Technical University of Denmark

Xiao Hu

30550: Satellite Based Positioning

Report for assignment G&H

January 4, 2019

DTU Space
National Space Institute

30550: Satellite Based Positioning, Report for assignment G&H

Author(s):
Xiao Hu

Supervisor(s):
Dr. Daniel Olesen

National Space Institute
Technical University of Denmark
Elektrovej Building 328, room 007
2800 Kongens Lyngby
Denmark
www.space.dtu.dk

E-mail: xiahaa@space.dtu.dk

Abstract

This report will present the theory and results accomplished for the assignment G&H which mainly includes the processing of real GPS-data and the simulation of the integration of the Galileo system and the GPS system.

Table of Contents

1 Assignment F: Extensions

This chapter begins with the initialization methods in 1.1.

1.1 Initialization

To recap, the position estimation problem is a non-linear optimization problem, which has to be solved iteratively. Generally speaking, it is normally very hard to converge to a global minimum. As a result, a good initial guess would be beneficial in several points:

- less iteration $\leftrightarrow$ fast speed of convergence.
- more likely to converge to a desired minimum.

This section will introduce several initialization methods which will serve as an initializer for the Gauss-Newton method.

1.1.1 The Bancroft Method

The Bancroft method allows obtaining a direct solution of the receiver position and the clock offset from at least four pseudoranges without any "a priori" knowledge for the receiver location. Revisit the model of pseudorange:

$$R = \rho + d\rho + c(dT - dt) + d_{ion} + d_{trop} + \epsilon \tag{1.1}$$

After eliminating all model terms including the ionospheric & tropospheric delays, the satellite clock error, the satellite orbit error, we will have:

$$R_c \approx \rho + c(dT) \tag{1.2}$$

Take the square on both sides,

$$(x_i^2 + y_i^2 + z_i^2 - R_c^2) - 2(x_i x + y_i y + z_i z - R_c(cdT)) + (x^2 + y^2 + z^2 - (cdT)^2) = 0 \tag{1.3}$$

Considering the inner product of Lorentz:

$$\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^T \mathbf{M} \mathbf{b} \tag{1.4}$$

$$= [a_1, a_2, a_3, a_4] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \tag{1.5}$$

the previous equation can be expressed in a more compact way as:

$$\frac{1}{2}\langle \begin{bmatrix} \mathbf{x_i} \\ R_c \end{bmatrix}, \begin{bmatrix} \mathbf{x_i} \\ R_c \end{bmatrix} \rangle - \langle \begin{bmatrix} \mathbf{x_i} \\ R_c \end{bmatrix}, \begin{bmatrix} \mathbf{x} \\ cdT \end{bmatrix} \rangle + \frac{1}{2}\langle \begin{bmatrix} \mathbf{x} \\ cdT \end{bmatrix}, \begin{bmatrix} \mathbf{x} \\ cdT \end{bmatrix} \rangle = 0 \tag{1.6}$$

This equation can be built for each visible satellite. By stacking them together, we will have

$$\mathbf{a} - \mathbf{BM}\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} + \Lambda\mathbf{1} = 0 \tag{1.7}$$

$$\Lambda = \frac{1}{2}\langle\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix}, \begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix}\rangle \tag{1.8}$$

$$\mathbf{a} = \begin{bmatrix}a_1\\ \vdots\\ a_N\end{bmatrix},\ a_i = \frac{1}{2}\langle\begin{bmatrix}\mathbf{x_i}\\ R_c\end{bmatrix}, \begin{bmatrix}\mathbf{x_i}\\ R_c\end{bmatrix}\rangle \tag{1.9}$$

Then,

$$\mathbf{BM}\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} = \mathbf{a} + \Lambda\mathbf{1} \tag{1.10}$$

$$\mathbf{B^T BM}\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} = \mathbf{B^T a} + \Lambda\mathbf{B^T 1} \tag{1.11}$$

$$\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} = \mathbf{M}(\mathbf{B^T B})^{-1}\mathbf{B^T}(\Lambda\mathbf{1} + \mathbf{a}) \tag{1.12}$$

From (1.8), and do the the inner product of Lorentz by using the following property that $\langle \mathbf{Mg}, \mathbf{Mh}\rangle = \langle \mathbf{g}, \mathbf{h}\rangle$, we will have a second order polynomial:

$$p_2\Lambda^2 + p_1\Lambda + p_0 = 0 \tag{1.13}$$

$$p_2 = \langle(\mathbf{B^T B})^{-1}\mathbf{B^T 1}, (\mathbf{B^T B})^{-1}\mathbf{B^T 1}\rangle \tag{1.14}$$

$$p_1 = 2\left(\langle(\mathbf{B^T B})^{-1}\mathbf{B^T 1}, (\mathbf{B^T B})^{-1}\mathbf{B^T a}\rangle - 1\right) \tag{1.15}$$

$$p_0 = \langle(\mathbf{B^T B})^{-1}\mathbf{B^T a}, (\mathbf{B^T B})^{-1}\mathbf{B^T a}\rangle \tag{1.16}$$

Solving this polynomial, we will have Λ and $\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix}$ can be computed. Since normally there are two roots for second order polynomial, there are two solutions for Λ. This ambiguity can be solved by checking the distance from the earth center to the receiver. Since the receiver cannot be further than the satellite, only one of the solutions would be reasonable.

1.1.2 DLT initialization

To use DLT for initial estimation, we have to assume $dT = 0$, then we will have the following relationship between the pseudorange and the geometric distance:

$$\rho \approx \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2},\ i = 1, 2, \cdots, n \tag{1.17}$$

$$\rho^2 \approx (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2,\ i = 1, 2, \cdots, n \tag{1.18}$$

$$d\rho_{ij}^2 \approx \rho_i^2 - \rho_j^2 \tag{1.19}$$

$$d\rho_{ij}^2 \approx (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 - (x - x_j)^2 - (y - y_j)^2 - (z - z_j)^2 \tag{1.20}$$

Do some simple extensions, we will have:

$$\begin{aligned} d\rho_{ij}^2 - (x_i^2 + y_i^2 + z_i^2) + (x_j^2 + y_j^2 + z_j^2) &= 2(x_j - x_i)x \\ &\quad + 2(y_j - y_i)y + 2(z_j - z_i)z \\ &= \begin{bmatrix} 2(x_j - x_i) & 2(y_j - y_i) & 2(z_j - z_i) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \end{aligned} \tag{1.21}$$

By stacking all measurements together, we will have a linear equation $\mathbf{A}[x,\ y,\ z]^T = \mathbf{b}$. By solving this linear system, we will have an estimation of $x,\ y,\ z$. Now we compute cdT:

$$\begin{bmatrix} \rho_1 - \sqrt{(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2} \\ \vdots \\ \rho_N - \sqrt{(x - x_N)^2 + (y - y_N)^2 + (z - z_N)^2} \end{bmatrix} = cdT \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{1.22}$$

Solve this to get an estimation of cdT. The final initial guess would be $\mathbf{x}_0 = [x,\ y,\ z,\ cdT]^T$.

1.1.3 SOCP initialization

The initialization can also be obtained with the Second-Order-Cone Programming (SOCP). Revisit the objective function:

$$\text{minimize}: \sum_{i=0}^{n} \left(\rho_i - (\sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} + cdT) \right)^2 \tag{1.23}$$

Now, use some slack variables $l_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}$ and the epigraph $t \geq \sum_{i=0}^{n} \rho_i - l_i - cdT)^2$, we will have:

$$\begin{aligned} &\text{minimize}: \ t \\ &\text{subject to}: t \geq \sum_{i=0}^{n} \rho_i - l_i - cdT)^2 \\ &\qquad l_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2},\ i = 1, 2, \cdots, n \end{aligned} \tag{1.24}$$

Relax $l_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}$ to $l_i \geq \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}$, we will have the standard SOCP problem:

$$\begin{aligned} &\text{minimize}: \ t \\ &\text{subject to}: (t, \frac{1}{1}, \rho_0 - l_0 - cdT, \cdots, \rho_n - l_n - cdT) \in \mathcal{Q}_r^{n+3} \\ &\qquad (l_i, (x - x_i), (y - y_i), (z - z_i)) \in \mathcal{Q}^4,\ i = 1, 2, \cdots, n \end{aligned} \tag{1.25}$$

By solving this convex SOCP problem, we will have a good estimation as $\mathbf{x}_0 = [x_0,\ y_0,\ z_0,\ cdT_0]^T$. The relaxation can be controlled by adding regularizations as:

$$\begin{aligned} &\text{minimize}: \ t + \lambda \sum_{i=0}^{n} l_i \\ &\text{subject to}: (t, \frac{1}{1}, \rho_0 - l_0 - cdT, \cdots, \rho_n - l_n - cdT) \in \mathcal{Q}_r^{n+3} \\ &\qquad (l_i, (x - x_i), (y - y_i), (z - z_i)) \in \mathcal{Q}^4,\ i = 1, 2, \cdots, n \end{aligned} \tag{1.26}$$

where λ can be seen as a trade-off weight.

1.1.4 Performance

In **ex7_position_estimation.m**, there is an option to choose the initialization method: 1-use user input, 2-DLT, 3-SOCP[1], 4-Bancroft method. Generally speaking, using initialization will make the convergence much faster (1 iteration for Bancroft methid, 3 iterations for DLT and SOCP methods, 7 usually for initialize with 0). Imaging the meansuret vector contains a large number of measurements, then it would be costy to compute the Jacobian matrix and further the Hessian matrix. In that sense, less iterations can save a lot of time.

1.2 FIM & Cramèr Rao bound

Generally speaking, the position estimation problem mentioned in GNSS field is nothing more than a parameter estimation problem. For parameter estimation, often the Cramèr Rao bound expresses a lower bound on the variance of unbiased estimators. Here we can define the $\theta = [x,\ y,\ z,\ cdT]$ and its unbiased estimator as $\hat{\theta}$. Than we need the Fisher Information Matrix to compute the the Cramèr Rao bound. In this case, revisit the definition of pseudorange after correction:

$$R = \rho + cdT + \epsilon \tag{1.27}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. It is easy to see that the random variable $R \sim \mathcal{N}(d, \sigma^2),\ where\ d = \rho + cdT$. Now suppose a series of random samples $R_1 \sim \mathcal{N}(d_1, \sigma_1^2),\ \cdots,\ R_n \sim \mathcal{N}(d_n, \sigma_n^2)$, the likelihood function is defined as:

$$f(\mathbf{x_i}|\theta) = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(R_i - d_i)^2}{2\sigma_i^2}} \tag{1.28}$$

The joint pdf of $R_1,\ \cdots,\ R_n$ is given as:

$$f_n(\mathbf{x_i}|\theta) = \prod_{i=1}^{n} f(\mathbf{x_i}|\theta) \tag{1.29}$$

then the log-likelihood function can be derived as:

$$l_n(\theta) = log f_n(\mathbf{x_i}|\theta) = \sum_{i=1}^{n} log f(\mathbf{x_i}|\theta) \tag{1.30}$$

$$log f(\mathbf{x_i}|\theta) = -\frac{1}{2} log(2\pi\sigma_i^2) - \frac{(R_i - d_i)^2}{2\sigma_i^2} \tag{1.31}$$

Taking the derivative,

$$\frac{\partial l_n(\theta)}{\partial \theta} = \sum_{i=1}^{n} \frac{\partial log f(\mathbf{x_i}|\theta)}{\partial \theta} \tag{1.32}$$

$$\frac{\partial log f(\mathbf{x_i}|\theta)}{\partial \theta} = -\frac{R_i - d_i}{\sigma_i^2} [\frac{\partial d_i}{\partial x},\ \frac{\partial d_i}{\partial y},\ \frac{\partial d_i}{\partial z},\ \frac{\partial d_i}{\partial cdT}]^T \tag{1.33}$$

$$d_i = \rho_i + cdT \tag{1.34}$$

$$[\frac{\partial d_i}{\partial x},\ \frac{\partial d_i}{\partial y},\ \frac{\partial d_i}{\partial z},\ \frac{\partial d_i}{\partial cdT}]^T = [\frac{x - x_i}{\rho_i}, \frac{y - y_i}{\rho_i}, \frac{z - z_i}{\rho_i}, 1]^T \tag{1.35}$$

[1] I use the MOSEK as the SOCP solver.

let $\psi_i = [\frac{x-x_i}{\rho_i}, \frac{y-y_i}{\rho_i}, \frac{z-z_i}{\rho_i}, 1]^T$, since $R_1, \cdots, R_n$ are independent, according to the definition of the Fisher Information Matrix:

$$\mathbf{I}(\theta) = E\left[\frac{\partial l_n(\theta)}{\partial \theta}\frac{\partial l_n(\theta)}{\partial \theta}^T\right]. \tag{1.36}$$

$$\mathbf{I}(\theta) = \sum_{i=1}^{n}\mathbf{I}_i, \text{since } E[(R_i - d_i)(R_j - d_j)] = 0, \text{ if } i \neq j. \tag{1.37}$$

$$\mathbf{I}_i = E\left[\frac{(R_i - d_i)^2}{\sigma_i^4}\psi_i\psi_i^T\right] \tag{1.38}$$

$$= (\sigma_i^2)^{-1}\psi_i\psi_i^T \tag{1.39}$$

$$\mathbf{I}(\theta) = \begin{bmatrix}\psi_1 & \cdots & \psi_n\end{bmatrix}\begin{bmatrix}(\sigma_1^2)^{-1} & & \\ & \ddots & \\ & & (\sigma_n^2)^{-1}\end{bmatrix}\begin{bmatrix}\psi_1^T \\ \vdots \\ \psi_n^T\end{bmatrix} \tag{1.40}$$

$$\begin{bmatrix}(\sigma_1^2)^{-1} & & \\ & \ddots & \\ & & (\sigma_n^2)^{-1}\end{bmatrix} = (\sigma_1^2)^{-1}\mathbf{P},\ (\sigma_1^2)^{-1}P_{ii} = (\sigma_i^2)^{-1} \tag{1.41}$$

$$\mathbf{A} = \begin{bmatrix}\psi_1^T \\ \vdots \\ \psi_n^T\end{bmatrix} \tag{1.42}$$

$$\mathbf{I}(\theta) = (\sigma_1^2)^{-1}\mathbf{A}^T\mathbf{P}\mathbf{A} \tag{1.43}$$

Then the Cramér–Rao bound is the inverse of the Fisher information matrix (since unbiased):

$$C_\theta \geq \mathbf{I}(\theta)^{-1} \tag{1.44}$$

$$\mathbf{I}(\theta)^{-1} = (\sigma_1^2)(\mathbf{A}^T\mathbf{P}\mathbf{A})^{-1} \tag{1.45}$$

Clearly, according to [2], the covariance obtained by the WLS estimator can reach this bound. Therefore, the WLS estimator is the optimal estimator. Now go back to the definition of the DOP matrix:

$$\mathbf{Q}_{DOP} = \frac{\mathbf{A^T P A}^{-1}}{\sigma_{prior}^2} \tag{1.46}$$

We can clearly see a strong relationship between the DOP matrix and the Cramér–Rao bound (or the Fisher Information Matrix). If we let $\sigma_1 = 1$, $\sigma_{prior} = 1$ and $\mathbf{P} = \mathbf{I}$, then $\mathbf{Q}_{DOP}$ reaches the Cramér–Rao bound. Since $\mathbf{A}$ only depneds on the relative geometry between visible satellites and the receiver, it can express the lowest bound of covirance under current geometric distribution of satellites and receiver.

1.3 GDOP

It can be seen from previous analysis that the geometric dilution of precision (GDOP) is an important parameter used for satellite selection and the evaluation of positioning accuracy.

It can be seen here that $(\mathbf{A^T P A})^{-1}$ needs to be computed for further calculate those DOP values. Generally speaking, matrix inversion is a computation expensive (the effect is not significant since it is a 4×4 matrix) and numerically unstable.

1.3.1 Fast GDOP computation

revisit the definition of GDOP:

$$GDOP = \sqrt{trace(\mathbf{A^T P A}^{-1})} = \sqrt{\frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \frac{1}{\lambda_3} + \frac{1}{\lambda_4}} \tag{1.47}$$

where $\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4$ are the eigenvalue of $\mathbf{A^T P A}$. Since $\mathbf{A^T P A}$ is symmetric and positive definite, all its eigenvalues are greater than 0.

Eigen Decomposition

The first alternative way of computing the GDOP value is by eigen decomposition for $\frac{\mathbf{A^T P A}}{\sigma_0^2}$ and then compute the dop values as follows:

$$\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4 = eig\left(\frac{\mathbf{A^T P A}}{\sigma_0^2}\right) \tag{1.48}$$

$$GDOP = \sqrt{\frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \frac{1}{\lambda_3} + \frac{1}{\lambda_4}} \tag{1.49}$$

Since the complexity of applying eigen decomposition is $\mathcal{O}(n^3)$ and there are only 4 divisions, 3 additions and 1 square-root left for computing the GDOP, the overall complexity may be less than the previous closed-form solution (depending on the constant associated with the $\mathcal{O}(n^3)$).

Closed-form solution

Now define $\mathbf{M} = \mathbf{A^T P A}$, in order to solve the GDOP in closed-form, the following set of features are used:

$$h_1(\lambda) = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = trace(\mathbf{M}) \tag{1.50}$$

$$h_2(\lambda) = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 + \lambda_4^2 = trace(\mathbf{M^2}) \tag{1.51}$$

$$h_3(\lambda) = \lambda_1^3 + \lambda_2^3 + \lambda_3^3 + \lambda_4^3 = trace(\mathbf{M^3}) \tag{1.52}$$

$$h_4(\lambda) = \lambda_1 * \lambda_2 * \lambda_3 * \lambda_4 = det(\mathbf{M}) \tag{1.53}$$

These equalities hold because $\mathbf{M}$ and its powers are symmetric matrices (hoffman and kunze 1961). These features are firstly used by researchers for approximating the GDOP using learning based approaches, e.g. the Artificial Neural Network (ANN) and Support-Vector Regression (SVR). The recent work proposed a closed-form solution with the aforementioned features by using the property of symmetric polynomials.

A symmetric polynomial is a polynomial $p(x_1,\ x_2,\ \cdots,\ x_n)$ in n variables such that when any two variables are interchanged, the polynomial remains the same. This can be more precisely defined as follows:

$$p(x_{\sigma_1},\ x_{\sigma_2},\ \cdots,\ x_{\sigma_n}) = p(x_1,\ x_2,\ \cdots,\ x_n) \tag{1.54}$$

where $x_{\sigma_1},\ x_{\sigma_2},\ \cdots,\ x_{\sigma_n}$ is any permuted sequence of $x_1,\ x_2,\ \cdots,\ x_n$. Two typical symmetric polynomials are:

- Power sum symmetric polynomials:

$$p_k(x_1,\ x_2,\ \cdots,\ x_n) = x_1^2 + x_2^2 + \cdots + x_n^2 \tag{1.55}$$

- Elementary symmetric polynomials:

$$e_k(x_1,\ x_2,\ \cdots,\ x_n) = \sum_{1 \le j_1 < j_2 < \cdots < j_k \le n} x_{j_1} x_{j_2} \cdots x_{j_k} \tag{1.56}$$

  The 0th degree elementary symmetric polynomial is defined by $e_0(x_1,\ x_2,\ \cdots,\ x_n) = 1$.

The close relationship between these two types of symmetric polynomials is further explained by the Newton–Girard formulae.

$$\begin{aligned} &k(-1)^k e_k(x_1,\ x_2,\ \cdots,\ x_n) + \\ &\quad \sum_{i=1}^{k} (-1)^{i+k} e_i(x_1,\ x_2,\ \cdots,\ x_n) p_{k-i}(x_1,\ x_2,\ \cdots,\ x_n) = 0 \end{aligned} \tag{1.57}$$

Now, revisit (1.47):

$$GDOP = \sqrt{\frac{\lambda_1\lambda_2\lambda_3 + \lambda_1\lambda_2\lambda_4 + \lambda_1\lambda_3\lambda_4 + \lambda_2\lambda_3\lambda_4}{\lambda_1\lambda_2\lambda_3\lambda_4}} \tag{1.58}$$

The numerator $\lambda_1\lambda_2\lambda_3 + \lambda_1\lambda_2\lambda_4 + \lambda_1\lambda_3\lambda_4 + \lambda_2\lambda_3\lambda_4$ can be written as $e_4(\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4)$. Consequently, by applying the Newton–Girard formulae, the numerator can be computed with

$$e_4(\lambda) = \frac{1}{3}\left[\frac{1}{2}\left(p_1(\lambda)^2 - p_2(\lambda)\right) p_1(\lambda) - p_1(\lambda)p_2(\lambda) + p_3(\lambda)\right] \tag{1.59}$$

$$p_1(\lambda) = h_1(\lambda) \tag{1.60}$$

$$p_2(\lambda) = h_2(\lambda) \tag{1.61}$$

$$p_3(\lambda) = h_3(\lambda) \tag{1.62}$$

$$\Leftrightarrow$$

$$e_4(\lambda) = \frac{0.5h_1(\lambda)^3 - 1.5h_1(\lambda)h_2(\lambda) + h_3(\lambda)}{3} \tag{1.63}$$

Overall, the GDOP can be written as

$$GDOP = \sqrt{\frac{0.5h_1(\lambda)^3 - 1.5h_1(\lambda)h_2(\lambda) + h_3(\lambda)}{3h_4}} \tag{1.64}$$

This proposed closed-form solution needs 144 floating operations for computing the GDOP, not including the operations for computing $\mathbf{M}$, dividing and the square-root.

Another closed form solution

Suppose the eigenvalues $\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4$ are known, the characteristic polynomial as

$$\begin{aligned}|\mathbf{A}-\lambda\mathbf{I}| =&(\lambda-\lambda_1)(\lambda-\lambda_2)(\lambda-\lambda_3)(\lambda-\lambda_4)\\ =&\lambda^4-(\lambda_1+\lambda_2+\lambda_3+\lambda_4)\,\lambda^3\\ &+(\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_1\lambda_4+\lambda_2\lambda_3+\lambda_2\lambda_4+\lambda_3\lambda_4)\lambda^2\\ &-(\lambda_1\lambda_2\lambda_3+\lambda_1\lambda_2\lambda_4+\lambda_1\lambda_3\lambda_4+\lambda_2\lambda_3\lambda_4)\lambda\\ &+\lambda_1\lambda_2\lambda_3\lambda_4\end{aligned} \tag{1.65}$$

It is well known that $|\mathbf{A}-\lambda\mathbf{I}|$ is a 4^{th} order polynomial which can be represented as

$$|\mathbf{A}-\lambda\mathbf{I}| = p_4\lambda^4+p_3\lambda^3+p_2\lambda^2+p_1\lambda+p_0 \tag{1.66}$$

It is easy to see that

$$GDOP=\sqrt{\frac{-p_1}{p_0}} \tag{1.67}$$

Through some symbolic computations and fully use the symmetric property, it can be concluded that it only needs around 110 floating operations to compute $p_0,\ p_1$. Theoretically speaking, this method will be the faster than the two aforementioned approaches, which can be shown in the experiment.

Experiment

An experiment has been done to compare the aforementioned three methods with the most traditional method $GDOP=\sqrt{trace(\mathbf{M}^{-1})}$. In this experiment, 100000 geometric matrices $\mathbf{A}$ have been generated randomly, the mean time for each GDOP computation and the total time for 100000 runs are shown in Table 1.1.

Table 1.1. Speed Comparison of four GDOP computation methods.

Method	**Mean Time: (s)**	**Total Time: (s)**
Baseline	1.1903e-05	1.1903
Closed Form	8.6366e-06	0.8637
Eigen Decomposition	6.3824e-06	0.6382
Proposed	3.3042e-06	0.3304

1.4 Augmented State

Revisit the composition of the pseudorange (omitting the satellite orbit error):

$$\rho = R+\underbrace{cdT}_{receiver}-\underbrace{cdt}_{satellite}+\underbrace{d_{iono}}_{satellite,receiver}+\underbrace{d_{trop}}_{satellite,receiver} \tag{1.68}$$

$$R=\sqrt{(x-x_{sv})^2+(y-y_{sv})^2+(z-z_{sv})^2} \tag{1.69}$$

Among those errors,

- cdT is receiver-dependent, identical for pseudorange measurements;
- cdt is satellite-dependent, unique for each pseudorange measurement;
- d_{iono} and d_{trop},are satellite and receiver dependent implicitly;

A further analysis can be concluded as follows:

- Since cdt is hard to model (in fact we need to add one unknown parameter for each pseudorange which means the number of unknown parameters are more than the number of measurements), it should be definitely eliminated before carrying out further computation.
- cdT can be modelled as one parameter to resolve. This is what we do normally.
- According to the common used model for modelling the ionospheric and tropospheric errors, d_{iono} and d_{trop} are related to their corresponding errors in polar direction (which is receiver and satellite independent) and the corresponding elevation angles. Consequently, we can add two more parameters x_5, x_6 and model the ionospheric error and tropospheric error as:

$$d_{iono} = x_5 * OF,\ OF = \left(1 - \left(\frac{RE * sin(zenith)}{RE + hI}\right)^2\right)^{-1/2} \tag{1.70}$$

$$d_{trop} = \frac{x_6}{sin(elv)} \tag{1.71}$$

With known satellite positions, Eq 1.71 can be written implicitly as the function of $\mathbf{x} = x,\ y,\ z,\ cdT$ as

$$d_{iono} = x_5 * f_1(\mathbf{x}, x_{sv}) \tag{1.72}$$

$$d_{trop} = \frac{x_6}{f_2(\mathbf{x}, x_{sv})} \tag{1.73}$$

$$\Leftrightarrow$$

$$d_{iono} = f_1^*(\mathbf{x}_{aug}) \tag{1.74}$$

$$d_{trop} = f_2^*(\mathbf{x}_{aug}) \tag{1.75}$$

$$\mathbf{x}_{aug} = [x,\ y,\ z,\ cdT,\ x_5,\ x_6]^T$$

In this way, we estimate $x_5,\ x_6$ together with $x,\ y,\ z,\ cdT$ through least-square optimization. The benefit of doing this is that we don't need to estimate the ionospheric delay and the tropospheric delay which depend on various factors and therefore hard to model. The cost we need to pay for this state augmentation is that the minimum number of visible satellites increases to 6 since there are 6 unknown parameters. Since now there the GPS, GLONASS, Galileo, Beidou system operated, observing at least 6 satellites are not so hard in open sky environment.

navSolverAug.m: do position estimation using the augmented state vector.

1.4.1 Result

- Estimation 0: Raw pseudoranges with no corrections;
- Estimation 1: Pseudoranges are corrected with satellite clock error, normal state vector $\mathbf{x}_{aug} = [x,\ y,\ z,\ cdT]^T$;

	x: (m)	**y**: (m)	**z**: (m)	**dT**: (s)
truth	3509042.2969	779567.15431	5251066.1743	0.0001
Estimation 0	3380435.0196	721266.27907	5082767.4168	-0.00028038122
Estimation 1	3509053.135	779569.77236	5251078.7139	0.00010006084
Estimation 2	3509042.2969	779567.15431	5251066.1743	0.0001
Estimation 3	3509042.2969	779567.15431	5251066.1743	0.0001

- Estimation 2: Pseudoranges are corrected with satellite clock error, augmented state vector $\mathbf{x}_{aug} = [x,\ y,\ z,\ cdT,\ x_5,\ x_6]^T$;
- Estimation 3: Pseudoranges are corrected with satellite clock error, the ionosperic delay and the tropospheric delay, normal state vector $\mathbf{x}_{aug} = [x,\ y,\ z,\ cdT]^T$;

	d_{iono}: (m)	d_{trop}: (m)
Truth	2.0097	3.0729
Estimation	2.0603	3.1652
Truth	4.0999	9.6935
Estimation	4.1897	10.3335
Truth	2.1160	3.2684
Estimation	2.0968	3.2327
Truth	2.8392	4.8038
Estimation	2.7667	4.6288
Truth	1.7710	2.6541
Estimation	1.7969	2.6982
Truth	2.5014	4.0340
Estimation	2.4129	3.8492
Truth	1.6440	2.4406
Estimation	1.6528	2.4553
Truth	2.4808	3.9904
Estimation	2.5186	4.0706
Truth	4.6837	16.6667
Estimation	4.6975	16.9891
Truth	3.6217	7.2313
Estimation	3.4984	6.7586
Truth	3.0255	5.2824
Estimation	3.1731	5.6970
Truth	2.2181	3.4619
Estimation	2.1557	3.3429

Table 1.2. Comparison of true and estimated ionospheric and tropospheric delay.

As can be seen, the estimations of the ionospheric delay and the tropospheric delay are fairly good with the mean errors being $-6.982e-04$ m and -0.0517 m and the standard deviation being 0.0782 m and 0.2975 m.

1.4.2 Geometric Meaning

Revisit the matrix $\mathbf{M}$ and assume the $\mathbf{P} = \mathbf{I}$:

$$\mathbf{M} = \mathbf{A^T A} \tag{1.76}$$

$$\mathbf{A} = \begin{bmatrix} \psi_1^T \\ \vdots \\ \psi_n^T \end{bmatrix} \tag{1.77}$$

$$\mathbf{M} = \sum_{i=1}^{n} \psi_i \psi_1^T \tag{1.78}$$

$$\psi_i \psi_i^T = \begin{bmatrix} \frac{(x-x_i)^2}{\rho_i^2} & (.) & (.) & (.) \\ (.) & \frac{(y-y_i)^2}{\rho_i^2} & (.) & (.) \\ (.) & (.) & \frac{(z-z_i)^2}{\rho_i^2} & (.) \\ (.) & (.) & (.) & 1 \end{bmatrix} \tag{1.79}$$

It can be seen that $trace(\mathbf{M}) = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 2n$. Since $\mathbf{M}$ is positive definite, so $\lambda_i > 0,\ i = 1, 2, 3, 4$. Consider the following optimization problem:

$$\begin{aligned} \text{minimize}: & \frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \frac{1}{\lambda_3} + \frac{1}{\lambda_4} \\ \text{subject to}: & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 2n \\ & \lambda_1 > 0, \lambda_2 > 0, \lambda_3 > 0, \lambda_4 > 0 \end{aligned} \tag{1.80}$$

The minimum value will be achieved when $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{2n}{4}$. Since $\mathbf{M} = \mathbf{A^T A}$, then we have $\mathbf{M} = \mathbf{V S^T U^T U S V^T} = \mathbf{V diag}(\mathbf{s_1^2}, \cdots, \mathbf{s_4^2}) \mathbf{V^T}$. So we know from $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4$ that $|s_1| = \cdots = |s_4|$, which means sigular vectors are equally distributed. So generally speaking, the more equally distributed of visible satellites, the smaller GDOP we obtain.

1. Hessian, prove strong convexity;2. use kkt condition to prove the optimum value is achieved when $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{2n}{4}$;3. since M is symmetric and positive definite, its eigen value is the square of its singular value. And from the geometrical meaning of singular value, we know that the minimum value can be achieved when the geometrical distribution is a sphere; 4. from strong convexity, we know that the more equally distributed, the smaller dop value it will have.

References

[1] Pratap Misra and Per Enge. Global positioning system: signals, measurements and performance second edition.

[2] Allan Aasbjerg Nielsen. *Least Squares Adjustment: Linear and Nonlinear Weighted Regression Analysis*. 2013.

National Space Institute
Technical University of Denmark
Elektrovej Building 328, room 007
2800 Kongens Lyngby
Denmark www.space.dtu.dk

E-mail: xiahaa@space.dtu.dk