Assessment of New Tracking Architectures for Future GNSS Receivers in Harsh Environments

Mathilde DUFOUR, ALTRAN Technology, mathilde.dufour@altran.com

Dr Christophe OUZEAU, ALTRAN Technology, christophe.ouzeau@altran.com

BIOGRAPHIES

Mathilde DUFOUR is a GNSS engineer consultant at ALTRAN Technologies. In order to fulfill her ENAC engineer and her TBS manager degrees, she worked on future GNSS receivers’ architecture design. In particular, she focused her research activity on frequency and carrier phase tracking strategy.

Dr Christophe OUZEAU graduated in 2005 with a master in astronomy at the Observatory of Paris. He works on the GNSS domain for 12 years. He holds a Ph.D. in signal processing from INPT/ENAC, managed by TéSA and financed by DGAC/DTI and contributed to the EUROCAE WG 62 on multi-constellation receivers’ architecture with technical inputs. In particular, his researches focused on degraded modes due to feared events occurrence. He was involved in many GNSS projects in particular concerning GNSS test bench for land-based and space applications, receivers’ architecture design and preliminary studies for new satellites payload design and new applications. For ALTRAN, he created an own-founded GNSS project called OBSYNA (Navigation Systems Observatory) in 2013 which main goals are to find new GNSS applications and to build future GNSS receivers’ architecture. In 2015 and 2016, he patented 3 solutions to improve vibrations and oscillations detection at GNSS receivers’ level and for seismic detection applications using GNSS signals. Today, he is GNSS Specialist at ALTRAN and leads several teams for own-founded research activities and customer projects like ESA, CNES, ESSP, Thales Alenia Space or Airbus Defence and Space France.

ABSTRACT

In GNSS receivers, conventional scalar tracking algorithms allows to synchronize incoming signals with their locally generated replicas in terms of code delay, Doppler frequency and carrier phase. In that way, basic DLL, FLL and PLL are used. The signals tracking depends upon their waveforms that impact the corresponding tracking channel setting, in particular the choice of tracking loops characteristics: correlators, discriminators, filter orders and bandwidths.

As a consequence, to improve the receiver robustness, it is of interest to study new signals processing strategy. Amongst the potential receivers’ architectures, it is fed by the authors of the GNSS domain as an important task to investigate vector Kalman-based tracking architectures.

In addition, vector tracking has many advantages over scalar tracking loops. The most commonly cited advantage is an increased immunity to interferences [4] [8]. The minimum carrier to noise power density ratio at which the receiver can operate is lowered by processing the signals in aggregate instead of separately.

In this paper, Kalman-based FPLL (Frequency and Phase Locked Loop) tracking loop models are discussed and their performances are assessed. The algorithms described are used to estimate jointly carrier phase and frequency tracking errors by one Kalman filter. Thus, the state vector is composed of tracking errors and the loops’ discriminators outputs are considered as measurements.

Two models are presented. The main difference between the two models concerns the definition of the filter gain that can be fixed or calculated according to measurements. Computing a variable gain requires tracking loop bandwidths knowledge, dynamics and measurement noise settings. This allows the filter to be robust against the receiver dynamics. In comparison, a fixed gain depends only on tracking loop bandwidths but its computation must be as optimal as possible because it weights all measurements in the same way.

This paper details models implementation: gain computation, choices of measurement matrix and how to estimate measurements. Inspired by two existing models (fixed gain and variable gain) that estimate carrier phase tracking errors and frequencies, the design of new models implies an adaptation of measurement matrix and gain. In various conditions (occurrence of feared events, filters setting…), the filters’ estimations are discussed. Finally, vector tracking is discussed while including Kalman
filter-based algorithms. Furthermore, to complete the analysis, feared event detection is discussed.

1 INTRODUCTION

With the development of new and existing navigation systems (GPS, GALILEO, GLONASS, QZSS ...), the variety of radio navigation signals increases greatly involving 8 frequency bands and about 20 different signals.

Additionally, the number of new satellites with various waveforms has increased, multiplying architecture possibilities for multi-frequency receivers.

Moreover, the architecture of receivers has to be able to adapt to many growing GNSS markets: land-based, mass market, transports, civil aviation, military and space applications... Even better, receivers which are flexible for several applications can be targeted. Therefore, software receivers which can be updated are the future of localization and navigation.

Thus, the architecture of GNSS receivers must be defined as a scalable system, taking into account the wide range of existing and expected signals and depending on the application case.

When each useful signal is acquired, basic DLL, PLL and FLL enable the synchronization between the incoming signal and its local replica in terms of code delay, Doppler frequency and carrier phase. Those tracking loops depend on tracking loops characteristics such as the filter order and the bandwidth. As FLL and PLL are mandatory to perform frequency and phase synchronization, a lack of robustness from the beginning of the synchronization can impact the next steps of the tracking process, and then the positioning process.

Therefore, to improve robustness, it is worth studying new architectures of the tracking process.

For the sake of robustness, the tracking architecture presented in this paper relies on a FPLL (Frequency and Phase Locked Loop) using a Kalman filter coupled with a state machine. A Kalman filter is a known way to replace a conventional tracking filter to provide better tracking error estimation.

To complete the architecture proposed, the idea is to extend the FPLL to all channels and to vectorise it, in order to take benefit of channels correlations.

By using a state machine to select settings according to the receiver environment, the number of Kalman filters used is limited and the receiver architecture is adaptive.

This paper highlights the following main contributions:
- Selection and setting of two FPLL algorithms using Kalman filter: jointly estimation of Doppler frequency and carrier phase errors by a Kalman filter,
- Identification of mostly encountered feared events and mapping with applications (not exhaustive),
- Discussion on robustness of proposed algorithms against feared events,
- Presentation of improvements to perform the tracking process under harsh environments,
- Preliminary assessment and discussion about patented algorithms to perform signals and feared events detection in noisy environments,
- Discussion on vector tracking and detection of feared events.

The paper is organized as follows. After a review of conventional tracking architecture, its differences with a Kalman-based tracking, new algorithms with their hypotheses are presented. Then, performances and robustness of those algorithms in harsh environments are analyzed. Finally, a discussion is led on how to integrate these algorithms in the receiver architecture to provide feared events detection.

2 FOCUS ON RECEIVERS SIGNAL TRACKING ARCHITECTURE

It is necessary to identify promising GNSS receivers’ architectures for each application. The environment of the receivers’ antenna will depend upon the applications and the feared events must be mapped with each kind of application. Finally, promising receivers’ algorithms must be assessed in details to know if:
- They are compliant with performances targeted for each application,
- They are robust against feared events,
- Algorithms can be embedded in targeted HW/SW platforms.

In the following, the focus is made on the user segment and in particular on the receivers’ architecture and its capability to process GNSS signals in various environments (and in particular in harsh environments). This part is a state of the art about existing receivers’ tracking architectures: conventional tracking, Kalman-based tracking and vector tracking.
Figure 1 illustrates the signals processing for each channel, including the acquisition and tracking processes until the data demodulation.

![Diagram](image)

**Figure 1: Acquisition, tracking and demodulation steps per channel**

### 2.1 Conventional tracking architecture

Tracking process enables to synchronize the incoming signal and its locally generated replica, and monitors the progress of code delay, Doppler frequency and carrier phase [1].

As for the acquisition process, tracking can process only one channel at a time, but several tracking processes are run in parallel. Some channels could be in acquisition mode\(^1\) whereas others would already be in tracking mode\(^2\). The high level conventional tracking loop structure is described in the Figure 2.

![Diagram](image)

**Figure 2: Conventional tracking loop architecture**

The correlator periodically achieves the correlation product between the input signal and the locally generated replica. The replica is generated through the NCO (Numerically Controlled Oscillator).

At the correlator output (of code, phase or frequency), the signal is split into a real part (channel I) and an imaginary part (channel Q). Here, only the real signal is processed because it deals with a GPS L1 C/A signal.

The integrator used at the correlator output is a low-pass filter with integration time \( T \).

Thus, in case of a complex signal, correlators’ outputs after integration can be modelled as following:

\[
I(t) = \frac{A}{2} d(t) \sin(\pi \varepsilon_t) R_c(\varepsilon_t) \cos(\varepsilon_\varphi) + n_I(t) \quad (Eq \ 1)
\]

\[
Q(t) = \frac{A}{2} \sin(\pi \varepsilon_t) R_c(\varepsilon_t) \sin(\varepsilon_\varphi) + n_Q(t) \quad (Eq \ 2)
\]

Where

- \( A \) is the input signal amplitude,
- \( d(t) \) is the modulation of the navigation message,
- \( R_c \) is the code autocorrelation,
- \( \varepsilon_t \) is the signal Doppler frequency error,
- \( \varepsilon_\varphi \) is the signal carrier phase error,
- \( n_I(t) \) and \( n_Q(t) \) are the additive and uncorrelated white Gaussian noises.

In this paper, the focus is made on the PLL convergence, in particular, the goal is to be able to demodulate the navigation data \( d(t) \) while:

\[
I(t) = \frac{A}{2} d(t) \sin(\pi \varepsilon_t) R_c(\varepsilon_t) \cos(\varepsilon_\varphi) + n_I(t) \quad \xrightarrow{\varepsilon_\varphi \to 0} \quad I(t) = \frac{A}{2} d(t) + n_I(t) \quad (Eq \ 3)
\]

\[
Q(t) = \frac{A}{2} \sin(\pi \varepsilon_t) R_c(\varepsilon_t) \sin(\varepsilon_\varphi) + n_Q(t) \quad \xrightarrow{\varepsilon_\varphi \to 0} \quad Q(t) = n_Q(t) \quad (Eq \ 4)
\]

The discriminator is used to extract the correlation product error in order to compute synchronization errors of code, phase and frequency.

Finally, the filter enables to face side effects of signal dynamics, mainly caused by satellite antenna and receiver relative dynamics. This bandpass loop filter is then characterised by its bandwidth and order (order 1: speed; order 2: accelerations; order 3: jerks…). The number of gains of a conventional loop filter corresponds to the tracking loop order. For an order 3 PLL [2]:

\[
K_1 = \frac{60}{23} B_k T_D \quad (Eq \ 5)
\]

\[
K_2 = \frac{4}{9} K_1^2 \quad (Eq \ 6)
\]

\[
K_3 = \frac{2}{27} K_1^3 \quad (Eq \ 7)
\]

In conventional tracking loops, each operation in each channel is independent. Pseudoranges and pseudorange rates are computed separately, by channel. It’s only in the

\(^1\) Acquisition mode: a state of one channel at a given time in signal processing.

\(^2\) Tracking mode: a state of one channel at a given time. It follows the acquisition mode in signal processing.
navigation filter that those measures are combined to provide the navigation solution. This type of tracking architecture is widespread in receivers. Nevertheless, conventional tracking presents some drawbacks. Although it gives correct navigation solutions in normal conditions (i.e. without considering feared events), it is well-known that conventional tracking process is less performant in harsh environments. Loss of positioning solutions or failures in code/carrier tracking is common. Indeed, weak signals or significant signal power drops impact the correlation process. Then, the navigation process is affected by lack of accuracy of estimated pseudoranges [3][4]. Moreover, it allows to provide as many pseudoranges as available channels. [5].

As it is explained in the next part of this paper, several parameters, such as discriminator type, filter order and loop bandwidth, have an impact on the signal tracking behaviour.

A FLL is more robust than a PLL since it does not control the exact value of the phase of the local carrier, however this loop will not allow the navigation message demodulation. Indeed, it is necessary to minimize the carrier phase error (convergence towards zero) to be able to demodulate navigation messages on a data channel (considering code synchronization is made).

The FPLL can also be used since it combines the advantages of both PLL and FLL. This loop filter uses two discriminator inputs for frequency and carrier phase errors. This loop is more robust than a conventional PLL.

2.2 Kalman-based tracking architecture

The Kalman filter based tracking loop consists in replacing the conventional loop filter by a Kalman loop filter. This architecture is represented in the Figure 3.

![Figure 3: Kalman-based tracking loop architecture](image)

One of the Kalman-based tracking advantages is the weighting of the quality of predictions compared to the calculation, before updating each tracking occurrence.

In theory, the Kalman-based algorithm provides the optimal gain. Thus, this tracking architecture is more robust than the conventional one:

- To weak signals because the Kalman filter adapts the filter bandwidth to the noise level [6]. The Kalman gain provides an adaptive bandwidth filtering. In particular, this avoids using long integration time and enables improving the sensitivity of tracking. Indeed, lock detectors allow declaring the loss of lock of the local replica to the input signal, at a low C/N0.

- To scintillations because it optimize automatically the loop filter minimizing the phase mean square error. In that fact, a Kalman-based tracking provides better performances than a conventional PLL [7].

The main problem of a Kalman-based tracking loop relates to the settings of noise matrix used to compute the gain. Indeed, performances depend on the use of a correct state space dynamic model and on the monitoring of the actual measurement noise [7].

Generally, the Kalman filter gain is variable, it adapts to measurements and particularly to the C/N0 level. But the Kalman filter gain can be fixed and the weighting is done by a command helping the filter updating according to measurements.

The Table 1 contains two Kalman-based tracking algorithms implemented at the PLL and FLL discriminators outputs that are considered as measurements. These algorithms estimate only phase errors by using estimated Doppler frequencies. The first algorithm presents a constant Kalman filter gain and uses an updating command whereas the second has a variable Kalman filter gain.
Table 1: Description of two Kalman-based tracking algorithms for carrier phase errors estimation

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>“KF fixed gain”[3]</th>
<th>“KF variable gain”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter gain</td>
<td>Fixed gain [3x1] Depending on ( B_{PLL} ) ( B_{PLL} ) and noise matrix</td>
<td>Variable gain [2x2]</td>
</tr>
<tr>
<td>State vector</td>
<td>( X_k = \begin{bmatrix} \Delta \phi_k \ \alpha_k \end{bmatrix} )</td>
<td>( X_k = \begin{bmatrix} \Delta \phi_k \ \alpha_k \end{bmatrix} )</td>
</tr>
<tr>
<td>Measurement vector</td>
<td>( Y_k = \begin{bmatrix} \varphi \alpha_f \end{bmatrix} )</td>
<td>( Y_k = \begin{bmatrix} \varphi \alpha_f \end{bmatrix} )</td>
</tr>
<tr>
<td>State transition matrix</td>
<td>( \phi = \begin{bmatrix} 1 &amp; Te &amp; Te^2/2 \ 0 &amp; 1 &amp; Te \ 0 &amp; 0 &amp; 1 \end{bmatrix} )</td>
<td>( \phi = \begin{bmatrix} 1 &amp; Te \ 0 &amp; 1 \end{bmatrix} )</td>
</tr>
<tr>
<td>Measurement matrix</td>
<td>( H = \begin{bmatrix} 1 &amp; Te/2 &amp; Te^2/2 \end{bmatrix} )</td>
<td>( H = \begin{bmatrix} 1 &amp; 0 \end{bmatrix} )</td>
</tr>
<tr>
<td>Command</td>
<td>( \Delta u_k = Te \cdot NCO_{PLL} )</td>
<td>/</td>
</tr>
<tr>
<td>A priori state estimation</td>
<td>( \hat{x}_{k</td>
<td>k-1} = \phi \hat{x}_{k-1</td>
</tr>
<tr>
<td>Measurement estimation</td>
<td>( \hat{y}<em>k = H \hat{x}</em>{k</td>
<td>k-1} - \frac{1}{2} \Delta u_k )</td>
</tr>
<tr>
<td>Innovation</td>
<td>( Y_k = \hat{y}_k )</td>
<td>( Y_k = \hat{y}_k )</td>
</tr>
<tr>
<td>State estimation</td>
<td>( \hat{x}_{k</td>
<td>k} = \hat{x}_{k</td>
</tr>
</tbody>
</table>

2.3 FLL-aided PLL tracking architecture

The FLL-aided PLL tracking architecture is a way to optimize the transition from the FLL loop to the PLL loop in case of dynamics. Indeed, due to the Doppler effect, frequency evolves quickly and the transition from the FLL loop to the PLL loop has to follow these frequency evolutions so that the carrier generator and the frequency synthesizer jointly work. In case of a FLL-aided PLL tracking, the FLL loop absorbs frequency tracking errors that allows a faster carrier phase synchronization. This architecture is presented on the Figure 4.

2.4 Vector tracking architecture

A vector tracking architecture is defined as the processing of all channels with one loop for both tracking and navigation [8]. This architecture combines tracking and navigation processes. A Kalman filter is used at navigation level to estimate both tracking errors and position. It deals with an extended Kalman filter.

Three kinds of vector tracking are mainly used: Vector Frequency Lock Loop (VFLL), Vector Delay Lock Loop (VDLL) and Vector Delay and Frequency Lock Loop (VDFLL).

In a Vector Frequency Lock Loop (VFLL), the frequency discriminator output and range-rate measurements are used. The velocity navigation states are used in an Extended Kalman Filter. The Figure 5 illustrates the architecture of VFLL tracking.
whereas the VFLL tracks the dynamics from all satellites signals.

In case of a Vector Delay Lock Loop, the architecture is the same with the use of code discriminators outputs and pseudo range measurements instead of frequency discriminators outputs and range-rate measurements. A VDLL tracking ensures robustness to low C/N0 ratios and interferences [4].

A VDFLL architecture is a combination of both previous architectures VFLL and VDLL to improve robustness in harsh environments. More particularly, this architecture provides better performances, in terms of tracking efficiency and position estimation, than a conventional tracking in case of dynamics and few visible satellites [4].

Therefore, more generally, vector tracking presents various advantages compared to conventional tracking [9]. Indeed, by correlation of channels, it allows to estimate N unknowns (3 positions and a clock bias) thanks to N measurements while conventional tracking estimates only N pseudoranges with N satellites’ signals [5]. Then, vector tracking provides better performances. It mainly enables a better reliability to face interferences and better detection ability. Vector tracking is more robust than conventional one in following points:

- The minimum signal-to-noise ratio under which the receiver can operate correctly is lower when signals are processed together rather than separately [8].
- In addition to this reliability to face interferences, vectorisation enables the reliability of the tracking when facing cycle slips and receiver’s dynamics.
- Cross-correlation allows to bridge signal outages of some satellites and to reacquire them immediately when they reappear [8].
- Vector tracking is also used to reduce multipath effects. Indeed, thanks to channels comparison, NLOS (Non Line Of Sight) signals are detected [4].

However, vectorising the tracking has some defects. The main issue is that an error on one channel spreads over all other channels. A problem on one channel can actually impact the others and lead to the receiver instability or to the desynchronization with all the satellites.

Furthermore, vectorised tracking is complex to implement [9] and has a longer launching process than a scalar one.

Finally, VFLL, VDLL and VDFLL do not allow to track carrier phase. These architectures require a conventional PLL in parallel. A solution can be to use a serial PLL connected to VDFLL architecture so that the PLL of each channel benefits from the frequency estimation by the VDFLL structure. Here, an anomaly in PLL tracking does not affect the VDFLL tracking [3].

This study is done to analyse another solution to improve tracking robustness by taking into account carrier phase estimation in order to accelerate navigation message demodulation. Contrary to vector tracking, it does not consider navigation process but just tracking process.

3 PROPOSED FREQUENCY AND CARRIER PHASE (FPLL) TRACKING ALGORITHMS USING KALMAN FILTER

3.1 Introduction

It deals with the implementation of a FPLL tracking using a Kalman filter on only one channel. It means carrier phase and frequency tracking errors are jointly estimated. Here, it is not a vector tracking architecture that is proposed but only the parallelization of PLL and FLL with one loop on one channel to begin.

The use of a Kalman filter rather than a conventional loop filter improves the estimation. Estimating simultaneously both PLL and FLL tracking errors makes the tracking process more robust to feared events, and, more precisely, to the receiver dynamics. A FPLL tracking loop allows detecting cycle slips when dynamics is important. This also reduces the convergence latency period when it comes to re-synchronize the signal in case of carrier phase or frequency stall.

It is important to note that this architecture does not modify the DLL tracking loop operation which occurs separately with a conventional loop filter.

3.2 Proposed algorithms description

Here, two implemented algorithms will be described. It deals with FPLL tracking using a Kalman filter. The novelty of these proposed algorithms is to estimate jointly carrier phase and frequency errors:

- “Kalman_based_FPLL_fixed_gain” derived from a paper written by Psiaki [10]. This tracking algorithm is a third order tracking and no more a second order. Thus, Kalman gain and command computations have to be reviewed compared to the reference paper (Table 1) to take into account PLL and FLL characteristics.
- “Kalman_based_FPLL_variable_gain” derived from a paper written by O’Driscoll and Lachapelle [11]. This article describes a third order tracking to estimate jointly code and carrier
phase errors. Here, it deals with the estimation of carrier phase and frequency errors. That is why the measurement noise matrix depends on PLL and FLL characteristics and no more on DLL and PLL ones.

Contrary to reference literature [10] [11], this paper exposes results of robustness to feared events of the proposed FPLL tracking algorithms in part 4.

To estimate simultaneously carrier phase tracking error and frequency tracking error, both proposed algorithms have the same state vector. It is composed of $\Delta \phi_k$ carrier phase tracking error, $\Delta f_k$ frequency tracking error and $\alpha_k$ Doppler shift rate at time $k$.

\[ X_k = \begin{bmatrix} \Delta \phi_k \\ \Delta f_k \\ \alpha_k \end{bmatrix} \quad \text{(Eq 8)} \]

Discriminators outputs are considered as measurements in both algorithms and the vector is expressed as:

\[ Y_k = \begin{bmatrix} \varepsilon \phi_k \\ \varepsilon f_k \end{bmatrix} \quad \text{(Eq 9)} \]

Where $\varepsilon \phi_k$ is the carrier phase discriminator output and $\varepsilon f_k$ the frequency discriminator output.

The main difference between these two models concerns Kalman gain computation: fixed or variable.

Computation of a fixed Kalman gain is based on the pole placement technique described on Appendix 1 [10] [12]. A fixed Kalman gain is not adaptive to measurements. So, a command is required to assist the Kalman filter in updating the state vector. This command needs to be conditioned by the receiver environment.

A variable Kalman gain matches conventional Kalman gain and needs noise matrix settings.

Here, the measurement noise matrix is a diagonal matrix without additive coefficients [13]. It is considered that correlation between state variables (carrier phase, Doppler frequency and Doppler shift rate) is made thanks to the Kalman filter equations and iterations. In this paper, the measurement noise matrix follows this model using conventional PLL and FLL tracking variances [14]:

\[ R = \begin{bmatrix} B_{PLL} \cdot \frac{C}{N0} & 0 & 0 \\ 0 & \frac{1}{2 \pi \nu e^2} \cdot \frac{B_{PLL} \cdot \nu}{C} \cdot \frac{1}{N0} \end{bmatrix} \quad \text{(Eq 10)} \]

Where:
- $B_{PLL}$ is PLL bandwidth,
- $B_{FLL}$ is FLL bandwidth,
- $\nu e$ is the sampling time.

Then, dynamics noise matrix is a diagonal matrix [3x3] adjusted through trial and error principle.

\[ Q = \begin{bmatrix} \sigma_{\varepsilon_1}^2 & 0 & 0 \\ 0 & \sigma_{\varepsilon_2}^2 & 0 \\ 0 & 0 & \sigma_{\varepsilon_3}^2 \end{bmatrix} \quad \text{(Eq 11)} \]

The Table 2 summarizes settings and calculations main steps of the proposed two FPLL tracking algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>“Kalman_based_FPLL__fixed_gain”</th>
<th>“Kalman_based_FPLL__variable_gain”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter gain</td>
<td>Fixed gain [3x2]</td>
<td>Variable gain [3x2]</td>
</tr>
<tr>
<td>State vector</td>
<td>$X_k = \begin{bmatrix} \Delta \phi_k \ \Delta f_k \ \alpha_k \end{bmatrix}$</td>
<td>$X_k = \begin{bmatrix} \Delta \phi_k \ \Delta f_k \ \alpha_k \end{bmatrix}$</td>
</tr>
<tr>
<td>Measurement vector</td>
<td>$Y_k = \begin{bmatrix} \varepsilon \phi_k \ \varepsilon f_k \end{bmatrix}$</td>
<td>$Y_k = \begin{bmatrix} \varepsilon \phi_k \ \varepsilon f_k \end{bmatrix}$</td>
</tr>
<tr>
<td>State transition matrix</td>
<td>$\phi = \begin{bmatrix} 1 \ 0 \ 0 \end{bmatrix}$</td>
<td>$\phi = \begin{bmatrix} 1 \ 0 \ 0 \end{bmatrix}$</td>
</tr>
<tr>
<td>Measurement matrix</td>
<td>$H = \begin{bmatrix} 1 &amp; Te &amp; Te^2/2 \ 0 &amp; 1 &amp; Te \ 0 &amp; 0 &amp; 1 \end{bmatrix}$</td>
<td>$H = \begin{bmatrix} 1 &amp; Te/2 &amp; Te^2/6 \ 0 &amp; 1 &amp; Te/2 \ 0 &amp; 0 &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Command</td>
<td>$\Delta u_k = \frac{T e \cdot N C O_{PLL} + (Te^2/6) \varepsilon f}{Te \cdot \varepsilon f}$</td>
<td>/</td>
</tr>
<tr>
<td>A priori state estimation</td>
<td>$\hat{X}_{k</td>
<td>k-1} = \phi \ast \hat{X}_{k-1</td>
</tr>
<tr>
<td>Measurement estimation</td>
<td>$\hat{Y}<em>k = H \hat{X}</em>{k</td>
<td>k-1} - \frac{1}{2} \Delta u_k$</td>
</tr>
<tr>
<td>Innovation</td>
<td>$Y_k - \hat{Y}_k$</td>
<td>$Y_k - \hat{Y}_k$</td>
</tr>
<tr>
<td>State estimation</td>
<td>$\hat{X}_{k</td>
<td>k} = \hat{X}_{k</td>
</tr>
</tbody>
</table>

Table 2: Description of two FPLL tracking algorithms for carrier phase and Doppler frequency errors estimation.
3.3 Hypotheses

In this paper, to test robustness of proposed algorithms, only GPS L1 C/A signals with a BPSK waveform are processed. The sampling frequency at the correlators’ outputs is set to 100 Hz.

This study deals with carrier phase tracking loop PLL and frequency tracking loop FLL. Some hypotheses are taken about the discriminator type, tracking loop order, loop bandwidth and integration time.

The loop order is conditioned by signal dynamics and needs to be adapted depending on the required robustness to dynamics effects.

The choice of loop bandwidth and integration time is based on thermal noise impacts analyses [14].

\[ \sigma_{PLL}^2 = \frac{B_L}{C} \left( 1 + \frac{1}{2 T_D C N_0} \right) \]  
\[ \sigma_{FLL}^2 = \frac{2 B_L}{\pi^2 T_D^2} \frac{C}{N_0} \left( 1 + \frac{1}{T_D C N_0} \right) \]  

(Eq 12)  
(Eq 13)

The loop bandwidth is an important parameter. Indeed, a low loop bandwidth decreases the thermal noise but also slows down the capability of the filter to follow dynamics changes. A high loop bandwidth increases the thermal noise impact but the loop filter is capable of following more accurately the dynamics changes.

In Table 3, main loop hypotheses are recalled for the FLL and PLL.

<table>
<thead>
<tr>
<th>Discriminator type</th>
<th>PLL</th>
<th>FLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop order</td>
<td>ATAN</td>
<td>Diff ATAN</td>
</tr>
<tr>
<td>Loop bandwidth $B_L$</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Integration time $T_D$</td>
<td>10 Hz</td>
<td>18 Hz</td>
</tr>
</tbody>
</table>

Table 3: Hypothesis on tracking loops PLL and FLL

In the following, if the simulation results refer to “static receiver mode”, only the satellites dynamics are taken into account and the receiver antenna phase centre is considered fixed in the ECEF WGS84 reference. If the simulation results refer to dynamics, some Doppler values are tested with Doppler shift rates and jerks detailed in the following.

3.4 Performances in “static receiver mode”

In this part, the performances of proposed FPLL tracking models are presented in “static receiver mode”. In this case, since measurements have low variations, the Kalman filter needs low updating values to follow them.

According to Figure 6 and Figure 7, the Kalman filter estimates correctly frequency errors in two FPLL tracking algorithms.

With “Kalman_based_FPLL_fixed_gain” model (Figure 6), estimations of carrier phase tracking errors present lightly wider variations (red) than carrier phase measurements (blue). And, in the “Kalman_based_FPLL_variable_gain” (Figure 7), there is an offset of 0.01 rad between carrier phase tracking errors estimation (red) and carrier phase measurements (blue). Nevertheless, these gaps are very low and can be solved by adding a command to accurately adjust measurements according to Kalman filter outputs.

Figure 6: Carrier phase and frequency tracking errors (red) with respect to carrier phase and frequency measurements (blue) in static mode with “Kalman_based_FPLL_fixed_gain” model

Figure 7: Carrier phase and frequency tracking errors (red) with respect to carrier phase and frequency measurements (blue) in static mode with “Kalman_based_FPLL_variable_gain” model

---

3 Static mode: simulation hypothesis described part 3.3.
Therefore, these two FPLL tracking algorithms show fair performances in static mode to estimate jointly carrier phase and frequency tracking errors.

### 3.5 Comparison with the conventional tracking

The Table 4 summarizes tracking error standard deviations and convergence times obtained with the conventional tracking algorithm and the two proposed FPLL tracking models in “static receiver mode”.

<table>
<thead>
<tr>
<th>Model</th>
<th>Conv time</th>
<th>Phase error STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional</td>
<td>-</td>
<td>0.01 rad</td>
</tr>
<tr>
<td>KF FPLL Fixed Gain</td>
<td>0.15 sec</td>
<td>0.004 rad</td>
</tr>
<tr>
<td>KF FPLL Variable Gain</td>
<td>0.2 sec</td>
<td>0.004 rad</td>
</tr>
</tbody>
</table>

Table 4: PLL performances of conventional tracking and proposed FPLL tracking in “static receiver mode”

This Table 4 highlights a better precision in estimation of FPLL tracking than conventional tracking because it shows lower variations of carrier phase tracking errors around convergent value. So, FPLL tracking reduces tracking errors compared to conventional tracking. Nevertheless, it introduces a slight convergence time.

Another advantage of FPLL tracking is its capability to estimate jointly carrier phase and frequency tracking errors whereas conventional tracking does it sequentially.

### 3.6 Receiver environment adjustment points

Here, are proposed key points on which it is possible to play to adapt the FPLL tracking models to the application environment.

Concerning “Kalman_based_FPLL_fixed_gain” model, a command is used to update the filter. To take into account the receiver environment, this command has to be updated and expressed according to it.

For the “Kalman_based_FPLL_variable_gain” algorithm, to consider the antenna environment requires an adaptation of noise matrix configuration or, directly an automation of Kalman gain computation without noise matrix settings.

To modify the measurement noise matrix R, a solution can be to constrain more the state variance model taking into account the impact of thermal noise, phase scintillations and receiver oscillator noise [6]:

\[
\sigma_{PLL}^2 = \sigma_{\phi T}^2 + \sigma_{\phi P}^2 + \sigma_{\phi osc}^2
\]  

(Eq 14)

Where:

- \(\sigma_{\phi T}^2\) the thermal noise impact,
- \(\sigma_{\phi P}^2\) phase scintillations variance,
- \(\sigma_{\phi osc}^2\) receiver oscillator noise impact.

The Figure 8 illustrates the relation between the carrier phase error standard deviation and the PLL bandwidth depending on what environment constraints are considered.

![Figure 8: Standard deviation of the phase error as function of the loop bandwidth and with vibration, phase error, thermal noise, dynamics and all the effects. The main hypotheses taken are: a TCXO oscillator, C/N0 = 30 dBHz, 1 ms integration time.](image)

To automate the variable Kalman gain and to avoid adjusting noise matrix, it is possible to use ARMA coefficients. A patented algorithm called “Kalman/ARMA” [15] connects Kalman gain and ARMA coefficients computations bypassing the use of noise matrix. This new algorithm is detailed in part 5.2.

### 3.7 FPLL tracking algorithms complexity

A complexity study is necessary to assess the portability of the algorithms on any code hosting platform and the use on various applications. This complexity analysis of the algorithms would include the study of the mandatory settings and an approximation of the number of calculation steps.

A Kalman-based tracking is more complex to implement in terms of computation than a conventional one.

Now, fixed and variable Kalman gain computation complexities are compared in case of a FPLL tracking architecture.
On one hand, a variable Kalman gain requires lots of necessary settings to adjust noise matrix as for a known third order Kalman-based tracking.

On the other hand, for a constant Kalman gain, settings are less numerous. However, to set a third order Kalman gain needs complex calculations. Moreover, the command used to update the filter must be configured in accordance with the receiver environment and so, the applications.

<table>
<thead>
<tr>
<th>Segment level</th>
<th>Some feared events</th>
</tr>
</thead>
</table>
| Satellite and receiver levels | • Loss of signal due to a satellite problem,  
• Code-carrier incoherency satellite induced by code-carrier divergence,  
• Jump in inter-frequency hardware bias,  
• Doppler frequency instability,  
• IODE (Issue Of Data Ephemeris) anomaly,  
• Erroneous ephemeris,  
• Noisy ephemeris,  
• Corrupted navigation message,  
• Signal distortion,  
• Drift in inter-frequency hardware bias. |
| Signal in space (SiS) level | • SIS step error (incl. clock jump),  
• SIS ramp error (incl. clock drift),  
• SIS acceleration error,  
• SIS sinusoid error,  
• SIS noise error (incl. excessive phase noise on carrier). |
| Regional level | • Excessive ionospheric spatial gradient,  
• Excessive ionospheric temporal gradient,  
• Excessive ionospheric scintillation,  
• Excessive troposphere. |
| Local level | • Excessive electromagnetic interference (including intentional sources of interference such as a scrambler),  
• Excessive multipath,  
• Non-line of sight conditions. |
| Receiver hardware level | • Damaged antenna,  
• Damaged cable,  
• Lack of power supply. |

Table 5: Short complexity comparison between fixed and variable gain in case of a FPLL tracking algorithm

Therefore, according to the Table 5, in a FPLL tracking architecture, it is easier to use a variable Kalman gain once noise matrix configuration is found than a constant one.

This short complexity analyse is a start to conclude if the proposed FPLL tracking algorithms are flexible and can be embedded to any hardware platform in order to be used for various GNSS applications.

4 RESISTANCE TO FEARED EVENTS OF PROPOSED FPLL TRACKING MODELS

With a growing GNSS market and the emergence of new specific GNSS need, new environment constraints appear and receivers robustness have to be improved to ensure signal processing in harsh environments. In this part, proposed FPLL algorithms robustness to some feared events will be analysed.

4.1 Feared events classification

At space segment, ground segment and user segment levels, various feared events may impact the final services performances. The feared events may be classified in five categories as shown in the Table 6.
from the feared events at a satellite level which also lead to a loss of signal.

All the feared events will not be addressed in this paper. Nevertheless, the focus will be made on weak signals, high dynamics, multipath effects, interferences and ionospheric scintillations.

4.2 Feared events/applications mapping

The diversification of GNSS applications means the emergence of new environment constraints. Indeed, feared events are really dependent on the environment of the receiver antenna.

The purpose of this part is to make a repartition of feared events according to their probability of occurrence for each application.

The five feared events considered are these for which a robustness study will be presented later in this paper: weak signals, dynamics, multipath effects, interferences and scintillations. And, the GNSS fields of application compared are: road, rail, maritime, agriculture, aviation, space LEO satellites and space GEO satellites.

A scale of feared events is proposed on Table 7 to build the mapping between these feared events and GNSS markets. The grade extends from 1 to 5 meaning the feared event has a low probability to be encountered to high probability.

<table>
<thead>
<tr>
<th>Grade</th>
<th>Signal strength</th>
<th>Multipath</th>
<th>Dynamics</th>
<th>Scintillations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-130 dBm</td>
<td>5 m/2°</td>
<td>5 kHz</td>
<td>0.2 S4 index</td>
</tr>
<tr>
<td>2</td>
<td>-135 dBm</td>
<td>5 m/90°</td>
<td>15 kHz</td>
<td>0.4 S4 index</td>
</tr>
<tr>
<td>3</td>
<td>-140 dBm</td>
<td>10 m/180°</td>
<td>25 kHz</td>
<td>0.6 S4 index</td>
</tr>
<tr>
<td>4</td>
<td>-145 dBm</td>
<td>20 m/360°</td>
<td>35 kHz</td>
<td>0.8 S4 index</td>
</tr>
<tr>
<td>5</td>
<td>-150 dBm</td>
<td>200 m/360°</td>
<td>45 kHz</td>
<td>1 S4 index</td>
</tr>
</tbody>
</table>

Table 7: Proposed scale of feared events for the mapping

In the Table 7, the criterion ‘signal strength’ corresponds to the minimal detected power of signal and allows to judge of the signal weakness. The ‘multipath’ criterion is based on height of reflective surfaces and on how much (in degrees) the receiver is surrounded by reflective surfaces. The ‘dynamics’ criterion is the Doppler shift value for a GPS L1 C/A. The Appendix 2 presents Doppler shift values computed in three GNSS applications. The ‘scintillations’ criterion is a scale to compare ionospheric scintillations probability of occurrence for each considered GNSS application. No metric is used to graduate the power of interferences because this feared event really depends on the receiver environment and can’t be generalised. So, just an appreciation of the occupied area has been taken into account to distinguish interferences impact according to the GNSS field of application.

Based on this graduation, the Figure 9 illustrates the mapping between studied feared events and chosen applications families.

Figure 9: Mapping between feared events and GNSS markets

Rail and road applications can be viewed as a same use case because almost same feared events are encountered. Indeed, these GNSS markets are particularly affected by scintillations and multipath effects due to buildings and vegetation.

As the main very restrictive feared events are scintillations and environments are similar (oceans and wide plains), maritime and agriculture fields can also be gathered.

Scintillations and interferences that can be continuous or pulsed are the most disturbing feared events in aviation.

High dynamics is a real constraint for space receivers in LEO satellite receivers. Indeed, for a GPS L1 C/A receiver, Doppler shift can reach values of 44.55 kHz (Appendix 3).

Space receivers embedded in GEO satellites mainly suffer from weak signals with signal strength of -151dBm for a GPS L1 C/A (Appendix 3).

In the following parts, robustness of proposed FPLL tracking algorithms to these five feared events spread in GNSS applications will be tested.

4.3 Low carrier to noise ratios

Multipath, interferences and high dynamics induces respectively signal fading, signal jamming and antenna pattern sweeping that generate variations of carrier to noise ratios. As previously shown, added to this, there are weak signals particularly in space GEO applications.

Thus, the first models’ performance test deals with evaluating robustness to several values of C/N0.
Conventional tracking gives divergent carrier phase tracking errors for low C/N0 while FPLL tracking algorithms provides a correct carrier phase synchronization. Figure 10 illustrates this distinction of performance on estimation of carrier phase tracking errors in case of C/N0=21 dBHz.

Therefore, proposed FPLL tracking algorithms (with variable and fixed Kalman gain) are more robust to low carrier to noise ratios than a conventional tracking.

4.4 Receiver dynamics

The receiver’s dynamics corresponds to all the movements of the receiver’s antenna with regard to the satellites radial movements. It implies frequency changes and, then has an impact on the calculation of the signal-to-noise ratio at the correlator output due to a moving correlator peak.

The worst case concerns space applications. Indeed, as provided in Appendix 3, LEO satellites embedded receivers have large Doppler shifts and Doppler shift rates which imply:

- A very good acquisition capability with optimized frequency cells search,
- A very good capability to track the signals, in particular with antenas attitudes evolutions,
- A capability to minimize the acquisition to tracking transition step,
- A capability to re-acquire (warm start) quickly the signals after tracking loss.

On the Figure 11 are plotted carrier phase tracking errors in case of the worst values of Doppler shift (44.55 kHz) and Doppler shift rate (13.53 Hz/s) for a LEO satellite receiver (GPS L1 C/A).

FPLL tracking architecture allows a correct synchronization in carrier phase with errors not varying much whereas conventional tracking provides divergent errors.

Thus, both proposed FPLL tracking algorithms are robust to high dynamics.

4.5 Ionospheric scintillations

Ionospheric scintillations are induced by ionospheric irregularities and affect GNSS signals in two ways: dispersion and diffraction. The first one affects the group delay and phase advance of the signal and, the second one scatters GNSS signal and causes fluctuations in the signal amplitude and phase.

The received signal affected by scintillations may be modelled as follows:

\[
s(t) = A\delta B_{\text{scintillation}}c(t)d(t)\cos(2\pi f_0 t - \theta + \delta \phi_{\text{scintillation}}) + n(t) \quad \text{(Eq 15)}
\]

Where:

- \(A\) is the signal amplitude,
- \(\delta B_{\text{scintillation}}\) is the scintillation magnitude,
- \(c(t)\) is the PRN code (L1 C/A BPSK here),
- \(d(t)\) is the navigation message,
- \(f_0\) is the nominal carrier frequency (intermediate frequency after down-conversion),
- \(\delta \phi_{\text{scintillation}}\) is the scintillation carrier phase,
- \(\theta\) is the carrier phase delay.

In the equation 15, the code delay is not taken into account. One can derive that scintillations will affect the correlators’ outputs as well as discriminators.
Scintillations were simulated, and the corresponding tracking performances were assessed. Figure 12 and Figure 13 present the phase and the amplitude of the generated scintillation.

![Figure 12: Generated scintillation phase](image)

![Figure 13: Generated scintillation amplitude](image)

On the Figure 14, the above curve shows that conventional tracking model provides convergent carrier tracking errors but around -6.3 radians. On the contrary, proposed FPLL tracking models allows to have convergent carrier phase errors around 0. Thus, proposed FPLL tracking is more robust to scintillations than conventional one. Moreover, a fixed Kalman gain in case of FPLL tracking seems a little bit more robust than a variable one that provides errors shifted of 0.05 rad.

### 4.6 Multipath

Multipath replicas are faded and delayed Non Line Of Sight RHCP signals (with at least two reflections to be polarized at the antenna level).

The following equations correspond to the correlator outputs in which a term represents multipath impact.

\[
I(t) = A \frac{d(t)\text{sinc}(\pi \epsilon_f)R_c(\epsilon_t)\cos(\epsilon_\phi)}{\text{Number of NLOS replicas}} + n_I(t) + \sum_{k=1}^{\text{Number of NLOS replicas}} A \frac{\alpha_k}{2} d(t)\text{sinc}(\pi \epsilon_f)R_c(\epsilon_t + \Delta \tau_k) \cos(\epsilon_\phi + \Delta \theta_k)
\]  
(Eq 16)

\[
Q(t) = A \frac{\text{sinc}(\pi \epsilon_f)R_c(\epsilon_t)\sin(\epsilon_\phi)}{\text{Number of NLOS replicas}} + n_Q(t) + \sum_{k=1}^{\text{Number of NLOS replicas}} A \frac{\alpha_k}{2} \text{sinc}(\pi \epsilon_f)R_c(\epsilon_t + \Delta \tau_k) \cos(\epsilon_\phi + \Delta \theta_k)
\]  
(Eq 17)

Where

- \(A\) is the input signal amplitude,
- \(d(t)\) is the modulation of the navigation message,
- \(R_c\) is the code autocorrelation,
- \(\epsilon_f\) is the signal Doppler frequency error,
- \(\epsilon_t\) is the signal code delay error,
- \(\epsilon_\phi\) is the signal carrier phase error,
- \(n_I(t)\) and \(n_Q(t)\) are the additive and uncorrelated white Gaussian noises,
- \(\Delta \tau_k\) is the k\textsuperscript{th} NLOS signal replica code delay,
- \(\Delta \theta_k\) is the k\textsuperscript{th} NLOS signal replica phase delay,
- \(\alpha_k\) is the k\textsuperscript{th} NLOS signal replica amplitude fading.

Multipath effects imply the presence of secondary peaks and depend on the waveform and the correlation characteristics. For the L1 C/A signal, a single correlation triangle results from the correlation of PRN codes with a BPSK modulation, whereas in presence of subcarriers with BOC modulations, there are secondary peaks.
The multipath impact depends on the processed signals and the discriminators characteristics to input in the tracking algorithm.

The Table 8 contains carrier phase error standard deviations of proposed FPLL tracking algorithms and conventional tracking model in order to compare their robustness.

<table>
<thead>
<tr>
<th>Model</th>
<th>Phase error STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional</td>
<td>0.33 rad</td>
</tr>
<tr>
<td>KF FPLL Fixed Gain</td>
<td>0.09 rad</td>
</tr>
<tr>
<td>KF FPLL Variable Gain</td>
<td>0.1 rad</td>
</tr>
</tbody>
</table>

Table 8: Comparison of carrier phase error standard deviations between conventional tracking and the two proposed FPLL tracking models in presence of linear multipath with 0.5 fading coefficient and $10^4$ seconds delay

All three models are robust to multipath effects with low carrier phase error standard deviations. However, FPLL tracking algorithms are more robust than conventional tracking. There is no really difference between fixed and variable Kalman gain in FPLL tracking algorithms’ robustness to multipath.

### 4.7 Carrier Waves Interferences

There are three main types of interference which are Carrier Waves (CW) interference, Wide Band (WB) interference and pulsed interference.

In this paper, only CW interference with sine-wave form is considered [14]:

$$CW(t) = \sqrt{P_{CW}} \cos(2\pi(f_I + \Delta f_J)t + \theta) \quad \text{(Eq 18)}$$

Where:

- $P_{CW}$ is the interference strength (in dBW),
- $\theta$ is the interference phase (in rad),
- $\Delta f_J$ is the frequency shift of the interference against the considered GNSS signal carrier frequency (in Hz),
- $f_I$ is the intermediate frequency of the considered GNSS signal (in Hz),
- $f_I + \Delta f_J$ is the central frequency (in Hz).

These interferences have an impact on the calculation of the signal-to-noise ratio at the correlator output as the interferences cause sinusoids on top of the correlator peak. The amplitude of the sinusoids is as important as the strength of interference is high.

To test robustness of FPLL tracking to interferences, a carrier waves one is generated with a power of -165 dBW. This performance test is made in the worst condition that is to say without relative Doppler between the interference and the useful signal meaning the interference stays on a spectrum line. And, the chosen spectrum line is a worst one which is characterized by a frequency of 227 kHz (it corresponds to a theoretical spectrum line of PRN 6) [13] (Appendix 4).

The Figure 15 represents carrier phase tracking errors for conventional tracking and for both proposed FPLL tracking models. There is a short time of convergence for the three algorithms but lower errors with FPLL than conventional tracking before convergence. Then, carrier phase errors vary more with the conventional tracking than with a FPLL one.

Figure 15: Carrier phase tracking errors (rad) with conventional model (blue), ‘Kalman_based_FPLL_fixed_gain’ (green) and ‘Kalman_based_FPLL_variable_gain’ (red) in case of a -165 dBW CW interference on the worst PRN6 spectrum line (C/N0=50dBHz)

The histogram of carrier phase tracking errors after convergence, on Figure 16, confirms the fact that FPLL tracking provides lower standard deviations than conventional tracking.

Figure 16: Histogram of carrier phase tracking errors standard deviations after convergence with conventional model (blue), ‘Kalman_based_FPLL_fixed_gain’ (green) and ‘Kalman_based_FPLL_variable_gain’ (red) in case of a -165 dBW CW interference on the worst PRN6 spectrum line (C/N0=50dBHz)
Therefore, proposed FPLL tracking architectures are more robust to carrier waves interferences in worst case than a conventional one. Moreover, according to the histogram, a variable Kalman gain (with a setting of noise matrix) is more robust to interferences than a fixed Kalman gain in case of a FPLL tracking.

4.8 Conclusion on proposed FPLL tracking models’ robustness

According to this study, a FPLL tracking algorithm has better performances and is more robust to feared events than a conventional tracking model, particularly in case of high dynamics. Moreover, using a FPLL tracking architecture enables to estimate jointly carrier phase and frequency tracking errors. Hence, it allows to jointly update the frequency synthesizer and the carrier NCO.

The choice of a variable Kalman gain seems to be wiser than a fixed one in case of a FPLL tracking affected by interferences. Except for this environment constraint, differentiation between fixed or variable Kalman gain in a FPLL tracking architecture cannot be really done on the robustness to feared events. It is the algorithms’ complexity and so, the flexibility to the application environment that make a choice possible between these two models.

Thus, proposed FPLL tracking algorithms are robust in harsh environments. It is a key point to answer to new and various GNSS needs linked with new environment constraints (part 4.2).

But, they require some improvements to take into account the receiver environment and to detect feared events.

5 INTEGRATION TO RECEIVER ARCHITECTURE AND DISCUSSION ABOUT DETECTION CAPABILITY

This part deals with the integration of the two proposed FPLL tracking algorithms in the receiver architecture to keep their robustness and bring detection of feared events. Thus, various possible signal processing architectures will be proposed and discussed.

5.1 Inter-channel tracking architecture

The first proposed tracking architecture is the inter-channel tracking. It is a kind of “vectorisation” of FPLL tracking but at tracking level. The idea is to extend proposed FPLL tracking algorithms by using a Kalman filter to estimate simultaneously carrier phase and frequency tracking errors of all tracking channels. That means it is a Kalman filter whose state vector is composed of carrier phase and frequency errors from all tracking channels.

This architecture is represented on the Figure 17. This architecture does still not modify the DLL tracking loop operation which occurs separately with a conventional loop filter.

![Figure 17: Inter-channel tracking architecture](image)

The interest of this architecture is to benefit from existing inter-channel correlation at synchronization level in order to be robust from this step of signal processing. Another advantage of inter-channel correlation is to detect tracking anomalies and feared events like interferences by comparison between channels’ measurements.

To improve this inter-channel tracking architecture and to take into account the receiver environment, it can be useful to add a state machine as represented on the Figure 18.

![Figure 18: Inter-channel tracking architecture with a state machine](image)

Indeed, this state machine uses environment registers that analyse and save environment data. It allows to optimize signal processing by a selection of useful channels for synchronization process and a selection of noise settings in case of a variable Kalman gain. This last selection capability improves the measurement and dynamics noise matrix configurations.
Thanks to cross correlation and this state machine, it strengthens robustness and it ensures detection of feared events and selection capabilities.

Nevertheless, problem on one channel can generate dysfunction of all tracking process.

### 5.2 Kalman/ARMA-based algorithm

The patented Kalman/ARMA algorithm [15] can be used to improve proposed FPLL tracking algorithms in two ways: settings of variable Kalman gain and interference detection.

The ARMA (Auto-Regressive Moving Average) model is a particularly simple parametric model for signals considered here as "discrete time" processes; purely nondeterministic. ARMA modeling assumes that the signal is generated by a linear difference equation, with a finite order, which only shows some past values of the signal multiplied by coefficients (AR part) to which is added a random term (MA part).

An ARMA process is therefore an IIR (Infinite Impulse Response) filter which input is a zero-mean white noise. The coefficients of the ARMA process are separated into two groups: AR \( \{a_i; i \in [1, p]\} \) and MA \( \{b_i; i \in [1, q]\} \).

The discrete time equation of the ARMA model is expressed as follows:

\[
y_k + \sum_{i=1}^{p} a_i y_{k-i} = V_k + \sum_{i=1}^{q} b_i V_{k-i} \quad \forall k \in N
\]

Where:
- \( p \) is the AR part order of the ARMA model,
- \( q \) is the MA part order of the ARMA model,
- \( V \) is a discrete white noise (as input) with zero mean with variance \( \sigma^2 \),
- \( y \) is an output time sampled signal from the considered ARMA process.

In this patented algorithm, the coefficients of an ARMA model are identified in an optimal way by approximating the ARMA model by a long AR model, that is to say with higher order than the orders of the classical ARMA model. This method does not solve the first \( p \) equations of Modified Yule Walker (MYW) but an infinite number of equations. Hence, its accuracy is increased. It is the AR part which contains the useful information concerning the signal to be extracted. The MA part contains the noise characteristics.

Then, thanks to the ARMA’s coefficients computation, the state transition matrix \( \phi \) and more particularly, the Kalman filter gain can be configured.

In case of a second order Kalman-based tracking architecture, the relationship between the ARMA model and the filter is provided by the following equations:

\[
\phi = \begin{bmatrix} -a_1 & 1 \\ -a_2 & 0 \end{bmatrix} \quad \text{(Eq 20)}
\]

\[
K = \begin{bmatrix} 1 - \frac{b_2}{a_2} \\ \frac{b_1}{a_2} \end{bmatrix} \quad \text{(Eq 21)}
\]

Where:
- \( \phi \) is the state transition matrix,
- \( K \) is the Kalman filter gain,
- \( a_1 \) and \( a_2 \) are AR part coefficients,
- \( b_1 \) and \( b_2 \) are AR part coefficients.

More details about this algorithm are available in the Appendix 5.

This patented Kalman/ARMA algorithm is an idea to automate Kalman filter gain computation bypassing the demanding noise matrix configurations. Moreover, using ARMA coefficients allows to take into account the receiver environment and so, the variable Kalman gain is set up according to it. That is why it can be very useful to use it instead of a classic variable Kalman filter in a FPLL tracking architecture. It enables to have an adaptive gain to feared events.

On the other part, this algorithm can be used for detection of carrier waves interferences. Indeed, filtered tracking errors are analysed in case of sinusoidal signals and so, carrier wave interference can be detected by consideration of the innovation term.

Nevertheless, the presented relationship between the ARMA model and the Kalman filter characteristics deals with a Kalman-based second order tracking. Thus, using the Kalman/ARMA patent for a FPLL tracking model requires some adjustments.

### 5.3 FAPI (Fast Approximation Power Iterated) algorithm

The patented FAPI algorithm [16] can be used to improve proposed FPLL tracking algorithms in two ways: reduction of time of convergence and interference detection.

It is not a Kalman-based algorithm but a model based on subspaces decomposition and power iterated method. FAPI algorithm provides better results than a Kalman filter. Moreover, it requires less demanding settings. The comparison is presented in the Appendix 6. Thus, it is an idea to replace the frequency and phase tracking architectures presented in this paper.
The FAPI algorithm provides several advantages. First, it is stable and fast. Secondly, it guarantees the signal space base orthonormality at each iteration. It also avoids matrix inversion and square matrix. Moreover, recursive least square method is used in order to track sinusoids’ amplitude and phase. Thanks to all these benefits, this algorithm has better performances than several tracking algorithms based on subspaces decomposition and power iterated method like Projection Approximation Subspace Tracking (PAST), Orthogonal Projection Approximation Subspace tracking (PAST) and Novel Information Criterion (NIC). The FAPI model is also faster than these other algorithms. The comparison between these algorithms is presented in the Appendix 7.

To prove the interest of the FAPI algorithm in a FPLL tracking architecture, some preliminaries results are exposed hereafter.

Figure 19 and Figure 20 represent how a FAPI model can track theoretical frequency and carrier phase variations of a down-converted signal.

Contrary to Kalman/ARMA algorithm, the FAPI algorithm may not be used to set the FPLL tracking but to decrease the loops time of convergence and to improve robustness against high dynamics since it seems robust against brutal variations in frequencies. Moreover, further investigations are expected to demonstrate that FAPI may contribute to make more robust acquisition to tracking transition. In addition, both Kalman/ARMA and FAPI algorithms are candidates to provide interference detection.

5.4 An improved vector tracking architecture

The architecture discussed hereafter is an improved vector tracking considering here tracking and navigation levels as it is explained in part 2.4. This architecture is illustrated in the Figure 21.
The first change point compared to a known vector tracking is the use of a FPLL tracking at tracking level instead of a conventional PLL associated to a VDFLL including navigation level. As known vector tracking and extended Kalman filter at navigation level already provide robustness, adding proposed FPLL tracking should improve performances in harsh environments.

Then, as seen previously, cross-correlation is a way to detect processing anomalies and feared events by comparison of channels’ outputs.

Finally, the receiver environment is monitored thanks to registers to save some parameters like the Dilution Of Precision (DOP), the number of satellites in view, the constellations, the signals’ C/N0 and so on... It is interesting to monitor the environment thanks to metrics registers to set the receiver algorithms (in particular the algorithms presented in this paper) to perform a targeted operation. Thus, this state machine allows to select the most promising algorithms, settings and useful channels for targeted applications. These registers accelerate signal processing, reduce demanding computations and reset tracking process.

A key point to improve again the vector tracking architecture can be the use of the Kalman/ARMA patent [15] instead of the extended Kalman filter at navigation level to compute position. The Kalman/ARMA algorithm may allow to detect the receiver antenna phase centre oscillations and vibrations at the PVT level.

The remaining problem is if there is a dysfunction on a channel, it has an impact on the entire signal processing architecture.

6 CONCLUSION AND FUTURE WORK FOR FLEXIBILITY CHALLENGE

6.1 Conclusion

With a growing GNSS market, new client needs appear. This emergence of specific GNSS applications causes the emergence of new environments constraints. Receivers have to be able to process signals in harsh environments. That is why known tracking architectures discussed at the beginning of this paper, must be improved to answer to three main constraints:

- Robustness to various feared events according to the receiver environment,
- Detection of feared events and tracking anomalies capability,
- Flexibility to several GNSS applications, new and diverse GNSS signals, and several hardware platforms.

In this paper, two FPLL tracking algorithms are assessed to jointly estimate carrier phase and frequency on one channel and without considering navigation process. Their performances are tested in harsh environments to prove their robustness to various GNSS applications. Finally, various tracking architectures are discussed to integrate proposed FPLL tracking models in the receiver architecture, to improve these algorithms’ robustness and to add feared events detection capability.

6.2 Perspectives about flexibility challenge

This paper deals with robustness and provides a discussion on detection capability but it remains the question of flexibility. Proposed FPLL algorithms may answer to challenging applications as geolocalisation in rail market, LEO satellites embedded GNSS receivers or, racers’ positioning in triathlon competition…

First, tracking architectures have to be able to process various new GNSS signals like Galileo E1 and E5. So, performances of proposed FPLL tracking algorithms have to be tested on other signals and some choices have to be made on algorithms computation and settings. Moreover, multithread studies have to be led to process simultaneously signals from different constellations, of different frequencies and various waveforms.

Secondly, software algorithms have to be embedded to miniaturized hardware platforms in order to adapt to several applications. This step requires an electronic chip study, a CPU power analysis to define the memory repartition, a compatibility study between the software and the hardware target, and also, a management of real time processes and multithreads.

ACKNOWLEDGMENTS

The authors thank the ALTRAN GNSS team, and more particularly, Romain Crapart (GNSS engineer involved in GNSS space receiver projects) and Jean-Philippe Puy (Signal processing expert) for all their involvement and help.

The authors wish to thank also the ENAC SIGNAV Laboratory for answering some questions and presenting its thoughts and remarks on this study.
REFERENCES


[13] Jong-Hoon Won, Dominik Dötterböck and Bernd Eisefeller (2010), *Performance comparison of different forms of Kalman filter approaches for vector-based GNSS signal tracking loop*, The institute of geodesy and navigation, University FAF Munich, Germany


[16] ALTRAN submission order IFB W17 ALR OMA/FH for: “FAPI”
APPENDIX 1: Computation of a fixed Kalman gain

A state system has the following form:

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx + Du
\end{align*}
\]

Where:

- \(x\) is the state,
- \(y\) is the system output,
- \(u\) is the system input.

The Kalman filter system of equations is the following:

\[
\begin{align*}
\hat{X}_{k|k-1} &= \phi \cdot \hat{X}_{k-1|k-1} + \Gamma \cdot e_k \\
\hat{Y}_k &= H\hat{X}_{k|k-1} + u_k
\end{align*}
\]

Where:

- \(X\) is the state vector,
- \(\phi\) is the state transition matrix,
- \(\Gamma\) is the state transition dynamic vector,
- \(e\) is the dynamic white noise or the dynamic error with mean zero and variance \(\sigma_e^2\),
- \(Y\) is the measurement vector,
- \(H\) is the observation matrix,
- \(u\) is the measurement white noise with mean zero and variance \(\sigma_u^2\).

The fixed gain is obtained at the equilibrium state. The equilibrium state can be calculated with the pole assignment. This technique enables to obtain a closed loop system which poles are assigned appropriately.

In the case of a state system with multiple variables, pole placement consists in imposing the eigenvalues of the matrix \(M=\phi - KH\).

In the «KF_fixed_gain» model:

\[
M = \begin{bmatrix}
1 & Te & Te^2/2 \\
0 & 1 & Te \\
0 & 0 & 1
\end{bmatrix} - \begin{bmatrix}
K_1 \\
K_2 \\
K_3
\end{bmatrix} \begin{bmatrix}
Te/2 & Te^2/6
\end{bmatrix}
\]

Yet, eigenvalues of a fixed gain Kalman filter can be seen as close to those of a Butterworth filter. Indeed, the Butterworth filter is a linear filter with constant gain in the band.

\[
\begin{align*}
\alpha &= e^{-2\pi BPLL Te} \\
\beta &= e^{-(1+i\sqrt{3})\pi BPLL Te} \\
\gamma &= e^{-(1-i\sqrt{3})\pi BPLL Te}
\end{align*}
\]

M-matrix characteristic polynomial calculation:

\[
\begin{align*}
det(M - \lambda I_3) &= 1 - K_1 + \frac{K_2 Te - K_3 Te^2}{2} + \frac{3 - 2K_1 - \frac{2K_3 Te^2}{3}}{2} \lambda + \frac{3 - K_1 - \frac{K_2 Te}{2} - \frac{K_3 Te^2}{6}}{2} \lambda^2 - \lambda^3 \\
\end{align*}
\]

Order 3 polynomial:

\[
\begin{align*}
(\alpha - \lambda)(\lambda - \beta)(\lambda - \gamma) &= -\lambda^3 + (\alpha + \gamma + \beta)\lambda^2 - (\gamma\alpha + \alpha\beta + \beta\gamma)\lambda + \alpha\beta\gamma
\end{align*}
\]

Identification:

Calculating the K gain is equal to resolving the following matrix system:

\[
\begin{bmatrix}
\alpha + \gamma + \beta \\
-\gamma\alpha + \alpha\beta + \beta\gamma \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
-1 & -Te/2 & -Te^2/6 \\
2 & 0 & -2Te^2/3 \\
-1 & Te/2 & -Te^2/6 \\
3 & -3 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
K_1 \\
K_2 \\
K_3
\end{bmatrix}
\]

20
APPENDIX 2: Computation of Doppler shift values in three GNSS application cases

Doppler values considered in this paper are estimated as the worst case values for different use cases, characterized by the maximum speed and typical altitude. The use cases are:

- On ground use-case (pedestrian or car usage): maximum speed of 40m/s (144km/h) and altitude of 0m,
- On flight use-case (plane): maximum speed of 250m/s (900km/h) and altitude of 10km,
- Orbital use-case (satellite): a speed of 7670 m/s for an altitude of 400km (ISS).

On the GNSS satellite site, the GPS constellation is considered, with an altitude of 20200km and a speed of 3872m/s. The L1 band is considered as it has the highest frequency (worst case).

The Doppler value is a combination of the value induced by the satellite speed and the value induced by the device speed. The radial speed only has to be considered.

Doppler frequency offset is computed using the following formula:

\[ \Delta f = f_0 \cdot \frac{v_{\text{Rad}}}{c} \]

Where:

- \( f_0 \) is the carrier frequency,
- \( v_{\text{Rad}} \) is the radial speed from one device toward the other one,
- \( c \) is the speed of light.

The worst case geometry is obtained when the device has the satellite at the horizon: the device speed is then radial, while the satellite radial speed is at its maximum value.

Doppler induced by satellite speed

The worst case radial satellite speed is computed using the following formula:

\[ v_{\text{Sat}}^{\text{Rad}} = v_{\text{Sat}} \cdot \frac{r + a_{\text{Dev}}}{r + a_{\text{Sat}}} \]

Where:

- \( v_{\text{Sat}} \) is the satellite speed,
- \( r \) is the earth radius,
- \( a_{\text{Dev}} \) is the device altitude,
- \( a_{\text{Sat}} \) is the satellite altitude.

For the different use cases previously defined, we can compute:

<table>
<thead>
<tr>
<th>Use case</th>
<th>( \Delta f_{\text{Sat}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>On ground</td>
<td>4879Hz</td>
</tr>
<tr>
<td>On flight</td>
<td>4887Hz</td>
</tr>
<tr>
<td>Orbital</td>
<td>5186Hz</td>
</tr>
</tbody>
</table>

Doppler induced by device speed

As a worst case, we can consider the device speed is always radial (while the satellite is at its horizon).

The Doppler frequency offset induced by the device speed is:

<table>
<thead>
<tr>
<th>Use case</th>
<th>( \Delta f_{\text{Dev}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>On ground</td>
<td>210Hz</td>
</tr>
<tr>
<td>On flight</td>
<td>1314Hz</td>
</tr>
<tr>
<td>Orbital</td>
<td>40.5kHz</td>
</tr>
</tbody>
</table>

The on flight estimation may be underestimated by not considering the Doppler induced in case of high values of jerk.

Combined Doppler frequency offset

By combining these 2 estimates, some coarse values of Doppler frequency offset can be provided for the 3 use cases:

<table>
<thead>
<tr>
<th>Use case</th>
<th>( \Delta f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>On ground</td>
<td>4kHz</td>
</tr>
<tr>
<td>On flight</td>
<td>6kHz</td>
</tr>
<tr>
<td>Orbital</td>
<td>50kHz</td>
</tr>
</tbody>
</table>
APPENDIX 3: Doppler estimation for GNSS receivers embedded in LEO (circular orbit with 500 km altitude) and GEO satellites

Table 9: Values of Doppler shift and Doppler shift rate for GNSS receivers on some LEO (400 km altitude) and GEO satellites (35800 km altitude)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>LEO</th>
<th>GEO</th>
<th>LEO</th>
<th>GEO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max acquisition threshold (LOST)</td>
<td>122.6 dBm (31 GPS L1)</td>
<td>131.6 dBm (30 GAL, SOY)</td>
<td>151.6 dBm (31 GPS L1)</td>
<td>156.6 dBm (30 GAL, SOY)</td>
</tr>
<tr>
<td>Range of Doppler shift on L1</td>
<td>-44.5 kHz</td>
<td>-43.0 kHz</td>
<td>-14.0 kHz</td>
<td>-13.5 kHz</td>
</tr>
<tr>
<td>Range of Doppler shift rate on L1</td>
<td>-40.5 kHz</td>
<td>-38.0 kHz</td>
<td>-13.0 kHz</td>
<td>-12.5 kHz</td>
</tr>
<tr>
<td>Range of Doppler shift rate on L1</td>
<td>-75.9 Hz/s</td>
<td>-67.9 Hz/s</td>
<td>5.2 Hz/s</td>
<td>5.7 Hz/s</td>
</tr>
<tr>
<td>Estimated range of Doppler shift</td>
<td>33.26 kHz</td>
<td>33.03 kHz</td>
<td>26.5 kHz</td>
<td>26.2 kHz</td>
</tr>
<tr>
<td>Estimated range of Doppler shift</td>
<td>-50.0 Hz/s</td>
<td>-50.1 Hz/s</td>
<td>2.0 Hz/s</td>
<td>2.1 Hz/s</td>
</tr>
</tbody>
</table>

APPENDIX 4: L1 C/A and E1 OS high amplitude code spectrum lines

The worst code spectrum lines in terms of power level, for each signal are given hereafter and provided for each PRN. These power levels take into account the PRN code FFT and the Fourier transform of the materialization waveform. These values are obtained by generating PRN code values, then computing the spectrum of the transmitted waveforms and comparing all code spectrum lines of GPS L1 C/A and future Galileo E1 OS signals.

In the Table 10 are identified the amplitudes (AMPL) and frequency (FREQ) of the worst code spectrum lines for each PRN of GPS L1 C/A and Galileo E1 OS code. The amplitude values presented are the power level with regard to the full signal power level.

Table 10: Worst line characteristics for each PRN for GPS C/A code

<table>
<thead>
<tr>
<th>C/A CODE PRN N°</th>
<th>WORST LINE FREQ (kHz)</th>
<th>WORST LINE AMPL (dB)</th>
<th>C/A CODE PRN N°</th>
<th>WORST LINE FREQ (kHz)</th>
<th>WORST LINE AMPL (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>42</td>
<td>-22.7</td>
<td>20</td>
<td>30</td>
<td>-22.7</td>
</tr>
<tr>
<td>2</td>
<td>263</td>
<td>-23.12</td>
<td>4</td>
<td>32</td>
<td>-22.12</td>
</tr>
<tr>
<td>3</td>
<td>108</td>
<td>-22.84</td>
<td>7</td>
<td>32</td>
<td>-22.84</td>
</tr>
<tr>
<td>4</td>
<td>332</td>
<td>-22.98</td>
<td>10</td>
<td>22</td>
<td>-22.98</td>
</tr>
<tr>
<td>5</td>
<td>23</td>
<td>-21.53</td>
<td>15</td>
<td>22</td>
<td>-21.53</td>
</tr>
<tr>
<td>6</td>
<td>227</td>
<td>-21.39</td>
<td>18</td>
<td>22</td>
<td>-21.39</td>
</tr>
<tr>
<td>7</td>
<td>78</td>
<td>-23.27</td>
<td>22</td>
<td>22</td>
<td>-23.27</td>
</tr>
<tr>
<td>8</td>
<td>166</td>
<td>-21.50</td>
<td>26</td>
<td>22</td>
<td>-21.50</td>
</tr>
<tr>
<td>10</td>
<td>16</td>
<td>-22.45</td>
<td>33</td>
<td>22</td>
<td>-22.45</td>
</tr>
<tr>
<td>11</td>
<td>123</td>
<td>-22.64</td>
<td>37</td>
<td>22</td>
<td>-22.64</td>
</tr>
<tr>
<td>12</td>
<td>199</td>
<td>-22.08</td>
<td>40</td>
<td>22</td>
<td>-22.08</td>
</tr>
<tr>
<td>13</td>
<td>214</td>
<td>-23.12</td>
<td>43</td>
<td>22</td>
<td>-23.12</td>
</tr>
<tr>
<td>14</td>
<td>130</td>
<td>-22.35</td>
<td>46</td>
<td>22</td>
<td>-22.35</td>
</tr>
<tr>
<td>15</td>
<td>69</td>
<td>-21.90</td>
<td>49</td>
<td>22</td>
<td>-21.90</td>
</tr>
<tr>
<td>16</td>
<td>154</td>
<td>-22.58</td>
<td>52</td>
<td>22</td>
<td>-22.58</td>
</tr>
<tr>
<td>17</td>
<td>138</td>
<td>-22.10</td>
<td>56</td>
<td>22</td>
<td>-22.10</td>
</tr>
<tr>
<td>18</td>
<td>183</td>
<td>-21.40</td>
<td>59</td>
<td>22</td>
<td>-21.40</td>
</tr>
<tr>
<td>19</td>
<td>223</td>
<td>-23.17</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX 5: Kalman/ARMA-based algorithm

The discrete time equation of the ARMA model is expressed as follows:

\[ y_k + \sum_{i=1}^{p} a_i y_{k-i} = V_k + \sum_{i=1}^{q} b_i V_{k-i} \quad \forall k \in \mathbb{N} \]

Where:
- \( p \) is the AR part order of the ARMA model,
- \( q \) is the MA part order of the ARMA model,
- \( V \) is a discrete white noise (as input) with zero mean with variance \( \sigma_v^2 \),
- \( y \) is an output time sampled signal from the considered ARMA process.

The ARMA model \( z \)-transfer function in the frequency domain is:

\[ H(z^{-1}) = \frac{1 + \sum_{i=1}^{q} b_i z^{-i}}{1 + \sum_{i=1}^{p} a_i z^{-i}} \]

And \( Y(z^{-1}) = H(z^{-1}) U(z^{-1}) \)

Where:
- \( Y(z^{-1}) \) is the \( y \) \( z \)-transform,
- \( U(z^{-1}) \) is the \( V \) \( z \)-transform,
- \( z = e^{2\pi j f T_e} \),
- \( T_e \) is the sampling period,
- \( f \) is the frequency,
- \( a_i, i \in [1,p] \) are the AR part coefficients of the ARMA process,
- \( b_i, i \in [1,q] \) are the MA part coefficients of the ARMA process.

This representation is much more accurate than the FFT one and more adapted to the tracking of signals in noisy environments.

In this Kalman/ARMA algorithm, the coefficients of an ARMA model are identified in an optimal way by approximating the ARMA model by a long AR model, that is to say with higher order than the orders of the classical ARMA model. This method does not solve the first \( p \) equations of Modified Yule Walker (MYW) but an infinite number of equations.

This method consists in solving Modified Yule Walker equations to calculate the AR part and determine the MA part as function of the AR part of the ARMA process and the long MA model.

AR part

It consists in solving the first \( p \) Modified Yule Walker equations for \( i \in [1,m] \):

\[ \sum_{j=1}^{p} R_{yy}(i+j-1)a_j = -R_{yy}(i) \]

With \( R_{yy} \) the autocorrelation of the output time sampled signal.

It may be noted: \( A = \phi^{-1} \Gamma \)

Where:
- \( \Phi \) is the state transition matrix expressed as:

\[
\phi = \begin{bmatrix}
\varphi_{yy}(q) & \varphi_{yy}(q-1) & \cdots & \varphi_{yy}(q-p+1) \\
\varphi_{yy}(q+1) & \varphi_{yy}(q) & \cdots & \varphi_{yy}(q-p+2) \\
\vdots & \vdots & \ddots & \vdots \\
\varphi_{yy}(q+p-2) & \varphi_{yy}(q+p-1) & \cdots & \varphi_{yy}(q)
\end{bmatrix}
\]

- \( \Gamma \) is the state transition dynamic vector expressed as:

\[
\Gamma = \begin{bmatrix}
\varphi_{yy}(q+1) \\
\varphi_{yy}(q+2) \\
\vdots \\
\varphi_{yy}(q+p)
\end{bmatrix}
\]

MA part

The ARMA model transfer function is:

\[ H(z^{-1}) = \frac{1 + \sum_{i=1}^{q} b_i z^{-i}}{1 + \sum_{i=1}^{p} a_i z^{-i}} \]

The equivalent MA transfer function is:

\[ H(z^{-1}) = \sigma_y \left[ 1 + \sum_{i=1}^{\infty} \bar{r}(n,i) z^{-i} \right] \]

As a consequence, the MA part obtained as a function of the long-MA model and the AR part is:

- For \( j \in [1, q] \):

\[ b_j = \bar{r}(n,j) + \sum_{i=1}^{\min(j,p)} a_i \bar{r}(n,j-i) \]

- For \( j \in [q+1, m] \):
\[ 0 = \bar{r}(n, j) + \sum_{i=1}^{\min(j, p)} a_i \bar{r}(n, j-i) \]

The long MA model is dependent on the long AR model coefficients.

The initialization is made thanks to two matrix:

\[ \text{mat} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -a_1 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \text{mat}(m,1) & \cdots & -a_1 & 1 \end{bmatrix} \]

And, \( \text{mat1} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -a_1 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \text{mat}(m+1,1) & \cdots & -a_1 & 1 \end{bmatrix} \)

- For \( i \in [j+2, m] \) and \( j \in [1, q] \), if \( i > k \):

\[ \text{mat}(i, j) = -\sum_{k=1}^{\min(i, p)} a_k \text{mat}(i-k, j) \]

It is the impulse response of the AR part.

- For \( i \in [j+2, m+1] \) and \( j \in [1, q] \), if \( i > k \):

\[ \text{mat1}(i, j) = -\sum_{k=1}^{\min(i, p)} a_k \text{mat1}(i-k, j) \]

And for \( i \in [1, m] \),

\[ \text{col}(i) = -\text{mat1}(i+1,1) + \bar{r}(n,i) \]

Thus, for \( j \in [1, q] \), the pseudo-inverse matrix (least squares) is:

\[ B = [\text{mat}^t * \text{mat}]^{-1} \text{mat}^t \text{.col} \]

If the MA part is known, the dual algorithm remains the same but the AR and MA (classical and long) parts are inverted.

**Link between the ARMA model and the filter**

In case of a Kalman-based second order tracking, the relationship between the ARMA model and the filter is provided by the following equation:

\[ \frac{Y(z^{-1})}{U(z^{-1})} = H(I - \phi . z^{-1})^{-1}. K^* . z^{-1} \]

\[ = \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \]

Where:

- \( H = [1 \ 0] \) is the measurement matrix,
- \( \phi = [\phi_1 \ \phi_3] \) is the state transition matrix,
- \( K^* = \phi K \) is the Kalman filter predicted gain

with \( K^* = \begin{bmatrix} K_1^* \\ K_2^* \end{bmatrix} \) and \( K = \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} \).

Therefore, ARMA coefficients can be expressed as:

\[ a_1 = -(\phi_1 + \phi_4) \]
\[ a_2 = \phi_1 \phi_4 - \phi_2 \phi_3 \]
\[ b_1 = \phi_1 \phi_4 - \phi_2 \phi_3 + K_2^* \phi_3 - K_1^* \phi_4 \]
\[ b_2 = K_1^* - \phi_1 - \phi_4 = K_1^* + a_1 \]

And, the Kalman filter gain can be expressed as:

\[ K = \begin{bmatrix} 1 - \frac{b_2}{a_2} \\ \frac{b_2 a_1}{a_2} \end{bmatrix} \]
APPENDIX 6: Comparison of FAPI algorithm and Kalman filter

Here, is compared the FAPI algorithm with a Kalman filter for a generated signal by damping oscillator method.

The Figure 22 illustrates the estimation the generated signal by the FAPI model and a Kalman filter.

![Comparison of FAPI and Kalman filter estimations of a generated signal](image)

The FAPI algorithm provides better results than the Kalman filter. Moreover, this algorithm requires less settings than the Kalman filter because it just needs to define the size of a weight matrix while the Kalman filter needs demanding noise settings. More the size of this weight matrix increases more the results will be better but computation speed will be lower. For a Kalman filter, it’s a better noise matrix configuration that provides better results but it is not as easy as for the FAPI algorithm.

APPENDIX 7: Comparison between FAPI algorithm and other subspaces decomposition methods

Here, the FAPI algorithm is compared to PAST (Projection Approximation Subspace Tracking), OPAST (Orthogonal Projection Approximation Subspace Tracking) and API (Approximation Power Iterative) algorithms that are all based on subspaces decomposition.

This comparison is made according to two scenarios:
- 1st scenario: on a periodic signal $x(k)$ with $k = \{1, 2, \ldots, N\}$.

$$x(k) = \begin{cases} \sin(2\pi f_0 k T_e) + \sin(6\pi f_0 k T_e), & \text{if } k < \frac{N}{2} \\ \sin(4\pi f_0 k T_e) + \sin(12\pi f_0 k T_e), & \text{else} \end{cases}$$

The $W_{n,r}$ subspace dimensions are defined as: $r=4$ and $n=100$.
- 2nd scenario: on a predicted signal thanks to an AR model identified by the least square method. The input signal is generated by a damping oscillator model.

The Table 11 contains estimation errors of all compared methods of subspaces decomposition.

<table>
<thead>
<tr>
<th></th>
<th>1st scenario</th>
<th>2nd scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAST</td>
<td>0.41049</td>
<td>0.021069</td>
</tr>
<tr>
<td>OPAST</td>
<td>0.41049</td>
<td>0.021069</td>
</tr>
<tr>
<td>API</td>
<td>0.32733</td>
<td>0.020655</td>
</tr>
<tr>
<td>FAPI</td>
<td>0.32733</td>
<td>0.020655</td>
</tr>
</tbody>
</table>

Table 11: Estimation errors of decomposition methods

In conclusion, PAST and OPAST algorithms provide the same results and API and FAPI methods provide the same results.

The API and FAPI models have better results in terms of speed of convergence and low estimation errors. These methods are robust to signal frequency change.

The FAPI method is the best one because it is faster than the API model in terms of computation.