Tjandra, N., Clore, G.M. (2004) *J. Am. Chem. Soc.*, **26**, 6258-6273.

[65] Lemaster, D.M. and Richards, F.M. (1985) *Biochemistry*, **24**, 7263-7268.

[66] Leutner, M., Gschwind, R.M., Liermann, J., Schwarz, C., Gemmecker, G. and Kessler, H. (1998) *J. Biomol. NMR*, **11**, 31-43.

[67] Liao, D.I., Silverton, E., Seok, Y.J., Lee, B.R., Peterkofsky, A. and Davies, D.R. (1996) *Structure*, **4**, 861-872.

[68] Lipsitz, R.S., and Tjandra, N. (2004) *Annu. Rev. of Biophys. Biomol. Struct.* , **33** 387-413.

[69] Liu, A.Z., Riek, R., Wider, G., von Schroetter, C., Zahn, R. and Wuthrich, K. (2000) *J. Biomol. NMR*, **16**, 127-138.

[70] Losonczi, J.A., Andrec, M., Fischer, M.W.F. and Prestegard, J.H. (1999) *J. Magn. Reson.*, **138**, 334-342.

[71] Lukin, J.A., Gove, A.P., Talukdar, S.N. and Ho, C. (1997) *J. Biomol. NMR*, **9**, 151-166.

[72] McGuffin, L.J., Bryson, K. and Jones, D.T. (2000) *Bioinformatics*, **16**, 404-405.

[73] Meadows, R.P., Olejniczak, E.T. and Fesik, S.W. (1994) *J. Biomol. NMR*, **4**, 79-96.

[74] Meiler, J., Blomberg ,N., Nilges, M., Griesinger, C.. (2000) *J. Biomol. NMR*, **16**, 245-252.

[75] Montelione, G.T., Emerson, S.D., and Lyons, B.A. (1992) *Biopolymers* , **32**, 327-334.

[76] Montelione, G.T., Zheng, D.Y., Huang, Y.P.J., Gunsalus, K.C., Szyperski, T. (2000) *Nat. Struct. Biol.* , **7**, 982-985.

[77] Moseley, H.N.B. and Montelione, G.T. (1999) *Curr. Opin. Struct. Biol.*, **9**, 635-642.

[78] Mueller, G.A., Choy, W.Y., Yang, D.W., Forman-Kay, J.D., Venters, R.A. and Kay, L.E. (2000) *J. Mol. Biol.*, **300**, 197-212.

[79] Mumenthaler, C., Güntert, P., Braun, W., Wüthrich, K. (1997) *J. Biomol. NMR*, **10**, 351-362.

[80] Mumenthaler, C., Braun, W. (1995) *J. Mol. Biol.*, **254**, 465-480.

[81] Neidig, K.P., Geyer, M., Gorler, A., Antz, C., Saffrich, R., Beneicke, W. and Kalbitzer, H.R. (1995) *J. Biomol. NMR*, **6**, 255-270.

[82] Nilges, M., Macias, M.J., O'Donoghue, S.I., Oschkinat, H. (1997) *J. Mol. Biol.*, **269**, 408-422.

[83] Nilges, M., Clore, G.M., and Gronenborn, A.M. (1988) *FEBS Lett.*, **239**, 317-324.

[84] Nilges, M. (1995) *J. Mol. Biol.*, **245**, 645-660.

[85] Olson, J.B. and Markley, J.L. (1994) *J. Biomol. NMR*, **4**, 385-410.

[86] Oshiro, C.M., Kuntz, I.D., (1993) *Biopolymers*, **33** 107-115.

[87] Ou, H.D., Lai, H.C., Serber, Z. and Dotsch, V. (2001) *J. Biomol. NMR*, **21**, 269-273.

[88] Pardi, A., Billeter, M., and Wüthrich, K. (1984) *J. Mol. Biol.*, **180**, 741-751.

[89] Prestegard, J.H. and Kishore, A.I. (2001) *Curr. Opin. Chem. Biol.*, **5**, 584-590.

[90] Pristovsek, P., Ruterjans, H. and Jerala, R. (2002) *J. Comput. Chem.*, **23**, 335-340.

[91] Riek, R., Wider, G., Pervushin, K. and Wuthrich, K. (1999) *Proc. Natl. Acad. Sci. U. S. A.*, **96**, 4918-4923.

[92] Rohl, C.A., Baker, D. (2002) *J. Am. Chem. Soc.*, **124**, 2723-2729.

[93] Schubert, M., Smalla, M., Schmieder, P. and Oschkinat, H. (1999) *J. Magn. Reson.*, **141**, 34-43.

[94] Schwaiger, M., Lebendiker, M., Yerushalmi, H., Coles, M., Groger, A., Schwarz, C., Schuldiner, S. and Kessler, H. (1998) *Eur. J. Biochem.*, **254**, 610-619.

[95] Schwieters, C.D., Kuszewski, J.J., Tjandra, N., Clore, M.G. (2003) *J. Magn. Reson.*, **160**, 65-73.

[96] Sharff, A.J., Rodseth, L.E. and Quiocho, F.A. (1993) *Biochemistry*, **32**, 10553-10559.

[97] Simons, K.T., Strauss, C., Baker, D. (2001) *J. Mol. Biol.*, **306**, 1191-1199.

[98] Simons, K.T., Kooperberg, C., Huang, E., Baker, D. (1997) *J. Mol. Biol.*, **268**, 209-225.

[99] Skrynnikov, N.R. and Kay, L.E. (2000) *J. Biomol. NMR*, **18**, 239-252.

[100] Spera, S. and Bax, A. (1991) *J. Am. Chem. Soc.*, **113**, 549-5492.

[101] Tashiro, M., Tejero, R., Zimmerman, D.E., Celda, B., Nilsson, B. and Montelione, G.T. (1997) *J. Mol. Biol.*, **272**, 573-590.

[102] Tian, F., Valafar, H. and Prestegard, J.H. (2001) *J. Am. Chem. Soc.*, **123**, 11791-11796.

[103] Tugarinov, V., Muhandiram, R., Ayed, A. and Kay, L.E. (2002) *J. Am. Chem. Soc.*, **124**, 10025-10035.

[104] Valafar, H., Prestegard, J.H. (2004) *J. Magn. Reson.*, **167**, 228-241.

[105] Vathyam, S., Byrd, R.A. and Miller, A.F. (1999) *J. Biomol. NMR*, **14**, 293-294.

[106] Venter, J.C. et al. (2001) *Science* , **291**, 1304-1351.

[107] Venters, R.A., Farmer, B.T., Fierke, C.A. and Spicer, L.D. (1996) *J. Mol. Biol.*, **264**, 1101-1116.

[108] Vijaykumar, S., Bugg, C.E. and Cook, W.J. (1987) *J. Mol. Biol.*, **194**, 531-544.

[109] Vuister, G.W., Grzesiek, S., Delaglio, F., Wang, A.C., Tschudin, R., Zhu, G., and Bax, A. (1994) *Meth. Enzymol.* **239**, 79-105.

[110] Wagner, G., Braun, W., Havel, T.F., Schaumann, T., Go, N. and Wüthrich, K. (1987) *J. Mol. Biol.*, **196**, 611-639.

[111] Wang, Y.J. and Jardetzky, O. (2002) *J. Am. Chem. Soc.*, **124**, 14075-14084.

[112] Wang, A.C., Grzesiek, S., Tschudin, R., Lodi, P.J. and Bax, A. (1995) *J. Biomol. NMR*, **5**, 376-382.

[113] Warren, J.J. and Moore, P.B. (2001) *J. Magn. Reson.*, **149**, 271-275.

[114] Wüthrich, K. (2003) *Angew. Chem.-Int. Edit.*, **42**, 3340-3363.

[115] Wüthrich K. (2003) *Angew. Chem.*, **115**, 3462-3486.

[116] Xu, X.P. and Case, D.A. (2002) *Biopolymers*, **65**, 408-423.

[117] Xu, X.P. and Case, D.A. (2001) *J. Biomol. NMR*, **21**, 321-333.

[118] Xu, Y., Wu, J., Gorenstein, D., Braun, W.J. (1999) *Magn. Reson.*, **136**, 76-85.

[119] Yang, D.W., Venters, R.A., Mueller, G.A., Choy, W.Y. and Kay, L.E. (1999) *J. Biomol. NMR*, **14**, 333-343.

[120] Zhang, F.L. and Bruschweiler, R. (2002) *J. Am. Chem. Soc.*, **124**, 12654-12655.

[121] Zimmerman, D.E. and Montelione, G.T. (1995) *Curr. Opin. Struct. Biol.*, **5**, 664-673.

[122] Zimmerman, D.E., Kulikowski, C.A., Huang, Y.P., Feng, W.Q., Tashiro, M., Shimotakahara, S., Chien, C.Y., Powers, R., Montelione, G.T. (1997) *J. Mol. Biol.*, **269**, 592-610.

[123] Zweckstetter, M. and Bax, A. (2002) *J. Biomol. NMR*, **23**, 127-137.

[124] Zweckstetter, M., Hummer, G. and Bax, A. (2004) *Biophys. J.*, **86**, 3444-3460.

[125] Zweckstetter, M. and Bax, A. (2000) *J. Am. Chem. Soc.*, **122**, 3791-3792.

[126] Zweckstetter, M. (2003) *J. Biomol. NMR*, **27**, 41-56.

[127] Zweckstetter, M. and Bax, A. (2001) *J. Am. Chem. Soc.*, **123**, 9490-9491.

Appendix A

Usage of MARS

A.1 Getting started

A.1.1 Input

MARS is a program for backbone assignment of ^{13}C/^{15}N labeled proteins. Accordingly, following input is required:

1. Obligatory

   - parameter setup file (mars.inp)
   - chemical shift table (SPARKY format)
   - primary sequence (FASTA format)
   - secondary structure prediction file (PSIPRED format)

   When a 3D structure is known and RDCs values are available

- PDB file
- RDC table (PALES format)

2. Optional

   - table that allows restriction of the amino acid type and/or fixing of an assignment
   - table that allows fixing of sequential connectivities between pseudoresidues

A.1.2 How to run MARS

1. Prepare your chemical shift table.
2. Get your primary sequence in FASTA format.
3. Get a secondary structure prediction using the Psipred web server.
4. Adjust the parameter setup file (`mars.inp`).
5. Type '`runmars mars.inp`'

A.1.3 Output

1. Assignment result filtered for high, medium and low reliability ('`assignment_AA.out`').
2. Assignment result including alternative assignments that show up with a 10 % probability ('`assignment_AAs.out`').
3. The most likely assignment for each pseudoresidue ('`assignment_PR.out`').
4. Summary of all possible connectivities ('`connectivity.out`').
5. Summary of reduced possible connectivities ('`connectivity_reduced.out`').

6. Chemical shift table with updated assignments that can be read into SPARKY (‘`sparky_all.out`’).

7. Detailed information about predicted chemical shifts, number of reliable assignments, number of constraints for each pseudoresidue, matrices matching experimental and back-calculated chemical shifts and/or RDCs and pseudoenergy matrices at each iteration step (‘`mars.log`’).

A.2 Setting up input files

A.2.1 Obligatory

1. A Mars run is controlled by the parameter setup file (`mars.inp`). This has to be adjusted to the available experimental data. Please see below for a detailed description of the parameters. Lines with a ‘#’ sign as first character as well as empty line are ignored. Do not change the variable names such as `nIter`.

   `mars.inp (MARSHOME/example/noStructure/1ubq/input)`

```
fragSize:   5                   # Maximum length of pseudoresidue fragments
cutoffCO:   0.25                # Connectivity cutoff (ppm) of CO [0.25]
cutoffCA:   0.2                                     # Connectivity cutoff (ppm) of CA [0.5]
cutoffCB:   0.5                                     # Connectivity cutoff (ppm) of CB [0.5]
cutoffHA:   0.25                                    # Connectivity cutoff (ppm) of HA [0.25]


fixConn:    fix_con.tab         # Table for fixing sequential connectivity
fixAss:     fix_ass.tab             # Table for fixing residue type and(or) assignment


pdb:        0                   # 3D structure available [0/1]
resolution:     NO                  # Resolution of 3D structure [Angstrom]
pdbName:    NO                  # Name of PDB file (protons required!)
tensor:     NO                  # Method for obtaining alignment tensor [0/1/2/3/4]
nIter:      NO                  # Number of iterations [2/3/4]

dObsExh:    NO                  # Name of RDC table for exhaustive SVD (PALES format)
dcTab:      NO                  # Name of RDC table (PALES format)

deuterated:     0                   # Protonated proteins [0]; perdeuterated proteins [1]
sequence:   1ubq_fasta.tab      # Primary sequence (FASTA format)
```

```
secondary:  1ubq_psipred.tab    # Secondary structure (PSIPRED format)
csTab:      1ubq_cs.tab         # Chemical shift table
```

2. The chemical shift table follows the SPARKY format. It consists of a header, pseudoresidues and chemical shifts. The header has to be defined before the listing of chemical shift values starts and includes the variable names for the chemical shifts. Currently 10 different chemical shifts are supported and should be indicated by ‘CA’, ‘CA-1’, ‘CB’, ‘CB-1’, ‘CO’, ‘CO-1’, ‘HA’, ‘HA-1’, ‘H’ and ‘N’. These variable names have to be in the same order as the columns for the different chemical shifts. The first column has to be the pseudoresidue column and other columns are chemical shift columns. Pseudoresidue means the name of the group of peaks which share the same (or similar due to the experimental imperfection) N and HN chemical shifts. Lines with a ‘#’ sign as first character as well as empty lines are ignored. Missing chemical shift values have to be indicated by ‘ - ’.

   1ubq_cs.tab(MARSHOME/example/noStructure/1ubq/1ubq_cs.tab)

```
        N       CO-1        H       CA-1        CA
PR_2        123.220     170.540     8.900       54.450      55.080
PR_3?       115.340     175.920     8.320       55.080      -
PR_4?       118.110     172.450     8.610       59.570      55.210
PR_5GLY     121.000     175.320     9.300       55.210      60.620
PR_6GLY     127.520     -           8.820       60.620      54.520
PR_7        115.400     177.140     8.730       54.520      60.470
PR_8        121.330     176.910     9.100       60.470      57.580
PR_9        105.590     178.800     7.630       57.580      61.400
PR_10??     108.890     175.520     7.810       61.400      45.460
:
:
:
```

   Any combination of characters can be pseudoresidue names but the number of characters of the name has to be less than 25.

3. The primary sequence of the protein has to be in FASTA format.

   IMPORTANT: ‘X’ and ‘Z’ can not be used for the characters of a sequence.

   1ubq_fasta.tab (MARSHOME/example/noStructure/1ubq/1ubq_fasta.tab)

```
> ubq
MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN
IQKESTLHLVLRLRGG
```

4. Secondary structure prediction table has to be in Psipred format. Use the Psipred web server to get the table.

   `1ubq_psipred.tab (MARSHOME/example/noStructure/1ubq/1ubq_psipred.tab)`

```
PSIPRED PREDICTION RESULTS

Key

Conf: Confidence (0=low, 9=high)
Pred: Predicted secondary structure (H=helix, E=strand, C=coil)
  AA: Target sequence



Conf: 968896699888999867863189999999997689875658887777738887136726
Pred: CEEEEECCCCCEEEEEECCCCCHHHHHHHHHHHHCCCHHHEEEEECCEECCCCCCHHHHC
  AA: MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN
              10        20        30        40        50        60


Conf: 8988889999950699
Pred: CCCCCEEEEEEECCCC
  AA: IQKESTLHLVLRLRGG
              70
```

If a 3D structure and experimental RDCs are available:

5. All standard PDB files can be used (including MOLMOL files).

   IMPORTANT: When using shape-prediction all atoms in the PDB file will be used including pseudo atoms (ANI)

6. Experimental dipolar couplings are supplied according to the PALES table format:

   - The protein sequence should be given as shown by one or more 'DATA SEQUENCE' lines. Space characters in the sequence will be ignored.
   - The table must include columns for residue ID, three-character residue name and the atom name for both atoms that are involved in the dipolar coupling as well

as the dipolar coupling itself, its error and a weighting factor. Segment ID and Chain ID are optional.

IMPORTANT: The atom notation must match that of the PDB file.

- The table must include a 'VARS' line that labels the corresponding columns of the table.
- The table must include a 'FORMAT' line that defines the data type of the corresponding columns of the table.
- Lines with a '#' sign as first character as well as empty lines are ignored.

```
DATA SEQUENCE MQIFVKTLTG KTITLEVEPS DTIENVKAKI QDKEGIPPDQ QRLIFAGKQL
DATA SEQUENCE EDGRTLSDYN IQKESTLHLV LRLRGG

VARS   RESID_I RESNAME_I ATOMNAME_I RESID_J RESNAME_J ATOMNAME_J D       DD    W
FORMAT %5d     %6s       %6s        %5d     %6s       %6s    %9.3f   %9.3f %.2f

    2    GLN       N      2    GLN     HN     -15.524     1.000 1.00
    3    ILE       N      3    ILE     HN      10.521     1.000 1.00
    4    PHE       N      4    PHE     HN       9.648     1.000 1.00
    5    VAL       N      5    VAL     HN       6.082     1.000 1.00

    1    MET       C      2    GLN     HN       3.993     0.333 3.00
    2    GLN       C      3    ILE     HN      -5.646     0.333 3.00
    3    ILE       C      4    PHE     HN       1.041     0.333 3.00
    4    PHE       C      5    VAL     HN       0.835     0.333 3.00

    1    MET       C      2    GLN      N       2.651     0.125 8.00
    2    GLN       C      3    ILE      N      -3.768     0.125 8.00
    3    ILE       C      4    PHE      N       1.463     0.125 8.00
    4    PHE       C      5    VAL      N      -1.726     0.125 8.00

    2    GLN       N      2    GLN     HN     -15.524     1.000 1.00
    3    ILE       N      3    ILE     HN      10.521     1.000 1.00
    4    PHE       N      4    PHE     HN       9.648     1.000 1.00
    5    VAL       N      5    VAL     HN       6.082     1.000 1.00

    1    MET      HA      1    MET     CA     -38.341     1.000 0.50
    2    GLN      HA      2    GLN     CA      11.662     1.000 0.50
    3    ILE      HA      3    ILE     CA      18.424     1.000 0.50
    4    PHE      HA      4    PHE     CA      26.733     1.000 0.50
```

A.2.2 Optional

1. When additional information such as specific amino acid type labeling or initial manual assignments are available assignment of pseudoresidues can be restricted to single or to

a set of residues. The first column has to be a pseudoresidue name followed by residue numbers or amino acid types to which the assignment should be restricted. Assignments can be fixed one by one by specifying the corresponding residue numbers or restrict it to a whole residue fragment by specifying the starting and ending residue number (inclusive) connected by '-' (without a blank in between the start and end number!). At the same time, amino acid types can be fixed by specifying the corresponding one letter code. More than one amino acid type can be specified by concatenation of the corresponding one letter codes (i.e. attach additional one-letter codes without blank in between).

fix_ass.tab (MARSHOME/example/noStructure/1ubq/fix_ass.tab)

```
PR_3 3
PR_10 10-15 23 34
PR_12 12 34-36
PR_13 13
PR_14 14 16 HKT
PR_15 LFR 66-69 13-16 9 71
PR_16 EVA
```

2. Also sequential connectivities can be fixed. This is especially useful when assignment is done iteratively by Mars and manually. The first and second column are pseudoresidue names. The first column is the name of the pseudoresidue for which the intra-residual chemical shift can be connected to the inter-residual chemical shift of the pseudoresidue in the second column.

fix_con.tab (MARSHOME/example/noStructure/1ubq/fix_con.tab)

```
PR_2    PR_3
PR_3    PR_4
PR_4    PR_5
PR_11   PR_12
PR_12   PR_13
PR_13   PR_14
PR_25   PR_26
PR_26   PR_27
```

A.3 Setting up assignment parameters

1. **fragSize:** Sequential connectivity is established by matching inter- and intra-residual chemical shifts. Fragments comprising up to `fragSize` pseudoresidues are searched for exhaustively. The maximum segment length `fragSize` is a compromise between the desired total execution time of a MARS assignment run and the ability to reliably place PR segments onto the protein sequence.

   According to our tests a `fragSize` of 5 is large enough to get reliable assignments (pseudoresidue fragments with length five can in most cases be placed uniquely into the protein sequence when intra- and inter-residual $^{13}C^{\alpha}$ and $^{13}C^{\beta}$ chemical shifts are available).

   For smaller proteins or if more computing power is available larger fragment sizes (six or seven) can be employed. This is expected to be useful if, for example, no $^{13}C^{\beta}$ chemical shift information is available.

2. **cutoff:**

   (a) cutoffCO is the tolerance value (ppm) for matching intra- and inter-residual chemical shifts of C'.

   (b) cutoffCA is the tolerance value (ppm) for matching intra- and inter-residual chemical shifts of $^{13}C^{\alpha}$.

   (c) cutoffCB is the tolerance value (ppm) for matching intra- and inter-residual chemical shifts of $^{13}C^{\beta}$.

   (d) cutoffHA is the tolerance value (ppm) for matching intra- and inter-residual chemical shifts of $^{1}H^{\alpha}$.

Cutoff values should be determined according to the resolution of the spectra. If chemical shifts were obtained from standard HNCACB, CBCACONH and HNCO experiments reasonable values will be

```
Ex.)
    cutoffCO: 0.1
    cutoffCA: 0.5
    cutoffCB: 0.5
    cutoffHA: 0.1
```

Note that too small error bounds will lead to a small number of reliable assignments.

3. **fixConn:** This is optional. If you want to fix sequential connectivities, prepare a table like fix_con.tab and specify the table name, otherwise set the `fixConn` parameter to NO.

   NOTE: At one iteration step MARS generates 60 assignment solutions and extracts reliable assignments from these solutions. After the first iteration step MARS automatically fixes reliable assignments and reliable sequential connectivities obtained from previous iteration steps without user intervention. The iteration is continued until the number of reliable assignments does not increase any more. Therefore, one can see fixed assignments and fixed sequential connectivities on the screen during a MARS run although the user didn't fix anything at the start of MARS.

```
Ex.)
    fixConn: NO
        or
    fixConn: fix_conn.tab
```

4. **deuterated:** If a protein is perdeuterated, set the deuterated parameter to 1. Otherwise put it to 0.

```
Ex.)
    deuterated: 0
        or
    deuterated: 1
```

5. **sequence:** Specify the name of the file that contains the primary sequence of your protein in FASTA format.

```
Ex.)
    sequence: 1ubq_fasta.tab
```

6. **secondary:** Specify the name of the file that contains the secondary structure information of your protein in PsiPred format.

```
Ex.)
    secondary: 1ubq_psipred.tab
```

7. **csTab:** Specify the name of the file that contains the experimental chemical shifts (SPARKY format).

```
Ex.)
    csTab: 1ubq_cs.tab
```

If no 3D structure or RDCs are available, put the additional parameters as below:

```
pdb:         0
resolution:      NO
pdbName:     NO
tensor:      NO
nIter:       NO

dObsExh:     NO
dcTab:       NO
```

If a 3D structure and experimental RDC are available, following parameters have to be set up.

8. **pdb:** Put the pdb flag pdb to 1, in order to use RDCs and the known 3D structure (otherwise set it to 0).

```
Ex.)
    pdb: 1
```

9. **resolution:** Specify the resolution of your crystal structure. If you don't know the resolution of the structure because it is a homology model, set the resolution to ∼ 4.0. In this case it will be useful to perform multiple assignment runs with decreasing values for the resolution parameter (suggested range is 2.0 $<$ resolution $<$ 6.0). The optimum value corresponds to the assignment run where the maximum number of reliable assignments was obtained.

```
Ex.)
        resolution: 1.8
```

10. **pdbName:** Name of file containing the coordinates of the 3D structure. All standard PDB files (including Molmol) can be used. IMPORTANT: Protons have to be present.

```
Ex.)
        pdbName: 1ubq.pdb
```

11. **tensor:** Method for obtaining an initial estimate of the alignment tensor. Four different modes are available that can automatically be accessed by specifying 1, 2, 3 or 4. The standard mode is 3.

    - If 1 is selected, MARS will use a 'gridSearch' for estimating the orientation of the alignment tensor.
    - If 2 is selected, MARS will use exhaustive back-calculation ('exhSVD'). (`dObsExh` parameter has to be setup!)
    - If 3 is selected, MARS will use singular value decomposition ('SVD').
    - If 4 is selected, MARS will use shape-prediction ('shapePred').

```
Ex.)
        tensor: 3
```

It is recommended to use 1 or 3 for the tensor parameter. Modes 2 and 4 require additional knowledge or RDCs in nearly neutral alignment media.

12. **nIter:** MARS refines the initial alignment tensor estimate (obtained by the tensor method specified above) several times using 'SVD' based on the reliable assignments obtained in previous iteration steps. Here, the number of refinement steps of the alignment tensor, `nIter`, can be defined. According to our tests 2 refinement steps are enough.

```
Ex.)
        nIter: 2
```

13. **dObsExh:** For exhaustive back-calculation (tensor mode 2) an RDC table is required that contains RDCs of a specific amino acid type. If the tensor mode is 1, 3 or 4, put the `dObsExh` parameter to NO.

```
Ex.)
    dObsExh: NO
        or
    dObsExh: dObs_1ubq_GLY.tab
```

14. **dcTab:** Name of file that contains the experimental RDC values (in PALES format).

```
Ex.)
    dcTab: dObs_1ubq.tab
```

A.4 Output

1. **assignment_AA.out:** The first column is the residue number of the protein; the second column is the pseudoresidue that the residue is assigned to. The third column indicates the degree of reliability of each assignment. Three levels of reliability are distinguished: **H** indicates high reliability as defined in the MARS paper. **M** and **L** do not fulfill all the criteria required for **H** reliability and the specific criteria employed are adjusted automatically according to the completeness of the input data. Please see below for the robustness of assignments labeled as **M** and **L**.

   `assignment_AA.out`

```
MET_1
GLN_2      PR_2 (M)
ILE_3      PR_3 (M)
PHE_4      PR_4 (H)
VAL_5      PR_5 (H)
LYS_6
THR_7
LEU_8
THR_9      PR_9 (M)
GLY_10     PR_10 (H)
LYS_11     PR_11 (H)
THR_12     PR_12 (H)
ILE_13     PR_13 (H)
THR_14     PR_14 (H)
LEU_15     PR_15 (H)
GLU_16     PR_16 (M)
VAL_17
GLU_18
PRO_19
SER_20     PR_20 (L)
ASP_21
THR_22
    :
    :
    :
```

2. **assignment_AAs.out:** The first column is the residue number of the protein. Additional columns list pseudoresidues that can be assigned to this residue. Numbers in parenthesis are assignment probabilities. Only pseudoresidues with an assignment probability of higher than 10% are shown. `assignment_AA.out` is a subset of the assignments here.

`assignment_AAs.out`

```
MET_1
GLN_2      PR_2 (96)
ILE_3      PR_3 (100)
PHE_4      PR_4 (100)
VAL_5      PR_5 (100)
LYS_6      PR_6 (63)  PR_8 (30)
THR_7      PR_7 (76)
LEU_8      PR_8 (61)
THR_9      PR_9 (100)
GLY_10     PR_10 (100)
LYS_11     PR_11 (100)
THR_12     PR_12 (100)
ILE_13     PR_13 (100)
THR_14     PR_14 (100)
LEU_15     PR_15 (100)
GLU_16     PR_16 (100)
VAL_17     PR_17 (73)
GLU_18
PRO_19
SER_20     PR_20 (86)
ASP_21     PR_21 (65)
THR_22     PR_57 (33)
    :
    :
    :
```

3. **assignment_PR.out:** It lists the most likely assignment for each pseudoresidue present in the input chemical shift table. The first column is the pseudoresidue and the second is the residue (to which the pseudoresidue can be assigned to most likely). NOTE: 'The most likely assignment' does not mean reliable assignment and two pseudoresidues can also be assigned to one residue. The information present in assignment_PR.out is useful if a pseudoresidue is not assigned to any residue in `assignment_AAs.out` and one asks himself what it might be assigned to.

   `assignment_PR.out`

```
PR_2     GLN_2
PR_3     ILE_3
PR_4     PHE_4
PR_5     VAL_5
PR_6     LYS_6
PR_7     THR_7
PR_8     LEU_8
PR_9     THR_9
PR_10    GLY_10
PR_11    LYS_11
PR_12    THR_12
PR_13    ILE_13
PR_14    THR_14
PR_15    LEU_15
```

```
PR_16    GLU_16
PR_17    VAL_17
PR_18    GLU_18
PR_20    GLN_40
PR_21    GLN_41
PR_22    SER_57
    :
    :
    :
```

4. **connectivity.out:** All possible sequential connectivities between pseudoresidues are listed. All numbers are pseudoresidue numbers. The first column (closed by ‘– >’) is the pseudoresidue number for which connectivities are listed. If no additional entries are present no connectivities could be found for that pseudoresidue. Otherwise, all pseudoresidue numbers are listed for which the inter-residual chemical shift can be matched to the intra-residual chemical shift of the pseudoresidue in the first column.

   `connectivity.out`

```
PR_2   -->  PR_3    PR_5    PR_35   PR_43   PR_69   PR_74
PR_3   -->  PR_4    PR_23   PR_30   PR_56
PR_4   -->  PR_3    PR_5    PR_29   PR_35   PR_43
PR_5   -->  PR_6    PR_8    PR_71
PR_6   -->  PR_2    PR_7    PR_49   PR_55
PR_7   -->  PR_6    PR_8
PR_8   -->  PR_9    PR_21
PR_9   -->  PR_10
PR_10  -->  PR_11   PR_48   PR_76
PR_11  -->  PR_12   PR_42   PR_75
PR_12  -->  PR_13   PR_24   PR_62   PR_67
PR_13  -->  PR_14   PR_32
PR_14  -->  PR_15
PR_15  -->  PR_16   PR_44
PR_16  -->  PR_17   PR_69   PR_74
PR_17  -->  PR_18   PR_60
PR_18  -->  PR_16   PR_47
PR_20  -->  PR_9    PR_21
PR_21  -->  PR_22   PR_40   PR_41   PR_50   PR_73
PR_22  -->  PR_4    PR_23   PR_30   PR_56
```

5. **connectivity_reduced.out:** All possible sequential connectivities between pseudoresidues are filtered for reliable assignments (*i.e.* it is a subset of `connectivity.out`).

   `connectivity_reduced.out`

```
PR_2   -->  PR_3
PR_3   -->  PR_4
PR_4   -->  PR_5
PR_5   -->  PR_6    PR_8    PR_71
PR_6   -->  PR_2    PR_7    PR_49   PR_55
PR_7   -->  PR_6    PR_8
PR_8   -->  PR_9    PR_21
PR_9   -->  PR_10
PR_10  -->  PR_11
PR_11  -->  PR_12
PR_12  -->  PR_13
PR_13  -->  PR_14
PR_14  -->  PR_15
PR_15  -->  PR_16   PR_44
PR_16  -->  PR_17   PR_69   PR_74
PR_17  -->  PR_18   PR_60
PR_18  -->  PR_16
PR_20  -->  PR_9    PR_21
PR_21  -->  PR_22   PR_40   PR_41   PR_50   PR_73
PR_22  -->  PR_23   PR_56
```

A.5 Important points to remember

1. Spectra calibration and proper peak grouping are the most important points.

2. When grouping inter- and intra-chemical shifts try to use the same spectrum for extraction of inter- and intra-chemical shifts of a given atom type. For example, when you want to get intra- and interresidual chemical shifts of ^{13}C$^{\alpha}$, extract both chemical shifts from the HNCA spectrum. Only take the interresidual ^{13}C$^{\alpha}$chemical shift from a one way connectivity spectrum like HN(Co)CA, if the interresidual peak in the HNCA is too weak or overlapping. In that case, bigger connectivity cutoffs (`cutoffCO`, `cutoffCA`, `cutoffCB`, and `cutoffHA`) have to be used due to imperfections in spectrum calibration. Nevertheless, Mars does not care where you got the intra- and inter-chemical shifts from!

3. Be careful of folded peaks!

Appendix B

Source Code

B.1 runmars

```
#! /usr/bin/awk -f

BEGIN{
        Output="mars.log"
        NULL=""
        checkTime=1

        # Check starting time.
        if(checkTime){
                    Day1 = strftime("%d")
                    Hour1 = strftime("%H")
                    Min1 = strftime("%M")
                    Sec1 = strftime("%S")
                    print strftime("%a %b %d %H:%M:%S %Z %Y") > "mars.log"
        }

        printf "\n" > "mars.log"
        close("mars.log")



        # Initialization
        Sol_num=50
        reli_diff_num=0



        }close(ARGV[1])
        printf "-----------------------------------------------------------------------\n" >> "mars.log"
        printf "\n" >> "mars.log"
        close("mars.log")



        OK_num=0
        reli_numP=0

        # Remove CheckpointMars before running MARS
        com=sprintf("(rm -f CheckpointMars)")
        print | com
        close(com)

        # Clear a screen
```

```
com=sprintf("(clear)")
print | com
close(com)

# Run runmars_noIter
com=sprintf("($MARSHOME/runmars_noIter %s %s %s 1)",ARGV[1],ARGV[2],ARGV[3])
print | com
close(com)
OK_num++

# Check whether runmars_noIter has been finished successfully or not.
if(getline < "CheckpointMars" <=0){
    exit
}close("CheckpointMars")

# Run making_stati.awk.
com=sprintf("(awk -f $MARSHOME/making_stati.awk >> mars.log)")
print | com
close(com)
printf "\n" >> "mars.log"
close("mars.log")


# AC is the number of AA.
# RC is the number of correct reliable assignment.
# RW is the number of wrong reliable assignment.
printf "\n   AC    RC     RW\n" >> "mars.log"
printf "--------------------\n" >> "mars.log"
close("mars.log")


# Write the assignment results in mars.log file.
com=sprintf("($MARSHOME/result.awk)")
print | com
close(com)

printf "--------------------\n" >> "mars.log"
close("mars.log")


# Make mars_C_format.inp to run mars
while(getline < "mars_C_format.inp" >0) if ($1=="DipCoup:"){
    st=$2
}close("mars_C_format.inp")



### Run mars with adding 1.0 noisy to the CSs.
OK_run=1
while(OK_run==1 && OK_num<=Sol_num){

    # Copy ana_reliable_assignment.txt to ana_reliable_assignment.txt_prev.
    # It is to use to compare previous and current the number of reliable assignments.
    com=sprintf("(cp ana_reliable_assignment.txt  ana_reliable_assignment.txt_prev)")
    print | com
    close(com)

    # Make fixed connectivity table for mars
    com=sprintf("(awk -v tablename=%s -f $MARSHOME/making_fixConn.awk)",fix_conn)
    print | com
    close(com)

    # Check the number of fixed connectivities in the fixed_conn_by_Mars.tab.
    Is_fixed_conn=0
    while(getline < "fixed_conn_by_Mars.tab" >0){
        Is_fixed_conn++
    }close("fixed_conn_by_Mars.tab")
```

```
# Make fixed assignment table for mars
com=sprintf("(awk -f $MARSHOME/making_fixAssi.awk %s)",ARGV[1])
print | com
close(com)

# Check the number of fixed assignments in the fixed_assi_by_Mars.tab.
Is_fixed_assi=0
while(getline < "fixed_assi_by_Mars.tab" >0){
    Is_fixed_assi++
}close("fixed_assi_by_Mars.tab")


# Make mars_C_format_i.inp.
# The mars_C_format_i.inp has information of size of noise,
# the number of fixed assignments and connectivities.
printf "" > "mars_C_format_i.inp"
while(getline < "mars_C_format.inp" >0){
    if($5!="shift:" && $2!="connectivity:" && $2!="assignment:")
        print >> "mars_C_format_i.inp"
    if($5=="shift:")
        printf "Diviations for disturbing  chemical shift: %10.3f\n",1.0 >> "mars_C_format_i.inp"

    if($2=="connectivity:" && Is_fixed_conn>0)
        printf "Fix connectivity:    fixed_conn_by_Mars.tab\n" >> "mars_C_format_i.inp"
    if($2=="connectivity:" && Is_fixed_conn==0)
        printf "Fix connectivity:    NO\n" >> "mars_C_format_i.inp"
    if($2=="assignment:" && Is_fixed_assi>0)
        printf "Fix assignment:      fixed_assi_by_Mars.tab\n" >> "mars_C_format_i.inp"
    if($2=="assignment:" && Is_fixed_assi==0)
        printf "Fix assignment:      NO\n" >> "mars_C_format_i.inp"
}close("mars_C_format.inp")
close("mars_C_format_i.inp")

com=sprintf("(clear)")
print | com
close(com)


# If a structure and RDCs are available, then do it.
if (st==1){
    com=sprintf("($MARSHOME/mars mars_C_format_i.inp anneal st %d)",OK_num+1)
    print | com
    close(com)
    OK_num++
}

# If a structure and RDCs are not available, then do it.
if (st==0){
    com=sprintf("($MARSHOME/mars mars_C_format_i.inp anneal nost %d)",OK_num+1)
    print | com
    close(com)
    OK_num++
}

# Check assignment results
com=sprintf("($MARSHOME/result.awk)")
print | com
close(com)


# Check the number of current reliable assignment.
reli_num=0
while(getline < "ana_reliable_assignment.txt" >0)
    reli_num++
close("ana_reliable_assignment.txt")

# Compare the number of previous and current reliable assignments.
# If the number of current reliable assignments hasn't been increased,
    # then escape the loop, otherwise go to the beginning of the loop.
    if(reli_num - reli_numP <= reli_diff_num)
        OK_run=0
    else
        reli_numP=reli_num

}

printf "--------------------\n" >> "mars.log"
close("mars.log")







### Run mars with adding 0.5 noisy to the CSs.
OK_run=1
while(OK_run==1 && OK_num<=Sol_num){

    # Copy ana_reliable_assignment.txt to ana_reliable_assignment.txt_prev.
    # It is to use to compare previous and current the number of reliable assignments.
    com=sprintf("(cp ana_reliable_assignment.txt  ana_reliable_assignment.txt_prev)")
    print | com
    close(com)

    # Make fixed connectivity table for mars
    com=sprintf("(awk -v tablename=%s -f $MARSHOME/making_fixConn.awk)",fix_conn)
    print | com
    close(com)

    # Check the number of fixed connectivities in the fixed_conn_by_Mars.tab.
    Is_fixed_conn=0
    while(getline < "fixed_conn_by_Mars.tab" >0){
        Is_fixed_conn++
    }close("fixed_conn_by_Mars.tab")

    # Make fixed assignment table for mars
    com=sprintf("(awk -f $MARSHOME/making_fixAssi.awk %s)",ARGV[1])
    print | com
    close(com)

    # Check the number of fixed assignments in the fixed_assi_by_Mars.tab.
    Is_fixed_assi=0
    while(getline < "fixed_assi_by_Mars.tab" >0){
        Is_fixed_assi++
    }close("fixed_assi_by_Mars.tab")


    # Make mars_C_format_i.inp.
    # The mars_C_format_i.inp has information of size of noise,
    # the number of fixed assignments and connectivities.
    printf "" > "mars_C_format_i.inp"
    while(getline < "mars_C_format.inp" >0){
        if($5!="shift:" && $2!="connectivity:" && $2!="assignment:")
            print >> "mars_C_format_i.inp"
        if($5=="shift:")
            printf "Diviations for disturbing  chemical shift: %10.3f\n",0.5 >> "mars_C_format_i.inp"

        if($2=="connectivity:" && Is_fixed_conn>0)
            printf "Fix connectivity:    fixed_conn_by_Mars.tab\n" >> "mars_C_format_i.inp"
        if($2=="connectivity:" && Is_fixed_conn==0)
            printf "Fix connectivity:    NO\n" >> "mars_C_format_i.inp"
        if($2=="assignment:" && Is_fixed_assi>0)
            printf "Fix assignment:      fixed_assi_by_Mars.tab\n" >> "mars_C_format_i.inp"
        if($2=="assignment:" && Is_fixed_assi==0)
            printf "Fix assignment:      NO\n" >> "mars_C_format_i.inp"
    }close("mars_C_format.inp")
    close("mars_C_format_i.inp")

    com=sprintf("(clear)")
    print | com
    close(com)


    # If a structure and RDCs are available, then do it.
    if (st==1){
        com=sprintf("($MARSHOME/mars mars_C_format_i.inp anneal st %d)",OK_num+1)
        print | com
        close(com)
        OK_num++
    }

    # If a structure and RDCs are not available, then do it.
    if (st==0){
        com=sprintf("($MARSHOME/mars mars_C_format_i.inp anneal nost %d)",OK_num+1)
        print | com
        close(com)
        OK_num++
    }

    # Check assignment results
    com=sprintf("($MARSHOME/result.awk)")
    print | com
    close(com)


    # Check the number of current reliable assignment.
    reli_num=0
    while(getline < "ana_reliable_assignment.txt" >0)
        reli_num++
    close("ana_reliable_assignment.txt")

    # Compare the number of previous and current reliable assignments.
    # If the number of current reliable assignments hasn't been increased,
    # then escape the loop, otherwise go to the beginning of the loop.
    if(reli_num - reli_numP <= reli_diff_num){
        OK_run=0

        com=sprintf("(mv assignment_AA_prev.out assignment_AA.out;
        mv assignment_AAs_prev.out assignment_AAs.out;
        mv assignment_PR_prev.out assignment_PR.out  )")
        print | com
        close(com)

        com=sprintf("(mv connectivity_prev.out connectivity.out;
        mv connectivity_reduced_prev.out connectivity_reduced.out;
        mv assignment_prev.log assignment.log)")
        print | com
        close(com)

    }else{
        reli_numP=reli_num

    }
}

printf "--------------------\n\n" >> "mars.log"
close("mars.log")


# Calculate running time.
if(checkTime){
    Day2 = strftime("%d")
    Hour2 = strftime("%H")
```

```
            Min2 = strftime("%M")
            Sec2 = strftime("%S")

            if(Sec2-Sec1<0){
                Sec2+=60
                Min2-=1
            }

            if(Min2-Min1<0){
                Min2+=60
                Hour2-=1
            }

            if(Hour2-Hour1<0){
                Hour2+=24
                Day2-=1
            }

            printf "\n\nRunning time   %03d:%02d:%02d\n"
            , Hour2-Hour1+24*(Day2-Day1),Min2-Min1,Sec2-Sec1 >> "mars.log"
            close("mars.log")
        }


        com=sprintf("(cat assignment.log >> mars.log)")
        print | com
        close(com)
        close("mars.log")

        # Make sparky format files
        com=sprintf("(awk -f $MARSHOME/making_sparkyAll.awk > sparky_all.out)")
        print | com
        close(com)
        close("sparky_all.out")

        # Make sparky format files
        com=sprintf("(awk -f $MARSHOME/making_sparky.awk)")
        print | com
        close(com)


        # Remove all dummy files.
        if(all_output==NULL){
            com=sprintf("(rm -f assignment.log mars_C_format_grid_cs.inp mars_C_format.inp
            mars_C_format_i.inp mars_C_format_cs_temp.inp mars_C_format_cs.inp)")
            print | com
            close(com)

            com=sprintf("(rm -f ana_bestfirst_assignment.txt ana_reliable_assignment.txt
            ana_reliable_assignment.txt_prev ana_switched_assignment.txt psipred_3C.tab reliable_sort.tab)")
            print | com
            close(com)

            com=sprintf("(rm -f fixed_assi_by_Mars.tab fixed_type_by_Mars.tab fixed_conn_by_Mars.tab
            mars_PR_ID.tab CheckpointMars PeaksProTuermchen fixed_conn_marsFormat.tab)")
            print | com
            close(com)
        }

        printf "MARS has been successfully finished !!\n\n" > "/dev/stderr"
}
```

B.2 runmars_noIter

```
#! /usr/bin/awk -f


BEGIN{
    de=-9999
    dummy=888
    NULL=""


    # Read in the mars.inp
    while(getline < ARGV[1] >0){

        if($1=="fragSize:") LengthFrag=$2
        if($1=="cutoffCO:") cutoff_CO=$2
        if($1=="cutoffCA:") cutoff_CA=$2
        if($1=="cutoffCB:") cutoff_CB=$2
        if($1=="cutoffHA:") cutoff_HA=$2

        if($1=="fixConn:") fix_conn=$2
        if($1=="fixAss:") fix_table=$2

        if($1=="pdb:") struc=$2
        if($1=="resolution:") resol=$2
        if($1=="pdbName:") pdb=$2
        if($1=="tensor:") tensor=$2
        if($1=="nIter:") iter=$2

        if($1=="dObsExh:") dcnameEx=$2
        if($1=="jcTab:") jcname=$2
        if($1=="dcTab:") dcname=$2

        if($1=="deuterated:") d_effect=$2
        if($1=="sequence:") pdb_or_seqname=$2
        if($1=="secondary:") psipredname=$2
        if($1=="csTab:") csname=$2


    }
    close (ARGV[1])

    # Read in the MARSHOME directory
    com="(echo $MARSHOME)"
    com | getline home
    close(com)



#=======================================================================================================
#=======================================================================================================




    #To check LengthFrag
    if(LengthFrag != int(LengthFrag) || LengthFrag <3 || LengthFrag > 7){
        printf "Check the fragSize value!!! (3<= fragSize <=7)\n",LengthFrag
        exit
    }

    #To check cuoff
    if(cutoff_CO <=0 || cutoff_CA <=0 || cutoff_CB <=0 || cutoff_HA <=0){
        printf "Check the cutoff values!!!\n"
        exit
    }
```

```
#To check pdb
if(struc !=0 && struc !=1){
    printf "Check the pdb flag!!!\n"
    exit
}

#To check resol
if(struc==1) if(resol<=0 || resol>10 ){
    printf "Check the resolution value!!!\n"
    exit
}

#To check pdb
if(toupper(pdb) !="NO" && struc==1) if(getline < pdb <=0){
    printf "Check the %s table!!!\n",pdb
    exit
}close(pdb)

#To check tensor
if(struc==1) if(tensor!=int(tensor) || tensor<0 || tensor>4 ){
    printf "Check the tensor value!!! (0<= tensor <=4)\n"
    exit
}

#To check iter
if(struc==1) if(iter!=int(iter) || iter<1 || iter>5 ){
    printf "check the nIter value!!! (1<= nIter <=5)\n"
    exit
}

#To check fix_conn
if(toupper(fix_conn) !="NO"){
    if(getline < fix_conn <=0){
        printf "Check the %s table!!!\n",fix_conn
        exit
    }close(fix_conn)

    com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", fix_conn)
    print | com
    close(com)
}

#To check fix_table
if(toupper(fix_table) !="NO"){
    if(getline < fix_table <=0){
        printf "Check the %s table!!!\n",fix_table
        exit
    }close(fix_table)

    com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", fix_table)
    print | com
    close(com)
}

#To check dcnameEx
if(toupper(dcnameEx) !="NO"){
    if(getline < dcnameEx <=0){
        printf "Check the %s table!!!\n",dcnameEx
        exit
    }close(dcnameEx)

    com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", dcnameEx)
    print | com
    close(com)
}

#To check dcname
```

```
    if(toupper(dcname) !="NO"){
        if(getline < dcname <=0){
            printf "Check the %s table!!!\n",dcname
            exit
        }close(dcname)

        com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", dcnameEx)
        print | com
        close(com)
    }

    #To check pdb_or_seqname
    if(toupper(pdb_or_seqname) !="NO"){
        if(getline < pdb_or_seqname <=0){
            printf "Check the %s table!!!\n",pdb_or_seqname
            exit
        }close(pdb_or_seqname)

        com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", pdb_or_seqname)
        print | com
        close(com)
    }

    #To check psipredname
    if(toupper(psipredname) !="NO"){
        if(getline < psipredname <=0){
            printf "Check the %s table!!!\n",psipredname
            exit
        }close(psipredname)

        com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", psipredname)
        print | com
        close(com)
    }

    #To check csname
    if(toupper(csname) !="NO"){
        if(getline < csname <=0){
            printf "Check the %s table!!!\n",csname
            exit
        }close(csname)

        com=sprintf("$MARSHOME/making_Ms2Unix.awk %s", csname)
        print | com
        close(com)
    }

    #To remove
        com=sprintf("( rm -f Check_CS_header)")
    print | com
    close(com)



#====================without structures


    if (struc==0){

         #To make 1L sequence table
        com=sprintf("(awk -f $MARSHOME/making_1seq.awk st=%s %s > SEQuence1L.tab)",struc,pdb_or_seqname)
        print | com
        close(com)
        close("SEQuence1L.tab")

        #To check output SEQuence1L.tab
        if(getline < "SEQuence1L.tab" <=0){
            printf "Check the %s\n",pdb_or_seqname
            exit
        }close("SEQuence1L.tab")

        # Read in SEQuence1L.tab.
        while(getline < "SEQuence1L.tab" >0){
            if(NF==0 || $0 ~ /^[^ACDEFGHIKLMNPQRSTVWY]/){
                printf "Check the %s\n",pdb_or_seqname
                exit
            }
        }close("SEQuence1L.tab")


         #To make 3L sequence table
        com=sprintf("(awk -f $MARSHOME/making_3seq.awk SEQuence1L.tab > SEQuence3L.tab)")
        print | com
        close(com)
        close("SEQuence3L.tab")


         #To make Cs_expt.tab
        com=sprintf("(awk -v exptname=%s -f $MARSHOME/making_expt_cs.awk  > Cs_expt.tab)",csname)
        print | com
        close(com)
        close("Cs_expt.tab")


         #To check CS header
        if(getline < "Check_CS_header" <= 0){
            printf "\n"
            close("Check_CS_header")
            exit
        }
        com=sprintf("( rm -f Check_CS_header)")
        print | com
        close(com)


         #To make fixed connectivity table
        if (toupper(fix_conn) !="NO"){
        com=sprintf("(awk -f $MARSHOME/making_connFormat.awk -v table=%s > fixed_conn_marsFormat.tab)",fix_conn)
            print | com
            close(com)

            fix_conn="fixed_conn_marsFormat.tab"
        }

         #To make fixed assignment table
        if (toupper(fix_table) !="NO"){
            com=sprintf("(awk -f $MARSHOME/making_divide.awk %s)",fix_table)
            print | com
            close(com)
        }

        #To check fixed assignment table
        IsPrFixed=0
        while(getline < fix_table >0){
            for(i=2;i<=NF;i++)
                if($i ~ /[1-9]/)
                    IsPrFixed=1
            if(IsPrFixed)
                break
        }close(fix_table)

        if(IsPrFixed) if(getline < "fixed_assi_by_Mars.tab" <=0){
            printf "Check the %s table!!!\n",fix_table
            exit
        }close("fixed_assi_by_Mars.tab")
```

```
#To check fixed amino acid type
IsAminoFixed=0
while(getline < fix_table >0){
    for(i=2;i<=NF;i++)
        if($i ~ /[ACDEFGHIKLMNPQRSTVWY]/)
            IsAminoFixed=1
    if(IsAminoFixed)
        break
}close(fix_table)

if(IsAminoFixed) if(getline < "fixed_type_by_Mars.tab" <=0){
    printf "Check the %s table!!!\n",fix_table
    exit
}close("fixed_type_by_Mars.tab")

 #To check AA number
while(getline < "SEQuence3L.tab" > 0) if(NF==1)
    AA_number++
close("SEQuence3L.tab")

 #To check the number of peaks in cs table
while(getline < "Cs_expt.tab" >0){
    printf "%5d\n",$1 > "mars_PR_ID.tab"
    if(peak_num <= $1)
        peak_num=$1
}
close("Cs_expt.tab")
close("mars_PR_ID.tab")


 #To make mars_C_format.inp
com=sprintf("(awk -f $MARSHOME/making_input_script.awk peaknum=%d AAnum=%d Dcrmsd=%d exptname=%s %s
> mars_C_format.inp)",peak_num,AA_number,0,csname,ARGV[1])
print | com
close(com)
close("mars_C_format.inp")


 #To make connectivity table
com=sprintf("(awk -f $MARSHOME/making_connectivity.awk Cs_expt.tab > PeaksProTuermchen)")
print | com
close(com)
close("PeaksProTuermchen")


 #To make Cs_pred.tab table
com=sprintf("(awk -f $MARSHOME/making_Second_table.awk %s > psipred_3C.tab)",psipredname)
print | com
close(com)
close("psipred_3C.tab")

com=sprintf("(awk -f $MARSHOME/making_secondary_cs_usingScore.awk %s d_offset=%d
> Cs_pred.tab)",ARGV[1],d_effect)
print | com
close(com)
close("Cs_pred.tab")


 ######To run MARS in the condition of anneal and nostructure
com=sprintf("($MARSHOME/mars mars_C_format.inp anneal nost)")
print | com
close(com)



com=sprintf("(awk -v tablename=%s -f $MARSHOME/making_fixConn.awk)",fix_conn)
print | com
close(com)
```

```
    if(all_output==NULL){
        com=sprintf("(rm -f palesConv.tab)")
        print | com
        close(com)
    }

    printf "Mars\nMars\n" > "CheckpointMars"
    close("CheckpointMars")



exit
}
#===================== Without structures  end ==========================================




#===================== With structures ==================================================


if (struc==1){

    cspred=1
     #===================== Without CS prediction software
    if (cspred==1){

         #To make 1L sequence table
        com=sprintf("(awk -f $MARSHOME/making_1seq.awk st=%s %s > SEQuence1L.tab)",struc,pdb)
        print | com
        close(com)
        close("SEQuence1L.tab")

        #To check output SEQuence1L.tab
        if(getline < "SEQuence1L.tab" <=0){
        printf "Check the %s or %s\n",pdb_or_seqname, pdb
        exit
        }close("SEQuence1L.tab")
        while(getline < "SEQuence1L.tab" >0){
            if(NF==0 || $0 ~ /^[^ACDEFGHIKLMNPQRSTVWY]/){
                printf "Check the %s\n",pdb_or_seqname
                exit
            }
        }close("SEQuence1L.tab")


         #To make 3L sequence table
        com=sprintf("(awk -f $MARSHOME/making_3seq.awk SEQuence1L.tab > SEQuence3L.tab)")
        print | com
        close(com)
        close("SEQuence3L.tab")


         #To make fixed connectivity table
        if (toupper(fix_conn) !="NO"){
            com=sprintf("(awk -f $MARSHOME/making_connFormat.awk -v table=%s
            > fixed_conn_marsFormat.tab)",fix_conn)
            print | com
            close(com)

            fix_conn="fixed_conn_marsFormat.tab"
        }


         #To make fixed assignment table
        if (toupper(fix_table) !="NO"){
            com=sprintf("(awk -f $MARSHOME/making_divide.awk %s)",fix_table)
            print | com
            close(com)
        }


         #To make Cs_expt.tab
        com=sprintf("(awk -v exptname=%s -f $MARSHOME/making_expt_cs.awk
        >  s_expt.tab)",csname)
        print | com
        close(com)
        close("Cs_expt.tab")


        #To check CS header
        if(getline < "Check_CS_header" < 0){
            printf "\n"
            close("Check_CS_header")
            exit
        }
        com=sprintf("( rm -f Check_CS_header)")
        print | com
        close(com)


         #To make connectivity table
        com=sprintf("(awk -f $MARSHOME/making_connectivity.awk Cs_expt.tab
        > PeaksProTuermchen)")
        print | com
        close(com)
        close("PeaksProTuermchen")


         #To make Cs_pred_secondary_usingScore.tab table
        if(Psipred==NULL){
            com=sprintf("(awk -f $MARSHOME/making_Second_table.awk %s
            > psipred_3C.tab)",psipredname)
            print | com
            close(com)
            close("psipred_3C.tab")
        }

        com=sprintf("(awk -f $MARSHOME/making_secondary_cs_usingScore.awk %s d_offset=%d
        > Cs_pred.tab)",ARGV[1],d_effect)
        print | com
        close(com)
        close("Cs_pred.tab")
    }




    #==================== With CS prediction software


    if (cspred==2){

    }




     #To make Dc_expt.tab
    com=sprintf("(awk -f $MARSHOME/making_expt_dc.awk %s)",dcname)
    print | com
    close(com)


     #To make
```

```
com=sprintf("($PALESHOME/pales -daHist   -noave  -inD %s -outD histo.pal)",dcname)
print | com
close(com)

 #To read Da R
while(getline < "histo.pal" >0){
    if ($1=="DATA" && $2=="Da")      Da=$3
    if ($1=="DATA" && $2=="Da_ERR")  Da_ERR=$3
    if ($1=="DATA" && $2=="R")       R=$3
    if ($1=="DATA" && $2=="R_ERR")   R_ERR=$3
}
close("histo.pal")


 #running -daMl
if(Da_ERR !=0){
    com=sprintf("($PALESHOME/pales -daMl -inD %s -outD lm.tab  -lDa %10.3f -hDa %10.3f
    -incDa 0.05 -lR 0.05 -hR 0.65 -incR 0.05)",dcname,Da-Da_ERR,Da+Da_ERR)
    print | com
    close(com)
}
else{
    Da_ERR=2.0
    com=sprintf("($PALESHOME/pales -daMl -inD %s -outD lm.tab  -lDa %10.3f -hDa %10.3f
    -incDa 0.05 -lR 0.05 -hR 0.65 -incR 0.05)",dcname,Da-Da_ERR,Da+Da_ERR)
    print | com
    close(com)
}

 #extracting rmsd
while(getline < "lm.tab" >0){
    if ($2 ~ /^ML$/ && $3 ~ /^Da$/ && $4 !~ /^Err$/)
        $4 > 0 ? rmsd=$4*resol*0.07 : rmsd=-$4*resol*0.07

    if ($2 ~ /^ML$/ && $3 ~ /^Da$/ && $4 !~ /^Err$/)
        $4 >0 ? lm_Da = $4*0.463281e-4 : lm_Da = -($4*0.463281e-4)

    if ($2 ~ /^ML$/ && $3 ~ /^R$/ && $4 !~ /^Err$/)
        lm_Dr=lm_Da*$4
}
close("lm.tab")

 #To check the number of peaks in cs table
while(getline < "Cs_expt.tab" >0) if(csnum <= $1){
    csnum=$1
}
close("Cs_expt.tab")
if(csnum<1){
    printf "Check %s table!!!\n",csname
    exit
}

 #To make making PR_ID
com=sprintf("(awk -f $MARSHOME/making_PR_ID.awk)")
print | com
close(com)

 #To check the number of peaks in dc table
while(getline < "Dc_expt.tab" >0) if(dcnum <= $1)
    dcnum=$1
close("Dc_expt.tab")
if(dcnum<1){
    printf "Check %s table!!!\n",dcname
    exit
}

 #To check the number of peaks
csnum > dcnum ?  peak_num=csnum : peak_num=dcnum
```

```
 #To check AA number
while(getline < "SEQuence3L.tab" > 0) if(NF==1)
    AA_number++
close("SEQuence3L.tab")


 #To make mars_C_format.inp
com=sprintf("(awk -f $MARSHOME/making_input_script.awk peaknum=%d AAnum=%d Dcrmsd=%.3f
exptname=%s %s> mars_C_format.inp)",peak_num,AA_number,rmsd,csname,ARGV[1])
print | com
close(com)
close("mars_C_format.inp")


#To make dummyDc.tab
com=sprintf("(awk -f $MARSHOME/making_dummyDc.awk > PalesFormatDummy.tab)")
print | com
close(com)
close("PalesFormatDummy.tab")

 #==================== Searching alignment tensor

# Grid search
if (tensor ==1){


    # Make mars.inp file to calculate a rms matrix  between predicted and measured CS
    com=sprintf("(awk -f $MARSHOME/making_grid_script_cs.awk mars_C_format.inp
    > mars_C_format_grid_cs.inp)")
    print | com
    close(com)
    close("mars_C_format_grid_cs.inp")


    # Make mars.inp file to calculate a rms matrix between predicted and measured RDCs
    com=sprintf("(awk -f $MARSHOME/making_grid_script_dc.awk mars_C_format.inp
    > mars_C_format_grid_dc.inp)")
    print | com
    close(com)
    close("mars_C_format_grid_dc.inp")


    # Calculate rms matrix between predicted and measured CS.
    com=sprintf("($MARSHOME/mars mars_C_format_grid_cs.inp noanneal st)")
    print | com
    close(com)


    # Read in angles.
    gridnum=0
    EulerAngle = sprintf ("%s/nsc.Orientation.122.36.Quad.tab",home)
    while(getline < EulerAngle >0){
        gridnum++
        angpsi[gridnum]=$1; angtheta[gridnum]=$2; angphi[gridnum]=$3
    }
    close(EulerAngle)


    # Print out the default a rms value to initialize the minimum rmd value.
    printf "psi      theta      phi      100000000\n" > "angle_chi2.tab"
    close("angle_chi2.tab")


    # A loop to search for the alignment tensor which have the minimum rms value.
    for(i=1;i<=gridnum;i++){
        # Calculate RDCs with the given structure and three euler angles.
        com=sprintf("($PALESHOME/pales -bestFit -nofixed -pdb %s
        -inD PalesFormatDummy.tab -outD grid.pal -daMax %4.3e -daMin %4.3e -da %4.3e
        -drMax %4.3e -drMin %4.3e -dr %4.3e -psiMax %10.2f -psiMin %10.2f -psi %10.2f
        -thetaMax %10.2f -thetaMin %10.2f -theta %10.2f -phiMax %10.2f -phiMin %10.2f
        -phi %10.2f)",pdb,lm_Da,lm_Da,lm_Da,lm_Dr,lm_Dr,lm_Dr,
        angpsi[i],angpsi[i],angpsi[i],angtheta[i],angtheta[i],angtheta[i],
        angphi[i],angphi[i],angphi[i])
        print | com
        close(com)


         #To make Dc_pred.tab for MARS
        com=sprintf("(awk -v num=%s -f $MARSHOME/making_pred_dc.awk grid.pal)",AA_number)
        print | com
        close(com)


         ####### Run MARS with the structure and RDCs
        com=sprintf("($MARSHOME/mars  mars_C_format_grid_dc.inp noanneal st)")
        print | com
        close(com)


        # Read in the rms values, the output of the previous step of MARS-run.
        while(getline < "ana_sum_minimum_chi2.txt" >0)
            sum_minimum = $1
        close("ana_sum_minimum_chi2.txt")


        # Accumulate the rmd values in the table of angle_chi2.tab
        printf "%10.3f %10.3f %10.3f %10.3f\n",angpsi[i],angtheta[i],angphi[i]
        ,sum_minimum >> "angle_chi2.tab"
        close("angle_chi2.tab")

    }


    # Get the Euler angles which have the minimum rms values between predicted and measured values
    minChi2=1000000
    while(getline < "angle_chi2.tab" > 0){
        if($4 < minChi2){
            minChi2=$4;goodpsi=$1;goodtheta=$2;goodphi=$3
        }
    }
    close("angle_chi2.tab")


    # Calculate the RDCs with the Euler angles.
    com=sprintf("($PALESHOME/pales -bestFit -nofixed -pdb %s
    -inD PalesFormatDummy.tab -outD grid.pal -daMax %4.3e -daMin %4.3e -da %4.3e
    -drMax %4.3e -drMin %4.3e -dr %4.3e -psiMax %10.2f -psiMin %10.2f-psi %10.2f
    -thetaMax %10.2f -thetaMin %10.2f -theta %10.2f
    -phiMax %10.2f -phiMin %10.2f -phi %10.2f)",
    pdb,lm_Da,lm_Da,lm_Da,lm_Dr,lm_Dr,lm_Dr,
    goodpsi,goodpsi,goodpsi,goodtheta,goodtheta,goodtheta,
    goodphi,goodphi,goodphi)
    print | com
    close(com)


     #To make Dc_pred.tab for MARS-run
        com=sprintf("(awk -v num=%s -f $MARSHOME/making_pred_dc.awk grid.pal)",AA_number)
        print | com
        close(com)

}
```

```
# Calculate RDCs with the given structure and RCDs from a specific type of a residue
if (tensor ==2){
    com=sprintf("($PALESHOME/pales -bestFit -pdb %s -inD %s -exhaust
    -outD exhaust.pal)",pdb,dcname_exhaust)
    print | com
    close(com)


    # Get the alignment tensor from the result of the previous step.
    while(getline < "exhaust.pal" >0){

        if($1~/^DATA$/ && $2~/^SAUPE$/){
            zz=$3
            rr=$4
            xy=$5
            xz=$6
            yz=$7
        }
    }close("exahust.pal")


    # Calculate the RDCs with the alignment tensor obtained from the previous step.
    com=sprintf("($PALESHOME/pales -bestFit -saupe %4.3e %4.3e %4.3e %4.3e %4.3e
    -pdb %s -inD PalesFormatDummy.tab  -outD saupe.pal)",zz,rr,xy,xz,yz,pdb)
    print com
    print | com
    close(com)


     #To make Dc_pred.tab
    com=sprintf("(awk -v num=%s -f $MARSHOME/making_pred_dc.awk saupe.pal)",AA_number)
    print | com
    close(com)
}


# Run the MARS without the structure and then run with the structure.
if (tensor ==3){

    #To make mars.inp to run MARS without the structure.
    while(getline < ARGV[1] >0){
        if ($1=="pdb:")
            printf "pdb: 0              # For using structure select\n"
            > "mars_cs_tensor3.inp"
        else
            print > "mars_cs_tensor3.inp"
    }close(ARGV[1])
    close("mars_cs_tensor3.inp")


    #To make mars_C_format_cs.inp
    com=sprintf("(awk -f $MARSHOME/making_input_script.awk peaknum=%d AAnum=%d
    Dcrmsd=%d exptname=%s %s > mars_C_format_cs_temp.inp)",peak_num,AA_number,0,
    csname,"mars_cs_tensor3.inp")
    print | com
    close(com)
    close("mars_C_format_cs_temp.inp")
    while(getline < "mars_C_format_cs_temp.inp" >0){
        if ($2=="name:")
            printf "pdb name:  NO\n" > "mars_C_format_cs.inp"
        else
            print > "mars_C_format_cs.inp"
    }close("mars_C_format_cs_temp.inp")
    close("mars_C_format_cs.inp")


      #Run MARS without the structure and RDCs.
    com=sprintf("($MARSHOME/mars mars_C_format_cs.inp anneal nost)")
    print | com
    close(com)

}


# Calculate RDCs with shape prediction mode.
if (tensor ==4){

    # Calculate an alignment tensor using PALES with stPales mode.
    com=sprintf("($PALESHOME/pales -stPales -inD %s -pdb %s -outD shape.pal)",dcname,pdb)
    print | com
    close(com)


    # Get the alignment tensor from the previous result.
    while(getline < "shape.pal" >0) if($1 ~ /^DATA$/ && $2 ~ /^SAUPE$/){
        saupeMx=sprintf("%3.4e %3.4e %3.4e %3.4e %3.4e",$3,$4,$5,$6,$7)
    }
    close("shape.pal")


    # Calculate RDCs with the alignment tensor.
    com=sprintf("($PALESHOME/pales -bestFit -inD PalesFormatDummy.tab -pdb %s
    -saupe %s -outD shape.pal)",pdb,saupeMx)
    print | com
    close(com)


     #To make Dc_pred.tab for input of MARS
        com=sprintf("(awk -v num=%s -f $MARSHOME/making_pred_dc.awk shape.pal)",AA_number)
        print | com
        close(com)

}

 #==================== End of part of searching alignment tensor




#==================== Part of refining Alignment Tensor

# Run MARS with structure before refinement if the tensor-mode is 3.
if (tensor != 3){
    com=sprintf("($MARSHOME/mars mars_C_format.inp anneal st)")
    print | com
    close(com)
}



for(i=1;i<=iter;i++){

    com=sprintf("(awk -f $MARSHOME/saupeRelaiable.awk Dc_expt.tab
    | awk -f $MARSHOME/unique.awk > assignedDC.tab)")
    print | com
    close(com)
    close("assignedDC.tab")


    relNum=0
    while(getline < "assignedDC.tab" >0){
        if(($1+$4)>1 && $7!=888) relNum++
    }
    close("assignedDC.tab")
```

```
            if (relNum < 6){
                com=sprintf("(awk -f $MARSHOME/saupeAll.awk Dc_expt.tab
                | awk -f $MARSHOME/unique.awk > assignedDC.tab)")
                print | com
                close(com)
                close("assignedDC.tab")
            }


            com=sprintf("($PALESHOME/pales -bestFit -inD assignedDC.tab -pdb %s -outD bestfit.pal)",pdb)
            print | com
            close(com)

            while(getline < "bestfit.pal" >0){
                if ($1 ~ /^DATA$/ && $2 ~ /^SAUPE$/){
                    zz=$3; rr=$4; xy=$5; xz=$6; yz=$7
                }
            }
            close("bestfit.pal")


            com=sprintf("($PALESHOME/pales -bestFit -saupe %3.4e %3.4e %3.4e %3.4e %3.4e
            -inD PalesFormatDummy.tab -pdb %s -outD bestfit.pal)",zz ,rr ,xy ,xz ,yz,pdb)
            print | com
            close(com)


             #To make Dc_pred.tab
                com=sprintf("(awk -v num=%s -f $MARSHOME/making_pred_dc.awk
                bestfit.pal)",AA_number)
                print | com
                close(com)



            com=sprintf("($MARSHOME/mars mars_C_format.inp anneal st)")
            print | com
            close(com)
        }


        com=sprintf("(awk -v tablename=%s -f $MARSHOME/making_fixConn.awk)",fix_conn)
        print | com
        close(com)

        if(all_output==NULL){
            com=sprintf("(rm -f assignedDC*.tab bestfit*.pal histo.pal grid.pal shape.pal
            mars_cs_tensor3.inp palesConv.tab)")
            print | com
            close(com)

            com=sprintf("(rm -f ana_TotalKorr.txt  ana_sum_minimum_chi2.txt
            mars_C_format_grid_dc.inp angle_chi2.tab PalesFormatDummy.tab)")
            print | com
            close(com)
        }

        printf "Mars\nMars\n" > "CheckpointMars"
        close("CheckpointMars")



    }

    #==================== End of with structures ==========================
}
```

B.3 making_secondary_cs_usingScore.awk

```
BEGIN{

    # Copy the chemical shift table.
    com = sprintf ("cp $MARSHOME/cs_source.tab .")
    system(com)
    close(com)

    # Copy the table which has the information of CS effects due to the previous residue.
    com = sprintf ("cp $MARSHOME/cs_source_p.tab .")
    system(com)
    close(com)

    # Copy the table which has the information of CS effects due to the following residues.
    com = sprintf ("cp $MARSHOME/cs_source_f.tab .")
    system(com)
    close(com)

    # Copy the table which has the information of the chemical shift effects of perdeuterated state.
    com = sprintf ("cp $MARSHOME/cs_source_d.tab .")
    system(com)
    close(com)


    # Initialization
    N=0;H=0;pC=0;C=0;CA=0;pCA=0;CB=0;pCB=0;HA=0;pHA=0
    de=-9999
    S[0]=0


    # Read in the cs_source.tab.
    # Y[i,1] is the random coil state CS of amino acids.
    # The i indicates amino acid type.
    # 1,2,3,4,5 and 6 are CS of N, C, CA, CB, HN and HA, respectively.
    # YH is the helix state CS and YE is the beta strand state CS.

    i=0
    while (getline < "cs_source.tab" > 0){
        i++
        Y[i,1] = $1;Y[i,2] = $4;Y[i,3] = $7;Y[i,4] = $10;Y[i,5] = $13;Y[i,6] = $16
        YE[i,1] = $2;YE[i,2] = $5;YE[i,3] = $8;YE[i,4] = $11;YE[i,5] = $14;YE[i,6] = $17
        YH[i,1] = $3;YH[i,2] = $6;YH[i,3] = $9;YH[i,4] = $12;YH[i,5] = $15;YH[i,6] = $18
    }close("cs_source.tab")


    # Read in the cs_source_p.tab.
    # X[i,1] is CS effects due to the previous residue for the random coil state of amino acids.
    # The i indicates amino acid type.
    # 1,2,3,4,5 and 6 are CS of N, C, CA, CB, HN and HA, respectively.
    # XH is the helix state CS and XE is the beta strand state CS.

    i=0
    while (getline < "cs_source_p.tab" > 0){
        i++
        XE[i,1] = $1;XE[i,2] = $4;XE[i,3] = $7;XE[i,4] = $10;XE[i,5] = $13;XE[i,6] = $16
        X[i,1] = $2;X[i,2] = $5;X[i,3] = $8;X[i,4] = $11;X[i,5] = $14;X[i,6] = $17
        XH[i,1] = $3;XH[i,2] = $6;XH[i,3] = $9;XH[i,4] = $12;XH[i,5] = $15;XH[i,6] = $18
    }close("cs_source_p.tab")


    # Read in the cs_source_f.tab.
    # Z[i,1] is CS effects due to the previous residue for the random coil state of amino acids.
    # The i indicates amino acid type.
    # 1,2,3,4,5 and 6 are CS of N, C, CA, CB, HN and HA, respectively.
    # ZH is the helix state CS and ZE is the beta strand state CS.

    i=0
    while (getline < "cs_source_f.tab" > 0){
        i++
        ZE[i,1] = $1;ZE[i,2] = $4;ZE[i,3] = $7;ZE[i,4] = $10;ZE[i,5] = $13;ZE[i,6] = $16
        Z[i,1] = $2;Z[i,2] = $5;Z[i,3] = $8;Z[i,4] = $11;Z[i,5] = $14;Z[i,6] = $17
        ZH[i,1] = $3;ZH[i,2] = $6;ZH[i,3] = $9;ZH[i,4] = $12;ZH[i,5] = $15;ZH[i,6] = $18
    }close("cs_source_f.tab")


    # Read in the cs_source_d.tab.
    # dCa[i] is CS effects to CS of CA due to the perdeuterated state.
    # dCb[i] is CS effects to CS of CB due to the perdeuterated state.
    i=0
    while (getline < "cs_source_d.tab" > 0){
        i++
        dCa[i]=$1
        dCb[i]=$2
    }close("cs_source_d.tab")



    # Read in the psipred_3C.tab.
    # S[k] is an amino acid type, and k is the residue number of a protein.
    # AA[k] is a three letter code of primary sequence.
    k=0
    while (getline < "psipred_3C.tab" > 0){
        k++
        if($1=="ALA") {S[k] = 1;AA[k]=$1}
        if($1=="CYS") {S[k] = 2;AA[k]=$1}
        if($1=="ASP") {S[k] = 3;AA[k]=$1}
        if($1=="GLU") {S[k] = 4;AA[k]=$1}
        if($1=="PHE") {S[k] = 5;AA[k]=$1}
        if($1=="GLY") {S[k] = 6;AA[k]=$1}
        if($1=="HIS") {S[k] = 7;AA[k]=$1}
        if($1=="ILE") {S[k] = 8;AA[k]=$1}
        if($1=="LYS") {S[k] = 9;AA[k]=$1}
        if($1=="LEU") {S[k] = 10;AA[k]=$1}
        if($1=="MET") {S[k] = 11;AA[k]=$1}
        if($1=="ASN") {S[k] = 12;AA[k]=$1}
        if($1=="PRO") {S[k] = 13;AA[k]=$1}
        if($1=="GLN") {S[k] = 14;AA[k]=$1}
        if($1=="ARG") {S[k] = 15;AA[k]=$1}
        if($1=="SER") {S[k] = 16;AA[k]=$1}
        if($1=="THR") {S[k] = 17;AA[k]=$1}
        if($1=="VAL") {S[k] = 18;AA[k]=$1}
        if($1=="TRP") {S[k] = 19;AA[k]=$1}
        if($1=="TYR") {S[k] = 20;AA[k]=$1}

        # Read secondary structure scores.
        # H means helix, E beta strand, and C random coil.
        if($2=="H") {Helix[k] = $3/9;Sheet[k] = 0;Coil[k]=1-$3/9}
        if($2=="E") {Helix[k] = 0;Sheet[k] = $3/9;Coil[k]=1-$3/9}
        if($2=="C") {Helix[k] = (1-$3/9)/2;Sheet[k] = (1-$3/9)/2;Coil[k]=$3/9}
    }close("psipred_3C.tab")


    # Read in the CYSS.tab.
    # The 21 indicates the CS of oxidized CYS.
    # $1 is the residue-number of protein.
    while(getline < "CYSS.tab" >0){
        S[$1] = 21
    }close("CYSS.tab")


    # Initialization
    for(i=0;i<=k;i++)
```

```
        {n[i]=de;h[i]=de;c[i]=de;ca[i]=de;cb[i]=de;ha[i]=de}


}

END{

    # Calculate CS considering the previous and following residue effect on the CS,
    # perdeuterated effects, and secondary structure prediction scores.
    S[k+1]=0
    for(i=1;i<=k;i++){

        # Calculate N
        if (Y[S[i],1] != de)
            n[i]=(X[S[i-1],1]*Coil[i-1] + XH[S[i-1],1]*Helix[i-1] + XE[S[i-1],1]*Sheet[i-1])
            + (Y[S[i],1] + YH[S[i],1]*Helix[i] + YE[S[i],1]*Sheet[i] )
            + (Z[S[i+1],1]*Coil[i+1] + ZH[S[i+1],1]*Helix[i+1] + ZE[S[i+1],1]*Sheet[i+1])
        if (Y[S[i],1] == de)
            n[i]=de

        # Calculate HN
        if (Y[S[i],5] != de)
            h[i]=(X[S[i-1],5]*Coil[i-1] + XH[S[i-1],5]*Helix[i-1] + XE[S[i-1],5]*Sheet[i-1])
            + (Y[S[i],5] + YH[S[i],5]*Helix[i] + YE[S[i],5]*Sheet[i] )
            + (Z[S[i+1],5]*Coil[i+1] + ZH[S[i+1],5]*Helix[i+1] + ZE[S[i+1],5]*Sheet[i+1])
        if (Y[S[i],5] == de)
            h[i]=de

        # Calculate CO
        if (Y[S[i],2] != de)
            c[i]=(X[S[i-1],2]*Coil[i-1] + XH[S[i-1],2]*Helix[i-1] + XE[S[i-1],2]*Sheet[i-1])
            + (Y[S[i],2] + YH[S[i],2]*Helix[i] + YE[S[i],2]*Sheet[i] )
            + (Z[S[i+1],2]*Coil[i+1] + ZH[S[i+1],2]*Helix[i+1] + ZE[S[i+1],2]*Sheet[i+1])
        if (Y[S[i],2] == de)
            c[i]=de

        # Calculate CA
        if (Y[S[i],3] != de && d_offset==0)
            ca[i]=(X[S[i-1],3]*Coil[i-1] + XH[S[i-1],3]*Helix[i-1] + XE[S[i-1],3]*Sheet[i-1])
            + (Y[S[i],3] + YH[S[i],3]*Helix[i] + YE[S[i],3]*Sheet[i] )
            + (Z[S[i+1],3]*Coil[i+1] + ZH[S[i+1],3]*Helix[i+1] + ZE[S[i+1],3]*Sheet[i+1])
        if (Y[S[i],3] != de && d_offset==1)
            ca[i]=dCa[S[i]] + (X[S[i-1],3]*Coil[i-1] + XH[S[i-1],3]*Helix[i-1] + XE[S[i-1],3]*Sheet[i-1])
            + (Y[S[i],3] + YH[S[i],3]*Helix[i] + YE[S[i],3]*Sheet[i] )
            + (Z[S[i+1],3]*Coil[i+1] + ZH[S[i+1],3]*Helix[i+1] + ZE[S[i+1],3]*Sheet[i+1])
        if (Y[S[i],3] == de)
            ca[i]=de

        # Calculate CB
        if (Y[S[i],4] != de && d_offset==0)
            cb[i]=(X[S[i-1],4]*Coil[i-1] + XH[S[i-1],4]*Helix[i-1] + XE[S[i-1],4]*Sheet[i-1])
            + (Y[S[i],4] + YH[S[i],4]*Helix[i] + YE[S[i],4]*Sheet[i] )
            + (Z[S[i+1],4]*Coil[i+1] + ZH[S[i+1],4]*Helix[i+1] + ZE[S[i+1],4]*Sheet[i+1])
        if (Y[S[i],4] != de && d_offset==1)
            cb[i]=dCb[S[i]] + (X[S[i-1],4]*Coil[i-1] + XH[S[i-1],4]*Helix[i-1] + XE[S[i-1],4]*Sheet[i-1])
            + (Y[S[i],4] + YH[S[i],4]*Helix[i] + YE[S[i],4]*Sheet[i] )
            + (Z[S[i+1],4]*Coil[i+1] + ZH[S[i+1],4]*Helix[i+1] + ZE[S[i+1],4]*Sheet[i+1])
        if (Y[S[i],4] == de)
            cb[i]=de

        # Calculate HA
        if (Y[S[i],6] != de)
            ha[i]=(X[S[i-1],6]*Coil[i-1] + XH[S[i-1],6]*Helix[i-1] + XE[S[i-1],6]*Sheet[i-1])
            + (Y[S[i],6] + YH[S[i],6]*Helix[i] + YE[S[i],6]*Sheet[i] )
            + (Z[S[i+1],6]*Coil[i+1] + ZH[S[i+1],6]*Helix[i+1] + ZE[S[i+1],6]*Sheet[i+1])
        if (Y[S[i],6] == de)
            ha[i]=de
    }
    # Print out the CS
    for(i=1;i<=k;i++){
        printf "%5d %5s",i,AA[i]
        printf "%10.3f",n[i]
        printf "%10.3f",h[i]
        printf "%10.3f",c[i]
        printf "%10.3f",ca[i]
        printf "%10.3f",cb[i]
        printf "%10.3f",ha[i]
        printf "%10.3f",c[i-1]
        printf "%10.3f",ca[i-1]
        printf "%10.3f",cb[i-1]
        printf "%10.3f",ha[i-1]
        printf "\n"
    }

    # Remove the unnecessary files
    com = "rm -f cs_source.tab cs_source_p.tab cs_source_f.tab cs_source_d.tab"
    system(com)
    close(com)
}
```

B.4 ITAS

```
#!/usr/bin/tclsh


### Caution
# check
# lm.tab
# dip
# csf
# fasta
# psipred
# pdb
# prevAss
# step_2 1:ON(do)   0:OFF(skip)
# step_3 1:ON(do)   0:OFF(skip)

# CAUTION CAUTION
# Before restarting check Result directory
# If final result directory is ok, then fix the prevAss.
# If final result directory is not ok, then remove the result directory and fix the prevAss.
# Check reference pdb file.


########################## INPUT PARAMETERS #################################
set protein_name    1ctx
set iter            10
set chain_id        _
set iter_num        20
set time_run        1:00
###########################   Define Paths ##########################
set com_num     102
set com_name    gwdu
###########################   Define Paths of programs ##############
set pfold       /home/mpg1/MBPC/yjung/progs/blast/rosettaNMR-v1_2/rosetta_source/pFOLD.New.lnx
set make_frag   /home/mpg1/MBPC/yjung/progs/rosettaFRAGMENTS-v1_1/make_fragments.pl
set mars        ./runmars
set marsHome    /home/mpg1/MBPC/yjung/bin/MARS/Mars
set PALES       /usr/users/yjung/bin/MARS/pales/linux/pales
set molmol      /home/mpg1/MBPC/yjung/progs/MOLMOL/molmol
set bindir      /home/mpg1/MBPC/yjung/progs/Mars
set ext1        _03_06.200_v1_1
set ext2        _09_06.200_v1_1
set abext1      _03_05.200_v1_1
set abext2      _09_05.200_v1_1
########################## Restart PARAMETERS #################################
set prevAss 54
set step_2  1
set step_3  1
########################## Check PARAMETERS #################################
set   firstAA       3
set   lastAA        58
set   firstDC       3
set   lastDC        58
#############################################################################


 proc main {} \
{
    global Rundir pfold make_frag mars molmol bindir ext1  ext2 dctab  cstab\
    protein_name chain_id iter abext1 abext2 firstAA lastAA firstDC lastDC\
    PALES prevAss step_2 step_3 com_num com_name marsHome iter_num time_run


# ------> Step_0    Run MARS without structure... <-------

    set Rundir [exec pwd]
    catch {exec cp $bindir/paths.txt $Rundir/.}

    set fastaName [exec awk {{if($1=="sequence:") print $2}} mars.inp]
    set dctab [exec awk {{if($1=="dcTab:") print $2}} mars.inp ]
    set cstab [exec awk {{if($1=="csTab:") print $2}} mars.inp ]


    if {![file isfile $protein_name$chain_id.fasta]} {
        exec cp $fastaName $protein_name$chain_id.fasta
    }




    if {![file isdirectory Result0]} {
        puts "Changing directory to $Rundir"
        puts "Current path: [exec pwd]"
        puts "Protein name: $protein_name"
        puts "DC table: $dctab"
        puts "CS table: $cstab"
        puts "Iteration number: $iter"
        set call [construct_mars_input 0 "NO"]

# set mars_input mars.inp
        puts  "Running mars without a structure"
        catch {exec $mars mars_temp.inp >& mars.log } result

        catch {exec mkdir Result0} result
        catch {eval exec cp [glob ana*] Result0} result
        catch {eval exec cp [glob assignment*] Result0} result
        puts  "Total assginalbe number:\
        [exec awk {{for(i=3;i<=NF;i++) if($i!=-9999){num++;break}}END{print num}} Cs_expt.tab]\
                ([exec awk {{if($3!=-9999 && $4!=-9999) num++}END{print num}} Cs_expt.tab])"
        puts  "MARS assigned successfully without structure\n\n\n"
        set iniDir 1

    } else {
        for {set iniDir 1} {$iniDir <= $iter} {incr iniDir} {
            if {![file isdirectory Result$iniDir]} {
                break
            }
        }
    }


    exec awk -f $marsHome/making_expt_dc.awk $dctab
    set   BestAssignName     "ana_bestfirst_assignment.txt"
    set   ReliableAssignName "ana_reliable_assignment.txt"

########################### For number of Iterations#####################

    for {set i $iniDir} {$i <= $iter} {incr i} {

# ------> Step_1    Preparing Rosetta input... <-------


    # Make a directory and chagne Result directory
        catch {exec mkdir Result$i }
        set Resdir Result$i


    # Make Rosetta input
        catch {exec awk -f $bindir/RemoveAmbiguous.awk\
        argv=$ReliableAssignName $ReliableAssignName > mars.ara } result
        catch {exec awk {{print $2 , $1 }} mars.ara > mars.cpk } result
        catch {exec sort -k1n mars.cpk > mars.spk } result
        catch {exec awk -f $bindir/RemoveAmbiguous.awk argv=mars.spk mars.spk > mars.inv } result
        catch {exec awk {{print $2 , $1 }} mars.inv > mars.ass } result
        catch {exec $bindir/rosetta_format.com } result


    # Check Reliable assignment
        set  thisAss [exec awk {{if(NF==2) count++}END{print count}} mars.ass]
        puts "Result$i"
        puts "Previous assignment number: $prevAss"
        puts "The TOTAL   assignment number of reliable assignments: $thisAss"
        # puts "[exec awk {{if($1==$2 && $2!=1) print}} mars.ass]"

        if { $thisAss > $prevAss } {
            puts "RELIABLE ASSIGNMENTS are taken for the Rosetta input."
        } \
        else {
            if {[file exists $BestAssignName ]} {
                catch {exec $bindir/mostprobass $BestAssignName mars.mpa } result
                catch {exec awk {{print $2 , $1 }} mars.mpa > mars.cpk } result
                                catch {exec sort -k1n mars.cpk > mars.spk } result
                                catch {exec awk -f $bindir/RemoveAmbiguous.awk\
                                argv=mars.spk mars.spk > mars.inv } result
                catch {exec awk {{print $2 , $1 }} mars.inv > mars.ass } result
                catch {exec $bindir/rosetta_format.com } result

    # Check Bestfirst assignment
                puts "BEST-FIRST ASSIGNMENTS are taken for the Rosetta input"
                puts "The TOTAL   assignment number of most probable best-first assignment:\
                      [exec awk {{if(NF==2) count++}END{print count}} mars.ass]"

            }
        }

    # Check Rosetta CS input
        catch {exec awk -f $bindir/csRosetta.awk Cs_expt.tab > inputAll_CS.tab } result
        puts "The TOTAL   number of pseudo-residues used for Rosetta: \
        [exec awk {BEGIN{de=9999}{for(i=3;i<=NF;i++) if($i!=de){num++;break}}\
        END{print num}} assignedCsRosetta.tab]\
            ([exec awk {{if($7!=9999) num++}END{print num}} assignedCsRosetta.tab])"

    # Check Rosetta RDC input
        set count   [exec awk {
                            BEGIN{
                                de=9999
                                count=0
                            }

                            $3~/^N$/ && $5~/^HN$/ {
                                if($7!=de) count++
                            }

                            END{
                                print count
                            }
                        } assignedDcRosetta.tab
                    ]

        puts "DC  N-HN: $count"
        set count [exec awk {BEGIN{de=9999;count=0} $3~/^N$/ && $5~/^C$/ {if($7!=de) count++}\
        END{print count}} assignedDcRosetta.tab]
        puts "DC   N-C: $count"

        set count [exec awk {BEGIN{de=9999;count=0} $3~/^HN$/ && $5~/^C$/ {if($7!=de) count++}\
        END{print count}} assignedDcRosetta.tab]
        puts "DC  HN-C: $count"

        set count [exec awk {BEGIN{de=9999;count=0} $3~/^C$/ && $5~/^CA$/ {if($7!=de) count++}\
        END{print count}} assignedDcRosetta.tab]
        puts "DC  C-CA: $count"

        set count [exec awk {BEGIN{de=9999;count=0} $3~/^CA$/ && $5~/^HA$/ {if($7!=de) count++}\
        END{print count}} assignedDcRosetta.tab]
        puts "DC CA-HA: $count"

        set count [exec awk {BEGIN{de=9999;count=0} $3~/^HN$/ && $5~/^CA$/ {if($7!=de) count++}\
        END{print count}} assignedDcRosetta.tab]
        puts "DC HN-CA: $count"

        puts "Total RDC number: [exec awk {BEGIN{num=0} NF==6 && $6!=9999{num++}\
        END{print num}} assignedDcRosetta.tab]"

        set prevAss $thisAss


# ------> Step_2    Generate new fragments with couplings and shifts... <-------
        if {$step_2} {
    # Clean up the previous Rosetta input
            catch {eval exec rm -f [glob status*]} result
                catch {exec rm -f $protein_name$chain_id.psipred $protein_name$chain_id.psipred_ss2 \
                $protein_name$chain_id.checkpoint $protein_name$chain_id.check $protein_name$chain_id.chsft \
                aa$protein_name$ext1 aa$protein_name$ext2 aa$protein_name$abext1 aa$protein_name$abext2} result

    # Make Rosetta input name for CS and RDC
            catch {exec cp assignedCsRosetta.tab $protein_name$chain_id.chsft_in } result
            catch {exec cp assignedDcRosetta.tab $protein_name$chain_id.dpl } result

            puts "Generating New Fragments....."
            catch {run_fragment  $com_name $com_num}


    # In case of initial assignment is zero
            catch {exec mv aa$protein_name$abext1 aa$protein_name$ext1 } result
            catch {exec mv aa$protein_name$abext2 aa$protein_name$ext2 } result
        } else {
            set step_2  1
            puts "Generating New Fragments....."
        }


# ------> Step_3    Run it for all processors and wait for the process to get over.  <-------

        if {$step_3} {
            puts "assemblying structures......"
            catch {exec mkdir decoys}
            catch {exec mkdir score }

            set decoy_dir $Rundir/decoys
            set score_dir $Rundir/score
#           set num [Construct_rosetta $protein_name $chain_id 1000 $Rundir]
#           catch { exec $protein_name.run.com } result
            run_assembly 1000

        } else {
            set step_3  1
            set decoy_dir $Rundir/decoys
            set score_dir $Rundir/score
            puts "assemblying structures......"
        }


# -------->Step_4   Run mars with that structure <----------

    # Select the 20 best structures
        set strucList [Select_struct $score_dir/aa$protein_name.sc]

        exec rm -f  all_reliable_assignment.txt all_best_assignment.txt
        for {set str 0} { $str < 20 } { incr str } {

            set best_struc [lindex $strucList $str]
            puts "MARS assigning with the $best_struc"
            catch {exec mkdir $Resdir/Mars$str} result
            catch {exec cp $decoy_dir/$best_struc $Rundir }
            catch {exec cp $decoy_dir/$best_struc $Resdir/Mars$str}
            catch {exec $bindir/molmol.com $Rundir/$best_struc}
            catch {exec mv output.pdb $best_struc }
            set call [construct_mars_input 1 $best_struc]

            catch {exec $mars mars_temp.inp >& mars.log} info


            catch {exec cat ana_reliable_assignment.txt >> all_reliable_assignment.txt } result
            catch {exec cat ana_bestfirst_assignment.txt >> all_best_assignment.txt } result

            catch {eval exec mv [glob ana*] $Resdir/Mars$str } result
            catch {exec rm -f  $best_struc} result

        }

# Check rmsd and R
        set bestdecoy [exec awk {BEGIN{min=1000} $1~/^aa/ {if($2<min) {min=$2;name=$1}}\
        END{print name}} $Rundir/score/aa$protein_name.sc]
        catch {exec $bindir/rmsApply.tcl $protein_name.pdb\
        $Rundir/decoys/$bestdecoy $firstAA $lastAA >& rmsd.log}
       catch {set rmsd [exec awk { $1~/^Backbone$/ {print $3}} rmsd.log]}
       catch {puts "RMSD between $protein_name.pdb ($firstAA-$lastAA) and best decoy: $rmsd"}

        catch {exec $bindir/molmol.com $Rundir/decoys/$bestdecoy}
        catch {exec $PALES -bestFit -pdb output.pdb -inD $dctab -outD dc.out -s1 $firstDC -sN $lastDC}
        puts "Correlation value R between ALL RDCs with $bestdecoy   ($firstDC-$lastDC):\
        [exec awk { $2 ~/^CORR$/ && $3~/^R$/ {print $4}} dc.out]"
        exec awk {BEGIN{printf "DATA SEQUENCE\n\nVARS   RESID_I RESNAME_I ATOMNAME_I RESID_J RESNAME_J
                ATOMNAME_J D DD W\nFORMAT %s %s %s %s %s %s %s %s %s\n\n"\
                ,"%5d","%6s","%6s","%5d","%6s","%6s","%9.3f","%9.3f","%.6f"}NF==6{\
                if($3=="N"  && $5=="HN") {Invsca=1;$6=-$6}\
                if($3=="N"  && $5=="C" ) {Invsca=8;$6=-$6}\
                if($3=="HN" && $5=="C" ) {Invsca=3;}\
                if($3=="C"  && $5=="CA") {Invsca=2;}\
                if($3=="CA" && $5=="HA") {Invsca=0.5;}\
                if($3=="HN" && $5=="CA") {Invsca=3;}\
                printf "%5d %6s %6s %5d %6s %6s %9.3f %9.3f %.6f\n"\
                ,$2,"XXX",$3,$4,"XXX",$5,$6,1/Invsca,Invsca}} assignedDcRosetta.tab > temp.dObs

        catch {exec $bindir/molmol.com $protein_name.pdb}
        catch {exec $PALES -bestFit -pdb output.pdb -inD temp.dObs -outD dc.out -s1 $firstDC -sN $lastDC}
        puts "Correlation value R between ASSIGNED RDCs with $protein_name.pdb   ($firstDC-$lastDC):\
        [exec awk { $2 ~/^CORR$/ && $3~/^R$/ {print $4}} dc.out]\n\n"

# Move Rosetta input files
        catch {exec mv $protein_name$chain_id.chsft  aa$protein_name$ext1 aa$protein_name$ext2 $Resdir}  result
        catch {exec mv assignedCsRosetta.tab $Resdir/$protein_name$chain_id.chsft_in  } result
        catch {exec mv assignedDcRosetta.tab $Resdir/$protein_name$chain_id.dpl  } result
        catch {exec mv $decoy_dir $Resdir }
        catch {exec mv $score_dir $Resdir }

# Making input source for next Rosetta input
        catch {exec sort -u all_reliable_assignment.txt > $Rundir/ana_reliable_assignment.txt} result
        catch {exec sort -n all_best_assignment.txt > $Rundir/ana_bestfirst_assignment.txt } result
        catch {exec cp ana_reliable_assignment.txt $Resdir/ana_reliable_assignment.txt_nextInput}
        catch {exec cp ana_bestfirst_assignment.txt $Resdir/ana_bestfirst_assignment.txt_nextInput}
        catch {eval exec cp [glob assignment*] $Resdir} result
        puts "\n\n"
    }

# Calculating Correlation R
    catch {exec $bindir/molmol.com $protein_name.pdb}
    catch [exec $PALES -bestFit -pdb output.pdb -inD $dctab -outD dc.out -s1 $firstDC -sN $lastDC]
    puts "Correlation value R of $protein_name.pdb ($firstDC-$lastDC):\
    [exec awk { $2 ~/^CORR$/ && $3~/^R$/ {print $4}} dc.out]\n\n"
    catch {exec rm -f x0 auto_i.inp auto.inp inputAll_CS.tab} result

# Clean up directory
    catch {exec rm -f aa$protein_name$chain_id.psipred aa$protein_name$chain_id.psipred_ss2\
    aa$protein_name$chain_id.checkpoint temp.dObs x0 mars.ara mars.ass mars.inv mars.mpa\
    output.pdb dc.out auto.inp auto_i.inp rotated.pdb BestStruc} result
    catch {eval exec rm -f [glob aa$protein_name*.pdb] } result
    catch {eval exec rm -f [glob bestfit*.pal] } result
    catch {eval exec rm -f [glob assignedDC*tab] } result

}

proc construct_mars_input { struc_flag pdbname } {

    exec    awk -v flag=$struc_flag -v name=$pdbname {
                {
                    if($1=="pdb:")
                        $2=flag

                    if($1=="pdbName:")
                        $2=name

                    if($1=="resolution:")
                        $2=4.0

                    print
                }
            } mars.inp > mars_temp.inp
```

```
}

 proc Select_struct { scorefile }\
{


    puts "sort -k2 -n $scorefile > sorted"
    catch { exec sort -k2 -n $scorefile > sorted}
    catch {exec awk {/aa/{printf "%s\n",$1}} sorted > BestStruc}
    set sortedstruc [exec head -20 BestStruc]
    set best_struc [split $sortedstruc "\n"]
    return $best_struc

}
 proc Construct_rosetta {protein_name chain_id struc_count Rundir} \
{
    set pfold   /home/mpg1/MBPC/yjung/progs/blast/rosettaNMR-v1_2/rosetta_source/pFOLD.New.lnx

    set input [open "$protein_name.run.com" w 0600]
    puts $input "#!/bin/tcsh"
    puts $input "set HOSTFILE=/home/mpg1/MBPC/yjung/progs/rosettaFRAGMENTS-v1_1/host"
    puts $input "foreach host \( \`cat \$HOSTFILE \` \)"
    puts $input "\( ssh -C \$USER@\$host \"cd $Rundir; \(nice -19 $pfold aa $protein_name\
    $chain_id -no_filters -nstruct $struc_count \& \) \" \) \>\& assembly.log \& "
    puts $input "sleep 15"
    puts $input "end"
    puts $input "wait"
    close $input

    catch {exec chmod 700 $protein_name.run.com} result
}

proc Construct_fragment_script {protein_name chain_id Rundir}\
{
    set make_frag /home/mpg1/MBPC/yjung/progs/rosettaFRAGMENTS-v1_1/make_fragments.pl

    set file [open "$protein_name.frag.com" w 0600]
    puts $file "#!/bin/tcsh"
    puts $file "$make_frag -id $protein_name$chain_id -rundir $Rundir $protein_name$chain_id.fasta -verbose"
    puts $file "wait"
    close $file

    catch {exec chmod 700 $protein_name.frag.com} result
}


proc Rmsd_calc_script {ref_pdbname pdbname}\
{
    set file [open "rmsd.com" w 0600]
    puts $file "#!/bin/tcsh"
    puts $file "molmol -f /home/mpg1/MBPC/yjung/progs/Mars/rmsd.mac -t $ref_pdbname $pdbname"
    puts $file "exit"
    exec chmod 700 rmsd.com

    catch {exec ./rmsd.com} result
    puts $result
}


proc run_fragment_bsub  {protein_name chain_id Rundir} {

    set make_frag /home/mpg1/MBPC/yjung/progs/rosettaFRAGMENTS-v1_1/make_fragments.pl
    set gwd gwdl
    exec bsub -n 1 -W 48:00 -M 900000 -K "  $make_frag -id $protein_name$chain_id\
    -rundir $Rundir $protein_name$chain_id.fasta -verbose >& fragment.log "

}
```

```
proc run_fragment  {com_name com_num} {
    global protein_name chain_id Rundir make_frag

    puts "In $com_name$com_num it's running"

    puts "$make_frag -id $protein_name$chain_id -rundir $Rundir $protein_name$chain_id.fasta\
    -verbose   >& fragment.log"
    exec $make_frag -id $protein_name$chain_id -rundir $Rundir $protein_name$chain_id.fasta\
    -verbose   >& fragment.log
    puts "It's done.\n\n\n"
}


proc run_assembly {struc_count} {

    global pfold Rundir protein_name chain_id bindir iter_num time_run

    set nameFile $Rundir/decoys/aa$protein_name$struc_count.pdb
    set rosettaDone 0

    puts "Login gwdg-wk and gwdg-wb machines"
    run_assemblyIngwdg 1000

    while {$rosettaDone!=1} {

        if {[file exists $nameFile]} {
            if {[exec ls -l $nameFile | awk {{print $5}}]} {
                set rosettaDone 1
            }
        }

        catch {exec bjobs >& bjobs.tab}
        if {[exec wc bjobs.tab | awk {{print $1}}] < 20} {
            puts "bsub  -e \"/home/temp1/yjung/%J.err\" -W $time_run -q \"gwdg-pcser\" \"$pfold aa\
            $protein_name $chain_id -no_filters -nstruct $struc_count > /dev/null\""
            catch {exec  bsub -e "/home/temp1/yjung/%J.err" -W $time_run -q "gwdg-pcser"   "$pfold aa\
            $protein_name $chain_id -no_filters -nstruct $struc_count > /dev/null"} pcser_message

            puts "bsub  -e \"/home/temp1/yjung/%J.err\" -W $time_run -m \"hgrouppcpar\" \"$pfold aa\
            $protein_name $chain_id -no_filters -nstruct $struc_count > /dev/null\""
            catch {exec  bsub -e "/home/temp1/yjung/%J.err" -W $time_run -m "hgrouppcpar"  "$pfold aa\
            $protein_name $chain_id -no_filters -nstruct $struc_count > /dev/null"} pcpar_message
            exec sleep 1
        }
    }

}


proc run_assemblyIngwdg {struc_count} {

    global protein_name chain_id Rundir pfold


    foreach gwdg { gwdg-wk01 gwdg-wk02 gwdg-wk03 gwdg-wk04 gwdg-wk05 gwdg-wk06 gwdg-wk07 gwdg-wk08\
        gwdg-wk09 gwdg-wk10 gwdg-wk11 gwdg-wk12 gwdg-wk13  gwdg-wk14  gwdg-wk15\
        gwdg-wk20 gwdg-wb01 gwdg-wb02 gwdg-wb03 gwdg-wb04 gwdg-wb05 gwdg-wb06} {
        puts "login $gwdg"
        exec  ssh -C -n $gwdg " tcsh  ; cd $Rundir ; ( nice -19 $pfold aa $protein_name\
        $chain_id -no_filters -nstruct $struc_count & ) " >& assembly.log &
        exec sleep 10
    }
}

main
exit
```

Lebenslauf

Name	**Young-Sang Jung**
Geburtsdatum	3.Dezember 1970
Geburtsort	Pusan (Südkorea)
Staatsangehörigkeit	Koreanisch (Südkorea)
Religion	Evangelisch
Familienstand	ledig

Ausbildung

1989	Allgemeine Hochschulreife (Dong-In-Schule, Pusan)
1989-1990	Beginn des Studiums der Physik an der Universität Dong-A in Pusan
1990-1992	Wehrdienst
1993-1996	Fortsetzung des Studiums der Physik an der Universität Dong-A in Pusan
1996	Bachelor of Physics
1997-1999	Anfertigung einer Master-Arbeit am Institut für Physik an der Yonsei Universität in Seoul, Südkorea Thema der Master-Arbeit: ”‘An NMR Investigation of $LiMn_2O_4$”’; Betreuer: Prof. Samhyeon Lee
1997-1999	Wissenschaftlicher Mitarbeiter am Institut für Physik an der Yonsei Universität in Seoul, Südkorea
1997-1999	Stipendium für studentische Exzellenz der staatlichen Bildungsbehörde
1999	Master of Physics
2000-2002	Wissenschaftlicher Mitarbeiter am Institut für Biochemie an der Yonsei Universität in Seoul, Südkorea
2002-2005	Wissenschaftlicher Mitarbeiter am Institut für biophysikalische Chemie, Abteilung NMR basierte Strukturbiologie (Prof.Griesinger) in Göttingen
2002-2005	Anfertigung einer Doktorarbeit unter der Anleitung von Dr. Markus Zweckstetter und Prof. Dr. Christian Griesinger; Thema der Doktorarbeit: ”‘Rapid Determination of Protein Structures in Solution Using NMR Dipolar Couplings”’.

Göttingen, den 23.12.2004