Computational-Efficient Iterative TDOA Localization Scheme Using a Simplified Multidimensional Scaling-Based Cost Function

Liming Li\textsuperscript{1,2}, Baogang Ding\textsuperscript{1}, Rui Sun\textsuperscript{1} and Wuyang Jiang\textsuperscript{1}

\textsuperscript{1}College of Urban Railway Transportation, Shanghai University of Engineering Science, Shanghai 201620, China
\textsuperscript{2}College of Information Science and Technology, Donghua University, Shanghai 201620, China

*Email: jiangwy_sues@126.com

Abstract. This work deals with source localization with time-difference-of-arrival (TDOA) measurements. Recently, a multidimensional scaling (MDS)-based iterative localization scheme is introduced in the literature, where an MDS-based cost function is defined as the norm of the difference matrix between two scalar product matrices. The minimizer of the MDS-based cost function which is computed by Newton’s iteration is considered as the estimate of the source position. However, the scalar product matrices of the MDS-based cost function are of high order, which need a lot of computations in each step of the Newton’s iteration. In this paper, a computational-efficient iterative localization scheme is proposed. In the proposed scheme, a simplified MDS-based cost function is constructed from two low-order matrices that are converted from the high-order scalar product matrices by a few steps of Lanczos iteration, and then the Newton’s iteration is applied to find the minimizer of the simplified MDS-based cost function. Simulation results show that the localization accuracy of the proposed scheme is nearly the same as that of the original scheme, whereas the computational complexity of the proposed scheme is about 20\% as much as that of the original scheme.

1. Introduction

It is important to locate a single passive source through time-difference-of-arrival (TDOA) measurements from an array of spatially separated sensors at known locations in radar, sonar, multimedia, wireless sensor networks, and wireless communications [1]. Usually, the localization problem is converted to the minimization problem of a specifically defined cost function, where the minimizer of the cost function is considered as the estimate of the source position. The existing cost functions include the hyperbolic least squares cost function [2], the spherical least squares cost function [3-5], the squared range-based least squares cost function [6], the squared range difference weighted least squares cost function [7], and so on.

Recently, a new cost function for TDOA localization has been introduced, which is defined as the norm of the difference matrix between two scalar product matrices in the multidimensional scaling (MDS) framework [8-14]. The minimizer of the MDS-based cost function can be computed by the Newton’s iteration [15] as long as the derivatives of the MDS-based cost function are computed. However, since the order of the scalar product matrices is the same as the sensor number, the
computational complexity of computing the derivatives of the MDS-based cost function will be very high when the sensor array contains large number of sensors.

In this paper, a computational-efficient iterative TDOA localization scheme using a simplified MDS-based cost function is proposed. The proposed scheme is based on the subspace property of the scalar product matrix that, by several steps of Lanczos iteration, the scalar product matrix can be converted to a low-order matrix whose order is no more than 6. Based on the subspace property, a simplified MDS-based cost function is defined using the low-order matrices. And then, the derivatives of the simplified MDS-based cost function are computed. Finally, the Newton’s iteration is used to compute the minimizer of the simplified MDS-based cost function. Simulation results show that the localization accuracy of the proposed MDS-based iterative localization scheme is nearly the same as that of the original MDS-based iterative localization scheme, whereas the computational complexity of the proposed MDS-based iterative localization scheme is about 20% as much as that of the original MDS-based iterative localization scheme.

The notations used in this paper are defined as follows. Boldface lowercase letter \( \mathbf{a} \) and boldface uppercase letter \( \mathbf{A} \) represent vector and matrix, respectively. \( \mathbf{I} \) and \( \mathbf{\theta} \) stand for \( N \times N \) dimensional column vectors of all ones and all zeros, respectively. \( \mathbf{E} \) stands for \( N \times N \) identity matrix. \( \mathbf{O} \) stands for zero matrix. \( \text{diag}(a_1, \ldots, a_n) \) stands for diagonal matrix whose entries are \( a_1, \ldots, a_n \). \( \| \| \) stands for Euclidean norm of a vector. Finally, \( \text{rank}(\cdot), \text{tr}(\cdot), (\cdot)^\top \), and \( \| \|_p \) stand for rank, trace, transpose, and Frobenius norm of a matrix, respectively.

2. System Model

Consider an array of \( M \geq 6 \) sensors and a single source in a 3-dimensional (3-D) space. The positions of the sensors are known, denoted by \( \mathbf{s}_m = [x_m, y_m, z_m]^\top \), \( m = 1, \ldots, M \). Assign the first sensor as the reference. The position of the source is unknown, denoted by \( \mathbf{u} = [x, y, z]^\top \), where the true position of the source is denoted by \( \mathbf{u}_0 = [x_0, y_0, z_0]^\top \). Denote the range between the \( m \)-th sensor and the source by \( d_m = \| \mathbf{s}_m - \mathbf{u} \| \), where the true range is denoted by \( d_m^0 = \| \mathbf{s}_m - \mathbf{u}_0 \| \). Denote the range difference between the \( m \)-th range and the 1-th range by \( d_{m1} = d_m - d_1 \), where the true range difference is denoted by \( d_{m1}^0 = d_m^0 - d_1^0 \). Denote the vector of ranges by \( \mathbf{d} = [d_1, \ldots, d_M]^\top \), where the vector of true ranges is denoted by \( \mathbf{d}^0 = [d_1^0, \ldots, d_M^0]^\top \). Denote the x-coordinate vector, y-coordinate vector, and z-coordinate vector of sensors by \( \mathbf{x} = [x_1, \ldots, x_M]^\top \), \( \mathbf{y} = [y_1, \ldots, y_M]^\top \), \( \mathbf{z} = [z_1, \ldots, z_M]^\top \), respectively. Denote \( \mathbf{D} = \text{diag}(1, 1, 1, -1) \). Define the position coordinates matrix by \( \mathbf{Z} = [\mathbf{x} - M \mathbf{I}_M, \mathbf{y} - M \mathbf{I}_M, \mathbf{z} - M \mathbf{I}_M, \mathbf{d}^0]^\top \), where the true position coordinates matrix is denoted by \( \mathbf{Z}_0 = [\mathbf{x} - M \mathbf{I}_M, \mathbf{y} - M \mathbf{I}_M, \mathbf{z} - M \mathbf{I}_M, \mathbf{d}^0]^\top \). Define the scalar product matrix by \( \mathbf{B} = \mathbf{Z}^\top \mathbf{D} \mathbf{Z} \), where the true scalar product matrix is denoted by \( \mathbf{B}_0 = \mathbf{Z}_0^\top \mathbf{D} \mathbf{Z}_0 \).

By multiplying the signal propagation speed, the TDOA measurement converts to the range difference measurement \( \hat{d}_{m1} \), which is modelled as \( \hat{d}_{m1} = d_{m1}^0 + q_m \), \( m = 1, \ldots, M \), where \( q_m, m = 2, \ldots, M \) is the measurement noise of the range difference, and \( q_1 = 0 \). Assume \( \mathbb{E}[q_m] = 0, m = 2, \ldots, M \). Denote \( \mathbf{q} = [q_1, \ldots, q_M]^\top \) and \( \mathbf{Q} = \mathbb{E}[\mathbf{qq}^\top] \in \mathbb{R}^{M \times M} \).

According to the definitions, \( \mathbf{B} = \mathbf{Z}^\top \mathbf{D} \mathbf{Z} \). Thus, the \((i,j)\)-th entry of \( \mathbf{B} \) can be expressed as:

\[
[B]_{i,j} = \frac{1}{2} (d_i - d_j)^2 - \frac{1}{2} \left( (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \right), \quad 1 \leq i, j \leq M.
\] (1)

If we substitute \( \hat{d}_{m1} \) for \( d_{m1}^0 \) in (1), \( 1 \leq m \leq M \), we get the noisy scalar product matrix \( \hat{\mathbf{B}} \in \mathbb{R}^{M \times M} \), whose \((i,j)\)-th entry is:
\[
[B]_{i,j} = \frac{1}{2} \left( \hat{d}_{i,i} - \hat{d}_{j,j} \right)^2 - \frac{1}{2} \left[ \left( x_i - x_j \right)^2 + \left( y_i - y_j \right)^2 + \left( z_i - z_j \right)^2 \right], \quad 1 \leq i, j \leq M.
\] (2)

Note that \(B\) and \(\hat{B}\) are symmetric matrices. \(B\) is a function matrix with respect to \(u\), whereas \(\hat{B}\) is a constant matrix independent of \(u\). The cost function with respect to \(u\) is defined as [9]:

\[
f(u) = \frac{1}{2} \| B - \hat{B} \|_F^2.
\] (3)

The \(u_0 = [\hat{x}_0, \hat{y}_0, \hat{z}_0]^T\) which minimizes (3) can be defined as the estimate of the source position.

3. Proposed Iterative Localization Scheme

In this section, the iterative localization scheme is proposed. In Subsection 3.1, the Lanczos iteration is applied to the scalar product matrix to get a low-order matrix. In Subsection 3.2, the simplified MDS-based cost function is defined using the low-order matrix. In Subsection 3.3, the derivatives of the simplified MDS-based cost function are computed. In Subsection 3.4, the proposed localization scheme is presented. In Subsection 3.5, the computational complexity analysis is presented.

3.1. Applying Lanczos Iteration on the Scalar Product Matrix

According to the definition of the noisy scalar product matrix \(\hat{B}\) in (2), it can be derived that \(\text{rank}(\hat{B}) \leq 6\). As a result, applying Lanczos iteration [16] on \(\hat{B}\) can turn \(\hat{B}\) into a tridiagonal matrix \(T\), whose order is no more than 6. The details of Lanczos iteration are shown in Alg. 1 in Table 1. Note that the third column of Table 1 shows the flops of each step, which will be used to count the computational complexity in Subsection 3.5.

| Index | Computations | Flops |
|-------|--------------|-------|
| (S1)  | Denote the first column of \(\hat{B}\) by \(b \in \mathbb{R}^M\). | \(3M\) |
|       | Denote \(r = \text{rank}(\hat{B})\). |       |
|       | Let \(k = 0\), \(\beta_0 = 1\), \(q_0 = 0 \in \mathbb{R}^M\), \(q_i = b / \| b \| \in \mathbb{R}^M\), \(r_i = q_i \in \mathbb{R}^M\). |       |
| (S2)  | while \(k < r\) |       |
| (S3)  | \(q_{i+1} = r_i / \beta_i \in \mathbb{R}^M\) | \(M\) |
| (S4)  | \(k \leftarrow k + 1\) | \(0\) |
| (S5)  | \(\alpha_i = q_i^T \hat{B} q_i \in \mathbb{R}\) | \(2M^2 - M - 1\) |
| (S6)  | \(r_i = (\hat{B} - \alpha_i E_M) q_i - \beta_i q_{i-1} \in \mathbb{R}^M\) | \(2M^2 + M\) |
| (S7)  | \(\beta_i = \| r_i \| \in \mathbb{R}\) | \(2M\) |
| (S8)  | end (while) | \(0\) |
| (S9)  | Denote \(T_i = \begin{bmatrix} \alpha_i & \beta_i \\ \beta_i & \alpha_2 & \beta_2 \\ \vdots & \ddots & \ddots \\ \beta_{i-1} & \alpha_{i-1} & \beta_{i-1} \end{bmatrix} \in \mathbb{R}^{i\times i}\). | \(0\) |
3.2. Definition of the Simplified MDS-Based Cost Function

Alg. 1 in Table 1 has computed \( q_1, \ldots, q_r \), which is a set of orthonormal vectors in \( \mathbb{R}^M \). The set of \( q_1, \ldots, q_r \) can be completed to form an orthonormal basis of \( \mathbb{R}^M \) by adjoining \( q_{r+1}, \ldots, q_M \in \mathbb{R}^M \). Denote \( Q_n = [q_1, q_2, \ldots, q_r] \in \mathbb{R}^{M \times r} \) and \( Q_a = [q_{r+1}, q_{r+2}, \ldots, q_M] \in \mathbb{R}^{M \times (M-r)} \). It can be verified that \( Q_n^T \hat{B} Q_n = T_r \) and \( \hat{B} Q_a \) = 0. Since Frobenius norm is orthogonal invariant, the cost function (3) can be expressed as:

\[
f(u) = \frac{1}{2} \| B - \hat{B} \|_F^2 = \frac{1}{2} \| Q_n^T Q_r \|_F^2 (\hat{B} - B) (Q_n^T Q_r) ^T_\mathbb{R}.
\]  

(4)

Since \( Q_n^T \hat{B} Q_n = T_r \) and \( \hat{B} Q_a = 0 \), (4) can be simplified to:

\[
f(u) = \frac{1}{2} \left\| Q_n^T \hat{B} Q_n \right\|_F^2 - \frac{1}{2} \left\| Q_n^T B Q_n \right\|_F^2 = \frac{1}{2} \left\| \hat{B} Q_n \right\|_F^2 + \frac{1}{2} \left\| Q_n^T B Q_n \right\|_F^2.
\]  

(5)

When \( B \) is close to \( \hat{B} \), \( \| Q_n^T B Q_n \|_F \) and \( \| Q_n^T \hat{B} Q_n \|_F \) will be close to \( \| Q_n^T \hat{B} Q_n \|_F = 0 \) and \( \| Q_n^T B Q_n \|_F = 0 \), respectively. Denote

\[
B_r = Q_n^T B Q_n.
\]  

(6)

We can define the simplified MDS-based cost function as

\[
f_r(u) = \frac{1}{2} \| B_r - T_r \|_F^2.
\]  

(7)

In the following, we aim at finding the minimizer of the simplified MDS-based cost function (7) instead of the original MDS-based cost function (3).

3.3. Computation of the Derivatives of the Simplified MDS-Based Cost Function

The algorithm for computing the first- and second-derivatives of the simplified MDS-based cost function (7) is presented in Alg. 2 in Table 2. The derivatives of the simplified MDS-based cost function are necessary to compute the Newton direction and the step length. Since \( r = \text{rank}(\hat{B}) \leq 6 \), the orders of the matrices in Alg. 2 are no more than 6 except for (S2), (S7), and (S12). Note that the third and the fourth columns of Table 2 show the flops of each step which will be used to count the computational complexity in Subsection 3.5.

| Index | Computations | Flops for (7) | Flops for (3) |
|-------|--------------|--------------|--------------|
| (S1)  | Compute \( xI_M - x, yI_M - y, zI_M - z \in \mathbb{R}^M \). \( d_m^2 = \| u - s_m \|^2, d_m = \sqrt{d_m^2}, d_m = d_m^2 \cdot d_m, m = 1, \ldots, M \). Denote \( d = [d_1, \ldots, d_M] ^T \in \mathbb{R}^M \). Denote \( d_a = [\| d_a \|^T, \ldots, \| d_a \|^T] ^T \in \mathbb{R}^M \). Denote \( d_{\alpha} = [\sqrt{d_a}, \ldots, \sqrt{d_a}] ^T \in \mathbb{R}^M \). | 10M | 10M |
| (S2)  | \( Z_r = [x - xI_M, y - yI_M, z - zI_M] ^T \cdot Q_a \in \mathbb{R}^{k \times M} \). | 8rM - 4r | 0 |
| (S3)  | \( B_r = Z_r^T D Z_r \in \mathbb{R}^{n \times n} \). | 3.5r^2 + 3.5r | 3.5M^2 - 3.5M |
| (S4)  | Compute \( B_r - T_r \). | 2r - 1 | 0.5M^2 - 0.5M |
| (S5)  | \( f_r(u) = \frac{1}{2} \text{tr}
\left[ (B_r - T_r)^T \right] \in \mathbb{R} \). | 2r^2 | 2M^2 |

Table 2. Details of Alg. 2, computing the derivatives of \( f_r(u) \).
(S6) \[ \frac{\partial d}{\partial x} = d_j \odot (xJ_M - x) \in \mathbb{R}^m, \]
\[ \frac{\partial d}{\partial y}, \frac{\partial d}{\partial z} \in \mathbb{R}^m \] are computed similarly.

(S7) \[ \frac{\partial Z}{\partial x} = \left[-I_M, 0_M, 0_M, \frac{\partial d}{\partial x}\right]^T \cdot Q, \in \mathbb{R}^{4r}. \]
\[ \frac{\partial Z}{\partial y}, \frac{\partial Z}{\partial z} \in \mathbb{R}^{4r} \] are computed similarly.

(S8) \[ \frac{\partial B}{\partial x} = \left(\frac{\partial Z}{\partial x}\right)^T DZ, + \left(\frac{\partial Z}{\partial x}\right)^T DZ_y \in \mathbb{R}^{r''} \]
\[ \frac{\partial B}{\partial y}, \frac{\partial B}{\partial z} \in \mathbb{R}^{r''} \] are computed similarly.

(S9) \[ \frac{\partial f}{\partial x} = \text{tr} \left(\frac{\partial B}{\partial x} (B_s - T_s)\right) \in \mathbb{R} \]
\[ \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \in \mathbb{R} \] are computed similarly.

(S10) \[ \forall f_s(u) = \left[\frac{\partial f_s}{\partial x}, \frac{\partial f_s}{\partial y}, \frac{\partial f_s}{\partial z}\right]^T \in \mathbb{R}^3 \]

(S11) \[ \frac{\partial^2 d}{\partial x^2} = d_{ij} \odot [d \odot d \odot (xJ_M - x) \odot (xJ_M - x)] \in \mathbb{R}^m, \]
\[ \frac{\partial^2 d}{\partial y^2}, \frac{\partial^2 d}{\partial z^2}, \frac{\partial^2 d}{\partial x \partial y}, \frac{\partial^2 d}{\partial x \partial z}, \frac{\partial^2 d}{\partial y \partial z} \in \mathbb{R}^m \] are computed similarly.

(S12) \[ \frac{\partial^2 Z}{\partial x^2} = \left[0_M, 0_M, 0_M, \frac{\partial^2 d}{\partial x^2}\right]^T \cdot Q, \in \mathbb{R}^{4r}, \]
\[ \frac{\partial^2 Z}{\partial y^2}, \frac{\partial^2 Z}{\partial z^2}, \frac{\partial^2 Z}{\partial x \partial y}, \frac{\partial^2 Z}{\partial x \partial z}, \frac{\partial^2 Z}{\partial y \partial z} \in \mathbb{R}^{4r} \] are computed similarly.

(S13) \[ \frac{\partial^2 B}{\partial x^2} = \left(\frac{\partial^2 Z}{\partial x^2}\right)^T DZ, + \left(\frac{\partial Z}{\partial x}\right)^T DZ_y \frac{\partial Z}{\partial x} \]
\[ + \left[\frac{\partial^2 Z}{\partial x^2}\right]^T DZ_y + \left(\frac{\partial Z}{\partial x}\right)^T DZ_y \frac{\partial Z}{\partial x} \in \mathbb{R}^{r''}, \]
\[
\frac{\partial^2 B}{\partial x \partial y} = \left( \frac{\partial Z_i}{\partial x} \right)^T D Z_i + \left( \frac{\partial Z_i}{\partial y} \right)^T D \left( \frac{\partial Z_i}{\partial y} \right) + \left[ \left( \frac{\partial^2 Z_i}{\partial x \partial y} \right)^T D Z_i + \left( \frac{\partial Z_i}{\partial x} \right)^T D \left( \frac{\partial Z_i}{\partial y} \right) \right] \in \mathbb{R}^{r''}.
\]

\[
\frac{\partial^2 B}{\partial y^2}, \frac{\partial^2 B}{\partial z^2}, \frac{\partial^2 B}{\partial y \partial z}, \frac{\partial^2 B}{\partial z \partial x} \in \mathbb{R}^{r''} \text{ are computed similarly.}
\]

(S14) \[
\frac{\partial^2 f}{\partial x \partial y} = \text{tr} \left[ \left( \frac{\partial^2 B}{\partial x \partial y} \right) \left( B \right) - \left( \frac{\partial B}{\partial x} \right) \right] \in \mathbb{R}, \quad 24r^2 - 6 \quad 24M^2 - 6
\]

(S15) \[
V^2 f(u) = \begin{bmatrix}
\frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\
\frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\
\frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2}
\end{bmatrix} \in \mathbb{R}^{3 \times 3}
\]

3.4. The Proposed Iterative Method to Minimizing the Simplified MDS-Based Cost Function

The proposed localization scheme by minimizing the simplified MDS-based cost function is presented in Alg. 3 in Table 3. Alg. 3 includes two loops. The outer loop is the iteration of (S5) to (S10) of Alg. 3, where each iteration needs to use Alg. 2 once. The inner loop is inherently contained in the line search algorithm in (S7) of Alg. 3. Since the line search algorithm is standard [15], we do not present its detailed steps here. We only need to notice that the line search algorithm is itself iterative, where each iteration needs to use (S1) to (S10) of Alg. 2 several times. Note that the third and the fourth columns of Table 3 show the flops of each step which will be used to count the computational complexity in Subsection 3.5.

Table 3. Details of Alg. 3, the proposed localization scheme.

| Index | Computations | Flops for (7) | Flops for (3) |
|-------|--------------|---------------|---------------|
| (S1)  | Compute \( \hat{B} \) according to (2). | \( 6M^2 - 6M \) | \( 6M^2 - 6M \) |
| (S2)  | Apply Alg. 1 on \( \hat{B} \) to compute \( T_s \in \mathbb{R}^{r''} \) and \( Q_s \in \mathbb{R}^{M \times r''} \). | \( 4rM^2 + 3rM + 3M - r \) | 0 |
| (S3)  | Select an initial point \( u = [x, y, z]^T \). | 0 | 0 |
| (S4)  | Use Alg. 2 to compute derivatives of (7): \( f_i(u), V f_i(u), V^2 f_i(u) \). | \( 29rM + 79r^2 \) | \( 70.5M^2 + 16.5M - 9 \) |
| (S5)  | while \( \| V f_i(u) \| > 10^{-3} \) | 5 | 5 |
| (S6)  | The Newton direction is: | 28 | 28 |
\[ p = -\left(\nabla^2 f_s(u)\right)^{-1} \cdot \nabla f_s(u) \in \mathbb{R}^3. \]

(S7) Use line search algorithm to compute: the step length \( \alpha \in \mathbb{R} \).

\[
\begin{align*}
(17rM + 22r^2 + 13M - 3r - 4) \times n_{\text{iter}}(k) + (19.5M^2 + 7.5M - 3) \times n_{\text{iter}}(k)
\end{align*}
\]

(S8) \( u \leftarrow u + \alpha \cdot p. \)

(S9) Use Alg. 2 to compute derivatives of (7): \( f_s(u), \nabla f_s(u), \nabla^2 f_s(u). \)

\[
\begin{align*}
29rM + 79r^2 + 70.5M^2 + 16.5M - 9 + 25M - 6r - 10
\end{align*}
\]

(S10) end (while)

(S11) The estimate of the source position is \( u. \)

3.5. Computational Complexity Analysis

In this subsection, the computational complexity of the proposed localization scheme Alg. 3 is computed and compared with that of the iterative localization scheme using the original MDS-based cost function (3). We use flops to quantify the volume of work associated with a computation. A flop is a floating point add, subtract, multiply, or divide [16].

Firstly, we count the flops associated with Alg. 1. The flops of Alg. 1 are listed in the third column of Table 1.

Secondly, we count the flops associated with Alg. 2. We also count the flops associated with the similar algorithm which computes the derivatives of the original MDS-based cost function (3), where all of the multipliers \( Q \), from the right in (S2), (S7), and (S12) of Alg. 2 are omitted, and all of the \( T \), in (S4), (S5), (S9), and (S14) of Alg. 2 are replaced by \( B \). The flops of Alg. 2 are listed in the third and fourth columns of Table 2. Specifically, the third column of Table 2 shows the flops of computing the derivatives of the proposed simplified MDS-based cost function (7), whereas the fourth column of Table 2 shows the flops of computing the corresponding derivatives of the original MDS-based cost function (3).

Finally, we count the flops associated with Alg. 3. We also count the flops associated with the similar algorithm which computes the minimizer of the original MDS-based cost function (3), where (S2) of Alg. 3 is omitted. The flops of Alg. 3 are listed in the third and fourth columns of Table 3. Specifically, the third column of Table 3 shows the flops of minimizing of the proposed simplified MDS-based cost function (7), whereas the fourth column of Table 3 shows the flops of minimizing the original MDS-based cost function (3).

Assume that Alg. 3 needs \( n_{\text{iter}} \) iterations. In the \( k \)-th, \( k = 1, \ldots, n_{\text{iter}} \) iteration, assume that the line search step of (S7) of Alg. 3 needs \( n_{\text{iter}}(k) \) iterations of (S1) to (S10) of Alg. 2 to compute \( f_s(u) \) and \( \nabla f_s(u) \). Then, the flops of Alg. 3 can be expressed with respect to \( M, r, n_{\text{iter}}, \) and \( n_{\text{iter}}(k) \). The number of flops of Alg. 3 for (7) is:

\[
\begin{align*}
(17rM + 22r^2 + 13M - 3r - 4) \sum_{k=1}^{n_{\text{iter}}(k)} n_{\text{iter}}(k) + (29rM + 79r^2 + 25M - 6r + 29) \cdot n_{\text{iter}} + (4rM^2 + 6M^2 + 32rM + 79r^2 + 22M - 7r - 10) \cdot n_{\text{iter}}(k)
\end{align*}
\]

The number of flops of Alg. 3 for (3) is:

\[
\begin{align*}
(19.5M^2 + 7.5M - 3) \sum_{k=1}^{n_{\text{iter}}(k)} n_{\text{iter}}(k) + (70.5M^2 + 16.5M + 30) \cdot n_{\text{iter}} + (76.5M^2 + 10.5M - 9) \cdot n_{\text{iter}}(k)
\end{align*}
\]

As has been indicated in Subsection 3.1, \( r \leq 6 \). According to (8)-(9), when \( M \) is large, the flops of Alg. 3 for (7) will be significantly smaller than that for (3).

4. Simulations
In this section, the simulation results are presented to demonstrate the performance of the proposed localization scheme.

Suppose there are $M$ sensors and 1 source in the 3-D space, all of which are uniformly distributed in the cube of $[0\text{m},100\text{m}] \times [0\text{m},100\text{m}] \times [0\text{m},100\text{m}]$. The range difference measurements which are converted from TDOA measurements are generated by adding the zero-mean Gaussian noises to the true values. Assume the covariance matrix of $[q_2,\ldots,q_M]^\top$ is $[3]$

$$Q_q = \mathcal{E}[q_2,\ldots,q_M]^\top \cdot [q_2,\ldots,q_M] = 0.5\sigma^2 \left( E_{M-1} + I_{M-1} I_{M-1}^\top \right).$$

Since $q_1 = 0$, the covariance matrix of $q = [q_1,\ldots,q_M]^\top$ is:

$$Q = \mathcal{E} \{ qq^\top \} = \begin{bmatrix} 0 & 0 \\ 0 & Q_{M-1} \end{bmatrix}.$$  

Three localization methods are compared. The first method, called *proposed method*, is Alg. 3 which is the iterative method to find the minimizer of the simplified MDS-based cost function (7). The second method, called *minimizer of (3)*, is the iterative method to find the minimizer of the original MDS-based cost function (3). The third method, called *hyperbolic minimization*, is the iterative method to find the minimizer of the hyperbolic least squares cost function [2], which is the benchmark of iterative methods. Both the localization accuracy and the computational complexity of the three localization methods are compared.

Firstly, the estimate errors of the three methods are presented. The estimate error is defined as $\|\tilde{u} - u\|$, where $\tilde{u}$ is the estimate of $u$, 10,000 experiments are conducted, where the estimate errors of the three methods are recorded. Figure 1 shows the cumulative distribution functions (CDFs) of the three methods, where the sensor number $M = 20$ and the noise deviation $\sigma = (\sqrt{2} \times 0.5)\text{m}$. Figure 2 shows the medium values of the estimate errors of the three methods, where the sensor number $M = 20$ and the noise deviation $\sigma = (\sqrt{2} \times (0.5k))\text{m}$, $k = 1,2,\ldots,10$.

![Figure 1. CDFs of proposed method, minimizer of (3), and hyperbolic minimization.](image-url)
Figure 2. Medium values of proposed method, minimizer of (3), and hyperbolic minimization.

Secondly, the flops of the three methods are presented in Figure 3, where the noise deviation $\sigma = (\sqrt{2} \times 0.5) m$ and the sensor number $M = 10k$, $k = 1, 2, \ldots, 10$.

Figure 3. Flops of proposed method, minimizer of (3), and hyperbolic minimization.

According to Figures 1, 2, and 3, two conclusions can be obtained. Firstly, compare the proposed method and the minimizer of (3). The estimate error of the former is nearly the same with that of the latter, whereas the number of flops of the former is smaller than that of the latter. Specifically, when the sensor number is larger than 30, the number of flops of the former is about 20% as much as that of the latter. This means that the simplified MDS-based cost function (7) achieves nearly the same localization accuracy as the original MDS-based cost function (3), whereas it needs much fewer computations than (3). Secondly, compare the proposed method and the hyperbolic minimization. The estimate error of the former is always smaller than that of the latter, whereas the number of flops of the
former is larger than that of the latter. This means that the simplified MDS-based cost function (7) is more accurate than the hyperbolic least squares cost function.

5. Conclusion

Localization schemes with TDOA measurements using the MDS framework is a promising technique. In this work, a simplified MDS-based cost function is proposed, where the high-order scalar product matrices are converted to low-order matrices whose orders are no more than 6. This conversion significantly reduces the computations of the Newton’s iteration to find the minimizer of the cost function, whereas keeps the same localization accuracy as the original MDS-based cost function. Future works may include localization of multiple sources using the MDS framework.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant 51975347. The authors would like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[1] So H C 2019 Source localization: algorithms and analysis Handbook of position location: theory, practice, and advances ed S A Zekavat and R M Buehrer (New Jersey: John Wiley & Sons, Inc.) chapter 3 pp 59-106
[2] Huang Y, Benesty J, Elko G W and Mersereati R M 2001 Real-time passive source localization: a practical linear-correction least-squares approach IEEE Trans. on Speech and Audio Process. 9(8) 943-56
[3] Chan Y T and Ho K C 1994 A simple and efficient estimator for hyperbolic location IEEE Trans. on Signal Process. 42(8) 1905-15
[4] Ho K C and Xu W 2004 An accurate algebraic solution for moving source location using TDOA and FDOA measurements IEEE Trans. on Signal Process. 52(9) 2453-63
[5] Ho K C 2012 Bias reduction for an explicit solution of source localization using TDOA IEEE Trans. on Signal Process. 60(5) 2101-14
[6] Beck A, Stoica P and Li J 2008 Exact and approximate solutions of source localization problems IEEE Trans. on Signal Process. 56(5) 1770-8
[7] Chen S and Ho K C 2013 Achieving asymptotic efficient performance for squared range and squared range difference localizations IEEE Trans. on Signal Process. 61(11) 2836-49
[8] Wei H-W, Wan Q, Chen Z-X, Ye S-F 2008 Multidimensional scaling-based passive emitter localisation from range-difference measurements IET Signal Process. 2(4) 415-23
[9] Wei H-W, Peng R, Wan Q, Chen Z-X, Ye S-F 2010 Multidimensional scaling analysis for passive moving target localization with TDOA and FDOA measurements IEEE Trans. on Signal Process. 58(3) 1677-88
[10] Jiang W, Xu C, Pei L and Yu W 2016 Multidimensional scaling-based TDOA localization scheme using an auxiliary line IEEE Signal Process. Lett. 23(4) 546-50
[11] Cao J-M, Wan Q, Ouyang X-X and Ahmed H I 2017 Multidimensional scaling-based passive emitter localisation from time difference of arrival measurements with sensor position uncertainties IET Signal Process. 11(1) 43-50
[12] Wei H-W and Lu P-Z 2019 Analytical proof to two fundamental corollaries in multidimensional scaling-based localisation IET Signal Process. 13(8) 747-53
[13] Saeed N, Nam H, Al-Naffouri T Y and Alouini M-S 2019 A state-of-the-art survey on multidimensional scaling-based localization techniques IEEE Commun. Surv. Tutor. 21(4) 3565-83
[14] Wei H-W and Lu P-Z 2020 On optimality of weighted multidimensional scaling for range-based localization IEEE Trans. on Signal Process. 68(2) 2105-13
[15] Nocedal J and Wright S J 1999 Numerical optimization (New York: Springer) pp 59-60
[16] Golub G H and Van Loan C F 2013 *Matrix computations* (Baltimore: The Johns Hopkins Univ. Press, 4nd edn) pp 546-56, 12