International Global Navigation Satellite Systems Society
IGNSS Symposium 2015

Colombo Theatres, Kensington Campus, UNSW Australia
6 8 December 2016

The effect of sampling frequency and front-end bandwidth on the DLL code tracking performance

Vinh T Tran
Australian Centre for Space Engineering Research
University of New South Wales, Sydney, Australia

Nagaraj C Shivaramaiah
University of Colorado at Boulder, USA

Thuan D Nguyen
NAVIS centre, Hanoi University of Science and Technology, Hanoi, Viet Nam

Andrew G Dempster
Australian Centre for Space Engineering Research
University of New South Wales, Sydney, Australia

ABSTRACT

The synchronization of the received pseudorandom (PRN) code and the local generated replica is fundamental to compute user position in Global Navigation Satellite System (GNSS) receivers. The more accurate correlation output and Delay Locked Loop (DLL) code tracking error are described in this paper from the hardware receiver point of view. Estimation is based on the number of samples per code chip and the residual code phase of the code Numerical Controlled Oscillator (NCO) after a code chip is generated. The theoretical as well as experiment results show that the relation between the sampling frequency and front-end filter bandwidth has a strong influence in the code tracking. Increasing the sampling frequency can help to improve the DLL tracking performance if and only if the front-end bandwidth is much smaller than the sampling frequency. The more accurate estimation of the correlation and the DLL tracking error, proposed in this paper, can generally apply on precisely modeling GNSS receiver baseband signal processing.

1 Introduction

Global Navigation Satellite System (GNSS) receiver technology has changed dramatically since the first reception of Global Positioning System (GPS) signals, from complex and partly-analogue circuits to sophisticated, small, multichannel digital receivers or fully software defined radio architectures as presented in (Tran et al., 2016a). However, the navigation concept is still based on the estimation of the range between the user position to a set of at least 4 satellites using trilateration. This range measurement is accomplished by the synchronisation of the received pseudorandom (PRN) code and a locally generated replica inside the receiver to estimate a satellite time of transmission, which subtracted from a local time of reception gives rise to the so-called pseudorange (Parkinson, 1996). The code synchronisation is performed by a Delay Lock Loop (DLL), while a Phase Lock Loop (PLL) is used to keep track of the carrier. The better the code alignment, the greater the accuracy in the user position estimate.

A new direction for estimating the correlation output and DLL tracking error are recalculated in this paper from the hardware receiver perspective. The instantaneous residual code phase of each code chip is used to accurately estimate the correlator output and the DLL code tracking error. The impact of the sampling frequency on correlation output, which is usually presented through the sampling filter bandwidth, while the number of samples per code chip is assumed to be infinite (Akos and Braasch, 1996; Kaplan and Hegarty, 2005; Jang et al., 2012; Kou and Morton, 2013; Betz, 2015), will be calculated accurately based on the number of samples per code chip. Furthermore, both theoretical and experiment results show that the relation between the sampling frequency and front-end filter bandwidth has a strong influence on the code tracking. Increasing the sampling frequency can help to improve the DLL tracking performance if and only if the front-end bandwidth is much smaller than the sampling frequency.

This paper is organized as follows. Section II provides mathematical models of the intermediate frequency (IF) and baseband signals according to the number of samples per code chip and the Numerically Controlled Oscillator (NCO) residual code phase estimation. Section III presents the correlation output calculation depending on the sampling frequency. The effect of samlping frequency and front-end bandwidth on the DLL tracking error is analyses in Section IV. Concluding remarks follow in Section V.

2 Signal processing model and residual code phase estimation

Figure 1 illustrates the block diagram of the digital signal processing core of a GNSS receiver, where the locally generated signal keeps track of the incoming signal by adjusting the Code NCO and Carrier NCO. The performance of the receiver depends on the accuracy of this synchronisation.

The signal at the input of the digital part of a GNSS receiver is generally an IF signal, obtained by down-converting a radio-frequency (RF) signal. The front-end filter is here approximated by an ideal pre-correlation filter, which is a rectangular bandpass filter (BPF) centred at the IF

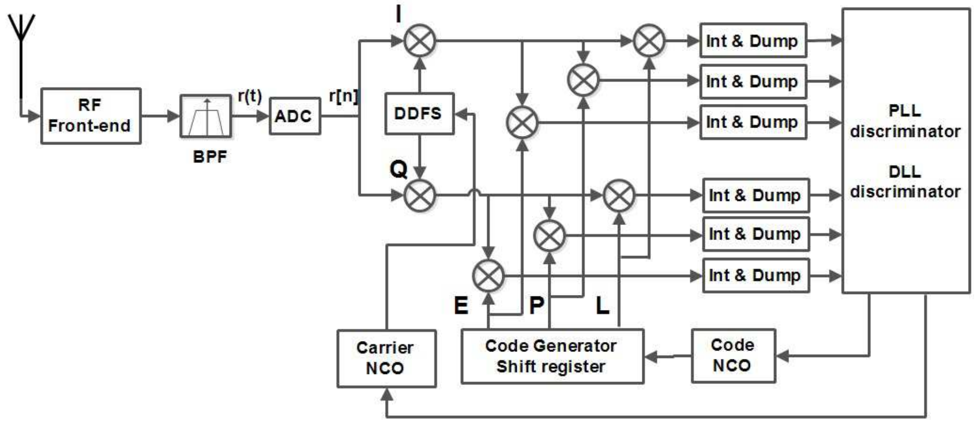

Figure 1: Block diagram of a digital GNSS receiver

frequency (f_{IF}) with two-sided bandwidth β_r, passing frequencies are $|f - f_{IF}| \leq \beta_r/2$. The IF signal of one satellite can be written as:

$$r(t) = \sqrt{2C_s} \sum_{k=-\infty}^{\infty} C_{\{k\}_N} \Pi(t - kT_c - \tau) d(t-\tau) cos(2\pi(f_{IF} + f_D)t + \theta) + w(t) \tag{1}$$

where C_s is the received signal power; Π is the rectangular function; $C_k = \pm 1$ is the k^{th} PRN code chip, $\{k\}_N$ is k modulus N, and N is the PRN code length (N=1023 for GPS L1 C/A); T_c is the code chip duration; $d(t-\tau)$ is the binary phase shift keying (BPSK) modulated navigation message and $d(t) = \pm 1$; τ is the timing delay of the received signal; f_{IF} and f_D denote the intermediate frequency and the carrier Doppler shift, respectively; θ is the random unknown carrier phase; $w(t)$ is Additive White Gaussian Noise (AWGN) with zero mean and Power Spectral Density (PSD) N_0.

The IF signal is then sampled by an analog-to-digital converter (ADC). The ADC generates a sequence of samples $r[nT_s]$, obtained by sampling $r(t)$ at the sampling frequency $(f_s = 1/T_s) \geq 2(f_{IF} + \beta_r/2)$. The notation $r[n] = r[nT_s]$ indicates a discrete time sequence of the received signal $r(t)$:

$$r[n] = \sqrt{2C_s} \sum_{k=-\infty}^{\infty} \sum_{m=0}^{M-1} C_{\{k\}_N} \delta[n - m - kM - \lfloor\tau\rfloor_{T_s}] d[n - \lfloor\tau\rfloor_{T_s}] cos[2\pi(f_{IF} + f_D)n + \hat{\theta}] + w[n] \tag{2}$$

where δ is the Dirac delta function, $\lfloor\tau\rfloor_{T_s} = floor(\tau/T_s)$ is the delay of the received signal after sampling; M is the number of signal samples per PRN code chip ($MT_s \leq T_c$); $\hat{\theta}$ is the discrete time of the carrier phase sequence. However, this paper is not focusing on the carrier phase as it does not have any effect on the topic of discussion. Therefore, the symbol $\hat{\theta}$ is replaced by θ for the rest of this paper.

2.1 Residual code phase estimation

The estimation of the pseudorange, which is used to calculate user position, is closely aligned with the ability to locate the PRN code chip transition. The more precisely the transition is detected, the more accurate the user position. The residual code phase (θ_{NCO}) is exploited to improve the accuracy.

Let $\eta(m)$ denote the code NCO register contents at the m^{th} clock tick, so the NCO difference equation is:

$$\eta(m) = \{\eta(m-1) + 1/n_s\} modulus - 1 \tag{3}$$

where $n_s = f_s/f_c$, and $f_c = 1/T_c$ is the PRN chipping rate; for example, $f_c = 1.023$ MHz for GPS L1 C/A. The register content undergoes one crossing (overflow) every n_s samples on average. Whenever overflow occurs, the PRN code generator is clocked, the $(k-1)^{th}$ of the PRN code C_{k-1} is incremented to the k^{th} PRN code C_k. The remaining content of the NCO after this overflow is the instantaneous residual code phase ($\theta_{NCO}(k)$) and $0 \leq \theta_{NCO} < 1/n_s$.

The number of samples per code chip, n_s, is generally a non-integer value. Even if the number of samples per code chip is designed to be an integer value, drifts in the local oscillator and Doppler shifts cause it to become a non-integer value. Therefore, the number of received signal samples correlated with k^{th} PRN code is either $\lfloor n_s \rfloor$ or $\lfloor n_s \rfloor + 1$, where the exact number is determined by a given $\theta_{NCO}(k)$ as illustrated in Figure 2 and:

$$M = \begin{cases} \lfloor n_s \rfloor + 1 & (0 \le \theta_{NCO}(k) < \theta_A) \\ \lfloor n_s \rfloor & (\theta_A \le \theta_{NCO}(k) < 1/n_s) \end{cases} \tag{4}$$

where $\theta_A = 1 - \frac{\lfloor n_s \rfloor}{n_s}$ and $\lfloor . \rfloor$ is the floor function. Consequently, the chip-phase of each chip in fractions of a chip or the so-called residual code phase of each PRN code chip can be calculated as:

$$\theta_{NCO}(k) = \begin{cases} \theta_{NCO}(k-1) + \frac{\lfloor n_s \rfloor + 1}{n_s} - 1 & (0 \le \theta_{NCO}(k-1) < \theta_A) \\ \theta_{NCO}(k-1) + \frac{\lfloor n_s \rfloor}{n_s} - 1 & (\theta_A \le \theta_{NCO}(k-1) < 1/n_s) \end{cases} \tag{5}$$

where k is in the range $[0, NT_0 - 1]$, T_0 is the number of PRN code periods and N is the PRN code length.

3 Correlation output calculation

Assuming that the incoming baseband signal is correlated with the local replica code for T seconds, which is equal to T_0 PRN code periods and the navigation data bit $d(n - \lfloor \tau \rfloor_{T_s})$ does not change during this period. The correlation output of the incoming signal and the Prompt tap is:

$$\begin{aligned} R(\tau) =& \frac{C_s}{N_L} \sum_{n=0}^{N_L-1} \sum_{k=0}^{T_0N-1} \sum_{m=0}^{M-1} C_{\{k\}_N} \delta[n - m - kM - \lfloor \tau \rfloor_{T_s}] \\ & \times \sum_{k=0}^{T_0N-1} \sum_{m=0}^{M-1} C_{\{k\}_N} \delta[n - m - kM + \lfloor \theta_{NCO}(k) T_c \rfloor_{T_s}] \\ =& \frac{C_s}{N_L} \sum_{n=0}^{N_L-1} \sum_{k=0}^{T_0N-1} \sum_{m=0}^{M-1} C_{\{k\}_N} \delta[n - m - kM] \\ & \times \sum_{k=0}^{T_0N-1} \sum_{m=0}^{M-1} C_{\{k\}_N} \delta[n - m - kM - \lfloor \tau + \theta_{NCO}(k) T_c \rfloor_{T_s}] \end{aligned} \tag{6}$$

where C_s is the received signal power and N_L is the actual number of samples that are correlated.

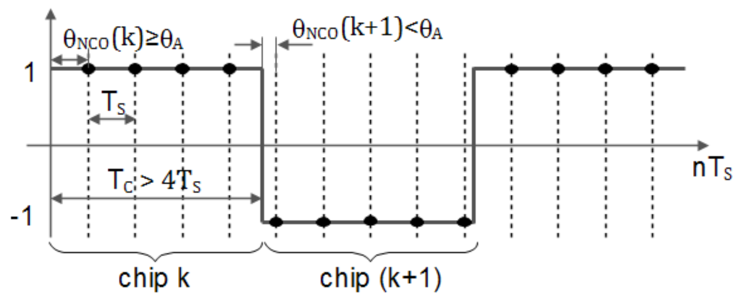

Figure 2: Effect of residual code phases on the number of samples per code chip with $4f_c < f_s < 5f_c$

Considering the local replica PRN as:

$$C_{local}(n) = \sum_{k=0}^{T_0N-1}\sum_{m=0}^{M-1} C_{\{k\}_N}\delta[n-m-kM] \tag{7}$$

Applying discrete time Fourier transform on the local replica PRN with $\omega = 2\pi f T_s$:

$$\begin{aligned} C(\omega) &= \sum_{n=-\infty}^{\infty}\sum_{k=0}^{T_0N-1}\sum_{m=0}^{M-1} C_{\{k\}_N}\delta[n-m-kM]e^{-i\omega n} \\ &= \sum_{k=0}^{T_0N-1} C_{\{k\}_N}e^{-i\omega kM}\sum_{m=0}^{M-1}e^{-i\omega m}\sum_{n=-\infty}^{\infty}\delta[n] \\ &= \frac{sin(M\omega/2)}{sin(\omega/2)}e^{-i\frac{(M+1)\omega}{2}}\sum_{k=0}^{T_0N-1} C_{\{k\}_N}e^{-i\omega kM} \end{aligned} \tag{8}$$

so it can be illustrated in frequency domain as:

$$C(f) = \frac{sin(M\pi f T_s)}{sin(\pi f T_s)}e^{-i(M+1)\pi f T_s}\sum_{k=0}^{T_0N-1} C_{\{k\}_N}e^{-i2\pi f T_s kM} \tag{9}$$

and its reverse Fourier transform is:

$$\sum_{k=0}^{T_0N-1}\sum_{m=0}^{M-1} C_k\delta[n-m-kM] = T_s\int_{-\beta_r/2}^{\beta_r/2} C(f)e^{i2\pi f T_s n}df \tag{10}$$

where β_r is the receiver front-end bandwidth.
Replacing (10) to (6), it yields:

$$\begin{aligned} R(\tau) &= \frac{C_sT_s}{N_L}\sum_{n=0}^{N_L-1}\sum_{k=0}^{T_0N-1}\sum_{m=0}^{M-1} C_{\{k\}_N}\delta[n-m-kM-\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s}]\int_{-\beta_r/2}^{\beta_r/2} C(f)e^{i2\pi f T_s n}df \\ &= \frac{C_sT_s}{N_L}\int_{-\beta_r/2}^{\beta_r/2} C(f)\sum_{n=0}^{N_L-1}\sum_{k=0}^{T_0N-1}\sum_{m=0}^{M-1} C_{\{k\}_N}\delta[n-m-kM-\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s}]e^{i2\pi f T_s n}df \\ &= \frac{C_sT_s}{N_L}\int_{-\beta_r/2}^{\beta_r/2} C(f)\sum_{m=0}^{M-1}e^{i2\pi f T_s m}\sum_{k=0}^{T_0N-1} C_{\{k\}_N}e^{i2\pi f T_s(kM+\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s})}df \\ &= \frac{C_sT_s}{N_L}\int_{-\beta_r/2}^{\beta_r/2}\left(\frac{sin(\pi f M T_s)}{sin(\pi f T_s)}\right)^2\sum_{l=0}^{T_0N-1} C_{\{l\}_N}e^{-i2\pi f T_s lM}\sum_{k=0}^{T_0N-1} C_{\{k\}_N}e^{i2\pi f T_s(kM+\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s})}df \\ &= \frac{C_sT_s}{N_L}\int_{-\beta_r/2}^{\beta_r/2}\left(\frac{sin(\pi f M T_s)}{sin(\pi f T_s)}\right)^2\left(\sum_{k=0}^{T_0N-1}|C_{\{k\}_N}|^2e^{i2\pi f T_s(\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s})}\right. \\ &\qquad\left. + \sum_{k=0}^{T_0N-1}\sum_{l=0,l\neq k}^{T_0N-1} C_{\{k\}_N}C_{\{l\}_N}e^{i2\pi f T_s((k-l)M+\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s})}\right)df \\ &= \frac{C_s}{N_L}\sum_{k=0}^{T_0N-1}\int_{-\beta_r/2}^{\beta_r/2} T_s\left(\frac{\sin(\pi f M T_s)}{\sin(\pi f T_s)}\right)^2 e^{i2\pi f T_s(\lfloor\tau+\theta_{NCO}(k)T_c\rfloor_{T_s})}df \end{aligned} \tag{11}$$

because for a pseudorandom noise code sequence C_k, k=0, 1, 2, ..., N, all misaligned chips with $k \neq l$ have equal probability of having $+1$ or -1 values. Their product sum approaches zero. Noted that $-T_c \leq (\tau + \theta_{NCO}(k)T_c) \leq T_c$.

Denoting $G_s(f) = T_s\left(\frac{\sin(\pi f M T_s)}{\sin(\pi f T_s)}\right)^2$, the PSD normalised to unit power of the sampling PRN code is:

$$G_s(f)_{PSD} = \frac{T_s}{N_L}\sum_{k=0}^{T_0 N-1}\left(\frac{\sin(\pi f M T_s)}{\sin(\pi f T_s)}\right)^2 e^{i2\pi f T_s(\lfloor\theta_{NCO}(k)T_c\rfloor_{T_s})} \tag{12}$$

It is obvious that $\theta_{NCO}(k)T_c < T_s$ thus $\lfloor\theta_{NCO}(k)T_c\rfloor_{T_s} = 0$ and:

$$G_s(f)_{PSD} = \frac{T_0 N M}{N_L} \times \frac{M T_s \sin^2(\pi f M T_s)}{M^2 \sin^2(\pi f T_s)} \tag{13}$$

observe that $\lim_{f_s\to\infty} T_o N M = N_L$, $\lim_{f_s\to\infty} M T_s = T_c$ and $\lim_{f_s\to\infty} \sin(\pi f T_s) = \pi f T_s$ so:

$$\begin{aligned}\lim_{f_s\to\infty} G_s(f)_{PSD} &= \lim_{f_s\to\infty}\left(\frac{M T_s \sin^2(\pi f T_c)}{M^2 \sin^2(\pi f T_s)}\right)\\ &= T_c sinc^2(\pi f T_c)\end{aligned} \tag{14}$$

This is the well-known PSD developed by the authors in (Parkinson, 1996). The above equation means that this widely applied PSD equation is only correct when the sampling frequency is much higher than the PRN code rate.

4 Coherent Early minus Late DLL error analysis

4.1 Theoretical code tracking loop error estimate

In this paper, the code tracking loop uses the model that was proposed in (Betz and Kolodziejski, 2000, 2009a,b) and is re-illustrated in Figure 3. The received signal plus noise enters a time-of-arrival (TOA) estimator. The previous estimate of the signals TOA is also provided to the TOA estimator. The TOA estimator uses an integration time of T seconds to produce an unsmoothed TOA estimate. This estimate is the update to the previous TOA estimate based on the received signal plus noise. Unsmoothed TOA estimates are processed by a code-tracking loop that acts like a TOA smoothing filter, producing smoothed TOA estimates that are then provided to the TOA estimator as previous TOA estimates. The variance of the unsmoothed TOA estimate from the discriminator is σ_u^2, and the variance of the smoothed TOA estimate at the output of the code tracking loop (Betz and Kolodziejski, 2009b), denoted as σ_s^2, is :

$$\sigma_s^2 \cong \sigma_u^2 2B_L T(1 - 0.5B_L T) \tag{15}$$

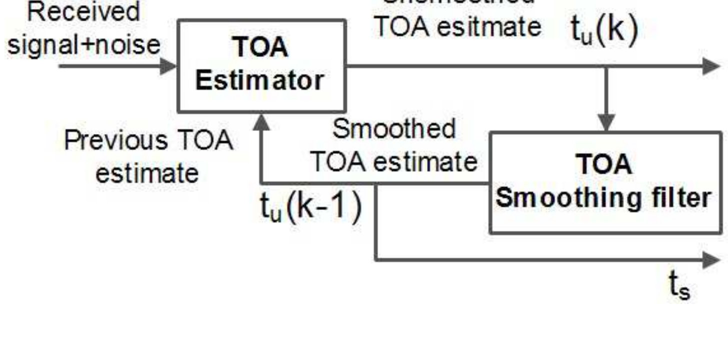

Figure 3: Representation of code tracking loop Betz and Kolodziejski (2009b)

where B_L is the noise bandwidth of the DLL tracking loop, T is the integration time, and $0 < B_L T \leq 0.5$.

Nevertheless, this estimate does not accurately consider the effect of the sampling frequency. Therefore, it was re-estimated in (Tran et al., 2016b) and the error is broken into two components, the part that is influenced by the sampling frequency, σ_1^2, and the part that is effected by the code lock loop noise filter, σ_{uN}^2, where:

$$\sigma_1^2 = \frac{\left(\sum_{k=0}^{T_0 N-1} \frac{1}{N_L} \int_{-\beta_r/2}^{\beta_r/2} G_s(f) \sin(\pi f \Delta) \sin(\pi f (2\theta_{NCO}(k) T_c - T_s)) \cos(\pi f T_s) df\right)^2}{\left(\sum_{k=0}^{T_0 N-1} \frac{2\pi}{N_L} \int_{-\beta_r/2}^{\beta_r/2} f G_s(f) \sin(\pi f \Delta) \cos(2\pi f \theta_{NCO}(k) T_c) df\right)^2} \tag{16}$$

$$\sigma_{uN}^2 = \frac{\frac{N_0 \beta_r}{2 C_s N_L^2} \sum_{k=0}^{T_0 N-1} \int_{-\beta_r/2}^{\beta_r/2} G_s(f) \sin^2(\pi f \Delta) df}{\left(\sum_{k=0}^{T_0 N-1} \frac{2\pi}{N_L} \int_{-\beta_r/2}^{\beta_r/2} f G_s(f) \sin(\pi f \Delta) \cos(2\pi f \theta_{NCO}(k) T_c) df\right)^2} \tag{17}$$

with Δ is the Early - Late code space, is usually one code chip for BPSK signal $\Delta = T_c$.

It is clear that σ_1^2 is constant during tracking. The DLL noise filter, therefore, has no effect on these values. Only the error contributed by noise σ_{u1N}^2 is filtered by the loop tracking filter. Consequently, the 1-sigma code tracking jitter (in metres) is:

$$\sigma_s = C\sqrt{\sigma_1^2 + \sigma_{uN}^2 2 B_L T (1 - 0.5 B_L T)} \tag{18}$$

with C is the speed of light.

4.2 Effect of the front-end filter bandwidth and the received C/N_0

According to (Kaplan and Hegarty, 2005; Betz, 2015), the received carrier to noise ratio C/N_0 has a strong influence on the DLL tracking error. Nevertheless, the effect of the sampling frequency and the front-end ADC filter bandwidth was not considered. Those results are revised in this section by evaluating results obtained by (18).

Figure 4 presents the DLL jitter versus different sampling frequencies for various C/N_0 when the two sided bandwidth is fixed $\beta_r = 2 f_c$ (for example $\beta_r = 2.046$ MHz for GPS L1 C/A signal). Observe that the 1-sigma DLL tracking error is high when the received signal is weak ($C/N_0 = 10, 15, 20$ dB-Hz). Whereas, it is insignificant when the received C/N_0 is in the commonly received C/N_0 range of a receiver (from 25 dB-Hz to 45 dB-Hz). It is equivalent to results presented in (Kaplan and Hegarty, 2005). The other trend which can be observed is that the DLL jitter decreases when the sampling frequency increases, because when more samples are used to represent the incoming and local generated signals, the more accuracy the code chip transition can detect. The tracking error is thus reduced as a result. However, the DLL jitter is still significant at some peak point due to the sampling frequency. It is considerable for every received C/N_0, but it is most significant when the received signal is strong ($C/N_0 > 30$ dB-Hz). Moreover, this DLL jitter is not only significant when the sampling rate (f_s) is an integer multiple of the nominal PRN code chipping rate (f_c), but it is also considerable when the ratio

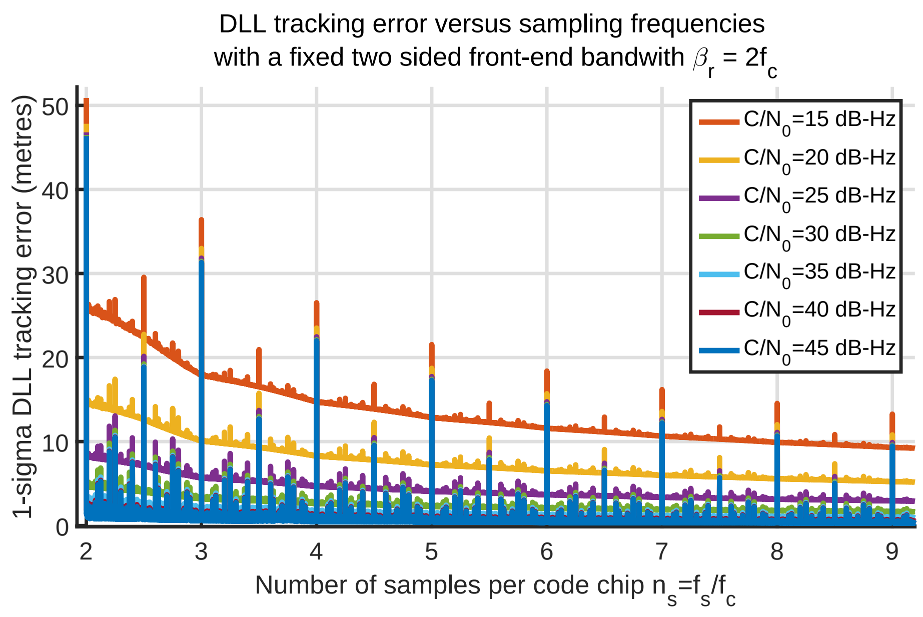

Figure 4: DLL tracking error versus different sampling frequencies (step=$10^{-3} f_c$) with a fixed front-end bandwidth $\beta_r = 2 f_c$ for GPS L1 C/A signal with B_L=0.5 Hz and T= 1ms

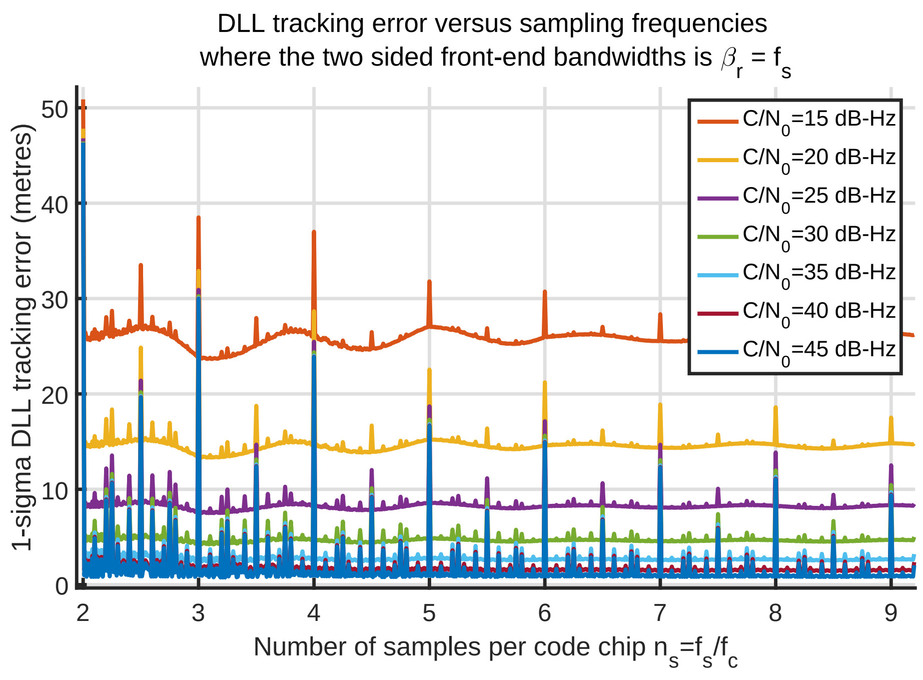

Figure 5: DLL tracking error versus different sampling frequencies (step=$10^{-3} f_c$) with front-end bandwidths $\beta_r = f_s$ for GPS L1 C/A signal with B_L=0.5 Hz and T= 1ms

f_s/f_c is rational. Therefore, the effect of the sampling frequency should be carefully taken into account when choosing the front-end ADC sampling rate.

The other aspect influencing the DLL jitter is the front-end bandwidth (β_r). The DLL tracking errors versus sampling frequency with different two sided front-end bandwidths $\beta_r = 2f_c$ and $\beta_r = f_s$ are respectively illustrated in Figure 4 and Figure 5. Observe that the DLL jitter is decreased when the sampling frequency is increased, if the front-end bandwidth is fixed, $\beta_r = 2f_c$, as shown in Figure 4; because the noise power is only governed by the front-end bandwidth, and the coherent integration time, but it is not affected by the sampling frequency, as shown in (17). The DLL error contributed by the noise power is thus the same while the sampling frequency is increasing. Consequently, the DLL jitter, excluding the peak impacted by the sampling frequency, is reduced while the sampling frequency is higher. On the other hand, the DLL tracking error, which is controlled only by the noise power, is approximately constant if the front-end bandwidth is enlarged according to the sampling frequency, $\beta_r = f_s$, as shown in Figure 5. The wider the front-end bandwidth, the more signal power received. However the received signal also includes the AWGN, thus, the noise power is also increased. Consequently, the DLL tracking noise error fluctuates slightly, but it is approximately unchanged in general, because the DLL jitter decrease, which occurs by increasing the sampling frequency and front-end bandwidth, is approximately offset by the error increase controlled by the growth of the noise power. The DLL jitter, therefore, becomes significant when the received signal power is weak ($C/N_0 < 30$ dB-Hz).

4.3 Experiment results

The previous sections describe analysis results based on the proposed theoretical coherent EML DLL tracking error. Experiments were also setup to examine these results. The bladeRF front-

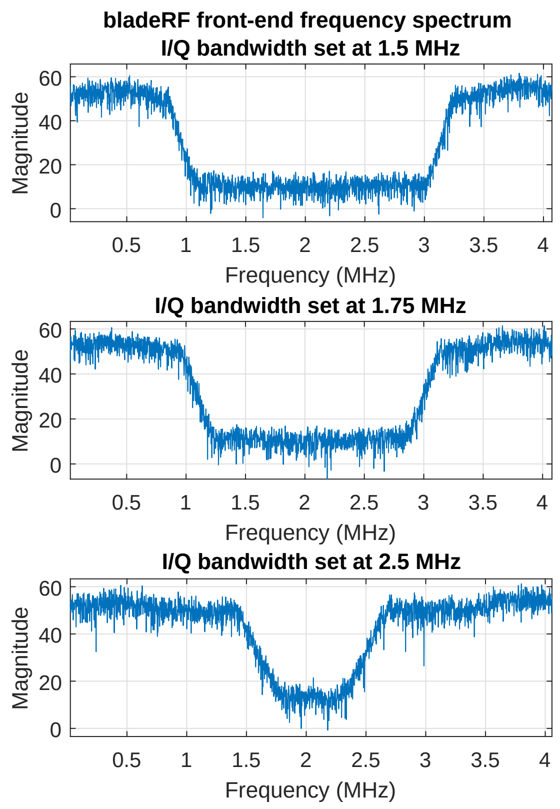

Figure 6: Real complex bladeRF bandwidth versus different I/Q bandwidth settings

Table 1: DLL tracking error (in metres) versus different two sided front-end bandwidths. Real GPS L1 C/A signal is recorded by the bladeRF front-end and processed with DLL tracking parameters B_L = 0.5 Hz, T= 1ms.

C/N_0	Two sided bandwidth (β_r)		
(PRN)	2.2 MHz	2.7 MHz	3.6 MHz
51 dB-Hz (8)	15.51	16.36	18.13
45 dB-Hz (7)	16.55	18.08	19.76
42 dB-Hz (1)	22.97	26.25	25.48
40 dB-Hz (16)	25.74	27.63	29.15
35 dB-Hz (3)	27.33	29.63	31.34

end was used to collect live GPS L1 C/A signal data from an active roof antenna. The sampling frequency was chosen at $f_s = 4.092$ MHz ($n_s = 4$), and signal was processed with various In-phase (I) and Quadrature (Q) front-end bandwidths 1.5, 1.75 and 2.5 MHz, equivalent to actual complex front-end bandwidths $\beta_r \approx 2.2$, 2.7, and 3.6 MHz, respectively, as observed in Figure 6. The DLL tracking error was calculated using the variance of 10,000 DLL discriminator output instances with coherent integration time T=1ms and noise bandwidth $B_L = 0.5$ Hz. The uBlox 6T also simultaneously processed the received signal to estimate the C/N_0 of each satellite. It is obvious that the experiment result, as presented in Table 1, supports the previous theoretical analysis. The narrower the front-end bandwidth, the smaller the DLL tracking error.

All in all, the effect of front-end bandwidth also needs to be considered as well as the effect of the sampling frequency and the received C/N_0. Increasing the sampling frequency helps to reduce the DLL jitter, and it is more effective for weak signals if and only if the front-end bandwidth is fixed and much smaller than the sampling frequency.

5 Conclusion

The more accurate correlation output and Delay Locked Loop (DLL) code tracking error are described in this paper from the hardware receiver perspective. Estimation is based on the number of samples per code chip and the residual code phase of the code NCO after a code chip is generated. The theoretical as well as experiment results show that the relation between the sampling frequency and front-end filter bandwidth has a strong effect on the DLL jitter. It is obvious that the DLL tracking error is decreased while the sampling frequency is increased and it is more efficient for weak signals ($CN_0 < 30$ dB-Hz) if and only if the the front-end bandwidth is fixed and much smaller than the sampling frequency. The accurate estimation of the correlation output and the DLL tracking error can generally apply on precisely modeling GNSS receiver baseband signal processing.

References

Dennis M Akos and Michael S Braasch. A software radio approach to global navigation satellite system receiver design. In *Proceedings of the 52nd Annual Meeting of The Institute of Navigation*, pages 455–463, 1996.

John W Betz. *Engineering Satellite-Based Navigation and Timing: Global Navigation Satellite Systems, Signals, and Receivers*. John Wiley & Sons, 2015. doi: 10.1002/9781119141167.

URL http://onlinelibrary.wiley.com/book/10.1002/9781119141167.

John W Betz and Kevin R Kolodziejski. Extended theory of early-late code tracking for a bandlimited gps receiver. *Navigation*, 47(3):211–226, 2000.

John W Betz and Kevin R Kolodziejski. Generalized theory of code tracking with an early-late discriminator part ii: noncoherent processing and numerical results. *IEEE Transactions on Aerospace and Electronic Systems*, 45(4):1557–1564, 2009a.

J.W. Betz and K.R. Kolodziejski. Generalized theory of code tracking with an early-late discriminator part i: Lower bound and coherent processing. *Aerospace and Electronic Systems, IEEE Transactions on*, 45(4):1538–1556, Oct 2009b. ISSN 0018-9251. doi: 10.1109/TAES.2009.5310316.

Jaegyu Jang, Matteo Paonni, and Bernd Eissfeller. Cw interference effects on tracking performance of GNSS receivers. *IEEE Transactions on Aerospace and Electronic Systems*, 48(1): 243–258, 2012.

Elliott D Kaplan and Christopher J Hegarty. *Understanding GPS: principles and applications*. Artech house, 2005.

Y. Kou and Y. Morton. Oscillator frequency offset impact on software gps receivers and correction algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 49(4):2158–2178, OCTOBER 2013. ISSN 0018-9251. doi: 10.1109/TAES.2013.6621808.

Bradford W Parkinson. *Global Positioning System: Theory and Applications*, volume 1. American Institute of Aeronautics and Astronautics, 1996.

Vinh T. Tran, Nagaraj C. Shivaramaiah, and Andrew G. Dempster. Feasibility analysis of baseband architectures for multi-GNSS receivers. *GPS Solution*, 2016a. doi: 10.1007/s10291-016-0542-0. URL http://link.springer.com/article/10.1007/s10291-016-0542-0.

Vinh T. Tran, Nagaraj C. Shivaramaiah, Thuan D Nguyen, Eamonn Glenoon, W Cheong Joon, and Andrew G. Dempster. A generalized theory of the effect of sampling frequency on GNSS code tracking. *Submitted to Aerospace and Electronic Systems, IEEE Transactions on*, 2016b.