Determined optimization technique for solving over-determined linear systems

Akala[1], A. O., Adepoju[2], J. A., Adeloye[1], A. B., Somoye[3], E. O., Oyebola[1], O. O., Oyeyemi[1], E. O., Olopade[1], M. O., Adewale[1], A. O., Yinka-Banjo[4], C. and Ojiako[4], C.

[1]*Department of (Applied) Physics, University of Lagos, Akoka, Yaba, Lagos, Nigeria.*
[2]*Department of Mathematics, University of Lagos, Akoka, Yaba, Lagos, Nigeria.*
[3]*Department of Physics, Lagos State University, Ojo, Lagos, Nigeria.*
[4]*Department of Computer Science, University of Lagos, Akoka, Yaba, Lagos, Nigeria.*

E-mail: akalaovie2004@yahoo.com

(Received 15 August 2011, accepted 28 September 2011)

Abstract

This paper presents an optimization technique for solving linear system problems with more number of equations than unknown variables using Euclidean Space theory and least squares method. In view to automating the technique, we developed software in FORTRAN code for a generalized case. The technique was applied to determine the position of a Global Positioning System (GPS) receiver on the earth surface for arbitrarily positioned twelve GPS satellites. The technique is numerically friendly, and can be conveniently used to simulate problems involving linear systems.

Keywords: Euclidean norm, pseudo-inverse, Gaussian elimination, Global Positioning System (GPS).

Resumen

Este trabajo presenta una técnica de optimización para resolver problemas de sistemas lineales con mayor número de ecuaciones que variables desconocidas usando la teoría del Espacio Euclidiano y método de mínimos cuadrados. En vista de la automatización de la técnica, hemos desarrollado software en código FORTRAN para un caso generalizado. La técnica fue aplicada para determinar la posición de un Sistema de Posicionamiento Global (GPS) receptor en la superficie de la tierra arbitrariamente posicionando doce satélites GPS. La técnica es numéricamente fácil, y puede ser convenientemente utilizado para simular problemas que afectan a los sistemas lineales.

Palabras clave: Norma euclidiana, pseudo-inversa, eliminación de Gauss, Sistema de Posicionamiento Global (GPS).

2000 MSC: 65F20, 15A09, 15A90 **ISSN 1870-9095**

I. INTRODUCTION

Solving linear system problems with more number of equations than unknown variables could be challenging and laborious. One of such problems in practice is the simultaneous observation of more than four satellites by a GPS receiver [1, 2, 3]. In this paper, we set it as our objective to develop an optimization technique for tackling such problems.

The technique involves several mathematical apparatus that are interlinked. Euclidean Space theory and least squares method are used to transform the matrix defined by the coefficients of the unknown variables to obtain an invertible symmetric square matrix, **g**. The matrix is then augmented with the determined vector **p**, and solved by Gaussian elimination method to obtain the final solution for the unknown variables.

In view to automating the technique, we developed software in FORTRAN code for a generalized case (see appendix A). We apply the technique to determine the position of a GPS receiver on the earth surface for arbitrarily positioned twelve GPS satellites on a constellation [1, 4, 5].

The mathematical theories and derivations are presented in Section 2. Section 3 discusses the numerical algorithm; application of the technique is presented in Section 4. Section 5 gives insight discussion into the results obtained from the application, and conclusion is drawn in Section 6.

II. THEORY AND DERIVATIONS

An over-determined linear equation is defined in matrix notation as,

$$A\hat{\mathbf{x}} \approx \mathbf{b}\,, \qquad (1)$$

where A is m x n matrix (m>n), $\hat{\mathbf{x}}$ is an unknown n-dimensional parameter vector, and **b** is a known measured vector. Our interest is to find the solution of Eq. (1). *A* being a non-square matrix makes direct exact solution of Eq. (1) impossible. Under this condition, the number of equations (m) is more than the number of variables (n). To determine unique solutions for such equations, an optimization technique is used. To this end, we define a residual **r**, of Eq. (1) as; $\mathbf{r} = A\mathbf{x} - \mathbf{b}$.

The Euclidean norm squared of the residual is minimized to attain an optimization procedure. The Euclidean norm squared of the residual is defined as,

$$\|\mathbf{r}\|^2 = \sum_{i=1}^{m}\left([A\mathbf{x}]_i - \mathbf{b}_i\right)^2 \quad i = 1, 2, 3, \ldots.m, \tag{2}$$

where $[\mathrm{Ax}]_i$ represents the i-th component of vector $A\mathbf{x}$. We can simplify Eq. (2) further by adopting the principle of least squares. In n-dimensional Euclidean Space, the squared norm of a is $a^T a$, [6], where a^T is the transpose of a.

$$\|\mathbf{a}\|^2 = (\mathbf{a},\mathbf{a}) = a^T a. \tag{3}$$

Eq. (2) can therefore be re-written in form of Eq. (3) as,

$$\|r\|^2 = (A\mathbf{x}-\mathbf{b})^T(A\mathbf{x}-\mathbf{b}) = (A\mathbf{x})^T(A\mathbf{x}) - \mathbf{b}^T A\mathbf{x} - (A\mathbf{x})^T\mathbf{b} + \mathbf{b}^T\mathbf{b} \tag{4a}$$

but, $\mathbf{b}^T A\mathbf{x} = (A\mathbf{x})^T\mathbf{b}$, therefore,

$$(A\mathbf{x}-\mathbf{b})^T(A\mathbf{x}-\mathbf{b}) = (A\mathbf{x})^T(A\mathbf{x}) - 2(A\mathbf{x})^T\mathbf{b} + \mathbf{b}^T\mathbf{b}. \tag{4b}$$

The minimum value for the Euclidean norm squared of the residual can then be determined when its derivative with respect to **x** assumes value zero, that is,

$$\frac{d}{d\mathbf{x}}\left[(A\mathbf{x})^T(A\mathbf{x}) - 2(A\mathbf{x})^T\mathbf{b} + \mathbf{b}^T\mathbf{b}\right] = 2A^T A\hat{\mathbf{x}} - 2A^T\mathbf{b} = 0. \tag{5a}$$

From Eq. (5a), the minimizing vector $\hat{\mathbf{x}}$ is the solution of the normal equation. Thus,

$$A^T A\hat{\mathbf{x}} = A^T\mathbf{b}. \tag{5b}$$

Vector $\hat{\mathbf{x}}$ can be easily solved by multiplying $(A^TA)^{-1}$ by $A^T\boldsymbol{b}$. The product (A^TA) results to an invertible symmetric square matrix unlike matrix-A.

$$\hat{\mathbf{x}} = (A^T A)^{-1} A^T\mathbf{b}, \tag{6}$$

where $(A^TA)^{-1}A^T$ is the pseudo-inverse of matrix-A [6].

III. NUMERICAL ALGORITHM

Numerically, the procedure involved in solving Eq. (6) is cumbersome. For numerical applications, a more direct approach to solve this equation is the Gaussian elimination method. Before application, we define the product (A^TA), which is a square matrix as **g** and that of $A^T\mathbf{b}$, which is a vector as **p**. Thus,

$$\begin{bmatrix} g_{11} & g_{12} & g_{13} & \cdot & \cdot & \cdot & g_{1n} \\ g_{21} & g_{22} & g_{23} & \cdot & \cdot & \cdot & g_{2n} \\ g_{31} & g_{32} & g_{33} & \cdot & \cdot & \cdot & g_{3n} \\ \cdot & \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & \cdot & & & & \cdot \\ g_{n1} & g_{n2} & g_{n3} & \cdot & \cdot & \cdot & g_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \cdot \\ \cdot \\ \cdot \\ p_n \end{bmatrix}. \tag{7}$$

Matrix- **g** is then augmented with vector **p** to obtain a new matrix $\tilde{\mathbf{g}} = [\mathbf{g} \;\; \mathbf{p}$. We define the entries of **p** as g_{ij}; $j = 1, 2, \ldots\ldots n$.

Therefore, the augmented matrix is,

$$\begin{bmatrix} g_{11} & g_{12} & g_{13} & \cdot & \cdot & g_{1n} & g_{1,n+1} \\ g_{21} & g_{22} & g_{23} & \cdot & \cdot & g_{2n} & g_{2n,n+1} \\ g_{31} & g_{32} & g_{33} & \cdot & \cdot & g_{3n} & g_{3n,n+1} \\ \cdot & \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & & & \cdot \\ g_{n1} & g_{n2} & g_{n3} & \cdot & \cdot & g_{nn} & g_{nn,n+1} \end{bmatrix}. \tag{8}$$

The Gaussian elimination algorithm operates by reducing the order of a linear system from n to 1 which can easily be solved so that other variables are thereafter solved by back substitution. Defining a linear system of order k as,

$$g^{(k)}\mathbf{x} = \mathbf{p}^{(k)}. \tag{9}$$

The unknown variables $x_1, x_2, \ldots\ldots x_{k-1}$ can be eliminated at successive stages. We defined the row multiplier as $m_{ik} = g_{ik}^{(k)} / g_{kk}^{(k)}$; $i = k+1, \ldots n$. This is used to eliminate the unknown variable x_k from the linear equations, such that the new entries after the operations will be,

$$g_{ij}^{(k+1)} = g_{ij}^{(k+1)} - m_{ik} \cdot g_{kj}^{(k)};\; j = k+1, \ldots\ldots. n+1. \tag{10}$$

The earlier rows from 1 to k are left intact, and zeros are introduced into the column k below the diagonal element. By continuing this process, after n-1 steps, we obtain,

$$\begin{bmatrix} g_{11}^{(1)} & g_{12}^{(1)} & g_{13}^{(1)} & . & . & g_{1n}^{(1)} & g_{1,n+1}^{(1)} \\ 0 & g_{22}^{(2)} & g_{23}^{(2)} & . & . & g_{2n}^{(2)} & g_{2n,n+1}^{(2)} \\ 0 & 0 & g_{33}^{(3)} & . & . & g_{3n}^{(3)} & g_{3n,n+1}^{(3)} \\ . & . & . & & & & . \\ . & . & . & . & . & . & . \\ . & . & . & & & & . \\ 0 & 0 & 0 & . & . & g_{nn}^{(n)} & g_{nn,n+1}^{(n)} \end{bmatrix}. \quad (11)$$

Matrix (11) is a matrix of order n, such that, $g^{(n)}\mathbf{x} = \mathbf{p}^{(n)}$, and can be re-written as,

$$u\mathbf{x} = \mathbf{q}, \quad (12)$$

where u is the upper triangular matrix and vector $\mathbf{q}$ is vector $\mathbf{p}$ in order n, *i.e.* $\mathbf{q} = u = \left[u_{i,n+1}\right]$. Eq. (12) gives the final solution of Eq. (1) by back substitution process. Thus,

$$x_n = u_{n,n+1}/u_{nn}. \quad (13)$$

Also,

$$x_k = 1/u_{kk}\left[u_{k,n+1} - \sum_{j=k+1}^{n} u_{kj}x_j\right]; k = n-1, n-2, \ldots.1. \quad (14)$$

Eqs. (13) and (14) are the required solutions. We developed a FORTRAN code to automate the technique.

IV. APPLICATION

This technique can be found applicable in any linear system problem with more equations than unknown variables. We apply it to a system of twelve GPS satellites on a constellation. The intention is to determine the position of a receiver on the earth surface with these satellites in view [7]. The coordinates of the satellites are assumed to be; SV_1(3,1,2,1;9), SV_2(1,2,1.5,1;8), SV_3(2,1,2,1;8), SV_4(3,1,1,1;7), SV_5(1,2,2,1;9), SV_6(3,2,1,1;10), SV_7(2,2,3,1;12), SV_8(1,1,3,1;8), SV_9(1,3,3,1;13) SV_{10}(2,2,2,1;10), SV_{11}(3,1,2,1;10) and SV_{12}(2,2,1,1;8). The first three entries for each satellite are the satellite position term in Cartesian coordinate. The fourth entry is the receiver clock bias and the last entry represents difference between measured and determined pseudoranges. These sets of coordinates are used as the input data for the software code.

$$\begin{bmatrix} 3 & 1 & 2 & 1 \\ 1 & 2 & 1.5 & 1 \\ 2 & 1 & 2 & 1 \\ 3 & 1 & 1 & 1 \\ 1 & 2 & 2 & 1 \\ 3 & 2 & 1 & 1 \\ 2 & 2 & 3 & 1 \\ 1 & 1 & 3 & 1 \\ 1 & 3 & 3 & 1 \\ 2 & 2 & 2 & 1 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \approx \begin{bmatrix} 9 \\ 8 \\ 8 \\ 7 \\ 9 \\ 10 \\ 12 \\ 8 \\ 13 \\ 10 \\ 10 \\ 8 \end{bmatrix}. \quad (15)$$

The variables x : x_1, x_2, x_3 are the GPS user position coordinates and x : x_4 is the clock bias error [4]. Eq. (15) is obviously over-determined. The code transforms Eq. (15) to a square matrix,

$$\begin{bmatrix} 56 & 37 & 43.5 & 24 \\ 37 & 38 & 40 & 20 \\ 43.5 & 40 & 52.25 & 23.5 \\ 24 & 20 & 23.5 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 222 \\ 195 \\ 228 \\ 112 \end{bmatrix}. \quad (16)$$

The Gaussian elimination procedure of the code then modifies Eq. (16) to an upper triangular matrix,

$$\begin{bmatrix} 56 & 37 & 43.5 & 24 & 222 \\ 0 & 13.55 & 11.25 & 4.14 & 48.32 \\ 0 & 0 & 9.11 & 1.42 & 15.41 \\ 0 & 0 & 0 & 0.23 & -0.31 \end{bmatrix}. \quad (17)$$

Matrix (17) yields, $x_1, = 1.48215$, $x_2 = 2.39867$, $x_3 = 1.90319$, $x_4 = -1.35582$ (see appendix B). This output is discussed in Section 5.

V. DISCUSSION

This technique proffers direct and numerically stable procedure for solving linear systems with limited unknown variables. In this section, we first of all discussed the case of GPS satellites that are located on an orbital plane and the position of a user on any part of the earth surface is to be determined with these satellites in view [7]. In practice, four satellites are usually positioned on each of the six orbital planes to form a GPS constellation of twenty-four satellites [1, 4, 5]. In real sense, a receiver is designed to observe more than four satellites simultaneously, leading to superfluous linear systems. The technique presented herein tackles such problems.

All the satellites coordinate entries here are arbitrarily chosen. Under practical application, they must be initially determined before applying this technique. Each element is defined as,

$$a_{ij} = \frac{x_k^{(i)} - x_{k(0)}}{\rho_i}, \qquad (18)$$

where $x_k^{(i)}$ represents the k-th position variable for the i-th satellite and ρ_i is the pseudo-range for the i-th satellite. Each of the last column entries is taken as unity because they are receiver dependent and therefore the same for all satellites signals and pseudoranges [1, 2].

After evaluation of Eq. (17), the first three variables ($x : x_1, x_2, x_3$) are the GPS user position coordinates in metres and $x : x_4$ is the user bias clock error in dimension of distance. To transform this quantity to dimension of time (seconds), there is need to divide it with the speed of light. To appreciate the final result, the user position in Cartesian coordinate system must be converted to the flat earth coordinate system in terms of earth's latitude, longitude and altitude (see [4, 5] for details).

VI. CONCLUSION

An optimization technique for evaluating linear systems with more number of equations than unknown variables has been developed. The technique is numerically friendly and simple to apply. As part of our efforts to enhance its lucidity, we apply it to determine the position of a GPS user on the earth surface for arbitrarily positioned twelve GPS satellites. The technique was validated by self-consistency method.

However, we admit that ill-conditioning of the (A^TA) product of the pseudo-inverse may constitute threat to the accuracy of the technique. Also, the starting point involves minimizing the norm of measured quantity error, *A***x-b** as opposed to that of the small error in the unknown parameter, **x** which could have actually been needed for better accuracy. These challenges would be taken into consideration in future studies.

REFERENCES

[1] Kintner, P. M. and Ledvina, B. M., *The Ionosphere, Radio Navigation and Global Navigation Satellite Systems*, Adv. Space Res. **35**, 788–811 (2005).

[2] Ozludemir, M. T., *The Stochastic Modeling of GPS observations*, Turkish J. Eng. Env. Sc. **28**, 223-231 (2004).

[3] Basu, S. S., Basu, C. E., Vallandares, H. C., Yeh, S. Y., Su, E., MacKenzie, P. J., Sultan, J., Aarons, F. J., Rich, P., Doherty, K., Groves, M. and Bullet, T. W., *Ionospheric Effects of Major Magnetic Storms During the International Space Weather Period of September and October 1999: GPS Observations, VHF/UHF Scintillations, and in-situ Density Structures at Middle and Equatorial Latitudes*, J. Geophs. Res. **106**, 30389-30413 (2001).

[4] Hofmann-Wellenhof, B., Lichtenegger, H. and Collins, J., *Global Positioning System:Theory and Practice*, 4th Ed. (Spriger-Verlag Wien, New York, 1997).

[5] Tsui, J. B., *Fundamentals of Global Positioning System Receiver: A Software Approach*, (John Wiley & Sons Inc, New York, 2000).

[6] Kreyszig, E., *Advanced Engineering Mathematics*, 8th Ed. (John Wiley & Sons (Asia) Pte Ltd., Singapore, 2005).

[7] Davies, K. and Smith, E. K., *Ionospheric Effects on Land Mobile Satellite Systems*, IEEE Antennas and Propagation Magazine **44**, 24-31 (2002).

Appendix A: Code

```
      Program optz
C     Program to solve over-determined linear systems
     Parameter (m=m,n=n)
     Dimension a(m,n),P(m),c(n,n),D(n),x(n)
      Open(unit=1,file='optz.out',status='new')
      Write(*,*)'Enter the coefficient of variables'
      Read(*,*)((a(i,j),j=1,n),i=1,m)
      Write(*,*)'Enter the values of p'
      Read(*,*)(p(i),i=1,m)
C      Determining the product of transpose of matrix A and
A
          Write(1,*)
          do 39 j=1,n
          do 35 i=1,n
          C(i,j)=0.0
 35          continue
           do 40 i=1,n
          do 36 k=1,m
          C(i,j)=c(i,j)+(a(k,j)*a(k,i))
 36          continue
 40           continue
 39          continue
 43          Format(2x,5F7.2)
C
*************************************************
**********
C       Determining the product of A transpose and p
            Do 46 i=1,n
           D(i)=0.0
 46           continue
           Do 47 i=1,n
           Do 49 j=1,m
           D(i)=D(i)+a(j,i)*p(j)
 49           continue
 47           continue
C
*************************************************
********
C       Determining the augmented matrix
       Do 52 j=1,m
        c(i,m)=D(i)
 52      continue
       Write(1,*)
       Write(1,*)'The initial Augmented matrix g'
```

```
      Do 53 i=1,n
      Write(1,43)(c(i,j),j=1,m)
 53   Continue
C     Solving the augmented matrix with Gaussian
Elimination
      Do 54 j=1,n-1
      Do 54 i=j+1,n
      T=c(i,j)/c(j,j)
      Do 54 k=1,n+1
       c(i,k)=c(i,k)-c(j,k)*T
 54     continue
      Write(1,*)
      Write(1,*)'The final Augmented matrix g'
      Do 56 i=1,n
      Write(1,43)(c(i,k),k=1,n+1)
 56   Continue
 55    Format(2x,5F10.5)
       x(4)=c(4,5)/c(4,4)
C     **********Starting the back substitution
procedure*******
       Do 58 i=n-1,1,-1
       sum=0.0
       Do 57 j=i+1,n
       x(n)=c(n,n+1)/c(n,n)
       Sum=sum+c(i,j)*x(j)
        x(i)=(c(i,n+1)-sum)/c(i,i)
 57   Continue
 58    Continue
       Write(1,*)
       Write(1,59)(i,x(i),i=1,n-1)
 59    Format(2x,'x(',i2,')=',F8.5)
       Write(1,60)x(n)
 60    Format(2x,'x(n)=',F8.5)
        stop
        end
```

Appendix B: Sample Output

```
The initial Augmented matrix g
  56.00  37.00  43.50  24.00 222.00
  37.00  38.00  40.00  20.00 195.00
  43.50  40.00  52.25  23.50 228.00
  24.00  20.00  23.50  12.00 112.00

The final Augmented matrix g
  56.00  37.00  43.50  24.00 222.00
    .00  13.55  11.26   4.14  48.32
    .00    .00   9.11   1.42  15.41
    .00    .00    .00    .23   -.31

 x( 1)=  1.48215
 x( 2)=  2.39867
 x( 3)=  1.90319
 x( 4)= -1.35582
```