Kinematic Relative GPS Positioning Using State-Space Models: Computational Aspects

Xiao-Wen Chang, Mengjun Huang, *McGill University, Canada*

BIOGRAPHIES

Dr. Chang is Associate Professor of Computer Science at McGill University. He holds a B.Sc and an M.Sc in Computational Mathematics from Nanjing University, China, and a Ph.D in Computer Science from McGill University. His main research area is scientific computing with particular emphasis on matrix computations and applications. He has been involved in GNSS research since 1998 and his current research interests include various estimation techniques for GNSS positioning and fast algorithms for ambiguity determination. His publications can be found at www.cs.mcgill.ca/∼chang.

Mr. Huang has just graduated from McGill University with an M.Sc in Computer Science. He holds a B.Sc in Computer Science from East China University of Science and Technology. He is interested in numerical algorithms and software design for linear estimation and their applications in GNSS positioning.

ABSTRACT

For kinematic relative GPS positioning the often used approach is to establish a discrete linearized state-space model, and then apply a standard Kalman filtering technique to estimate the positions. However, there are a few shortcomings with this approach. Since the integer ambiguities are constant (suppose there are no cycle slips), the state-space model for positioning is special. In order to apply a standard Kalman filtering technique, this approach enlarges the time-variant state vectors (e.g., position-velocity vectors) to include the integer ambiguities. This not only makes the estimation problem larger—leading to higher computational cost, but also results in singular covariance matrices for the process noise vectors in the process equations since the covariance matrix corresponding to the integer ambiguity vector is a zero matrix. Then some standard Kalman filtering techniques, such as the square root information filter cannot be applied. The typical method people have used to handle this singularity problem is to artificially assign a small covariance matrix to the ambiguity vector. But this is an approximation and does not look elegant in theory.

In order to avoid the above problems, for short baseline kinematic relative positioning, we present a computationally efficient and numerically reliable approach. Our approach can be regarded as an extension of the standard information square root filtering technique. The basic idea is to write all available measurement equations (we use both code and carrier phase measurements based on L1 signals) and process equations together to form a large linearized model, which has special structures. Then based on this model, we develop a recursive algorithm to compute the least squares estimates of positions and velocities. We obtain not only the estimate of the current position (this is called filtering), but also the estimates of previous positions (this is called smoothing). Our algorithm is computationally efficient and numerically reliable.

One thing which makes the estimation problem much more complicated and also more interesting is that the integer ambiguity vector may change due to cycle slips, loss of signals, and satellite rising/setting. In this paper we show how to handle this problem in our algorithm. Unlike the typical literature in GNSS, we give details about the algorithm so that people can implement it without difficulties.

1 INTRODUCTION

The often used approach to kinematic relative GPS positioning is to employ differencing techniques to establish linearized state-space models and then apply the Kalman filtering techniques to estimate the receiver positions (and velocities et al), see, e.g., [8, 9, 11, 12, 13, 14, 17, 19]. The state-space model can be written as:

$$\boldsymbol{y}_k = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{z} + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_k^v), \tag{1}$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}_k \boldsymbol{x}_k + \boldsymbol{u}_{k+1}, \quad \boldsymbol{u}_{k+1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{k+1}^u), \tag{2}$$

for $k = 1, 2, \ldots$, where (1) and (2) are referred to as the measurement (or observation) equation and the process (or state) equation, respectively, $\boldsymbol{x}_k$ is the vector including position, velocity et al, $\boldsymbol{z}$ is the integer ambiguity vector, $\boldsymbol{A}_k$, $\boldsymbol{B}_k$ and $\boldsymbol{F}_k$ are known matrices, and $\boldsymbol{v}_k$ and $\boldsymbol{u}_{k+1}$ are the random measurement noise vector and process noise vector, respectively, following normal distributions. In order to apply a standard Kalman filtering technique, a typical approach is to stack $\boldsymbol{x}_k$ and $\boldsymbol{z}$ as a state vector, so the above model is transformed to:

$$\boldsymbol{y}_k = \begin{bmatrix} \boldsymbol{A}_k & \boldsymbol{B}_k \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{z} \end{bmatrix} + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_k^v), \tag{3}$$

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{z} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_k & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{z} \end{bmatrix} + \begin{bmatrix} \boldsymbol{u}_{k+1} \\ \boldsymbol{w}_{k+1} \end{bmatrix}, \tag{4}$$

$$\begin{bmatrix} \boldsymbol{u}_{k+1} \\ \boldsymbol{w}_{k+1} \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{\Sigma}_{k+1}^u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}\right). \tag{5}$$

This approach not only (unnecessarily) makes the estimation problem larger so that the computational cost will be higher, but also results in a singular covariance matrix for each process noise vector (note that the noise covariance matrix of $\boldsymbol{w}_{k+1}$ is a zero matrix in (5)) so that some standard Kalman filter techniques, such as the square root information filter (see, e.g., [10]) cannot be applied. To overcome this singularity problem, the typical method (see, e.g., [12, 14]) is to artificially assigned a scaled identity matrix $\delta \boldsymbol{I}$ (δ is a small positive scalar) as the covariance matrix to $\boldsymbol{w}_{k+1}$. But this approximation does not conform with the physical characteristic of the integer ambiguity vector and may cause a numerical problem if δ is too small or lead to a loss of accuracy of the position estimates if it is not small enough.

In this paper, for short baseline kinematic positioning we propose a new approach to avoid the above difficulties. We combine all the measurement equations (for L1 signals) and process equations to form a linearized state-space model. Based on the model, we design a recursive approach to compute the least squares (LS) estimates of the positions. Computing the corresponding error covariance matrices of the estimates are also considered. We exploit the structure of the model to make the algorithm computationally efficient and use orthogonal transformations to make it numerically reliable, and also take the storage efficiency into consideration. This approach avoids the mentioned drawbacks of the current approach and is an extension of the recursive least squares (RLS) method presented in [4], which used only the code and carrier measurement equations for positioning, and an extension of the square root information filter given in [16].

The rest of this paper is organized as follows. In Section 2 we present the state-space model. In Section 3 we use orthogonal transformations which make full use of the structures of the model to compute the LS estimates of the positions and velocities, and the corresponding error covariance matrices, and show how to handle satellites rising and setting. Finally a summary is given in Section 4.

Throughout this paper bold upper case letters are used to denote matrices and bold lower case letters are used to denote vectors. The unit matrix is denoted by $\boldsymbol{I}$ and its ith column by $\boldsymbol{e}_i$, while $\boldsymbol{e} \equiv [1, 1, \ldots, 1]^T$ (we use $\equiv$ to mean "is defined to be"). The 2-norm of a vector or a matrix is denoted by $\|\cdot\|_2$. The jth column of a matrix $\boldsymbol{A}$ is denoted by $\boldsymbol{A}(:, j)$, the (i, j) entry of $\boldsymbol{A}$ by $\boldsymbol{A}(i, j)$, and the ith entry of a vector $\boldsymbol{a}$ by $\boldsymbol{a}(i)$. The mean and covariance are denoted by $\mathcal{E}\{\cdot\}$ and $\text{cov}\{\cdot\}$, respectively. For a random vector $\boldsymbol{x}$ which is normally distributed with mean $\boldsymbol{u}$ and covariance $\boldsymbol{\Sigma}$, we denote it by $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{u}, \boldsymbol{\Sigma})$.

2 THE MATHEMATICAL MODEL

In this section, we establish a state-space model for short baseline kinematic relative positioning based on the measurement equations and process equations.

In kinematic relative positioning, one receiver is set up at a surveyed site whose exact position is known, and the other receiver is roving and its positions are to be estimated. We assume that the carrier phase and code measurements of the reference receiver are available at the roving receiver. We also assume the baseline is short such that the ionospheric reflection and tropospheric reflection can (almost) be canceled after applying the between-receiver single-difference technique. We consider only the L1 carrier since many receivers can only receive the L1 signal. But it is easy to extend our approach to the dual frequency case.

For short baseline relative positioning, in [4], we derived the following single differenced carrier phase and code measurement equations corresponding to satellite i at epoch k

$$\phi_k^i = \lambda^{-1}(\omega_k^i \boldsymbol{e}_k^i)^T \boldsymbol{b}_k + N^i + \beta_k^\phi + \nu_k^i, \tag{6}$$

$$\rho_k^i = \lambda^{-1}(\omega_k^i \boldsymbol{e}_k^i)^T \boldsymbol{b}_k + \beta_k^\rho + \mu_k^i, \tag{7}$$

where the units of each of the terms in above equations are *number of wavelengths*, ϕ_k^i and ρ_k^i are the single differenced carrier phase measurement and code measurement, respectively; λ is the wavelength of the

L1 carrier; $\boldsymbol{b}_k$ is the baseline vector pointing from the reference receiver to the roving receiver; ω_k^i satisfies

$$\omega_k^i = \|2\boldsymbol{h}_r^i - \boldsymbol{b}_k\|_2/(\|\boldsymbol{h}_r^i\|_2 + \|\boldsymbol{h}_r^i - \boldsymbol{b}_k\|_2), \qquad (8)$$

with $\boldsymbol{h}_r^i$ being the vector pointing from the reference receiver to satellite i (note that ω_k^i is close to 1); $\boldsymbol{e}_k^i$ is the unit vector pointing from the midpoint of the baseline to satellite i; N^i is the single differenced integer ambiguity; β_k^ϕ is the single differenced receiver clock error (including hardware delay and initial phase) for the carrier phase measurement; β_k^ρ is the single differenced receiver clock error (including hardware delay) for the code measurement; and ν_k^i and μ_k^i are the single differenced noises (including multipath errors) for the carrier phase measurement and the code measurement, respectively. We have an important assumption (see [19, Sec 3.4]) for (6) and (7) that all the carrier phase noises ν_k^i (or code noises μ_l^j) for different satellites and different epochs are unbiased independently distributed with the same normal distribution, and ν_k^i and μ_l^j are also assumed to be independent.

Suppose there are m visible satellites and define

$$\boldsymbol{y}_k^\phi \equiv \begin{bmatrix}\phi_k^1\\ \cdot \\ \phi_k^m\end{bmatrix},\ \boldsymbol{y}_k^\rho \equiv \begin{bmatrix}\rho_k^1\\ \cdot \\ \rho_k^m\end{bmatrix},\ \boldsymbol{E}_k \equiv \lambda^{-1}\begin{bmatrix}(\omega_k^1\boldsymbol{e}_k^1)^T\\ \cdot \\ (\omega_k^m\boldsymbol{e}_k^m)^T\end{bmatrix},$$

$$\boldsymbol{a} \equiv \begin{bmatrix}N^1\\ \cdot \\ N^m\end{bmatrix},\quad \boldsymbol{v}_k^\phi \equiv \begin{bmatrix}\nu_k^1\\ \cdot \\ \nu_k^m\end{bmatrix},\quad \boldsymbol{v}_k^\rho \equiv \begin{bmatrix}\mu_k^1\\ \cdot \\ \mu_k^m\end{bmatrix}. \qquad (9)$$

We can combine m single differenced carrier-phase measurement equations (6) and m single differenced code measurement equations (7) together to give

$$\boldsymbol{y}_k^\phi = \boldsymbol{E}_k\boldsymbol{b}_k + \boldsymbol{a} + \beta_k^\phi\boldsymbol{e} + \boldsymbol{v}_k^\phi,\ \boldsymbol{v}_k^\phi \sim \mathcal{N}(\boldsymbol{0}, \sigma_\phi^2\boldsymbol{I}_m), \quad (10)$$

$$\boldsymbol{y}_k^\rho = \boldsymbol{E}_k\boldsymbol{b}_k + \beta_k^\rho\boldsymbol{e} + \boldsymbol{v}_k^\rho,\ \boldsymbol{v}_k^\rho \sim \mathcal{N}(\boldsymbol{0}, \sigma_\rho^2\boldsymbol{I}_m), \qquad (11)$$

where we assume that the standard deviations σ_ϕ and σ_ρ are known.

Note that in $\boldsymbol{E}_k$, $\omega_k^i\boldsymbol{e}_k^i$ depends on the baseline $\boldsymbol{b}_k$. In other words, (10) and (11) are actually nonlinear equations in terms of $\boldsymbol{b}_k$. We write

$$\boldsymbol{E}_k \equiv \boldsymbol{E}(\boldsymbol{b}_k). \qquad (12)$$

This $\boldsymbol{E}_k$ is known once $\boldsymbol{b}_k$ is known. Since $\boldsymbol{E}(\boldsymbol{b}_k)$ is not very sensitive to changes in $\boldsymbol{b}_k$, with receiver's dynamic available, it is usually sufficient to use the predicted estimate $\boldsymbol{b}_{k|k-1}$ of $\boldsymbol{b}_k$ (see the end of Section 3.1) to compute $\boldsymbol{E}(\boldsymbol{b}_{k|k-1})$ and use it to replace $\boldsymbol{E}_k$ in (10) and (11). Then, when $\boldsymbol{y}_k^\rho$ and $\boldsymbol{y}_k^\phi$ are available, we can obtain a better estimate of $\boldsymbol{b}_k$ by the method to be given later. We could use the better estimate to re-compute an approximation to $\boldsymbol{E}_k$ and estimate $\boldsymbol{b}_k$ again if necessary. Thus from now we just assume that the $\boldsymbol{E}_k$ in (10) and (11) are known.

In the single differenced measurement equations (10) and (11), there are still the unknown errors β_k^ρ and β_k^ϕ. A typical way to remove these errors is by using the double difference technique. However, double difference makes the measurements correlated. Thus, we just follow [3] and [4] to use orthogonal transformations to eliminate these errors, leaving the measurement equations still uncorrelated.

We first eliminate β_k^ϕ in the carrier-phase measurement equation (10). Let $\boldsymbol{P} \in \mathcal{R}^{m\times m}$ be a Householder transformation (see, e.g., [6, p209]) such that

$$\boldsymbol{P}\boldsymbol{e} = \sqrt{m}\boldsymbol{e}_1,\quad \boldsymbol{P} \equiv \boldsymbol{I} - \frac{2\boldsymbol{u}\boldsymbol{u}^T}{\boldsymbol{u}^T\boldsymbol{u}},\quad \boldsymbol{u} = \boldsymbol{e}_1 - \frac{\boldsymbol{e}}{\sqrt{m}}. \quad (13)$$

Partition $\boldsymbol{P} \equiv \begin{bmatrix}\boldsymbol{p}^T\\ \bar{\boldsymbol{P}}\end{bmatrix}$, where

$$\bar{\boldsymbol{P}} = \Big[\frac{\boldsymbol{e}}{\sqrt{m}},\ \boldsymbol{I}_{m-1} - \frac{\boldsymbol{e}\boldsymbol{e}^T}{m-\sqrt{m}}\Big].$$

Multiplying both sides of (10) by $\boldsymbol{P}$ from the left, we obtain

$$\begin{bmatrix}\boldsymbol{p}^T\boldsymbol{y}_k^\phi\\ \bar{\boldsymbol{P}}\boldsymbol{y}_k^\phi\end{bmatrix} = \begin{bmatrix}\boldsymbol{p}^T\boldsymbol{E}_k\\ \bar{\boldsymbol{P}}\boldsymbol{E}_k\end{bmatrix}\boldsymbol{b}_k + \begin{bmatrix}\boldsymbol{p}^T\\ \bar{\boldsymbol{P}}\end{bmatrix}\boldsymbol{a} + \begin{bmatrix}\sqrt{m}\beta_k^\phi\\ \boldsymbol{0}\end{bmatrix} + \begin{bmatrix}\boldsymbol{p}^T\boldsymbol{v}_k^\phi\\ \bar{\boldsymbol{P}}\boldsymbol{v}_k^\phi\end{bmatrix}. \quad (14)$$

Since only the first equation in (14) includes the term β_k^ϕ, it can be dropped off for position estimation to give:

$$\bar{\boldsymbol{P}}\boldsymbol{y}_k^\phi = \bar{\boldsymbol{P}}\boldsymbol{E}_k\boldsymbol{b}_k + \bar{\boldsymbol{P}}\boldsymbol{a} + \bar{\boldsymbol{P}}\boldsymbol{v}_k^\phi, \qquad (15)$$

where $\bar{\boldsymbol{P}}\boldsymbol{v}_k^\phi \sim \mathcal{N}(\boldsymbol{0}, \sigma_\phi^2\boldsymbol{I}_{m-1})$. We have used the simple Householder transformation instead of the often used double difference technique to eliminate the unknown error β_k^ϕ, leaving the transformed measurements still uncorrelated. Since $\bar{\boldsymbol{P}}$ is $(m-1)\times m$, it does not have full column rank and we will not be able to get a unique estimate of $\boldsymbol{a}$. If we set $\bar{\boldsymbol{P}}\boldsymbol{a}$ as a new vector, we would lose the integer nature of $\boldsymbol{a}$. So following [3] and [4] we introduce the double differenced integer ambiguity (DDIA) vector $\boldsymbol{z}$:

$$\boldsymbol{z} = [N^2 - N^1, N^3 - N^1, \ldots, N^m - N^1]^T, \qquad (16)$$

where without loss of generality we choose satellite 1 as the *reference* satellite. Note that $\boldsymbol{z}$ is still a vector of integers. Define

$$\boldsymbol{G} \equiv \boldsymbol{I}_{m-1} - \frac{\boldsymbol{e}\boldsymbol{e}^T}{m-\sqrt{m}}, \qquad \boldsymbol{J} \equiv [-\boldsymbol{e}, \boldsymbol{I}_{m-1}]. \quad (17)$$

It is easy to verify from (13) that $\boldsymbol{G}$ is nonsingular and

$$\bar{\boldsymbol{P}} = \boldsymbol{G}\boldsymbol{J}, \qquad \bar{\boldsymbol{P}}\boldsymbol{a} = \boldsymbol{G}\boldsymbol{J}\boldsymbol{a} = \boldsymbol{G}\boldsymbol{z}. \qquad (18)$$

Thus (15) becomes

$$\bar{\boldsymbol{P}}\boldsymbol{y}_k^{\phi} = \bar{\boldsymbol{P}}\boldsymbol{E}_k\boldsymbol{b}_k + \boldsymbol{G}\boldsymbol{z} + \bar{\boldsymbol{P}}\boldsymbol{v}_k^{\phi}. \quad (19)$$

Similarly, we can eliminate β_k^{ρ} from the code measurement equation (11) and obtain

$$\bar{\boldsymbol{P}}\boldsymbol{y}_k^{\rho} = \bar{\boldsymbol{P}}\boldsymbol{E}_k\boldsymbol{b}_k + \bar{\boldsymbol{P}}\boldsymbol{v}_k^{\rho}, \quad (20)$$

where $\bar{\boldsymbol{P}}\boldsymbol{v}_k^{\rho} \sim \mathcal{N}(\boldsymbol{0}, \sigma_\rho^2 \boldsymbol{I}_{m-1})$.

In order to combine (19) and (20), we define

$$\boldsymbol{y}_k \equiv \begin{bmatrix} \sigma_\phi^{-1}\bar{\boldsymbol{P}}\boldsymbol{y}_k^{\phi} \\ \sigma_\rho^{-1}\bar{\boldsymbol{P}}\boldsymbol{y}_k^{\rho} \end{bmatrix}, \quad \boldsymbol{H}_k \equiv \begin{bmatrix} \sigma_\phi^{-1}\bar{\boldsymbol{P}}\boldsymbol{E}_k \\ \sigma_\rho^{-1}\bar{\boldsymbol{P}}\boldsymbol{E}_k \end{bmatrix}, \quad (21)$$

$$\boldsymbol{B} \equiv \begin{bmatrix} \sigma_\phi^{-1}\boldsymbol{G} \\ \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{v}_k \equiv \begin{bmatrix} \sigma_\phi^{-1}\bar{\boldsymbol{P}}\boldsymbol{v}_k^{\phi} \\ \sigma_\rho^{-1}\bar{\boldsymbol{P}}\boldsymbol{v}_k^{\rho} \end{bmatrix},$$

Thus we can rewrite (19) and (20) as

$$\boldsymbol{y}_k = \boldsymbol{H}_k\boldsymbol{b}_k + \boldsymbol{B}\boldsymbol{z} + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{2(m-1)}). \quad (22)$$

In this measurement equation, we used the geocentric Cartesian coordinate system. However, we would like to work with the local geodetic coordinate system later. This can easily be done by pre-multiplying $\boldsymbol{b}_k$ by the orthogonal transformation matrix which transforms the coordinates in the geocentric Cartesian coordinate system to the coordinates in the local geodetic coordinate system (N,E,U) with the reference receiver as the origin, and by post-multiplying $\boldsymbol{H}_k$ by the transpose of the orthogonal transformation matrix in (22). For details on the transformations of coordinates in different coordinate systems, see, e.g., [12, Secs 6.2, 7.1]. To simplify the notation, we just assume that the components of $\boldsymbol{b}_k \equiv [x_k, y_k, z_k]^T$ in (22) are already the coordinates in the local geodetic coordinate system.

Now we would like to introduce the process equations. There are different types of process equations to model the dynamics of receivers, see, e.g., [1, 2]. In this paper, we adopt the following process equations which are suitable for short baseline relative positioning with low dynamics (see, e.g., [2, Chap 6], [12, Sec 10.3.3], [14, Appendix L])

$$\boldsymbol{x}_{k+1} = \bar{\boldsymbol{F}}\boldsymbol{x}_k + \bar{\boldsymbol{u}}_{k+1}, \quad \bar{\boldsymbol{u}}_{k+1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}^{\bar{u}}), \quad (23)$$

for $k = 1, 2, \ldots$, where

$$\boldsymbol{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k, z_k, \dot{z}_k]^T$$

with $\dot{x}_k$, $\dot{y}_k$, $\dot{z}_k$ being the velocities in each direction, the transition matrix $\bar{\boldsymbol{F}}$ satisfies

$$\bar{\boldsymbol{F}} = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{C} \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix},$$

with T being the time interval between two consecutive epochs, and the noise covariance matrix

$$\boldsymbol{\Sigma}^{\bar{u}} = \begin{bmatrix} q_x\boldsymbol{W} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & q_y\boldsymbol{W} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & q_z\boldsymbol{W} \end{bmatrix}, \quad \boldsymbol{W} = \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix}$$

with q_x, q_y and q_z being the power spectral densities of the process noise.

In order to make the state vector $\boldsymbol{b}_k$ in the measurement equation (22) the same as the state vector $\boldsymbol{x}_k$ in the process equation (23), we modify the matrix $\boldsymbol{H}_k$ to include three zero columns:

$$\boldsymbol{A}_k \equiv [\boldsymbol{H}_k(:,1), \boldsymbol{0}, \boldsymbol{H}_k(:,2), \boldsymbol{0}, \boldsymbol{H}_k(:,3), \boldsymbol{0}],$$

thus (22) can be rewritten as

$$\boldsymbol{y}_k = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{z} + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{2(m-1)}). \quad (24)$$

In order to design a recursive algorithm for position-velocity estimation, we need to have an estimate and its error covariance matrix for the initial state vector $\boldsymbol{x}_1$. The usual assumption is that we have

$$\mathcal{E}\{\boldsymbol{x}_1\} = \boldsymbol{x}_{1|0}, \quad \text{cov}\{\boldsymbol{x}_{1|0} - \boldsymbol{x}_1\} = \boldsymbol{\Sigma}_{1|0},$$

which can be rewritten as

$$\boldsymbol{x}_{1|0} = \boldsymbol{x}_1 - \bar{\boldsymbol{u}}_1, \; \mathcal{E}\{\bar{\boldsymbol{u}}_1\} = \boldsymbol{0}, \; \text{cov}\{\bar{\boldsymbol{u}}_1\} = \boldsymbol{\Sigma}_{1|0}. \quad (25)$$

In practice, we can choose $\boldsymbol{x}_{1|0}$ and $\boldsymbol{\Sigma}_{1|0}$ by the following strategy. At the beginning ($k = 1$), we may have some idea about the location of the roving receiver, so can give an initial estimate of $\boldsymbol{b}_1$. If we do not have any good information about the location of the roving receiver, we can take a zero vector as an initial estimate of $\boldsymbol{b}_1$. Let the initial estimate of $\boldsymbol{b}_1$ be denoted by $[x_{1|0}, y_{1|0}, z_{1|0}]^T$. For each velocity component, we take zero as an initial estimate. Thus the initial estimate of $\boldsymbol{x}_1$ is

$$\boldsymbol{x}_{1|0} = [x_{1|0}, 0, y_{1|0}, 0, z_{1|0}, 0]^T.$$

Following [1], we take the error covariance matrix $\boldsymbol{\Sigma}_{1|0}$ of $\boldsymbol{x}_{1|0}$ to be a diagonal matrix. Due to the large uncertainty of $\boldsymbol{x}_{1|0}$, the diagonal elements of $\boldsymbol{\Sigma}_{1|0}$ should usually take large values. The choice of specific values depends on a specific application.

The equations (23)–(25) give the mathematical model for our positioning algorithm.

3 A RECURSIVE LS ALGORITHM

If a good process model for the roving receiver is not available, [4] has already shown how to use only measurement equations (see (22)) to design an efficient and numerically reliable recursive LS algorithm for positioning. Now we would like to extend this approach to the case where the process model (23) which is assumed to be an accurate model is available. In the extension, we will use the ideas given in [16], which presented a square-root information filter (a variant of the standard Kalman filter) to compute the LS estimates for a standard state-space model (formed by equation (1) with no the $\boldsymbol{B}_k\boldsymbol{z}$ term and equation (2)).

3.1 Position Estimation

Following [16], we first transform (23) and (25) so that the process noise vectors will be identically distributed with zero mean and unit covariance matrices as the measurement noise vectors in (24). Suppose we have the Cholesky factorizations

$$\boldsymbol{\Sigma}_{1|0}^{-1} = \boldsymbol{U}_1^T\boldsymbol{U}_1, \qquad (\boldsymbol{\Sigma}^{\bar{u}})^{-1} = \boldsymbol{U}^T\boldsymbol{U}, \tag{26}$$

where $\boldsymbol{U}_1$ and $\boldsymbol{U}$ are upper triangular. Notice that the time-invariant process noise covariance matrix $\boldsymbol{\Sigma}^{\bar{u}}$ is in a block-diagonal form, so we can easily obtain

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{U_w}/\sqrt{q_x} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{U_w}/\sqrt{q_y} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U_w}/\sqrt{q_z} \end{bmatrix},$$

where

$$\boldsymbol{U_w} = \begin{bmatrix} 6/(T\sqrt{3T}) & -\sqrt{3}/\sqrt{T} \\ 0 & 1/\sqrt{T} \end{bmatrix}.$$

Multiplying both sides of (25) and (23) by $\boldsymbol{U}_1$ and $\boldsymbol{U}$ from the left, respectively, leading to

$$\boldsymbol{c} \equiv \boldsymbol{U}_1\boldsymbol{x}_{1|0} = \boldsymbol{U}_1\boldsymbol{x}_1 - \boldsymbol{U}_1\bar{\boldsymbol{u}}_1 \equiv \boldsymbol{U}_1\boldsymbol{x}_1 + \boldsymbol{u}_1, \tag{27}$$

$$\boldsymbol{U}\boldsymbol{x}_{k+1} = \boldsymbol{U}\bar{\boldsymbol{F}}\boldsymbol{x}_k + \boldsymbol{U}\bar{\boldsymbol{u}}_{k+1} \equiv -\boldsymbol{F}\boldsymbol{x}_k - \boldsymbol{u}_{k+1}, \tag{28}$$

for $k = 1, 2, \ldots$, where

$$\boldsymbol{u}_1 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_6), \quad \boldsymbol{u}_{k+1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_6).$$

At epoch k, we combine all available process equations (27) and (28 and measurement equations (24) to form the following linear model

$$\begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{y}_1 \\ \boldsymbol{0} \\ \boldsymbol{y}_2 \\ \boldsymbol{0} \\ \cdot \\ \cdot \\ \boldsymbol{y}_k \end{bmatrix} = \left[\begin{array}{cccc|c} \boldsymbol{U}_1 & & & & \boldsymbol{0} \\ \boldsymbol{A}_1 & & & & \boldsymbol{B} \\ \boldsymbol{F} & \boldsymbol{U} & & & \boldsymbol{0} \\ & \boldsymbol{A}_2 & & & \boldsymbol{B} \\ & \boldsymbol{F} & \cdot & & \boldsymbol{0} \\ & & \cdot & & \cdot \\ & & \cdot & \boldsymbol{U} & \cdot \\ & & & \boldsymbol{A}_k & \boldsymbol{B} \end{array}\right] \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \cdot \\ \boldsymbol{x}_k \\ \boldsymbol{z} \end{bmatrix} + \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{v}_1 \\ \boldsymbol{u}_2 \\ \boldsymbol{v}_2 \\ \cdot \\ \cdot \\ \boldsymbol{u}_k \\ \boldsymbol{v}_k \end{bmatrix} \tag{29}$$

or

$$\boldsymbol{y}^{(k)} = \boldsymbol{A}^{(k)}\boldsymbol{x}^{(k)} + \boldsymbol{v}^{(k)}, \quad \boldsymbol{v}^{(k)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{30}$$

Notice that since $\boldsymbol{U}_1$ and $\boldsymbol{U}$ are nonsingular, $\boldsymbol{A}^{(k)}$ has full column rank.

For numerical stability, we use the QR factorization method to find the LS estimate of $\boldsymbol{x}^{(k)}$ in (30) (see [6, Sec 5.3]). Suppose we find an orthogonal matrix $\boldsymbol{Q}$ such that

$$(\boldsymbol{Q}^{(k)})^T\boldsymbol{A}^{(k)} = \begin{bmatrix} (\boldsymbol{Q}_1^{(k)})^T \\ (\boldsymbol{Q}_2^{(k)})^T \end{bmatrix} \boldsymbol{A}^{(k)} = \begin{bmatrix} \boldsymbol{R}^{(k)} \\ \boldsymbol{0} \end{bmatrix}, \tag{31}$$

where $\boldsymbol{R}^{(k)}$ is nonsingular upper triangular. Then the linear model (30) can be transformed to

$$\begin{bmatrix} (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{y}^{(k)} \\ (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{y}^{(k)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^{(k)} \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{x}^{(k)} + \begin{bmatrix} (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{v}^{(k)} \\ (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{v}^{(k)} \end{bmatrix},$$

where transformed noise vector still has zero mean and unit covariance matrix, i.e., $\begin{bmatrix} (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{v}^{(k)} \\ (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{v}^{(k)} \end{bmatrix} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Then the LS estimate $\hat{\boldsymbol{x}}^{(k)}$ satisfies

$$\boldsymbol{R}^{(k)}\hat{\boldsymbol{x}}^{(k)} = (\boldsymbol{Q}_1^{(k)})^T\boldsymbol{y}^{(k)},$$

and its error covariance matrix is

$$\text{cov}\{\boldsymbol{x}^{(k)} - \hat{\boldsymbol{x}}^{(k)}\} = [(\boldsymbol{R}^{(k)})^T\boldsymbol{R}^{(k)}]^{-1}. \tag{32}$$

Now we give details for computing (31). We will follow [4] and [16] to make full use of special structure of $\boldsymbol{A}^{(k)}$. First we set

$$\hat{\boldsymbol{R}}_1 \equiv \boldsymbol{U}_1, \quad \hat{\boldsymbol{C}}_1 \equiv \boldsymbol{0}.$$

For each j we find an orthogonal matrix $\bar{\boldsymbol{Q}}_j^T$ to zero $\boldsymbol{A}_j$ as follows:

$$\bar{\boldsymbol{Q}}_j^T \begin{bmatrix} \hat{\boldsymbol{R}}_j & \hat{\boldsymbol{C}}_j \\ \boldsymbol{A}_j & \boldsymbol{B} \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{R}}_j & \bar{\boldsymbol{C}}_j \\ \boldsymbol{0} & \boldsymbol{M}_j \end{bmatrix}, \tag{33}$$

where $\bar{\boldsymbol{R}}_j$ is nonsingular upper triangular and $\bar{\boldsymbol{Q}}_j$ is orthogonal. Considering the structures of $\hat{\boldsymbol{R}}_j \in \mathcal{R}^{6\times 6}$, $\boldsymbol{A}_j \in \mathcal{R}^{(2m-2)\times 6}$ and $\boldsymbol{B} \in \mathcal{R}^{(2m-2)\times(m-1)}$ in (33), we use Givens rotations to eliminate the non-zero elements of $\boldsymbol{A}_j$ columnwise from the left to the right and for each column from the bottom to the top. For each non-zero element in the ith row ($2 \le i \le 2m-2$) of $\boldsymbol{A}_j$, we combine it with the $(i-1)$th element in the same column to construct the rotation. For each non-zero element in the first row of $\boldsymbol{A}_j$, we combine it with the corresponding diagonal entry of $\hat{\boldsymbol{R}}_j$ in the same column to construct the rotation. We give part of the process for $m = 6$ schematically in Figure 1, where the letters indicate the order of the zero elements becoming non-zeros in $\boldsymbol{A}_k$ by the Givens rotations and these elements will be eliminated later; the numbers in squares indicate the order of these zero elements in $\boldsymbol{B}$ becoming non-zeros after the Givens rotations. Note that the last $m-1$ rows of $\boldsymbol{B}$ are all zeros before applying the rotations, thus after the rotations, the last $m-4$ rows of $\boldsymbol{M}_k$ are zeros. It is easy to know a total of $6m$ rotations are needed.

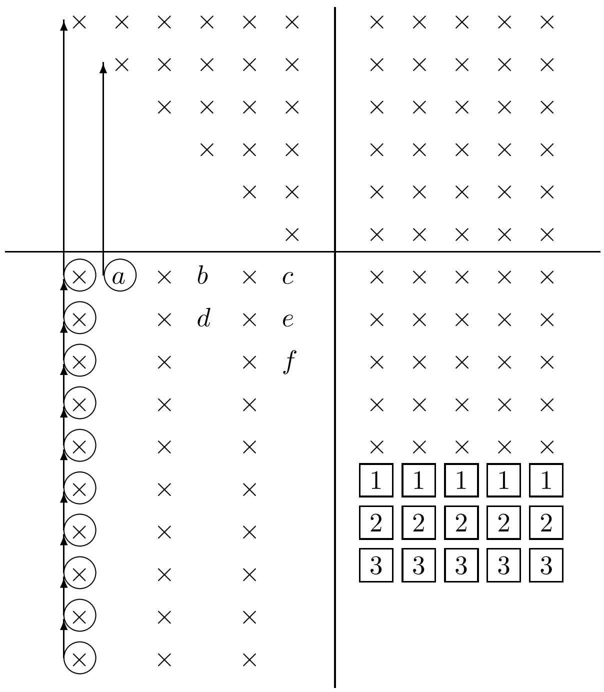

Figure 1: Scheme of transformation (33)

After the transformation (33), another orthogonal matrix $\tilde{\boldsymbol{Q}}_j$ is designed such that

$$\tilde{\boldsymbol{Q}}_j^T \begin{bmatrix} \bar{\boldsymbol{R}}_j & \boldsymbol{0} & \bar{\boldsymbol{C}}_j \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{M}_j \\ \boldsymbol{F} & \boldsymbol{U} & \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_j & \boldsymbol{R}_{j,j+1} & \boldsymbol{C}_j \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{M}_j \\ \boldsymbol{0} & \hat{\boldsymbol{R}}_{j+1} & \hat{\boldsymbol{C}}_{j+1} \end{bmatrix}, \quad (34)$$

where $\boldsymbol{R}_j$ and $\hat{\boldsymbol{R}}_{j+1}$ are both upper triangular. We also use Givens rotations to zero $\boldsymbol{F} \in \mathcal{R}^{6\times 6}$ by taking advantage of the upper triangular structure of $\bar{\boldsymbol{R}}_j \in \mathcal{R}^{6\times 6}$ and $\boldsymbol{U} \in \mathcal{R}^{6\times 6}$. The transformation on the first two block columns of (34) is schematically shown in Figure 2, where the number in a circle indicates the element is eliminated in the ith rotation, while the number in a square indicates this element is generated by the ith rotation.

When we apply the above transformations to $\boldsymbol{A}^{(k)}$, we also apply them to $\boldsymbol{y}^{(k)}$ in (29) or (30). The transformations are denoted by

$$\bar{\boldsymbol{Q}}_j^T \begin{bmatrix} \hat{\boldsymbol{c}}_j \\ \boldsymbol{y}_j \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{c}}_j \\ \boldsymbol{d}_j \end{bmatrix}, \quad \tilde{\boldsymbol{Q}}_j^T \begin{bmatrix} \bar{\boldsymbol{c}}_j \\ \boldsymbol{d}_j \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}_j \\ \boldsymbol{d}_j \\ \hat{\boldsymbol{c}}_{j+1} \end{bmatrix}, \quad (35)$$

where $\hat{\boldsymbol{c}}_1 \equiv \boldsymbol{c}$.

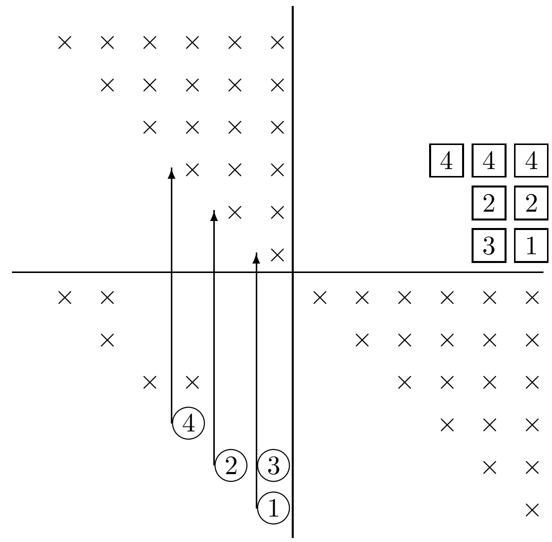

Figure 2: Scheme of transformation (34)

Therefore, finally we have transformed (29) to

$$\begin{bmatrix} \boldsymbol{c}_1 \\ \boldsymbol{d}_1 \\ \boldsymbol{c}_2 \\ \boldsymbol{d}_2 \\ \cdot \\ \cdot \\ \bar{\boldsymbol{c}}_k \\ \boldsymbol{d}_k \end{bmatrix} \approx \left[\begin{array}{ccccc|c} \boldsymbol{R}_1 & \boldsymbol{R}_{12} & & & & \boldsymbol{C}_1 \\ & \boldsymbol{0} & & & & \boldsymbol{M}_1 \\ & \boldsymbol{R}_2 & \boldsymbol{R}_{23} & & & \boldsymbol{C}_2 \\ & & \boldsymbol{0} & & & \boldsymbol{M}_2 \\ & & & \cdot & & \cdot \\ & & & \cdot & & \cdot \\ & & & & \bar{\boldsymbol{R}}_k & \bar{\boldsymbol{C}}_k \\ & & & & \boldsymbol{0} & \boldsymbol{M}_k \end{array}\right] \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \cdot \\ \boldsymbol{x}_k \\ \boldsymbol{z} \end{bmatrix} \quad (36)$$

where to simplify the presentation, we have omitted the transformed noise vector, which still has zero mean and a unit covariance matrix since we used orthogonal transformations, and so we used "≈" instead of "=". Later, we will continue to use this notation when we transform the model by orthogonal transformations.

Reordering the equations in (36), we obtain two submodels:

$$\begin{bmatrix} \boldsymbol{c}_1 \\ \cdot \\ \boldsymbol{c}_{k-1} \\ \bar{\boldsymbol{c}}_k \end{bmatrix} \approx \left[\begin{array}{cccc|c} \boldsymbol{R}_1 & \boldsymbol{R}_{12} & & & \boldsymbol{C}_1 \\ & \cdot & \cdot & & \cdot \\ & & \boldsymbol{R}_{k-1} & \boldsymbol{R}_{k-1,k} & \boldsymbol{C}_{k-1} \\ & & & \bar{\boldsymbol{R}}_k & \bar{\boldsymbol{C}}_k \end{array}\right] \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \cdot \\ \boldsymbol{x}_k \\ \boldsymbol{z} \end{bmatrix}. \quad (37)$$

$$\begin{bmatrix} \boldsymbol{d}_1 \\ \cdot \\ \boldsymbol{d}_{k-1} \\ \boldsymbol{d}_k \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{M}_1 \\ \cdot \\ \boldsymbol{M}_{k-1} \\ \boldsymbol{M}_k \end{bmatrix} \boldsymbol{z}. \quad (38)$$

Obviously we need to first estimate $\boldsymbol{z}$ by (38) and then estimate $\boldsymbol{x}_k, \ldots, \boldsymbol{x}_1$ by (37).

The estimate of the DDIA vector $\boldsymbol{z}$ can be computed in a recursive fashion. Suppose at epoch $k-1$ we have computed the following orthogonal transformations

$$\boldsymbol{T}_{k-1}^{T}\begin{bmatrix}\boldsymbol{M}_1\\ \boldsymbol{M}_2\\ \cdot\\ \boldsymbol{M}_{k-1}\end{bmatrix}=\begin{bmatrix}\boldsymbol{S}_{k-1}\\ \boldsymbol{0}\end{bmatrix},\quad \boldsymbol{T}_{k-1}^{T}\begin{bmatrix}\boldsymbol{d}_1\\ \boldsymbol{d}_2\\ \cdot\\ \boldsymbol{d}_{k-1}\end{bmatrix}=\begin{bmatrix}\boldsymbol{f}_{k-1}\\ \bar{\boldsymbol{f}}_{k-1}\end{bmatrix},$$

where $\boldsymbol{T}_{k-1}^{T}$ is orthogonal, and $\boldsymbol{S}_{k-1}$ is nonsingular upper triangular. Then at epoch k, with the newly obtained $\boldsymbol{M}_k$ and $\boldsymbol{d}_k$ (cf. (33) and (35)) we use Householder transformations to compute

$$\tilde{\boldsymbol{T}}_k^{T}\begin{bmatrix}\boldsymbol{S}_{k-1}\\ \boldsymbol{M}_k\end{bmatrix}=\begin{bmatrix}\boldsymbol{S}_k\\ \boldsymbol{0}\end{bmatrix},\ \tilde{\boldsymbol{T}}_k^{T}\begin{bmatrix}\boldsymbol{f}_{k-1}\\ \boldsymbol{d}_k\end{bmatrix}=\begin{bmatrix}\boldsymbol{f}_k\\ \hat{\boldsymbol{f}}_k\end{bmatrix},\ \bar{\boldsymbol{f}}_k\equiv\begin{bmatrix}\bar{\boldsymbol{f}}_{k-1}\\ \hat{\boldsymbol{f}}_k\end{bmatrix}. \tag{39}$$

In the above transformations, for computational efficiency, we have to make full use of the structures of $\boldsymbol{S}_{k-1}$ (which is upper triangular) and $\boldsymbol{M}_k$ (whose last $m-4$ rows are zero). Note that we never form or store $\boldsymbol{T}_{k-1}$ and $\tilde{\boldsymbol{T}}_k$. Then (38) is transformed to

$$\begin{bmatrix}\boldsymbol{f}_k\\ \bar{\boldsymbol{f}}_k\end{bmatrix}\approx\begin{bmatrix}\boldsymbol{S}_k\\ \boldsymbol{0}\end{bmatrix}\boldsymbol{z}. \tag{40}$$

So we can compute the estimate $\boldsymbol{z}_{k|k}$ of $\boldsymbol{z}$ by solving the $(m-1)\times(m-1)$ upper triangular system

$$\boldsymbol{S}_k\boldsymbol{z}_{k|k}=\boldsymbol{f}_k. \tag{41}$$

Then from (37) we can obtain the estimate $\boldsymbol{x}_{k|k}$ of $\boldsymbol{x}_k$ and estimates $\boldsymbol{x}_{j|k}$ of $\boldsymbol{x}_j$ for $j=k-1,k-2,\ldots,1$ by solving triangular linear systems:

$$\bar{\boldsymbol{R}}_k\boldsymbol{x}_{k|k}=\bar{\boldsymbol{c}}_k-\bar{\boldsymbol{C}}_k\boldsymbol{z}_{k|k} \tag{42}$$

$$\boldsymbol{R}_j\boldsymbol{x}_{j|k}=\boldsymbol{c}_j-\boldsymbol{R}_{j,j+1}\boldsymbol{x}_{j+1|k}-\boldsymbol{C}_j\boldsymbol{z}_{k|k}. \tag{43}$$

In other words, at epoch k, we can obtain not only the estimate of the current position, which is called the filtered estimate, but also the estimates of the previous positions, which are called the smoothed estimates. In real-time applications, smoothed estimates are usually not required, then at epoch k, only $\bar{\boldsymbol{R}}_k$, $\bar{\boldsymbol{C}}_k$, $\bar{\boldsymbol{c}}_k$, $\boldsymbol{S}_k$ and $\boldsymbol{f}_k$ should be temporarily stored for position estimation at next epoch.

Before we finish this section, we would like to mention that we need to get the predicated estimate $\boldsymbol{x}_{k|k-1}$ of $\boldsymbol{x}_k$ before we use the measurement equation (24) to obtain the filtered estimate $\boldsymbol{x}_{k|k}$. After we obtain the filtered estimate $\boldsymbol{x}_{k-1|k-1}$ of $\boldsymbol{x}_{k-1}$, from the process equation (23) (with k replaced by $k-1$), we can immediately obtain

$$\boldsymbol{x}_{k|k-1}=\bar{\boldsymbol{F}}\boldsymbol{x}_{k-1|k-1}.$$

The first, third, and fifth elements of $\boldsymbol{x}_{k|k-1}$ just form the predicated estimate $\boldsymbol{b}_{k|k-1}$ of the baseline vector $\boldsymbol{b}_k$. This $\boldsymbol{b}_{k|k-1}$ is used to compute an approximation to $\boldsymbol{E}_k$ to deal with the nonlinearity problem, see the discussion given in Section 2.

3.2 Computing Error Covariance Matrices

In order to have some idea about the errors of the estimates, we need to compute the corresponding error covariance matrices $\mathrm{cov}\{\boldsymbol{x}_{j|k}-\boldsymbol{x}_j\}$ for $j=1,2,\ldots,k$ at epoch k.

We first consider $j=k$. Combine the kth block equation of (37) and the top part of (40) to give

$$\begin{bmatrix}\bar{\boldsymbol{c}}_k\\ \boldsymbol{f}_k\end{bmatrix}\approx\begin{bmatrix}\bar{\boldsymbol{R}}_k & \bar{\boldsymbol{C}}_k\\ \boldsymbol{0} & \boldsymbol{S}_k\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_k\\ \boldsymbol{z}\end{bmatrix}. \tag{44}$$

Note that (44) determines the estimate $\boldsymbol{z}_{k|k}$ of $\boldsymbol{z}$ and the estimate $\boldsymbol{x}_{k|k}$ of $\boldsymbol{x}_k$. In order to obtain $\mathrm{cov}\{\boldsymbol{x}_{k|k}-\boldsymbol{x}_k\}$, we decouple $\boldsymbol{x}_k$ and $\boldsymbol{z}$ in (44) by multiplying an orthogonal $\boldsymbol{Z}_{k|k}^{T}$ to the coefficient matrix from the left to zero the $\bar{\boldsymbol{C}}_k$ block (see [3, 4]):

$$\boldsymbol{Z}_{k|k}^{T}\begin{bmatrix}\bar{\boldsymbol{R}}_k & \bar{\boldsymbol{C}}_k\\ \boldsymbol{0} & \boldsymbol{S}_k\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}_{k|k} & \boldsymbol{0}\\ \boldsymbol{N}_{k|k} & \boldsymbol{S}_{k|k}\end{bmatrix}, \tag{45}$$

where Givens rotations are used to take advantage of the upper triangular structures of $\bar{\boldsymbol{R}}_k$ and $\boldsymbol{S}_k$ and produce upper triangular $\boldsymbol{R}_{k|k}$. We zero $\bar{\boldsymbol{C}}_k$ columnwise from the left to the right, and for each column we start from the bottom to the top. Only one element of $\bar{\boldsymbol{C}}_k$ and one corresponding diagonal entry of $\boldsymbol{S}_k$ are used to construct one rotation. Since $\bar{\boldsymbol{C}}_k$ is $6\times(m-1)$, a total of $6(m-1)$ rotations are needed. Thus after applying $\boldsymbol{Z}_{k|k}^{T}$ to (44) from the left, we obtain

$$\begin{bmatrix}\boldsymbol{c}_{k|k}\\ \boldsymbol{f}_{k|k}\end{bmatrix}\approx\begin{bmatrix}\boldsymbol{R}_{k|k} & \boldsymbol{0}\\ \boldsymbol{N}_{k|k} & \boldsymbol{S}_{k|k}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_k\\ \boldsymbol{z}\end{bmatrix}. \tag{46}$$

Then obviously $\boldsymbol{x}_{k|k}$, which satisfies $\boldsymbol{R}_{k|k}\boldsymbol{x}_{k|k}=\boldsymbol{c}_{k|k}$, has the error covariance matrix:

$$\mathrm{cov}\{\boldsymbol{x}_{k|k}-\boldsymbol{x}_k\}=(\boldsymbol{R}_{k|k}^{T}\boldsymbol{R}_{k|k})^{-1}.$$

For each $j<k$, we can recursively compute the covariance matrix $\mathrm{cov}\{\boldsymbol{x}_{j|k}-\boldsymbol{x}_j\}$. Suppose the $(j+1)$th (block) equation in (37) was transformed to the following equation when we computed $\mathrm{cov}\{\boldsymbol{x}_{j+1|k}-\boldsymbol{x}_{j+1}\}$.

$$\boldsymbol{c}_{j+1|k}\approx\boldsymbol{R}_{j+1|k}\boldsymbol{x}_{j+1}, \tag{47}$$

where $\boldsymbol{R}_{j+1|k}$ is nonsingular upper triangular. To compute the covariance matrix $\mathrm{cov}\{\boldsymbol{x}_{j|k}-\boldsymbol{x}_j\}$, we stack the jth block equation in (37), equation (47), and the top part of (40) together to give:

$$\begin{bmatrix}\boldsymbol{c}_j\\ \boldsymbol{c}_{j+1|k}\\ \boldsymbol{f}_k\end{bmatrix}\approx\begin{bmatrix}\boldsymbol{R}_j & \boldsymbol{R}_{j,j+1} & \boldsymbol{C}_j\\ \boldsymbol{0} & \boldsymbol{R}_{j+1|k} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{S}_k\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_j\\ \boldsymbol{x}_{j+1}\\ \boldsymbol{z}\end{bmatrix}. \tag{48}$$

Note that equation (48) determines the estimates $\boldsymbol{z}_{k|k}$, $\boldsymbol{x}_{j+1|k}$ and $\boldsymbol{x}_{j|k}$. We just use the same method as we did for the transformation (45). We first use $\boldsymbol{C}_j \in \mathcal{R}^{6\times(m-1)}$ and $\boldsymbol{S}_k$ to zero $\boldsymbol{C}_j$, and then use the transformed $\boldsymbol{R}_{j,j+1} \in \mathcal{R}^{6\times 6}$ and $\boldsymbol{R}_{j+1|k}$ to zero the former by the Givens rotations. We need a total of $6(m+5)$ rotations. Thus (48) is transformed to

$$\begin{bmatrix} \boldsymbol{c}_{j|k} \\ \bar{\boldsymbol{c}}_{j+1|k} \\ \boldsymbol{f}_{j|k} \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{R}_{j|k} & \boldsymbol{0} & \boldsymbol{0} \\ \bar{\boldsymbol{R}}_{(j+1,j)|k} & \bar{\boldsymbol{R}}_{j+1|k} & \bar{\boldsymbol{C}}_{j+1|k} \\ \boldsymbol{N}_{j|k} & \boldsymbol{0} & \boldsymbol{S}_{j|k} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_j \\ \boldsymbol{x}_{j+1} \\ \boldsymbol{z} \end{bmatrix} \tag{49}$$

From the first block equation in (49) we obtain

$$\mathrm{cov}\{\boldsymbol{x}_{j|k} - \boldsymbol{x}_j\} = (\boldsymbol{R}_{j|k}^T \boldsymbol{R}_{j|k})^{-1}.$$

3.3 Fixing the Integer Ambiguities

In Section 3.1, we have computed the real-valued estimate $\boldsymbol{z}_{k|k}$ of the double difference integer ambiguity (DDIA) vector $\boldsymbol{z}$ by (41). However, the elements of $\boldsymbol{z}$ are integers. If we only use its real-valued estimate in (42) and (43), we may not be able to obtain high accurate estimates of the positions quickly. To fix the obtained real-valued estimate of $\boldsymbol{z}$ to an integer vector, we can use the original LAMBDA method [18] or the modified LAMBDA method [5], which is more computationally efficient.

To use either of these two methods at epoch k, we need two pieces of information. One is the real-valued vector $\boldsymbol{z}_{k|k}$, and the other is the error covariance matrix $\boldsymbol{\Sigma}_k$ of $\boldsymbol{z}_{k|k}$. From the top part of (40), we obtain

$$\boldsymbol{\Sigma}_k = \mathrm{cov}\{\boldsymbol{z}_{k|k} - \boldsymbol{z}\} = (\boldsymbol{S}_k^T \boldsymbol{S}_k)^{-1}. \tag{50}$$

Once $\boldsymbol{z}_{k|k}$ is fixed as a vector of integers and is validated, we can obtain the corresponding position-velocity estimates (43). Starting from epoch $k+1$, we will not need to estimate $\boldsymbol{z}$ any more. Therefore, to compute the estimate of $\boldsymbol{x}_i$ for any $i > k$, we directly use the integer-valued $\boldsymbol{z}_{k|k}$ as the estimate of $\boldsymbol{z}$ at that epoch. In other words, we solve the following upper triangular systems (cf. (42) and (43)):

$$\bar{\boldsymbol{R}}_i \boldsymbol{x}_{i|i} = \bar{\boldsymbol{c}}_i - \bar{\boldsymbol{C}}_i \boldsymbol{z}_{k|k}, \qquad i > k \tag{51}$$

$$\boldsymbol{R}_j \boldsymbol{x}_{j|i} = \boldsymbol{c}_j - \boldsymbol{R}_{j,j+1}\boldsymbol{x}_{j+1|i} - \boldsymbol{C}_j \boldsymbol{z}_{k|k}, \tag{52}$$

for $j = i-1, \ldots, 1$ and $i > k$.

3.4 Handling the Change of Ambiguity Vector

So far, we have assumed that $\boldsymbol{z}$ has a fixed dimension. In GPS-based positioning, this means that we have the same number of satellites in the whole observation span. However in practice, this is not always true. When we have a long observation span, the dimension of the DDIA vector $\boldsymbol{z}$ may change due to satellite rising and setting. Furthermore, when the signal from a satellite is missing for some reason at some epoch, the dimension of $\boldsymbol{z}$ changes (this can be regarded as satellite setting). While later the signal from this satellite is re-found, the dimension of $\boldsymbol{z}$ changes again (this can be regarded as satellite rising). We also have assumed that the value of each element of $\boldsymbol{z}$ is constant, but it changes when there is a *cycle slip* in the corresponding signal (see, e.g., [7, Sec 9.1.2]). Detection and repair of cycle slips is an important topic in GPS positioning. We can incorporate a method for cycle slip detection into our positioning algorithm. We will not use a method to repair cycle slips since it may not be easy, therefore when a cycle slip is detected between two epochs, we just assume there were satellite setting and rising between these two epochs, although physically the setting and rising satellites are identical. We will handle this problem by following [3, 4] where only measurement equations were used for positioning.

The major complicated task is to update the estimate of the DDIA vector $\boldsymbol{z}$ from one epoch to the next one. Once $\boldsymbol{z}$ has been estimated at an epoch, the position estimate at this epoch can easily be obtained, Therefore, the problem is to find a way to achieve the equivalents of the transformations (33), (34), (35) and (39). Since the DDIA vector may be different for different epochs, we will use $\boldsymbol{z}_k$ instead of $\boldsymbol{z}$ and $\boldsymbol{B}_k$ instead of $\boldsymbol{B}$ in the measurement equation (22) at epoch k.

Note from (16) that every element of the DDIA vector $\boldsymbol{z}$ has the form $N^n - N^i$, where n corresponds to a non-reference satellite and i is the reference satellite. It is important to be aware that we will use the same reference satellite for every element in every DDIA vector at a given epoch j and this reference satellite must be visible at that epoch j, so if it sets between epochs $k-1$ and k, for some $k > j$, then (see *Case 2* later) we can choose to use a different reference satellite at epoch k. When we talk about a DDIA vector at an epoch, a non-reference satellite means any satellite which is not the reference satellite at that epoch. If a satellite sets and rises again, it will be considered as a new satellite.

In the following we list different DDIA vectors we will use:

- $\tilde{\boldsymbol{z}}_k$: (whose elements correspond to) all the non-reference satellites that are visible for at least one epoch from epoch 1 to epoch k;
- $\tilde{\boldsymbol{z}}_k^d$: all the non-reference satellites that go down between epoch 1 and epoch k;

- $\boldsymbol{z}_k$: all the non-reference satellites that are visible at epoch k;
- $\boldsymbol{z}_k^r$: all the non-reference satellites that are visible at epoch $k-1$ and remain at epoch k;
- $\boldsymbol{z}_k^u$: all the non-reference satellites that come up between epoch $k-1$ and epoch k;
- $\boldsymbol{z}_k^d$: all the non-reference satellites that go down between epoch $k-1$ and epoch k.

Note that when a satellite rises, the dimension of $\tilde{\boldsymbol{z}}_k$ increases by one, but a setting satellite leaves the dimension unchanged. In our constant satellite case, $\tilde{\boldsymbol{z}}_k$ was just $\boldsymbol{z}$ in (16). The following are the relationships between these DDIA vectors which will be used later:

$$\tilde{\boldsymbol{z}}_k^d = \begin{bmatrix} \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_k^d \end{bmatrix}, \ \boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}, \ \tilde{\boldsymbol{z}}_k = \begin{bmatrix} \tilde{\boldsymbol{z}}_k^d \\ \boldsymbol{z}_k \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}. \tag{53}$$

Note that $\boldsymbol{z}_{k-1}$ is a rearrangement of the elements of $\boldsymbol{z}_k^r$ and $\boldsymbol{z}_k^d$, so we can find a permutation matrix $\boldsymbol{\Pi}_k = [\boldsymbol{\Pi}_k^{(1)}, \boldsymbol{\Pi}_k^{(2)}]$ such that

$$\boldsymbol{\Pi}_k^T \boldsymbol{z}_{k-1} = \begin{bmatrix} (\boldsymbol{\Pi}_k^{(1)})^T \boldsymbol{z}_{k-1} \\ (\boldsymbol{\Pi}_k^{(2)})^T \boldsymbol{z}_{k-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \end{bmatrix}. \tag{54}$$

Assume that at the end of epoch $k-1$ we have obtained the equivalent of the top part of (40) for epoch $k-1$

$$\tilde{\boldsymbol{f}}_{k-1} \approx \tilde{\boldsymbol{S}}_{k-1} \tilde{\boldsymbol{z}}_{k-1}, \tag{55}$$

where $\tilde{\boldsymbol{S}}_{k-1}$ is nonsingular upper triangular. We can partition $\tilde{\boldsymbol{z}}_{k-1}$ as

$$\tilde{\boldsymbol{z}}_{k-1} = \begin{bmatrix} \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_{k-1} \end{bmatrix}.$$

Then with compatible partitioning, we can rewrite (55) as

$$\begin{bmatrix} \boldsymbol{f}_{k-1}^{(1)} \\ \boldsymbol{f}_{k-1} \end{bmatrix} \approx \begin{bmatrix} \tilde{\boldsymbol{S}}_{k-1}^{(1)} & \tilde{\boldsymbol{S}}_{k-1}^{(2)} \\ \boldsymbol{0} & \boldsymbol{S}_{k-1} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_{k-1} \end{bmatrix} \tag{56}$$

where both $\tilde{\boldsymbol{S}}_{k-1}^{(1)}$ and $\boldsymbol{S}_{k-1}$ are nonsingular upper triangular. Notice that if no satellites rise or set from epoch 1 to epoch $k-1$, then the top part of (56) will disappear and the bottom part is just the top part of (40) with k replaced with $k-1$. In the following we will combine (56) with the relevant equations at epoch k which provide new information about the DDIA vectors in order to obtain the estimates of the DDIA vectors. We consider two cases separately.

Case 1: The reference satellite at epoch $k-1$ remains at epoch k. We still use it as the reference satellite at epoch k. Suppose at epoch $k-1$, we obtained the equivalent of the bottom part of (37):

$$\bar{\boldsymbol{c}}_{k-1} \approx \bar{\boldsymbol{R}}_{k-1} \boldsymbol{x}_{k-1} + \bar{\tilde{\boldsymbol{C}}}_{k-1} \tilde{\boldsymbol{z}}_{k-1}. \tag{57}$$

At epoch k, we first combine (57) with the process equation (28) to give

$$\begin{bmatrix} \bar{\boldsymbol{c}}_{k-1} \\ \boldsymbol{0} \end{bmatrix} \approx \begin{bmatrix} \bar{\boldsymbol{R}}_{k-1} & \boldsymbol{0} & \bar{\tilde{\boldsymbol{C}}}_{k-1} \\ \boldsymbol{F} & \boldsymbol{U} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k-1} \\ \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_{k-1} \end{bmatrix}. \tag{58}$$

As in (34), we apply the Givens rotations to zero $\boldsymbol{F}$ in the coefficient matrix in (58). Then (58) is transformed to (cf. (34) and (35))

$$\begin{bmatrix} \boldsymbol{c}_{k-1} \\ \hat{\boldsymbol{c}}_k \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{R}_{k-1} & \boldsymbol{R}_{k-1,k} & \tilde{\boldsymbol{C}}_{k-1} \\ \boldsymbol{0} & \hat{\boldsymbol{R}}_k & \hat{\tilde{\boldsymbol{C}}}_k \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k-1} \\ \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_{k-1} \end{bmatrix}. \tag{59}$$

Note that we have decoupled $\boldsymbol{x}_{k-1}$ from $\boldsymbol{x}_k$ and $\tilde{\boldsymbol{z}}_{k-1}$. Once the estimates of $\boldsymbol{x}_k$ and $\tilde{\boldsymbol{z}}_{k-1}$ are obtained, the estimate of $\boldsymbol{x}_{k-1}$ can be obtained immediately.

Now we consider to use the measurement equation (24) at epoch k, which, for the new situation, is written as

$$\boldsymbol{y}_k \approx \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{z}_k, \tag{60}$$

where a subscript k has been added to $\boldsymbol{B}$ and $\boldsymbol{z}$ and the noise vector has been omitted. Our task is to combine the bottom part of (59) and (60), both of which involve $\boldsymbol{x}_k$ and DDIA vectors, to obtain the equivalent of the bottom two block equations in (36). In order to do this, we have to rewrite them.

With compatible partitioning (cf. (53)), $\hat{\tilde{\boldsymbol{C}}}_k \tilde{\boldsymbol{z}}_{k-1}$ in (59) can be written as

$$\hat{\tilde{\boldsymbol{C}}}_k \tilde{\boldsymbol{z}}_{k-1} = \begin{bmatrix} \hat{\tilde{\boldsymbol{C}}}_k^{(1)} & \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_{k-1} \end{bmatrix}.$$

For $\hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{z}_{k-1}$, use (54) to write

$$\begin{aligned} \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{z}_{k-1} &= \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{\Pi}_k \boldsymbol{\Pi}_k^T \boldsymbol{z}_{k-1} \\ &= \begin{bmatrix} \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{\Pi}_k^{(1)} & \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{\Pi}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \end{bmatrix}. \end{aligned} \tag{61}$$

Therefore, we can rewrite the bottom part of (59) as

$$\hat{\boldsymbol{c}}_k \approx \begin{bmatrix} \hat{\boldsymbol{R}}_k & \hat{\tilde{\boldsymbol{C}}}_k^{(1)} & \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{\Pi}_k^{(1)} & \hat{\tilde{\boldsymbol{C}}}_k^{(2)} \boldsymbol{\Pi}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \end{bmatrix}. \tag{62}$$

Now we rewrite (60). We partition $\boldsymbol{B}_k$ compatibly with the partitioning of $\boldsymbol{z}_k$ in (53), so that we have

$$\boldsymbol{B}_k \boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{B}_k^{(1)} & \boldsymbol{B}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}.$$

Therefore (60) can be rewritten as

$$\boldsymbol{y}_k \approx \begin{bmatrix} \boldsymbol{A}_k & \boldsymbol{B}_k^{(1)} & \boldsymbol{B}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}. \tag{63}$$

Combining (62) with (63), we have

$$\begin{bmatrix} \hat{\boldsymbol{c}}_k \\ \boldsymbol{y}_k \end{bmatrix} \approx \begin{bmatrix} \hat{\boldsymbol{R}}_k & \hat{\tilde{\boldsymbol{C}}}_k^{(1)} & \hat{\tilde{\boldsymbol{C}}}_k^{(2)}\boldsymbol{\Pi}_k^{(1)} & \hat{\tilde{\boldsymbol{C}}}_k^{(2)}\boldsymbol{\Pi}_k^{(2)} & \boldsymbol{0} \\ \boldsymbol{A}_k & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}_k^{(1)} & \boldsymbol{B}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}. \tag{64}$$

In order to decouple $\boldsymbol{x}_k$ from the DDIA vectors, we multiply both sides of (64) by a sequence of Givens rotations from the left to zero $\boldsymbol{A}_k$, giving (cf. (33), (35)):

$$\begin{bmatrix} \bar{\boldsymbol{c}}_k \\ \boldsymbol{d}_k \end{bmatrix} \approx \begin{bmatrix} \bar{\boldsymbol{R}}_k & \bar{\tilde{\boldsymbol{C}}}_k^{(1)} & \bar{\tilde{\boldsymbol{C}}}_k^{(2)} & \bar{\tilde{\boldsymbol{C}}}_k^{(3)} & \bar{\tilde{\boldsymbol{C}}}_k^{(4)} \\ \boldsymbol{0} & \tilde{\boldsymbol{M}}_k^{(1)} & \tilde{\boldsymbol{M}}_k^{(2)} & \tilde{\boldsymbol{M}}_k^{(3)} & \tilde{\boldsymbol{M}}_k^{(4)} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}, \tag{65}$$

Note that the bottom block equation of (65) involves only DDIA vectors.

To compute the estimate of $\tilde{\boldsymbol{z}}_k$ or give the equivalent of (55) at epoch k, we want to combine (56) with the bottom part of (65). In order to do that, we use (54) and write $\tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{z}_{k-1}$ and $\boldsymbol{S}_{k-1}\boldsymbol{z}_{k-1}$ in (56) as follows:

$$\tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{z}_{k-1} = \begin{bmatrix} \tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{\Pi}_k^{(1)} & \tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{\Pi}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \end{bmatrix}, \tag{66}$$

$$\boldsymbol{S}_{k-1}\boldsymbol{z}_{k-1} = \begin{bmatrix} \boldsymbol{S}_{k-1}\boldsymbol{\Pi}_k^{(1)} & \boldsymbol{S}_{k-1}\boldsymbol{\Pi}_k^{(2)} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \end{bmatrix}. \tag{67}$$

Then we stack (56) on the bottom part of (65), and use (66) and (67) to give:

$$\begin{bmatrix} \boldsymbol{f}_{k-1}^{(1)} \\ \boldsymbol{f}_{k-1} \\ \boldsymbol{d}_k \end{bmatrix} \approx \begin{bmatrix} \tilde{\boldsymbol{S}}_{k-1}^{(1)} & \tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{\Pi}_k^{(1)} & \tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{\Pi}_k^{(2)} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}_{k-1}\boldsymbol{\Pi}_k^{(1)} & \boldsymbol{S}_{k-1}\boldsymbol{\Pi}_k^{(2)} & \boldsymbol{0} \\ \tilde{\boldsymbol{M}}_k^{(1)} & \tilde{\boldsymbol{M}}_k^{(2)} & \tilde{\boldsymbol{M}}_k^{(3)} & \tilde{\boldsymbol{M}}_k^{(4)} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_{k-1}^d \\ \boldsymbol{z}_k^d \\ \boldsymbol{z}_k^r \\ \boldsymbol{z}_k^u \end{bmatrix}. \tag{68}$$

Then we perform the following orthogonal transformations to (68):

$$\tilde{\boldsymbol{T}}_k^T \left[\begin{array}{cc|cc} \tilde{\boldsymbol{S}}_{k-1}^{(1)} & \tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{\Pi}_k^{(1)} & \tilde{\boldsymbol{S}}_{k-1}^{(2)}\boldsymbol{\Pi}_k^{(2)} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}_{k-1}\boldsymbol{\Pi}_k^{(1)} & \boldsymbol{S}_{k-1}\boldsymbol{\Pi}_k^{(2)} & \boldsymbol{0} \\ \tilde{\boldsymbol{M}}_k^{(1)} & \tilde{\boldsymbol{M}}_k^{(2)} & \tilde{\boldsymbol{M}}_k^{(3)} & \tilde{\boldsymbol{M}}_k^{(4)} \end{array}\right]$$

$$= \left[\begin{array}{c|c} \tilde{\boldsymbol{S}}_k^{(1)} & \tilde{\boldsymbol{S}}_k^{(2)} \\ \hline \boldsymbol{0} & \boldsymbol{S}_k \\ \hline\hline \boldsymbol{0} & \boldsymbol{0} \end{array}\right] \equiv \begin{bmatrix} \tilde{\boldsymbol{S}}_k \\ \hline\hline \boldsymbol{0} \end{bmatrix},$$

$$\tilde{\boldsymbol{T}}_k^T \begin{bmatrix} \boldsymbol{f}_{k-1}^{(1)} \\ \boldsymbol{f}_{k-1} \\ \boldsymbol{d}_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_k^{(1)} \\ \hline \boldsymbol{f}_k \\ \hline\hline \hat{\boldsymbol{f}}_k \end{bmatrix} \equiv \begin{bmatrix} \tilde{\boldsymbol{f}}_k \\ \hline\hline \hat{\boldsymbol{f}}_k \end{bmatrix},$$

where both $\tilde{\boldsymbol{S}}_k^{(1)}$ and $\boldsymbol{S}_k$ are nonsingular upper triangular. This has completed the update and provided the equivalents of (55) and its expanded form (56) for epoch k:

$$\tilde{\boldsymbol{f}}_k \approx \tilde{\boldsymbol{S}}_k \tilde{\boldsymbol{z}}_k, \tag{69}$$

or

$$\begin{bmatrix} \boldsymbol{f}_k^{(1)} \\ \boldsymbol{f}_k \end{bmatrix} \approx \begin{bmatrix} \tilde{\boldsymbol{S}}_k^{(1)} & \tilde{\boldsymbol{S}}_k^{(2)} \\ 0 & \boldsymbol{S}_k \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_k^d \\ \boldsymbol{z}_k \end{bmatrix}, \tag{70}$$

where we have used the relationships among difference DDIVs given in (53). We now can compute the LS estimates $\boldsymbol{z}_{k|k}$ of $\boldsymbol{z}_k$ and $\tilde{\boldsymbol{z}}_{k|k}^d$ of $\tilde{\boldsymbol{z}}_k^d$ by solving

$$\boldsymbol{S}_k \boldsymbol{z}_{k|k} = \boldsymbol{f}_k, \ \tilde{\boldsymbol{S}}_k^{(1)} \tilde{\boldsymbol{z}}_{k|k}^d = \boldsymbol{f}_k^{(1)} - \tilde{\boldsymbol{S}}_k^{(2)} \boldsymbol{z}_{k|k}, \ \tilde{\boldsymbol{z}}_{k|k} = \begin{bmatrix} \tilde{\boldsymbol{z}}_{k|k}^d \\ \boldsymbol{z}_{k|k} \end{bmatrix}.$$

Note that if no satellites rise or set between epochs $k-1$ and k, $\boldsymbol{z}_k^d$ and $\boldsymbol{z}_k^u$ will have no elements, therefore in (54) we can take $\boldsymbol{\Pi}_k = \boldsymbol{\Pi}_k^{(2)} = \boldsymbol{I}$, giving $\boldsymbol{z}_{k-1} = \boldsymbol{z}_k^r = \boldsymbol{z}_k$.

With the estimated DDIA vectors, we can compute the filtered estimates and smoothed estimates of position-velocity vectors. Using (53), from the top part of (65) we can obtain $\boldsymbol{x}_{k|k}$ by solving

$$\bar{\boldsymbol{R}}_k \boldsymbol{x}_{k|k} = \bar{\boldsymbol{c}}_k - \bar{\tilde{\boldsymbol{C}}}_k \tilde{\boldsymbol{z}}_{k|k}, \tag{71}$$

where

$$\bar{\tilde{\boldsymbol{C}}}_k \equiv [\bar{\tilde{\boldsymbol{C}}}_k^{(1)}, \bar{\tilde{\boldsymbol{C}}}_k^{(2)}, \bar{\tilde{\boldsymbol{C}}}_k^{(3)}, \bar{\tilde{\boldsymbol{C}}}_k^{(4)}]. \tag{72}$$

Suppose we have computed the estimate $\boldsymbol{x}_{j+1|k}$ of $\boldsymbol{x}_{j+1}$, we will show how to compute the smoothed estimates of $\boldsymbol{x}_{j|k}$ of $\boldsymbol{x}_j$. From (59), we see the equivalent of its first block equation for each epoch $j < k$ can be written as

$$\boldsymbol{c}_j \approx \boldsymbol{R}_j \boldsymbol{x}_j + \boldsymbol{R}_{j,j+1} \boldsymbol{x}_{j+1} + \tilde{\boldsymbol{C}}_j \tilde{\boldsymbol{z}}_j. \tag{73}$$

Since the elements of $\tilde{\boldsymbol{z}}_j$ are part of $\tilde{\boldsymbol{z}}_k$, there exists a matrix $\boldsymbol{P}_{j|k}$ which is a permutation matrix with possible zero columns added, such that

$$\tilde{\boldsymbol{z}}_j = \boldsymbol{P}_{j|k} \tilde{\boldsymbol{z}}_k,$$

so the corresponding estimate $\tilde{\boldsymbol{z}}_{j|k}$ of $\tilde{\boldsymbol{z}}_j$ satisfy

$$\tilde{\boldsymbol{z}}_{j|k} = \boldsymbol{P}_{j|k} \tilde{\boldsymbol{z}}_{k|k}.$$

Then we can compute the smoothed estimate $\boldsymbol{x}_{j|k}$ by solving

$$\boldsymbol{R}_j \boldsymbol{x}_{j|k} = \boldsymbol{c}_j - \boldsymbol{R}_{j,j+1} \boldsymbol{x}_{j+1|k} - \tilde{\boldsymbol{C}}_j \tilde{\boldsymbol{z}}_{j|k}. \tag{74}$$

Finally we would like to discuss how to compute the error covariance matrices $\text{cov}\{\boldsymbol{x}_{j|k} - \boldsymbol{x}_j\}$ for $j = 1, 2, \ldots, k$. First we consider $j = k$. We combine the top part of (65) with (69) and use (53) to give

$$\begin{bmatrix} \bar{\boldsymbol{c}}_k \\ \tilde{\boldsymbol{f}}_k \end{bmatrix} \approx \begin{bmatrix} \bar{\boldsymbol{R}}_k & \bar{\tilde{\boldsymbol{C}}}_k \\ \boldsymbol{0} & \tilde{\boldsymbol{S}}_k \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_k \end{bmatrix}, \tag{75}$$

where $\bar{\tilde{\boldsymbol{C}}}_k$ is defined by (72). So we can use the same approach of estimating the filtered error covariance matrix $\text{cov}\{\boldsymbol{x}_{k|k} - \boldsymbol{x}_k\}$ presented in Section 3.2 to (75). We find an orthogonal $\boldsymbol{Z}_{k|k}$ to transform (75) to (cf. (45), (46)):

$$\begin{bmatrix} \boldsymbol{c}_{k|k} \\ \tilde{\boldsymbol{f}}_{k|k} \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{R}_{k|k} & \boldsymbol{0} \\ \boldsymbol{N}_{k|k} & \tilde{\boldsymbol{S}}_{k|k} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \tilde{\boldsymbol{z}}_k \end{bmatrix}, \tag{76}$$

where $\boldsymbol{R}_{k|k}$ is nonsingular upper triangular. Therefore we have

$$\text{cov}\{\boldsymbol{x}_{k|k} - \boldsymbol{x}_k\} = (\boldsymbol{R}_{k|k}^T \boldsymbol{R}_{k|k})^{-1}.$$

Now we consider how to compute $\text{cov}\{\boldsymbol{x}_{j|k} - \boldsymbol{x}_j\}$ in the order of $j = k-1, k-2, \ldots, 1$. Suppose in the process of finding $\text{cov}\{\boldsymbol{x}_{j+1|k} - \boldsymbol{x}\}$, we obtained the following equation (cf. the top block equation in (76)):

$$\boldsymbol{d}_{j+1|k} = \boldsymbol{R}_{j+1|k} \boldsymbol{x}_{j+1}. \tag{77}$$

where $\boldsymbol{R}_{j+1|k}$ is nonsingular upper triangular. We combine (73), (77) and (69) together to give:

$$\begin{bmatrix} \boldsymbol{c}_j \\ \boldsymbol{c}_{j+1|k} \\ \tilde{\boldsymbol{f}}_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_j & \boldsymbol{R}_{j,j+1} & \tilde{\boldsymbol{C}}_j \boldsymbol{P}_{j|k} \\ \boldsymbol{0} & \boldsymbol{R}_{j+1|k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \tilde{\boldsymbol{S}}_k \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_j \\ \boldsymbol{x}_{j+1} \\ \tilde{\boldsymbol{z}}_k \end{bmatrix}. \tag{78}$$

The above equation has the same structure as (48) in, so we can adopt the same approach to computing $\text{cov}\{\boldsymbol{x}_{j|k} - \boldsymbol{x}_j\}$ given there. The Givens rotations can be found to transform (78) to

$$\begin{bmatrix} \boldsymbol{c}_{j|k} \\ \bar{\boldsymbol{c}}_{j+1|k} \\ \tilde{\boldsymbol{f}}_{j|k} \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{R}_{j|k} & \boldsymbol{0} & \boldsymbol{0} \\ \bar{\boldsymbol{R}}_{(j+1,j)|k} & \bar{\boldsymbol{R}}_{j+1|k} & \bar{\tilde{\boldsymbol{C}}}_{j+1|k} \\ \boldsymbol{N}_{j|k} & \boldsymbol{0} & \tilde{\boldsymbol{S}}_{j|k} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_j \\ \boldsymbol{x}_{j+1} \\ \tilde{\boldsymbol{z}}_k \end{bmatrix},$$

where $\boldsymbol{R}_{j|k}$ is nonsingular upper triangular. Thus it follows that

$$\text{cov}\{\boldsymbol{x}_{j|k} - \boldsymbol{x}_j\} = (\boldsymbol{R}_{j|k}^T \boldsymbol{R}_{j|k})^{-1},$$

Case 2: The reference satellite (satellite 1, say) of epoch $k-1$ goes down between epochs $k-1$ and k. We can choose any satellite visible at epoch $k-1$ and remaining at epoch k, without loss of generality assuming that is satellite 2, to be the reference satellite at epoch k. Suppose at epoch k, the new measurement equation is

$$\boldsymbol{y}_k \approx \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{z}_k, \tag{79}$$

where $\boldsymbol{z}_k$ is the DDIA vector with satellite 2 as the reference satellite. At the end of epoch $k-1$ we have (55), where $\tilde{\boldsymbol{z}}_{k-1} \in \mathcal{R}^{m-1}$ say,

$$\tilde{\boldsymbol{z}}_{k-1} \equiv [N^2 - N^1, N^3 - N^1, \ldots, N^m - N^1]^T.$$

Define the corresponding vector $\bar{\boldsymbol{z}}_{k-1}$ with satellite 2 as the reference satellite, along with the matrix $\boldsymbol{K}$

$$\bar{\boldsymbol{z}}_{k-1} \equiv [N^1 - N^2, N^3 - N^2, \ldots, N^m - N^2]^T,$$
$$\boldsymbol{K} \equiv \begin{bmatrix} -1 & \boldsymbol{0} \\ -\boldsymbol{e} & \boldsymbol{I}_{m-2} \end{bmatrix}.$$

It is easy to verify that b

$$\boldsymbol{K}\boldsymbol{K} = \boldsymbol{I}, \quad \bar{\boldsymbol{z}}_{k-1} = \boldsymbol{K} \tilde{\boldsymbol{z}}_{k-1}. \tag{80}$$

This implies that we can easily transform a DDIA vector with one satellite as the reference satellite to another DDIA vector with another satellite as the reference satellite. Define $\bar{\boldsymbol{S}}_{k-1} \equiv \tilde{\boldsymbol{S}}_{k-1} \boldsymbol{K}$, then we can rewrite (55) as

$$\tilde{\boldsymbol{f}}_{k-1} \approx \tilde{\boldsymbol{S}}_{k-1} \tilde{\boldsymbol{z}}_{k-1} = \tilde{\boldsymbol{S}}_{k-1} \boldsymbol{K}\boldsymbol{K} \tilde{\boldsymbol{z}}_{k-1} = \bar{\boldsymbol{S}}_{k-1} \bar{\boldsymbol{z}}_{k-1}.$$

If we find an orthogonal matrix from the left to triangularize $\bar{\boldsymbol{S}}_{k-1}$, we get essentially the same situation as in *Case 1*, therefore we then can follow the steps in *Case 1* to handle it.

4 Summary

A recursive LS approach was presented for short baseline kinematic GPS positioning by using combined carrier phase and code measurements and a process model. Unlike the approaches using standard Kalman filter in the literature, our approach gives advantages in numerical stability and efficiency by using orthogonal transformations and taking full advantage of the structure of the problem.

We gave full computational details for computing position-velocity estimates (including both filtered and smoothed position-vector estimates) as well as the corresponding error covariance matrices. We also handled the computation for possible satellite setting and rising. If a process model in a specific application is not the same as the one used in this paper, our approach can be modified without difficulty. Our approach can also be modified easily to handle dual frequency signals.

References

[1] P. Axelrad and R.G. Brown. GPS navigation algorithms. In: B.W. Parkinson, J.J. Spilker, P. Axelrad, and P. Enge (ed), *Global Positioning System: Theory and Applications*, Vol 1. American Institute of Aeronautics and Astronautics, Inc, Washington, DC, 1996. pp. 409–433.

[2] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan. *Estimation with Applications To Tracking and Navigation*. John Wiley & Sons, Inc., New York, 2001.

[3] X.-W. Chang and C.C. Paige. An orthogonal transformation algorithm for GPS Positioning. *SIAM J. Sci. Comp.*, 24:1710-1732, 2003.

[4] X-W. Chang, C. C. Paige, and L. Yin. Code and carrier phase based short baseline GPS positioning: Computational aspects. *GPS Solutions*, 7(4):230 – 240, 2004.

[5] X.-W. Chang, X. Yang, and T. Zhou. MLAMBDA: A modified LAMBDA method for integer ambiguity determination. The ION 2005 Annual Meeting, June 27-29, Cambridge, Massachusetts.

[6] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Maryland, 3rd edition, 1996.

[7] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *GPS Theory and Practice*. Springer, New York, 5th edition, 1997.

[8] P. Y. C. Hwang and R. G. Brown. GPS navigation; combining pseudo-range with continuous carrier phase using a Kalman filter, navigation. *Journal of The Institute of Navigation*, 37:181–196, 1990.

[9] X.-X. Jin. Algorithm for carrier-adjusted DGPS positioning and some numerical results. *Journal of Geodesy*, 71:411–422, 1997.

[10] T. Kailath, A.H. Sayed, and B. Hassibi. Linear Estimation. Prentice Hall, Upper Saddle River, NJ, 2000.

[11] D. Kim and R. B. Langley. A reliable approach for ambiguity resolution in real-time long-baseline kinematic GPS applications. In *Proceedings of ION GPS 2000*, Salt Lake City, Utah, September 19-22, pp. 1081-1091.

[12] A. Leick. *GPS Satellite Surveying*. John Wiley & Sons, Inc., New York, 1995.

[13] G. Liu and G. Lachapelle. Ionosphere weighted GPS cycle ambiguity resolution. In *Proceedings of National Technical Meeting, The Institute of Navigation* San Diego, CA, 28-30 Janary, 2002, pp. 889-899.

[14] M. C. Macabiau. *Analysis of the Feasibility of Using GPS Carrier Phase Ambiguity Resolution Techniques for Precision Approaches*. PhD thesis, L'institut national polytechnique de Toulouse, September 1997.

[15] P. Misra and P. Enge. *Global Positioning System, Signals, Measurements and Performance*. Ganga-Jamuna Press, Lincoln, Massachusetts, 2001.

[16] C. C. Paige and M. A. Saunders. Least squares estimation of discrete linear dynamic system using orthogonal transformations. *SIAM Journal on Numerical Analysis*, 14:180–193, 1977.

[17] X. Qin, S. Gourevitch, and M. Kuhl. Very precise GPS: development status and results. In *Proceedings of ION GPS 1992*, September 16-18, 1992, Albuquerque, NM, pp. 615-624.

[18] P. J. Teunissen and A. Kleusberg. *GPS for Geodesy*. Springer, Berlin, 3nd edition, 1998.

[19] C. C. J. M. Tiberius. *Recursive Data Processing for Kinematic GPS Surveying*. PhD thesis, Department of Mathematical Geodesy and Positioning, Delft University of Technology, 1998.