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 nonlinear normal mode analysis in the Cartesian coordinates that extrapolates a motion computed from instantaneous linear and angular velocities to large amplitudes. The scheme can be considered as a natural evolution of the widely used rotations-translations of blocks (RTB) method [3, 4].

1.2 NMA theory

Let us consider a molecular system with $N$ atoms at an equilibrium position $q_0 \in \mathbb{R}^{3N}$. Let $V : \mathbb{R}^{3N} \rightarrow \mathbb{R}$ be the potential energy of the molecular system. Let us also introduce $q \in \mathbb{R}^{3N}$, a small time-dependent molecular displacement of our system around $q_0$. The potential energy $V$ in the vicinity of $q_0$ thus can be given by its quadratic approximation, which allows to analytically solve the Newton’s equation of motion,

$$M(q + \dot{q}_0) + \nabla V(q_0 + q) \approx M\ddot{q} + Kq = 0,$$

(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 $\Lambda$ of the mass-weighted stiffness matrix $K_w = M^{-1/2}KM^{-1/2}$.

$$K_w = L\Lambda L^T.$$  

(1.2)

Let $\eta \in \mathbb{R}^{3N}$ be the projection of $q$ into the eigenspace of $K_w$ and let us call $(\lambda_i)_{i=0\ldots3N}$ the diagonal values in $\Lambda$. Then, left multiplication of eq. 1.1 by $L^T M^{1/2}$ gives the following system of uncoupled
Chapter 1. Theory

equations,

\[ \eta = L^T M^{1/2} q \]

\[ \ddot{\eta}_i + \lambda_i \eta_i = 0 \quad i = 1 \ldots 3N' \]

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

\[ \sqrt{M_b} \dot{q} = \sum_{k=1}^{N_b} \sqrt{m_k} \dot{q}_k \quad \text{for a translation} \]

\[ I^{1/2} \dot{\theta} = \sum_{k=1}^{N_b} \sqrt{m_k} (q_k \times \dot{q}_k) \quad \text{for a rotation} \]

where \( M_b \) is the total mass of the rigid block, \( I \) is the rigid block’s inertia tensor, \( \dot{q} \) 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 \( \dot{q} \) are then obtained by differentiating (1.4) with respect to \( \dot{q}_k \) [5]. This leads to translation \( P_t \) and rotation \( P_r \) matrices of size \( 3N_b \times 3 \) each, computed for each of the rigid blocks and written through their \( k \) 3 \times 3 square components,

\[ P_{tk} = \sqrt{\frac{m_k}{M_b}} E_3 \quad \text{for a translation} \]

\[ P_{rk} = -\sqrt{m_k (I)^{-1/2}} [r_k - r^{\text{COM}}] \times \quad \text{for a rotation} \]

where \( k \) is one of \( N_b \) atom indices, \( r_k \) is the position of the corresponding atom in the block, and \( r^{\text{COM}} \) the position of the block’s center of mass (COM). The rigid block’s displacement \((\delta, \theta)\)
6-vector is then obtained by summing up the displacements in the RTB coordinate frame,

\[
\delta = \sum_{k=1}^{N_b} P_t^T \delta k, \quad \text{for a translation}
\]

\[
\theta = \sum_{k=1}^{N_b} P_r^T \theta k, \quad \text{for a rotation}
\]

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

\[
P = \begin{pmatrix}
P_t^1 & P_r^1 \\
\vdots & \vdots \\
P_t^n & P_r^n
\end{pmatrix}
\]

(1.7)

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

\[
P^T K_w P = \tilde{L} \tilde{\Lambda} \tilde{L}^T,
\]

(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

\[
K_w = (PL) \tilde{\Lambda} (PL)^T.
\]

(1.9)

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

\[
L^w = PL.
\]

(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 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,

\[
\vec{v} = M_b^{-1/2} \vec{v}_w,
\]

\[
\vec{\omega} = I^{-1/2} \vec{\omega}_w.
\]

(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

\[
\Delta \vec{x} = a \vec{v}
\]

\[
\vec{\Delta} = \vec{\omega} / \| \vec{\omega} \|_2,
\]

\[
\Delta \phi = a \| \vec{\omega} \|_2,
\]

(1.12)

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

\[
\Delta \vec{x} = \Delta \vec{x}_\perp + \Delta \vec{x}_\parallel,
\]

(1.13)

where \( \Delta \vec{x}_\perp \) is orthogonal to \( \vec{n} \) and \( \Delta \vec{x}_\parallel \) is collinear to \( \vec{n} \), and represent the \( \Delta \vec{x}_\parallel \) related motion as a pure rotation about a new center \( \vec{r}_0 \), such that the final rigid blocks’s position is

\[
\vec{A}' = R(\Delta \phi, \vec{n})(\vec{A} - \vec{r}_0) + \vec{r}_0 + \Delta \vec{x}_\parallel,
\]

(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 [6].
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 implemented an all-atom Anisotropic Network Model (ANM) [7, 8], 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,

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

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, \(\gamma\) 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, 7, 8],

\[
H_{ij} = -\frac{\gamma}{(d_{ij}^0)^2} \begin{pmatrix}
(x_{ij}^0)^2 & x_{ij}^0 y_{ij}^0 & x_{ij}^0 z_{ij}^0 \\
x_{ij}^0 y_{ij}^0 & (y_{ij}^0)^2 & y_{ij}^0 z_{ij}^0 \\
x_{ij}^0 z_{ij}^0 & y_{ij}^0 z_{ij}^0 & (z_{ij}^0)^2
\end{pmatrix},
\]

\(i \neq j\)

\[
H_{ii} = -\sum_{j \neq i} H_{ij},
\]

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 [9].

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/CRHS Grenoble, France, 2018.          **
** e-mail: sergei.grudinin@inria.fr                                      **
** http://team.inria.fr/nano-d/software/nolb-normal-modes/                **
******************************************************************************

USAGE:


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.
Chapter 2. Program options

-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. Please note that s=0 suppresses the output.

-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.

--rand
Random seed for decoy generation. Off by default.
2.1 Usage

```
--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.

--analyze
   Analyze the computed modes. Off by default.

--collectivity
   Prints modes collectivity. Off by default.

--nlin
   Deterministic nonlinear transitions if the reference structure is
given, off by default.

--nIter <number of additional Hessian matrix computations>
   Number of additional computations of the Hessian matrix if the
reference structure is given, 0 by default. If enabled, the
recommended value is 5.

-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: 1.0, April 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 nonlinear 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.
Version 0.9 from March 2018:
   Added basic analysis of the motions.
Version 1.0 from April 2018:
```
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 option 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 option 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 option 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 option 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 option 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 option 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.
### 2.2 Main options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>--hetatm</strong></td>
<td>Flag to read heteroatoms from the input file. False by default. Should be used for lipids, small molecules, etc.</td>
</tr>
<tr>
<td><strong>--trajectory</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>--frames &lt;max number of input trajectory frames&gt;</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>-r &lt;maximum rmsd&gt;, --rmsd &lt;maximum rmsd&gt;</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>--dist &lt;RMSD distribution&gt;</strong></td>
<td>This option defines the distribution of the decoys RMSD, 0 for constant RMSD (default), 0.5 for quadratic dependence, 1 for linear dependence, etc.</td>
</tr>
<tr>
<td><strong>--rand</strong></td>
<td>Flag to reinitialize the random seed for decoy generation, off by default.</td>
</tr>
<tr>
<td><strong>-m, --minimize</strong></td>
<td>Flag to minimize the obtained structures, off by default.</td>
</tr>
<tr>
<td><strong>--nSteps &lt;number of minimization steps&gt;</strong></td>
<td>This option sets the number of minimization steps, 500 by default.</td>
</tr>
<tr>
<td><strong>-t &lt;minimization tolerance&gt;, --tol &lt;minimization tolerance&gt;</strong></td>
<td>This option sets the minimization tolerance, 0.1 by default.</td>
</tr>
<tr>
<td><strong>--covScaling &lt;scaling of covalent interactions&gt;</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>-dcd &lt;dcd file name&gt;, --dcd &lt;dcd file name&gt;</strong></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 2. Program options

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.

--analyze

This flag turns on the analysis of the computed normal modes. Currently, only the collectivities are computed. Off by default.

--collectivity

This flag turns on outputting the collectivities of the modes. Off by default.

--nlin

This flag turns on a deterministic nonlinear transition computation if the reference structure is given. No additional diagonalizations are required. Off by default.

--nIter

This flag specifies the number of additional computations of the Hessian matrix if the reference structure is given, 0 by default. If enabled, the recommended value is 5. Will only take effect if the --nlin flag is on.

--blocks <Rigid blocks ids filename>

Name of the text file where rigid blocks are specified. Currently, only residue IDs are supported. Rigid blocks are separated by the "#" signs and each resID starts with a new line. All omitted residues will be considered as individual rigid blocks. For example, the following file will add two additional rigid blocks composed of residues with IDs 1-3 and 4-6:

```
# 1 2 3 # 4 5 6
```

-h, --help

Flag to display usage information and exit.

--version

Flag to display version information and exit.
2.3 Experimental options

--log
Flag to display the ChangeLog information and exit.

--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!),

---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---

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
```

The first section lists the structural parameters, the number of atoms, chains, and the number of
Chapter 3. Examples

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 
```
3.2 Comparison with linear NMA

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,

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,
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 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,

Finally, 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,
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,
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,

You may see that 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 [10]. 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]................................................. : 0.00687768
[4]................................................. : 0.00721599
[5]................................................. : 0.00790467
[6]................................................. : 0.00920148
[7]................................................. : 0.0115896
[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---
```

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.](image_url)
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,

```
Number of files.................................................. : 100
Output RMSD...................................................... : 10 Å
Using outputfilename............................................. : 2ch7_nlb_decoys.pdb

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,

```
Number of files.................................................. : 100
Output RMSD...................................................... : 10 Å
Using outputfilename............................................. : 2ch7_nlb_decoys.pdb
Current energy................................................... : 0.272494
Number of minimization steps................................. : 0
Current Gradient norm.......................................... : 0.0210788
Displacement .................................................... : 0.00167775
RMSD after minimization........................................ : 2.45655

Current energy................................................... : 6.27796
Number of minimization steps................................. : 1
Current Gradient norm.......................................... : 0.0738622
Displacement .................................................... : 0.0135281
RMSD after minimization........................................ : 7.13781

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
```

3.8 Structural ensembles
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) [11]. Figure 3.3 shows a summary of these computed for the proteins from the Protein Docking Benchmark v5 [12].

Figure 3.3: Transitions between the unbound (u) and bound (b) states of proteins from the Protein Docking Benchmark v5 [12]. The top x-axis shows $C_\alpha$ 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 $\geq 2$ 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

```
Datasheet  ----------------------------------------------------------
Input PDB file................................................... : 3L89_l_u.pdb
Number of chains read............................................ : 1
Number of atoms read............................................. : 1018

Input PDB file................................................... : 3L89_l_b.pdb
Number of chains read............................................ : 1
Number of atoms read............................................. : 987

Total number of gaps ............................................ : 0
Alignement....................................................... :
CEEPPTFEAMELIGKPKPYYEIGERVDYKCKKGYFYIPPLATHTIDNHTWLPVSDAC
CEEPPTFEAMELIGKPKPYYEIGERVDYKCKKGYFYIPPLATHTIDNHTWLPVSDAC

YRETCPYIRDPLNGQAVPANGTYEFGYQMHFICNEGYYLIGEEILYCELKGSVA1WGBKFP
YRETCPYIRDPLNGQAVPANGTYEFGYQMHFICNEGYYLIGEEILYCELKGSVA1WGBKFP
```
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:

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

Where the 'benchmark.py' script is
```python
#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][0][:4].lower()) # column name
    labels.append("%.1f"%(initRMSD))
    previousRmsd = initRMSD
```

3.10 Nonlinear structural transitions

Here we demonstrate the way to produce deterministic nonlinear transitions as in the previous example.
**Example 13** In this example we produce a nonlinear 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 -n 7 --nlin
```

This creates a '3L89_l_u_nlb.pdb' pdb trajectory file for the best nonlinear transition between the states, and also the following output (cut for brevity):

```
RMSD reduction : ................................................ :
  4.445432  2.258740  2.177358  1.470716  1.411807  1.337628  1.337624
Amplitudes :
  454.891 -76.1758  197.929 -39.3717  -7.06865  83.5584  3.02367
RMSD reduction : ................................................ :
  4.041845  2.112556  2.041760  1.442365  1.394135  1.393912  1.333585
RMSD reduction : ................................................ :
  3.680333  1.982142  1.920188  1.412448  1.372653  1.372480  1.323201
...
RMSD reduction : ................................................ :
  1.192304  1.190237  1.189976  1.189690  1.189683  1.189598  1.189536
Initial RMSD..................................................... : 4.44543
Final RMSD...................................................... : 1.18954
Reduction in RMSD by............................................ : 0.732414
```

The RMSD reduction values in lines correspond to the analytical estimation of these for the linear case. The first value in the column correspond to the nonlinear case. We see that the final nonlinear RMSD to the target is 1.19 Å, whereas the final linear RMSD to the target is 1.34 Å.

### 3.11 Analysis of modes

To analyse the obtained normal modes, one can use different criteria, for example, the reduction of the RMSD between two states of a molecular system. Another characteristic is the collectivity $\kappa$, which is proportional to the exponential of the “information entropy” of the eigenvector $q_i$ [13]:

$$\kappa_i = \frac{1}{N} \exp \left\{ \sum_{n=1}^{N} q_{i,n}^2 \log q_{i,n}^2 \right\},$$

(3.1)

where the sum runs over all the $N$ atoms in the molecule, and

$$q_{i,n}^2 = \frac{q_{i,3n-2}^2 + q_{i,3n-1}^2 + q_{i,3n}^2}{m_n},$$

(3.2)

with the normalization factor $\alpha$ taken such that $\sum q_{i,n}^2 = 1$. $N\kappa$ gives an effective number of nonzero eigenvector components $q_{i,n}^2$. Thus, $\kappa$ is confined to the interval \{N$$^{-1};1\}. If $\kappa_1 = 1$, then the corresponding mode is maximally collective and has all the identical amplitudes $q_{i,n}^2$, which happens for rigid-body motion, for example. In the limit of extreme local motion, where a mode affects a single atom only, $\kappa_1 = 1$ is minimal and equals to $1/N$.

**Example 14** In the following example we will compute collectivities for the first twenty modes of the coiled-cil protein 2ch7.

```
NOLB 2ch7.pdb --analyze -n 20 -s 0
```

This creates the following output (cut for brevity),
We can see that the modes 10, 11, 13, 18, 19 are the most collective, and the modes 15 and 17 are the least collective.
4. Related methods

4.1 GUI

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

![NOLB graphical user interface for the SAMSON software platform.](image)

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/.


<table>
<thead>
<tr>
<th>A</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of basic molecular motions</td>
<td>Introduction</td>
</tr>
<tr>
<td>Analysis of modes</td>
<td>Main options</td>
</tr>
<tr>
<td>Analysis of protein docking poses</td>
<td>Morphing paths</td>
</tr>
<tr>
<td>Analysis of trajectories</td>
<td>NMA theory</td>
</tr>
<tr>
<td></td>
<td>nonlinear structural transitions</td>
</tr>
<tr>
<td>C</td>
<td>N</td>
</tr>
<tr>
<td>Clustering of MD trajectory frames</td>
<td>NMA theory</td>
</tr>
<tr>
<td>Collectivity</td>
<td>NMA theory</td>
</tr>
<tr>
<td>Comparison with linear NMA</td>
<td>non-linear structural transitions</td>
</tr>
<tr>
<td>Currently suppressed options</td>
<td>Structural transitions</td>
</tr>
<tr>
<td>D</td>
<td>P</td>
</tr>
<tr>
<td>Decoys</td>
<td>Protein Docking Benchmark</td>
</tr>
<tr>
<td>E</td>
<td>R</td>
</tr>
<tr>
<td>Example with a bigger system that contains heteroatoms</td>
<td>RTB projection method</td>
</tr>
<tr>
<td>Experimental options</td>
<td>Structural ensembles</td>
</tr>
<tr>
<td>F</td>
<td>S</td>
</tr>
<tr>
<td>Finding the best transition between two protein conformations</td>
<td>SAXS-guided structure optimization</td>
</tr>
<tr>
<td>G</td>
<td>T</td>
</tr>
<tr>
<td>GUI</td>
<td>The NOLB method</td>
</tr>
<tr>
<td>H</td>
<td>U</td>
</tr>
<tr>
<td>Heteroatoms</td>
<td>Usage</td>
</tr>
</tbody>
</table>