User Guide for NOLB : Non-Linear Normal Mode Analysis

Sergei Grudinin

User Guide

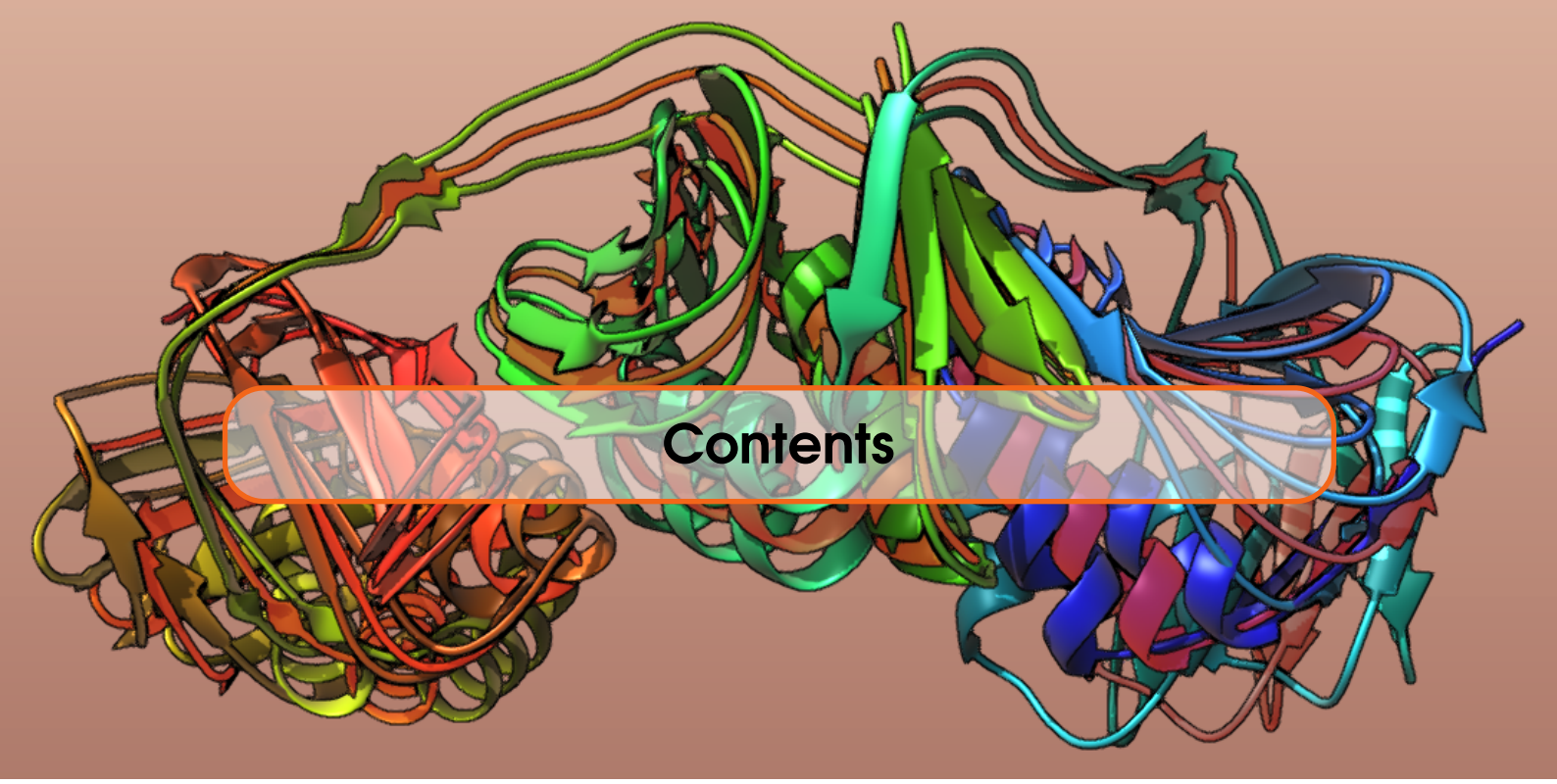
NOLB : Non-Linear Normal Mode Analysis, version 0.8, February 2018

Sergei Grudinin

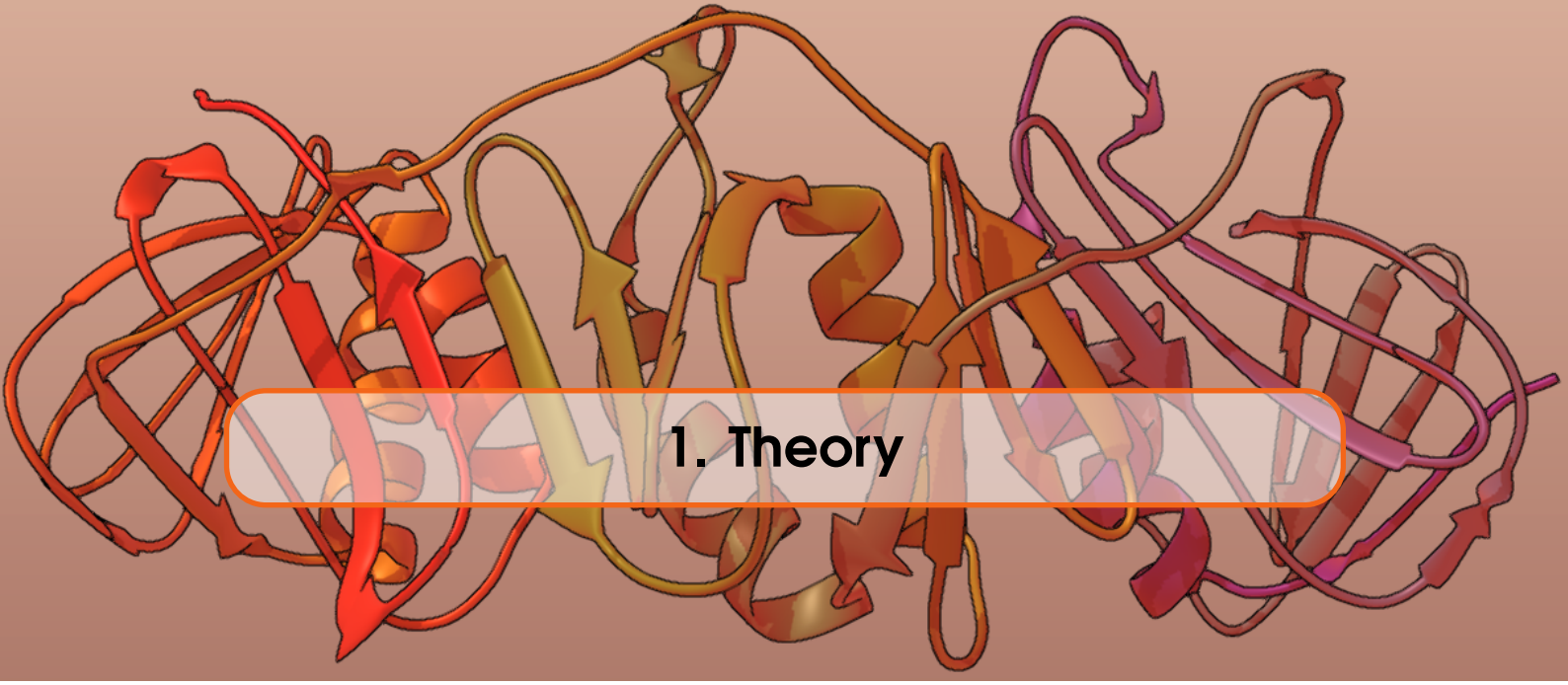
Inria/CNRS Grenoble, France

email : sergei.grudinin@inria.fr

http://team.inria.fr/nano-d/software/nolb-normal-modes/



Contents



1. Theory

1.1 Introduction

Normal mode analysis (NMA) is an old and well established technique [1] that has recently found many new applications in the field of structural biology and structural bioinformatics [2]. NMA uses a quadratic approximation of the potential energy, and thus it produces linear deformations of the initial structure, which are accurate only for small-amplitude motions. Larger amplitudes can destroy, for example, the secondary structure and break interatomic bonds, when NMA is applied to a protein. An obvious circumvention for this problem will be to take smaller amplitude steps and iteratively recompute and diagonalize the Hessian matrix from the updated positions. This procedure would indeed produce a more realistic deformation of the initial structure thanks to the non-linearity of the obtained deformation. However, such an approach requires multiple diagonalization steps, which may be computationally expensive for some of the applications. Thus, multiple attempts were made to introduce non-linear deformations without the need of multiple diagonalizations. Here we present a new scheme for *non-linear normal mode analysis* that extrapolates a motion computed from *instantaneous linear and angular velocities* to large amplitudes. The scheme can be considered as an evolution of the widely used rotations-translations of blocks (RTB) method.

1.2 NMA theory

Let us consider a molecular system with N_a atoms at an equilibrium position $q_0 \in \mathbb{R}^{3N_a}$. We call $V : \mathbb{R}^{3N_a} \mapsto \mathbb{R}$ the potential energy of our molecular system. We introduce $q : \mathbb{R}^+ \mapsto \mathbb{R}^{3N_a}$, a small time-dependent molecular displacement of our system around q_0. The potential energy V in the vicinity of q_0 is given by its quadratic approximation, which allows to *analytically* solve the Newton's equation of motion,

$$M(\ddot{q} + \ddot{q_0}) + \nabla V(q_0 + q) \approx M\ddot{q} + Kq = 0, \tag{1.1}$$

where M is the diagonal mass matrix, and K is the *Hessian matrix* (or *stiffness matrix*) of the potential energy V evaluated at the equilibrium position q_0. We then compute the square matrix of eigenvectors L and the diagonal matrix of eigenvalues Λ of the *mass-weighted stiffness matrix* $K_w = M^{-1/2}KM^{-1/2}$,

$$K_w = L\Lambda L^T. \tag{1.2}$$

We call $\eta : \mathbb{R}^+ \mapsto \mathbb{R}^{3N_a}$ the projection of q into the eigenspace of K_w and we call $(\lambda_i)_{i=0\ldots3N_a}$ the diagonal values in Λ. Then, left multiplication of eq. 1.1 by $L^T M^{1/2}$ gives the following system of

uncoupled equations,

$$\begin{aligned} \eta &= L^T M^{1/2} q \\ \ddot{\eta}_i + \lambda_i \eta_i &= 0 \quad i = 1 \ldots 3N_a \end{aligned}, \tag{1.3}$$

which can be solved by the classical ordinary differential equation (ODE) theory.

1.3 RTB projection method

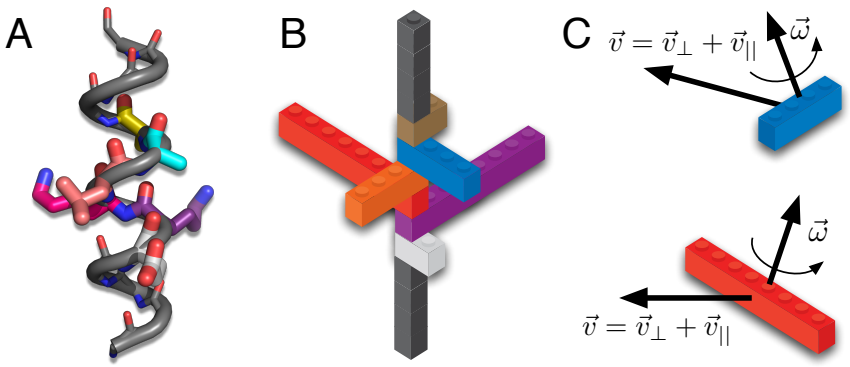


Figure 1.1: (A) All-atom representation of a protein. (B) RTB representation of the same protein. (C) Each rigid block has six degrees of freedom.

The main idea of the RTB projection method is to approximate the system as n rigid blocks. Figure 1.1.A-B schematically shows the all-atom and the RTB representations of the same system. The transition from the RTB coordinate system with $6n$ DOFs to the all-atom coordinate system with $3N$ DOFs is performed by an orthogonal projection matrix $P \in \mathbb{R}^{3N \times 6n}$. The conservation laws of the linear momentum and the angular momentum of a rigid block consisting of N_b atoms written in mass-weighted coordinates are

$$\begin{aligned} \sqrt{M_b}\dot{\tilde{q}} &= \sum_{k=1}^{N_b} \sqrt{m_k}\dot{q}_k \quad \text{for a translation} \\ \mathrm{I}^{1/2}\dot{\tilde{q}} &= \sum_{k=1}^{N_b} \sqrt{m_k}(q_k \times \dot{q}_k) \quad \text{for a rotation} \end{aligned}, \tag{1.4}$$

where M_b is the total mass of the rigid block, I is the rigid block's inertia tensor, $\tilde{q}$ is the blocks's displacement, m_k is the mass of the k^{th} atom of the block, and q_k is the displacement of the k^{th} atom of the block. The elements constituting P^T, the matrix projecting an all-atom motion q into a motion of rigid block $\tilde{q}$ are then obtained by differentiating (1.4) with respect to $\dot{q}_k$ [3]. This leads to translation $\mathrm{P_t}$ and rotation $\mathrm{P_r}$ matrices of size $3N_b \times 3$ each, computed for each of the rigid blocks and written through their k 3×3 square components,

$$\begin{aligned} \mathrm{P}_{tk} &= \sqrt{\frac{m_k}{M_b}}\mathrm{E}_3 && \text{for a translation} \\ \mathrm{P}_{rk} &= -\sqrt{m_k}(\mathrm{I})^{-1/2}[r_k - r^{COM}]_\times && \text{for a rotation} \end{aligned}, \tag{1.5}$$

where k is one of N_b atom indices, r_k is the position of the corresponding atom in the block, and r^{COM} the position of the block's center of mass (COM). The rigid block's displacement (δ, θ)

6-vector is then obtained by summing up the displacements in the RTB coordinate frame,

$$\begin{aligned} \delta &= \sum_{k=1}^{N_b} \mathrm{P}_{\mathrm{t}k}^{T}\, q_k \qquad \text{for a translation} \\ \theta &= \sum_{k=1}^{N_b} \mathrm{P}_{\mathrm{r}k}^{T}\, q_k \qquad \text{for a rotation} \end{aligned} \quad . \tag{1.6}$$

Having written these equations, we can write the projection matrix P as a diagonal block matrix,

$$P = \begin{pmatrix} \mathrm{P_t}^1 & \mathrm{P_r}^1 & & \\ \ddots & \ddots & \ddots & \\ & & \mathrm{P_t}^n & \mathrm{P_r}^n \end{pmatrix}. \tag{1.7}$$

The normal modes are then computed by the diagonalization of the RTB-projected mass-weighted stiffness matrix,

$$P^T K_w P = \tilde{L}\tilde{\Lambda}\tilde{L}^T, \tag{1.8}$$

where $\tilde{L}$ is the matrix composed of the RTB normal modes with the corresponding diagonal eigenvalue matrix $\tilde{\Lambda}$. This equation can be rewritten as

$$K_w = (P\tilde{L})\tilde{\Lambda}(P\tilde{L})^T. \tag{1.9}$$

The all-atom normal modes L^w (in mass-weighted coordinates) are then obtained as a projection of RTB normal modes $\tilde{L}$ according to

$$L^w = P\tilde{L}. \tag{1.10}$$

1.4 The NOLB method

Molecular vibrations in a multi-dimensional harmonic oscillator are all uncoupled and can be found by solving (1.3). Diagonalization of of the RTB-projected mass-weighted stiffness matrix gives a set of eigenvectors that are composed of *instantaneous linear velocities* and *instantaneous angular velocities* of individual rigid blocks $\{\vec{v}_w, \vec{\omega}_w\}$, see Fig. 1.1.C. For a rigid block with mass M_b and inertia tensor I, we first compute these in the non-mass weighted coordinates as follows,

$$\begin{aligned} \vec{v} &= M_b^{-1/2}\vec{v}_w \\ \vec{\omega} &= \mathrm{I}^{-1/2}\,\vec{\omega}_w \end{aligned} \quad . \tag{1.11}$$

Then, given a deformation amplitude a, the translational increment in the rigid block's position $\Delta\vec{x}$ and the angular increment in its orientation $\Delta\phi$ can be computed as

$$\begin{aligned} \Delta\vec{x} &= a\vec{v} \\ \vec{n} &= \vec{\omega}/||\vec{\omega}||_2, \\ \Delta\phi &= a||\vec{\omega}||_2 \end{aligned} \tag{1.12}$$

where the rigid block's rotation is described with a unit axis $\vec{n}$ passing though its COM and an angle ϕ. Finally, we rewrite the the increment in the rigid block's position $\Delta\vec{x}$ as a sum of two orthogonal vectors,

$$\Delta\vec{x} = \Delta\vec{x}_\perp + \Delta\vec{x}_{||}, \tag{1.13}$$

where $\Delta\vec{x}_\perp$ is orthogonal to $\vec{n}$ and $\Delta\vec{x}_{||}$ is collinear to $\vec{n}$, and represent the $\Delta\vec{x}_\perp$ related motion as a pure rotation about a new center $\vec{r}$, such that the final rigid blocks's position is

$$\vec{A}' = R(\Delta\phi, \vec{n})(\vec{A} - \vec{r}_0) + \vec{r}_0 + \Delta\vec{x}_{||}, \tag{1.14}$$

where $R(\Delta\phi, \vec{n})$ is the rotation matrix describing rigid block's rotation about an axis $\vec{n}$ by an angle $\Delta\phi$. See more details in [4].

1.5 Potential function

In principle, our approach can be used with any potential function. Currently, to omit the need of initial system's energy minimization, we have only implimented an all-atom Anisotropic Network Model (ANM) [5, 6], which is a type of elastic network model where the initial structure is always at equilibrium. Figure 1.2 shows a schematic representation of this model. The all-atom ANM has the following potential function,

$$V(q) = \sum_{d_{ij}^0 < R_c} \frac{\gamma}{2}(d_{ij} - d_{ij}^0)^2, \tag{1.15}$$

where d_{ij} is the distance between the i^{th} and the j^{th} atoms, d_{ij}^0 is the reference distance between these atoms, as found in the original structure, γ is the stiffness constant, and R_c is a cutoff distance, typically between 5 Å and 15 Å. The stiffness matrix corresponding to this potential function is composed of the following blocks [2, 5, 6],

$$\begin{aligned} H_{ij} &= -\frac{\gamma}{(d_{ij}^0)^2}\begin{pmatrix} (x_{ij}^0)^2 & x_{ij}^0 y_{ij}^0 & x_{ij}^0 z_{ij}^0 \\ y_{ij}^0 x_{ij}^0 & (y_{ij}^0)^2 & y_{ij}^0 z_{ij}^0 \\ z_{ij}^0 x_{ij}^0 & z_{ij}^0 y_{ij}^0 & (z_{ij}^0)^2 \end{pmatrix} \quad i \neq j \\ H_{ii} &= -\sum_{j \neq i} H_{ij} \end{aligned}, \tag{1.16}$$

where $x_{ij} = x_i - x_j$, $y_{ij} = y_i - y_j$, and $z_{ij} = z_i - z_j$. To rapidly compute this matrix, we use an efficient neighbor search algorithm [7].

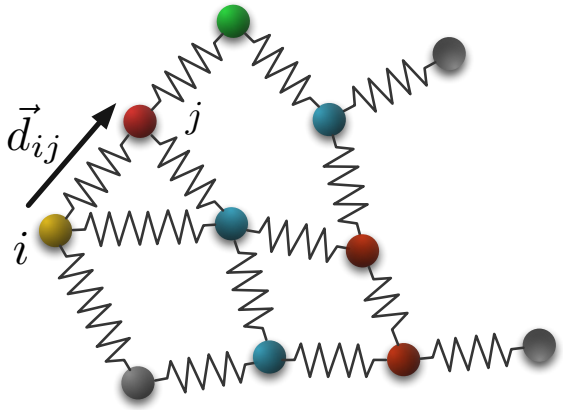
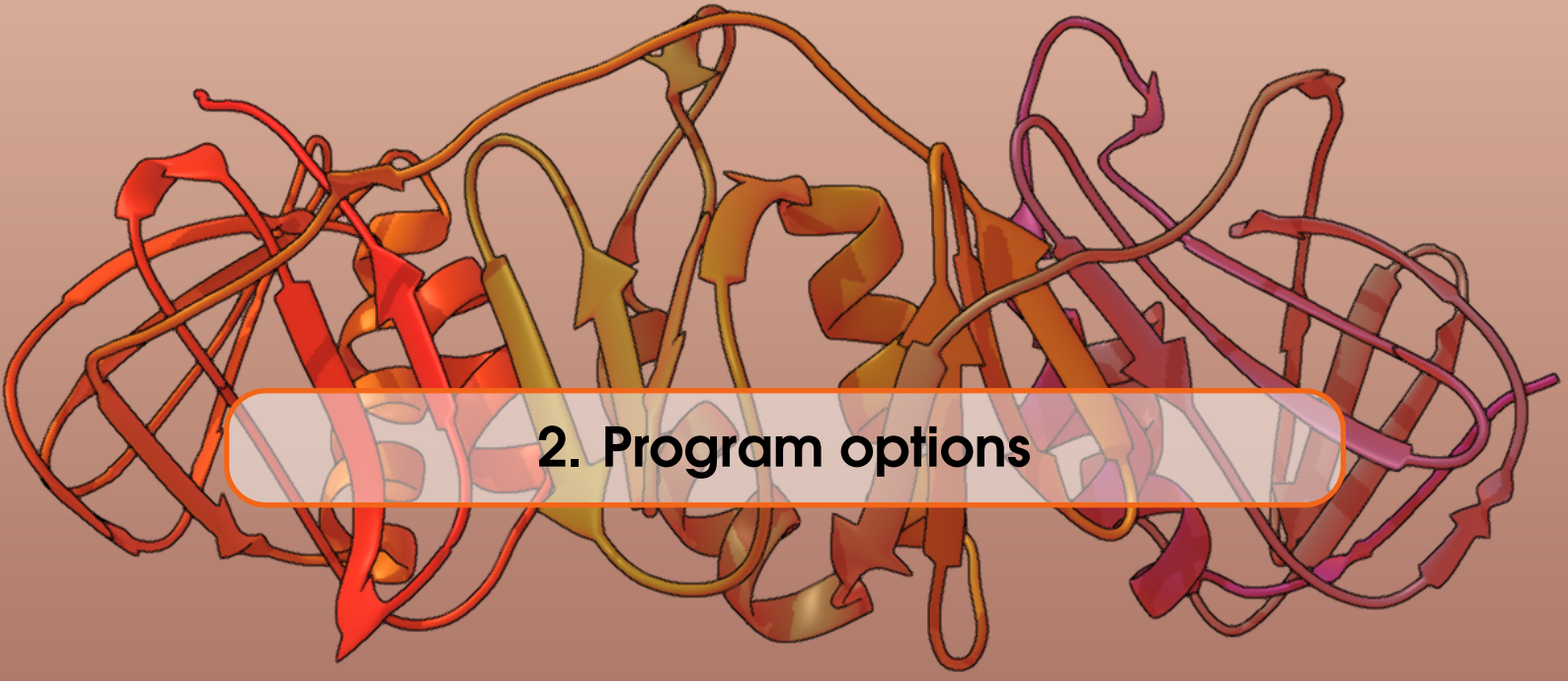


Figure 1.2: Schematic representation of the elastic network model.



2. Program options

2.1 Usage

Typing the '--help' or '-h' flag produces the brief and more detailed description of the program options,

```
NOLB --help
```

```
**********************************************************************
*----------------------------------------------------------------------*
*-----------NOLB : a Non-Linear Rigid Block NMA method-----------*
*----------Authors: Alexandre Hoffmann & Sergei Grudinin----------*
*-Copyright (c): Nano-D team, Inria/CNRS Grenoble, France, 2018.--*
*---------------- e-mail: sergei.grudinin@inria.fr ----------------*
*---- http://team.inria.fr/nano-d/software/nolb-normal-modes/ ----*
*----------------------------------------------------------------------*
**********************************************************************

USAGE:

   NOLB  <pdb filename> <pdb reference file> [-o <output name>] [-n <number
         of modes>] [-c <cutoff distance>] [-s <number of output frames>]
         [-a <maximum amplitude>] [-p <potential function>] [-m] [--linear]
         [--trajectory] [--hetatm] [--zdock <Zdock transform filename>]
         [--hex <Hex transform filename>] [--frames <max number of input
         trajectory frames>] [--covScaling <scaling of covalent
         interactions>] [-dcd <dcd file name>] [--clust] [--coords <input
         traj file>] [--weights <input weights file>] [-r <maximum rmsd>]
         [--dist <RMSD distribution>] [--nSteps <number of minimization
         steps>] [-t <minimization tolerance>] [--blocks <Rigid blocks ids
         filename>] [-h] [--version] [--log]


Where:

   <pdb filename>
     (required)  Input PDB file, can be a PDB multi-model trajectory file

   <pdb reference file>
     Reference PDB file. Only to compute the least RMSD motion. Disabled by
     default.

   -o <output name>,  --output <output name>
     Output file name. If nothing provided, a default name will be chosen.

   -n <number of modes>,  --modes <number of modes>
     Number of modes, 10 by default
```

```
-c <cutoff distance>,  --cutoff <cutoff distance>
  Interactions cutoff distance in angstroms, 5 by default

-s <number of output frames>,  --solutions <number of output frames>
  Number of frames in the output NMA trajectory, 10 by default

-a <maximum amplitude>,  --amplitude <maximum amplitude>
  Maximum amplitude, 1.0 by default. The absolute amplitude value is
  chosen automatically according to the system size. This option only
  adjusts the absolute value.

-p <potential function>,  --potential <potential function>
  Potential function, 0 = Elastic Network, 1 = Gaussian Network.
  Currently disabled.

-m,  --minimize
  Minimize obtained structures, off by default.

--linear
  Additionally compute the linear modes. Off by default.

--trajectory
  Save the output trajectory at equidistant points. Off by default.

--hetatm
  Read heteroatoms from the input file. False by default.

--zdock <Zdock transform filename>
  Zdock transforms file name. Experimental option.

--hex <Hex transform filename>
  Hex transforms file name. Experimental option.

--frames <max number of input trajectory frames>
  Max number of trajectory frames to use for trajectory analysis, 100 by
  default.

--covScaling <scaling of covalent interactions>
  Scaling of covalent interactions, 1 by default.

-dcd <dcd file name>,  --dcd <dcd file name>
  DCD file name. Requires corresponding PDB file.

--clust
  Cluster trajectory frames, off by default.

--coords <input traj file>
  Name of the text trajectory coordinate file that should correspond to
  input pdb. Experimental option.

--weights <input weights file>
  Name of the trajectory weights text file, where weights for different
  frames are provided. Experimental option.

-r <maximum rmsd>,  --rmsd <maximum rmsd>
  Maximum rmsd for decoy generation. In case of MD clustering - RMSD
  threshold.

--dist <RMSD distribution>
  Power dependence of RMSD distribution of generated decoys on the input
  RMSD, 0 for constant RMSD value (default), 1 for linear dependence,
  0.5 for quadratic dependence, etc.

--nSteps <number of minimization steps>
  Number of minimization steps, 500 by default.

-t <minimization tolerance>,  --tol <minimization tolerance>
  Minimization tolerance, 0.1 by default
```

```
--blocks <Rigid blocks ids filename>
  Rigid blocks ids file name.

-h,  --help
  Displays usage information and exits.

--version
  Displays version information and exits.

--log
  Displays ChangeLog information and exits.
```

Typing '--version' prints the current version of the program,

```
NOLB --version
```

```
NOLB  version: 0.8, February 2018
```

Typing '--log' prints the changelog of the program,

```
NOLB --log
```

```
NOLB  ChangeLog:
Version 0.1 from Feb 2016:
        Initial release.
Version 0.2 from March 2016:
        Added initial support of PDB trajectories.
Version 0.3 from Feb 2017:
        Added reading of hetero atoms. Adapted for SAMSON GUI. Multiple tests performed.
        Compiled on Linux, Win32 and MacOS.
Version 0.4 from May 2017:
        Added automatic tests for convergence. More of noninear PCA support.
Version 0.5 from June 2017:
        Added output of linear modes.
Version 0.6 from July 2017:
        Added generation of NMA decoys.
Version 0.7 from December 2017:
        Added possibility to manually specify rigid blocks.
Version 0.8 from February 2018:
        Added support for DCD trajectories. Initial support of MD rapid clustering. Fixed support
        of insertion codes when aligning the sequences.
```

2.2 Main options

```
<pdb filename> (required)
```

There is only one required argument, the path to the input PDB file. This can alternatively be a PDB multi-model trajectory file. In this case, the NMA will be performed based on the covariance matrix of the trajectory. A clustering of the trajectory can be additionally performed using the corresponding flag.

```
<pdb reference file>
```

This options provides the path to the reference PDB file. It is only used to compute the least-RMSD transition path. Disabled by default. The molecule may be different from the initial molecule. To make the calculations robust, first a sequence alignment is performed.

```
-o <output name>, --output <output name>
```

This options provides the output file name. If nothing provided, a default name based on the name of the input file will be chosen.

```
-n <number of modes>, --modes <number of modes>
```

This options provides the number of non-trivial modes to compute, 10 by default. If this number exceeds the size of the Hessian matrix, it will be adapted accordingly.

```
-c <cutoff distance>, --cutoff <cutoff distance>
```

This options specifies the interaction cutoff distance for the elastic network models (in angstroms), 5 by default. The Hessian matrix is constructed according to this interaction distance. Some artifacts should be expected for too short distances (< 5 Å).

```
-s <number of output frames>, --solutions <number of output frames>
```

This options provides the number of frames in the output NMA trajectory, 10 by default. The output trajectory is written in the PDB file with multiple models. This flag specifies the number of models in the output file. It also controls the number of models if pseudo-random decoys are generated.

```
-a <maximum amplitude>, --amplitude <maximum amplitude>
```

This options sets the maximum relative amplitude of NMA motion, 1.0 by default. The absolute amplitude value is chosen automatically according to the system size. This flag only adjusts the absolute value.

```
--linear
```

Flag to additionally compute the linear modes. Off by default. If computed, these are stored separately.

```
--hetatm
```

Flag to read heteroatoms from the input file. False by default. Should be used for lipids, small molecules, etc.

```
--trajectory
```

Flag to save the output trajectory at equidistant points. Off by default. If this option is disabled, then the trajectory is written according to the natural harmonic motion of the oscillator.

```
--frames <max number of input trajectory frames>
```

This is the option to choose the maximum number of trajectory frames to use for trajectory analysis of docking results, 100 by default. Only takes effect when analyzing the output of docking programs.

```
-r <maximum rmsd>, --rmsd <maximum rmsd>
```

If pseudo-random decoys are needed, this option specifies the maximum RMSD of the generated

decoys. If clustering of MD trajectories is required, this option will specify the clustering RMSD threshold.

```
--dist <RMSD distribution>
```

This option defines the distribution of the decoys RMSD, 0 for constant RMSD (default), 0.5 for quadratic dependence, 1 for linear dependence, etc.

```
-m, --minimize
```

Flag to minimize the obtained structures, off by default.

```
--nSteps <number of minimization steps>
```

This option sets the number of minimization steps, 500 by default.

```
-t <minimization tolerance>, --tol <minimization tolerance>
```

This option sets the minimization tolerance, 0.1 by default.

```
--covScaling <scaling of covalent interactions>
```

This flag specifies the parameter by which the covalent interactions are scaled during the computation of the Hessian matrix. It does not significantly change the results and set to 1.0 by default.

```
-dcd <dcd file name>, --dcd <dcd file name>
```

Name of the trajectory file in the DCD format. If it is specified, the first PDB argument must correspond to the topology of the DCD file.

```
--clust
```

This flag controls the clustering of the input trajectories. The user also needs to define the corresponding RMSD clustering threshold. Off by default.

```
-h, --help
```

Flag to display usage information and exit.

```
--version
```

Flag to display version information and exit.

```
--log
```

Flag to display the ChangeLog information and exit.

2.3 Experimental options

```
--zdock <Zdock transform filename>
```

This flag provides the path to the Zdock transforms file. Zdock protein docking poses are analyzed then using the covariance matrix.

```
--hex <Hex transform filename>
```

This flag provides the path to the Hex transforms file. Hex protein docking poses are analyzed then using the covariance matrix.

```
--coords <input traj file>
```

Name of the trajectory coordinate file that should correspond to the input pdb file. This one is used for the analysis of the covariance matrix.

```
--weights <input weights file>
```

Name of the trajectory weights text file, where weights for different frames are provided.

2.4 Currently suppressed options

```
-p <potential function>, --potential <potential function>
```

Potential function, 0 = Elastic Network, 1 = Gaussian Network. Currently, the ANM model is only used.

3. Examples

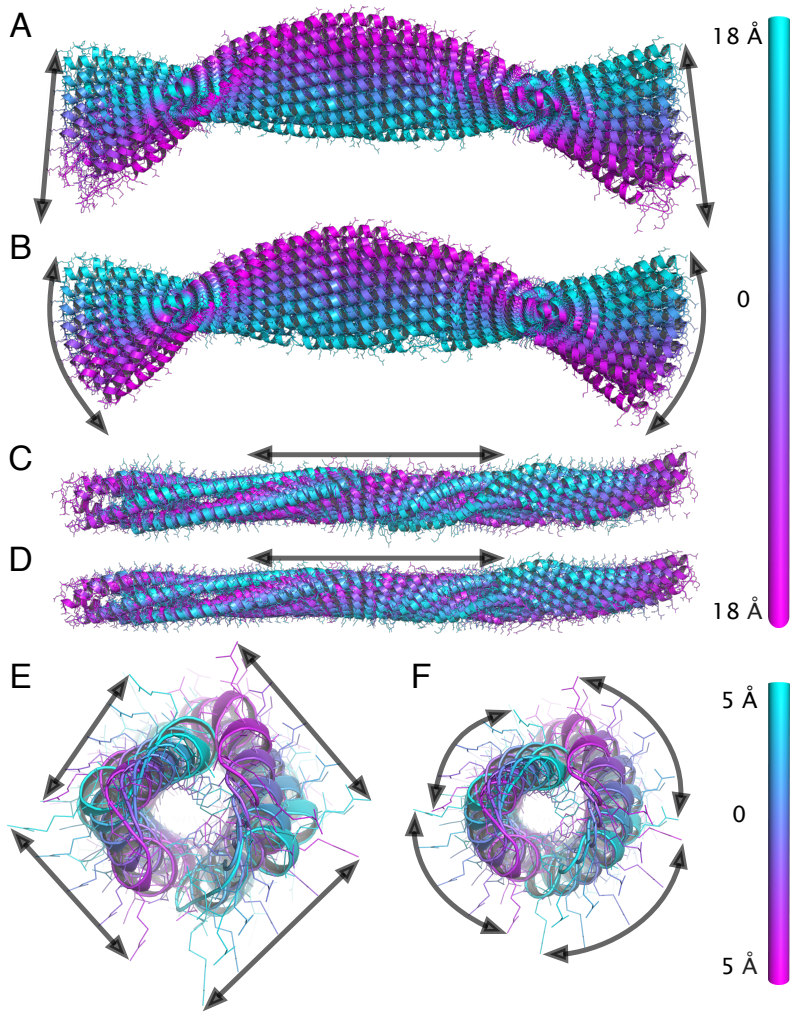


Figure 3.1: Comparison of linear (A, C, E) and non-linear (B, D, F) normal modes computed for a coiled coil protein (pdb code 2ch7). Three types of motions are shown, bending (A, B), stretching (C, D), and twisting (E, F). Several snapshots at different deformation amplitudes are superposed to each other. These are colored according to the values of the overall deformation, as measured by the RMSD. The colorbars show the RMSD with respect to the initial position. The arrows follow the trajectories of individual atoms.

3.1 Analysis of basic molecular motions

We will start with the NOLB NMA of a mid-size coiled-coil protein from the cytoplasmic domain of a bacterial chemoreceptor (pdb code 2ch7), which clearly demonstrates the three basic types of molecular vibration motions. These are bending, stretching and twisting.

Example 1 To compute the first 10 non-trivial normal modes for a 2ch7.pdb protein structure, type

```
NOLB 2ch7.pdb
```

This will produce the following output in the terminal (we suppose you are starting the program from the terminal!),

```
*********************************************************************
*-------------------------------------------------------------------*
*-----------NOLB : a Non-Linear Rigid Block NMA method------------*
*----------Authors: Alexandre Hoffmann & Sergei Grudinin----------*
*-Copyright (c): Nano-D team, Inria/CNRS Grenoble, France, 2017.--*
*--------------- e-mail: sergei.grudinin@inria.fr ----------------*
*---- http://team.inria.fr/nano-d/software/nolb-normal-modes/ ----*
*-------------------------------------------------------------------*
*********************************************************************
=====================Reading input PDB file=======================
Input PDB file..................................................... : 2ch7.pdb
Number of chains read.............................................. : 3
Number of atoms read............................................... : 4630
Found.............................................................. : 4641 covalent bonds
.................................................................. : 6264 angles
=====================Constructing Hessian=========================
Number of disconnected atoms....................................... : 0
Number of interactions............................................. : 631042
Memory required for the Hessian matrix............................. : 129.991 Mb
============================Doing NMA============================
Memory required for the RTB projection matrix...................... : 1.27167 Mb
Reduced Hessian size : ............................................ : 3702 x 3702
Number of computed modes........................................... : 10
Null-space size.................................................... : 1
Lowest mode frequency.............................................. : 0.000535409
Highest mode frequency............................................. : 0.00412211
Frequencies :
            [ 1]................................................... : 0.000535409
            [ 2]................................................... : 0.000542863
            [ 3]................................................... : 0.00135327
            [ 4]................................................... : 0.00153196
            [ 5]................................................... : 0.00167521
            [ 6]................................................... : 0.00261288
            [ 7]................................................... : 0.00278308
            [ 8]................................................... : 0.00347724
            [ 9]................................................... : 0.003539
            [10]................................................... : 0.00412211
====================Writing output PDB files=====================
Using output mask.................................................. : 2ch7
Using output mask for nonlinear modes.............................. : 2ch7_nlb_
====================================================================
============================ Timing : ==============================
====================================================================
Parsing arguments.................................................. : 0.001036 s
Reading input PDB file............................................. : 0.015923 s
Constructing Hessian............................................... : 0.549772 s
Doing NMA.......................................................... : 0.797914 s
Writing output PDB files........................................... : 0.775731 s
....................................................................
Total time : ...................................................... : 2.14038 s
====================================================================
```

The first section lists the structural parameters, the number of atoms, chains, and the number of

bonds and angles in the structure. These last two can be used during energy relaxation, which is currently disabled. Water molecules are ignored and not listed, heteroatoms can be read with an additional flag '--hetatm':

```
====================Reading input PDB file=====================
Input PDB file.............................................. : 2ch7.pdb
Number of chains read....................................... : 3
Number of atoms read........................................ : 4630
Found....................................................... : 4641 covalent bonds
............................................................ : 6264 angles
```

The second section lists parameters of the Hessian matrix. These are the number of interacting atom pairs, which can be controlled with the '--cutoff' flag, the number of disconnected atoms, which are stayed static, and the total memory required for the Hessian:

```
======================Constructing Hessian======================
Number of disconnected atoms................................ : 0
Number of interactions...................................... : 631042
Memory required for the Hessian matrix...................... : 129.991 Mb
```

The next section lists parameters of the RTB basis and the diagonalization procedure. It provides the size of the projection matrix, the size of the RTB Hessian, the size of the detected null-space, the number of modes and the corresponding frequencies. These are the square roots of the corresponding eigenvalues. The null-space is usually not detected fully, it should be 5 or 6 for non-symmetric systems, and skipped from the further analysis:

```
============================Doing NMA============================
Memory required for the RTB projection matrix.................. : 1.27167 Mb
Reduced Hessian size : ..................................... : 3702 x 3702
Number of computed modes.................................... : 10
Null-space size............................................. : 1
Lowest mode frequency....................................... : 0.000535409
Highest mode frequency...................................... : 0.00412211
Frequencies :
            [ 1]............................................ : 0.000535409
            [ 2]............................................ : 0.000542863
            [ 3]............................................ : 0.00135327
            [ 4]............................................ : 0.00153196
            [ 5]............................................ : 0.00167521
            [ 6]............................................ : 0.00261288
            [ 7]............................................ : 0.00278308
            [ 8]............................................ : 0.00347724
            [ 9]............................................ : 0.003539
            [10]............................................ : 0.00412211
```

The next section lists the masks of the output files:

```
====================Writing output PDB files====================
Using output mask........................................... : 2ch7
Using output mask for nonlinear modes....................... : 2ch7_nlb_
```

The output pdb trajectory files have the following names, '2ch7_nlb_1.pdb','2ch7_nlb_2.pdb',etc. The number in the name refers to the normal mode id starting from 1. The output mask can be changed with the '-o newMask' flag. Finally, the last section lists the timings of the program, split into individual contributions:

```
============================ Timing : ============================
================================================================
Parsing arguments........................................... : 0.001036 s
Reading input PDB file...................................... : 0.015923 s
Constructing Hessian........................................ : 0.549772 s
Doing NMA................................................... : 0.797914 s
Writing output PDB files.................................... : 0.775731 s
```

```
.............................................................
Total time : ................................................ : 2.14038 s
=============================================================
```

Here, the most time consuming operation is typically the diagonalization step. Writing trajectories can be very long as well, depending on the number of required modes and frames in the trajectories. Figure 3.1 shows some of the computed modes.

3.2 Comparison with linear NMA

Example 2 In this example we will compare the NOLB normal modes with the linear normal modes. For this, we will enable the '--linear' flag, and will also output more trajectory frames for a better visual perception of the motions using the '-s 20' flag. Finally, we will increase the motion amplitude by a factor of 2 with the '-a 2' flag:

```
NOLB 2ch7.pdb -s 20 -a 2 --linear
```

The output will be only different when listing the output files,

```
====================Writing output PDB files=====================
Using output mask.............................................. : 2ch7
Using output mask for linear modes............................. : 2ch7_linear_
Using output mask for nonlinear modes.......................... : 2ch7_nlb_
```

Now, we have also computed the linear NMA trajectories stored in '2ch7_linear_1.pdb', '2ch7_linear_2.pdb', etc. Figure 3.1 shows the visual difference between the computed linear and non-linear modes.

3.3 Another example with a bigger system that contains heteroatoms

Example 3 In this example we compute the first 50 normal modes for a larger system, the photosystem II complex (pdb code 5b5e) with about 50,000 of atoms,

```
NOLB 5b5e.pdb -n 50
```

This will produce the following output, which we will cut for brevity,

```
====================Reading input PDB file=====================
Input PDB file................................................. : 5b5e.pdb
Number of chains read.......................................... : 39
Number of atoms read........................................... : 40908
Found.......................................................... : 42093 covalent bonds
.............................................................. : 57363 angles
======================Constructing Hessian=====================
Number of disconnected atoms................................... : 0
Number of interactions......................................... : 6424422
Memory required for the Hessian matrix......................... : 1323.39 Mb
============================Doing NMA============================
Lowest mode frequency.......................................... : 0.00156711
Highest mode frequency......................................... : 0.00784686
Frequencies :
            [ 1]............................................... : 0.00156711
            [ 2]............................................... : 0.00175741
========================== Timing : ==========================
=============================================================
Parsing arguments.............................................. : 0.000678 s
Reading input PDB file......................................... : 0.099499 s
Constructing Hessian........................................... : 7.44606 s
Doing NMA...................................................... : 29.754 s
Writing output PDB files....................................... : 28.5004 s
.............................................................
```

```
Total time : ..................................................... : 65.8006 s
================================================================
```

Please note that the PDB output now takes a significant part of the total execution time. This system contains about 10,000 of small molecules, mostly lipids. In order to include these into the analysis, we repeat the computations with the '--hetatm' flag,

Example 4 This example is identical to the previous one, with all the heteroatoms (excluding water) included into the NMA calculations,

```
NOLB 5b5e.pdb -n 50 --hetatm,
```

This will produce the following output,

```
====================Reading input PDB file=====================
Input PDB file..................................................... : 5b5e.pdb
Number of chains read.............................................. : 39
Number of atoms read............................................... : 51391
Found.............................................................. : 52758 covalent bonds
.................................................................. : 71368 angles
======================Constructing Hessian=====================
Number of disconnected atoms....................................... : 0
Number of interactions............................................. : 9358367
Memory required for the Hessian matrix............................. : 1927.76 Mb
============================Doing NMA============================
Lowest mode frequency.............................................. : 0.00156711
Highest mode frequency............................................. : 0.00784686
Frequencies :
            [ 1]................................................... : 0.00156711
            [ 2]................................................... : 0.00175741
```

Please note the changed Hessian size and the spectrum of the normal modes. The visual inspection of the first modes demonstrates that these are still the same. However, their order might have changed because of the changed spectrum.

3.4 Finding the best transition between two protein conformations

In this example we are going to find the best direction in the NMA subspace that minimizes the RMSD between two molecular structures. We should note that the structures are first aligned to each other based on their sequence. Thus, the method only currently works with proteins.

Example 5 In this example we will compute the best non-linear trajectory and the corresponding RMSD in the reduced NOLB subspace between two protein conformations. For the goal (bound) conformation we choose the structure of the FAB 28 complex (pdb code 2hmi). For the start conformation we choose the unbound structure of the HIV1 reverse transcriptase (pdb code 1s6p). Then, we will compute the best transition using the lowest 4 modes as follows,

```
NOLB 1s6p.pdb 2hmi.pdb -n 4 -s 20
```

The NOLB method will first align the two conformations,

```
====================Reading input PDB file=====================
Input PDB file..................................................... : 1s6p.pdb
Number of chains read.............................................. : 3
Number of atoms read............................................... : 8027
Found.............................................................. : 8250 covalent bonds
.................................................................. : 11211 angles
====================Reading input PDB file=====================
Input PDB file..................................................... : 2hmi.pdb
Number of chains read.............................................. : 6
```

```
Number of atoms read..................................... : 11647
Found.................................................... : 16500 covalent bonds
........................................................ : 22422 angles
====================Aligning input PDB file=====================
Total number of gaps .................................... : 480
Alignement............................................... :
-----------------------------------PISPIETVPVKLKPGMDGPKVKQ
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxPISPIETVPVKLKPGMDGPKVKQ
                                      **********************
WY-------------------------------------------------------
WYQLEDIQMTQTTSSLSASLGDRVTISCSASQDISSYLNWYQQKPEGTVKLLIYYTSSLH
**
```

Then, it outputs the trajectory from the starting conformation to the goal using the optimal combination of the first 4 modes. It also prints the reduction in the overall RMSD for each mode,

```
====================Writing output PDB files====================
Using output mask........................................ : 1s6p
Using output mask for nonlinear modes.................... : 1s6p_nlb_
RMSD reduction : ........................................ :
1s6p.pdb        3.621983    3.169015    2.894955    2.235346    2.196610
Initial RMSD............................................. : 3.62198
Final RMSD............................................... : 2.19661
```

Finally, it outputs the timings. We can see that overall it only takes about two seconds,

```
=========================== Timing : ============================
====================================================================
Parsing arguments........................................ : 0.000248 s
Reading input PDB file................................... : 0.015575 s
Reading input PDB file................................... : 0.015943 s
Aligning input PDB file.................................. : 0.031428 s
Superposeing input PDB file.............................. : 0.000262 s
Constructing Hessian..................................... : 0.948881 s
Doing NMA................................................ : 1.16697 s
Writing output PDB files................................. : 0.221236 s
........................................................
Total time : ............................................ : 2.40055 s
====================================================================
```

The obtained result can be visualized with popular software packages, for example, as

```
pymol 1s6p_nlb.pdb 2hmi.pdb
```

3.5 Analysis of an MD trajectory or NMR ensemble

In the next example, we will compute the principal components of an MD or NMR trajectory. This will be done by the diagonalization of the covariance matrix. We will use a solution NMR structure of a DNA (pdb code 5izp), which contains 15 models.

Example 6 To compute the first 10 non-trivial normal modes for an ensemble NMR structure, type

```
NOLB 5izp.pdb -s 20 --linear
```

This produces the following output, cut for brevity,

```
====================Reading input PDB file======================
Input PDB file........................................... : 5izp.pdb
This is a trajectory file................................ :
Number of chains read.................................... : 2
Number of atoms read..................................... : 442
=====================Reading Trajectory=======================
Number of frames read.................................... : 15
```

```
==================Aligning Trajectory Frames====================
==============Constructing the Covariance matrix================
Memory Required for the Covariance Matrix..................... : 13.4146 Mb
Memory required for the RTB projection matrix.................. : 0.121399 Mb
============================ Timing : ============================
================================================================
Parsing arguments.............................................. : 0.000613 s
Reading input PDB file......................................... : 0.003657 s
Reading Trajectory............................................. : 0.005014 s
Aligning Trajectory Frames..................................... : 0.000192 s
Constructing the Covariance matrix............................. : 0.00419 s
Doing NMA...................................................... : 0.000838 s
Writing output PDB files....................................... : 0.112159 s
................................................................
Total time : .................................................. : 0.126667 s
================================================================
```

You may see that the trajectory analysis is very fast, the only noticeable time here is actually the output of the computed motions.

Example 7 In the following example, we will analyze a MD trajectory of a coiled-coil domain of a chemoreceptor saved in DCD format. The length of the simulations was about 1 us, and the frames were saved every 1 ns. To do so, we type

```
NOLB traj.pdb --dcd traj.dcd
```

This produces the following output, cut for brevity,

```
=====================Reading input PDB file=====================
Input PDB file................................................. : traj.pdb
Number of chains read.......................................... : 2
Number of atoms read........................................... : 3546
=====================Reading DCD Trajectory=====================
Trajectory File Name........................................... : traj.dcd
FRAMES......................................................... : 1040
NATOM.......................................................... : 3546
LNFREAT........................................................ : 3546
Number of frames read.......................................... : 1040
==================Aligning Trajectory Frames====================
==============Constructing the Covariance matrix================
Memory Required for the Covariance Matrix..................... : 863.396 Mb
Memory required for the RTB projection matrix.................. : 0.973938 Mb
============================Doing NMA===========================
Number of computed modes....................................... : 10
Lowest mode frequency.......................................... : 5.44193e-05
Highest mode frequency......................................... : 0.00422035
================================================================
============================ Timing : ============================
================================================================
Parsing arguments.............................................. : 0.000315 s
Reading input PDB file......................................... : 0.004046 s
Reading DCD Trajectory......................................... : 0.149615 s
Aligning Trajectory Frames..................................... : 0.094616 s
Constructing the Covariance matrix............................. : 1.47288 s
Doing NMA...................................................... : 0.086615 s
Writing output PDB files....................................... : 0.762107 s
................................................................
Total time : .................................................. : 2.5702 s
================================================================
```

Again, the trajectory analysis is very fast, and the principal components of a 1us simulations have been computed in just a few seconds.

3.6 Clustering of MD trajectory frames

In the next example, we will provide a fast clustering method based on the RapidRMSD library [8]. First, we will project the MD trajectory on the principal components, and then we will use a fast method to compute RMSDs between the flexible conformations, if these are obtained with a few collective motions. We will use a lysozyme MD trajectory (1960 atoms) with 10,000 frames.

Example 8 To cluster the frames of MD trajectory with RMSD threshold of 0.5 Å, type the following,

```
NOLB lyz.pdb -dcd lyz.dcd --clust --rmsd 0.5
```

This produces the following output, cut for brevity,

```
====================Reading input PDB file=====================
Input PDB file.................................................. : lyz.pdb
Number of chains read........................................... : 1
Number of atoms read............................................ : 1960
====================Reading DCD Trajectory=====================
Trajectory File Name............................................ : lyz.dcd
TITLE........................................................... : Created by DCD plugin
NATOM........................................................... : 1960
Number of frames read........................................... : 10000
============================Doing NMA============================
Number of computed modes........................................ : 10
Lowest mode frequency........................................... : 0.00203219
Highest mode frequency.......................................... : 0.0172908
Frequencies :
            [ 1]................................................ : 0.00203219
            [ 2]................................................ : 0.00281031
            [ 3]................................................ : 0.00587768
            [ 4]................................................ : 0.00721599
            [ 5]................................................ : 0.00790467
            [ 6]................................................ : 0.00920148
            [ 7]................................................ : 0.0119896
            [ 8]................................................ : 0.013423
            [ 9]................................................ : 0.0148008
            [10]................................................ : 0.0172908
====================Cluster the trajectory=====================
Fast : Created 492 cluster(s) out of 10000 poses.
Clusters........................................................ :
     0      1      2      3      4      5      6      7      8      9     10
  ...
  9982   9985
  9986   9987   9988   9990   9991   9998
================================================================
============================ Timing : ============================
================================================================
Parsing arguments............................................... : 0.00021 s
Reading input PDB file.......................................... : 0.002229 s
Reading DCD Trajectory.......................................... : 0.792337 s
Aligning Trajectory Frames...................................... : 0.530538 s
Constructing the Covariance matrix.............................. : 8.06337 s
Doing NMA....................................................... : 0.269174 s
Cluster the trajectory.......................................... : 0.560206 s
................................................................
Total time : ................................................... : 10.2181 s
```

You may see once again that the trajectory analysis including the subsequent clustering of a MD trajectory is very fast.

3.7 Analysis of protein docking poses

In our next example, we will compute principal components of protein-protein docking poses. We have chosen a standard example from the protein docking benchmark (pdb code 1d6r, with the receptor 1D6R_r_b.pdb and the ligand 1D6R_l_b.pdb). We have computed the first 1,000 of docking poses with the popular zDock package, Hex can be used as well here.

Example 9 To visualize the principal components of the computed docking poses type

```
NOLB outName.pdb  --zdock ./1D6R.zdock.out
```

This produces the following output,

```
====================Reading Zdock Transforms=====================
========================Reading PDB files=========================
Reading receptor PDB file..................................... : ./1D6R_r_b.pdb
Number of receptor chains read................................ : 1
Number of receptor atoms read................................. : 1629
Reading ligand PDB file....................................... : ./1D6R_l_b.pdb
Number of ligand chains read.................................. : 1
Number of ligand atoms read................................... : 427
====================Reading Docking Trajectory====================
Number of frames read......................................... : 100
============================Doing NMA=============================
Number of computed modes...................................... : 5
Lowest mode frequency......................................... : 2.39176e-05
Highest mode frequency........................................ : 0.00031764
Frequencies :
            [ 1]..................................................... : 2.39176e-05
            [ 2]..................................................... : 5.74957e-05
            [ 3]..................................................... : 8.65304e-05
            [ 4]..................................................... : 0.000198992
            [ 5]..................................................... : 0.00031764
====================Writing output PDB files====================
Using only.................................................... : 5 modes
Using output mask............................................. : outName
Using output mask for nonlinear modes......................... : outName_nlb
```

We should mention that this is just an example how to use the NOLB method for a simple PCA analysis, which probably cannot be used for practical applications.

3.8 Structural ensembles

Another application of the NOLB method is the generation of structural ensembles (also called 'decoys') with either a fixed deformation RMSD, or a certain distribution of these. Figure 3.2 shows a sample output of the method.

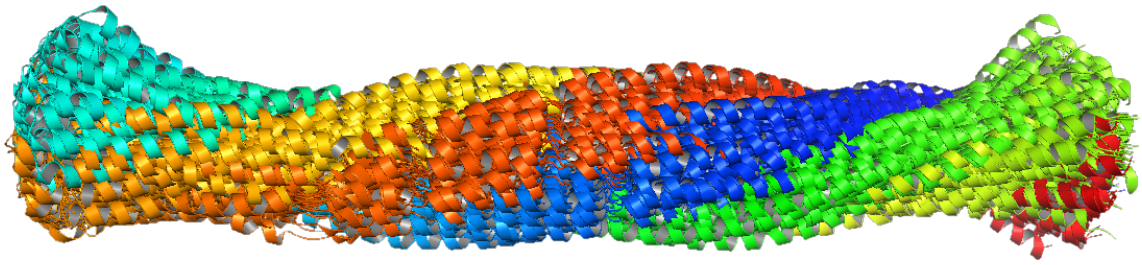


Figure 3.2: Illustration of 100 decoys of a coiled-coil system 2ch7 protein with RMSD of 10 Å from the starting structure.

Example 10 In the first example we will generate 100 decoys of a coiled-coil system at 10 Å from the starting structure.

```
NOLB 2ch7.pdb  -s 100 --rmsd 10
```

This creates a '2ch7_nlb_decoys.pdb' pdb trajectory file with 100 models with the following output,

```
==================Writing output Decoy files====================
Number of files.................................................. : 100
Output RMSD...................................................... : 10 A
Using outputfilename............................................. : 2ch7_nlb_decoys.pdb
================================================================
========================== Timing : ============================
================================================================
Parsing arguments................................................ : 0.000178 s
Reading input PDB file........................................... : 0.006173 s
Constructing Hessian............................................. : 0.0752 s
Doing NMA........................................................ : 0.605818 s
Writing output Decoy files....................................... : 0.848708 s
.................................................................
Total time : .................................................... : 1.53608 s
================================================================
```

Example 11 In the second example we will generate 100 decoys of a the same system with RMSD distributed linearly from 0 to 10 Å compared to the starting structure. We will also minimize the obtained decoys.

```
NOLB 2ch7.pdb  -s 100 --rmsd 10 --dist 1 -m
```

This creates a '2ch7_nlb_decoys.pdb' pdb trajectory file with 100 models with the following output,

```
==================Writing output Decoy files====================
Number of files.................................................. : 100
Output RMSD...................................................... : 10 A
Using outputfilename............................................. : 2ch7_nlb_decoys.pdb
Current energy................................................... : 0.272494
N minimization steps............................................. : 0
Current Gradient norm............................................ : 0.0210788
Displacement .................................................... : 0.00167775
RMSD after minimization.......................................... : 2.45655
---
Current energy................................................... : 6.27796
N minimization steps............................................. : 1
Current Gradient norm............................................ : 0.0738622
Displacement .................................................... : 0.0135281
RMSD after minimization.......................................... : 7.13781
================================================================
========================== Timing : ============================
================================================================
Parsing arguments................................................ : 0.000224 s
Reading input PDB file........................................... : 0.009928 s
Constructing Hessian............................................. : 0.069928 s
Doing NMA........................................................ : 0.603946 s
Writing output Decoy files....................................... : 2.1154 s
.................................................................
Total time : .................................................... : 2.79943 s
================================================================
```

3.9 Structural transitions

Yet another application of the NOLB method is the generation of linear and nonlinear structural transitions between different states of macromolecules (typically proteins). Figure 3.3 shows a summary of these computed for the proteins from the Protein Docking Benchmark v5 [9].

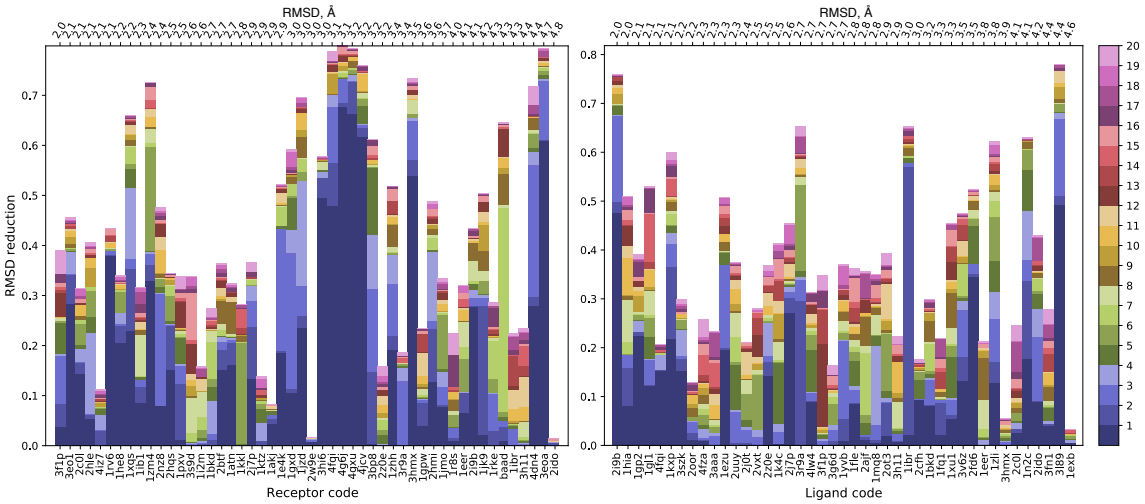


Figure 3.3: Transitions between the unbound (u) and bound (b) states of proteins from the Protein Docking Benchmark v5 [9]. The top x-axis shows C_α RMSD between the two states. The bottom x-axis lists the corresponding PDB codes of the complexes. The left plot shows the receptors, and the right plot shows the ligands, as labelled by the authors of the benchmark. Only structures with u-b RMSD ≥ 2 Å are shown. The y-axis shows the relative u-b transition that can be predicted using the optimal linear combination of some number of normal modes. Results for the range between 1 and 20 are shown in different colors (see the colorbar at the right).

Example 12 In the current example we will produce a transition between the unbound (u) and the bound (b) forms of the ligand from the 3l89 complex.

```
NOLB 3L89_l_u.pdb 3L89_l_b.pdb
```

This creates a '3L89_l_u_nlb.pdb' pdb trajectory file for the best transition between the states, and also the following output

```
======================Reading input PDB file=======================
Input PDB file..................................................... : 3L89_l_u.pdb
Number of chains read.............................................. : 1
Number of atoms read............................................... : 1018
======================Reading input PDB file=======================
Input PDB file..................................................... : 3L89_l_b.pdb
Number of chains read.............................................. : 1
Number of atoms read............................................... : 987
======================Aligning input PDB file======================
Total number of gaps .............................................. : 0
Alignement......................................................... :
CEEPPTFEAMELIGKPKPYYEIGERVDYKCKKGYFYIPPLATHTICDRNHTWLPVSDDAC
CEEPPTFEAMELIGKPKPYYEIGERVDYKCKKGYFYIPPLATHTICDRNHTWLPVSDDAC
************************************************************

YRETCPYIRDPLNGQAVPANGTYEFGYQMHFICNEGYYLIGEEILYCELKGSVAIWSGKP
YRETCPYIRDPLNGQAVPANGTYEFGYQMHFICNEGYYLIGEEILYCELKGSVAIWSGKP
************************************************************
```

```
PICEKV
PICEKV
******

==================Superposeing input PDB file==================
=====================Constructing Hessian=====================
Number of disconnected atoms..................................... : 0
Number of interactions........................................... : 22944
Interaction cutoff distance...................................... : 5 A
Memory required for the Hessian matrix........................... : 4.72632 Mb
All-atom Hessian size : ......................................... : 3054 x 3054
============================Doing NMA============================
Memory required for the RTB projection matrix.................... : 0.279602 Mb
Reduced Hessian size : .......................................... : 756 x 756
Number of computed modes......................................... : 10
Null-space size.................................................. : 1
Lowest mode frequency............................................ : 0.000266116
Highest mode frequency........................................... : 0.00198693
Frequencies :
            [ 1]................................................. : 0.000266116
            [ 2]................................................. : 0.000420865
            [ 3]................................................. : 0.000595849
            [ 4]................................................. : 0.00120617
            [ 5]................................................. : 0.00132804
            [ 6]................................................. : 0.00146345
            [ 7]................................................. : 0.00150491
            [ 8]................................................. : 0.00159594
            [ 9]................................................. : 0.00170802
            [10]................................................. : 0.00198693
====================Writing output PDB files====================
Using output mask................................................ : 3L89_l_u
Using output mask for nonlinear modes............................ : 3L89_l_u_nlb_
Number of aligned atoms.......................................... : 126
RMSD reduction : ................................................ :
    4.445432    2.258740    2.177358    1.470716    1.412096    1.411807    1.337768    1.337624
        1.325643    1.293560    1.182593
Initial RMSD..................................................... : 4.44543
Final RMSD....................................................... : 1.18259
Reduction in RMSD by............................................. : 0.733976
================================================================
========================== Timing : ==========================
================================================================
Parsing arguments................................................ : 0.00026 s
Reading input PDB file........................................... : 0.001994 s
Reading input PDB file........................................... : 0.002525 s
Aligning input PDB file.......................................... : 0.000866 s
Superposeing input PDB file...................................... : 4.9e-05 s
Constructing Hessian............................................. : 0.013787 s
Doing NMA........................................................ : 0.05935 s
Writing output PDB files......................................... : 0.017804 s
................................................................
Total time : .................................................... : 0.096657 s
================================================================
```

Please note, that here we have found a reduction of RMSD by 73% if the best combination of the first ten modes is used. This analysis can be performed on the whole benchmark, for example, with the following bash script,

```
for f in b5/*_r_u.pdb; do NOLB $f  ${f:0:${#f}-5}b.pdb -n 20 >> results-R-20modes.txt; done
for f in b5/*_l_u.pdb; do NOLB $f  ${f:0:${#f}-5}b.pdb -n 20 >> results-L-20modes.txt; done
```

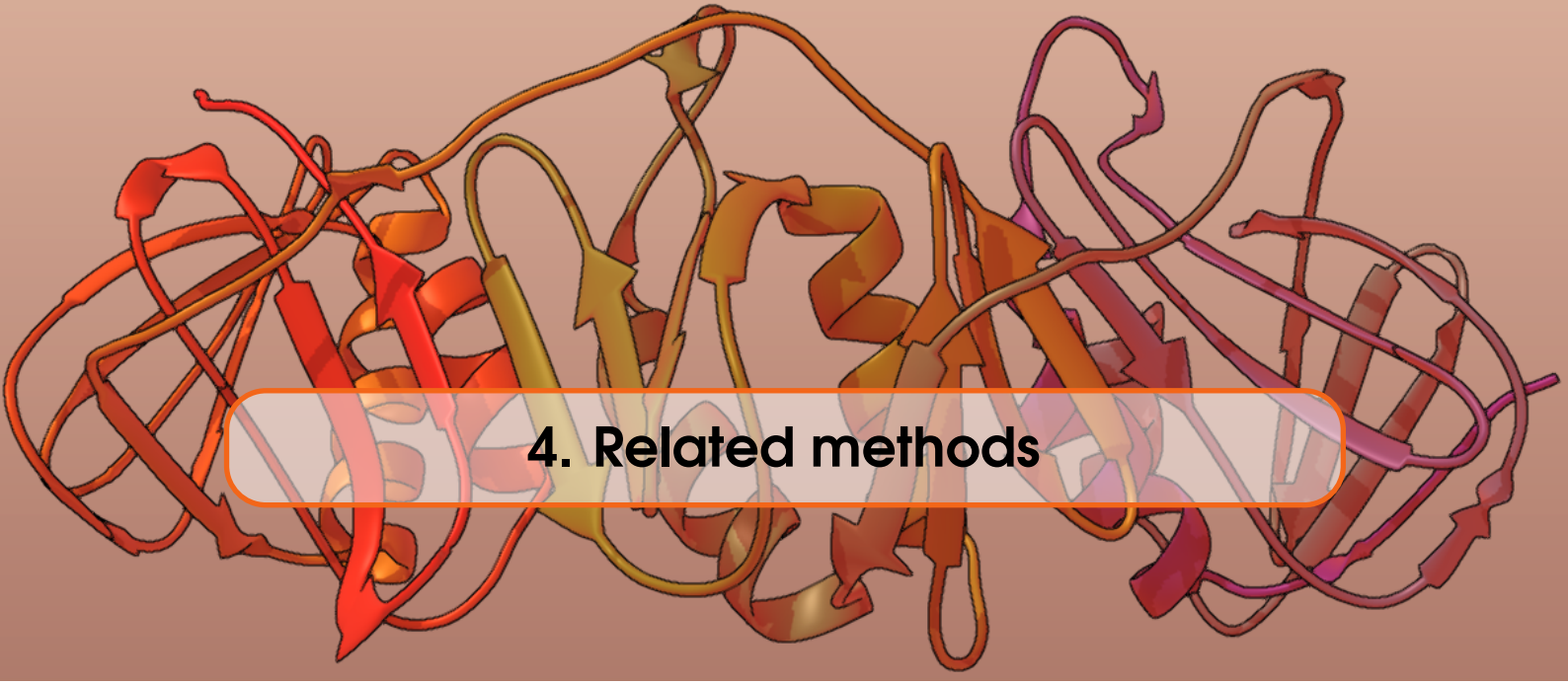
And the results can be presented as a plot (similar to the one in Figure 3.3) using the following python-based post-processing,

```
python benchmark.py results-R-20modes.txt
```

Where the 'benchmark.py' script is

```
#benchmark.py to print the RMSD reduction results
import sys, os
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import matplotlib.cm as cm
import collections

inputFile = sys.argv[-1]

with open(inputFile) as f:
    lines = f.readlines()

inputfiles = []
rmsds = []
frequencies = []

globalData = {}
startRMSD = 0
for line in lines:
    words = line.split()
    if (len(words)==0) : continue
    if (words[0] == "**********************************************************************") :
        if (len(rmsds)) :
            globalData[rmsds[0]] = [inputfiles, rmsds]
        inputfiles = []
        frequencies = []
        rmsds = []

    if (words[0]=="Input"):
        inputfiles.append(words[4].split('/')[-1])

    if (words[0][0]=="["):
        frequencies.append(float(words[-1]))

    if (startRMSD) :
        startRMSD = 0
        for w in words:
            rmsds.append(float(w))

    if (words[0]=="RMSD"):
        startRMSD = 1

globalData = collections.OrderedDict(sorted(globalData.items()))

cutoff = 2.0
nModes=20

Quality =  np.zeros(nModes)
QualityAbs =  np.zeros(nModes)
lenQ = 0
columns = []
labels = []
rows = []
expectedLength = len(list(globalData.values())[0][1])
print("expectedlength = ", expectedLength)
for dat in globalData:
    rmsds = globalData[dat][1]
    initRMSD = rmsds[0]
    if (initRMSD < cutoff or initRMSD > 5) : continue
    lenQ += 1
    for idx, r in enumerate(rmsds[1:nModes+1]):
        Quality[idx] += (initRMSD - r) / initRMSD
        QualityAbs[idx] += (initRMSD - r)

    if (initRMSD < cutoff) : continue
    columns.append(globalData[dat][0][:4].lower()) # column name
    labels.append("%.1f"%(initRMSD))
    previousRmsd = initRMSD
```

```
    reductions = []
    while(len(rmsds) < expectedLength) : rmsds.append(rmsds[-1])
    for idx, r in enumerate(rmsds[1:nModes+1]):
        reduction = previousRmsd - r
        previousRmsd = r
        relativeReduction = reduction / initRMSD
        if (len(rows)<idx+1) : rows.append([])
        rows[idx].append(relativeReduction)

index = np.arange(len(columns)) + 0.3
bar_width = 1.0

print ("Quality, Abs Quality")
for idx,q in enumerate(Quality):
    print (idx+1, q/lenQ, QualityAbs[idx] / lenQ)
exit

print (columns)
print (labels)

lenRows = len(rows)
colormap = plt.cm.Vega20b
colors = colormap(np.linspace(0, 1, lenRows))

normalize = mcolors.Normalize(vmin=0, vmax=lenRows)

# Initialize the vertical-offset for the stacked bar chart.
y_offset = np.array([0.0] * len(columns))

# Plot bars and create text labels for the table
fig = plt.figure()
ax1 = fig.add_subplot(111)
ax2 = ax1.twiny()

cell_text = []
for idx, row in enumerate(rows):
    plt.bar(index, row, bar_width, bottom=y_offset, color=colors[idx])
    y_offset = y_offset + row
    cell_text.append(['%1.1f' % (x/1000.0) for x in y_offset])

# setup the colorbar

ax1.set_ylabel("RMSD reduction", size=12)
ax1.set_xlabel("Ligand code", size=12)
ax2.set_xlabel("RMSD, $\t{\AA}$", size=12)

ax1.set_xticks(index)
ax2.set_xticks(index)

ax1.set_xticklabels(columns, rotation=90, size=10)
ax2.set_xticklabels(labels, rotation=90, size=10)

ax2.set_xlim(plt.axis()[:2])

print (plt.axis())
print (ax1.set_xlim(plt.axis()[0],plt.axis()[1]*1.26))
ax1.set_frame_on(False)

scalarmappaple = plt.cm.ScalarMappable(norm=normalize, cmap=colormap)
scalarmappaple.set_array(index)

plt.colorbar(scalarmappaple,ticks=range(1,nModes+1))
plt.show()
```



4. Related methods

4.1 GUI

A graphical user interfaces created for the SAMSON software platform is available at `https://www.samson-connect.net`. Figure 4.1 shows the current version of this GUI.

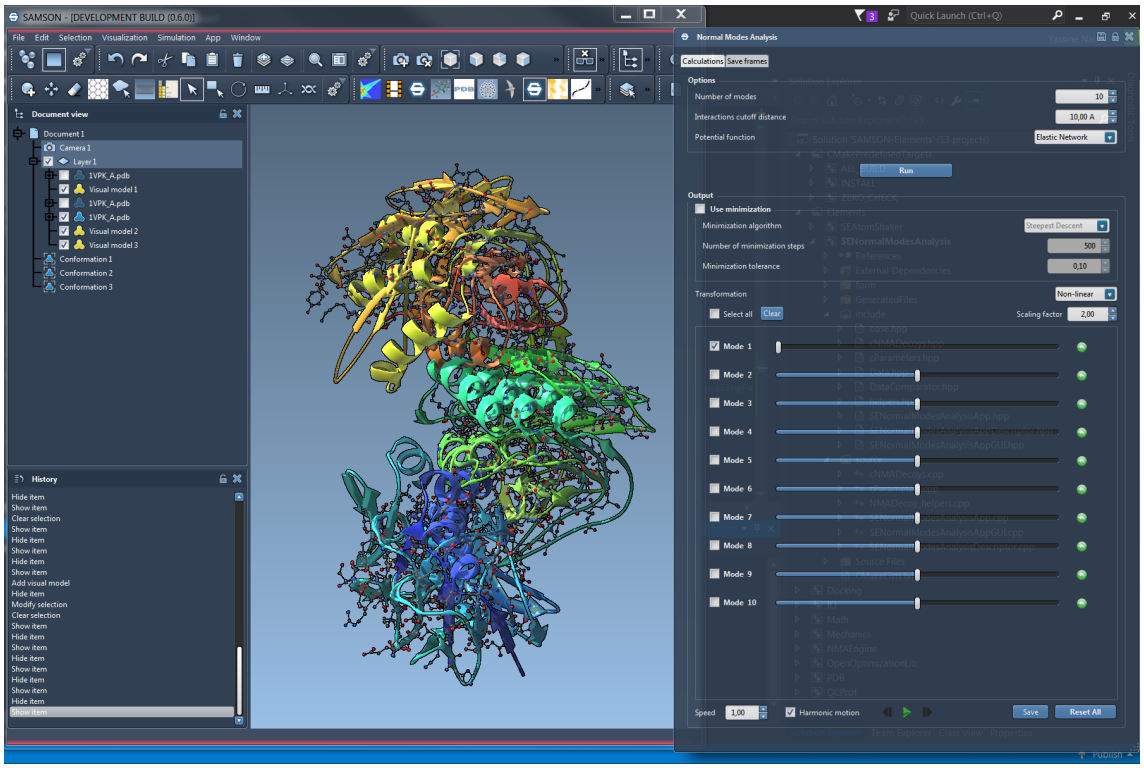


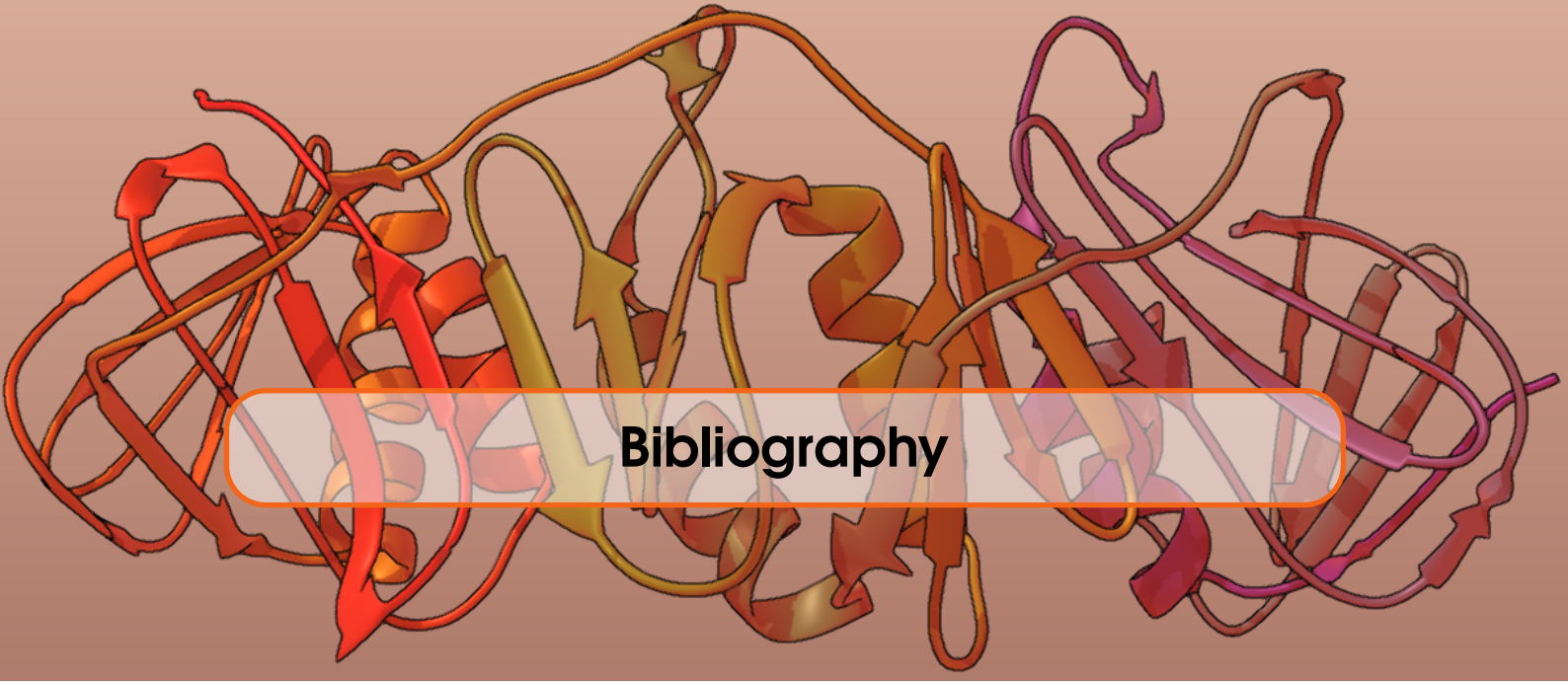
Figure 4.1: NOLB graphical user interface for the SAMSON software platform.

4.2 Morphing paths

The NOLB normal modes can be used to produce morphing pathways between multiple conformations of biological molecules. It is not a well tested approach yet. However, in the future it can be merged with the NOLB program.

4.3 SAXS-guided structure optimization

We have been also using the NOLB normal modes as a low-dimensional representation of the protein motion subspace in the recent CASP12 blind structure prediction exercise. More precisely, we used for in the SAXS-assisted protein targets, where we numerically computed the gradient of the SAXS loss function and optimized the structures towards the gradients in this low-dimensional space. This method will be distributed separately as a part of our Pepsi-SAXS scattering package at `https://team.inria.fr/nano-d/software/pepsi-saxs/`.



Bibliography

[1] E Bright Wilson, J C Decius, and Paul C Cross. *Molecular Vibrations: The Theory of Infrared and Raman Spectra*. McGraw-Hill, 1955 (cited on page 3).

[2] Ivet Bahar et al. "Normal Mode Analysis of Biomolecular Structures: Functional Mechanisms of Membrane Proteins". In: *Chem. Rev.* 110.3 (Mar. 2010), pages 1463–1497. DOI: 10.1021/cr900095e. URL: https://doi.org/10.1021%7B%%7D2Fcr900095e (cited on pages 3, 6).

[3] Timothy R Lezon et al. "Elastic Network Models for Biomolecular Dynamics: Theory and Application to Membrane Proteins and Viruses". In: *Handbook on Biological Networks*. World Scientific Pub Co Pte Lt, Dec. 2009, pages 129–158. DOI: 10.1142/9789812838803_0007. URL: http://dx.doi.org/10.1142/9789812838803%7B%5C_%7D0007 (cited on page 4).

[4] Alexandre Hoffmann and Sergei Grudinin. "NOLB: Nonlinear Rigid Block Normal-Mode Analysis Method". In: *J. Chem. Theory Comput.* 13.5 (2017), pages 2123–2134 (cited on page 5).

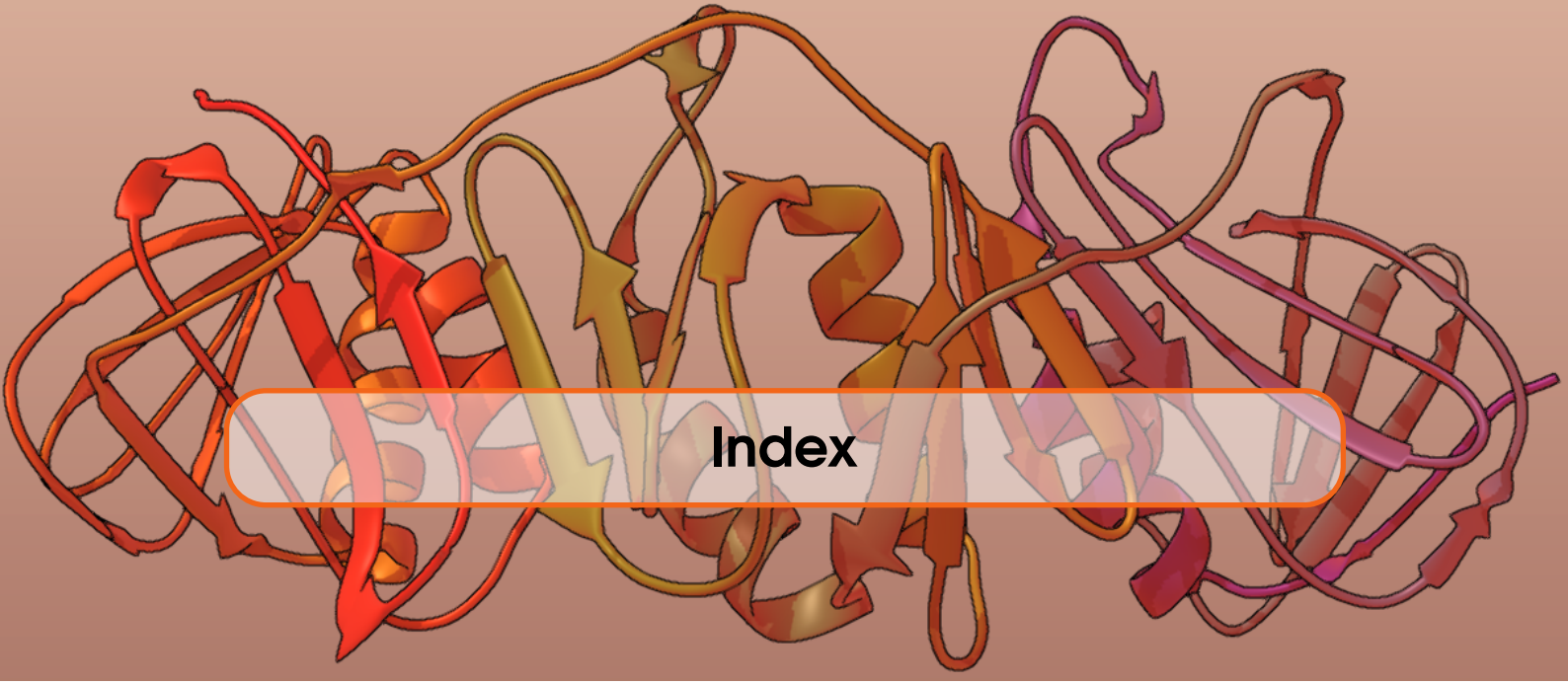
[5] Pemra Doruker, Ali Rana Atilgan, and Ivet Bahar. "Dynamics of Proteins Predicted by Molecular Dynamics Simulations and Analytical Approaches: Application to α-Amylase Inhibitor". In: *Proteins: Struct., Funct., Bioinf.* 40.3 (2000), pages 512–524 (cited on page 6).

[6] A R Atilgan et al. "Anisotropy of Fluctuation Dynamics of Proteins with an Elastic Network Model". In: *Biophys. J.* 80.1 (Jan. 2001), pages 505–515. DOI: 10.1016/s0006-3495(01)76033-x. URL: https://doi.org/10.1016%7B%%7D2Fs0006-3495%7B%%7D2801%7B%%7D2976033-x (cited on page 6).

[7] Svetlana Artemova, Sergei Grudinin, and Stephane Redon. "A Comparison of Neighbor Search Algorithms for Large Rigid Molecules". In: *J. Comput. Chem.* 32.13 (2011), pages 2865–2877 (cited on page 6).

[8] Emilie Neveu et al. "RapidRMSD : Rapid determination of RMSDs corresponding to motions of flexible molecules". Unpublished (cited on page 20).

[9] Thom Vreven et al. "Updates to the Integrated Protein-Protein Interaction Benchmarks: Docking Benchmark Version 5 and Affinity Benchmark Version 2." In: *J. Mol. Biol.* 427.19 (Sept. 2015), pages 3031–3041. ISSN: 1089-8638. DOI: 10.1016/j.jmb.2015.07.016 (cited on page 23).



Index