Andrew Torda
Björn Hansen

Universität Hamburg

Zentrum für Bioinformatik

Übung zur Vorlesung
Grundlagen der Strukturanalyse

Wintersemester 2011/2012

22.11.2011

Übung 3: Distance Geometry Methods

1. Contents

2. Introduction

Distance geometry is the method used to build 3D coordinates for molecules based on distance and sometimes angle information. If one had accurate and perfect distance measurements, it would be easy to calculate coordinates. In practice, this never happens. Typically, NMR spectra do not give a complete and accurate set of distances, and it is possible to make a number of different 3D structures which fit the set of distances derived from the spectra. The more (correct) NMR data one has, the smaller the set of alternative structures that fit this data, and so the real structure of the molecule is better determined.

An important question in real structure determination is, "how much data does one need?". To answer the question, one can work with synthetic data. You can take a real structure and extract all the distances from it and use these to calculate structures. You can throw away part of the data and see how well determined the structures are.

The more data you throw away, the worse the calculated structures will be. If you do not see this, you have probably typed something wrong.

3. The Tinker Package

The exercise is based on some programs from the *Tinker* package (http://dasher.wustl.edu/tinker). The distance geometry program that we will use is called *distgeom*. It needs the following files:

`*.xyz` *tinker*'s cartesian coordinate file format, which describes atom connectivity and position for each atom in the system and

`*.key` keyword parameter file. For *distgeom*, it lists the distance and dihedral restraints that must be applied to generate the conformation.

The *distgeom* program needs a *trial* structure, in the form of *xyz* file, in order to know the stereochemistry, composition and bond connectivity for the molecule. It generates structures based on the information in the *keyfile*, and optimizes them to make sure that chemical bonds are of the normal lengths, and non-bonded atoms are not too close together.

The results (one or many 3D structures) are stored in the xyz coordinate format as "cycle files". These have a file extension, e.g. `*.001`, `*.001_2`, `*.002...`, etc. In order to turn these into a normal PDB file, there is a program called ***xyzpdb***, which needs, in addition to an *xyz* file:

`*.seq`

sequence file, giving the sequence of the protein.

`*.prm`

potential energy parameter file. We will use `amber.prm` for this exercise.

4. Aim of the Exercise

The aim is to find the smallest set of restraints which can still produce a structure that looks correct.

A local program ***keyfilegen*** is used to generate the `*.key` file for *distgeom*, from a set of precalculated interatomic distances, of the kind that might normally be measured by NMR. These are the small interatomic distances - less than 5 Å. It will ask you for the PDB ID of the *.dis file, and the upper and lower error bounds and amount of distance restraints that should be generated. It then writes the `.key` file.

You will generate different key files, each specifying a set of artificial distance restraints. You can then run *distgeom* to see the effect on the generated structure when using:

- almost perfect distance restraints
- almost no restraints
- a realistic set of distance restraints

5. Instructions

Chimera

Use the version of chimera from the previous Übung.

Files

All the files and programs you need for this exercise can be found in the directory:

`~hansen/teaching/distgeom`

The files are:

`1L2Y.dis`	the set of interatomic distances.
`1L2Y.seq`	the sequence of the known structure.
`1L2Y.xyz`	the initial trial structure.
`1L2Yorg.pdb`	the known structure in PDB format.
`1L2Ytrl.pdb`	the initial trial structure in PDB format.
`amber.prm`	amber potential energy parameters.

The programs are:

distgeom	program for running distance geometry functions.
keyfilegen	program for generating the .key file
mxyzpdb	program to combine several .xyz files to one multi-model PDB.
pdbxyz	program for converting PDB format to .xyz format.
xyzpdb	program for converting .xyz format to PDB format.

The structure you will recreate has the protein databank code 1L2Y. First copy all the 1L2Y files into your working directory.

`> cp ~hansen/teaching/distgeom/1L2Y* .`

Use *chimera* to view the original structure `1L2Yorg.pdb`.

`> /usr/local/zbhtools/chimera-1.5.2/bin/chimera 1L2Yorg.pdb &`

Setting Distance Restraints

Run the program *keyfilegen* to create a set of artificial distance restraints

`> ~hansen/teaching/distgeom/keyfilegen`

The program will ask you some questions. For the moment, use the example values given below:

Please Enter the Upper Bound in Angstrom: *<type in the desired upper bound, e.g. 0.05>*

Please Enter the Lower Bound in Angstrom: *<type in the desired lower bound, e.g. 0.01>*

Please Enter the Amount of Restraints in %: *<type in the desired amount, e.g. 100>*

Please Enter the Name of the dis File: *<enter 1L2Y.dis>*

`1L2Y.key` is generated.

You have now generated a *keyfile* of containing every distance restraint between atoms separated by less than 5 Å, with an error between 0.01 and 0.05 Å. The error means that for a distance restraint of 3 Å, you allow the structure to have a distance from 2.99 to 3.05 Å.

When you use this program to generate other sets of distance restraints, you must take care when entering the parameters. The program will fail with an error if you enter any non-numeric characters for the bounds or amount of restraints.

Using The Tinker Program Distgeom

Run the Tinker program, *distgeom*, to generate the protein structure. If you copied all the files as described above, it should be in the current directory. The *keyfile* and initial coordinates are given to the program by the argument 1L2Y:

```
> ~hansen/teaching/distgeom/distgeom 1L2Y
```

It will now ask various questions. For the first run, you should use the default by simply pressing the return key(<cr>), this enters the default value shown in the square brackets:

Number of Distance Geometry Structures Desired [1]:

<default value is 1., <cr> or you can assign another value>

Impose Chirality Constraints on Tetrahedral Atoms [Y]:

<default is Yes, <cr> or you can type N for No>

Use "Floating: Chirality for -XH2- and -XH3 groups [N]:

<default is No, <cr> or you can type Y for Yes>

Impose Planarity and/or Chirality of Trigonal Atoms [Y]:

<default is Yes, <cr> or you can type N for No>

Impose Torsional Planarity on Adjacent Trigonal Atoms [Y]:

<default is Yes, <cr> or you can type N for No>

Do You Wish to Examine or Alter the Bounds Matrix [N]:

<default is No, <cr> or you can type Y for Yes>

Select the Enantiomer closest to the Input Structure [N]:

<default is No, <cr> or you can type Y for Yes>

Refinement via Minimization or Annealing [M orA,<CR>=A]:

<default is annealing, <cr> or you can type M for minimization>

The Calculation should take a few seconds if you used the default settings. Look for the result in a *cycle file*, where the first generated structure will be called `1L2Y.001`.

Converting .xyz to .pdb

First copy the potential parameter file into your working directory.

```
> cp ~hansen/teaching/distgeom/amber.prm .
```

Use the program *xyzpdb*, to convert one of the resulting structures to a PDB file.

```
> ~hansen/teaching/distgeom/xyzpdb 1L2Y.001 amber
```

The converted structure will be stored in the file *1L2Y.pdb* or *1L2Y.pdb_*X, if the file already exists. *X* is an integer greater than 1.

Rename this PDB file to something that describes how the structure was generated (using mv).

```
> mv 1L2Y.pdb 1L2Y100.pdb
```

The 100 stands for 100% restraints. Clear naming will be important when you try to compare the results for many different restraint sets later.

Looking at the Structure that is generated

Now you can use *chimera* to compare the generated structure (`1L2Y100.pdb`) with the real structure (`1L2Yorg.pdb`) and the initial trial structure (`1LL2Ytrl.pdb`), which was used to generate `1L2Y100.xyz`, and then used by *distgeom* to derive a new structure.

1. Start *chimera* with the three structures:

```
chimera 1L2Y100.pdb 1L2Yorg.pdb 1L2Ytrl.pdb &
```

2. Open the command line interface and enter:

```
match #0 #1
```

This command superimposes model #0 (*1L2Y100.pdb*) onto model #1 (*1L2Yorg.pdb)*. A message is printed:

```
RMSD between 304 atom pairs is 'X' angstroms.
```

'X' gives the average distance between the atoms in the model, and the original structure.

3.Now superimpose the initial trial structure onto the original structure:

```
match #2 #1
```

4.Finally, you can measure the RMSD between the generated structure and the trial structure:

```
match #2 #0
```

Making a Number of Structures

The distance geometry algorithm uses random numbers to generate the structure, and for some sets of restraints, many structures can be made. Run the *distgeom* program again to make 10 structures:

Number of Distance Geometry Structures Desired [1]: *<enter 10>*

Once again, use the default for all other questions.

The program will finish after a minute or so. You should now see 10 new cycle files:

```
1L2Y.001_2, 1L2Y.002, ..., 1L2Y.010
```

`1L2Y.001_2` is the same conformation as 1L2Y.001, but since 1L2Y.001 was already exist, *Tinker* has added the _2.

Making a Multiple Model PDB File

The 10 structures that were made in the last step are all derived from the same set of restraints, so it is sensible to group them into one PDB file for analysis in chimera. The program *mxyzpdb* does this:

```
> ~hansen/teaching/distgeom/mxyzpdb 1L2Y.0*
```

The argument `1L2Y.0*` is a regular expression, that matches the names of all 11 cycle files. *mxyzpdb* understands the Tinker renaming rules, and will always take the *latest* version of a cycle file, and so only the 10 new structures will be used. From the output of the program, you can see that *mxyzpdb* runs *xyzpdb* 10 times, once for each cycle file. The final message indicates that a new PDB files has been written:

```
10 models written to 1L2Y.m.pdb.
```

Examining an Ensemble of Models in chimera (1)

The first *chimera* tutorial introduced the tools that you should now use. First, analyze the ensemble of models by comparing them to the real structure:

1. Load the real structure and the multiple model set into *chimera*
2. Simplify the display to show only the backbone alpha-carbon trace (**chain @CA**)
3. Use the EnsembleMatch tool (Tools»MD/Ensemble Analysis»EnsembleMatch):
   Reference structure 1L2Yorg.pdb
   Alternative 1L2Y.m.pdb

The model ensemble was generated using all the distance restraints, with only a very small error tolerance. However, there is clearly some differences between the generated structures. Can you see what this difference is?

Examining an Ensemble of Models (2)

In a real distance geometry calculation, you would not have a known structure to compare with. In this case, it is normal to examine the different groups of conformations that are generated by the procedure:

1. Use the **EnsembleMatch** tool again:
Choose 1L2Y.m.pdb as both Reference and Alternative.
A different **EnsembleMatch** matrix will be generated, where each model is compared to each other. Can you see how many different groups of conformations exist in the ensemble?

Using Different Sets of Restraints

In order to generate more structures using different amount of restraints:

1. Run *keyfilegen* and enter a different percentage (%), but the same error bounds. The *1L2Y.key* file that is generated will overwrite the one you made in the previous step.
2. Run *distgeom*.
3. Convert any or all of the xyz files. Both *xyzpdb* and *mxyzpdb* will try to use the latest version of a cycle file, unless you explicitly give the version. Remember to rename the PDB files that you generate, using a name that describes the kind of distance geometry restraints that were used to generate the coordinates.
4. Examine the new structures in *chimera*.
   Try to see how few restraints are necessary to reconstruct the shape of 1L2Y, and how big an error is allowed before the structure generated by a given percentage of the restraints has no similarity to the true shape.

6. Assignment

Please answer the following questions in a brief written report (1-page), and send it to
hansen@zbh.uni-hamburg.de
not later than 12. December, 2011 with your full name written on it. Do not send TeX, openoffice or staroffice files. Plain ASCII or pdf files are preferred.

1. What is the smallest set of restraints that can still produce a reasonable structure of 1L2Y?
2. Explain what you have found from **EnsembleMatch**, why the generated structures can be so much different sometimes even they are from the same set of restraints?