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

Vinh Tran, Thuan Nguyen, Nagaraj Shivaramaiah, Andrew Dempster

Never Stand Still | Faculty of Engineering | Australian Centre for Space Engineering Research (ACSER)

Contents

- GNSS receiver model
- Local code generator and residual code phase estimation
- Problem and motivation
- Correlator output recalculation
- DLL tracking error estimation
- Experiment results

GNSS receiver model

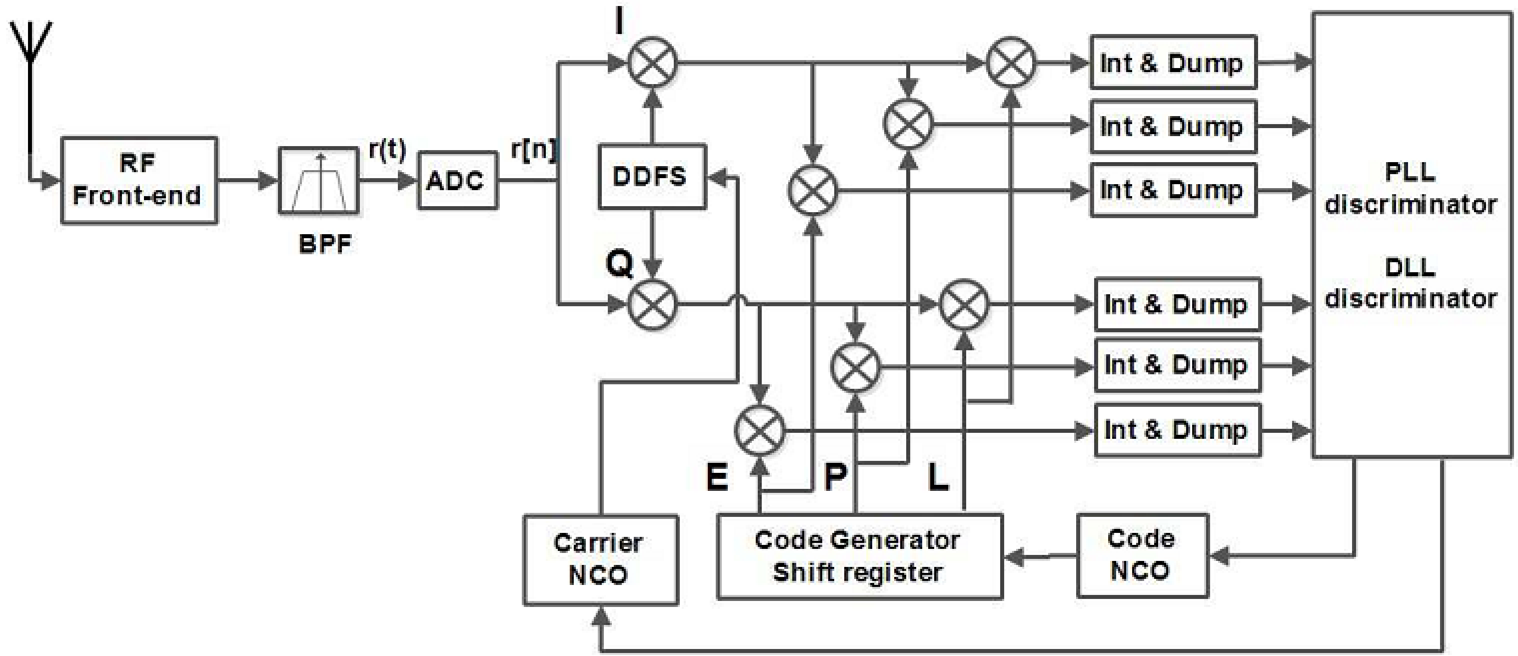
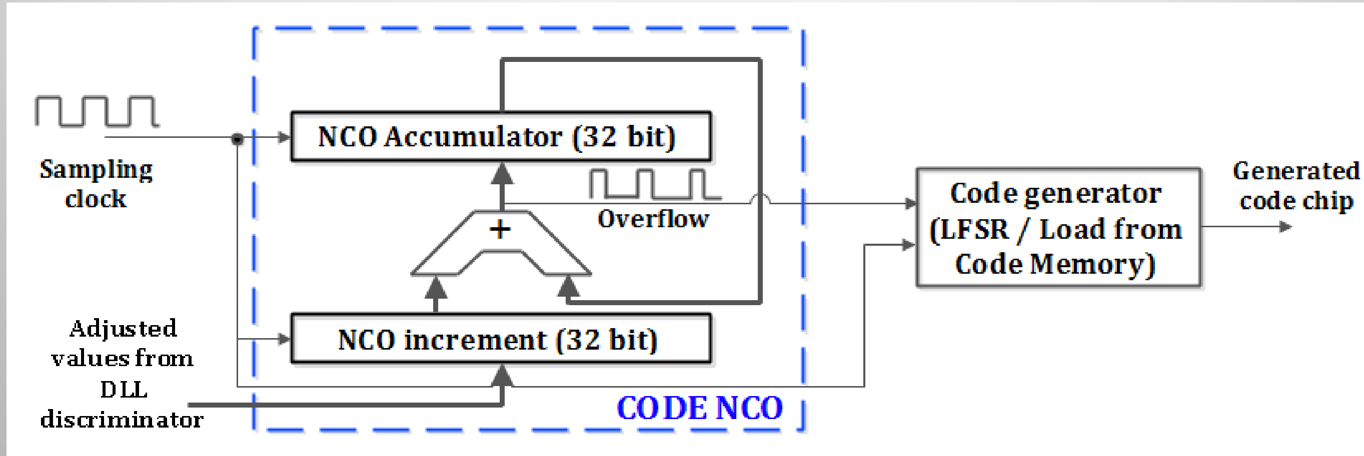
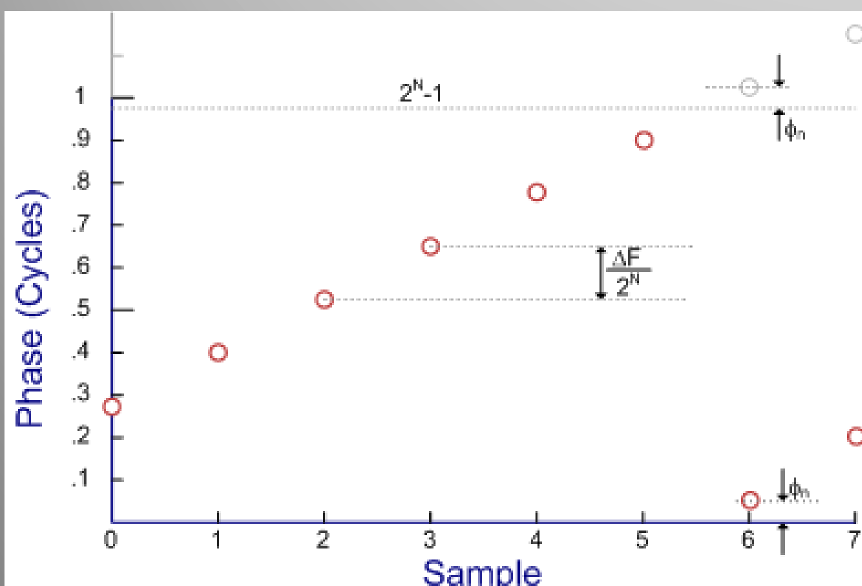
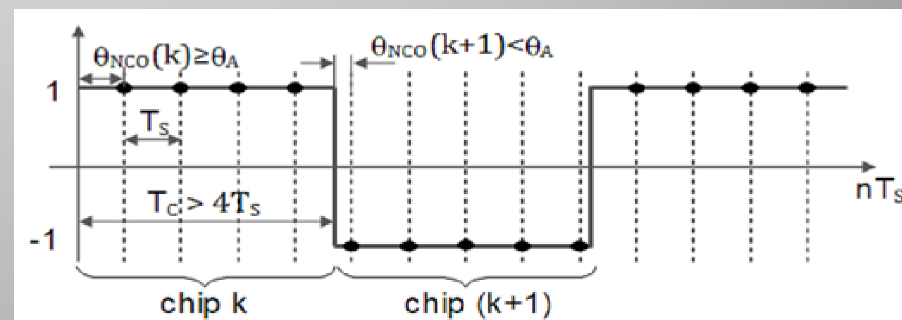


Figure 1: Block diagram of a digital GNSS receiver

Code generator and residual code phase estimation





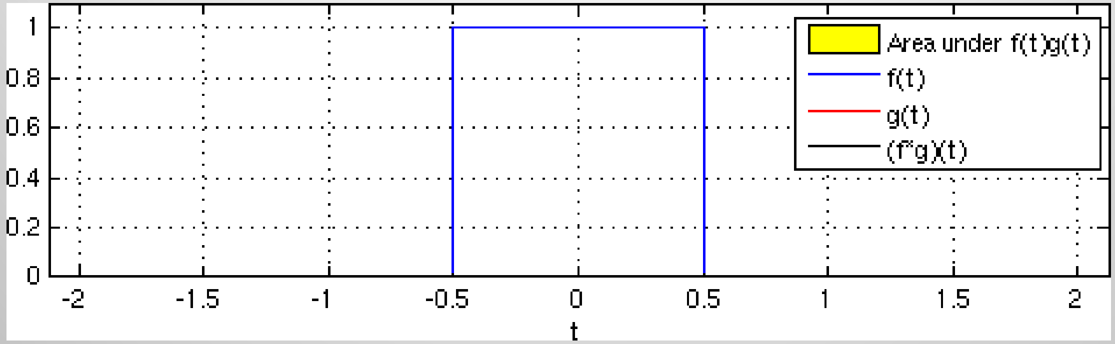


the number of received signal samples correlated with k^{th} PRN code is

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

where $\theta_A = 1 - \frac{\lfloor n_s \rfloor}{n_s}$ and $\lfloor . \rfloor$ is the floor function.

Conventional correlator output calculation



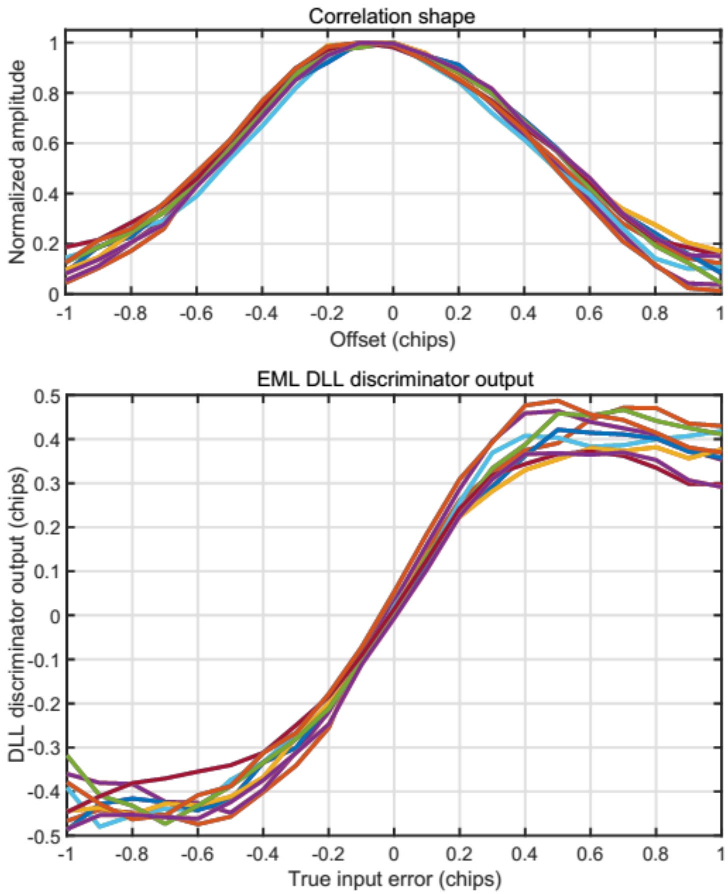
- Autocorrelation output R(τ) = $\begin{cases} 1 - \frac{\tau}{T_c} & |\tau| \le T_c \\ 0 & else\ where \end{cases}$
- Power spectral density

$$G_s(f)_{PSD} = T_c sinc^2(\pi f T_c)$$

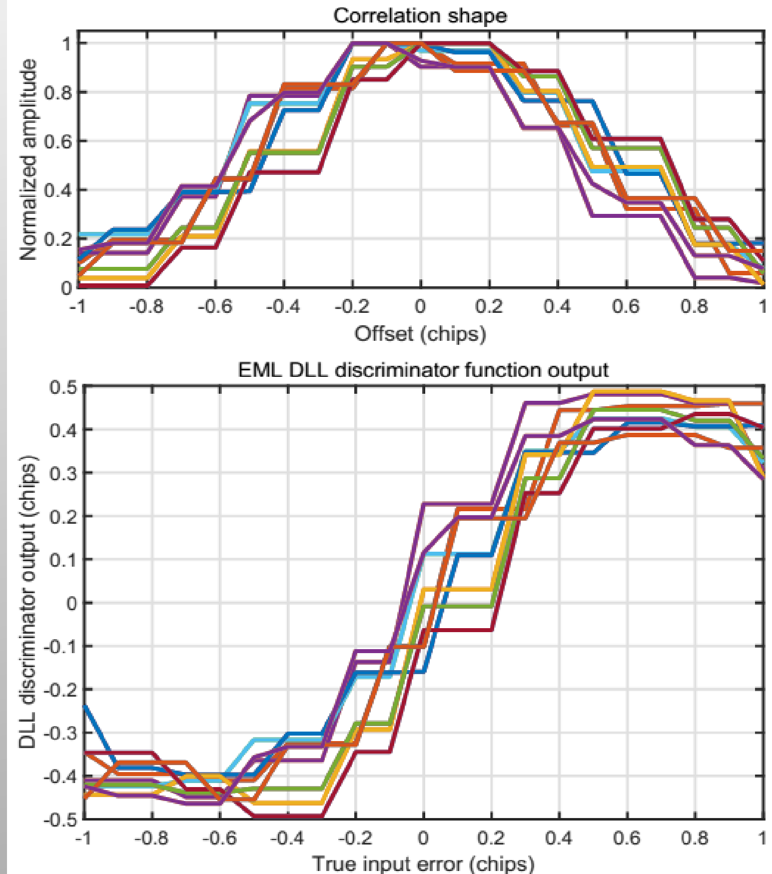
- Correlation output:

$$R(\tau) = \int_{-\beta_r/2}^{\beta_r/2} G_s(f)_{PSD} e^{j2\pi f \tau} df$$

Problems

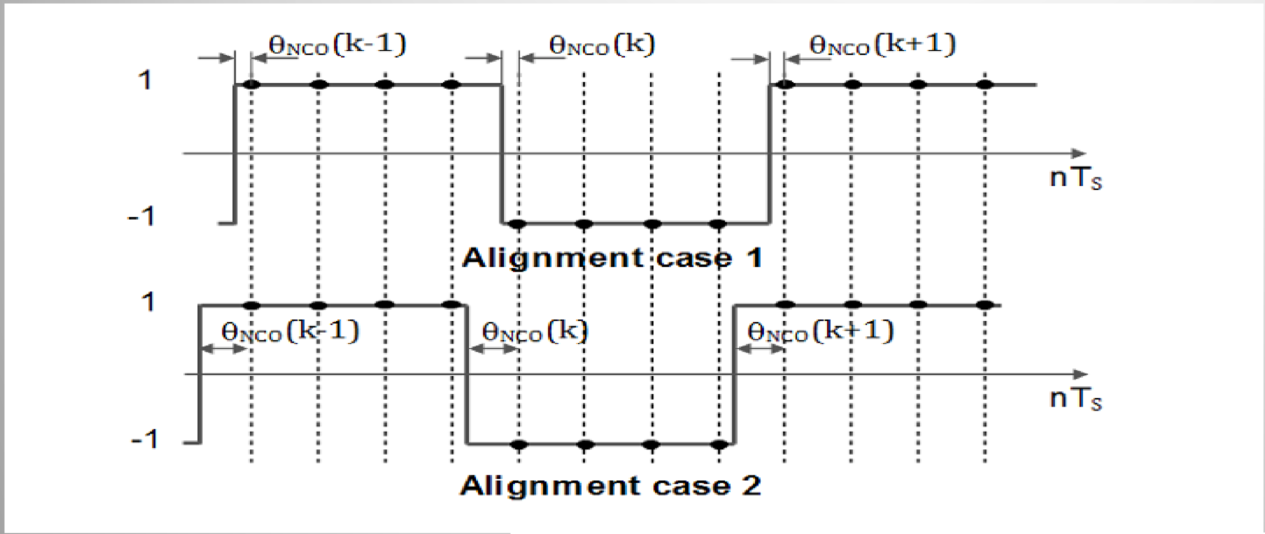


(b) f_s = 4.093 MHz



(a) f_s = 4.092 MHz

Problems



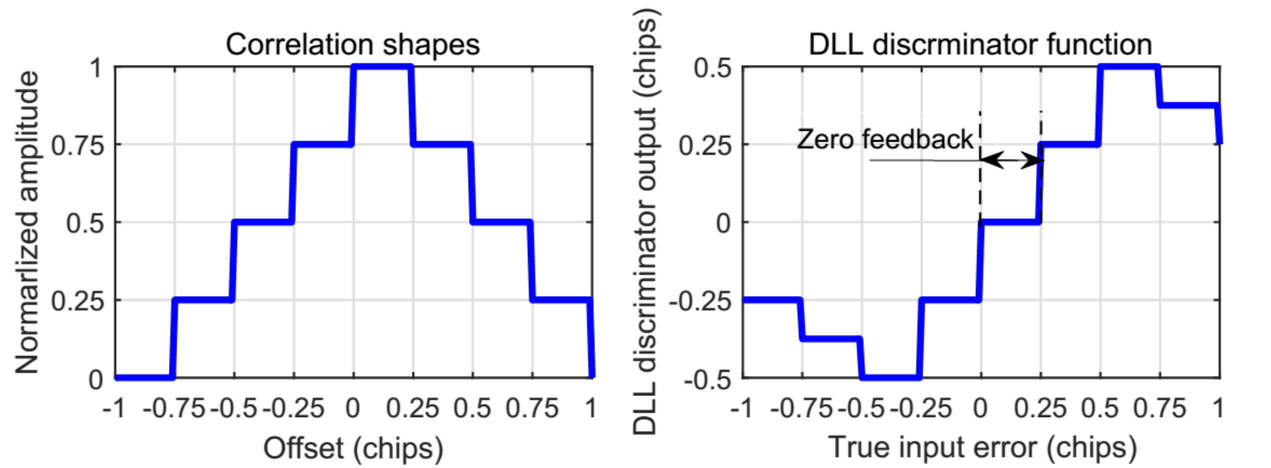


Fig. 3: Correlation shape and DLL discriminator function when $f_s = 4f_c$

Correlation output recalculation

$$R(\tau) = \frac{C_s}{N_L} \sum_{k=0}^{T_0 N - 1} \int_{-\beta_r/2}^{\beta_r/2} T_s \left(\frac{\sin(\pi f M T_s)}{\sin(\pi f T_s)} \right)^2 e^{i2\pi f T_s (\lfloor \tau + \theta_{NCO}(k) T_c \rfloor_{T_s})} df$$

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

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

Correlation shapes

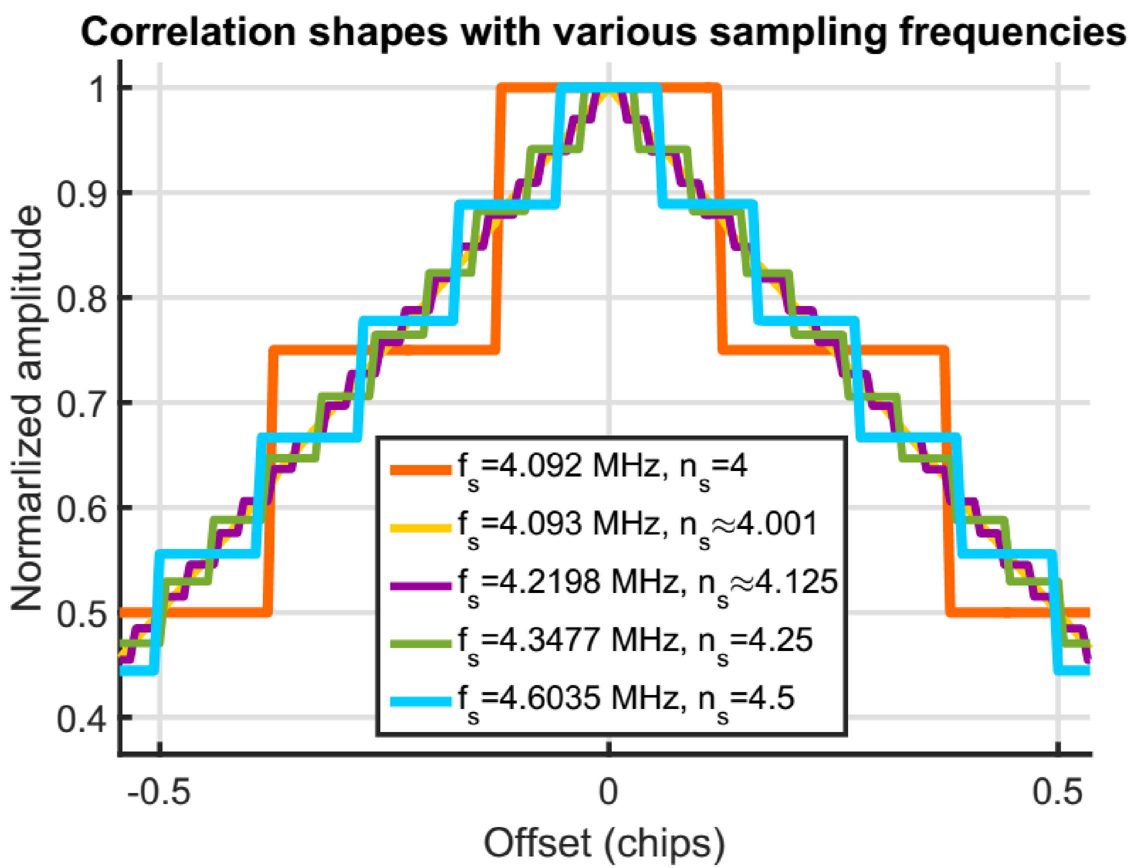
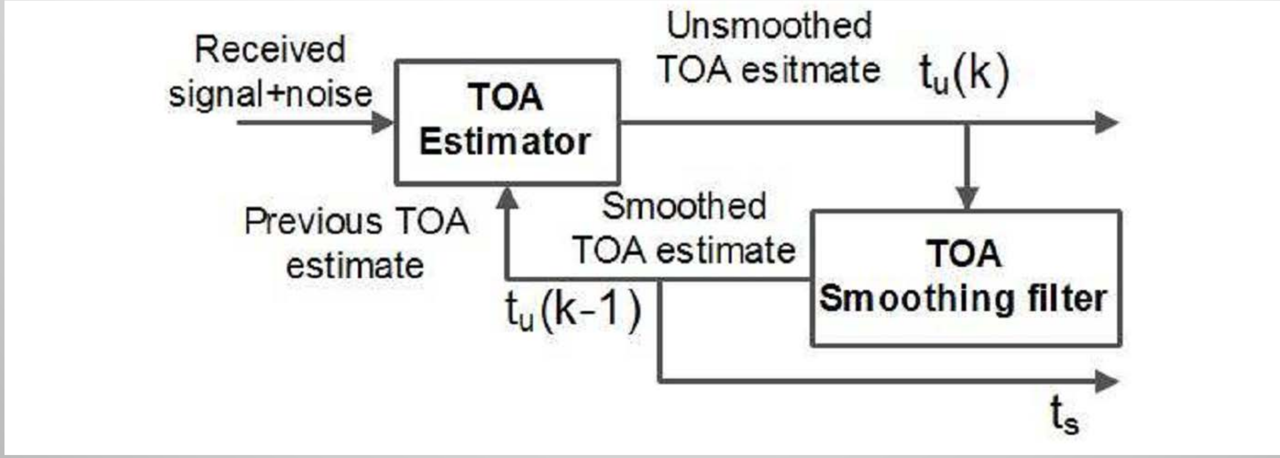


Fig. 4: Correlation shapes with various sampling frequencies

Conventional DLL tracking error estimation



DLL tracking error [1]:

$$\sigma_s^2 = \sigma_u^2 2B_L T(1 - 0.5B_L T)$$

where:

σ_u is variance of the unsmoothed TOA estimate from the discriminator

σ_s is variance of the smoothed TOA estimate from the discriminator

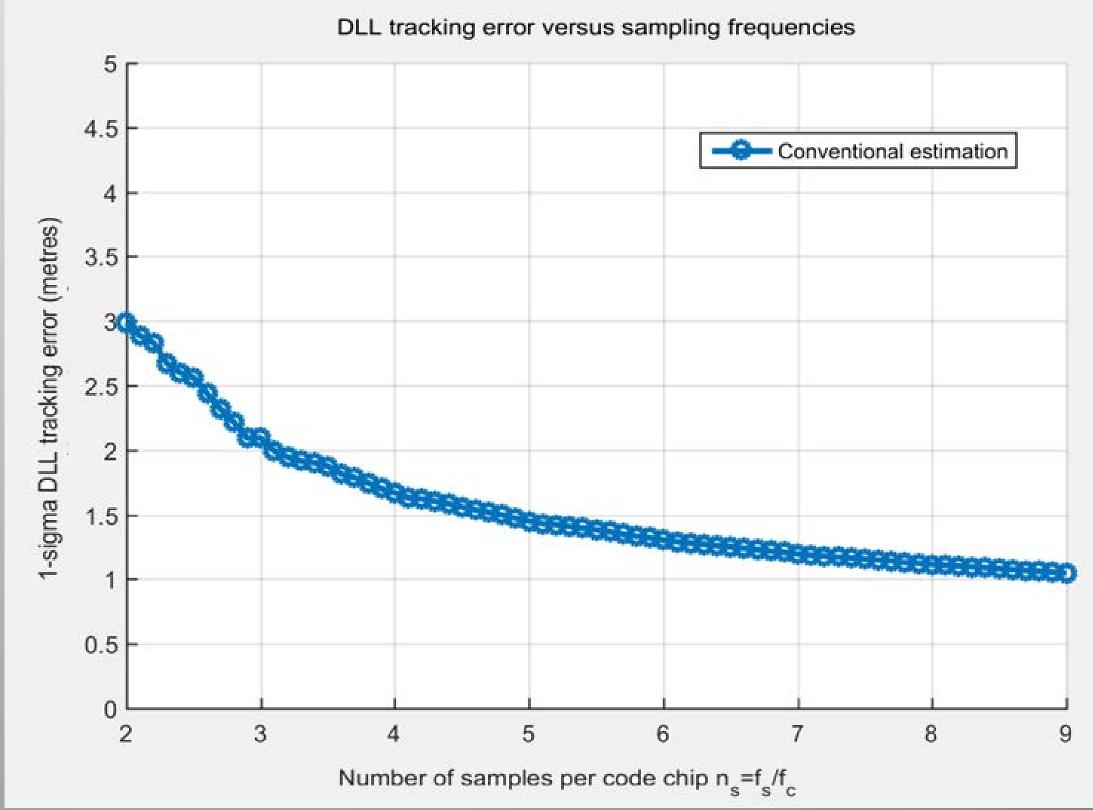
B_L is the noise filter bandwidth

T is the integration time

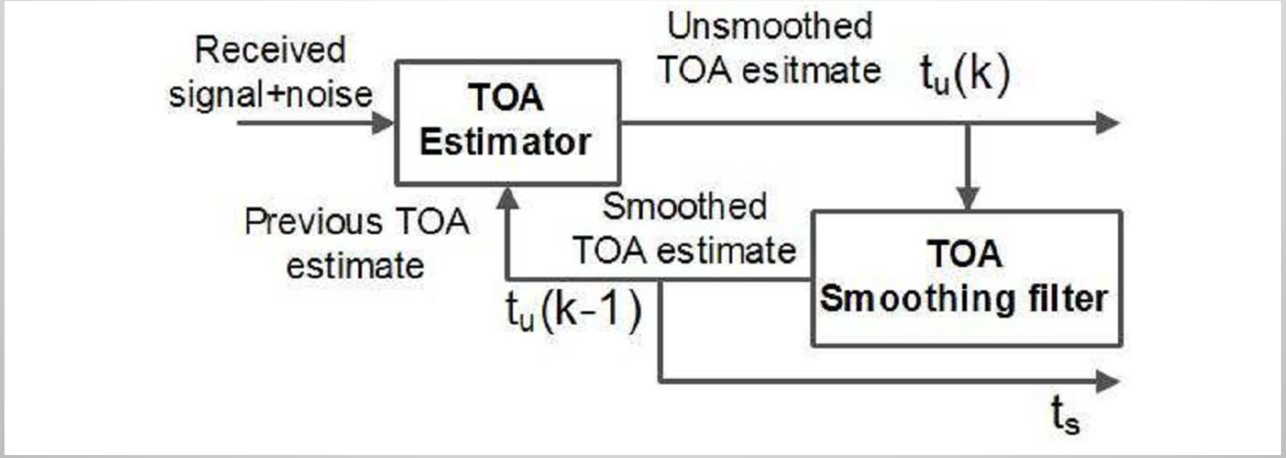
[1] John W Betz and Kevin R Kolodziejski. Generalized theory of code tracking with an early-late discriminator part ii: noncoherent processing and numerical results. IEEE Transactions onAerospace and Electronic Systems, 45(4):1557–1564, 2009a

Conventional DLL tracking error estimation

$$\sigma^2_{\text{CELP, white}} = \frac{B_L(1-0.5B_LT)\int_{-\beta_r/2}^{\beta_r/2} G_s(f)\sin^2(\pi f\Delta)df}{(2\pi)^2\frac{C_s}{N_0}\left(\int_{-\beta_r/2}^{\beta_r/2} fG_s(f)\sin(\pi f\Delta)df\right)^2}$$



DLL tracking error re-estimation



- DLL tracking error [2]:

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

where:

σ_{uN} is the error part that is effected by the noise power

σ_1 is the error part that is influenced by the sampling frequency

σ_s is variance of the smoothed TOA estimate from the discriminator

[2] 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 GNSScode tracking. Submitted to Aerospace and Electronic Systems, IEEE Transactions on, 2016b

Errors estimation

$$\sigma_1^2 = \frac{\left(\sum_{k=0}^{T_0N-1}\frac{1}{N_L}\int_{-\beta_r/2}^{\beta_r/2} G_s(f)\sin(\pi f\Delta)\sin(\pi f(2\theta_{NCO}(k)T_c - T_s))\cos(\pi f T_s)df\right)^2}{\left(\sum_{k=0}^{T_0N-1}\frac{2\pi}{N_L}\int_{-\beta_r/2}^{\beta_r/2} fG_s(f)\sin(\pi f\Delta)\cos(2\pi f\theta_{NCO}(k)T_c)df\right)^2}$$

$$\sigma_{uN}^2 = \frac{\frac{N_0\beta_r}{2C_sN_L^2}\sum_{k=0}^{T_0N-1}\int_{-\beta_r/2}^{\beta_r/2} G_s(f)\sin^2(\pi f\Delta)df}{\left(\sum_{k=0}^{T_0N-1}\frac{2\pi}{N_L}\int_{-\beta_r/2}^{\beta_r/2} fG_s(f)\sin(\pi f\Delta)\cos(2\pi f\theta_{NCO}(k)T_c)df\right)^2}$$

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

$$G_s(f) = T_s\left(\frac{\sin(\pi f M T_s)}{\sin(\pi f T_s)}\right)^2$$

DLL tracking error validation

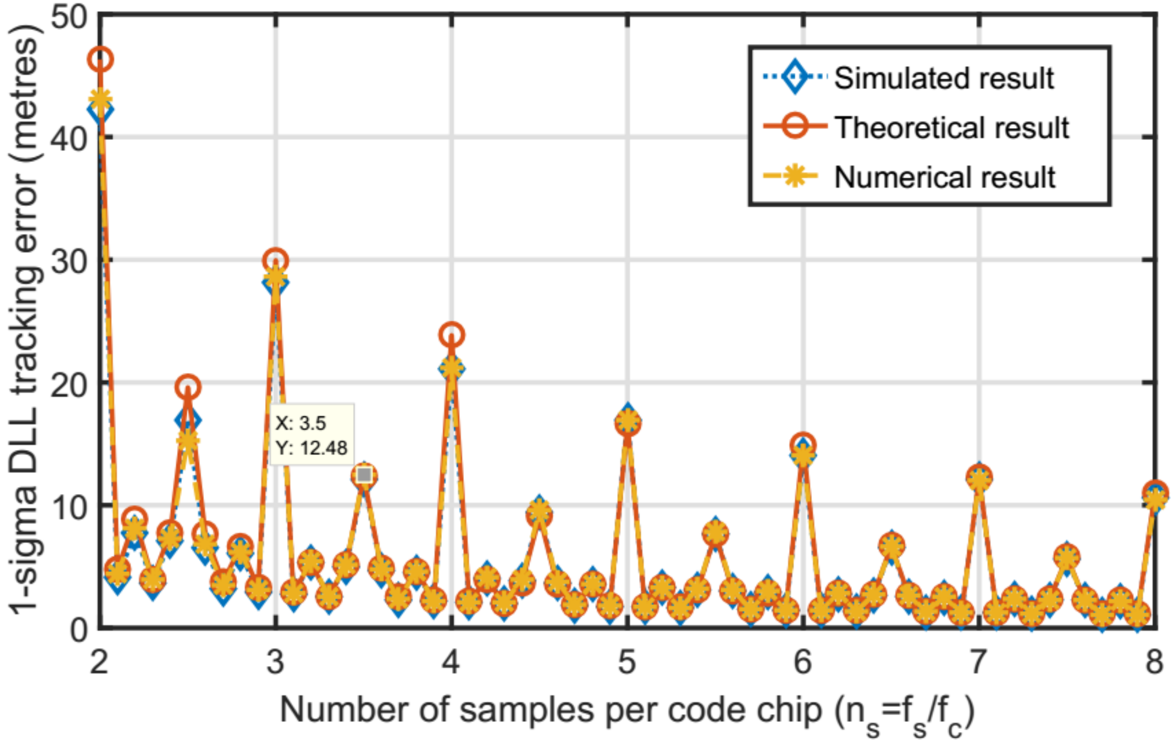
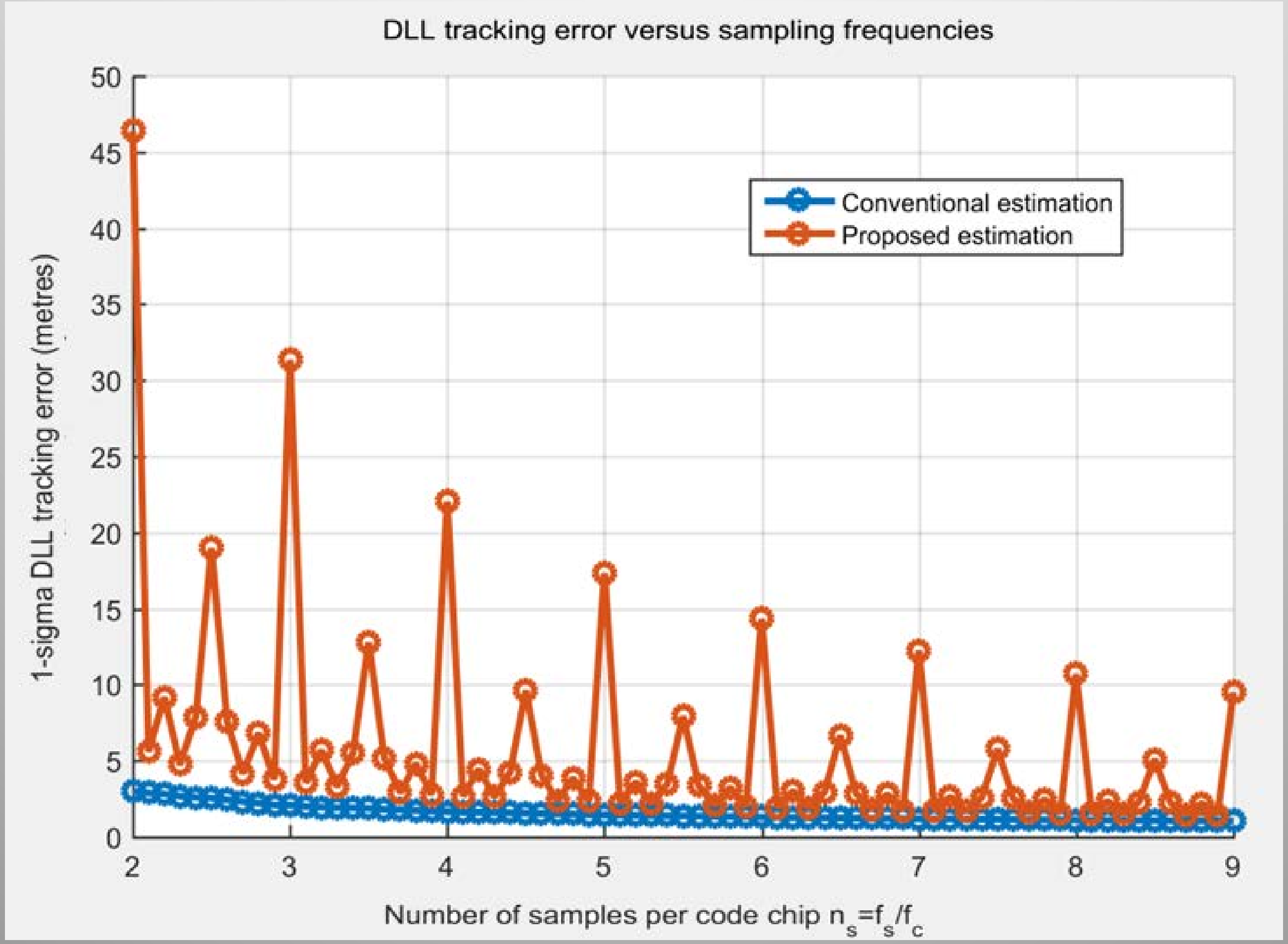


Fig. 11: DLL tracking error comparison among the simulated, numerical and theoretical models (step = $10^{-1}f_c$). GPS L1 C/A is used with T=1 ms, and $\beta_r = f_s$.

DLL tracking error re-estimation



DLL tracking error versus sampling frequency and frontend band width

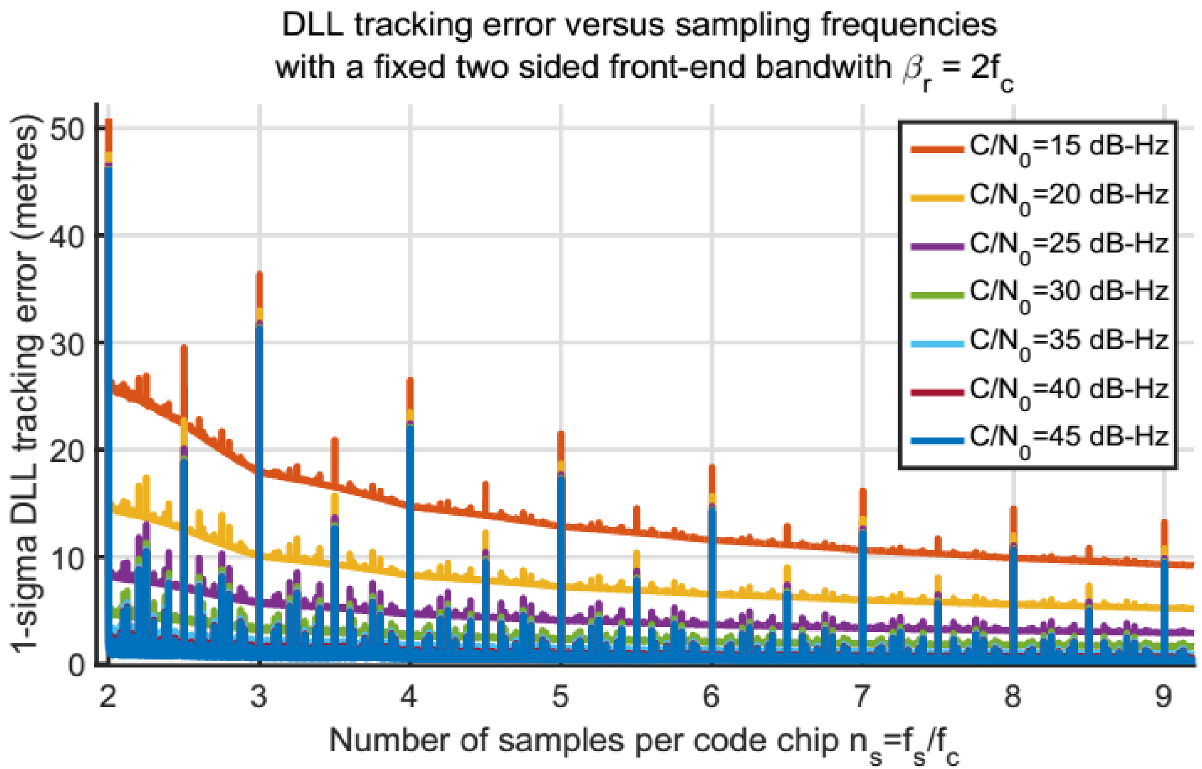


Figure 4: DLL tracking error versus different sampling frequencies (step=$10^{-3} f_c$) with a fixed front-end bandwidth $\beta_r = 2f_c$ for GPS L1 C/A signal with B_L=0.5 Hz and T= 1ms

DLL tracking error versus sampling frequency and frontend band width

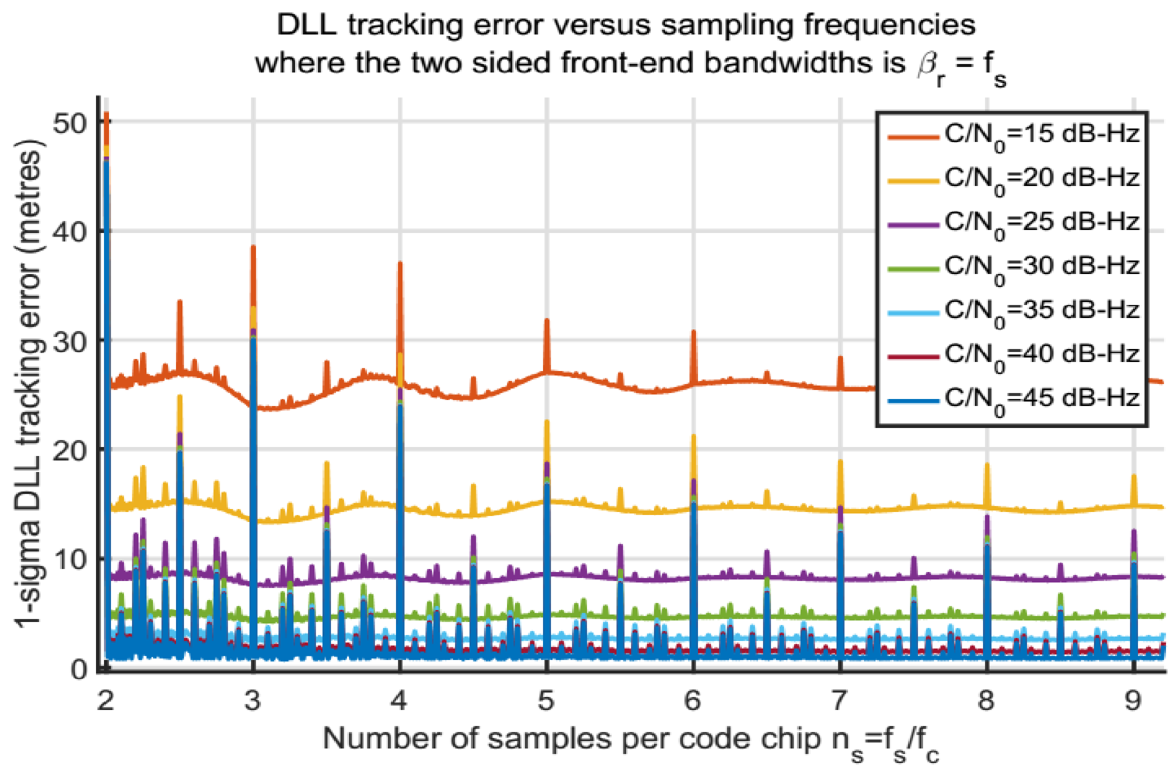
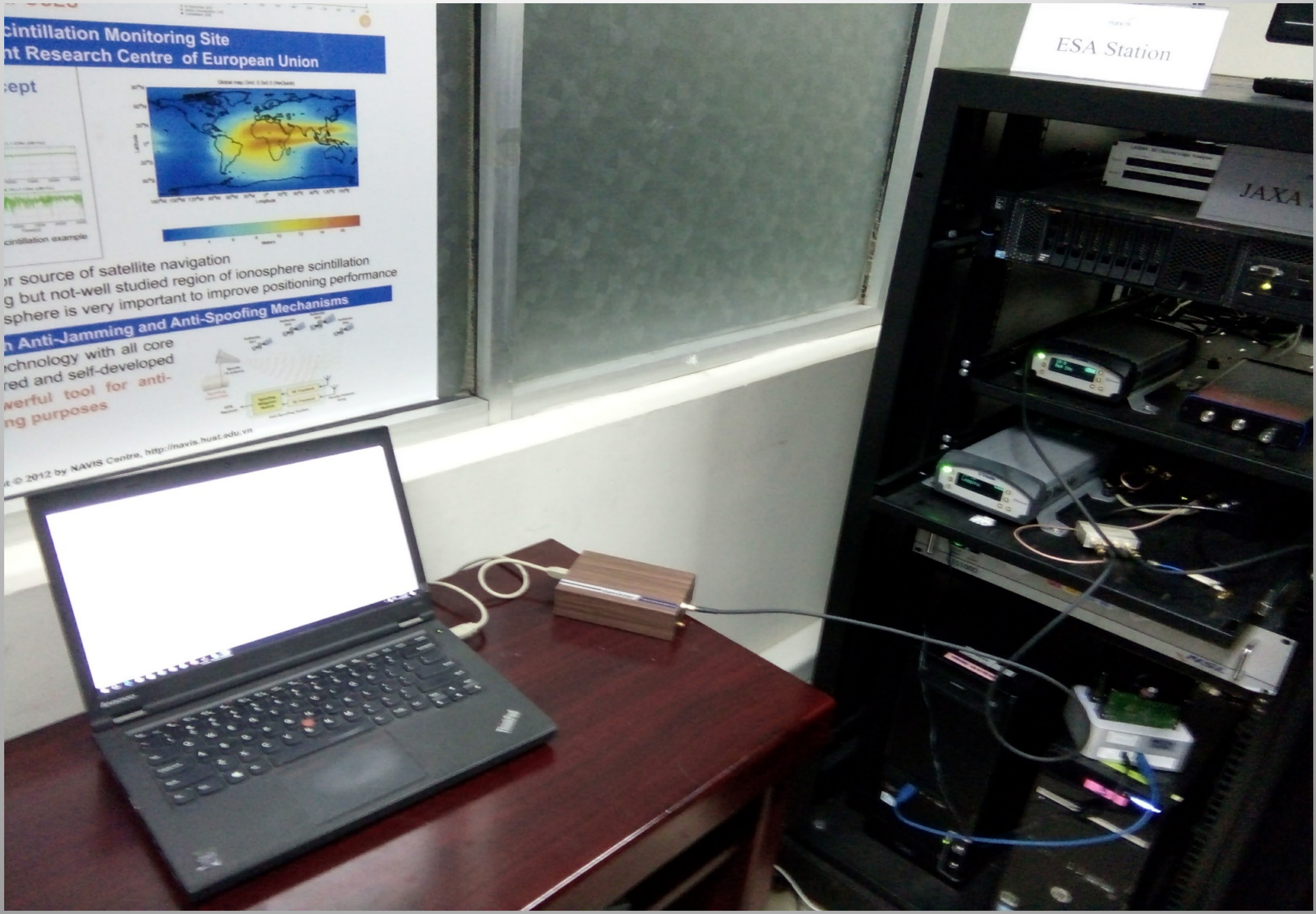


Figure 5: DLL tracking error versus different sampling frequencies (step=$10^{-3} f_c$) with front-end bandwidths $\beta_r = f_s$ for GPS L1 C/A signal with B_L=0.5 Hz and T= 1ms

Experiment setup



Experiment results

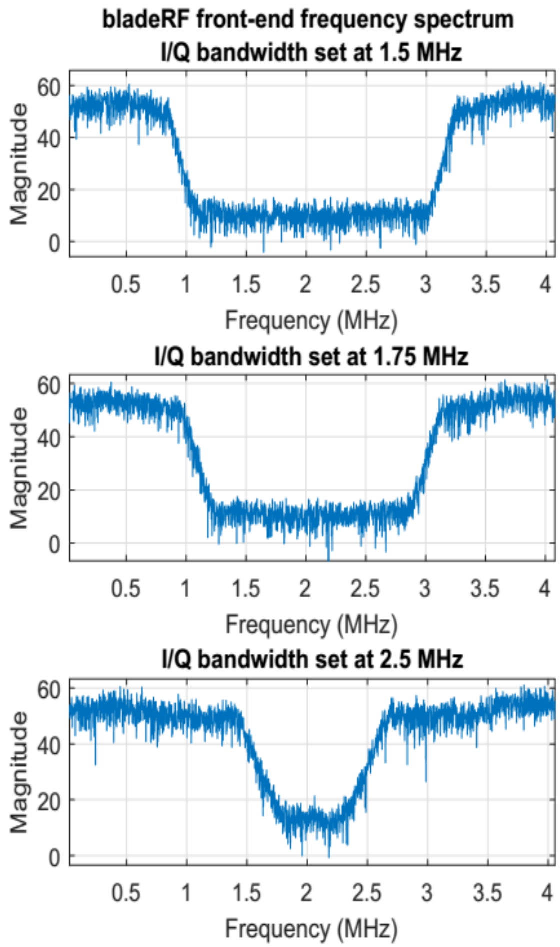
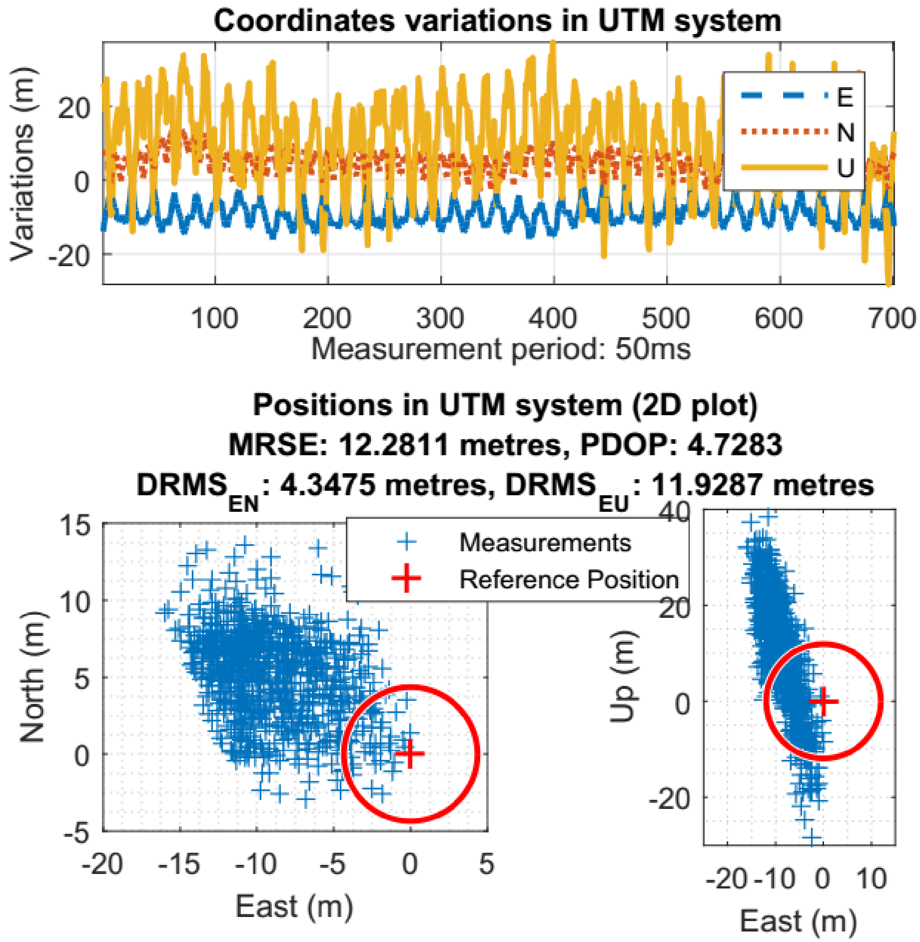


Figure 6: Real complex bladeRF bandwidth versus different I/Q bandwidth settings

C/N_0 (PRN)	Two sided bandwidth (β_r)		
	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

Receiver Position Variation



(e) Without jitter, $f_s = 4.092$ MHz

Conclusions

- The more accurate correlation output and Delay Locked Loop (DLL) code tracking error is proposed
- The relation between the sampling frequency and front-end filter bandwidth has a strong effect on the DLL jitter
- The DLL tracking error is decreased while the sampling frequency is increased and it is more efficient for weak signals (CN0 < 30 dB-Hz) if and only if the front-end bandwidth is fixed and much smaller than the sampling frequency

Thank you

and

Question ?