The Random Homotopy Method for Signal Tracking and Global Positioning Systems (GPS)

Pablo V. Negrón–Marrero*
Department of Mathematics
University of Puerto Rico
Humacao, PR 00791-4300

1 Introduction

Global positioning systems have become very popular in the last few years. They are used nowadays for travelling, exploration, navigation, etc.. The 24 satellite system on which GPS systems are based was originally developed for military purposes during the 1980's. A GPS system works as follows: the person interested in determining its location on the earth employs a *receiver* that reads or gets information from the satellite system. The satellites constantly send information on their current times and positions. With this information the receiver can compute the distances to the satellites. On an ideal situation, the receiver can determine its position over the surface of the earth from the information of three satellites, by computing a point on the intersection of three spheres whose radii can be determined from the information received from the satellites and the travelling speed of the signal. However, signal degradation due to atmospheric considerations, differences in the satellite's onboard clocks, and other physical factors, makes it necessary to process the information of at least four satellites.

On a global positioning system, the satellites play a passive rol in the sense that they send information on their current times and position, and is up to the receiver to compute its position. On the other hand in a tracking situation, the "satellites" now assume the rol of receivers or sensors, and only register the times when the signal from the unknown source is received or detected by them. This could be the situation, for example, if one is interested in the location where a shot was fired, or to determine the location of a cellular telephone call. With the times recorded by the sensors, and using the relative time differences among them, one can find the location of the signal.

In this mostly expository paper we discuss the mathematics of a three dimensional GPS system (Section (4)) and that of a two dimensional tracking system (Section (3)).

*pnm@math.uprh.edu

Both problems lead to the consideration of polynomial systems of equations. Thus we give an overview of the theory and practical issues behind one of the most effective methods for solving polynomial systems: the random homotopy method (Section (2)).

Both GPS and tracking system problems are of an inter disciplinary nature. For there successful solution, a good understanding of physics, engineering, and mathematics is required. We discuss some of the physics behind each problem, but emphasize on the mathematics (analytic geometry, algebraic curves) and numerics (homotopy methods) needed to tackle each of them. Thus the material in this paper could be used to motivate further discussions or projects in an applied mathematics course.

2 Some results on Polynomial Systems

Let $p_i : \mathbb{C}^n \to \mathbb{C}$ be a polynomial in the variables $z_1, z_2, \ldots, z_n$ with coefficients in $\mathbb{C}$ and degree d_i, $1 \leq i \leq n$. We define the polynomial vector function $\mathbf{p} : \mathbb{C}^n \to \mathbb{C}^n$ by:

$$\mathbf{p}(\mathbf{z}) = \begin{bmatrix} p_1(z_1, z_2, \ldots, z_n) \\ p_2(z_1, z_2, \ldots, z_n) \\ \vdots \\ p_n(z_1, z_2, \ldots, z_n) \end{bmatrix}$$

We consider the *polynomial system*:

$$\mathbf{p}(\mathbf{z}) = \mathbf{0}. \tag{1}$$

One way to solve this system is using the so called *homotopy methods*. In these methods one construct from or related to the system (1) another system, $\mathbf{e}(\mathbf{z}) = \mathbf{0}$, which is easier, in a certain sense, to solve than (1). Then by suitably "deforming" the solutions of this auxiliary system, we can get the solutions of the original system. This deformation can be carried out using the *homotopy*:

$$\mathbf{h}(\mathbf{z}, t) = (1 - t)\mathbf{e}(\mathbf{z}) + t\mathbf{p}(\mathbf{z}), \quad 0 \leq t \leq 1.$$

Note that

$$\mathbf{h}(\mathbf{z}, 0) = \mathbf{e}(\mathbf{z}), \quad \mathbf{h}(\mathbf{z}, 1) = \mathbf{p}(\mathbf{z}).$$

Suppose that $\mathbf{z}(t)$ is a solution curve of $\mathbf{h}(\mathbf{z}, t) = \mathbf{0}$ and suppose as well that $\mathbf{z}(\cdot)$ is differentiable. Since $\mathbf{h}(\mathbf{z}(t), t) = \mathbf{0}$, we can differentiate with respect to t to get that:

$$\mathbf{h}_{\mathbf{z}}(\mathbf{z}(t), t)\mathbf{z}'(t) + \mathbf{h}_t(\mathbf{z}(t), t) = \mathbf{0}. \tag{2}$$

The *homotopy method* consists basically in solving this differential equation for the curve $\mathbf{z}(\cdot)$ where the initial conditions are given by the roots of $\mathbf{e}(\mathbf{z})$. We now give conditions that guarantee the existence of the curve $\mathbf{z}(t)$.

Theorem 2.1 ([4], [5]). *Consider the polynomial system* (1). *Let*

$$\mathbf{h}(\mathbf{z}, t) = (1-t)\mathbf{e}(\mathbf{z}) + t\mathbf{p}(\mathbf{z}), \quad 0 \leq t \leq 1,$$

where $\mathbf{e}(\mathbf{z}) = (e_1(\mathbf{z}), \ldots, e_n(\mathbf{z}))$ *and*

$$e_k(\mathbf{z}) = a_k z_k^{d_k} - b_k, \quad 1 \leq k \leq n.$$

Hence there exists a set of measure zero $D \subset \mathbb{C}^n \times \mathbb{C}^n$ *such that if* $(\mathbf{a}, \mathbf{b}) \in (\mathbb{C}^n \times \mathbb{C}^n) \setminus D$, *where* $\mathbf{a} = (a_1, \ldots, a_n)$ *and* $\mathbf{b} = (b_1, \ldots, b_n)$, *we have that:*

1. *The set*
$$S = \{(\mathbf{z}, t) \in \mathbb{C}^n \times [0,1] \,:\, \mathbf{h}(\mathbf{z}, t) = \mathbf{0}\}.$$
consists of non-intersecting continuously differentiable curves defined for $t \in [0,1)$. *de curvas diferenciables definidas para* $t \in [0,1)$;

2. *each solution of* $\mathbf{p}(\mathbf{z}) = \mathbf{0}$ *is reached by one of the curves belonging to* S *and starting from one of the solutions of* $\mathbf{e}(\mathbf{z}) = \mathbf{0}$.

The *Bezóut number* of the system (1) is defined by:

$$d = d_1 \times d_2 \times \cdots \times d_n.$$

We now have:

Theorem 2.2 (Bezóut). *The total number of solutions of* (1) *in* $\mathbb{C}^n$ *counting multiplicities is at most* d.

The system (1) is called *defective* if the total number of solutions counting multiplicities is less than the Bezóut number d. Otherwise the system is called *non–defective*.

It follows from De Moivre's Theorem that the system "$\mathbf{e}(\mathbf{z}) = \mathbf{0}$" of Theorem (2.1) is non-defective. Hence if "$\mathbf{p}(\mathbf{z}) = \mathbf{0}$" is defective, then some of the curves in S will never reach $t = 1$. Since these curves are defined for all $t \in [0,1)$, those that never reach $t = 1$ must "blow up" as $t \nearrow 1$.

Example 2.3. If $n = 1$ so that the system (1) reduces to a single polynomial equation, the Bezóut number is just the degree of the polynomial, and by the Fundamental Theorem of Algebra we get that (1) is non–defective in this case. Thus all of the roots of (1) when $n = 1$ can be computed with the homotopy:

$$h(z, t) = (1-t)(az^d + b) + tp(z), \quad 0 \leq t \leq 1.$$

where $a, b \in \mathbb{C}$ are selected at random. The differential equation (2) can now be solved, using as initial conditions the d roots of $e(z) = az^d + b$ which can be computed using De Moivre's Theorem.

Example 2.4. Let A be an $n \times n$ matriz. We consider the eigenvalue problem:

$$A\mathbf{x} = \lambda\mathbf{x}, \quad \mathbf{x} \neq \mathbf{0}.$$

We add a normalization condition on the eigenvector of the form:

$$\mathbf{b}^t\mathbf{x} = 1.$$

If we set $\mathbf{z} = (\lambda, \mathbf{x}) = (\lambda, x_1, x_2, \ldots, x_n)$, then the eigenvalue problem is equivalent to the following polynomial system:

$$\begin{aligned}
p_1(\mathbf{z}) &\equiv a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n - \lambda x_1 = 0, && \text{(3a)}\\
p_2(\mathbf{z}) &\equiv a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n - \lambda x_2 = 0, && \text{(3b)}\\
&\vdots && \text{(3c)}\\
p_n(\mathbf{z}) &\equiv a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n - \lambda x_n = 0, && \text{(3d)}\\
p_{n+1}(\mathbf{z}) &\equiv b_1x_1 + b_2x_2 + \cdots + b_nx_n - 1 = 0. && \text{(3e)}
\end{aligned}$$

This system is defective as its Bezóut number is 2^n while it can have at most n solutions.

To work with defective systems we decompose each $p_k(\mathbf{z})$ as follows: P

$$p_k(\mathbf{z}) = p_k^H(\mathbf{z}) + p_k^R(\mathbf{z}), \quad 1 \leq k \leq n,$$

where p_k^H consists of all of the terms in p_k of degree d_k and $p_k^R = p_k - p_k^H$.

Example 2.5. If $p_1(z_1, z_2) = 3z_1^2 + z_1z_2 + z_1 - 3$, $p_2(z_1, z_2) = z_1^3z_2 - 5z_2$, then we have that:

$$\begin{aligned}
p_1 &= p_1^H + p_1^R, \quad p_1^H(z_1, z_2) = 3z_1^2 + z_1z_2, \quad p_1^R(z_1, z_2) = z_1 - 3,\\
p_2 &= p_2^H + p_2^R, \qquad p_2^H(z_1, z_2) = z_1^3z_2, \qquad p_2^R(z_1, z_2) = -5z_2.
\end{aligned}$$

Note that if

$$\mathbf{p}^H(\mathbf{z}) = (p_1^H(\mathbf{z}), p_2^H(\mathbf{z}), \ldots, p_n^H(\mathbf{z})), \tag{4}$$

then $\mathbf{p}^H(\mathbf{0}) = \mathbf{0}$. We have now:

Theorem 2.6 ([1], [9]). *If $\mathbf{z} = \mathbf{0}$ is the only solution of (4), then the system (1) is non–defective, i.e., it has exactly the Bezóut number of solutions counting multiplicities.*

The non-zero solutions of (4) are called the *roots or zeroes at infinity* of (1). For system having roots at infinity, like the one for the eigenvalue problem, the homotopy in Theorem (2.1) is not appropriate as many of the curves will diverge to "infinity" as $t \nearrow 1$. To work with defective systems the polynomial $\mathbf{e}(\mathbf{z})$ should have at least the same zeroes at infinity as (1). This goal can be accomplished by means of the so–called *random product homotopy* [5]. We will not go into the details of this method but we will illustrate it for the eigenvalue problem.

Example 2.7. For the system (3) of the eigenvalue problem we have that:

$$\begin{aligned} p_1^H(\mathbf{z}) &\equiv -\lambda x_1 = 0, \\ p_2^H(\mathbf{z}) &\equiv -\lambda x_2 = 0, \\ &\vdots \\ p_n^H(\mathbf{z}) &\equiv -\lambda x_n = 0, \\ p_{n+1}^H(\mathbf{z}) &\equiv b_1 x_1 + b_2 x_2 + \cdots + b_n x_n = 0. \end{aligned}$$

The roots at infinity are thus given by:

$$\{(\lambda, \mathbf{0}) \,:\, \lambda \in \mathbb{C}\} \cup \left\{(0, \mathbf{x}) \,:\, \mathbf{b}^t \mathbf{x} = 0\right\}.$$

A suitable random product homotopy for this problem is given by:

$$\begin{aligned} e_1(\mathbf{z}) &\equiv (\alpha_1 - \lambda)(x_1 + \beta_1) = 0, \\ e_2(\mathbf{z}) &\equiv (\alpha_2 - \lambda)(x_2 + \beta_2) = 0, \\ &\vdots \\ e_n(\mathbf{z}) &\equiv (\alpha_n - \lambda)(x_n + \beta_n) = 0, \\ e_{n+1}(\mathbf{z}) &\equiv b_1 x_1 + b_2 x_2 + \cdots + b_n x_n + \beta_{n+1} = 0, \end{aligned}$$

where the α's and β's are chosen at random but subject to,

$$\alpha_i \neq \alpha_j, \quad \beta_i \neq \beta_j \quad \forall\, i, j, \quad b_1\beta_1 + b_2\beta_2 + \cdots + b_n\beta_n + \beta_{n+1} \neq 0.$$

The system $\mathbf{e}(\mathbf{z}) = \mathbf{0}$ has the same roots at infinity as (3) and has exactly n roots or solutions given by:

$$(\lambda, \mathbf{x}) = (-\alpha_i, -\beta_1, -\beta_2, \ldots, -\beta_{i-1}, h_i, -\beta_{i+1}, \ldots, -\beta_n),$$

where $b_i h_i = -\beta_{n+1} - \sum_{j \neq i} b_j \beta_j$.

3 A Two–Dimensional Signal Tracking System

We discuss now an idealized two–dimensional signal tracking system. In this problem or situation, a source transmit or sends a signal, but does not broadcast any information about itself. One still needs to determine the location of the source. In the idealized situation in which no obstacles or signal degradation are considered, one can track the source from the times of receipt of the signal from three sensors. The general situation is best described by the following scenario. Suppose a gun is fired at an unknown location, and that three sensors, located in known positions, register the times when they detect the sound emitted by the gun. From this information we want to determine the unknown location.

We let (x_0, y_0) be the unknown location at which the gun was fired and (x_i, y_i), $i = 1, 2, 3$ be the known positions of the three sensors. Let d_i be the distance from the i–th sensor to the point (x_0, y_0), $i = 1, 2, 3$. We then have that

$$\begin{aligned}\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} &= d_1,\\ \sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2} &= d_2,\\ \sqrt{(x_3 - x_0)^2 + (y_3 - y_0)^2} &= d_3.\end{aligned}$$

Note that the d_i's are unknown. However, if v is the speed of a sound wave and t_i is the time registered by the i–th sensor, $i = 1, 2, 3$, then

$$d_2 - d_1 = v(t_2 - t_1) \equiv v_{12}, \quad d_3 - d_1 = v(t_3 - t_1) \equiv v_{13},$$

are known quantities. It follows now from the system above that

$$\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2} - \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} = v_{12}, \tag{5a}$$
$$\sqrt{(x_3 - x_0)^2 + (y_3 - y_0)^2} - \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} = v_{13}. \tag{5b}$$

Note that if we apply absolute value on both sides of these equations, we get the equations of two hyperbolas (possibly degenerate), the first one with foci at (x_1, y_1), (x_2, y_2), and the second one with foci at (x_1, y_1), (x_3, y_3).

We can eliminate the square roots from equations (5) to get the following system of equations for the unknown point (x_0, y_0):

$$A_{12}x_0^2 + B_{12}x_0y_0 + C_{12}y_0^2 + D_{12}x_0 + E_{12}y_0 + F_{12} = 0, \tag{6a}$$
$$A_{13}x_0^2 + B_{13}x_0y_0 + C_{13}y_0^2 + D_{13}x_0 + E_{13}y_0 + F_{13} = 0, \tag{6b}$$

where

$$A_{ij} = 4(v_{ij}^2 - (x_i - x_j)^2), \tag{7a}$$
$$B_{ij} = -8(x_i - x_j)(y_i - y_j), \tag{7b}$$
$$C_{ij} = 4(v_{ij}^2 - (y_i - y_j)^2), \tag{7c}$$
$$D_{ij} = 4((x_i - x_j)(x_i^2 + y_i^2 - x_j^2 - y_j^2) - (x_i + x_j)v_{ij}^2), \tag{7d}$$
$$E_{ij} = 4((y_i - y_j)(x_i^2 + y_i^2 - x_j^2 - y_j^2) - (y_i + y_j)v_{ij}^2), \tag{7e}$$
$$F_{ij} = -(x_i^2 + y_i^2 - x_j^2 - y_j^2)^2 + 2v_{ij}^2(x_i^2 + y_i^2 + x_j^2 + y_j^2) - v_{ij}^4, \tag{7f}$$

where $i = 1$, $j = 2, 3$. One can easily check that

$$\Delta_{ij} \equiv B_{ij}^2 - 4A_{ij}C_{ij} = 64v_{ij}^2((x_i - x_j)^2 + (y_i - y_j)^2 - v_{ij}^2).$$

On the other hand, by geometrical considerations we have that

$$v_{ij}^2 \leq (x_i - x_j)^2 + (y_i - y_j)^2.$$

It follows now that

$$\Delta_{ij} \geq 0, \tag{8}$$

from which we can conclude that each equation in (6) represents either a hyperbola (when we have strictly positive in the inequality above) or a parabola (when the expression above is identically zero). We consider now what happens in each of these cases.

1. **Case** $\Delta_{ij} = 0$: Note that $\Delta_{ij} = 0$ if and only if the three points (x_i, y_i), (x_j, y_j) and (x_0, y_0) are co–linear. Thus if the (x_i, y_i), $i = 1, 2, 3$ are not co–linear, we have that at least one of the equations in (6) represents a hyperbola. The other equation is given by the equation of the line joining (x_i, y_i), (x_j, y_j). Upon eliminating one of the variables and substituting into the other equation (the hyperbola), we get a quadratic in one variable whose roots can be easily computed. Thus in these case we get two solution pairs for (x_0, y_0), possibly complex, of multiplicity two each.

2. **Case** $\Delta_{ij} > 0$ **for all** i, j: The homogenous part of the system (6) is given by

$$A_{12}x_0^2 + B_{12}x_0y_0 + C_{12}y_0^2 = 0, \tag{9a}$$
$$A_{13}x_0^2 + B_{13}x_0y_0 + C_{13}y_0^2 = 0. \tag{9b}$$

   Let ξ_1, ξ_2 and η_1, η_2 be the roots respectively of the quadratics:

$$A_{12}\xi^2 + B_{12}\xi + C_{12} = 0, \quad A_{13}\eta^2 + B_{13}\eta + C_{13} = 0.$$

   It follows from (8) (with strict inequality) that these roots are all real with $\xi_1 \neq \xi_2$ and $\eta_1 \neq \eta_2$. Hence the solutions of (9) are given by:

$$\{(x_0, y_0) \,:\, x_0 = \xi_1 y_0 \quad \text{or} \quad x_0 = \xi_2 y_0\} \cap \{(x_0, y_0) \,:\, x_0 = \eta_1 y_0 \quad \text{or} \quad x_0 = \eta_2 y_0\}$$

   If none of the ξ_i's equal any of the η_j's, then this set contains only the origin, which implies that the system (6) is non–defective. Hence it has exactly four solutions in $\mathbb{C}^2$.

We implemented in MATLAB the homotopy method described in Section (2) to solve (6). The initial value problems for (2) were solved using the `ode45` package. In our computations we worked with the homotopy function of Theorem (2.1) as Case 2 described above is generic. In the following numerical example, the sensor coordinates are at:

$$(x_1, y_1) = (0, 1), \quad (x_2, y_2) = (3, 0), \quad (x_3, y_3) = (5, 3). \tag{10}$$

For the purpose of testing the code, we took the "unknown" location at $(0, -1)$ and input into the code the relative displacements or distances v_{12}, v_{13}. The results are shown in Table (1). We see that the numerics capture back the position of the unknown location at $(0, -1)$ (the last line in the table), and also points out another possible location at $(0.1094, 1.659)$ approximately (third line in the table). We also get two complex solutions which have no physical meaning. We show in Figure (1) a sketch of each of the hyperbolas (equations) in the system (6). The intersections of these graphs are precisely the unknown locations or sources which correspond to the real roots computed by the program. In the Appendix we list the sources of the codes used for the simulations.

x		y	
Real Part	Imaginary Part	Real Part	Imaginary Part
1.30729000e+000	1.63906553e-001	1.11445429e+000	1.49989127e+000
1.30729000e+000	-1.63906553e-001	1.11445429e+000	-1.49989127e+000
4.61405722e+000	1.56599859e-011	4.07377367e+000	1.96078249e-011
-3.25274746e-014	3.28549904e-013	-1.00000000e+000	1.16818924e-012

Table 1: The roots of the system (6) computed by the homotopy method for the placement of the sensors at (10).

4 A Three–Dimensional Global Positioning System

A GPS system works as follows: the person interested in determining its location on the earth employs a *receiver* that reads or gets information from the satellite system. The satellites constantly send information on their current times and positions. With this information the receiver can compute the distances to the satellites. On an ideal situation, the receiver can determine its position over the surface of the earth from the information of three satellites, by computing a point on the intersection of three spheres whose radii can be determined from the information received from the satellites and the travelling speed of the signal. We show that if signal degradation due to atmospheric considerations, differences in the satellite's onboard clocks, and other physical factors are discarded, then the location of the receiver can be computed from the information of at least four satellites.

We let (x_i, y_i, z_i), $i = 1, \ldots, 4$ denote the position of the four receiving satellites, and (x, y, z) is the (unknown) position of the receiver. Let t_i be the time elapsed for the signal to travel from the i–th satellite. Note that t_i includes atmospheric delays and differences on the satellite's onboard clocks with respect to standard GPS time. We let w represent the difference between the receiver's clock and standard GPS time. Then the unknown position (x, y, z) lies in the intersection of the following four spheres (see [3], [6], [8]):

$$\begin{aligned}
(x-x_1)^2 + (y-y_1)^2 + (z-z_1)^2 &= c^2(t_1+w)^2, && \text{(11a)}\\
(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2 &= c^2(t_2+w)^2, && \text{(11b)}\\
(x-x_3)^2 + (y-y_3)^2 + (z-z_3)^2 &= c^2(t_3+w)^2, && \text{(11c)}\\
(x-x_4)^2 + (y-y_4)^2 + (z-z_4)^2 &= c^2(t_4+w)^2, && \text{(11d)}
\end{aligned}$$

where c is the speed of light in a vacuum. If we subtract the second, third and fourth equations from the first one, we get that this system is equivalent to

$$\begin{aligned}
(x-x_1)^2 + (y-y_1)^2 + (z-z_1)^2 - c^2(t_1+w)^2 &= 0, && \text{(12a)}\\
(x_1-x_2)x + (y_1-y_2)y + (z_1-z_2)z + c^2(t_1-t_2)w &= \alpha_{12}, && \text{(12b)}\\
(x_1-x_3)x + (y_1-y_3)y + (z_1-z_3)z + c^2(t_1-t_3)w &= \alpha_{13}, && \text{(12c)}\\
(x_1-x_4)x + (y_1-y_4)y + (z_1-z_4)z + c^2(t_1-t_4)w &= \alpha_{14}, && \text{(12d)}
\end{aligned}$$

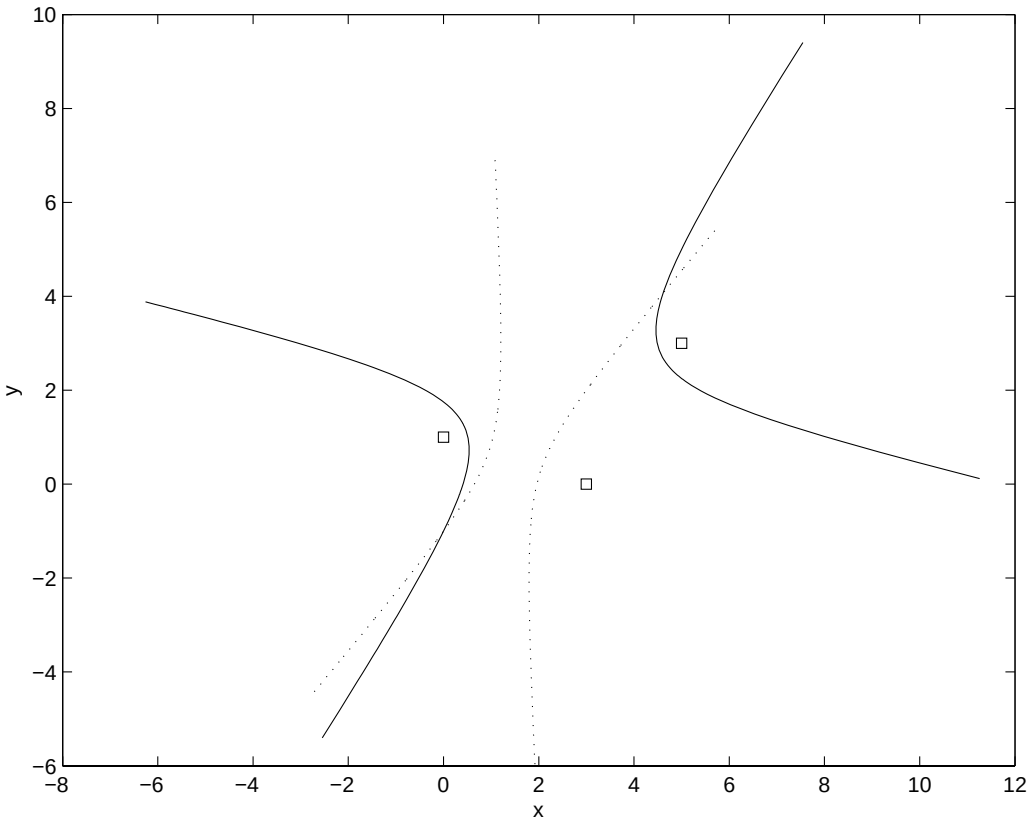

Figure 1: Graphs of the each of the hyperbolas (equations) in the system (6) for the case (10). The location of the sensors are given by the squares.

where

$$2\alpha_{ij} = x_i^2 + y_i^2 + z_i^2 - c^2t_i^2 - (x_j^2 + y_j^2 + z_j^2 - c^2t_j^2), \quad i = 1, j = 2, 3, 4.$$

The homogeneous part of this system is given by:

$$\begin{aligned} x^2 + y^2 + z^2 - c^2w^2 &= 0, && \text{(13a)} \\ (x_1 - x_2)x + (y_1 - y_2)y + (z_1 - z_2)z + c^2(t_1 - t_2)w &= 0, && \text{(13b)} \\ (x_1 - x_3)x + (y_1 - y_3)y + (z_1 - z_3)z + c^2(t_1 - t_3)w &= 0, && \text{(13c)} \\ (x_1 - x_4)x + (y_1 - y_4)y + (z_1 - z_4)z + c^2(t_1 - t_4)w &= 0. && \text{(13d)} \end{aligned}$$

Note that generically, the last three equations in the system (13) give that x, y, z are proportional to w. Upon substitution on the first equation, we then get that generically, the only solution of the system (13) is the trivial one, that is $x = y = z = w = 0$. Hence the system (12) or equivalently (11), generically has exactly two solutions in $\mathbb{C}^4$.

We implemented two methods for approximating solutions of the system (12): the homotopy method as described in Section (2); and what we call, the *direct method* which

is basically the procedure described above to show that the only solution of (13) is the trivial one. In the direct method, the last three equations in (12) are solved for x, y, z in terms of w. These expressions are plugged back into the first equation which becomes now a quadratic equation for w which can be easily solved for two, possibly different, values of w. We then can get the corresponding x, y, z from the corresponding expressions in terms of w. The condition under which the last three equations in (12) can be solved are given by the following lemma whose proof follows by inspection.

Lemma 4.1. *The matrix*

$$\begin{pmatrix} x_1 - x_2 & y_1 - y_2 & z_1 - z_2 \\ x_1 - x_3 & y_1 - y_3 & z_1 - z_3 \\ x_1 - x_4 & y_1 - y_4 & z_1 - z_4 \end{pmatrix} \tag{14}$$

is nonsingular if and only if no subset of three of the points (x_1, y_i, z_i), $i = 1, 2, 3, 4$, *is collinear.*

For the simulations we used the following data as quoted in [6] where the units are in *km* for distances and *sec* for times:

$$\begin{aligned}
x_1 &= 1.876371950559744 \times 10^3, \quad y_1 = -1.064143413406656 \times 10^4, \\
z_1 &= 2.42697646566144 \times 10^4, \quad t_1 = .07234683200, \\
x_2 &= 1.097666464137408 \times 10^4, \quad y_2 = -1.308147952230029 \times 10^4, \\
z_2 &= 2.035116937827073 \times 10^4, \quad t_2 = .06730845726, \\
x_3 &= 2.458513954435968 \times 10^4, \quad y_3 = -4.335023426659201 \times 10^3, \\
z_3 &= 9.08630032021747 \times 10^3, \quad t_3 = .06738499643, \\
x_4 &= 3.854136195752833 \times 10^3, \quad y_4 = 7.248575943442946 \times 10^3, \\
z_4 &= 2.526630462778753 \times 10^4, \quad t_4 = .07651971839, \\
c &= 2.99792458 \times 10^5.
\end{aligned}$$

The solutions computed using the direct method are shown in Table (2). One of the solutions (Solution 2) represents a point over the surface of the earth, using an approximate earth radius of 6,370 *km*, while the other one gives a point at approximately 4,800 *km* over the surface of the earth. A standard GPS receiver would convert the cartesian coordinates in Table (2) to spherical coordinates using the corresponding transformation formulas.

The solutions computed by the homotopy method are similar to those in Table (2) with some differences in the last three to four digits. These differences are due to the usual accumulation of errors in IVP solvers as the computation of the solution curve advances. In this particular problem however, the direct method was on average about a hundred times faster than the homotopy method. In the Appendix we list the sources of the codes used for the simulations.

	Solution 1	Solution 2
x	-8.00194042987681e+3	4.57611815228886e+3
y	3.71732980095105e+3	-2.31825073009694e+3
z	-6.85615532540829e+3	3.77053178592283e+3
w	-0.19133985872	0.00200000001

Table 2: The roots of the system (12) computed by the direct method for the data given below.

It is well known that the relative satellite geometry affects the computation of the receiver's position. This is because this relative geometry determines the conditioning of the matrix (14). We prove now the well known fact that the optimal satellite geometry is the one with one satellite in the user's zenith and three satellites with evenly spaced azimuths on the user's horizon. Let

$$\mathbf{a}_i = (x_i, y_i, z_i), \quad i = 1, 2, 3, 4, \quad \mathbf{u}_1 = \mathbf{a}_1 - \mathbf{a}_i, \quad i = 1, 2, 3.$$

Note that the $\{\mathbf{u}_i\}$ are precisely the rows of (14). We may assume without loss of generality, that the coordinate axes have been chosen so that the origin is at $\mathbf{a}_1$, $\mathbf{u}_1$ has terminal point on the x axis, and that $\mathbf{u}_1, \mathbf{u}_2$ both lie on the xy plane. If $(\rho_i, \theta_i, \phi_i)$ are the spherical coordinates of $\mathbf{u}_i$ in this coordinate system, then

$$\theta_1 = 0, \quad \phi_1 = \phi_2 = \pi/2.$$

Thus in this coordinate system, the matrix (14) has the form:

$$\begin{pmatrix} \rho_1 & 0 & 0 \\ \rho_2 \cos\theta_2 & \rho_2 \sin\theta_2 & 0 \\ \rho_3 \cos\theta_3 \sin\phi_3 & \rho_3 \sin\theta_3 \sin\phi_3 & \rho_3 \cos\phi_3 \end{pmatrix},$$

which has eigenvalues ρ_1, $\rho_2 \sin\theta_2$, $\rho_3 \cos\phi_3$. Thus the optimal condition number, the ratio of the maximum eigenvalue to the minimum, is achieved when $\rho_1 = \rho_2 = \rho_3$, $\theta_2 = \pi/2$ and $\phi_3 = 0$, which corresponds to the points arranged in a regular pyramid with vertex at $\mathbf{a}_1$, the base forming an equilateral triangle, and lateral faces forming isosceles right triangles. In practice due to the curvature of the earth and satellite disposition this configuration is not achievable, but the selection of one satellite in the user's zenith and three satellites with evenly spaced azimuths on the user's horizon is the best possible approximation.

5 Final Remarks

Our discussion of global positioning and tracking systems has been an ideal one in the sense that we have not considered the sources of error that normally affect the accurate computation of the desired location. The actual situation is that if one performs

successive readings with a receiver from a fixed position at different times, the positions computed or determined by the device vary with time. The three main sources of error in these systems are due to poor clock synchronization, variability of the atmospheric factors, and error in the information on the positions of the satellites.

The errors caused by these conditions are random in nature. By modelling these errors with normal distributions, one could estimate the radius d_{50} and d_{95} of the circles around a given fixed position, such that 50% and 95% respectively of the computed positions, lie within these circles. Commercially available GPS systems are designed such that $d_{50} = 40$ meters and $d_{95} = 100$ meters. By employing a technique called Precise Positioning Systems (PPS), which processes signals from the satellites at two frequencies, one can get d_{50} down to about 16 meters. Another technique for improving GPS precision is called Differential Global Positioning System (DGPS). This technique employs a fixed base station with known position. After reading its GPS position, the base station can send calibrating information to neighboring or local receivers, which then use this information to correct their actual GPS readings. DGPS are used in small geographical areas and can give d_{50} of about 9 meters. For further discussions of sources of errors in GPS systems, and PPS and DGPS systems, see [2], [3], [8].

In the previous section we proved that the optimal satellite geometry is the one with one satellite in the user's zenith and three satellites with evenly spaced azimuths on the user's horizon. When a receiver processes data from satellites whose geometric configuration differs greatly from the optimal one, the matrix in (14) becomes ill–conditioned and any errors present in the data can be magnified during the computations. This source of error is called geometric dilution of position (GDOP). In [6] an argument using the Implicit Function Theorem directly applied to the system (11) is given that shows that at least eight significant digits of precision are required in the time readings from the satellites for 100 meter accuracy in GPS systems[1].

It is very common that the receiver receives information from more than four satellites. In this case the resulting polynomial system is likely to be inconsistent. However it is not a good practice to discard the extra information, and a"generalized" solution is still computed using a nonlinear least square method. It may happen as well, that information from less than four satellites needs to be processed. In this case some additional information could be used to compute the position of the receiver, like if it is known that one it at sea level, say.

Acknowledgement: This research was sponsored in part by the National Security Agency (NSA) under grant number H98230-04-C-0486.

[1]We believe the authors of this paper are referring to d_{50} here.

References

[1] C. B. Garcia and T. Y. Li. On Number of Solutions to Polynomial Systems of Equations. *SIAM J. Numer. Anal.*, 17, 540–546, 1980.

[2] B. Hoffmann–Wellenhof, H. Lichtenegger, and J. Collins. Global Po-sitioning System: Theory and Practice. 5th Edition, Springer–Verlag, 1998.

[3] R. B. Langley. The Mathematics of GPS. *GPS World*, July/August, 1991.

[4] T. Y. Li and T. Sauer. Regularity Results for Solving Systems of Polynomials by Homotopy Method. *Numer. Math.*, 50, 283–289, 1987.

[5] T. Y. Li, T. Sauer, and J. A. Yorke. The Random Product Homotopy and Deficient Polynomial Systems. *Numer. Math.*, 51, 481–500, 1987.

[6] G. Nord, D. Jabon, and J. Nord. The Global Positioning System and the Implicit Function Theorem. *SIAM Review*, Vol. 40, No. 3, 692–696, 1998.

[7] G. Strang, K. Borre, and K. Borre. Linear Algebra, Geodesy, and GPS. Wellesley-Cambridge Press, 1997.

[8] R. B. Thompson. Global Positioning System: The Mathematics of GPS Receivers. *Mathematics Magazine*, 71(4), 260–269, 1998.

[9] B. L. van der Waerden. Die Alternative bei nichtlinearen Gleichungen. *Nachrichten der Gesellschaft der Wissenschaften zu Göttingen*, Math. Phys. Klasse, 77–87, 1928.

A MATLAB Programs

Code implementing the homotopy method for the two dimensional tracking system, without graphing routines.

```
p1=[0,1]; p2=[3,0]; p3=[5,3]; psol=[0,-1];
%
v12=norm(p2-psol)-norm(p1-psol); v13=norm(p3-psol)-norm(p1-psol);
%
[raices,Errors]=gps_homotopy(p1,p2,p3,v12,v13);

function [raices,H_vals]=gps_homotopy(p1,p2,p3,v12,v13)

x1=p1(1);y1=p1(2); x2=p2(1);y2=p2(2); x3=p3(1);y3=p3(2);

A12=-4*((x1-x2)^2-v12^2); B12=-8*(x1-x2)*(y1-y2);
C12=-4*((y1-y2)^2-v12^2);
D12=4*((x1-x2)*(x1^2+y1^2-(x2^2+y2^2))-v12^2*(x1+x2));
E12=4*((y1-y2)*(x1^2+y1^2-(x2^2+y2^2))-v12^2*(y1+y2));
F12=-(x1^2+y1^2-(x2^2+y2^2))^2+2*v12^2*(x1^2+y1^2+x2^2+y2^2)-v12^4;

A13=-4*((x1-x3)^2-v13^2); B13=-8*(x1-x3)*(y1-y3);
C13=-4*((y1-y3)^2-v13^2);
D13=4*((x1-x3)*(x1^2+y1^2-(x3^2+y3^2))-v13^2*(x1+x3));
E13=4*((y1-y3)*(x1^2+y1^2-(x3^2+y3^2))-v13^2*(y1+y3));
F13=-(x1^2+y1^2-(x3^2+y3^2))^2+2*v13^2*(x1^2+y1^2+x3^2+y3^2)-v13^4;

i=sqrt(-1); n=4;

a1=rand+rand*i; b1=rand+rand*i; w1=-b1/a1; r1=abs(w1);
theta1=angle(w1); xe(1)=r1^(1/2)*exp(theta1*i/2);
xe(2)=r1^(1/2)*exp((theta1+2*pi)*i/2);

a2=rand+rand*i; b2=rand+rand*i; w2=-b2/a2; r2=abs(w2);
theta2=angle(w2); ye(1)=r2^(1/2)*exp(theta2*i/2);
ye(2)=r2^(1/2)*exp((theta2+2*pi)*i/2);

y0=zeros(4,2); y0(1,:)=[xe(1),ye(1)]; y0(2,:)=[xe(1),ye(2)];
y0(3,:)=[xe(2),ye(1)]; y0(4,:)=[xe(2),ye(2)];

raices=zeros(n,3); H_vals=zeros(n,2); options =
odeset('RelTol',1e-10,'AbsTol',[1e-10 1e-10]);

for k=1:n [t_sol,y_sol] = ode45(@ediff,[0,1],y0(k,:),options);
m=size(y_sol,1); raices(k,:)=[t_sol(m),y_sol(m,:)];
H_vals(k,:)=H(t_sol(m),y_sol(m,:)); end
```

```
function yp=ediff(t,w) x=w(1); y=w(2);

Ht=[-(a1*x^2+b1)+A12*x^2+B12*x*y+C12*y^2+D12*x+E12*y+F12;...
    -(a2*y^2+b2)+A13*x^2+B13*x*y+C13*y^2+D13*x+E13*y+F13];
Hz=[2*(1-t)*a1*x+t*(2*A12*x+B12*y+D12),t*(B12*x+2*C12*y+E12);...
    t*(2*A13*x+B13*y+D13),2*(1-t)*a2*y+t*(B13*x+2*C13*y+E13)];

yp=-Hz\Ht;

end

function yp=H(t,w) x=w(:,1); y=w(:,2);

yp=[(1-t).*(a1*x.^2+b1)+t.*(A12*x.^2+B12*x.*y+C12*y.^2+D12*x+E12*y+F12),...
   (1-t).*(a2*y.^2+b2)+t.*(A13*x.^2+B13*x.*y+C13*y.^2+D13*x+E13*y+F13)];
end

end
```

Codes for the homotopy and direct methods for the solution of the GPS system (12).

```
P=[1.876371950559744e+3,-1.064143413406656e+4,2.42697646566144e+4;
    1.097666464137408e+4,-1.308147952230029e+4,2.035116937827073e+4;
    2.458513954435968e+4,-4.335023426659201e+3,9.08630032021747e+3;
    3.854136195752833e+3,7.248575943442946e+3,2.526630462778753e+4];
T=[.07234683200,.06730845726,.06738499643,.07651971839];
c=2.99792458e+5;
%
t1=cputime; [raices_h,Errors_h]=gps3_homotopy(P,T,c);
t_homo=cputime-t1; t1=cputime;
[raices_s,Errors_s]=gps3_solver(P,T,c); t_solver=cputime-t1;
disp(['Time for homotopy method:  ',num2str(t_homo,6),' secs'])
disp(['Time for direct solver:  ',num2str(t_solver,6),' secs'])

function [raices,H_vals]=gps3_homotopy(P,T,c)

x1=P(1,1);y1=P(1,2);z1=P(1,3); x2=P(2,1);y2=P(2,2);z2=P(2,3);
x3=P(3,1);y3=P(3,2);z3=P(3,3); x4=P(4,1);y4=P(4,2);z4=P(4,3);
t1=T(1);t2=T(2);t3=T(3);t4=T(4);

A12=x1^2+y1^2+z1^2-c^2*t1^2-(x2^2+y2^2+z2^2-c^2*t2^2);
A13=x1^2+y1^2+z1^2-c^2*t1^2-(x3^2+y3^2+z3^2-c^2*t3^2);
A14=x1^2+y1^2+z1^2-c^2*t1^2-(x4^2+y4^2+z4^2-c^2*t4^2);

i=sqrt(-1); n=2;
```

```
a=rand(1,4)+rand(1,4)*i; b=10^4*(rand(1,4)+rand(1,4)*i);
w1=-b(1)/a(1); r1=abs(w1); theta1=angle(w1);
xe(1)=r1^(1/2)*exp(theta1*i/2);
xe(2)=r1^(1/2)*exp((theta1+2*pi)*i/2);

y0=zeros(2,4); y0(1,:)=[xe(1),-b(2)/a(2),-b(3)/a(3),-b(4)/a(4)];
y0(2,:)=[xe(2),-b(2)/a(2),-b(3)/a(3),-b(4)/a(4)];

raices=zeros(n,5); H_vals=zeros(n,4); options =
odeset('RelTol',1e-10,'AbsTol',1e-10);

for k=1:n [t_sol,y_sol] = ode45(@ediff3,[0,1],y0(k,:),options);
m=size(y_sol,1); raices(k,:)=[t_sol(m),y_sol(m,:)];
H_vals(k,:)=H3(t_sol(m),y_sol(m,:)); end

function yp=ediff3(t,ww) x=ww(1); y=ww(2); z=ww(3); w=ww(4);

Ht=[-(a(1)*x^2+b(1))+(x-x1)^2+(y-y1)^2+(z-z1)^2-c^2*(t1+w)^2;...
    -(a(2)*y+b(2))+2*(x1-x2)*x+2*(y1-y2)*y+2*(z1-z2)*z+2*c^2*(t1-t2)*w-A12;...
    -(a(3)*z+b(3))+2*(x1-x3)*x+2*(y1-y3)*y+2*(z1-z3)*z+2*c^2*(t1-t3)*w-A13;...
    -(a(4)*w+b(4))+2*(x1-x4)*x+2*(y1-y4)*y+2*(z1-z4)*z+2*c^2*(t1-t4)*w-A14];

Hz=[2*(1-t)*a(1)*x+2*t*(x-x1),2*t*(y-y1),2*t*(z-z1),-2*t*c^2*(t1+w);...
    2*t*(x1-x2),(1-t)*a(2)+2*t*(y1-y2),2*t*(z1-z2),2*t*c^2*(t1-t2);...
    2*t*(x1-x3),2*t*(y1-y3),(1-t)*a(3)+2*t*(z1-z3),2*t*c^2*(t1-t3);...
    2*t*(x1-x4),2*t*(y1-y4),2*t*(z1-z4),(1-t)*a(4)+2*t*c^2*(t1-t4)];

yp=-Hz\Ht;

end

function yp=H3(t,ww) x=ww(:,1); y=ww(:,2); z=ww(:,3); w=ww(:,4);

yp=[(1-t).*(a(1)*x.^2+b(1))+t.*((x-x1)^2+(y-y1)^2
                                        +(z-z1)^2-c^2*(t1+w)^2),...
   (1-t).*(a(2)*y+b(2))+t.*(2*(x1-x2)*x+2*(y1-y2)*y
                                        +2*(z1-z2)*z+2*c^2*(t1-t2)*w-A12),...
   (1-t).*(a(3)*z+b(3))+t.*(2*(x1-x3)*x+2*(y1-y3)*y
                                        +2*(z1-z3)*z+2*c^2*(t1-t3)*w-A13),...
   (1-t).*(a(4)*w+b(4))+t.*(2*(x1-x4)*x+2*(y1-y4)*y
                                        +2*(z1-z4)*z+2*c^2*(t1-t4)*w-A14)];
end

end
```

```
function [raices,H_vals]=gps3_solver(P,T,c)

x1=P(1,1);y1=P(1,2);z1=P(1,3); x2=P(2,1);y2=P(2,2);z2=P(2,3);
x3=P(3,1);y3=P(3,2);z3=P(3,3); x4=P(4,1);y4=P(4,2);z4=P(4,3);
t1=T(1);t2=T(2);t3=T(3);t4=T(4);

A12=x1^2+y1^2+z1^2-c^2*t1^2-(x2^2+y2^2+z2^2-c^2*t2^2);
A13=x1^2+y1^2+z1^2-c^2*t1^2-(x3^2+y3^2+z3^2-c^2*t3^2);
A14=x1^2+y1^2+z1^2-c^2*t1^2-(x4^2+y4^2+z4^2-c^2*t4^2);

AA=[2*(x1-x2),2*(y1-y2),2*(z1-z2),-2*c^2*(t1-t2),A12;...
    2*(x1-x3),2*(y1-y3),2*(z1-z3),-2*c^2*(t1-t3),A13;...
    2*(x1-x4),2*(y1-y4),2*(z1-z4),-2*c^2*(t1-t4),A14];
BB=rref(AA); alpha=BB(:,4);beta=BB(:,5); A=alpha'*alpha-c^2;
B=2*(alpha'*(beta-P(1,:)')-c^2*t1);
C=(beta'-P(1,:))*(beta-P(1,:)')-c^2*t1^2; Disc=sqrt(B^2-4*A*C);
w1=(-B+Disc)/(2*A); w2=(-B-Disc)/(2*A);

raices=[w1*alpha'+beta',w1;w2*alpha'+beta',w2]; H_vals=H3(raices);

function yp=H3(ww) x=ww(:,1); y=ww(:,2); z=ww(:,3); w=ww(:,4);

yp=[(x-x1).^2+(y-y1).^2+(z-z1).^2-c^2*(t1+w).^2,...
    2*(x1-x2)*x+2*(y1-y2)*y+2*(z1-z2)*z+2*c^2*(t1-t2)*w-A12,...
    2*(x1-x3)*x+2*(y1-y3)*y+2*(z1-z3)*z+2*c^2*(t1-t3)*w-A13,...
    2*(x1-x4)*x+2*(y1-y4)*y+2*(z1-z4)*z+2*c^2*(t1-t4)*w-A14];
end

end
```