A Buildup-based Error Minimization Method with Application to Protein Structure Determination

Zhenli Sheng (盛镇醴)

email: szl@lsec.cc.ac.cn

Institute of Computational Mathematics and Scientific/Engineering Computing,
Chinese Academy of Sciences

April 9th, 2013

seminar talk

Outline

Outline

Distance Geometry Problem

Find the coordinate vectors $x_1, x_2, \ldots, x_n$ that satisfy several given distances between them. Mathematically, this problem can be stated as follow,

Find $x_1, x_2, \ldots, x_n$, such that

$$\|x_i - x_j\| = d_{ij}, \quad (i,j) \in S.$$

or

$$l_{ij} \leq \|x_i - x_j\| \leq u_{ij}, \quad (i,j) \in S.$$

- The data given may have some errors.
- This problem can be formulated as global optimization problem.
- It has many applications.

Application I: Graph Realization

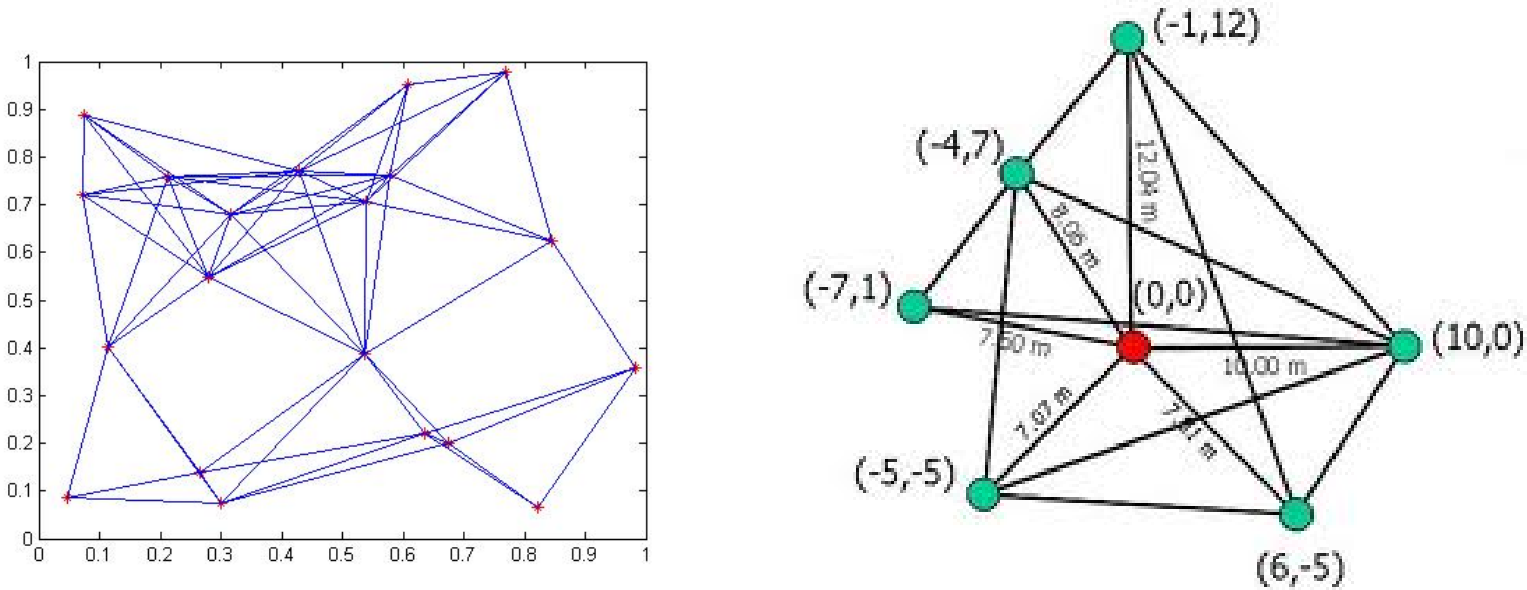


Figure 1 : Graph Realization in 2D

Given a graph G=(V,E), each edge has a weight.

Application II: Protein Structure Determination

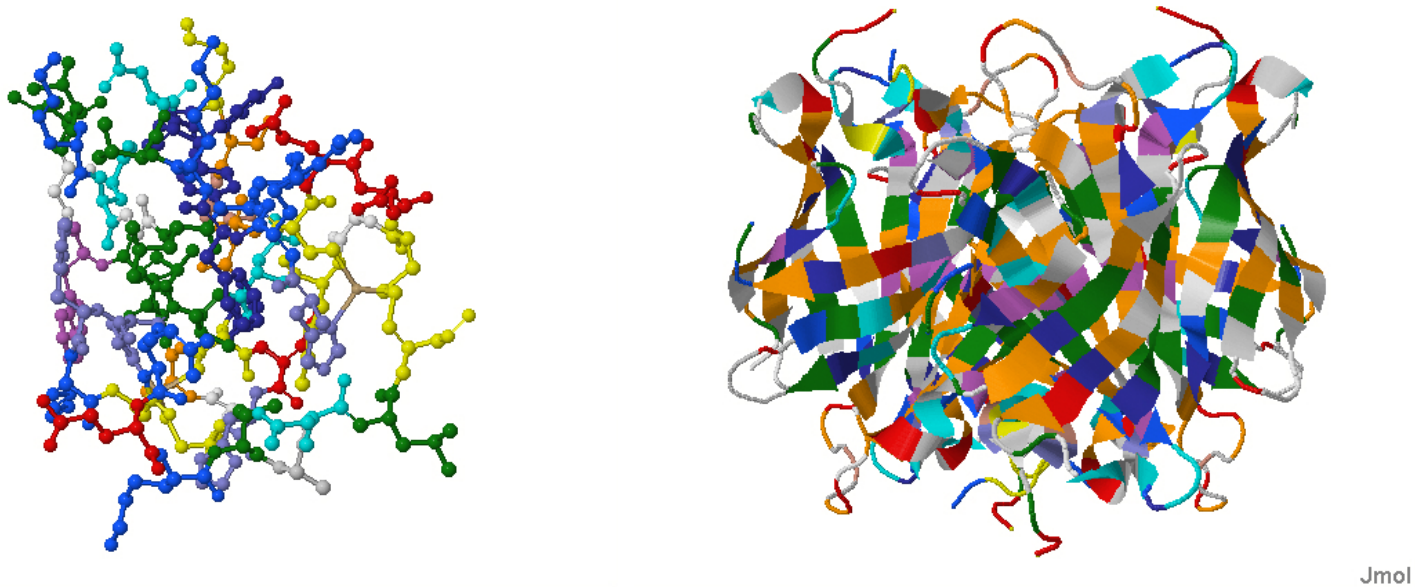


Figure 2 : Two proteins: 1PTQ and 1HQQ, in different display ways

Measure distances: NMR, X ray crystallography.

Application III: Sensor Network Localization

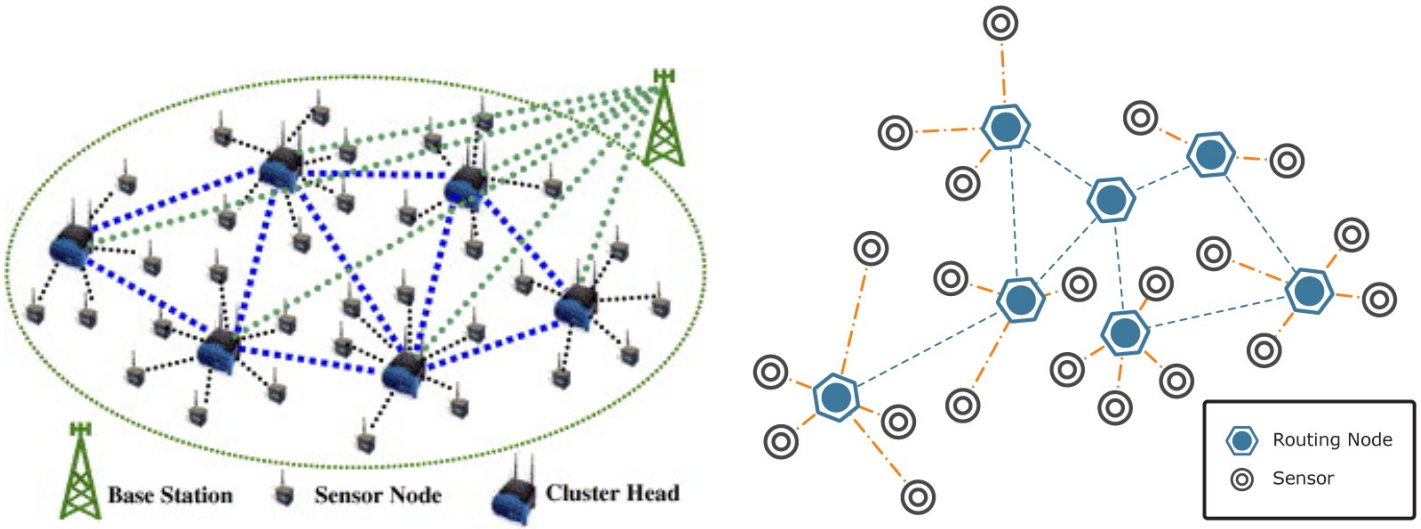


Figure 3 : Illustration of wireless sensor networks: anchor and sensor

Outline

Matrix Decomposition Method

Matrix decomposition method [Blumenthal 1953, Torgerson 1958] works for Distance Geometry problem with full set of exact distances.

Given a full set of distances, $d_{ij} = \|x_i - x_j\|, \quad i, j = 1, 2, \ldots, n$.

- Set $x_n = (0, 0, 0)^{\mathrm{T}}$, thus $d_{in} = \|x_i\|$, we have

$$\begin{aligned} d_{ij}^2 &= \|x_i - x_j\|^2 \\ &= \|x_i\|^2 - 2x_i^{\mathrm{T}} x_j + \|x_j\|^2 \\ &= d_{in}^2 - 2x_i^{\mathrm{T}} x_j + d_{jn}^2, \qquad i, j = 1, 2, \ldots, n-1 \end{aligned} \tag{1}$$

- Define $X = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ and $D = \{(d_{in}^2 - d_{ij}^2 + d_{jn}^2)/2 : i, j = 1, 2, \ldots, n-1\}$, $(1) \Rightarrow XX^{\mathrm{T}} = D$.

- Let $D = U\Sigma U^{\mathrm{T}}$, $V = U(:, 1:3)$ and $\Lambda = \Sigma(1:3, 1:3)$. Then $X = V\Lambda^{1/2}$ is the best rank-3 approximation. [Eckart-Young 1936]

Buildup Method

Algorithm 1: Buildup Method for Distance Geometry Problem with sparse data

Given: distance matrix D.

Step 1: Find a clique of four points and determine their coordinates.

Step 2: Choose a point to be added, apply liner or nonlinear least square to locate the point.

Step 3: Repeat Step 2 until all points are determined or no more points can be determined.

♠ It was first proposed by [Dong-Wu 2002]. We will specify details of each step later.

Determine first four points

- Find a clique of four points: we simply exploit greedy method
- How to determine their coordinates?

  Denote $x_i = (u_i, v_i, w_i)^{\mathrm{T}}$, set $x_1 = (0,0,0)^{\mathrm{T}}$ and $x_2 = (d_{12},0,0)^{\mathrm{T}}$.

$$\begin{cases} u_3^2 + v_3^2 = d_{31}^2 \\ (u_3 - u_2)^2 + v_3^2 = d_{32}^2 \end{cases} \Rightarrow \begin{cases} u_3 = (d_{31}^2 - d_{32}^2 + u_2^2)/(2u_2) \\ v_3 = \sqrt{d_{31}^2 - u_3^2}, \quad w_3 = 0 \end{cases}$$

$$\begin{cases} u_4^2 + v_4^2 + w_4^2 = d_{41}^2 \\ (u_4 - u_2)^2 + v_4^2 + w_4^2 = d_{42}^2 \\ (u_4 - u_3)^2 + (v_4 - v_3)^2 + w_4^2 = d_{43}^2 \end{cases} \Rightarrow \begin{cases} u_4 = (d_{41}^2 - d_{42}^2 + u_2^2)/(2u_2) \\ v_4 = \ldots \\ w_4 = \sqrt{d_{41}^2 - u_4^2 - v_4^2} \end{cases}$$

  $v_4 = (d_{42}^2 - d_{43}^2 + (u_4 - u_2)^2 + (u_4 - u_3)^2 + v_3^2)/(2v_3)$

- In the noisy case, the "sqrt" part may cause problems.

Which point to be chosen?

- Computational order
- Greedy strategy: choose the point which has the most known distances to the determined points
- Symmetric Reverse Cuthill-McKee order: MATLAB symrcm
  minimize the bandwidth $= \max\{|i-j| : d_{ij} \neq 0\}$

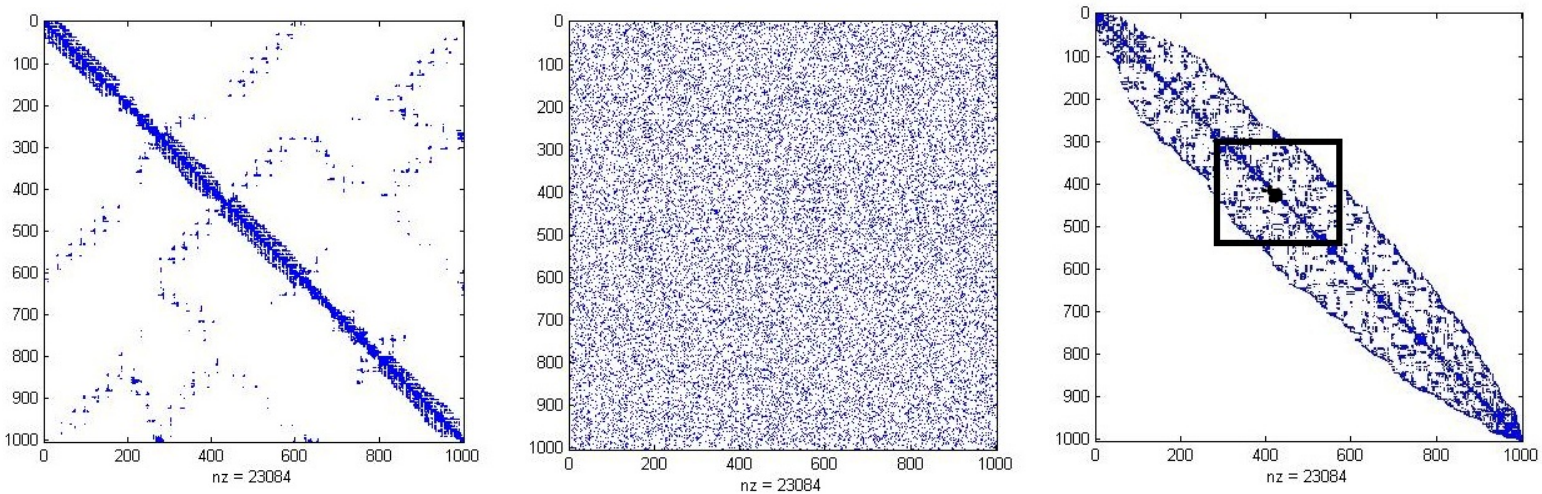


Figure 4 : 1AX8 distance matrix with different orders: original, randperm, symrcm

Determine the chosen point

Suppose x_j (to be determined) has l known distances with $x_i, x_2, \ldots, x_l$ (has been determined).

- linear least square [Wu-Wu 2007]:

  $d_{ij}^2 = \|x_i\|^2 - 2x_i^{\mathrm{T}} x_j + \|x_j\|^2, (i = 1, 2, \ldots, l). \Rightarrow Ax_j = b$, where

$$A = 2 \begin{pmatrix} x_{11} - x_{21} & x_{12} - x_{22} & x_{13} - x_{23} \\ x_{21} - x_{31} & x_{22} - x_{32} & x_{23} - x_{33} \\ \vdots & \vdots & \vdots \\ x_{l-1,1} - x_{l1} & x_{l-1,2} - x_{l2} & x_{l-1,3} - x_{l3} \end{pmatrix},$$

$$b = \begin{pmatrix} (\|x_1\|^2 - \|x_2\|^2) - (d_{1j}^2 - d_{2j}^2) \\ (\|x_2\|^2 - \|x_3\|^2) - (d_{2j}^2 - d_{3j}^2) \\ \vdots \\ (\|x_{l-1}\|^2 - \|x_l\|^2) - (d_{l-1,j}^2 - d_{lj}^2) \end{pmatrix}$$

  ♠ In the noisy case, solve $\min_{x_j} \|Ax_j - b\|_2$.

Determine the chosen point

Suppose x_j (to be determined) has l known distances with $x_1, x_2, \ldots, x_l$ (has been determined).

- nonlinear least square [Sit-Wu-Yuan 2009]:
  1. Calculate missing distances among these l points.
  2. All the distances among $l+1$ points are known, we apply matrix decomposition method to solve it.
  3. Move these points back to the original reference system, using $x_i, x_2, \ldots, x_l$ as common points.

♣ In this way, we determine the new point, while re-determine the l points, which can be viewed as adjustment.

Remarks about Buildup method

- Buildup method is almost perfect when the given distances are exact, as it is fast and accurate.
- Due to accumulation of round error, the result may damage if the number of the points is large (more than several thousands).
- Buildup method with nonlinear technique is more stable. However, it still can tolerate very small noise, usually no bigger than 0.01%.

Outline

Motivation

Two facts:

- It is almost impossible to find the global minimizer or a good local minimizer of any error function, from a randomly chosen initial point.
- Many existing methods can be casted into two parts:
  1) Find a good initial point:
     Embedding Method [Crippen-Havel 1988] trys to estimate the missing data, and then apply matrix decomposition method to obtain a solution.
     SDP Method [Biswas-Toh-Ye 2007] solves a relaxed SDP problem to produce an initial point.
  2) Minimize an error function to adjust the positions.
     remark: these methods don't talk too much about this procedure, but actually it is very crucial to their methods.

Our idea

- Incorporate Buildup Method and error function minimization.
- remark:

  Buildup method is extremely fast, which is its most charming feature, but it cannot tolerate large noises, so we try to apply error minimization procedure to make it stable.

How we come to the final algorithm

- At first, we try to adjust the recent M points after every buildup step, so we choose **symrcm** order to implement buildup method.
  much better, still not so stable
- Then, we add the I points to this procedure.
  even better, but takes more time
- Finally, I only use the I points, and choose greedy order to implement buildup procedure.
  fantastic!

Our algorithm

Algorithm 2: Buildup-based Error Minimization Method for Distance Geometry Problem with sparse and noisy distances

Given: distance matrix D.

Step 1: Find a clique of four points and determine their coordinates.

Step 2: Choose a point to be added, apply nonlinear least square to locate the point.

Step 3: Apply error minimization to these $l + 1$ points.

Step 4: Repeat Step 2-3 until all points are determined or no more points can be determined.

Step 5: Apply error minimization to all the determined points.

Error function minimization

Currently, we simply use line search BB method to minimize the following error functions,

$$Stress(x_1, x_2, \ldots, x_n) = \sum_{(i,j)\in S} (\|x_i - x_j\| - d_{ij})^2$$

or

$$Yuan(x_1, x_2, \ldots, x_n) = \sum_{(i,j)\in S} h(\frac{\|x_i - x_j\|}{d_{ij}})$$

where

$$h(x) = \begin{cases} \frac{1}{2}(x-1)^2, & \text{if } x > 1, \\ x - 1 - log(x), & \text{if } 0 < x < 1. \end{cases}$$

- The effect are more or less the same.
- This is the most time-consuming part in our algorithm, which need to be further studied in the future work.

Analysis of Stress function

Theorem 3.1 (Stress function)

Suppose

$$f(x_1, x_2, \ldots, x_n) = \sum_{(i,j)\in S} \frac{1}{2}(\|x_i - x_j\| - d_{ij})^2,$$

let $N(i) = \{j : d_{ij} \neq 0\}$ *denote the neighbours of point* i, *and* $x = (x_1; x_2; \ldots; x_n)$ *is a column vector in* $\mathbb{R}^{3n}$, *then we have*

$$\frac{\partial f}{\partial x_i} = \sum_{j\in N(i)} (1 - \frac{d_{ij}}{\|x_i - x_j\|})(x_i - x_j),$$

and for $k = 1, 2, 3$,

$$\frac{\partial^2 f}{\partial x_{ik}\partial x_{jk}} = \begin{cases} \sum_{j\in N(i)} (\frac{d_{ij}(x_{ik}-x_{jk})^2}{\|x_i - x_j\|^3} + 1 - \frac{d_{ij}}{\|x_i - x_j\|}), & if\ i = j, \\ -(\frac{d_{ij}(x_{ik}-x_{jk})^2}{\|x_i - x_j\|^3} + 1 - \frac{d_{ij}}{\|x_i - x_j\|}), & if\ j \in N(i), \\ 0, & otherwise. \end{cases}$$

Then, 0 and $(1, 1, \ldots, 1)^{\mathrm{T}}$ *is an eigenpair. Furthermore, Hessian matrix is positive semidefinite at optimal point, and can be negative semidefinite otherwise.*

Analysis of Yuan function

Theorem 3.2 (Yuan function)

Suppose

$$
\begin{aligned}
f(x) &= \sum_{(i,j)\in S} h\left(\frac{\|x_i - x_j\|}{d_{ij}}\right) \\
&= \sum_{(i,j)\in S} \begin{cases} \frac{1}{2}\left(\frac{\|x_i - x_j\|}{d_{ij}} - 1\right)^2, & if\ \|x_i - x_j\| \geq d_{ij}, \\ \frac{\|x_i - x_j\|}{d_{ij}} - 1 - log\left(\frac{\|x_i - x_j\|}{d_{ij}}\right), & otherwise \end{cases}
\end{aligned}
$$

the symbols are the same as those in the previous theorem, then we have

$$
\frac{\partial f}{\partial x_i} = \sum_{j\in N(i)} \begin{cases} \left(\frac{1}{d_{ij}^2} - \frac{1}{d_{ij}\|x_i - x_j\|}\right)(x_i - x_j), & if\ \|x_i - x_j\| \geq d_{ij}, \\ \left(\frac{1}{d_{ij}\|x_i - x_j\|} - \frac{1}{\|x_i - x_j\|^2}\right)(x_i - x_j), & otherwise \end{cases},
$$

to be continued...

Analysis of Yuan function(cont'd)

Theorem 3.3 (Yuan function)

and for $k = 1, 2, 3,$

$$\frac{\partial^2 f}{\partial x_{ik} \partial x_{jk}} = \begin{cases} \sum_{j \in N(i)} \omega(x), & if\ i = j, \\ -\omega(x), & if\ i \neq j, \\ 0, & otherwise \end{cases}$$

where

$$\omega(x) = \begin{cases} \frac{1}{d_{ij}^2} - \frac{1}{d_{ij}\|x_i - x_j\|} + \frac{(x_{ik} - x_{jk})^2}{d_{ij}\|x_i - x_j\|^3}, & if\ \|x_i - x_j\| \geq d_{ij}, \\ \frac{1}{d_{ij}\|x_i - x_j\|} - \frac{1}{\|x_i - x_j\|^2} - \frac{(x_{ik} - x_{jk})^2}{d_{ij}\|x_i - x_j\|^3} + \frac{(x_{ik} - x_{jk})^2}{\|x_i - x_j\|^4}, & otherwise. \end{cases}$$

Then, 0 and $(1, 1, \ldots, 1)^{\mathrm{T}}$ *is an eigenpair. Furthermore, Hessian matrix is positive semidefinite at optimal point, and can be negative semidefinite otherwise.*

Proof. The positive semidefinite property follows by Gershgorin Circle Theorem.

Why we are good?

- We re-use the same distance information (but only the given distances, not include calculated ones).
- In Buildup method equipped with nonlinear least square, if distances are noisy, then we have

$$\|x_i - x_j\| = d_{ij} + e_{ij},$$

where e_{ij} denotes noise.

$$\Rightarrow 2x_i^{\mathrm{T}} x_j = d_{in}^2 - d_{ij}^2 + d_{jn}^2 - 2d_{ij}e_{ij} - e_{ij}^2$$

- So, my conclusion is:
  When the noises are large, $(d_{ij}e_{ij} + e_{ij}^2/2)$ should not be ignored, namely, solving $X^{\mathrm{T}}X = D$ cannot give a very good solution to the original problem. But it can serve as a warm starting point to minimize an error function to obtain a better solution.

Outline

Numerical experiments

We simulate on real protein structures, but the distance matrix is generated by us.

1. Download PDB file from Internet to get the true positions of atoms.
2. Use *disk graph model* to generate distance matrix from these coordinates, with different cutoffs (6Å or 5Å) and variant noise level.

$$d_{ij} = d_{ij}(1 + noise * randn)$$

3. Make numerical experiments with these distance matrices. Distances are the only information we need to implement our algorithm.
4. Examine the RMSD error of calculated positions with the true positions.

$$RMSD(X, Y) = \min_{Q,T}\{\|X - YQ - T\|_F/\sqrt{n} : Q^{\mathrm{T}}Q = I\}$$

ID	num	per	degree			RMSD	fval	time(s)			ndet
			max	min	ave			before	post	total	
1PTQ	402	5.46	38	4	21.9	1.21e-001	4.82	0.9	0.4	1.3	402
1HOE	558	4.05	38	6	22.6	4.97e-002	6.65	1.2	0.6	1.8	558
1LFB	641	3.40	40	5	21.8	1.23e-001	6.94	1.5	0.9	2.4	641
1PHT	811	3.35	48	5	27.1	7.46e-002	12.68	3.5	1.5	5.0	806
1POA	914	2.51	39	4	22.9	7.83e-002	11.32	3.4	1.4	4.8	914
1AX8	1003	2.30	39	5	23.0	9.55e-002	13.09	4.3	1.5	5.9	1003
1F39	1534	1.47	40	5	22.6	1.31e-001	19.10	9.8	2.5	12.3	1534
1RGS	2015	1.12	41	3	22.6	2.86e-001	25.65	20.4	7.0	27.4	2010
1KDH	2846	0.83	43	4	23.6	1.46e-001	38.44	64.6	24.2	88.8	2846
1BPM	3671	0.66	42	3	24.4	2.47e-001	53.01	93.9	8.6	102.5	3668
1RHJ	3740	0.65	40	4	24.4	9.07e-002	52.71	106.8	5.8	112.6	3740
1HQQ	3944	0.60	40	3	23.7	6.51e+000	4147.21	128.0	36.8	164.8	3938
1TOA	4292	0.56	39	3	24.0	1.02e-001	58.24	148.4	11.9	160.2	4280
1MQQ	5681	0.44	44	5	25.2	3.85e+000	3277.59	350.1	57.4	407.6	5681

Table 1 : cutoff=5Å, noise=0.01

ID	num	per	degree			RMSD	fval	time(s)			ndet
			max	min	ave			before	post	total	
1PTQ	402	8.79	61	6	35.3	3.97e-002	12.07	2.8	0.6	3.4	402
1HOE	558	6.55	65	11	36.5	3.32e-002	17.11	4.1	0.8	4.9	558
1LFB	641	5.57	59	8	35.7	5.02e-002	19.77	8.3	3.7	11.9	641
1PHT	811	5.37	75	7	43.5	4.97e-002	31.24	9.5	1.4	10.9	811
1POA	914	4.07	67	8	37.2	4.90e-002	29.95	8.5	1.6	10.1	914
1AX8	1003	3.74	59	7	37.5	3.77e-002	32.31	9.6	1.8	11.4	1003
1F39	1534	2.43	62	7	37.2	4.26e-002	50.51	23.6	2.4	25.9	1534
1RGS	2015	1.87	66	4	37.7	5.51e-002	68.05	34.6	3.1	37.8	2015
1KDH	2846	1.36	64	5	38.8	7.75e-002	98.18	69.8	4.6	74.4	2846
1BPM	3671	1.12	64	4	40.9	4.55e-002	136.34	135.2	10.3	145.4	3671
1RHJ	3740	1.10	61	5	41.2	3.50e-002	140.69	143.6	6.5	150.2	3740
1HQQ	3944	1.00	64	5	39.5	7.31e-002	140.46	167.1	11.4	178.4	3944
1TOA	4292	0.94	62	4	40.1	6.01e-002	155.17	195.9	11.1	206.9	4292
1MQQ	5681	0.75	66	7	42.4	3.25e-002	219.44	397.1	17.5	414.6	5681

Table 2 : cutoff=6Å, noise=0.01

ID	num	per	degree			RMSD	fval	time(s)			ndet
			max	min	ave			before	post	total	
1PTQ	402	8.79	61	6	35.3	2.35e-001	300.77	5.0	1.3	6.3	402
1HOE	558	6.55	65	11	36.5	1.57e-001	426.42	9.5	1.5	11.0	558
1LFB	641	5.57	59	8	35.7	2.12e-001	494.55	21.0	12.1	33.1	641
1PHT	811	5.37	75	7	43.5	8.77e-001	784.13	36.4	7.5	43.9	811
1POA	914	4.07	67	8	37.2	4.88e-001	772.20	13.5	8.3	21.8	914
1AX8	1003	3.74	59	7	37.5	2.10e-001	811.68	15.6	15.5	31.1	1003
1F39	1534	2.43	62	7	37.2	2.06e-001	1264.26	30.7	10.5	41.2	1534
1RGS	2015	1.87	66	4	37.7	2.79e-001	1689.98	47.0	16.3	63.3	2015
1KDH	2846	1.36	64	5	38.8	3.38e-001	2435.99	84.4	23.6	108.1	2846
1BPM	3671	1.12	64	4	40.9	2.07e-001	3406.64	155.4	22.8	178.2	3671
1RHJ	3740	1.10	61	5	41.2	3.35e+000	3389.34	286.0	93.8	379.9	3740
1HQQ	3944	1.00	64	5	39.5	2.93e-001	3507.34	262.3	216.6	478.9	3944
1TOA	4292	0.94	62	4	40.1	1.84e-001	3870.54	222.1	30.3	252.4	4292
1MQQ	5681	0.75	66	7	42.4	1.61e-001	5484.36	450.2	29.9	480.1	5681

Table 3 : cutoff=6Å, noise=0.05

Table 2: Results with 70% of distances below 6Å and 5% noise on upper and lower bounds and with 50% of distances below 6Å and 1% noise on upper and lower bounds

PDB ID	No. of atoms	70% Distances, 5% Noise		50% Distances, 1% Noise	
		RMSD(Å)	CPU time (secs)	RMSD(Å)	CPU time (secs)
1PTQ	402	0.2794	93.8	0.7560	22.1
1HOE	558	0.2712	129.6	0.0085	32.5
1LFB	641	0.4392	132.5	0.2736	41.6
1PHT	814	0.4701	129.4	0.6639	53.6
1POA	914	0.4325	174.7	0.0843	54.1
1AX8	1003	0.8184	251.9	0.0314	71.8
1F39	1534	1.1271	353.1	0.2809	113.4
1RGS	2015	4.6540	613.3	3.5416	308.2
1KDH	2923	2.5693	1641.0	2.8222	488.4
1BPM	3672	2.4360	1467.1	1.0502	384.8
1RHJ	3740	0.9535	1286.1	0.1158	361.5
1HQQ	3944	8.9106	2133.5	1.6610	418.4
1TOA	4292	9.8351	2653.6	1.5856	372.6
1MQQ	5681	3.1570	1683.4	2.3108	1466.2

Figure 5 : Results in [Biswas-Toh-Ye 2007]

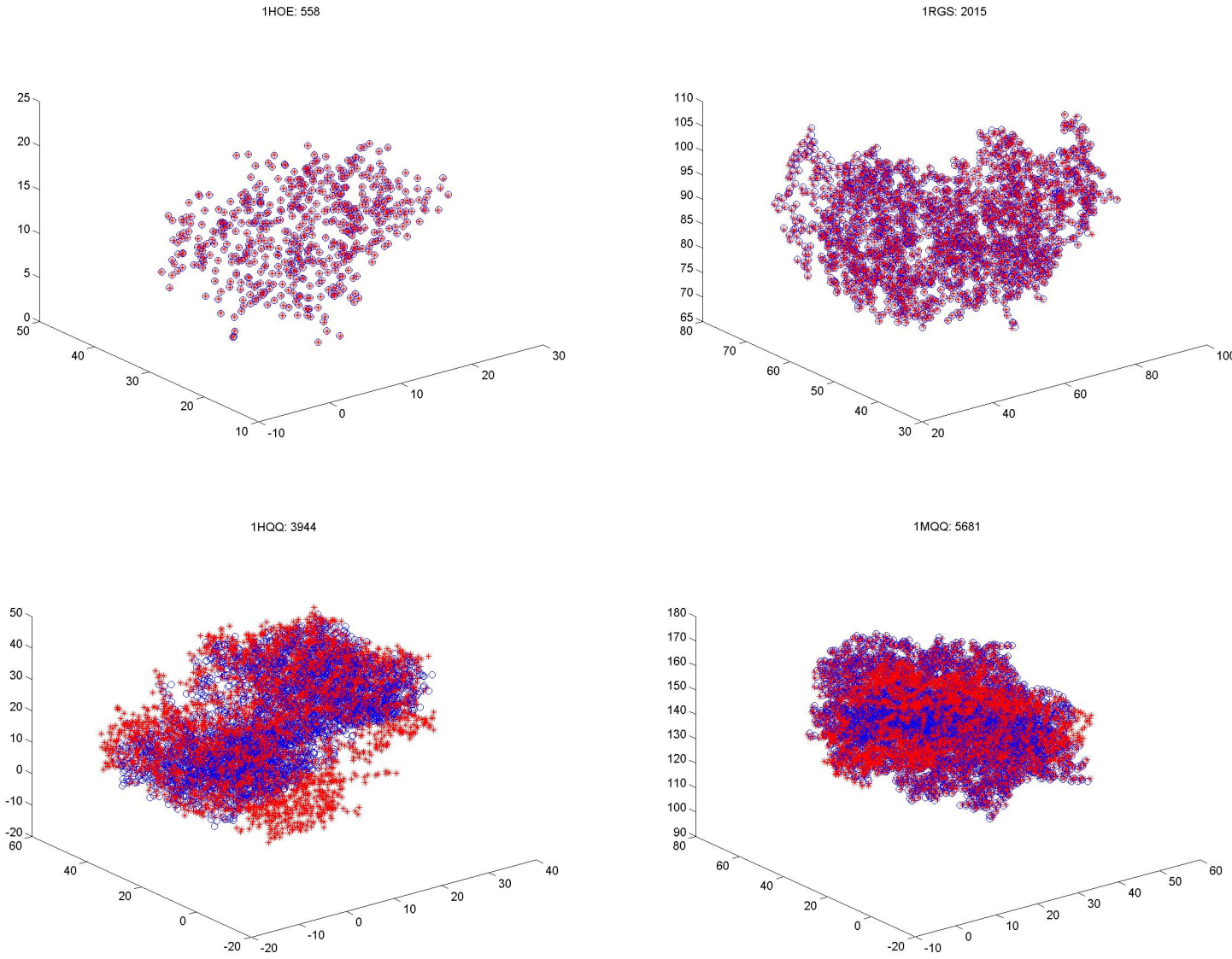


Figure 6 : cutoff=5Å, noise=0.01

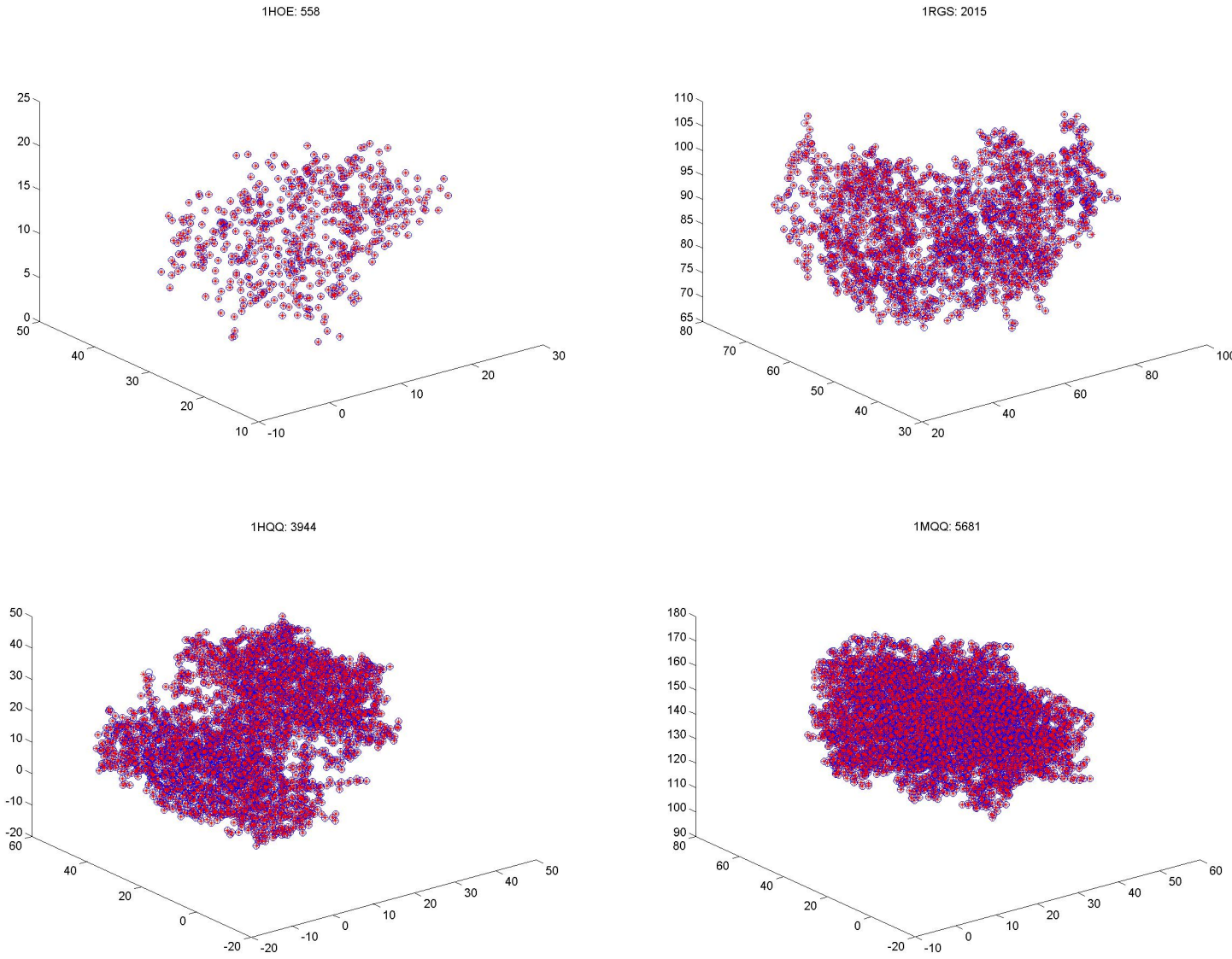


Figure 7 : cutoff=6Å, noise=0.01

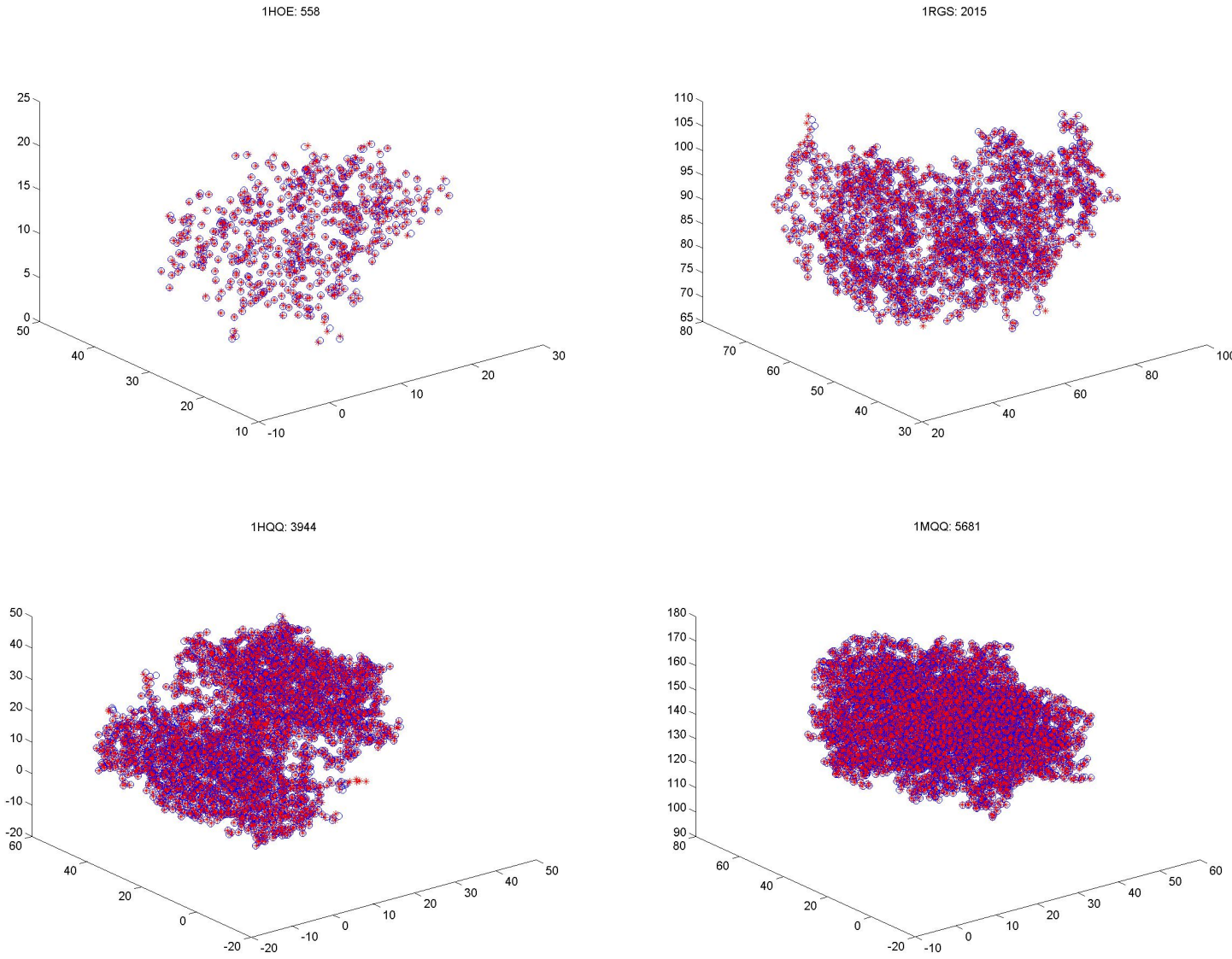
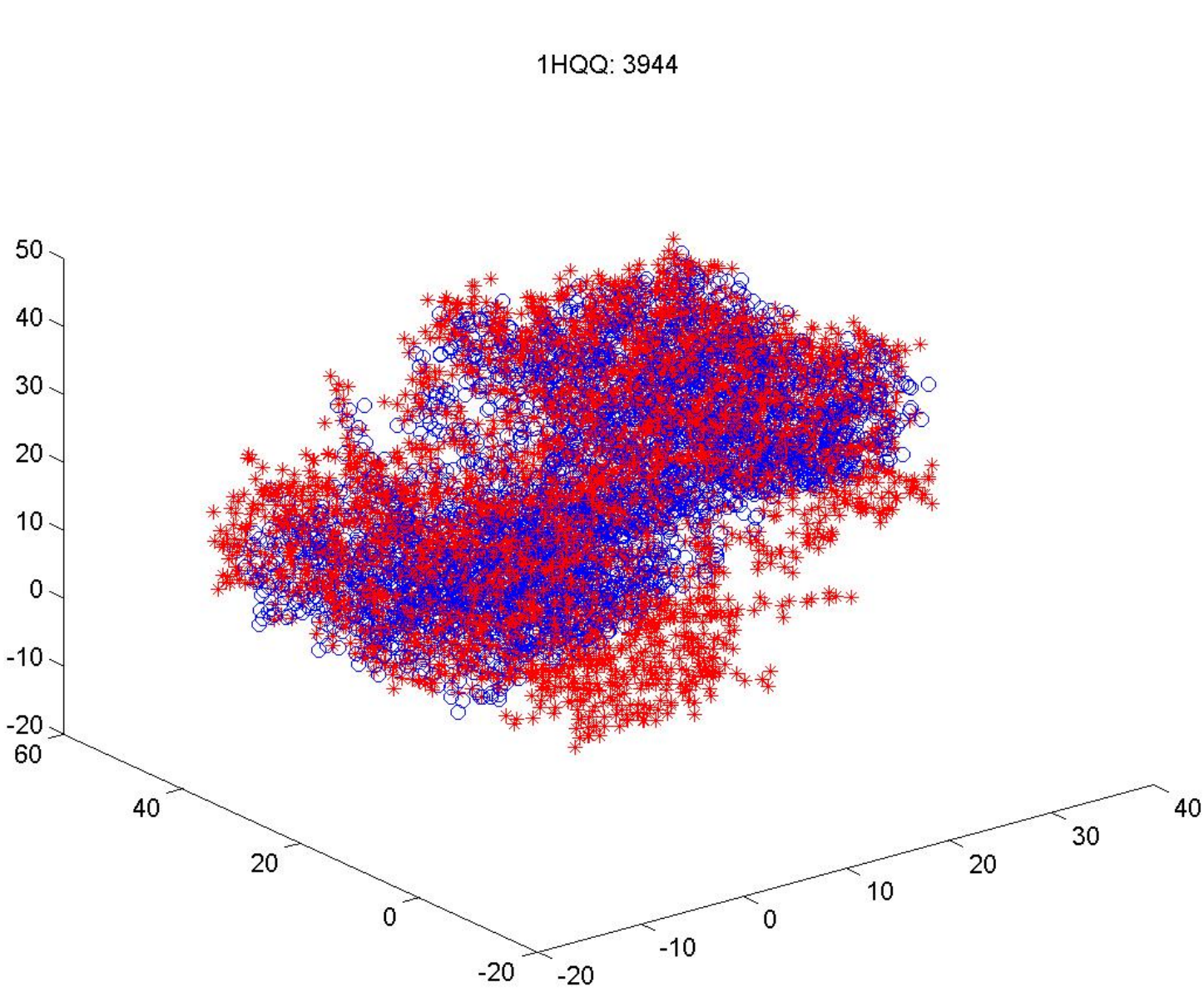
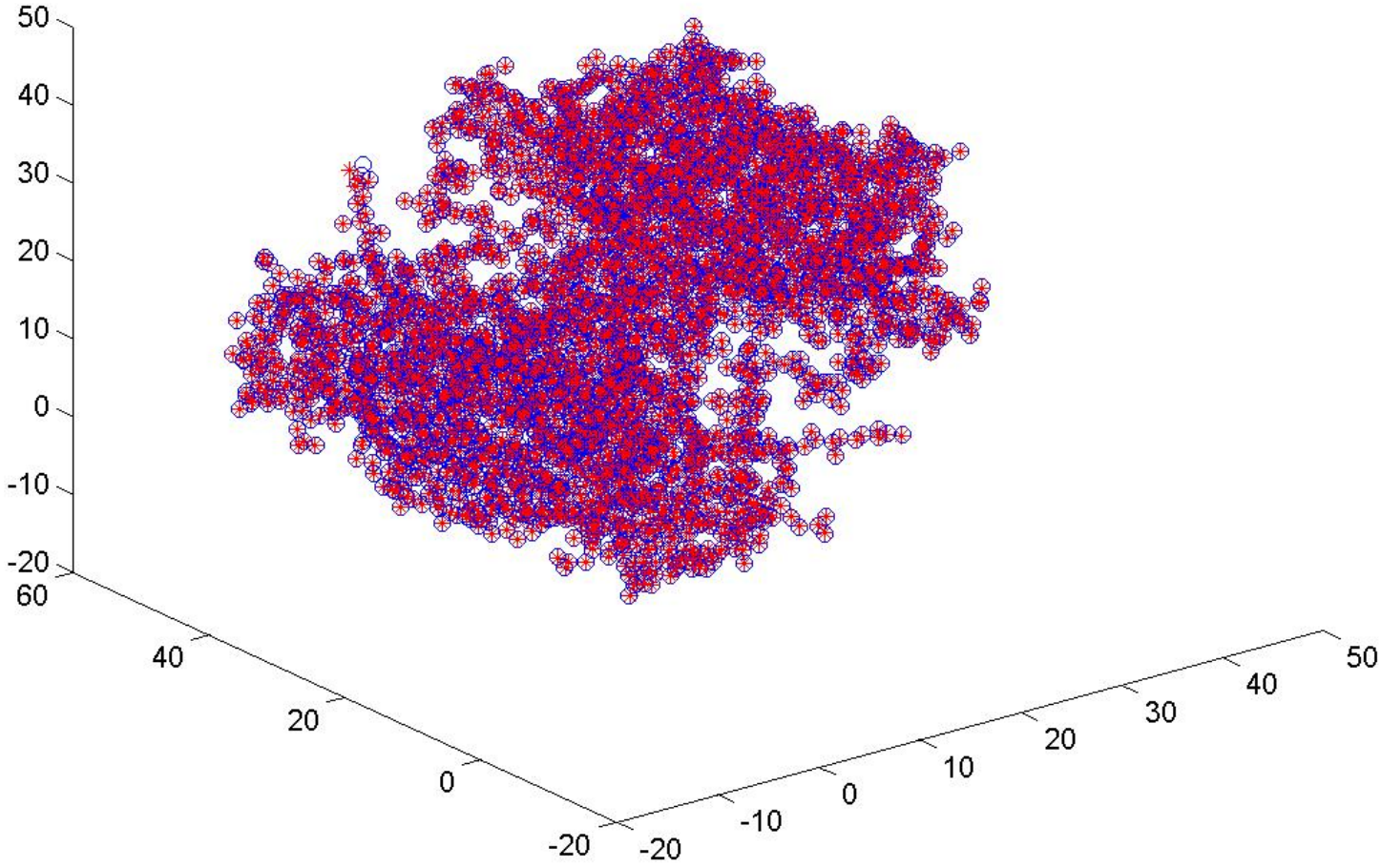
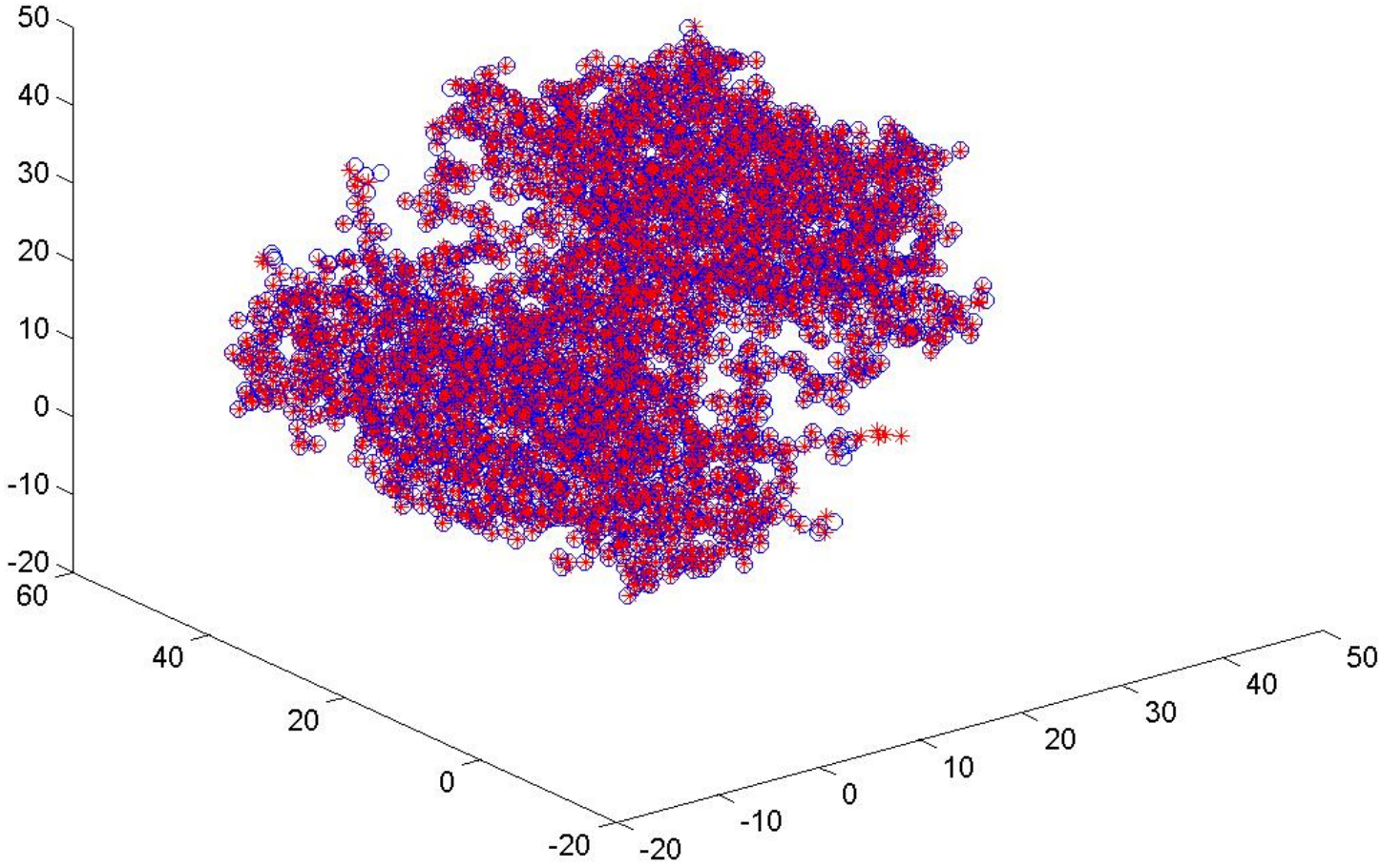


Figure 8 : cutoff=6Å, noise=0.05


1HQQ: 3944
50
40
30
20
10
0
-10
-20
60
40
20
0
-20
-20
-10
0
10
20
30
40


1HQQ: 3944
50
40
30
20
10
0
-10
-20
60
40
20
0
-20
-20
-10
0
10
20
30
40
50


1HQQ: 3944
50
40
30
20
10
0
-10
-20
60
40
20
0
-20
-20
-10
0
10
20
30
40
50

Outline

Ongoing works

- Further study the properties of error function, then design a fast algorithm.
- Extend our algorithm to solve distance geometry problem with distance bounds.
- Produce an animation to see how these points were built up one by one.

Thank you for your attention!

szl@lsec.cc.ac.cn