UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Geolocating User Equipment Through The Cellular Network

A realistic simulation of UE localization using TDoA, Tikhonov regularization and real-world data

Lorella Lauriello & Francesco Lucibello

DEPARTMENT OF MATHEMATICS

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
Naples, Italy 2025

Abstract

We present two methods to numerically determine the position of User Equipment communicating to a cellular tower in predefined area, without any privileged access over the cellular network except and given enough measurements of the network delay. Moreover, we outline a framework on how to perform the measurements and correctly plug them into the numerical methods.

Contents

1 Introduction

1.1 Glossary and Notations

We will be using the following terminology and notations.

- User Equipment (UE)
  The term 3G refers to the *third generation* cellular network technology, which superseded the previous network generation 2G, proving for faster network speeds. The two most prominent standards are *UMTS* and and *CDMA2000*. As defined in the UMTS standard[1], we will refer to *User Equipment (UE)* as a SIM and device using that SIM to transmit information over the 3G cellular network, in most cases the device it's a mobile phone or an IOT device. In the context of this article it's not important if the UE is communicating through the 2G, 3G, 4G or 5G network, so this term will be used for any of those networks.
- Tower and Base Station (BS)
  By *tower* or *cellular tower*, we refer to the any device that is the entry point for UE to the cellular network, that it is the first device the UE sends a signal to when accessing the cellular network; a cellular tower, properly, is the most common device that does this, but it's not the only one. In this context we use this term in more general fashion, a more appropriate term might be *base station*, but it refers to the devices specifically on land.
- Mobile Core Network (MCN)
  A generic term to define the set of interconnected servers of the cellular network that route messages to and from the aggregations points.
- TDoA (Time Difference of Arrival)

1.2 Description of the problem

Governments track people's activity and behaviors on a daily basis by gathering incredibly large amounts of data, one way this is done is through the cellular network, as reported, for example, by the newspaper The Guardian in [13], following the revelations of the world-famous whistleblower Edward Snowden.
Using numerical methods like TDoA it's possible to track a speaking person location in a room using microphones and measuring the time of arrivals of the sound waves to the microphones. We are interested into extending this idea to cellular towers, that is determining a person location with UE, by measuring the times of arrivals from their UE to our receivers. In this context we consider the following players:

- Victim
  A user of the cellular network, with UE, whose location is to be discovered.
- Attackers
  The group of individuals, without any special access to the cellular network, determining the location of the victim.
- Towers

[1] See for example [WSA03, p. 4.2]

> The first device the UE connects to when accessing the cellular network, typically, a land base station.

This problem is easy to solve when attackers control the towers, as in the case of governments or the network operators themselves; our problem assumes that no special access to the cellular network is required to solve this localization problem.

1.2.1 Scope

To solve the outlined problem, we describe in detail its physical model, then we formalize it into a general mathematical model; finally we define a less general model that we solve numerically. We will also point to useful resources on how to gather necessary data to perform a POC attack, we developed some of those tools.

2 Physical Description

The cellular network is an interconnected set of devices that operates under different technologies, namely 2G, 3G, etc.; we provide a technology-independent description, as in [Sim23].

2.1 Cellular network architecture

The following subjects are present:

- User Equipments
- Cell towers (or base stations)
- Aggregation points
- Mobile Core Network

And they are structured like follows.

- Each UE can be connected with only one tower at time.
- Each tower is connected with one and only one aggregation point.
- Each aggregation point is connected to the mobile core network.
- Every couple of subjects whose connection state is not stated above, are not connected with each other.[2]
- If it's possible to communicate from one subject to another, it's possible also the opposite.

In order for a information to travel from a user A to a user B, it is necessary a *routing algorithm*.

2.2 The routing algorithm

If Alice and Bob are users of the cellular network and Alice wants to send a message[3] to Bob, the following protocol is established.

1. Alice lists towers reachable by her UE and chooses the tower T_a with the best signal
2. Alice sends her (encrypted) message, directed to Bob, to the tower T_a

[2] For example, we didn't say that towers are connected with each other, then they are not.

[3] In the general meaning of "message".

3. The tower T_a sends the message to its aggregation point A_a
4. A_a sends the message to the MCN
5. The MCN identifies the aggregation point A_b such that the tower T_b, which Bob is using, is connected to A_b
6. The MCN sends the message to A_b
7. A_b sends the message to T_b
8. T_b sends the message to Bob

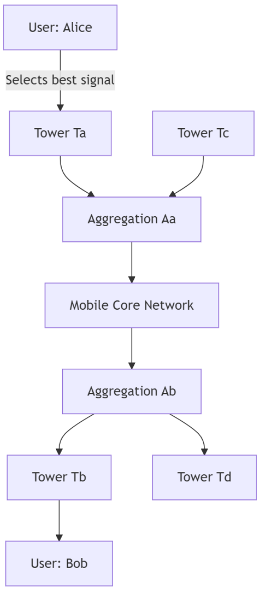

Figure 1: Communication of two users in the Cellular Network (Schematic)

The network structure is also represented in the Figure 1.

3 Mathematical Model

From the physical description, we derive a simpler mathematical model.

3.1 Assumptions

From now on when we talk about communication between towers, attackers or the victim, we mean the sending/receiving of a fixed message, when we use the term "between" when talking about a communication we're not discriminating over the direction of the communication, it is irrelevant.
Let $s \in \mathbb{N}$ a number that will have a practical meaning in section 4. We make the following assumptions:

1. The victim doesn't move during the attack.
2. The mean communication time between two nodes of the graph over a number of samples s is constant over time.
3. The mean communication times, over s samples, between the towers are known.
4. The geographical positions of the towers are known.

5. The communication time between a UE and a tower is the distance between them divided by the speed of light in the air c.
6. The victim connects to only one tower and it is the closest to the victim position.
7. The mean travel times can be "split up" over a path: if $i, j \in W$ and exists $k \in W$ such that $(i,k), (k,j) \in E$ and $i, j, k \neq N$,[4] then

$$\tau(i,j) = \tau(i,k) + \tau(k,j)\,.$$

3.2 Model definition

Let V be the victim and s as described in subsection 3.1. Let $T_1, \dots, T_{n_t}$ and $A_1, \dots, A_{n_a}$ be respectively the towers and the attackers, where $n_t, n_a \in \mathbb{N}$. Let N represent the "rest of the network", that is MCN and the aggregation points that allow the towers to exchange information with each other.
We formalize the network structure into a directed weighted graph[5] $G = (W, E, \tau)$, where its nodes are $W = \{T_1, \dots, T_{n_t}, A_1, \dots, A_n, V, N\}$ and its edges are E. $\tau : W^2 \to [0, \infty)$ is the function such that $\tau(i,j)$ is the mean of s samples of the times of communication between two nodes of the graph according to the algorithm in subsection 2.2, it is well defined for Assumption 2. τ is called *mean travel time over s samples* or simply *mean travel time*.
Following the specifications in subsection 2.1, we have

$$\exists\,!\; i_V \in \{1, \dots, n_t\} : (V, T_{i_V}) \in E$$

$$(T_i, N) \in E \quad \forall\, i \in \{1, \dots, n_t\}$$
$$(T_i, T_j) \notin E \quad \forall\, i, j \in \{1, \dots, n_t\}$$
$$(i,j) \in E \implies (j,i) \in E\,.$$

Without loss of generality, we can suppose that no more than one attacker is connected to the same tower because all the attackers have the same role and we consider them to be equivalent, also since the attackers must be connected to some tower to perform the attack, there must be at least one tower for each attacker. So $n_a \leq n_t$ and

$$\forall\, i \in \{1, \dots, n_a\}\; \exists\,!\, j \in \{1, \dots, n_t\} : (A_i, T_j) \in E\,,$$

Without loss of generality, we can suppose that the attackers don't communicate with each other during the attack, because it's not useful for the attack, so

$$(A_i, A_j) \notin E \quad \forall\, i, j \in \{1, \dots, n_a\}\,,$$

moreover the attackers don't communicate with the victim directly, so

$$(A_i, V), (V, A_i) \notin E \quad \forall\, i \in \{1, \dots, n_a\}\,.$$

[4] To understand this conditions, see subsection A.2.

[5] G is not properly a weighted graph because τ should be defined on E, while it's defined on a superset of it, W^2, but $\tau|_E$ is the function we're looking for, so it's just a formality, τ has more properties than a simple weight function.

3.3 Model Solution

The solution method is based on solving a TDoA inspired problem, which will be described more later; in this section we calculate the mean time of communication, over s samples, from the victim V to an attacker A_i, that is $\tau(V, A_i)$, and show that $\tau(V, A_i) - \tau(V, A_j)$ depends on the geographical position of the victim, where $j \neq i$. For Assumption 7, we have

$$\begin{aligned}\tau(V, A_i) &= \tau(V, T_{i_V}) + \tau(T_{i_V}, T_{A_i}) + \tau(T_{A_i}, A_i) = \\ &= \frac{\mathrm{d}(V, T_{i_V})}{c} + \tau(T_{i_V}, T_{A_i}) + \frac{\mathrm{d}(T_{A_i}, A_i)}{c}\,,\end{aligned}$$

which we can calculate because of Assumption 3, Assumption 4, and Assumption 5. Therefore

$$\begin{aligned}&\tau(V, A_i) - \tau(V, A_j) = \tau(T_{i_V}, T_{A_i}) + \frac{\mathrm{d}(T_{A_i}, A_i)}{c} - \tau(T_{i_V}, T_{A_j}) - \frac{\mathrm{d}(T_{A_j}, A_j)}{c} = \\ &\tau(T_{i_V}, T_{A_i}) - \tau(T_{i_V}, T_{A_j}) + \frac{\mathrm{d}(T_{A_i}, A_i) - \mathrm{d}(T_{A_j}, A_j)}{c}\,.\end{aligned}$$

So the last term depends in fact on the position of the victim because of Assumption 6, due to the term T_{i_V}, but, unfortunately, it does not depend on the distance of the victim from the tower (while $\tau(T_{i_V}, T_{A_i})$ and $\tau(T_{i_V}, T_{A_j})$ do). This implies that our results will be dependent on the victim position in the sense that they will depend on the tower the victim connects to.

4 POC attack

In order to perform a proof of concept attack, we need to satisfy the assumptions in subsection 3.1:

- The Assumption 1 is plausible if the attack happens in a small amount of time or the victim moves very slowly or doesn't move at all during the attack.
- The Assumption 2 and Assumption 3 are satisfiable if s is a large enough number determined experimentally, so that the error on its measure is known and negligible.
- The Assumption 4 is satisfiable by finding the positions of towers on an open database like OpenCellID[6], that is crowdsourced but not very updated in some areas. For more precise/reliable results it's possible to take measurements autonomously using cell data collector apps, like Tower Collector, that is an open source app for Android.
- The Assumption 5 is plausible due to the physics of the electromagnetic radiation and if the UE communicates with the tower without any delay.
- The Assumption 6 is plausible in most cases. In reality the tower that the UE chooses to connect is the one with the best signal, that, most of the times, is the one that is located the closest to the UE.[7]
- The Assumption 7 is plausible if there are no delays in a node k after a message was received from a node i and has to be sent to a node j.

4.1 Measuring times

As in subsection 3.3, to apply TDoA and solve the problem numerically, the attackers need to measure the communication time from the victim to the i-th attacker.

The attack we propose is plausible in an area accessible by the attackers, that is small enough to be effectively mapped by the attackers. The attack works as follows, for simplicity we suppose the attackers are just A_1 and A_2:

1. The attackers agree on a message m to send over the cellular network.
2. The attackers agree on a number of samples s that is big enough to get constant results when averaging the measurements of the network delays between tower communications, the determination of s is guided by experimentation and efficiency needs.
3. The entire area is scanned for towers using map data collector tool, the attackers get the number of accessible towers in the area n_t and the cell ids.
4. By triangulating the tower positions[8] or looking on open databases if available, the attackers get the tower positions $T_1, \dots, T_{n_t}$.
5. For each couple of towers (T_i, T_j)
    (a) A_1 connects to T_i and A_2 connects to T_j

[6] Also https://github.com/beacondb/beacondb could be used.

[7] This assumption could be substituted by *The tower the victim connects to is one with the best signal for the victim.*, but then it's necessary to use one of the aforementioned measurements methods or open databases to create a "signal map" where every unit of the map has the associated tower with the strongest signal.

[8] A RSSI-based localization technique can be used in this case, because the attackers measure the tower signal strengths.

(b) A_1 sends s times the message m to A_2, they can get $\tau(T_i, T_j)$.[9]

(c) A_2 sends s times the message m to A_1, they can get $\tau(T_j, T_i)$.

The complexity, of communication measurements of size s, of this algorithm is n_t^2, if one assumes that $\tau(T_i, T_j) = \tau(T_j, T_i)$, the complexity is $\frac{n_t^2}{2}$. It's possible to reduce the complexity further with different assumptions, but they wouldn't work with our TDoA solution, as explained in subsection A.2.

4.2 Silent SMS

In our experiments we measured some real network delay values and we averaged them. The attackers need to agree on a message m and a number of samples s, for our measurements we used an SMS with the content "test" and we sent $s = 50$ of those SMS, in order to make the process fast and without buying expensive equipment, we provide a script available at `https://codeberg.org/frollo/cellular-network-geolocalization`, that uses an Android phone with Android SMS Gateway app installed. In real case scenario it could be useful to send a *Silent SMS*, that is an SMS that gives no notification to the user that receives it,[10] making it ideal of an undetectable attack.

4.3 Our measurements

In our experiments we mapped the area of the Monte Sant'Angelo (MSA) university campus, it took about 3 hours in total, we applied the algorithm with complexity $\frac{n_t^2}{2}$. Even though our devices detected more than 10 towers, but 3 towers had a very strong signal that made it very difficult for our devices to steadily connect another tower. In total we measured the delays between 4 towers, we believe that with better hardware or software[11] it's possible to measure more towers more easily and increase the measured times accuracy.

[9] The attacker A_2 measures

$$\tau(A_1, A_2) = \tau(A_1, T_i) + \tau(T_i, T_j) + \tau(T_j, A_2)\,,$$

A_2 can calculate numerically $\tau(A_1, T_i)$ and $\tau(T_j, A_2)$ because the positions of the towers are known and because of Assumption 5 and subtract them to the measured value.

[10] We used an open source app called Silent SMS Detector that sends and detects silent SMS, but it doesn't allow to send SMS in bulk and get more accurate times for measurements.

[11] Android has in place some limitation to system log access, that made it more difficult to get precising timings, moreover we didn't find a way to force Android to manually switch to a reachable cell tower of our choice. Probably iOS is much worse.

5 Mathematical methods

In this section, we focus on the mathematical methods used to solve the problem described previously. These methods can be applied to solve a general localization problem, even one arising from a different physical context. For this reason, we use terms such as signal detectors and signal source; we follow some of mathematical results as given in [Zha21] and [BH].
The methodology used in this work can be defined as a TDoA-inspired methodology. This localization technique is applicable in a two-dimensional environment where the positions of a set of signal detectors are known, and each detector is capable of measuring a physical quantity related to the signal emitted by the source. For instance, the detectors may measure the exact time when they detect the signal, the signal's intensity, or other related quantities. The key point is that the physical quantity measured by each detector is the same for all detectors.
Moreover, the relationship between the chosen physical quantity and the source's location is known, meaning that we have a model that relates this physical quantity to the source position. Based on this, a set of candidate positions is established within the localization area. For each candidate position, the model is used to predict the values that the detectors would measure if the signal were emitted from that position. Once the signal source emits the signal, the detectors record the actual values corresponding to the signal. Then, based on the predicted values from the model for each candidate position and the measured values, a probability is assigned to each candidate location, representing the likelihood that the source is at that position. The source is then estimated to be at the candidate position with the highest probability.
In this case, the method is TDoA-inspired because the physical quantity (measured or estimated) is the signal reception times, from which the Time Difference of Arrival (TDoA) is calculated for each pair of detectors.
This section is dedicated to the mathematical methods used to assign a likelihood value to each candidate position. To achieve this goal, we need to construct an appropriate system of linear equations and solve the corresponding least squares problem.

5.1 Linear system construction

To solve the localization problem, we first need to construct a system of linear equations:

$$\mathbf{Ax} = \mathbf{b}$$

where:

- $\mathbf{A}$ is a $P \times M$ matrix, where P is the number of detector pairs and M is the number of candidate positions.
- $\mathbf{b}$ is a vector of length P.

The entries in the matrix A are the predicted TDoA values for each pair of detectors $p = (i, j)$ at each candidate position $r^{(k)}$:

$$A_{p,k} = \Delta t_{ij}(r^{(k)}) = \mathbf{t}_i(r^{(k)}) - \mathbf{t}_j(r^{(k)}).$$

The entries in the vector b are the measured TDoA values for each pair of detectors $p = (i, j)$:

$$b_p = \Delta t_{ij}^{\text{meas}} = \mathbf{t}_i^{\text{meas}} - \mathbf{t}_j^{\text{meas}}.$$

Note that both the entries of A and the coefficients of b are given in the problem. In fact, for the entries of A, the signal arrival time from the k-th candidate position to the i-th detector is estimated through the model. On the other hand, the coefficients of b come from real measurements.

5.2 Least Squares Problem resolution

A solution $\mathbf{x}$ of the corresponding least squares problem

$$\min_{\mathbf{x} \in \mathbb{R}^M} \|\mathbf{Ax} - \mathbf{b}\|$$

can be interpreted as a weight vector, where each component x_k indicates the likelihood that the source is at the candidate position r_k.
When you solve the least squares problem, you are effectively minimizing the error between $\mathbf{Ax}$ and $\mathbf{b}$. For each $\mathbf{x} \in \mathbb{R}^M$, $\mathbf{Ax}$ is a linear combination of the columns of $\mathbf{A}$ weighted by the scalars x_k. Since the k-th column of $\mathbf{A}$ contains the predicted TDoA values for the k-th candidate position, the k-th component x_k represents how much the predicted TDoA values for the candidate position r_k contribute to the minimization of the error, that is, how well these predicted values fit the measured data.
In the context of localization, the solution vector $\mathbf{x}$ can be thought of as a measure of the likelihood of the source being at each candidate position. The component of $\mathbf{x}$ that has the highest value corresponds to the candidate position that is the most likely location for the sound source.

5.3 Solution of the Least Squares Problem using the Moore-Penrose Pseudoinverse

To solve the least squares problem

$$\min_{\mathbf{x} \in \mathbb{R}^M} \|\mathbf{Ax} - \mathbf{b}\|$$

we use the Moore-Penrose pseudoinverse of the matrix $\mathbf{A}$. One solution can be obtained by computing the pseudoinverse $\mathbf{A}^\dagger$ and multiplying it by the measured data vector $\mathbf{b}$:

$$\mathbf{x}_{\text{pinv}} = \mathbf{A}^\dagger \mathbf{b}$$

However, the matrix $\mathbf{A}$ is typically ill conditioned. This implies that the system is ill posed, which means that small perturbations $\delta\mathbf{b}$ in the data can result in large changes $\delta\mathbf{x}$ in the solution.
Because of ill-posedness of the system, we need to use some regularization technique. In this case, we use the Tikhonov regularization.

5.4 Tikhonov regularization

Tikhonov regularization consists of replacing the usual least squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x}-\mathbf{b}\|$$

with the penalized problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 + \lambda^2\|\mathbf{x}\|^2$$

for an appropriate choice of the parameter λ.
For each $\lambda \neq 0$, the penalized problem has a unique solution given by:

$$\mathbf{x}_\lambda = \left(\mathbf{A}^T\mathbf{A} + \lambda^2\mathbf{I}\right)^{-1}\mathbf{A}^T\mathbf{b}$$

The choice of λ modifies the sensitivity of the solution to perturbations on $\mathbf{b}$. This means that:

- A larger λ increases stability (i.e., reduces the amplification of $\delta\mathbf{b}$).
- A smaller λ makes the method closer to the unregularized problem.

It seems that we are interested in a value of λ as large as possible to increase the stability of the solution. Unfortunately, λ cannot be too large, because by solving the penalized problem, we accept a suboptimal $\mathbf{x}$ that gives a slightly larger norm $\|\mathbf{A}\mathbf{x}-\mathbf{b}\|$. This means that a larger λ may bias the solution away from the true least squares answer.
A value of λ is optimal if the solution to the corresponding penalized problem is close to the solution of the least squares problem (i.e the residual norm is small) and improves the stability of the solution. In other words, the optimal λ guarantees the optimal balance between fitting the data and maintaining a stable solution. We cannot prioritize the stability of the solution because we must preserve the meaning of the solution, which is a vector where each component represents the likelihood that the signal source is at the corresponding candidate position. To achieve this, we need to obtain a solution that is close to the solution of the least squares problem, which, as mentioned earlier, has this meaning. In this way, for an appropriate value of λ, we can still interpret the solution $\mathbf{x}_\lambda$ as a measure of the likelihood of the source being at each candidate position.

5.5 Choice of λ Using the L-Curve Method

The selection of the regularization parameter λ is crucial for balancing the trade-off between the stability of the solution and its proximity to the true least squares solution. One effective method to determine the optimal value of λ is the L-curve method. The L curve is a graphical representation: it's about to create a plot with $\log\|Ax_\lambda - b\|$ on the horizontal axis, and $\log\|x_\lambda\|$ on the vertical axis, sampled over a wide range of λ values (varying over orders of magnitude).
By plotting these two quantities for different values of λ, the L-curve forms a characteristic "L" shape. The corner of the L-curve corresponds to the point of optimal regularization, where the solution transitions from fitting the data well (small residual) to a more regularized, stable solution (smaller solution norm). The optimal value of λ is chosen at this corner, as it represents the best compromise between data fidelity and solution stability.

6 Computational methods

This section will be dedicated to the computational methods used to numerically solve the problem discussed in this work.
For this reason, we will use the specific terminology of the problem we aim to solve. In this context, the attackers play the role of signal detectors, while the victim is the signal source.
We have implemented the code in the MATLAB programming language and it is freely accessible at `https://codeberg.org/frollo/geolocating-ue-through-the-cellular-net`

6.1 Environment and Data Simulation

- The first task of this code is to represent the problem in a two-dimensional environment. In fact, the positions of the towers and the attackers are initially provided in geographic coordinates (latitude and longitude). In other words, a more or less extensive geographical area (Italy, Campania, MSA university campus) is considered, where the geographical positions of at least 4 towers and the same number of attackers are known. Therefore, all the geographic coordinates are converted to cartesian coordinates using the `projfwd` function. Specifically, the function `projfwd(proj, lat, lon)` converts the geographic coordinates (latitude and longitude) into cartesian coordinates (`x, y`) based on the projection defined by `proj`. `proj` defines a map projection, which dictates how the Earth's curved surface is projected onto a flat 2D plane. We use the EPSG:6876 projection, that projects an surface that covers the entire Italy, this projection preserves the distance between two points with good approximation.
- The second step of the algorithm is to discretize a portion of the previously created cartesian plane into a grid, starting from a point chosen within the localization area. The positions of this grid are the candidate positions for the victim's location, meaning known positions for which the TDoA values are estimated using the time model.
- At this point, an undirected graph is created through a symmetric matrix defined as follows: the nodes of the graph are all the players in our problem (towers, attackers, candidate positions), so the matrix will be square with dimension N, where

  $$N = n_{\text{towers}} + n_{\text{attackers}} + n_{\text{candidate\_positions}}.$$

  The entries of this matrix represent the connections between towers, attackers and candidate position. Each attacker is connected to its closest tower, and each candidate position is similarly connected to its closest tower. In terms of matrix entries: the element $G(\text{idx1}, \text{idx2})$ is equal to 1 if idx1 corresponds to an index of an attacker or a candidate position, and idx2 is the index of its closest tower, or vice versa. All other components of G are zero.
  This way, only relevant interactions between nodes are represented in the matrix.

- Then the algorithm constructs the matrix A, which means that the predicted TDoA values for each attackers pair and for each candidate position are computed.
  In particular, for each candidate position r_k and each attacker a_i, the signal arrival time from the candidate position to the attacker is calculated according to the model. That is, by consulting the graph, the nearest tower to the attacker and the nearest tower to the candidate position are identified, and the signal arrival time is:
  $$t_i(r_k) = \frac{\mathrm{d}(a_i, T_{a_i})}{c} + \frac{\mathrm{d}(r_k, T_{r_k})}{c} + \mathrm{comm}(T_{a_i}, T_{r_k})$$
  where T_{a_i} is the position of the tower nearest to attacker a_i, T_{r_k} is the position of the tower nearest to the candidate position r_k, and $\mathrm{comm}(T_{a_i}, T_{r_k})$ is the mean communication time between the two towers. Note that $\mathrm{comm}(T_{a_i}, T_{r_k})$ is a given data of the problem, stored in the code in an array where each component represents the communication time between two towers.
  Then the Time Difference of Arrival are computed and stored in A.
- To solve a numerical problem, we need the measured TDoA values for the signal emitted from an unknown position. As described in subsection 4.3, in the MSA scenario, we measured the tower delays and gathered the tower positions, the rest of the data has been simulated.
  The measured TDoA values are also simulated as follows: a position is fixed within the localization area, it is transformed into cartesian coordinates on the plane, and the TDoA values are calculated for each pair of attackers from the model used for each candidate position. To simulate the data measurement, Gaussian noise is added. That's how we obtain the right-hand side vectors of the linear system.

6.2 Inverse Problem solution

- The second part of the code is dedicated to the inverse problem solution. First, we solve the Least Squares problem by computing the Moore-Penrose pseudoinverse using the MATLAB function `pinv`. This function computes the pseudoinverse based on Singular Value Decomposition of matrix A.
  Subsequently, we determine the optimal value of λ in order to apply Tikhonov regularization. According to mathematical methods explained previously, we implemented L curve method by resolving multiple penalized problem with different value of λ. So the optimal value of λ is established by finding corner via maximum curvature.
  At this point, we are ready to solve the penalized problem with the optimal value of λ.
  We plot a heatmap of the reconstructed likelihood over the candidate grid both for the solution of Moore Penrose pseudoinverse and the solution obtained with Tikhonov regularization and visualize the map, tower and attacker positions, the true victim's position.
- In the last part of the code, the noise level added to the vector of known terms (b) is varied. For each noise level, both the least squares problem and the

penalized problem (Tikhonov regularization) are solved in order to analyze and study the stability of the two solutions.

7 Results

Our code is capable of running a an attack simulation in 3 different attack scenario: MSA university campus, Campania and Italy. In this section, we will discuss the numerical results obtained by running our code in the Campania setting. Specifically, the code was executed by selecting the geographic locations of 4 towers in Campania (from the sources mentioned previously), the geographic locations of 4 attackers and the victim's geographic location.

We plot the heatmaps of the reconstructed likelihood based on the solution of the least squares problem, the L-curve used to determine the optimal value of lambda, and the heatmap of the reconstructed likelihood based on the solution of the penalized problem with the optimal lambda value.

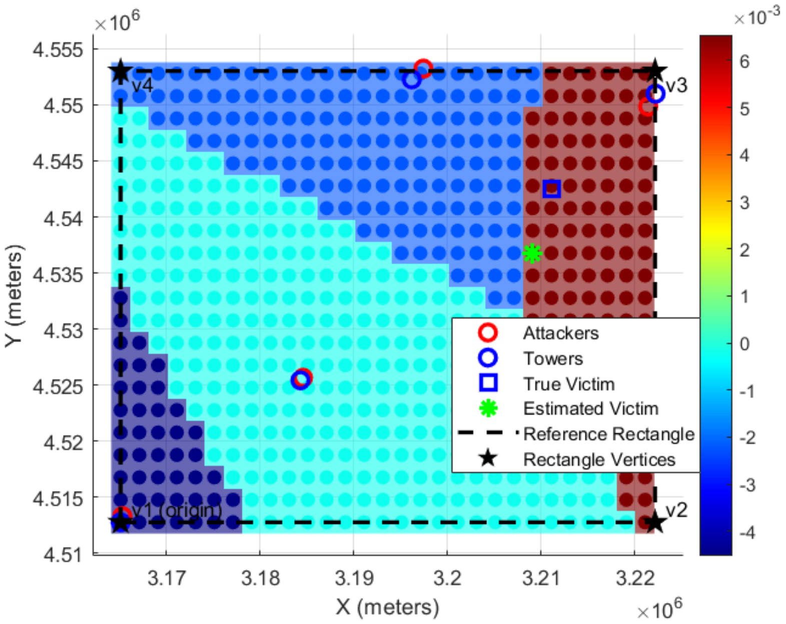

Figure 2: Heatmap of the LSP solution

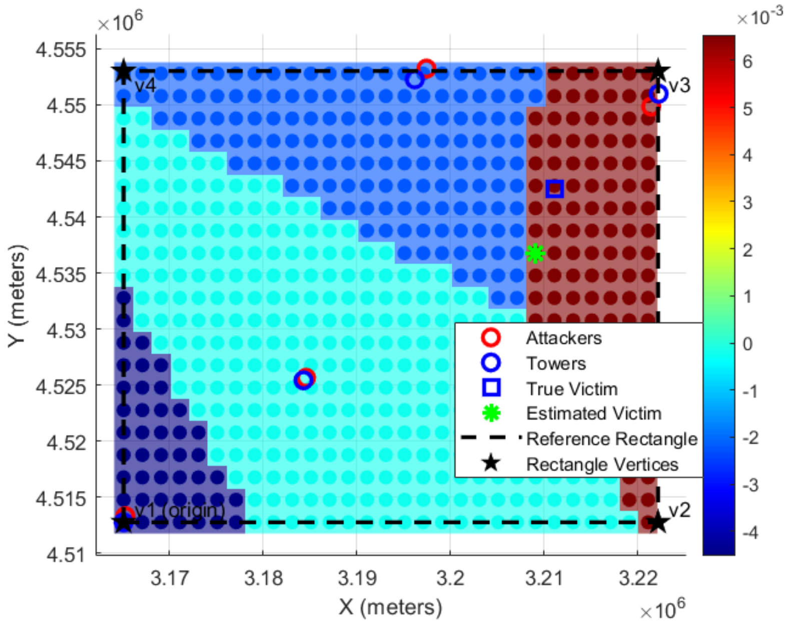

Figure 3: Heatmap of Tikhonov solution

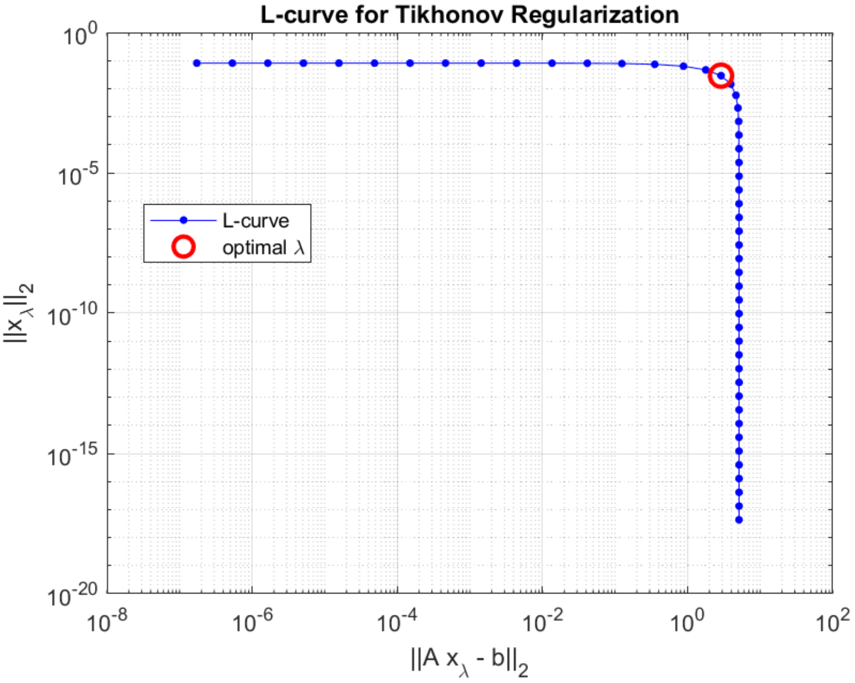

Figure 4: L-curve

We present the plots related to the stability of the two methods.

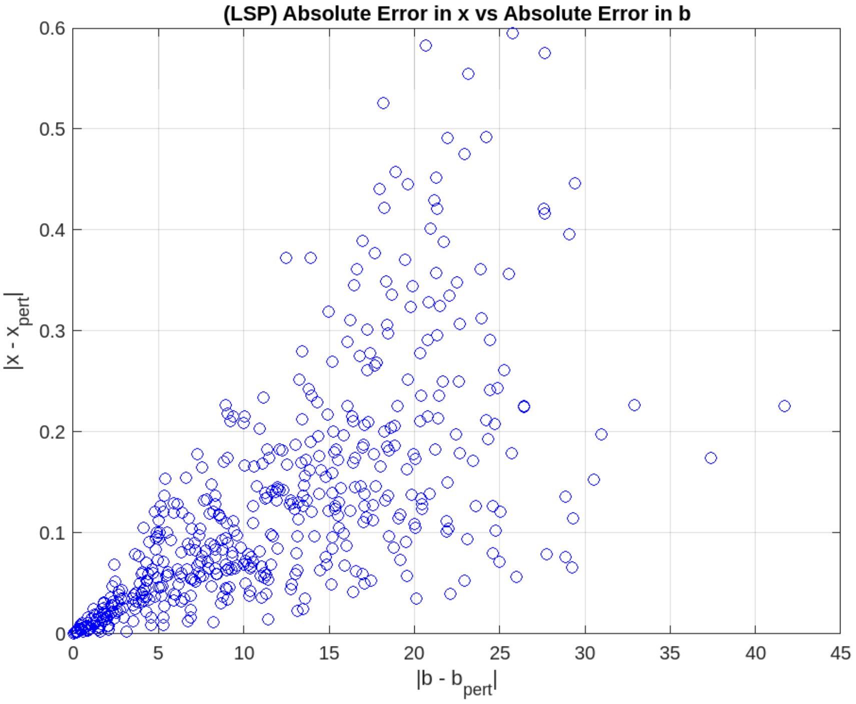

Figure 5: Stability of LSP

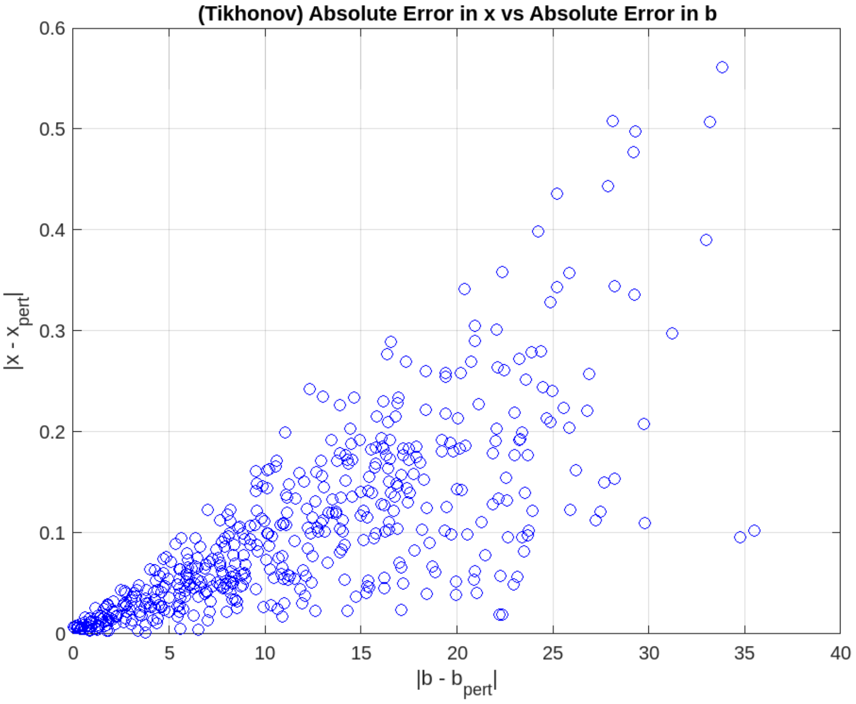

Figure 6: Stability of Tikhonov

First of all, one can note that both heatmaps Figure 2 and Figure 3 exhibit a non-continuous distribution of the likelihood. This is due to the fact that each candidate position is assigned to its nearest tower, and the vector of predicted TDoA values for a candidate position depends solely on the tower it is connected to. As a result, two candidate positions connected to the same tower will have identical columns in matrix A. Moreover, each component of the vector $\mathbf{x}$ measures how well the predicted TDoA values (the column) for the corresponding position fit the measured differences. Consequently, candidate positions assigned to the same tower will result equally probable. In other words, the likelihood distribution over the candidate positions follows their assignment to the towers, which is not continuous. The second thing to note is that the two methods (Least Squares Problem and Tikhonov regularization) exhibit almost identical behavior both in terms of heatmaps and solution stability. The heatmaps graphically represent the two solutions x_{LSP} and $x_{\text{Tikhonov\_opt}}$ resulting from the execution of the algorithm. The similarity of the heatmaps suggests that the two solutions are numerically close. In fact, it turns out that $\|x_{\text{LSP}} - x_{\text{Tikhonov\_opt}}\| = 0.08$. From this, we can conclude that the value of λ_{opt} selected by the algorithm (implementing L curve method) is a good value in terms of data fitting.

However, from the similarity of solution stability (Figure 5 and Figure 6) the value of λ_{opt} does not achieve the second goal of Tikhonov regularization, which is to obtain a more stable solution with respect to data perturbations.

This is certainly unexpected from a theoretical and general perspective, but not in the specific case, considering the results identified by the algorithm. In fact, in this case, the matrix constructed with the input data is moderately ill-conditioned, and consequently, the least squares problem is not as unstable as one would theoretically expect. This is confirmed from a numerical point of view: indeed, during the execution of the algorithm, the smallest non zero singular value of the matrix A was computed, which is found to be 3.5×10^1. This value is very important because it provides an estimate of the perturbation on the solution x. Indeed, in general, let $\sigma_{\min}$ be the smallest non-zero singular value of the matrix A, it follows that

$$\|\Delta x\| \leq \sigma_{\min}^{-1} \|\Delta b\|$$

Therefore, since in this case $\sigma_{\min}^{-1}$ is not a very large value (on the order of 10^{-2}), we can conclude that the perturbation on b is not significantly amplified.

On the other hand, it is expected that Tikhonov regularization will still improve the stability of the solution. In this case, the numerical value of λ_{opt} identified by the algorithm is 8.2×10^1. The reason why the stability of the solution does not improve is that the value of λ_{opt} is of the same order of magnitude as the smallest non-zero singular value of the matrix A. Indeed, now the perturbation of the solution is related to the perturbation on the data as follows:

$$\|\Delta x\| \leq (2\lambda_{\text{opt}})^{-1} \|\Delta b\|.$$

Another confirmation that the matrix is moderately ill-conditioned is the plot of the L-curve, which does not exhibit the typical and expected L-shape (Figure 4). In a moderately ill-conditioned problem, the system does not amplify the noise significantly, which means that the effect of increasing λ results in a more homogeneous variation between the norm of the residual and the norm of the solution.

The algorithm provides the value of λ corresponding to the point of maximum curvature, which is why it gives a value of λ_{opt} on the order of 10^1. However,

by observing Figure 4, it is expected that by increasing λ_{opt} by a few orders of magnitude, stability can be improved without significantly penalizing the residual norm, which remains more or less similar.
This is what was done by multiplying λ_{opt} by 10^2.The following results were obtained:

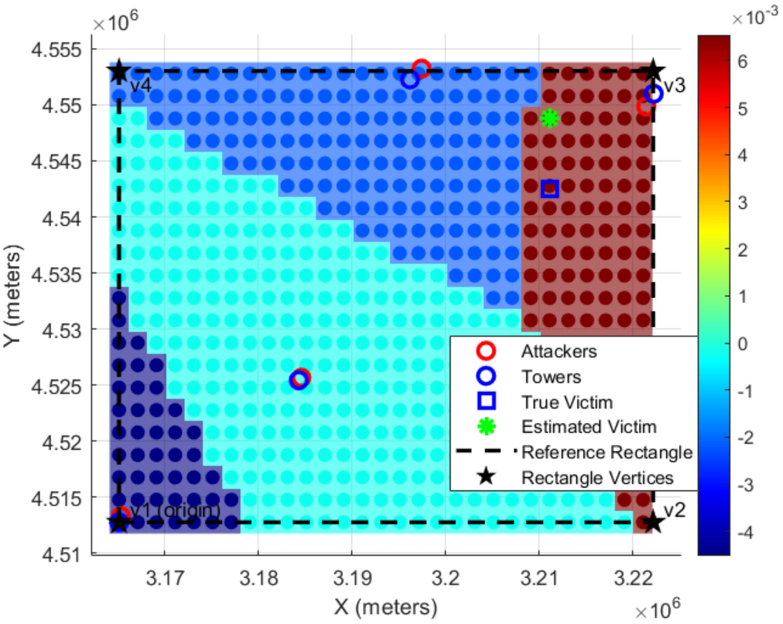

Figure 7: Heatmap of Tikhonov with $\lambda = \lambda_{\text{opt}} \cdot 10^2$

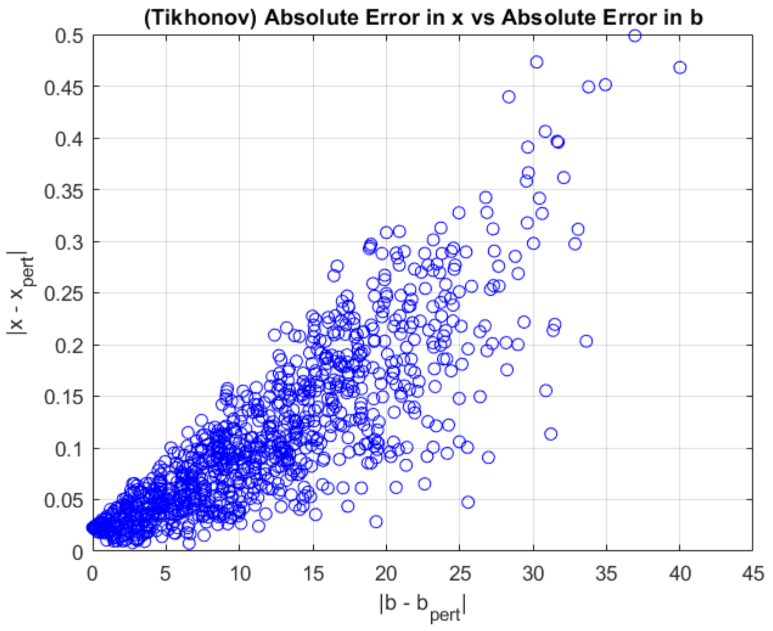

Figure 8: Stability of Tikhonov with $\lambda = \lambda_{\text{opt}} \cdot 10^2$

It can be observed that the stability begins to slightly increase, but the heatmap is similar to the previous ones. This means that the solution changes only marginally.

Just to ensure that our algorithm correctly identified the value of λ_{opt}, we numerically modified the matrix A (essentially, we set the speed of propagation to 1 m/s), resulting in a more ill-conditioned matrix. In this case, the L-curve exhibits the typical L-shape.

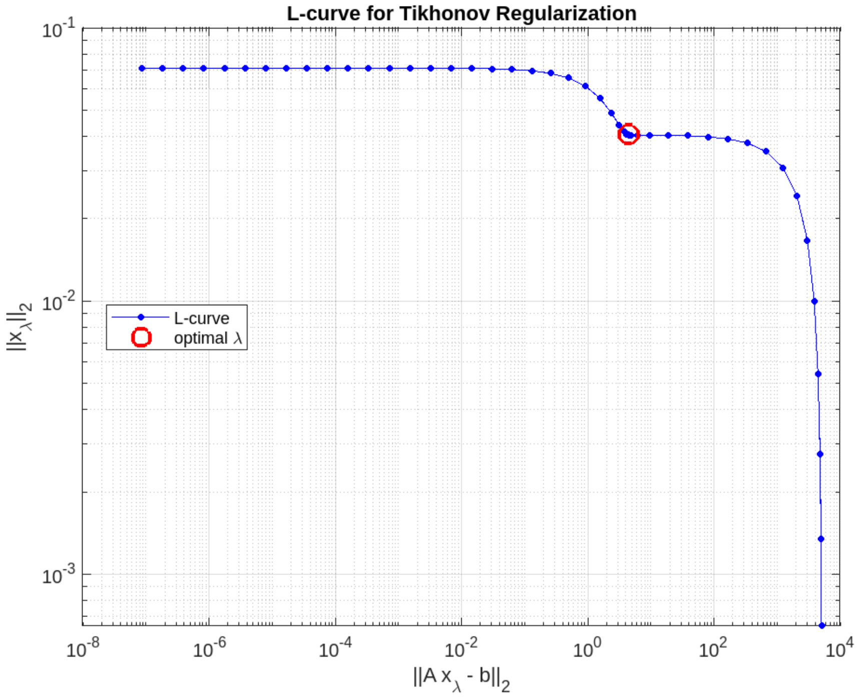

Figure 9: L-curve with speed set to 1 m/s

8 Conclusion

We conclude this work with some considerations on the method used and suggestions on how to improve it, based on the problems encountered during the solution process. From the resolution of a specific case, it emerges that both methods return a good overall solution because, in the heatmaps, the most probable positions are concentrated around the true position of the victim. However, in both cases, the estimated position cannot be considered accurate because, in the heatmap, there are some equiprobable positions to the one returned in the output, but these are geometrically closer to the true victim's position. The reason lies in the fact that, according to the mathematical model used, candidate positions associated with the same tower are indistinguishable in terms of TDoA values. In this way, the accuracy of the solution obtained depends on the distribution of towers in the localization area: if there are many towers and they are well distributed throughout the area, the candidate positions associated with each tower are few. Consequently, the candidate positions are more differentiated. This problem could be addressed, in principle, by developing a more accurate TDoA value model that, for example, depends on the distance from the position to the closest tower, in order to differentiate candidate positions that connect to the same tower. We believe that it would be more effective and easier to switch to another method entirely, for example ToA, could have been more suitable for our use case because it would preserve the distance of the victim from the tower it is connected to.

A Appendix 1: alternative mathematical models

A.1 A more general model

We propose the following, more general, model.

- We adopt Assumption 1, Assumption 2, Assumption 3, Assumption 4.
- τ, on an edge, can be modeled by 3 contributes. For every $(i,j) \in E$

  $$\tau(i,j) = \sigma_s(i,j) + \tilde{\tau}(i,j) + \sigma_r(i,j) \quad \forall\, (i,j) \in E\,,$$

  where
  - $\sigma_s(i,j)$ is the *mean sending time delay from the node i to the node j*: this is the mean time taken to the node i to start sending the message to the node j.
  - $\tilde{\tau}(i,j)$ is the *mean physical travel time*: this is the mean time required for the message to travel from the node i to the node j, without interruptions. In particular $\tilde{\tau}(U, T_U)$, where U is a user of the cellular network (victim or attacker) and T_U is the tower it is connected to, is the distance between U and T_U divided by the speed of light in the air.
  - $\sigma_r(i,j)$ is the *mean receiving time delay from the node i to the node j*: this is the mean time taken to the node j to finish the reception of the message from the node i.
- The victim connects to only one tower and there is an algorithm to find the tower given the victim position.
- The mean travel times can be "split up" over a path adding a delay for each nodes the path goes through: if $i, j \in W$ and exists $k \in W$ such that $(i,k), (k,j) \in E$, then

  $$\tau(i,j) = \tau(i,k) + \delta_k + \tau(k,j)\,,$$

  where δ_k is the *mean transfer delay of the node k* and it is the time between end of the receiving process from a node i with a pending message request to another node j and the start of the sending process to the other node.[12]

We prove that the difference of times for TDoA still depend on the position of the victim.

$$\tau(V, A_i) = \tau(V, T_{i_V}) + \delta_{T_{i_V}} + \tau(T_{i_V}, T_{A_i}) + \delta_{T_{A_i}} + \tau(T_{A_i}, A_i)$$

then

$$\begin{aligned}\tau(V, A_i) - \tau(V, A_j) &= \tau(V, T_{i_V}) + \delta_{T_{i_V}} + \tau(T_{i_V}, T_{A_i}) + \delta_{T_{A_i}} + \tau(T_{A_i}, A_i) - \\ &- \tau(V, T_{i_V}) - \delta_{T_{i_V}} - \tau(T_{i_V}, T_{A_j}) - \delta_{T_{A_j}} - \tau(T_{A_j}, A_j) = \\ &= \tau(T_{i_V}, T_{A_i}) - \tau(T_{i_V}, T_{A_j}) + \tau(T_{A_i}, A_i) - \tau(T_{A_j}, A_j)\,.\end{aligned}$$

[12] Since the GSM standards, towers use a mechanism called *timing advance* to take a bulk of message at the same time, eve if the UEs are in different locations, so when the messages get to the tower can be many and they are not all forwarded to the MBN, implying delay that depends on the tower capacity and network load.

This is enough to prove that difference of times depends on the position of the victim tower and therefore on the position of the victim, so TDoA is applicable, but once again we don't get an explicit dependence from the distance of the victim from the tower. That being sad it's still possible to model $\tau(V, T_{i_V})$ as function of the distance victim-tower

$$\begin{aligned}\tau(V, T_{i_V}) &= \sigma_s(V, T_{i_V}) + \tilde{\tau}(V, T_{i_V}) + \sigma_r(V, T_{i_V}) = \\ &= \sigma_s(V, T_{i_V}) + \frac{\mathrm{d}(V, T_{i_V})}{c} + \sigma_r(V, T_{i_V}) \,.\end{aligned}$$

notice that exists and is unique T_{i_V} because of A.1. We didn't consider this model explicitly in the code, but as we did our numerical solution, and our measurements, it would fit also this more general scenario.

A.2 The model with the least measurement complexity

We want to a different algorithm to measure $\tau(T_i, T_j)$, where T_i, T_j are two different towers, in order to reduce the complexity of the measurements described in subsection 4.1. We assume all the axioms in subsection 3.1 and we add the following

- $$\tau(T_i, T_j) = \tau(T_i, N) + \tau(N, T_j) \,.$$
- $$\tau(T_i, T_j) = \tau(T_j, T_i) \,.$$

This allows for a great improvement in complexity of the measurements of the couples of towers, because just one attacker can measure, for every $i \in \{1, \dots, n_t\}$, the time $\tau(T_i, T_i)$, that is

$$\begin{aligned}\tau(T_i, T_i) &= \tau(T_i, N) + \tau(N, T_i) \,, \\ \tau(T_j, T_j) &= \tau(T_j, N) + \tau(N, T_j) \,,\end{aligned}$$

then

$$\begin{aligned}&\tau(T_i, T_i) + \tau(T_j, T_j) = \tau(T_i, N) + \tau(N, T_i) + \tau(T_j, N) + \tau(N, T_j) = \\ &= \tau(T_i, T_j) + \tau(T_j, T_i) = 2\tau(T_i, T_j)\end{aligned}$$

and the attacker can get all the couples $\tau(T_i, T_j)$ with n_t steps.
But we can't apply TDoA in this scenario, in fact

$$\tau(V, A_i) = \tau(V, T_{i_V}) + \tau(T_{i_V}, T_{A_i}) + \tau(T_{A_i}, A_i) \,,$$

then

$$\begin{aligned}&\tau(V, A_i) - \tau(V, A_j) = \tau(V, T_{i_V}) + \tau(T_{i_V}, T_{A_i}) + \tau(T_{A_i}, A_i) - \tau(V, T_{i_V}) - \tau(T_{i_V}, T_{A_j}) - \tau(T_{A_j}, A_j) = \\ &= \tau(T_{i_V}, T_{A_i}) - \tau(T_{i_V}, T_{A_j}) + \tau(T_{A_i}, A_i) - \tau(T_{A_j}, A_j) = \\ &= \tau(T_{i_V}, N) + \tau(N, T_{A_i}) - \tau(T_{i_V}, N) - \tau(N, T_{A_j}) + \tau(T_{A_i}, A_i) - \tau(T_{A_j}, A_j) = \\ &= \tau(N, T_{A_i}) - \tau(N, T_{A_j}) + \tau(T_{A_i}, A_i) - \tau(T_{A_j}, A_j) \,,\end{aligned}$$

doesn't depend at all on the position of the victim.

References

[WSA03] B.H. Walke, P. Seidenberg, and M.P. Althoff. *UMTS: The Fundamentals*. Wiley, 2003. ISBN: 9780470845578. URL: `https://books.google.it/books?id=KRlUvPWeTYQC`.

[13] "Snowden documents show NSA gathering 5bn cell phone records daily". In: *The Guardian* (Dec. 2013). Accessed: 2025-04-17. URL: `https://www.theguardian.com/world/2013/dec/04/nsa-storing-cell-phone-records-daily-snowden`.

[Zha21] Xian-Da Zhang. *A Matrix Algebra Approach to Artificial Intelligence*. Springer Singapore, 2021. ISBN: 978-981-15-2772-2. URL: `https://doi.org/10.1007/978-981-15-2770-8`.

[Sim23] Adam Simmons. *What is a Cell Tower? Understanding How Cell Towers Work*. Accessed: 2025-04-17. Sept. 2023. URL: `https://dgtlinfra.com/what-is-a-cell-tower-how-work/`.

[BH] João Carlos Alves Barata and Mahir Saleh Hussein. "The Moore–Penrose Pseudoinverse: A Tutorial Review of the Theory". In: ().

DEPARTMENT OF MATHEMATICS
UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
Naples, Italy 2025
www.unina.it

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II