A New Error Function and Its Application to Distance Geometry Problem

Zhenli Sheng (szl@lsec.cc.ac.cn)
Institute of Computational Mathematics and Scientific/Engeering Computing,
Academy of Mathematics and Systems Science, Chinese Academy of Sciences

Brief Introduction

Distance Geometry Problem is to find the coordinate vectors $x_1, x_2, \ldots, x_n$ that satisfy several given distances between them. Mathematically, Find $x_1, x_2, \ldots, x_n$, such that

$$\|x_i - x_j\| = d_{i,j}, \quad (i,j) \in S.$$
$$\text{or} \quad l_{i,j} \le \|x_i - x_j\| \le u_{i,j}, \quad (i,j) \in S.$$

Motivation

Similar to Hooke's law, we construct force function:

$$F(x) = \begin{cases} x - 1, & x \ge 1, \\ 1 - \frac{1}{x}, & x < 1. \end{cases}$$

which prevents the spring from the stationary point $x = 1$, then the energy function is h(x).

Solution Idea

Alternative Direction Method

- Fix the others, adjust one point each time.
- Go Newton's step or solve a small trust region subproblem at each iteration.

Gradient Method

Search direction can be negative gradient or the direction proposed in [4].

Stepsize can be chosen as alternative BB step considered in [2]:

$$\alpha_k^{ABB} = \begin{cases} \alpha_k^{BB1} = \frac{\|s_{k-1}\|^2}{s_{k-1}^T y_{k-1}}, & \text{for odd k}, \\ \alpha_k^{BB2} = \frac{s_{k-1}^T y_{k-1}}{\|y_{k-1}\|^2}, & \text{for even k}. \end{cases}$$

or by nonmonotone line search in [3]:

$$f(x_k + \alpha_k d_k) \le C_k + \delta \alpha_k \nabla f(x_k)^T d_k$$

where $\alpha_k = \overline{\alpha}_k \rho^{h_k}$ and h_k is the largest integer such that the above inequality holds. C_k is chosen as a convex combination of all the previous function values.

Starting point: Geometric Buildup Method in [1] is very fast but accumulation of round error may ruin the result when the number of the points is large. However, it can be used as a warm starting point.

References

[1] A. Sit, Z. Wu and Y. Yuan(2009), *A geometric buildup algorithm for the solution of the distance geometry problem using least-squares approximation.*

[2] Y. Dai and R. Fletcher(2003), *Projected Barzilar-Borwein method for large-scale box-caonstrained quadratic programming.*

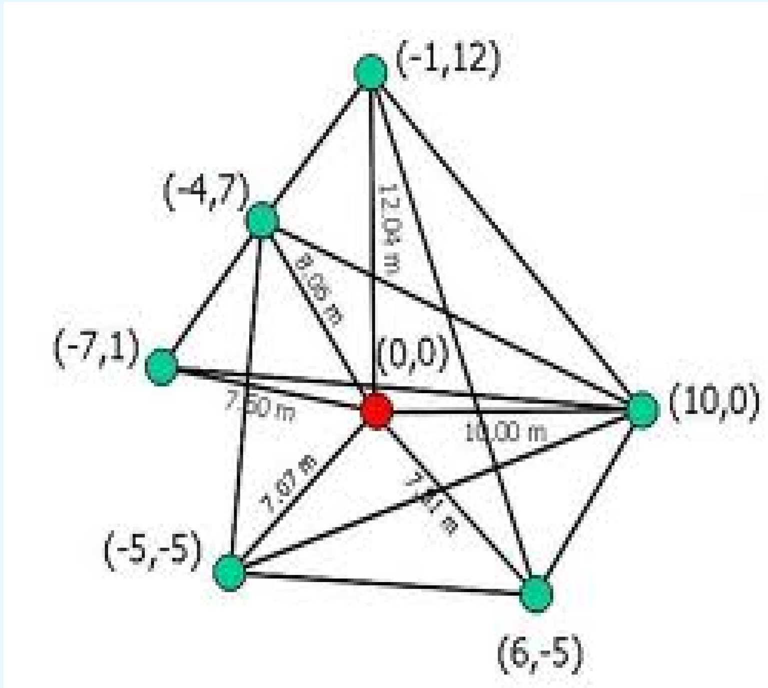
[3] H. Zhang and W. Hager(2004), *A nonmonotone line search technique and its application to unconstrained optimization.*

[4] D. Liu and J. Nocedal (1989), *On the limited memory BFGS method for large scale optimization.*

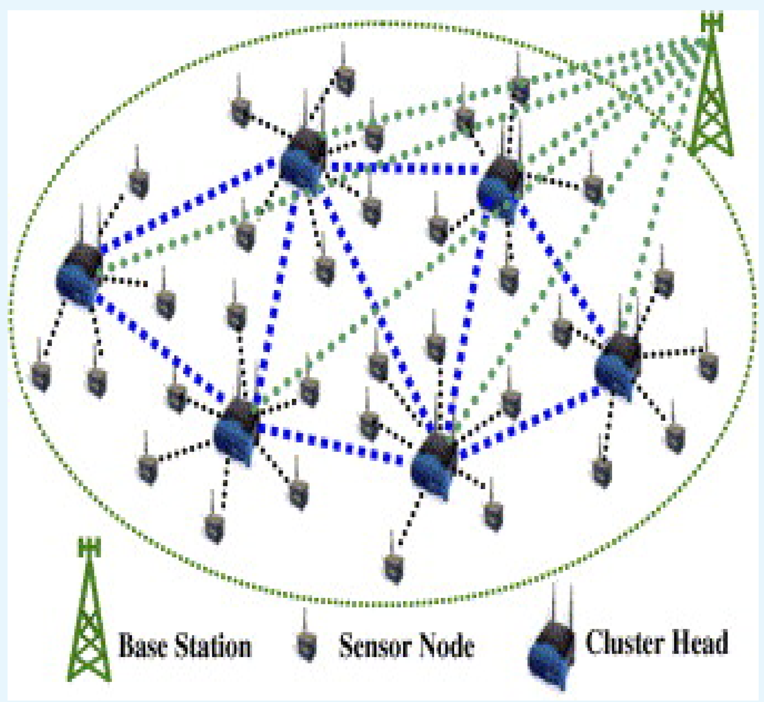
Acknowledgements

Thank my supervisor Prof. Ya-xiang Yuan for the useful advice to my research and foundation support for me to attend this workshop, also Dr. Xin Liu for the helpful discussion.

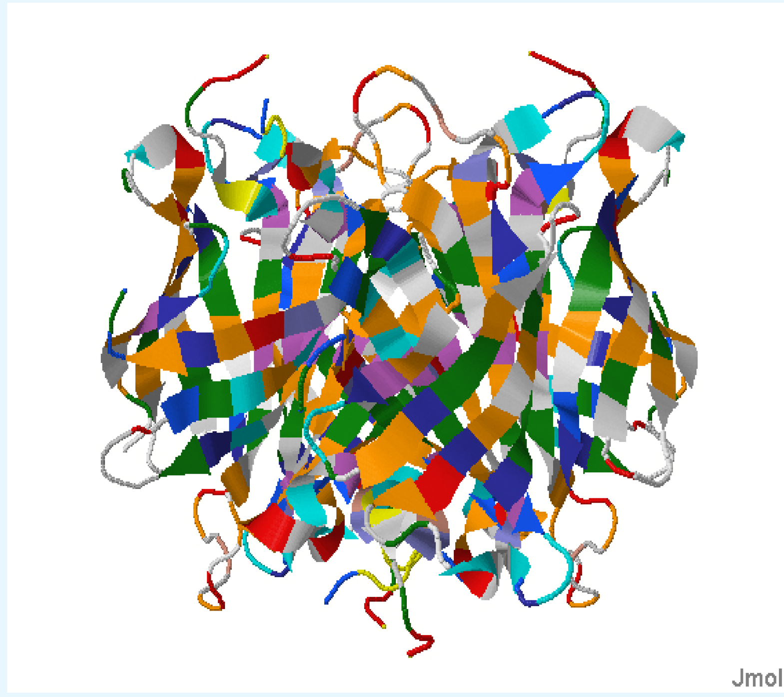
Applications



Graph Realization



Sensor Network Localization



Protein Structure Determination

Error Functions

Traditional error functions:

- stress function: $\sum_{(i,j)\in S}(\|x_i - x_j\| - d_{i,j})^2$,
- smoothed stress function: $\sum_{(i,j)\in S}(\|x_i - x_j\|^2 - d_{i,j}^2)^2$,
- generalized stress function: $\sum_{(i,j)\in S} \min^2\{\frac{\|x_i - x_j\|^2 - l_{i,j}^2}{l_{i,j}^2}, 0\} + \max^2\{\frac{\|x_i - x_j\|^2 - u_{i,j}^2}{u_{i,j}^2}, 0\}$.

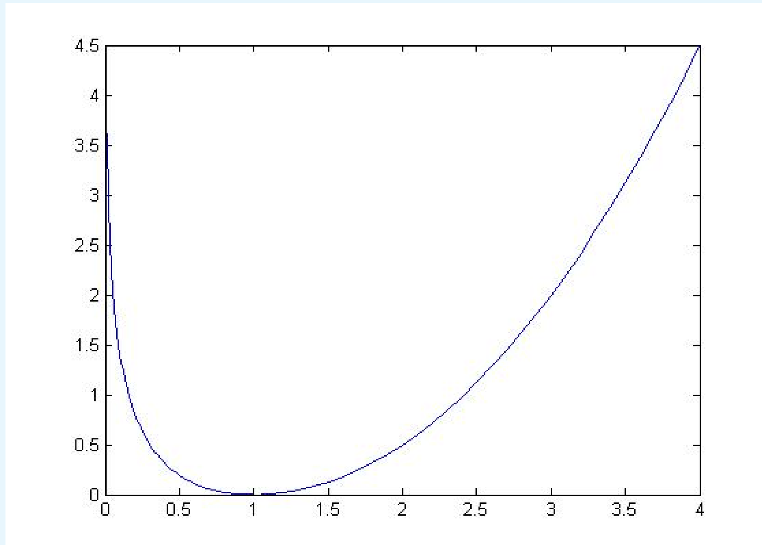
All these functions tend to have too many local minimizers.

Our proposed error function:

Define h: $\mathbb{R}_{++} \to R$ as below,

$$h(x) = \begin{cases} \frac{1}{2}(x-1)^2, & x \ge 1, \\ x - (1 + ln(x)), & x < 1. \end{cases}$$

- h(x) is twice continuously differentiable in $(0, +\infty)$, and it achieves its minimum 0 at 1.
- The error function is $\sum_{(i,j)\in S} h(\frac{\|x_i - x_j\|^2}{d_{i,j}^2})$.



Algorithm Framewaork

Algorithm 1: Trust Region Error Minimization Method

Initialization: Choose starting points (or calculate by Buildup) and set the parameters
while *stopping criteria not satisfied* **do**
  for *i=1:n* **do**
    Solve trust region subproblem to obtain trial step s_i, let $r_i = \frac{\overline{f}(x_i) - \overline{f}(x_i + s_i)}{q(x_i) - q(x_i + s_i)}$.
    According to r_i to determine to accept s_i or not, and adjust Δ_i;

Algorithm 2: Alternative BB/Nonmonotone Line Search Method

Initialization: Choose starting points (or calculate by Buildup) and set the parameters
while *stopping criteria not satisfied* **do**
  Calculate search direction d_k and stepsize α_k; Set $x_k \leftarrow x_k + \alpha_k d_k, k \leftarrow k+1$.

Numerical Results