Technical University of Denmark

Xiao Hu

30550: Satellite Based Positioning

Report for assignment F

November 20, 2018

DTU Space
National Space Institute

30550: Satellite Based Positioning, Report for assignment F

Author(s):
Xiao Hu

Supervisor(s):
Dr. Allan Aasbjerg Nielsen

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 F which mainly includes the computation of receiver's position from a set of pseudoranges using the Gauss-Newton method, the estimation of the corresponding dilution of precision (DOP) values.

Besides those normal requests, some relevant contents are also included as a personal interests, which includes several initialization methods (the Bancroft method, the direct linear transformation method, the second order cone programming method), the Fisher Information Matrix (FIM), the CramèrRao bound and their connections with the DOP matrix, several GDOP fast computation methods, the relationship between the geometric distribution of satellites and the optimal GDOP, fast optimal subset (subset of satellites with the minimum GDOP) selection method.

Please notice that the second part is far beyond the course requirement, consequently, if the reader is not interested, it can be skipped.

Table of Contents

1 Assignment F: Least Squares Adjustment and DOP

The main objective of the assignment is to estimate the receiver's position by applying the least squares adjustment and further estimate the DOP values.

1.1 Theory

1.1.1 Position Estimation

According to [1] and the lecture slides, the pseudorange R can be modeled as:

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

where R is the pseudorange, ρ is the geometric distance between the receiver and satellite, $d\rho$ is the orbit error (set to 0 in this experiment), c is the speed of light, dT is the receiver clock error, dt is the satellite clock error, d_{ion} and d_{trop} are the ionospheric delay and tropospheric delay, respectively, e is the error from the multipath effect and receiver noise. Now supposing all biased errors (e.g. the ionospheric effect, the tropospheric effect, the satellite clock error, the multipath error) have been compensated and only the gaussian random noise remained, the following pseudorange equation between the ith satellite and the receiver is given by

$$R_i = ||\mathbf{x} - \mathbf{x_i}|| + cdT + \epsilon_i \tag{1.2}$$

where $\mathbf{x} = [x,\ y,\ z]^T$, $\mathbf{x}_i = [x_{si},\ y_{si},\ z_{si}]^T$ are the $3D$ position vectors of the receiver and the i^{th} satellite, respectively, cdT is the receiver clock bias in unit of meter. The position estimation problem is nothing more than a unconstrained nonlinear least square problem defined as:

$$\underset{\mathbf{x}}{min} \sum_{i=1}^{k} [R_i - (||\mathbf{x} - \mathbf{x_i}|| + cdT)]^2 \tag{1.3}$$

In order to determine the position of a receiver, pseudoranges from $n \geq 4$ satellites must be used at the same time. For positioning calculation, the first-order Taylor expansion is applied to (1.2) around the approximate position of the receiver $\hat{\mathbf{x}} = [\hat{x}, \hat{y}, \hat{z}]^T$:

$$R_i = ||\hat{\mathbf{x}} - \mathbf{x_i}|| + cdT + \mathbf{a}_i \Delta\mathbf{x} + c\Delta dT + \epsilon_i \tag{1.4}$$

$$\Leftrightarrow$$

$$\Delta R_i = R_i - (||\hat{\mathbf{x}} - x_i|| + cdT) = \mathbf{a}_i \Delta\mathbf{x} + c\Delta dT + \epsilon_i \tag{1.5}$$

$$\mathbf{a}_i = \left[\frac{\hat{x} - x_i}{||\hat{\mathbf{x}} - \mathbf{x_i}||},\ \frac{\hat{y} - y_i}{||\hat{\mathbf{x}} - \mathbf{x_i}||},\ \frac{\hat{z} - z_i}{||\hat{\mathbf{x}} - \mathbf{x_i}||} \right] \tag{1.6}$$

The linear measurement equation can be obtained by stacking (1.6) together:

$$\underbrace{\begin{bmatrix}\Delta R_1\\ \Delta R_2\\ \vdots\\ \Delta R_k\end{bmatrix}}_{\mathbf{\Delta R}} = \underbrace{\begin{bmatrix}\mathbf{a}_1 & 1\\ \mathbf{a}_2 & 1\\ \vdots & \vdots\\ \mathbf{a}_k & 1\end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix}\Delta \mathbf{x} & c\Delta dT\end{bmatrix}}_{\mathbf{\Delta x}} + \underbrace{\begin{bmatrix}\epsilon_1\\ \epsilon_2\\ \vdots\\ \epsilon_k\end{bmatrix}}_{\epsilon} \tag{1.7}$$

where $\mathbf{A}$ is the design matrix which captures the receiver–satellite geometry. Usually (1.7) is solved by the Gauss-Newton method, which means the two steps are iterated until a given condition is met:

$$\text{solve: } \mathbf{A\Delta\hat{x}} = \mathbf{\Delta R} \tag{1.8}$$

$$\hat{\mathbf{x}} = \hat{\mathbf{x}} + \mathbf{\Delta\hat{x}} \tag{1.9}$$

If $\mathbf{A}$ is well behaved (which can be verified by its condition number), Eq (1.8) can be solved as

- $\mathbf{\Delta\hat{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{\Delta R}$ if measurements are weighted equally;
- $\mathbf{\Delta\hat{x}} = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{W}\mathbf{\Delta R}$ where $\mathbf{W}$ is the weighted matrix;
- Define $\bar{\mathbf{x}} = [\mathbf{x}^T\ 1]^T$ and rewrite Eq 1.8 as $\bar{\mathbf{A}}\bar{\mathbf{x}} = \mathbf{\Delta}\bar{\mathbf{R}}$. Then do the singular value decomposition for $\mathbf{USV}^T = \bar{\mathbf{A}}$, find the column of $\mathbf{V}$ corresponding to the smallest singular value $\mathbf{v}$, finally normalize the last element of $\mathbf{v}$ to 1 to recover $\mathbf{\Delta\hat{x}}$ (`v=H(:, end);x=v(1:end-1)./v(end)` in MATLAB). **Experiments showed this solution is faster than `A\b`.**

Normally, Eq (1.3) needs a good initial value. Otherwise, it either needs longer time to converge or might be traped in some undesired local minimums. Some initialization methods will be present in the next chapter if the reader is interested.

1.1.2 DOP

Since $\epsilon_1,\ \epsilon_2,\ \cdots,\ \epsilon_k$ are independent identical random variables and $\epsilon_i \sim \mathcal{N}(0, \sigma_0^2)$, it can be derived that the covariance matrix $\mathbf{Q}_{\hat{\mathbf{x}}}$ equals to $\sigma_0^2(\mathbf{A^T W A})^{-1}$ for weighted case. The DOP matrices is further defined as

$$\mathbf{Q}_{DOP} = \frac{\mathbf{Q}_{\hat{\mathbf{x}}}}{\sigma_0^2\sigma_p^2} = \frac{(\mathbf{A^T W A})^{-1}}{\sigma_p^2} \tag{1.10}$$

$$\mathbf{Q}_{DOP_{ENU}} = \mathbf{R}_{ECEF}^{ENU}\mathbf{Q}_{DOP}{\mathbf{R}_{ECEF}^{ENU}}^T \tag{1.11}$$

where σ_p^2 are the prior variances of measurements. Several relevant DOP values are defined as:

- $PDOP = \sqrt{trace(\mathbf{Q_{DOP}})}$ defines the geometric dilution of precision;
- $PDOP = \sqrt{trace(\mathbf{Q_{DOP}}(1:3, 1:3))}$ defines the position ($3D$) dilution of precision;
- $TDOP = \sqrt{\mathbf{Q_{DOP}}(4,4)}$ defines the time dilution of precision;
- $HDOP = \sqrt{trace(\mathbf{Q_{DOP_{ENU}}}(1:2,\ 1:2))}$ defines the horizontal dilution of precision;
- $VDOP = \sqrt{trace(\mathbf{Q_{DOP_{ENU}}}(3,\ 3))}$ defines the vertical dilution of precision;

1.1.3 Residual Analysis & Confidence Ellipsoid

According to [**?**], the covariance matrices for $\bar{\mathbf{x}}$, $\hat{\mathbf{R}}$ and $\hat{\epsilon}$ are given as:

$$\mathbf{Q_R} = \sigma_0^2 \mathbf{W}^{-1} \tag{1.12}$$

$$\mathbf{Q}_{\hat{\mathbf{x}}} = \sigma_0^2 (\mathbf{A^T W A})^{-1} \tag{1.13}$$

$$\mathbf{Q}_{\hat{\mathbf{R}}} = \sigma_0^2 \mathbf{A} (\mathbf{A^T W A})^{-1} \mathbf{A^T} \tag{1.14}$$

$$\mathbf{Q}_{\hat{\epsilon}} = \sigma_0^2 \left(\mathbf{W}^{-1} - \mathbf{A} (\mathbf{A^T W A})^{-1} \mathbf{A}^T \right) = \mathbf{Q_R} - \mathbf{Q}_{\hat{\mathbf{R}}} \tag{1.15}$$

σ_0 can be estimated with the help of the SSE (Sum of Squared Error), MSE (Mean Squared Erro) and RMSE (Root Mean Squared Error):

$$\hat{\epsilon} = \mathbf{\Delta R} - \mathbf{A \Delta \hat{x}} \tag{1.16}$$

$$SSE = \hat{\epsilon}^T \mathbf{W} \hat{\epsilon} \tag{1.17}$$

$$MSE = \frac{SSE}{n - p} \tag{1.18}$$

$$\hat{\sigma}_0 = RMSE = \sqrt{MSE} \tag{1.19}$$

where n denotes the number of measurements and p is the number of parameters to estimate.
The confidence ellipsoid can be used to define an envelop within which the confidence the above a certain level. The confidence ellipsoid is defined as:

$$(\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{Q}_{\hat{\mathbf{x}}}^{-1} (\mathbf{x} - \hat{\mathbf{x}}) = q \tag{1.20}$$

$$\mathbf{y}^T (\mathbf{V \Lambda V}^T)^{-1} \mathbf{y} = q, \text{ by Eigen decomposition;} \tag{1.21}$$

$$\mathbf{y}^T (\mathbf{V \Lambda}^{-1/2} \Lambda^{-1/2} \mathbf{V}^T) \mathbf{y} = q, \text{ by Unitary matrix and matrix square root} \tag{1.22}$$

$$\mathbf{z}^T \mathbf{z} = q, \ \mathbf{z} = \mathbf{\Lambda}^{-1/2} \mathbf{V}^T y \tag{1.23}$$

$$\frac{z_1^2}{\lambda_1 q} + \frac{z_2^2}{\lambda_2 q} + \cdots + \frac{z_p^2}{\lambda_p q} = 1 \tag{1.24}$$

where q is the fractile of a certain confidence. If σ_0 is known beforehand, $(\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{Q}_{\hat{\mathbf{x}}}^{-1} (\mathbf{x} - \hat{\mathbf{x}})$ follow a χ^2 distribution with p as the degree of freedom. If σ_0 is unknown, $\hat{\sigma}_0$ can be estimated with (1.19), which means $(n-p)\sigma_0^2 \sim \chi^2(n-p)$. Since $(\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{A}^T \mathbf{W A} (\mathbf{x} - \hat{\mathbf{x}})$ follow a χ^2 distribution with p as the degree of freedom, then

$$\frac{(\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{A}^T \mathbf{W A} (\mathbf{x} - \hat{\mathbf{x}}) / p}{\sigma_0^2} \sim \mathcal{F}(p, \ n - p) \tag{1.25}$$

So q can selected as the 95% fractile of the $\chi^2(p)$ distribution (if σ_0 known) or $\mathcal{F}(p, \ n-p)$ distribution (otherwise).

1.2 Tasks

- Implement a MATLAB routine for least squares estimation of a GPS position with the Gauss-Newton method as the solver.
- Compare position estimation results obtained from pseudoranges before and after atmospheric and satellite clock errors correction.
- Implement a MATLAB routine for calculating the DOP-values: HDOP, VDOP, PDOP and GDOP.

1.3 Code

List of relevant functions in the attachment:

- **ex7_position_estimation.m**: main entry;
- **navSolver.m** entry for position estimation;
- **solveGaussNewton.m** Gauss-Newton solver;
- **computeDOP.m** DOP calculator;
- **lssolver.m** solver for $\mathbf{AX} = \mathbf{b}$.
- **ex7_dop_evaluation.m** estimate 24 hours DOP values;
- **assign3.m**: GUI entry;

navSolver Some of previous made functions are also used for this assignment:

- **sp3fileParser.m**: parsing the sp3 file;
- **find_data.m**: extract relevant data of given epoch from the whole sp3 data;
- **find_neighbor_ids.m**: find K nearest neighbors for interpolation;
- **interp_sat_pos.m**: interpolation;
- **calc_pseudorange.m**: pseudorange composition;

1.4 Experiments

Position Estimation& Comparison

The antenna is assumed to be placed near the DTU 101 with the latitude, longitude and height being $(55.78575300466123, 12.525384183973078, 40)$. The epoch time is $2018 - 09 - 16 : 12 : 00 : 05$.
The position estimation results are shown in Table 1.1. It can be seen from Table 1.1 and

Table 1.1. Position Estimation Results.

	x: (m)	**y**: (m)	**z**: (m)	**dT**: (ms)
Truth	3509042.2969	779567.15431	5251066.1743	0.1
Initial	0	0	0	0
Raw	3380435.0196	721266.27907	5082767.4168	−0.2804
Corrected	3509042.2969	779567.15431	5251066.1743	0.1

Table 1.2. Position Estimation Errors.

	err_x: (m)	err_y: (m)	err_z: (m)	err_{dT}: (ms)
Raw	−128607.2773	−58300.87524	−168298.7576	−0.3804
Corrected	$-1.21e^{-08}$	$-1.05e^{-09}$	$-3.45e^{-08}$	$-1.95e^{-16}$

Table 1.2 that the difference is quite significant. After the correction of atmospheric delays and satellite clock errors, the accuracy can be improved a lot. This is easy to understand from the mathematical point of view. Revisit (1.1) and rethink the composition of the ionospheric and tropospheric delays as well as the satellite clock errors, we can simply

divide delays and errors into four groups:

$$R = \rho + \underbrace{d\rho - c(dt)}_{sat} + \underbrace{c(dT)}_{rec} + \underbrace{d_{ion} + d_{trop}}_{sat-rec} + \underbrace{\epsilon}_{random} \tag{1.26}$$

where sat means satellite dependent, rec means receiver dependent, $sat - rec$ means receiver, and satellite jointly dependent, $random$ means random error or noisy. Since the optimized vector x, y, z, dT which only contains the receiver information, it is clear that it can not correctly model the error imposed in pseudoranges. Therefore, although the adjustment still succeeds to minimize the cost function, there is no guarantee that the local minimum it finds corresponds to the truth. In fact, it can be verified that in some cases, the error at the true point is even larger than the error of the local minimum.
The convergence plots with regarding to the cost function and each adjustment are shown in Fig 1.3. The 95% confidence ellipsoid are drawn in Fig 1.4. The DOP values for this experiment are given in Table 1.3. From[**?**], we can see that the GDOP is less than 2,

Table 1.3. DOP Values.

	$HDOP$: (m)	$VDOP$: (m)	$PDOP$: (m)	$GDOP$: (ms)
Raw	0.77052	1.1923	1.4196	1.6068
Corrected	0.76687	1.1813	1.4084	1.5912

which means the geometric distribution of visible satellites is quite good for positioning. The corresponding skyplot of all visible satellites is shown in Fig 1.5. From Fig 1.5, we

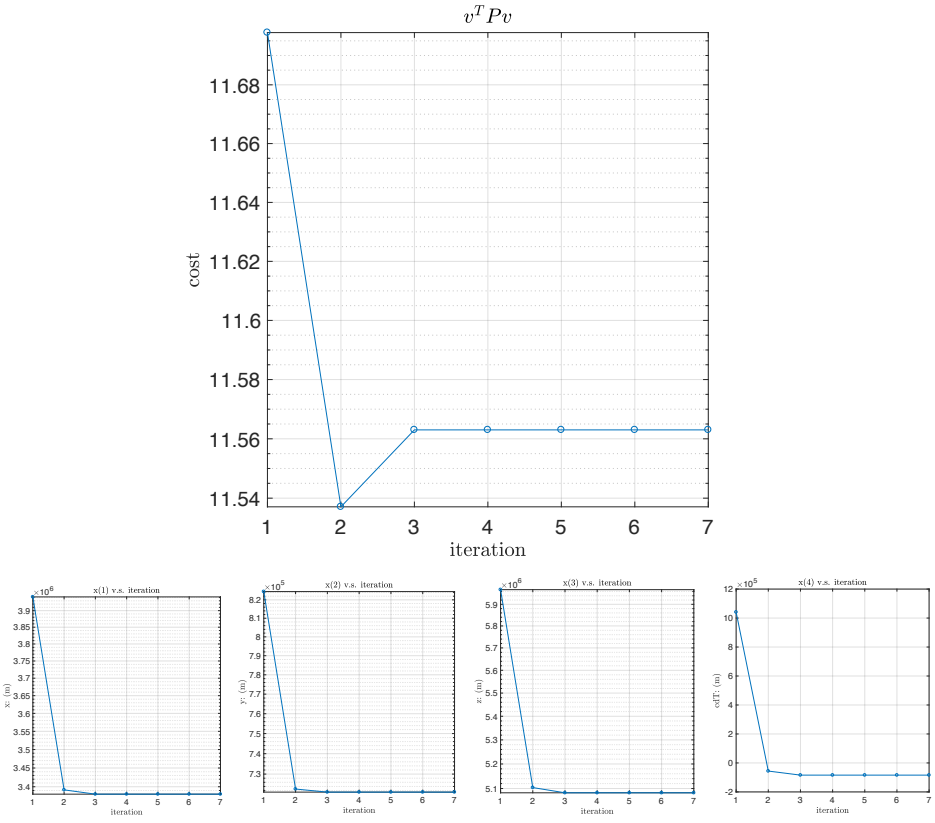

Figure 1.1. Convergence Plots for position estimation using raw pseudoranges.

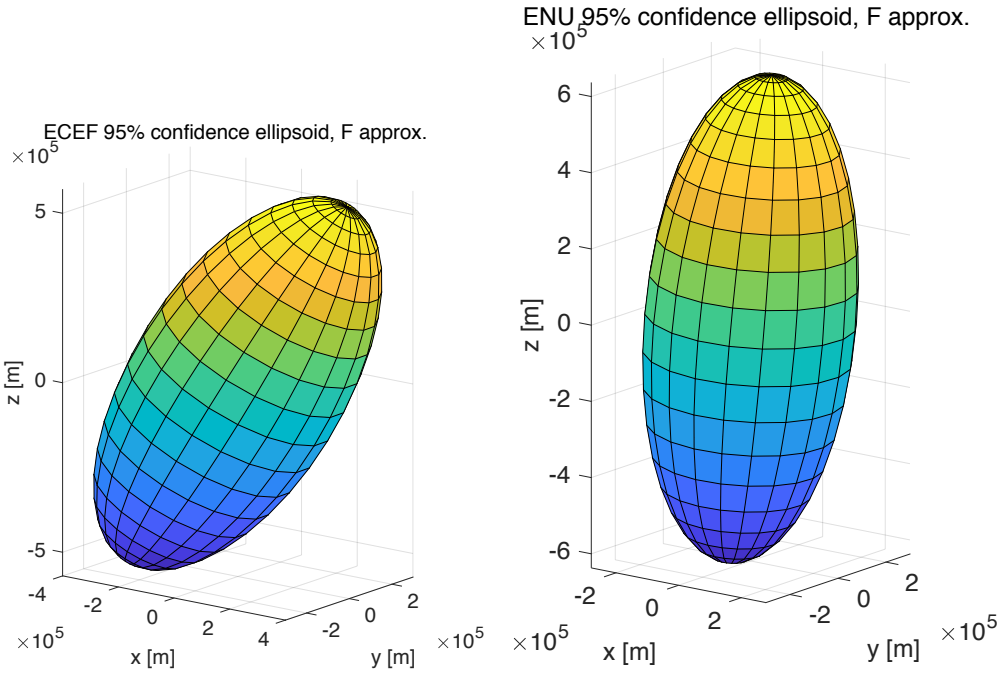

Figure 1.2. 95% Confidence Ellipsoid in ECEF and ENU coordiante frames for position estimation using raw pseudoranges.

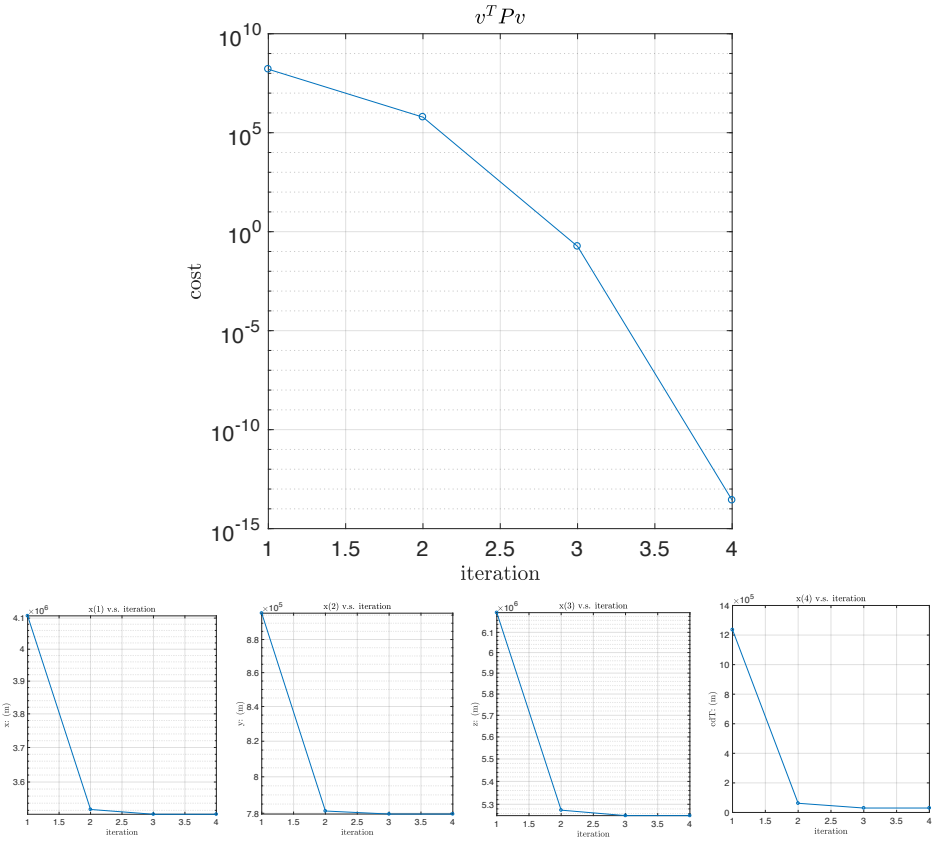

Figure 1.3. Convergence Plots for position estimation using corrected pseudoranges.

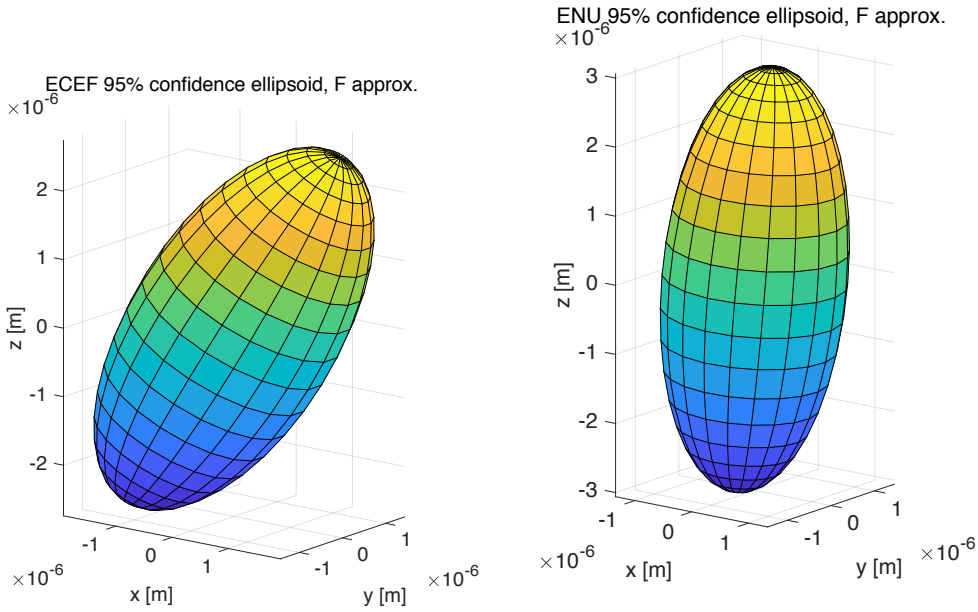

Figure 1.4. 95% Confidence Ellipsoid in ECEF and ENU coordiante frames for position estimation using corrected pseudoranges.

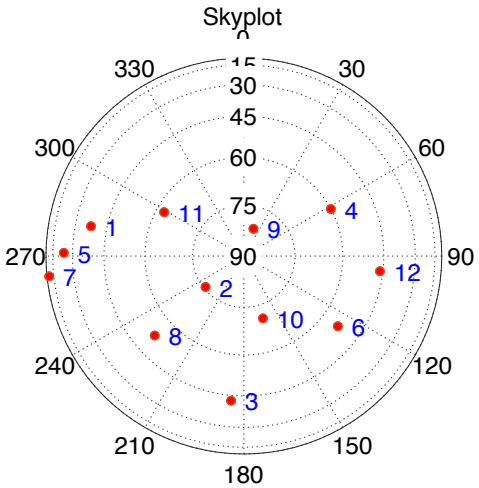

Figure 1.5. The corresponding skyplot of all visible satellites.

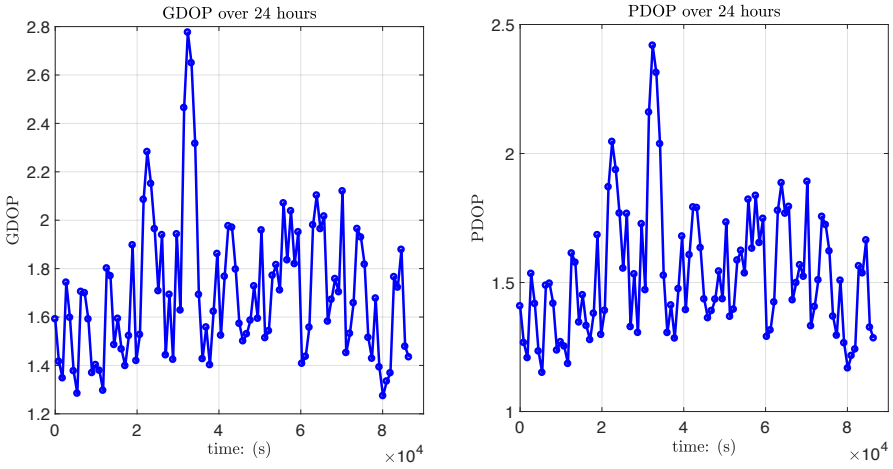

Figure 1.6. 95% GDOP and PDOP values of 24 Hours.

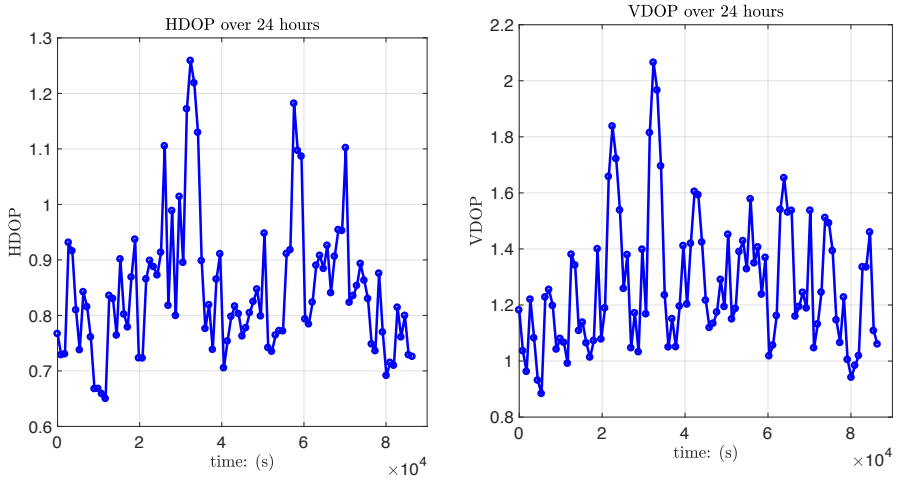

Figure 1.7. 95% HDOP and VDOP values of 24 Hours.

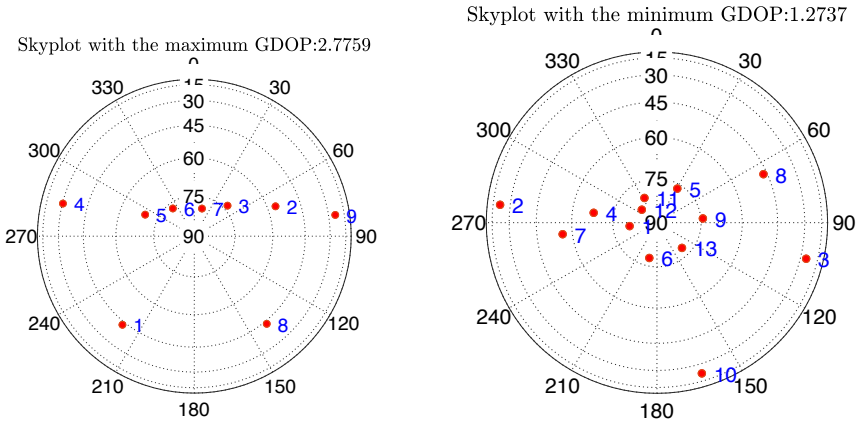

Figure 1.8. 95% Skyplots corresponds to the maximum and minimum GDOP and VDOP epoches.

can clearly see that visible satellites are placed well with several satellites along the zenith direction but also several satellites with different elevation angles, which will be beneficial for vertical position estimation.

DOP

In this experiment, I repeated the previous experiment from $2018-09-16:12:00:00$ to $2018-09-17:12:00:00$ with a time step being 15 minutes. The aim of this experiment is to evaluate the variance of those DOP values. It can be understood that the DOP values depend on the relative geometry between satellites positions and the receiver's position. The results are shown in Fig 1.6 and Fig 1.7. Moreover, the skyplots corresponding to the maximum and the minimum GDOP values are drawn in Fig 1.8. It is clear that the minimum GDOP corresponding epoch had a better geometric distribution of visible satellites than the maximum GDOP corresponding epoch. Here better geometry means more equally distributed.

1.4.1 GUI

A MATLAB GUI is designed and shown in Fig 1.9. Latitude, longitude, and altitude can be arbitrarily selected simply by sliding corresponding sliders. TEC value and receiver clock error can be easily changed. Epoch value can be either typed or selected via the corresponding sliders. By clicking the compute button, the position estimation will run

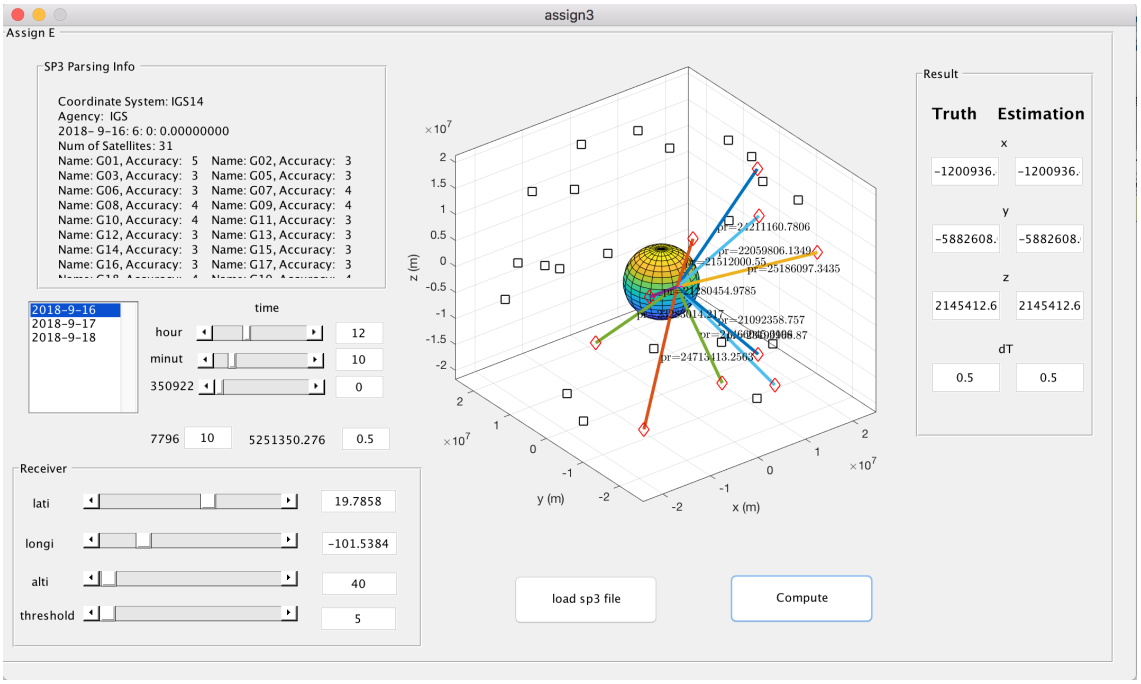

Figure 1.9. Snapshot of the designed GUI.

automatically and the results will be shown in the leftmost panel.

1.4.2 Conclusion

In this experiment, the position estimation is carried out with the least square estimation. By comparing the position estimation results, it can be shown that correctly eliminating the atmospheric delays and the satellite clock errors can significantly improve the position estimation result.

However, it should be noted here the real position estimation model is more complex than the one used here. Since in order to compensate the rotation of the ECEF coordinate for satellites' positions ($p_{sat} = R_{earth\_rotation}\ p_{sat}$), the signal transmission time has to be computed to compute how much the ECEF frame has rotated. Unfortunately, the transmission time also depends on the receiver's position since $t = \frac{\rho}{c}$. So generally speaking, the overall problem is a highly non-linear system, which means the real performance would drop a lot.

2 Assignment F: Extensions

This chapter begins with the initialization methods in 2.1.

2.1 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.

2.1.1 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:

$$R = \rho + d\rho + c(dT - dt) + d_{ion} + d_{trop} + \epsilon \tag{2.1}$$

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

$$R_c \approx \rho + c(dT) \tag{2.2}$$

Take the square on both sides,

$$(x_i^2 + y_i^2 + z_i^2 - R_c^2) - 2(x_i x + y_i y + z_i z - R_c(cdT)) + (x^2 + y^2 + z^2 - (cdT)^2) = 0 \tag{2.3}$$

Considering the inner product of Lorentz:

$$\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^T \mathbf{M} \mathbf{b} \tag{2.4}$$

$$= [a_1, a_2, a_3, a_4] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \tag{2.5}$$

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

$$\frac{1}{2}\langle \begin{bmatrix} \mathbf{x_i} \\ R_c \end{bmatrix}, \begin{bmatrix} \mathbf{x_i} \\ R_c \end{bmatrix} \rangle - \langle \begin{bmatrix} \mathbf{x_i} \\ R_c \end{bmatrix}, \begin{bmatrix} \mathbf{x} \\ cdT \end{bmatrix} \rangle + \frac{1}{2}\langle \begin{bmatrix} \mathbf{x} \\ cdT \end{bmatrix}, \begin{bmatrix} \mathbf{x} \\ cdT \end{bmatrix} \rangle = 0 \tag{2.6}$$

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

$$\mathbf{a} - \mathbf{BM}\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} + \Lambda\mathbf{1} = 0 \tag{2.7}$$

$$\Lambda = \frac{1}{2}\langle\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix}, \begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix}\rangle \tag{2.8}$$

$$\mathbf{a} = \begin{bmatrix}a_1\\ \vdots\\ a_N\end{bmatrix},\ a_i = \frac{1}{2}\langle\begin{bmatrix}\mathbf{x_i}\\ R_c\end{bmatrix}, \begin{bmatrix}\mathbf{x_i}\\ R_c\end{bmatrix}\rangle \tag{2.9}$$

Then,

$$\mathbf{BM}\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} = \mathbf{a} + \Lambda\mathbf{1} \tag{2.10}$$

$$\mathbf{B^T BM}\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} = \mathbf{B^T a} + \Lambda\mathbf{B^T 1} \tag{2.11}$$

$$\begin{bmatrix}\mathbf{x}\\ cdT\end{bmatrix} = \mathbf{M}(\mathbf{B^T B})^{-1}\mathbf{B^T}(\Lambda\mathbf{1} + \mathbf{a}) \tag{2.12}$$

From (2.8), 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:

$$p_2\Lambda^2 + p_1\Lambda + p_0 = 0 \tag{2.13}$$

$$p_2 = \langle(\mathbf{B^T B})^{-1}\mathbf{B^T 1}, (\mathbf{B^T B})^{-1}\mathbf{B^T 1}\rangle \tag{2.14}$$

$$p_1 = 2\left(\langle(\mathbf{B^T B})^{-1}\mathbf{B^T 1}, (\mathbf{B^T B})^{-1}\mathbf{B^T a}\rangle - 1\right) \tag{2.15}$$

$$p_0 = \langle(\mathbf{B^T B})^{-1}\mathbf{B^T a}, (\mathbf{B^T B})^{-1}\mathbf{B^T a}\rangle \tag{2.16}$$

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.

2.1.2 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:

$$\rho \approx \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2},\ i = 1, 2, \cdots, n \tag{2.17}$$

$$\rho^2 \approx (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2,\ i = 1, 2, \cdots, n \tag{2.18}$$

$$d\rho_{ij}^2 \approx \rho_i^2 - \rho_j^2 \tag{2.19}$$

$$d\rho_{ij}^2 \approx (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 - (x - x_j)^2 - (y - y_j)^2 - (z - z_j)^2 \tag{2.20}$$

Do some simple extensions, we will have:

$$\begin{aligned} d\rho_{ij}^2 - (x_i^2 + y_i^2 + z_i^2) + (x_j^2 + y_j^2 + z_j^2) &= 2(x_j - x_i)x \\ &\quad + 2(y_j - y_i)y + 2(z_j - z_i)z \\ &= \begin{bmatrix} 2(x_j - x_i) & 2(y_j - y_i) & 2(z_j - z_i) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \end{aligned} \tag{2.21}$$

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:

$$\begin{bmatrix} \rho_1 - \sqrt{(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2} \\ \vdots \\ \rho_N - \sqrt{(x - x_N)^2 + (y - y_N)^2 + (z - z_N)^2} \end{bmatrix} = cdT \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{2.22}$$

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

2.1.3 SOCP initialization

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

$$\text{minimize}: \sum_{i=0}^{n} \left(\rho_i - (\sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} + cdT) \right)^2 \tag{2.23}$$

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:

$$\begin{aligned} &\text{minimize}: \ t \\ &\text{subject to}: t \geq \sum_{i=0}^{n} \rho_i - l_i - cdT)^2 \\ &\qquad l_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2},\ i = 1, 2, \cdots, n \end{aligned} \tag{2.24}$$

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:

$$\begin{aligned} &\text{minimize}: \ t \\ &\text{subject to}: (t, \frac{1}{1}, \rho_0 - l_0 - cdT, \cdots, \rho_n - l_n - cdT) \in \mathcal{Q}_r^{n+3} \\ &\qquad (l_i, (x - x_i), (y - y_i), (z - z_i)) \in \mathcal{Q}^4,\ i = 1, 2, \cdots, n \end{aligned} \tag{2.25}$$

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:

$$\begin{aligned} &\text{minimize}: \ t + \lambda \sum_{i=0}^{n} l_i \\ &\text{subject to}: (t, \frac{1}{1}, \rho_0 - l_0 - cdT, \cdots, \rho_n - l_n - cdT) \in \mathcal{Q}_r^{n+3} \\ &\qquad (l_i, (x - x_i), (y - y_i), (z - z_i)) \in \mathcal{Q}^4,\ i = 1, 2, \cdots, n \end{aligned} \tag{2.26}$$

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

2.1.4 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.

2.2 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:

$$R = \rho + cdT + \epsilon \tag{2.27}$$

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:

$$f(\mathbf{x_i}|\theta) = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(R_i - d_i)^2}{2\sigma_i^2}} \tag{2.28}$$

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

$$f_n(\mathbf{x_i}|\theta) = \prod_{i=1}^{n} f(\mathbf{x_i}|\theta) \tag{2.29}$$

then the log-likelihood function can be derived as:

$$l_n(\theta) = log f_n(\mathbf{x_i}|\theta) = \sum_{i=1}^{n} log f(\mathbf{x_i}|\theta) \tag{2.30}$$

$$log f(\mathbf{x_i}|\theta) = -\frac{1}{2} log(2\pi\sigma_i^2) - \frac{(R_i - d_i)^2}{2\sigma_i^2} \tag{2.31}$$

Taking the derivative,

$$\frac{\partial l_n(\theta)}{\partial \theta} = \sum_{i=1}^{n} \frac{\partial log f(\mathbf{x_i}|\theta)}{\partial \theta} \tag{2.32}$$

$$\frac{\partial log f(\mathbf{x_i}|\theta)}{\partial \theta} = -\frac{R_i - d_i}{\sigma_i^2} [\frac{\partial d_i}{\partial x},\ \frac{\partial d_i}{\partial y},\ \frac{\partial d_i}{\partial z},\ \frac{\partial d_i}{\partial cdT}]^T \tag{2.33}$$

$$d_i = \rho_i + cdT \tag{2.34}$$

$$[\frac{\partial d_i}{\partial x},\ \frac{\partial d_i}{\partial y},\ \frac{\partial d_i}{\partial z},\ \frac{\partial d_i}{\partial cdT}]^T = [\frac{x - x_i}{\rho_i}, \frac{y - y_i}{\rho_i}, \frac{z - z_i}{\rho_i}, 1]^T \tag{2.35}$$

[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:

$$\mathbf{I}(\theta) = E\left[\frac{\partial l_n(\theta)}{\partial \theta}\frac{\partial l_n(\theta)}{\partial \theta}^T\right]. \tag{2.36}$$

$$\mathbf{I}(\theta) = \sum_{i=1}^{n}\mathbf{I}_i, \text{since } E[(R_i - d_i)(R_j - d_j)] = 0, \text{ if } i \neq j. \tag{2.37}$$

$$\mathbf{I}_i = E\left[\frac{(R_i - d_i)^2}{\sigma_i^4}\psi_i\psi_i^T\right] \tag{2.38}$$

$$= (\sigma_i^2)^{-1}\psi_i\psi_i^T \tag{2.39}$$

$$\mathbf{I}(\theta) = \begin{bmatrix}\psi_1 & \cdots & \psi_n\end{bmatrix}\begin{bmatrix}(\sigma_1^2)^{-1} & & \\ & \ddots & \\ & & (\sigma_n^2)^{-1}\end{bmatrix}\begin{bmatrix}\psi_1^T \\ \vdots \\ \psi_n^T\end{bmatrix} \tag{2.40}$$

$$\begin{bmatrix}(\sigma_1^2)^{-1} & & \\ & \ddots & \\ & & (\sigma_n^2)^{-1}\end{bmatrix} = (\sigma_1^2)^{-1}\mathbf{P},\ (\sigma_1^2)^{-1}P_{ii} = (\sigma_i^2)^{-1} \tag{2.41}$$

$$\mathbf{A} = \begin{bmatrix}\psi_1^T \\ \vdots \\ \psi_n^T\end{bmatrix} \tag{2.42}$$

$$\mathbf{I}(\theta) = (\sigma_1^2)^{-1}\mathbf{A}^T\mathbf{P}\mathbf{A} \tag{2.43}$$

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

$$C_\theta \geq \mathbf{I}(\theta)^{-1} \tag{2.44}$$

$$\mathbf{I}(\theta)^{-1} = (\sigma_1^2)(\mathbf{A}^T\mathbf{P}\mathbf{A})^{-1} \tag{2.45}$$

Clearly, according to [**?**], 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:

$$\mathbf{Q}_{DOP} = \frac{\mathbf{A^T P A}^{-1}}{\sigma_{prior}^2} \tag{2.46}$$

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.

2.3 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.

2.3.1 Fast GDOP computation

revisit the definition of GDOP:

$$GDOP = \sqrt{trace(\mathbf{A^T P A}^{-1})} = \sqrt{\frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \frac{1}{\lambda_3} + \frac{1}{\lambda_4}} \tag{2.47}$$

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:

$$\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4 = eig\left(\frac{\mathbf{A^T P A}}{\sigma_0^2}\right) \tag{2.48}$$

$$GDOP = \sqrt{\frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \frac{1}{\lambda_3} + \frac{1}{\lambda_4}} \tag{2.49}$$

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:

$$h_1(\lambda) = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = trace(\mathbf{M}) \tag{2.50}$$

$$h_2(\lambda) = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 + \lambda_4^2 = trace(\mathbf{M^2}) \tag{2.51}$$

$$h_3(\lambda) = \lambda_1^3 + \lambda_2^3 + \lambda_3^3 + \lambda_4^3 = trace(\mathbf{M^3}) \tag{2.52}$$

$$h_4(\lambda) = \lambda_1 * \lambda_2 * \lambda_3 * \lambda_4 = det(\mathbf{M}) \tag{2.53}$$

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:

$$p(x_{\sigma_1},\ x_{\sigma_2},\ \cdots,\ x_{\sigma_n}) = p(x_1,\ x_2,\ \cdots,\ x_n) \tag{2.54}$$

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:

$$p_k(x_1,\ x_2,\ \cdots,\ x_n) = x_1^2 + x_2^2 + \cdots + x_n^2 \tag{2.55}$$

- Elementary symmetric polynomials:

$$e_k(x_1,\ x_2,\ \cdots,\ x_n) = \sum_{1 \le j_1 < j_2 < \cdots < j_k \le n} x_{j_1} x_{j_2} \cdots x_{j_k} \tag{2.56}$$

  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.

$$\begin{aligned} &k(-1)^k e_k(x_1,\ x_2,\ \cdots,\ x_n) + \\ &\quad \sum_{i=1}^{k} (-1)^{i+k} e_i(x_1,\ x_2,\ \cdots,\ x_n) p_{k-i}(x_1,\ x_2,\ \cdots,\ x_n) = 0 \end{aligned} \tag{2.57}$$

Now, revisit (2.47):

$$GDOP = \sqrt{\frac{\lambda_1\lambda_2\lambda_3 + \lambda_1\lambda_2\lambda_4 + \lambda_1\lambda_3\lambda_4 + \lambda_2\lambda_3\lambda_4}{\lambda_1\lambda_2\lambda_3\lambda_4}} \tag{2.58}$$

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

$$e_4(\lambda) = \frac{1}{3}\left[\frac{1}{2}\left(p_1(\lambda)^2 - p_2(\lambda)\right) p_1(\lambda) - p_1(\lambda)p_2(\lambda) + p_3(\lambda)\right] \tag{2.59}$$

$$p_1(\lambda) = h_1(\lambda) \tag{2.60}$$

$$p_2(\lambda) = h_2(\lambda) \tag{2.61}$$

$$p_3(\lambda) = h_3(\lambda) \tag{2.62}$$

$$\Leftrightarrow$$

$$e_4(\lambda) = \frac{0.5h_1(\lambda)^3 - 1.5h_1(\lambda)h_2(\lambda) + h_3(\lambda)}{3} \tag{2.63}$$

Overall, the GDOP can be written as

$$GDOP = \sqrt{\frac{0.5h_1(\lambda)^3 - 1.5h_1(\lambda)h_2(\lambda) + h_3(\lambda)}{3h_4}} \tag{2.64}$$

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

$$\begin{aligned}|\mathbf{A}-\lambda\mathbf{I}| =&(\lambda-\lambda_1)(\lambda-\lambda_2)(\lambda-\lambda_3)(\lambda-\lambda_4)\\ =&\lambda^4-(\lambda_1+\lambda_2+\lambda_3+\lambda_4)\,\lambda^3\\ &+(\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_1\lambda_4+\lambda_2\lambda_3+\lambda_2\lambda_4+\lambda_3\lambda_4)\lambda^2\\ &-(\lambda_1\lambda_2\lambda_3+\lambda_1\lambda_2\lambda_4+\lambda_1\lambda_3\lambda_4+\lambda_2\lambda_3\lambda_4)\lambda\\ &+\lambda_1\lambda_2\lambda_3\lambda_4\end{aligned} \tag{2.65}$$

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

$$|\mathbf{A}-\lambda\mathbf{I}| = p_4\lambda^4+p_3\lambda^3+p_2\lambda^2+p_1\lambda+p_0 \tag{2.66}$$

It is easy to see that

$$GDOP=\sqrt{\frac{-p_1}{p_0}} \tag{2.67}$$

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 2.1.

Table 2.1. 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

2.4 Augmented State

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

$$\rho = R+\underbrace{cdT}_{receiver}-\underbrace{cdt}_{satellite}+\underbrace{d_{iono}}_{satellite,receiver}+\underbrace{d_{trop}}_{satellite,receiver} \tag{2.68}$$

$$R=\sqrt{(x-x_{sv})^2+(y-y_{sv})^2+(z-z_{sv})^2} \tag{2.69}$$

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:

$$d_{iono} = x_5 * OF,\ OF = \left(1 - \left(\frac{RE * sin(zenith)}{RE + hI}\right)^2\right)^{-1/2} \tag{2.70}$$

$$d_{trop} = \frac{x_6}{sin(elv)} \tag{2.71}$$

With known satellite positions, Eq 2.71 can be written implicitly as the function of $\mathbf{x} = x,\ y,\ z,\ cdT$ as

$$d_{iono} = x_5 * f_1(\mathbf{x}, x_{sv}) \tag{2.72}$$

$$d_{trop} = \frac{x_6}{f_2(\mathbf{x}, x_{sv})} \tag{2.73}$$

$$\Leftrightarrow$$

$$d_{iono} = f_1^*(\mathbf{x}_{aug}) \tag{2.74}$$

$$d_{trop} = f_2^*(\mathbf{x}_{aug}) \tag{2.75}$$

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

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.

2.4.1 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 2.2. 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.

References

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

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