

Optimal Neural-Based Navigation Satellite Selection

Dan Simon and Hossny El-Sherief
 TRW Systems Integration Group
 PO Box 1310
 San Bernardino, CA 92402-1310

Abstract

The application of neural-based methods to optimal satellite subset selection for navigation use is discussed. The optimal satellite subset is chosen by minimizing a quantity known as Geometric Dilution of Precision (GDOP), which is given by the trace of the inverse of the measurement matrix. An artificial neural network learns the functional relationships between the entries of a measurement matrix and the eigenvalues of its inverse, and thus generates GDOP without inverting a matrix.

1 Introduction

A Global Positioning System (GPS) receiver generates a user position and time by measuring the range from the user to four or more GPS satellites [1]–[2], but a GPS receiver can process only a subset of available satellite signals. Before processing, the receiver must decide which subset to use. The optimal choice can be made by using the subset which results in the smallest magnification of satellite errors onto resultant user position and time errors.

This approach taken in this paper is applicable to any number of satellite signals. It is based on the learning properties of artificial neurons, and as such gives an approximate rather than an exact answer. Its primary advantage lies in the fact that no matrix inversions are required. This translates into less required computational time for satellite subset selection, and the ability to begin navigating sooner with the best satellite subset.

2 Geometric Dilution of Precision

A GPS receiver measures the ranges (R_1, R_2, \dots, R_n) between the user and n GPS satellites. The GPS satellite positions are (x_i, y_i, z_i) , $(i = 1, \dots, n)$. The four unknowns which the user needs to determine are the offset T between receiver

time and GPS time, and the user position (x, y, z) . Denote the user's best estimate of time offset and position as \hat{T} and $(\hat{x}, \hat{y}, \hat{z})$. Denote the corresponding best estimates of range as $(\hat{R}_1, \hat{R}_2, \dots, \hat{R}_n)$. The errors between the true and estimated quantities are denoted by $\Delta x = x - \hat{x}$, $\Delta y = y - \hat{y}$, $\Delta z = z - \hat{z}$, $\Delta T = T - \hat{T}$, and $\Delta R_i = R_i - \hat{R}_i$. The errors of the user's estimate of time and position can be determined by solving the following n simultaneous nonlinear equations for Δx , Δy , Δz , and ΔT [3].

$$\begin{aligned} (\hat{x} + \Delta x - x_i)^2 + (\hat{y} + \Delta y - y_i)^2 + (\hat{z} + \Delta z - z_i)^2 \\ = (\hat{R}_i + \Delta R_i - c\hat{T} - c\Delta T)^2, \quad (i = 1, \dots, n) \end{aligned} \quad (1)$$

where c is the speed of light. These equations can be linearized to obtain the matrix equation

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & 1 \\ a_{21} & a_{22} & a_{23} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & 1 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \\ c\Delta T \end{pmatrix} = \begin{pmatrix} \Delta R_1 - c\hat{T} \\ \Delta R_2 - c\hat{T} \\ \vdots \\ \Delta R_n - c\hat{T} \end{pmatrix} \quad (2)$$

The $n \times 4$ matrix in Eq. 2 is called the *measurement matrix*, and its elements are given by

$$\begin{aligned} a_{i1} &= (\hat{x} - x_i)/(\hat{R}_i - c\hat{T}) \\ a_{i2} &= (\hat{y} - y_i)/(\hat{R}_i - c\hat{T}) \\ a_{i3} &= (\hat{z} - z_i)/(\hat{R}_i - c\hat{T}) \end{aligned} \quad (3)$$

Equation 2 can be written more compactly as

$$A\bar{x} = \bar{r} \quad (4)$$

which has a least-squares solution given by $\bar{x} = (A^T A)^{-1} A^T \bar{r}$. The uncertainty of the solution of user position and time is therefore related to the uncertainty of the measured ranges as follows.

$$\text{cov}(\bar{x}) = (A^T A)^{-1} A^T \text{cov}(\bar{r}) [(A^T A)^{-1} A^T]^T. \quad (5)$$

If the covariance of \bar{r} is normalized to an identity matrix, we obtain a simplified expression for the covariance of user position and time.

$$\text{cov}(\bar{r}) = I \implies \text{cov}(\bar{x}) = (A^T A)^{-1}. \quad (6)$$

A useful scalar measure of the uncertainty of the solution of the user position and time is the trace of the above matrix. This quantifies the magnification of GPS range measurement errors (e.g., due to satellite and receiver inaccuracies) onto user position and time errors. Geometric Dilution of Precision (GDOP) is thus defined as

$$\text{GDOP} = [\text{trace}(A^T A)^{-1}]^{1/2}. \quad (7)$$

So GDOP can be calculated by inverting a 4×4 matrix. But how can GDOP be computed without resorting to matrix inversion? The way that humans learn most effectively is through induction, which is reasoning from the particular to the general. A computer algorithm can be designed to inductively generate a mathematical function by generalizing from known input/output relationships [5, p. 182]. Two ways of doing this are reviewed in the following sections.

3 Neural-Based Approximation

The term *backpropagation* refers to a general learning rule which is implemented in an artificial neural network. Good overviews can be found in [6, 7]. A typical backpropagation network has three layers of neurons – an input layer, a middle layer (also called a hidden layer), and an output layer. The outputs of the input layer neurons are weighted to activate the middle layer neurons, and the outputs of the middle layer neurons are weighted to activate the output layer neurons. The effectiveness of backpropagation in learning complex, multidimensional functions can be partially explained by Kolmogorov's Theorem [7, 8, 9]. This theorem states that any functional $\mathcal{R}^m \rightarrow \mathcal{R}^n$ mapping can be exactly represented by a three-layer neural network with $(2m+1)$ middle-layer neurons, assuming that the input components are normalized to lie in the range $[0, 1]$.

Knowing that GDOP is equal to the square root of the trace of $(A^T A)^{-1}$ (see Eq. 7), we will use the following general facts about the trace and eigenvalues of a matrix to compute GDOP.

(1) The trace of a matrix is equal to the sum of its eigenvalues [10, p. 40]. More specifically, the GDOP of a GPS navigation solution is equal to the square root of the sum of the eigenvalues of $(A^T A)^{-1}$, where A is defined in Eqs. 2–4.

(2) The determinant of a matrix is equal to the product of its eigenvalues [11, p. 332].

(3) If $(A^T A)$ has eigenvalues λ_i , then $(A^T A)^k$ has the eigenvalues λ_i^k , where k is any integer [4, p. 292], [10, p. 43].

Using $\vec{\lambda}$ to denote the four-element vector of the eigenvalues of $A^T A$, we define the following functions.

$$\begin{aligned} f_1(\vec{\lambda}) &= \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = \text{trace}(A^T A) \quad (8) \\ f_2(\vec{\lambda}) &= \lambda_1^2 + \lambda_2^2 + \lambda_3^2 + \lambda_4^2 = \text{trace}[(A^T A)^2] \\ f_3(\vec{\lambda}) &= \lambda_1^3 + \lambda_2^3 + \lambda_3^3 + \lambda_4^3 = \text{trace}[(A^T A)^3] \\ f_4(\vec{\lambda}) &= \lambda_1 \lambda_2 \lambda_3 \lambda_4 = \det(A^T A) \end{aligned}$$

Using the above notation, the GDOP which we wish to calculate is precisely given by

$$\text{GDOP} = (\lambda_1^{-1} + \lambda_2^{-1} + \lambda_3^{-1} + \lambda_4^{-1})^{1/2}. \quad (9)$$

So GDOP can be viewed as a scalar functional of the $\mathcal{R}^4 \rightarrow \mathcal{R}^4$ mapping $f(\vec{\lambda})$.

$$\text{GDOP} = \text{GDOP}[f(\vec{\lambda})] \quad (10)$$

The mapping from $f(\vec{\lambda})$ to GDOP cannot be determined analytically. But this complex, nonlinear mapping is the type of problem at which neural networks excel. A neural network can be designed with four easily computable inputs (f_1, f_2, f_3, f_4) , one hidden layer, and four outputs $(\lambda_1^{-1}, \lambda_2^{-1}, \lambda_3^{-1}, \lambda_4^{-1})$. The outputs can then be summed to give the square of GDOP.

4 Neural-Based Classification

Note that in picking a satellite subset to use for navigation, the receiver does not need to compute GDOP for every satellite subset. It only needs to find a subset which gives a satisfactorily low GDOP. So it can be argued that using backpropagation to find an approximating function to $\text{GDOP}[f(\vec{\lambda})]$ (see Eq. 10) entails more work than necessary. A more efficient approach is to create a network which classifies satellite subsets according to GDOP. If a network can be trained to classify a satellite group into one of n sets (S_1, \dots, S_n) according to GDOP, then those satellite groups which are classed in the best set are candidates for navigation use.

There are many different network architectures which have been proposed for classification. One recently proposed architecture is the Optimal Interpolative Net (OI Net). The OI Net is a three-layer classification network which grows during training according to how many neurons are necessary for correct classification. The efficient recursive learning formulation presented in [12] makes the OI Net an attractive architecture.

The OI Net can be used to select a good group of satellites with which to navigate by classifying groups

according to GDOP. Given a group G of satellites, the OI Net can be trained to classify the group as follows.

$$G \in S_i \quad \text{iff} \quad \text{GDOP}(G) < T_i \quad (i = 1, \dots, n) \quad (11)$$

where $\text{GDOP}(G)$ is the GDOP of satellite group G , and T_i ($i = 1, \dots, n$) is some set of thresholds.

Now the issue becomes one of determining suitable thresholds T_i to use in the classification. If too many thresholds are used, it becomes difficult to find a diverse enough set of training samples, and the probability of misclassification increases. But if too few thresholds are used, the classification becomes too coarse, too many satellite groups are correctly classified in the best set, and there is less chance of selecting the best satellite group.

One method of solving the dilemma of how many classes to use is to implement multiple OI Nets. Each OI Net uses all of the training samples, and is trained to classify a satellite group in one of two sets. Each OI Net uses a different threshold for classification. So we use n OI Nets, and the j -th OI Net classifies a satellite group according to whether its GDOP is less than or greater than the threshold T_j . We will use the convention $T_1 < T_2 < \dots < T_n$. A sequence of OI Nets running in series or parallel operates as follows.

(1) Set $j = 1$.

(2) If the j -th OI Net classifies any of its inputs as being less than T_j , then any of the corresponding satellite groups can be used for navigation. Exit the loop.

(3) If the test in step (2) fails, i.e. if the j -th OI Net classifies all of its inputs as being greater than T_j , then increment j by one and repeat step (2).

Note that if all n OI Nets classify the inputs as being greater than their respective thresholds, then all of the satellite groups have a GDOP greater than T_n . In this case, any satellite group can be used for navigation.

5 Simulation Results

The two neural networks described in this paper were simulated on a VAX 8650 computer. Training took place for a GPS receiver located 5000 feet above San Francisco in an 18-satellite constellation. Once each hour, for 12 hours, the measurement matrix (A) was generated for each visible four-satellite subset. The functions $f_i(\cdot)$ ($i = 1, 2, 3, 4$) (see Eqs. 8) were calculated, and λ_i^{-1} ($i = 1, 2, 3, 4$) were calculated. The function values f_i were normalized to the range $[0.2, 0.8]$, saved in a training file, and then used to train the neural networks. If $f_4(\cdot)$ (the determinant of $A^T A$) was less than 0.12, the satellite set was immediately discarded from consideration. Such a low determinant

can be shown to correspond to a GDOP too high for possible use. At each measurement time there were between five and seven visible satellites. There were thus between 15 and 35 four-satellite sets.

The networks were then tested on a simulated missile trajectory originating from Vandenberg Air Force Base in California. During the 120 seconds of powered flight, the missile travelled approximately 200 miles horizontally and 150 miles vertically. The trained neural networks were used to choose the best satellite group every two seconds. There were between five and seven satellites visible during the 120-second boost phase, and the satellite configuration with the best GDOP changed twice during that time.

5.1 Backpropagation Simulation

Backpropagation was used to learn the correct network weights and external inputs so that the four $f_i(\lambda)$ inputs were mapped into the correct eigenvalue inverses. The backpropagation network had four input neurons corresponding to f_i ($i = 1, 2, 3, 4$), nine hidden layer neurons, and four output neurons corresponding to λ_i^{-1} ($i = 1, 2, 3, 4$) (see Eqs. 8). The network was training with a learning rate of 0.9 and a momentum rate of 0.7.

It can be easily shown from Eqs. 2 - 4 that $\text{trace}(A^T A) = 2n$, where n is the number of satellites processed by the receiver. So the input $f_1(\cdot)$ is a constant. This constant was still used as an input to the neural network, however, for the sake of generality. In addition, this constant input increased the size of the network, thus affording more flexibility in learning.

The backpropagation net calculated a total of 1036 GDOPs during the simulated test flight with a maximum error of 4.00% and an rms error of 1.44%.

5.2 Optimal Interpolative Net Simulation

Several OI Nets were trained for several different thresholds of GDOP. Each OI Net had only three input neurons corresponding to (f_2, f_3 , and f_4) (recall that f_1 is constant). Each OI Net had two output neurons. Ideally, a satellite group with a GDOP less than the threshold would have an output vector of $[1, 0]$ and a group with a GDOP greater than the threshold would have an output vector of $[0, 1]$. The number of hidden layer neurons varied from one OI Net to the next as determined by the learning algorithm. Each OI Net used a fitting parameter of 0.1, an ill-conditioning threshold of $1e-8$, and an error reduction threshold of 0.001 [12]. Table 1 shows the characteristics and test results of several OI Nets which were trained and tested with the data described in this section.

GDOP Threshold	Number of Hidden Neurons	Percent Correct Classifications
2.4	12	97.0
2.6	14	93.1
2.8	17	87.4
3.0	16	94.0
3.2	15	100
3.4	12	92.3

Table 1: OI Net Characteristics (150 Training Inputs)

5.3 Comparison of Backpropagation and OI Net Learning

The most notable difference between backpropagation approximation and OI Net classification is the huge difference in training time. Backpropagation required about four hours and 47 minutes of VAX CPU time, while OI Net training required about one second of training time. So using an OI Net results in a savings of about 99.994% of the training time required for backpropagation. However, backpropagation gives a more accurate approximation than the OI Net. Backpropagation can be used for arbitrarily exact approximation, but the OI Net can only be used for either/or classification. The decision of which type of network to use must be made on the basis of the relative importance of training time versus exactness.

6 Conclusion

Two approximate methods of choosing an optimal subset of visible navigation satellites have been presented. The methods are based on the learning abilities of artificial neurons, and apply regardless of how many satellites the navigation system is tracking. A simulation study was shown to verify the applicability of the proposed approach. The results given in this paper clearly demonstrate the feasibility of neural-based satellite selection.

The strength of using a neural net to determine GDOP is its computational efficiency. If a conventional LU-decomposition method is used to invert a 4×4 matrix in order to determine GDOP, a total of 160 floating point operations are required for each matrix inversion. There may be well over 100 possible satellite configurations from which to choose. The computational effort required to determine the best satellite set could result in a short but noticeable delay before a navigation solution could be generated.

This delay could be important for aircraft, missiles, or other platforms for which immediate navigation data are desired. An electronic implementation of a neural network of the type described in this paper would be able to determine the best satellite configuration (in terms of GDOP) virtually instantaneously. This would enable the use of a high-integrity navigation solution without the delay required for many matrix inversions.

References

- [1] H. El-Sherief, "Aerospace Applications of Global Positioning System," Electrical Engineering Seminar, University of Nevada, Reno, 1992.
- [2] P. Janiczek and S. Gilbert, editors, *Global Positioning System, Vols. 1-3*, Washington, DC: Institute of Navigation, 1980, 1984, 1986.
- [3] P. Jorgensen, "Navstar/Global Positioning System 18-Satellite Constellations," in *Global Positioning System (Navigation Redbook)*, Washington, DC: Institute of Navigation, 2:1-12, 1984.
- [4] B. Noble and J. Daniel, *Applied Linear Algebra*, Third Edition, Englewood Cliffs, NJ: Prentice-Hall, 1988.
- [5] C. Townsend and D. Feucht, *Designing and Programming Personal Expert Systems*, Blue Ridge Summit, PA: Tab Books, 1986.
- [6] P. Simpson, *Artificial Neural Systems*, New York, NY: Pergamon Press, 1990.
- [7] M. Caudill, "Neural Networks Primer: Part III," *AI Expert*, pp. 53-59, June 1988.
- [8] R. Hecht-Nielsen, "Kolmogorov's Mapping Neural Network Existence Theorem," *Proceedings of the IEEE First International Conference on Neural Networks: Vol. III*, pp. 11-14, 1987.
- [9] K. Hornik et al., "Multilayer Feedforward Neural Networks are Universal Approximators," *Neural Networks*, 2:359-366, 1989.
- [10] R. Horn and C. Johnson, *Matrix Analysis*, New York, NY: Cambridge University Press, 1990.
- [11] G. Golub and C. Van Loan, *Matrix Computations*, Second Edition, Baltimore, MD: The Johns Hopkins University Press, 1990.
- [12] S. Sin and R. deFigueiredo, "An Evolution-Oriented Learning Algorithm for the Optimal Interpolative Net," *IEEE Transactions on Neural Networks*, 3:315-323, 1992.