TFE4188 - Introduction to Lecture 6

Oversampling and Sigma-Delta ADCs

Goal for today

Understand **why** there are different ADCs

Introduction to **oversampling** and **delta-sigma** modulators

1999, R. Walden: Analog-to-digital converter survey and analysis

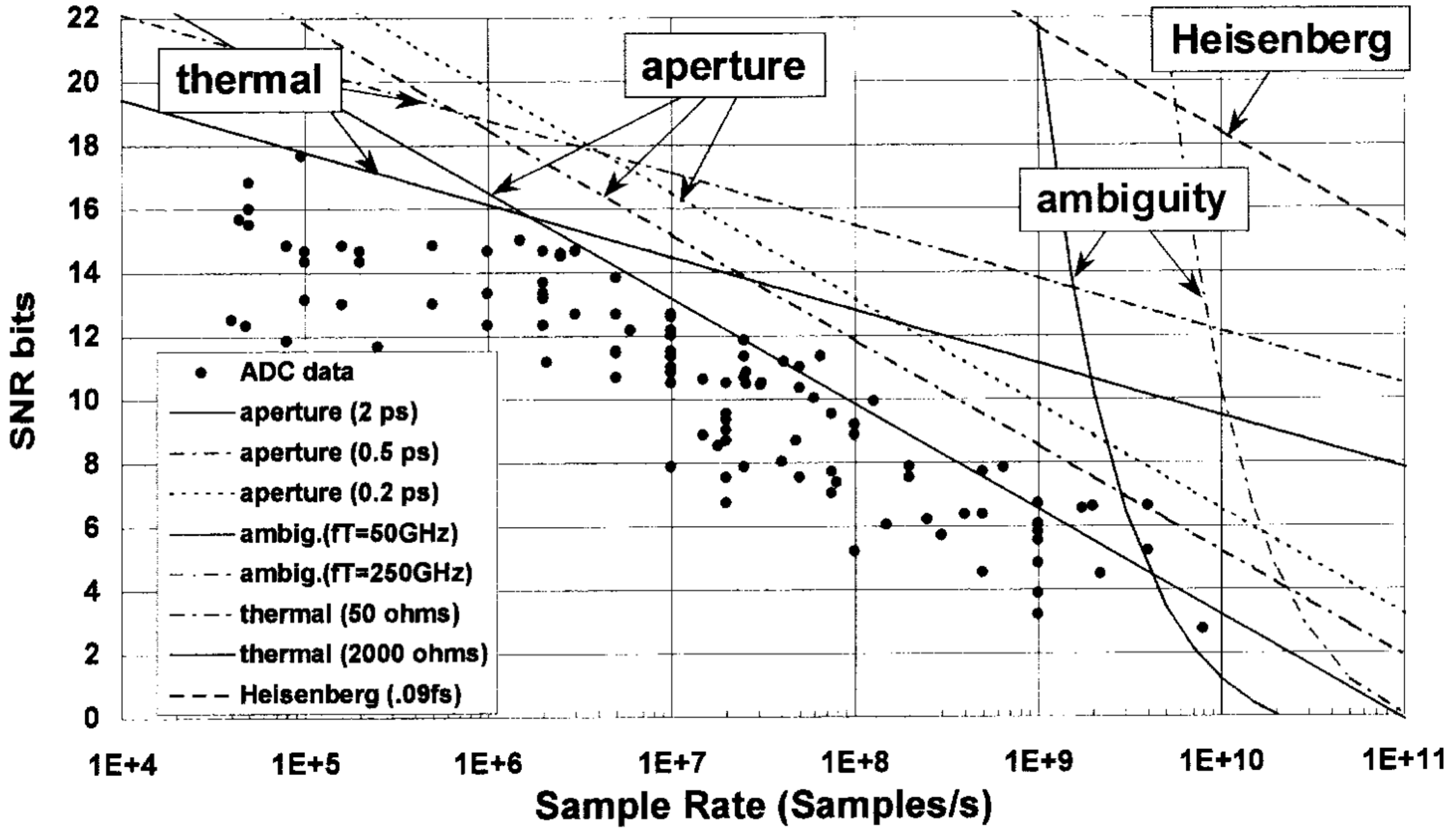
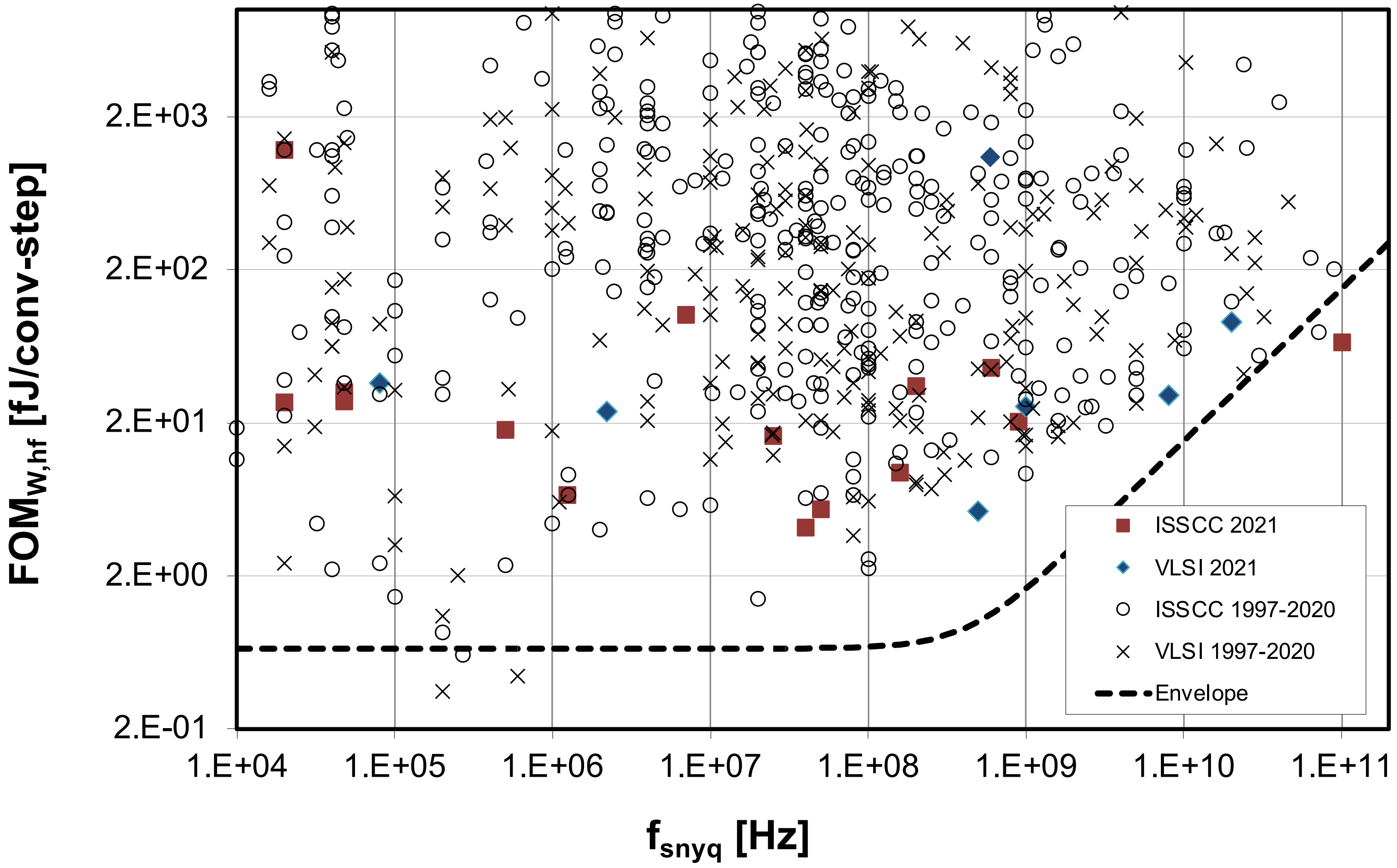


Fig. 7. Signal-to-noise ratio according to SNR-bits = (SNR(dB) − 1.76)/6.02. Three sets of curves show performance limiters due to thermal noise, aperture uncertainty, and comparator ambiguity. The Heisenberg limit is also displayed.

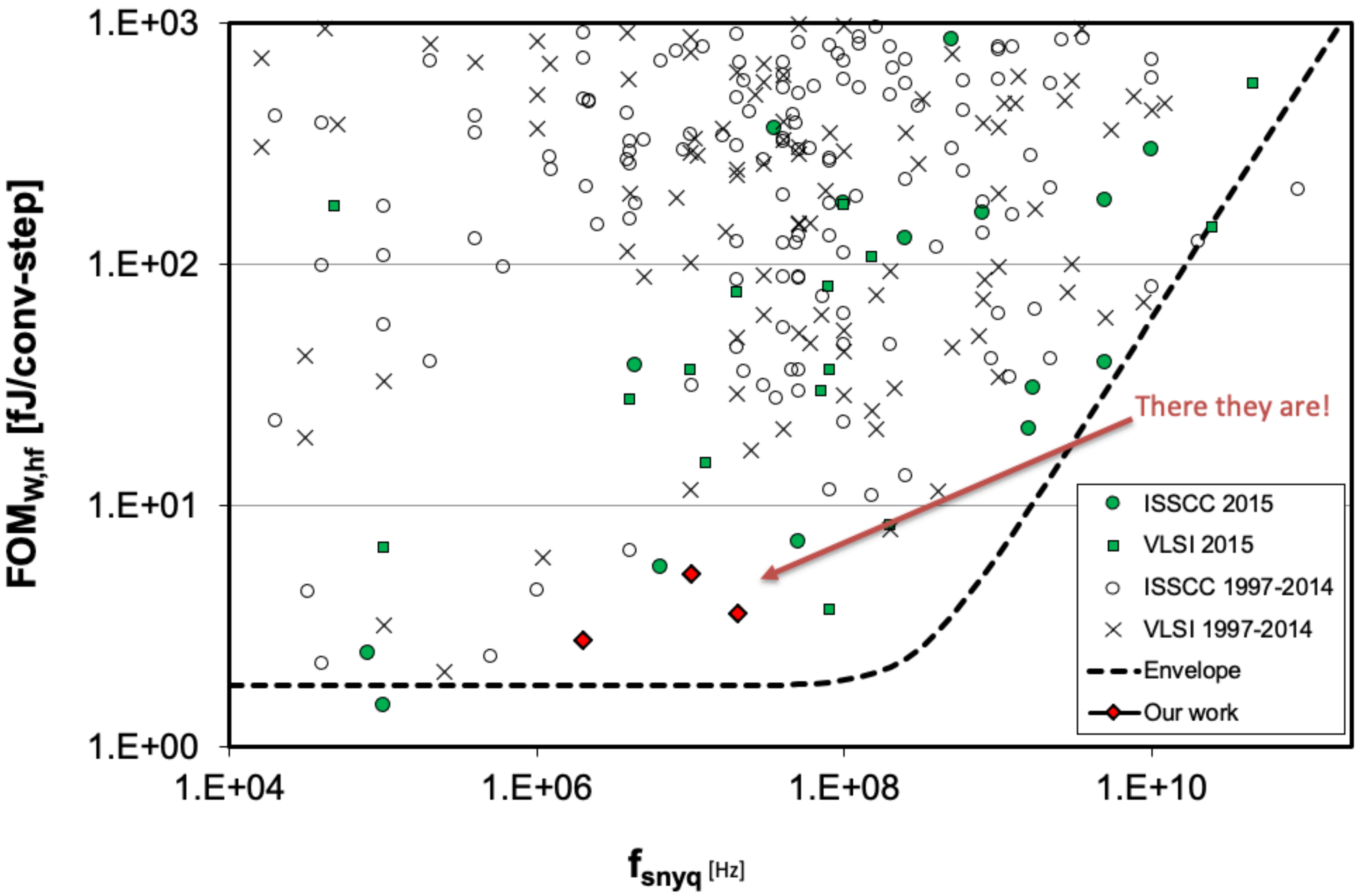


B. Murmann, ADC Performance Survey 1997-2021 (ISSCC & VLSI Symposium)

$$FOM_W = \frac{P}{2^B f_s}$$

Below 10 fJ/conv.step is good

Below 1 fJ/conv.step is extreme



People from NTNU have made some of the worlds best ADCs

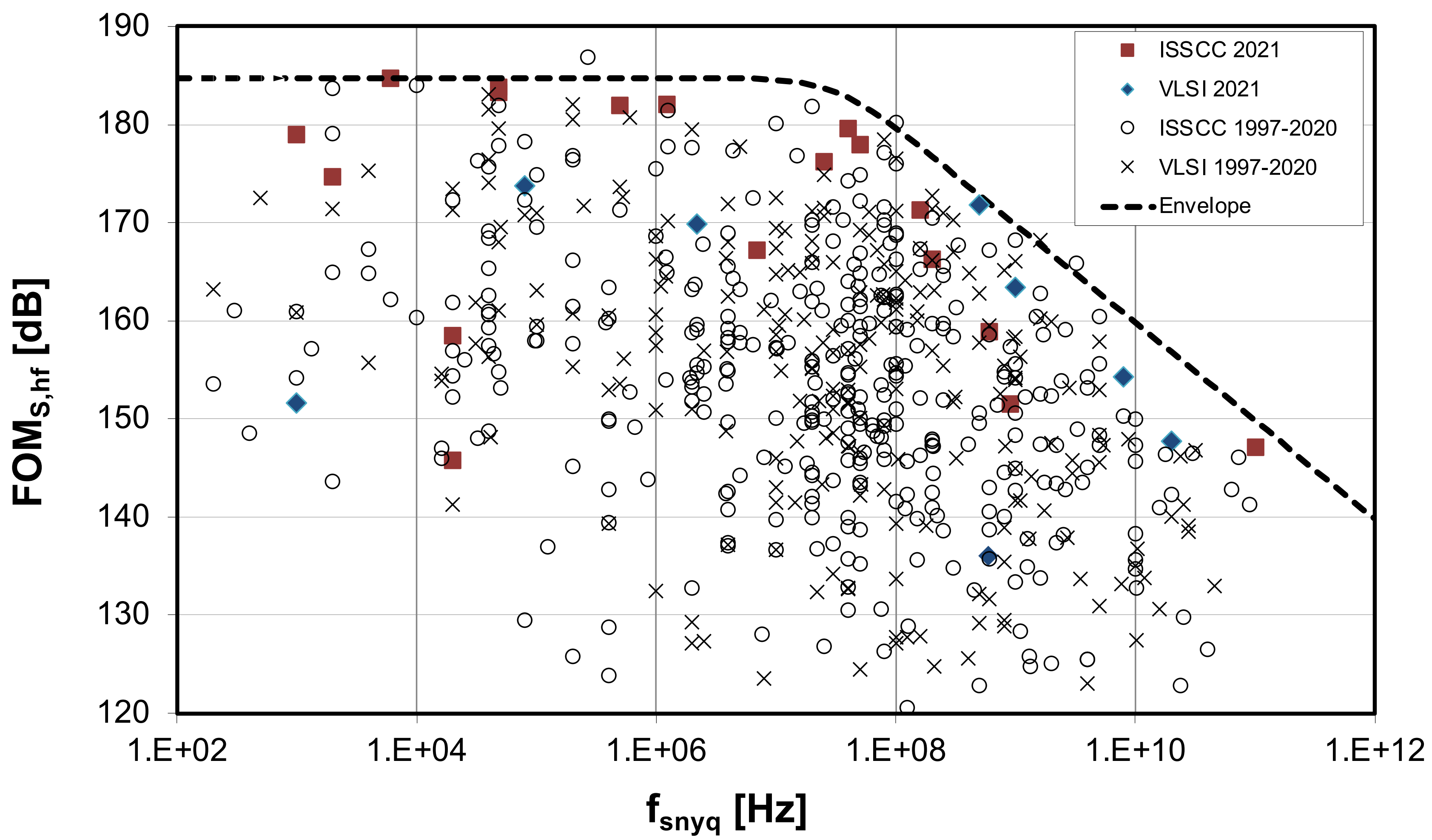
A Compiled 9-bit 20-MS/s 3.5-fJ/conv.step SAR ADC in 28-nm FDSOI for Bluetooth Low Energy Receivers

A 68 dB SNDR Compiled Noise-Shaping SAR ADC With On-Chip CDAC Calibration

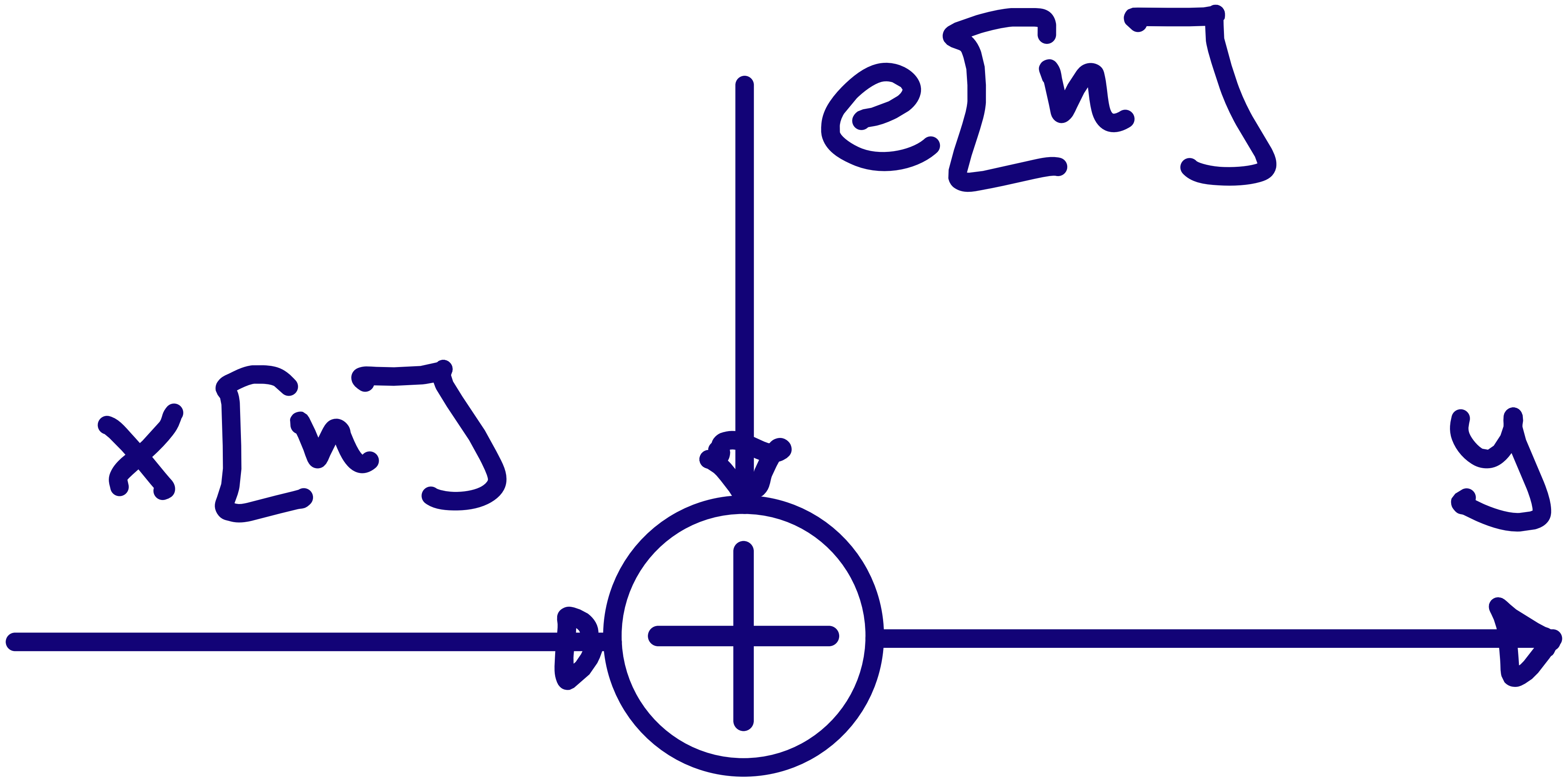
B. Murmann, ADC Performance Survey 1997-2021 (ISSCC & VLSI Symposium)

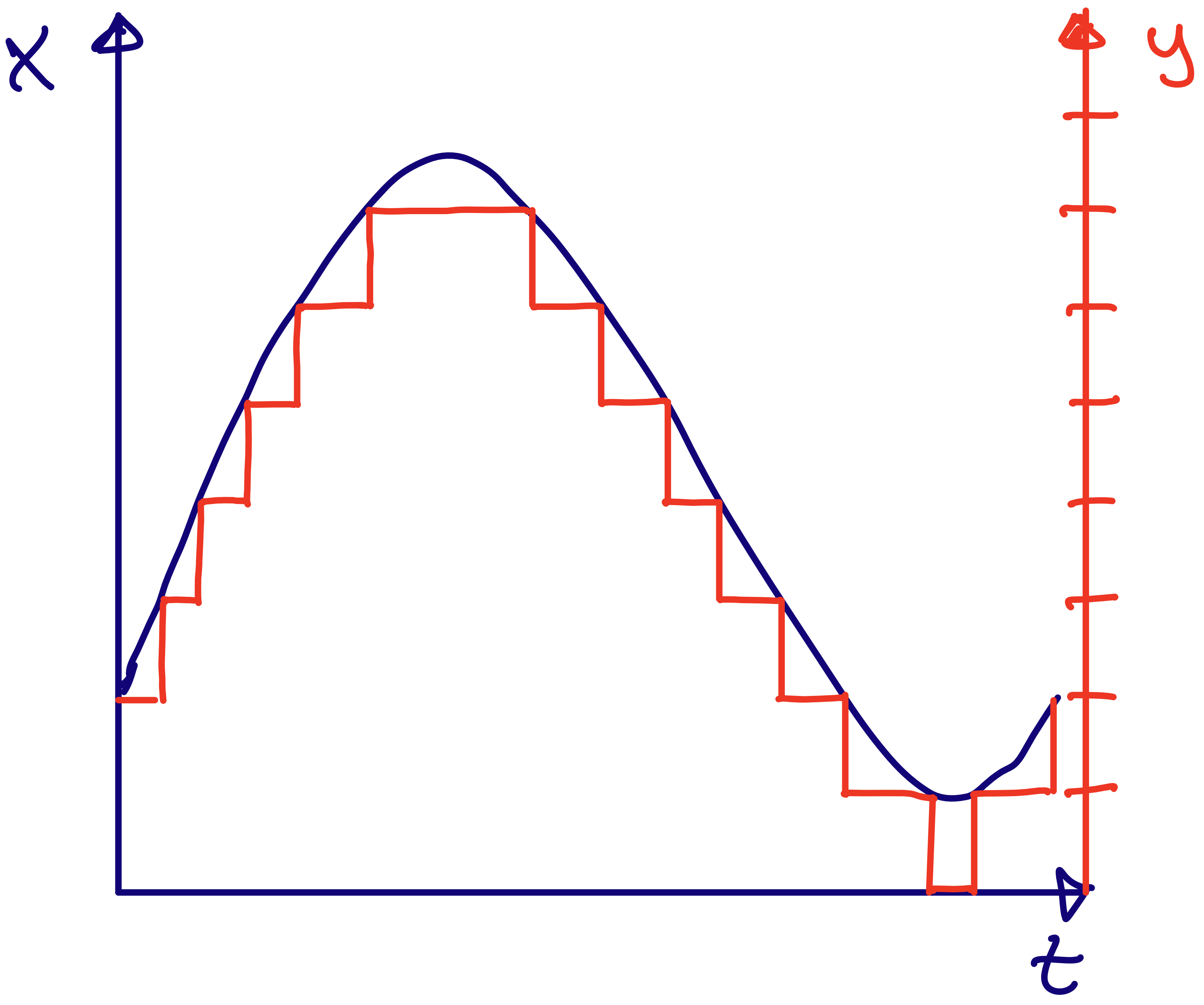
$$FOM_S = SNDR + 10\log\left(\frac{f_s/2}{P}\right)$$

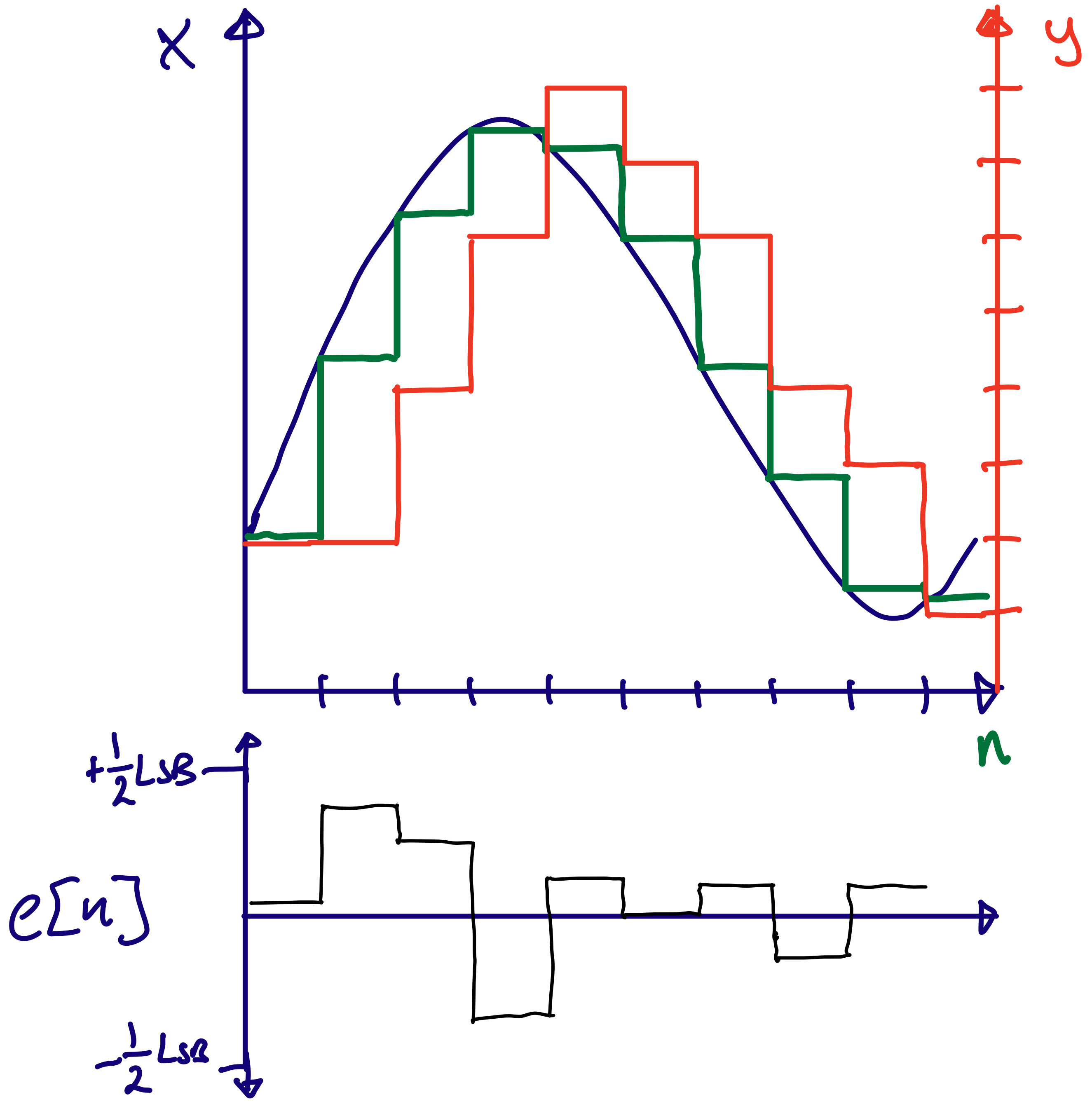
Above 180 dB is extreme



Quantization


e[n]
x[n]
y[n]


x
y
t



x
y
n
$+\frac{1}{2}$LSB
e[n]
$-\frac{1}{2}$LSB

$$e_n(t) = \sum_{p=1}^{\infty} A_p \sin p\omega t$$

where p is the harmonic index, and

$$A_p = \begin{cases} \delta_{p1} A + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi A) & , p = \text{ odd} \\ 0 & , p = \text{ even} \end{cases}$$

$$\delta_{p1} \begin{cases} 1 & , p = 1 \\ 0 & , p \neq 1 \end{cases}$$

and $J_p(x)$ is a Bessel function of the first kind, A is the amplitude of the input signal.

If we approximate the amplitude of the input signal as

$$A = \frac{2^n - 1}{2} \approx 2^{n-1}$$

where n is the number of bits, we can rewrite as ....

See The intermodulation and distortion due to quantization of sinusoids

$$e_n(t) = \sum_{p=1}^{\infty} A_p \sin p\omega t$$

$$A_p = \delta_{p1} 2^{n-1} + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi 2^{n-1}), p = odd$$

$$\overline{e_n(t)} = 0$$

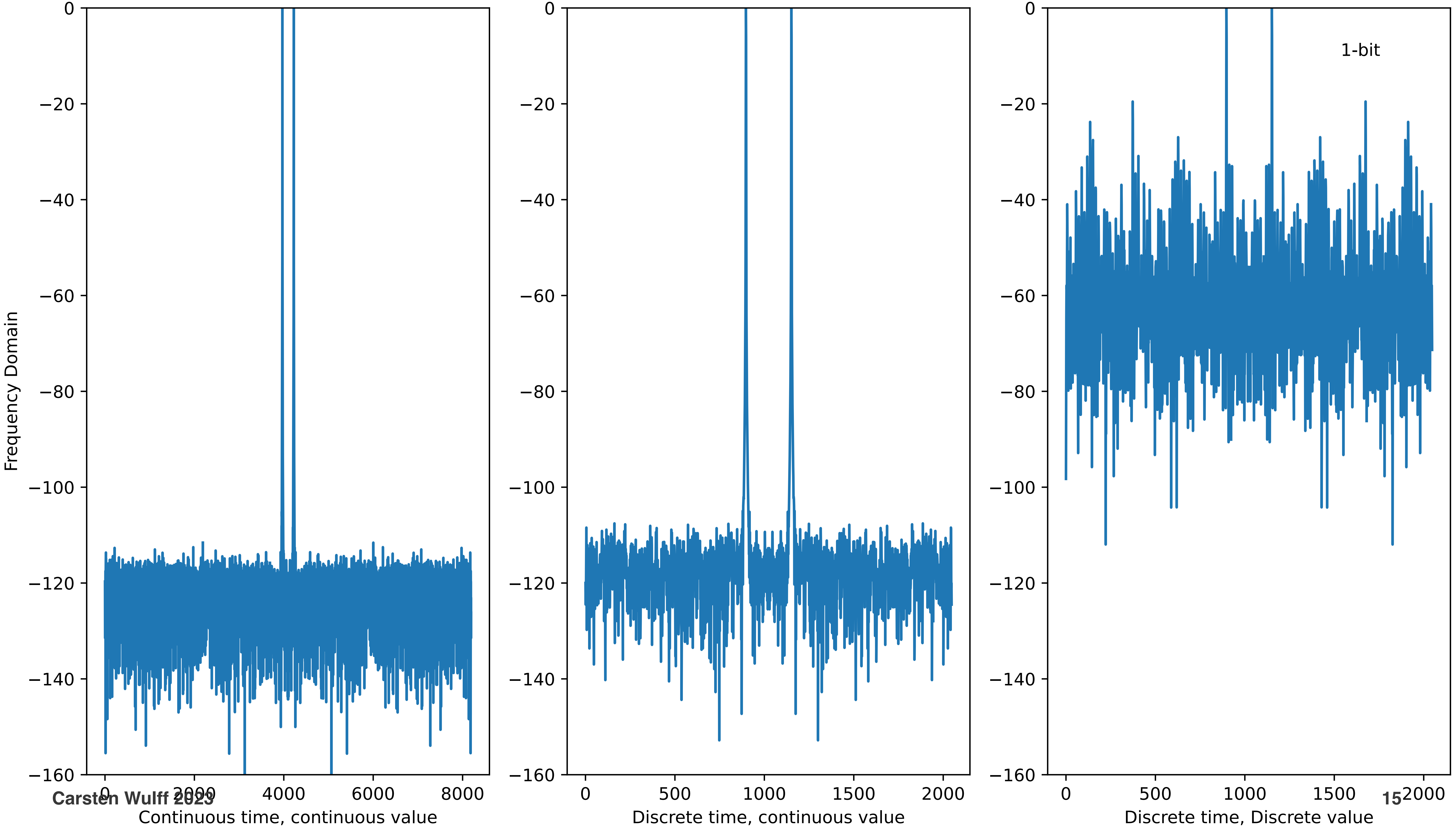
$$\overline{e_n(t)^2} = \frac{\Delta^2}{12}$$

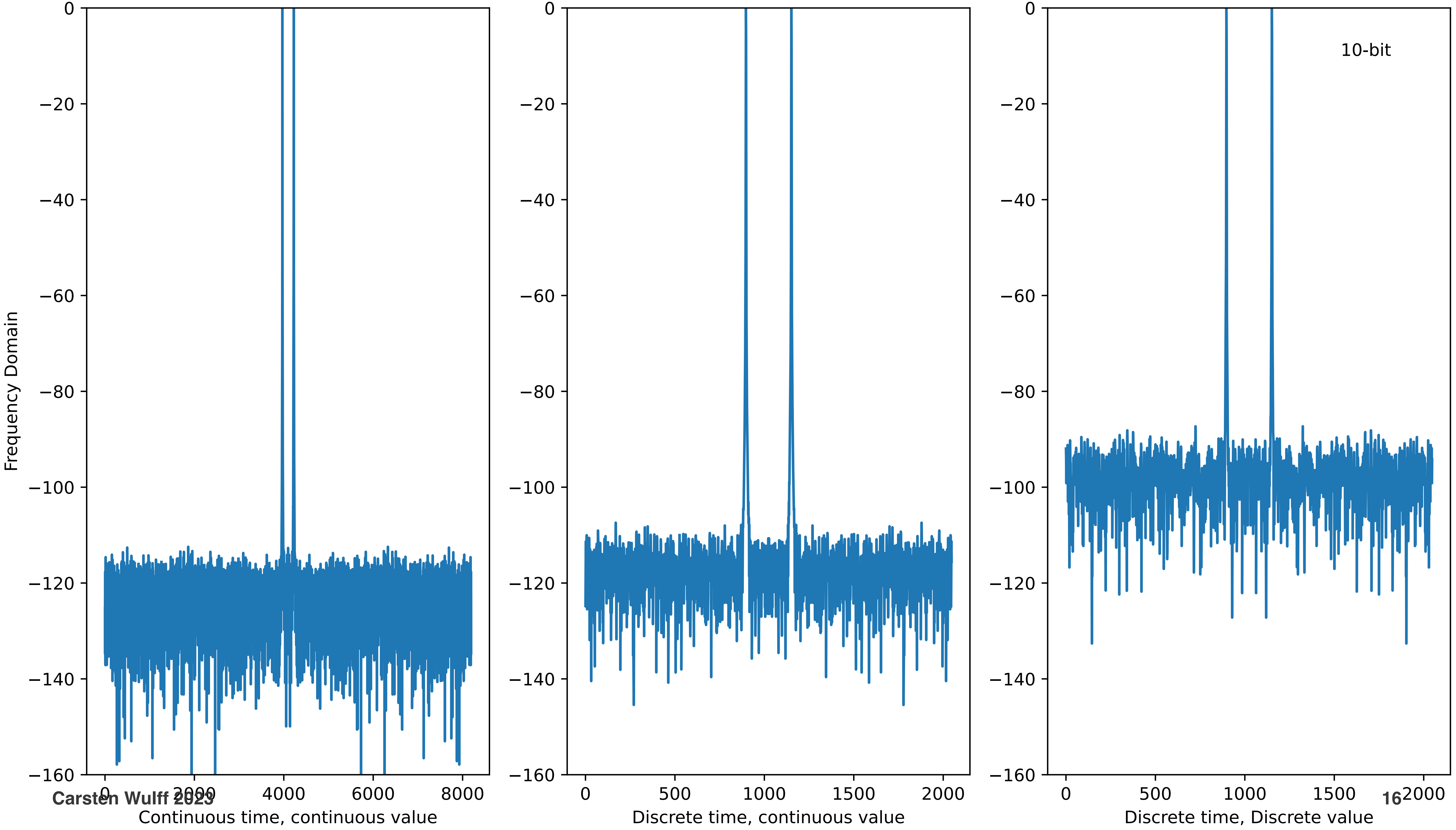
$$SQNR = 10\log\left(\frac{A^2/2}{\Delta^2/12}\right) = 10\log\left(\frac{6A^2}{\Delta^2}\right)$$

$$\Delta = \frac{2A}{2^B}$$

$$SQNR = 10\log\left(\frac{6A^2}{4A^2/2^B}\right) = 20B\log 2 + 10\log 6/4$$

$$SQNR \approx 6.02B + 1.76$$


Frequency Domain
1-bit
Continuous time, continuous value
Discrete time, continuous value
Discrete time, Discrete value



Frequency Domain
Continuous time, continuous value
Discrete time, continuous value
Discrete time, Discrete value
10-bit
0
−20
−40
−60
−80
−100
−120
−140
−160
0
2000
4000
6000
8000
500
1000
1500
2000

Oversampling

In-band quantization noise at an oversampling ratio (OSR)

$$\overline{e_n(t)^2} = \frac{\Delta^2}{12OSR}$$

$$SQNR = 10\log\left(\frac{6A^2}{\Delta^2/OSR}\right) = 10\log\left(\frac{6A^2}{\Delta^2}\right) + 10\log(OSR)$$

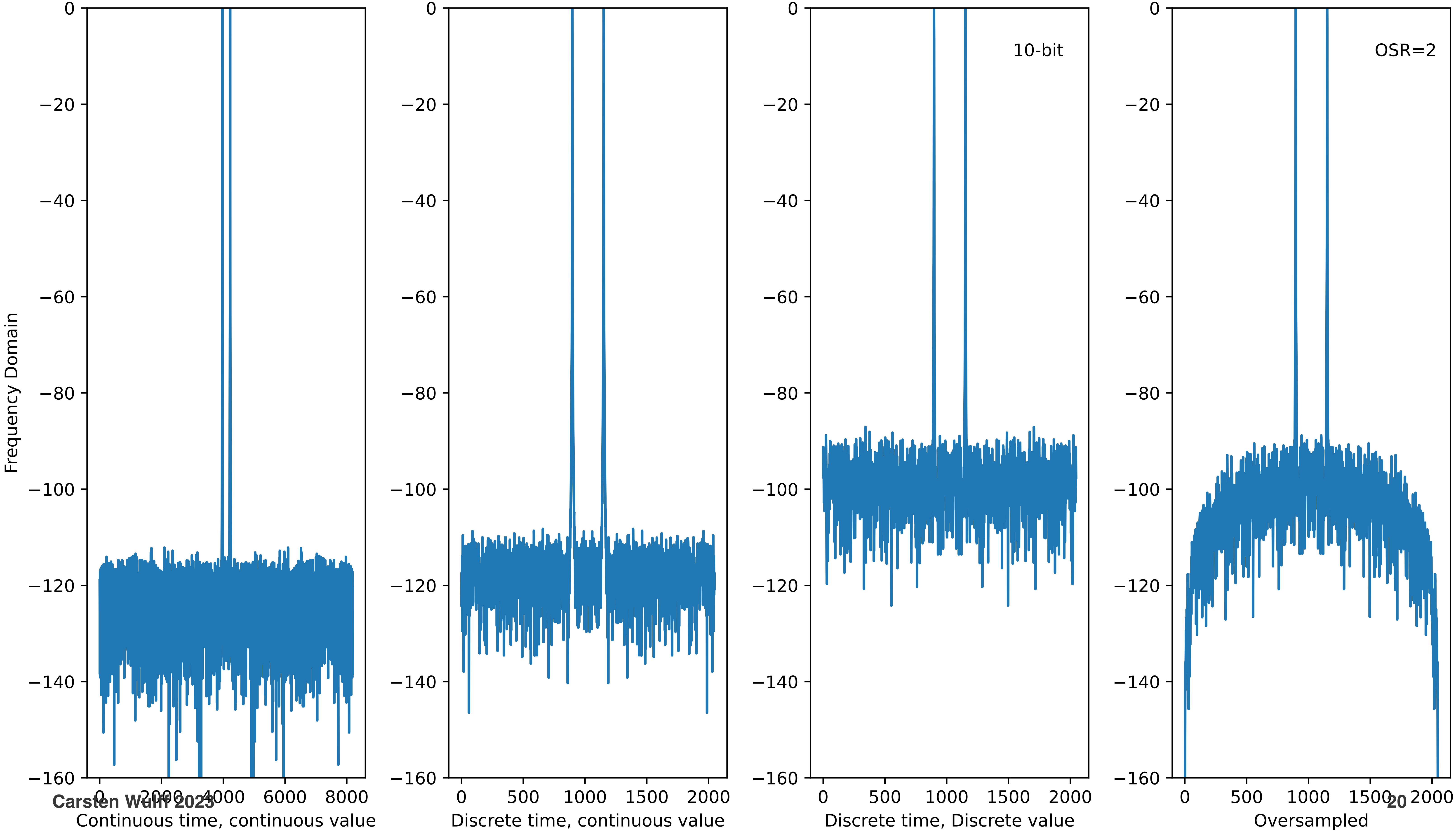
$$SQNR \approx 6.02B + 1.76 + 10\log(OSR)$$

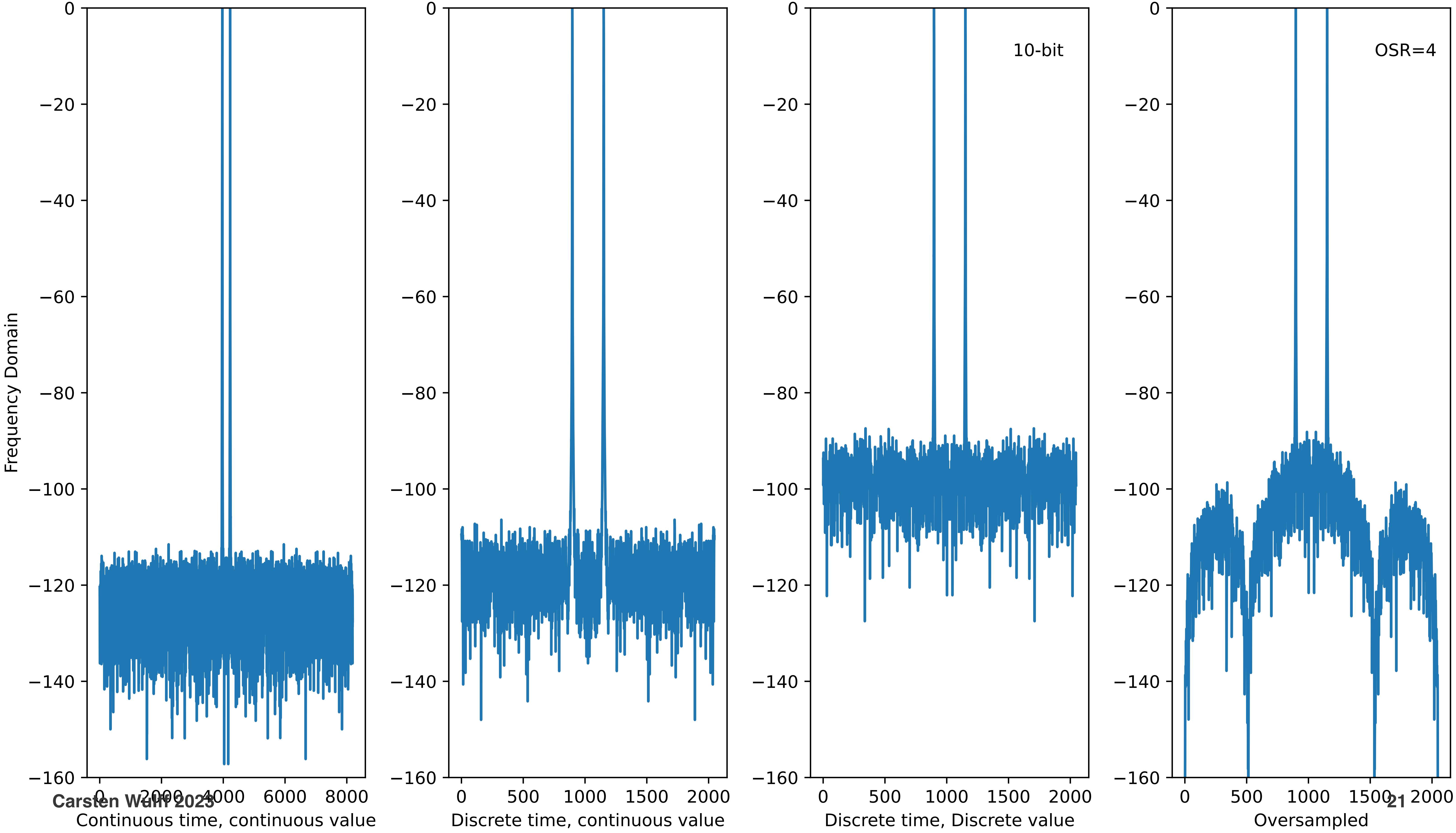
$$10\log(2) \approx 3dB$$
$$10\log(4) \approx 6dB$$

0.5-bit per doubling of OSR

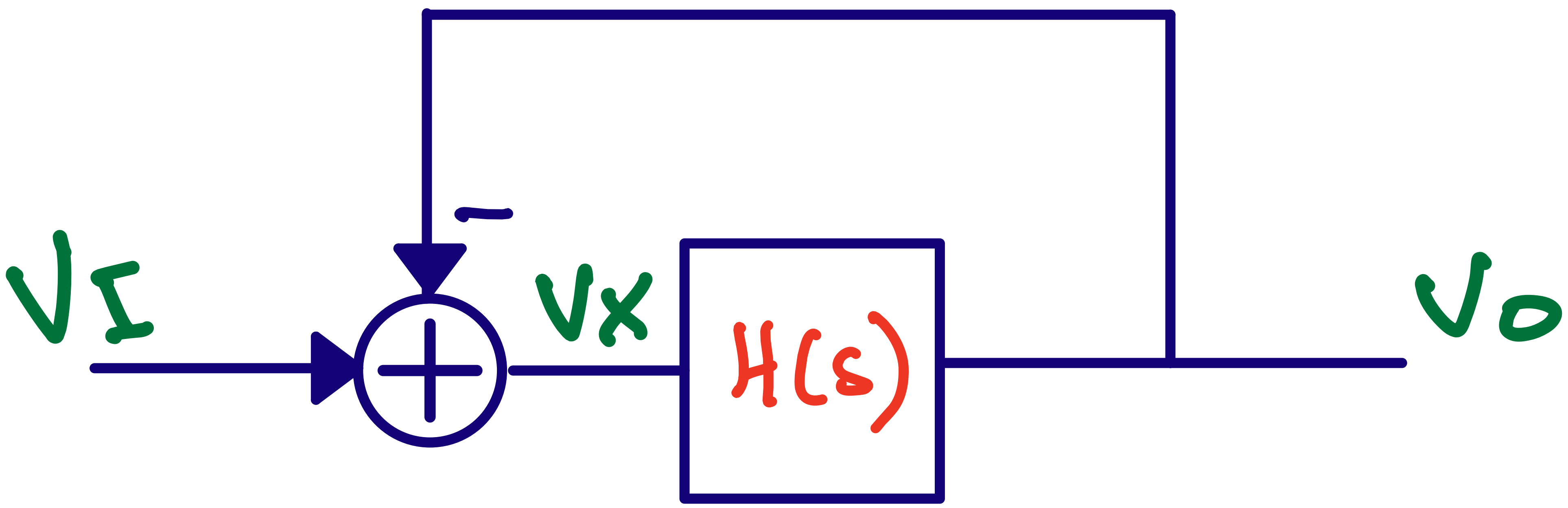
```python
def oversample(x,OSR):
    N = len(x)
    y = np.zeros(N)

    for n in range(0,N):
        for k in range(0,OSR):
            m = n+k
            if (m < N):
                y[n] += x[m]
    return y
```


Frequency Domain
10-bit
OSR=2
Continuous time, continuous value
Discrete time, continuous value
Discrete time, Discrete value
Oversampled

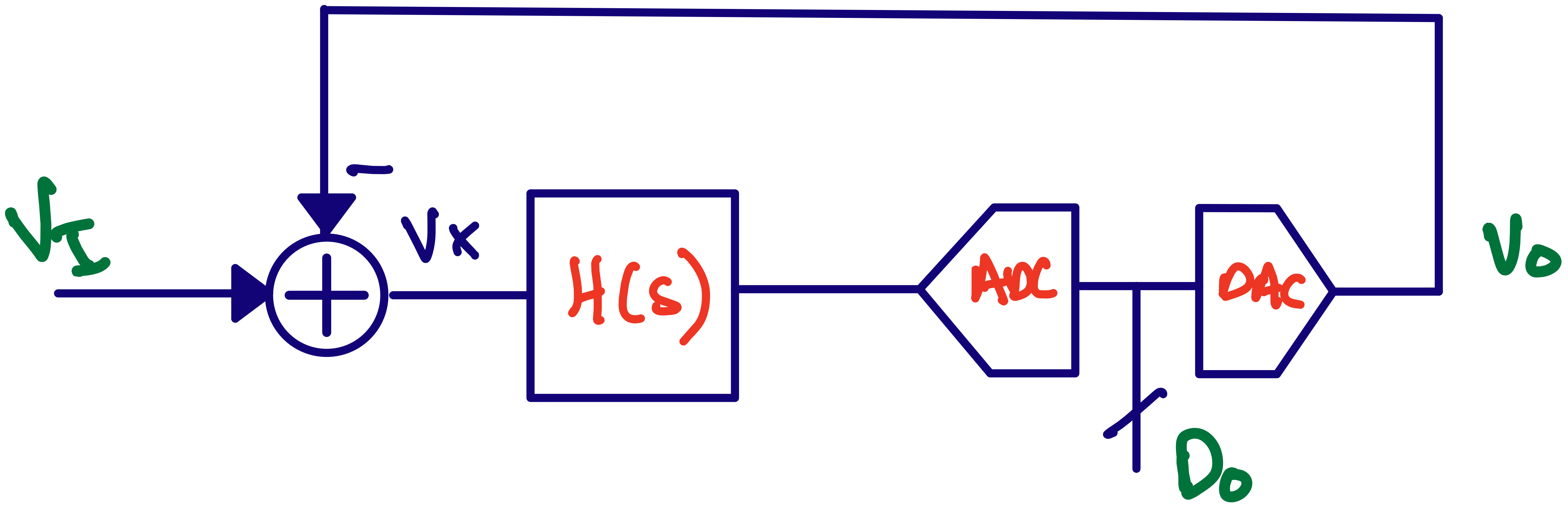

Frequency Domain
Continuous time, continuous value
Discrete time, continuous value
Discrete time, Discrete value
Oversampled
10-bit
OSR=4
0
−20
−40
−60
−80
−100
−120
−140
−160
0 2000 4000 6000 8000
0 500 1000 1500 2000

Noise Shaping

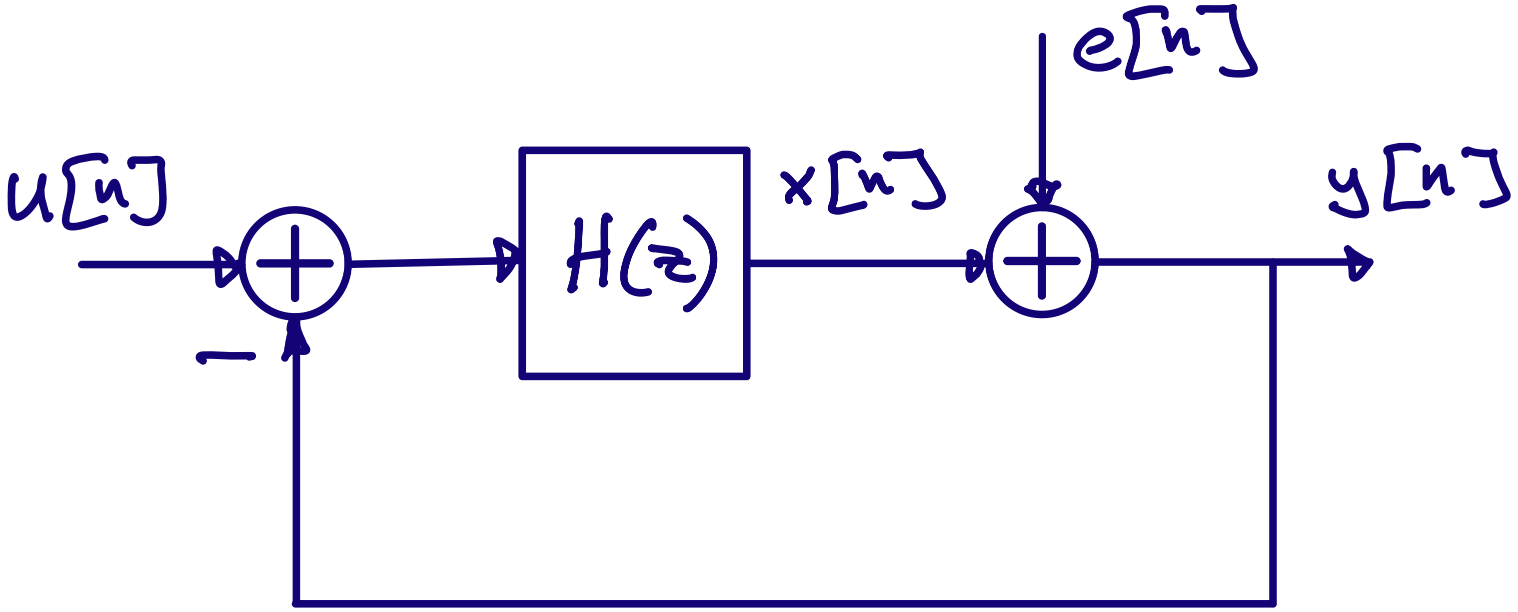


$$V_I - V_o = V_x \qquad V_o = V_x H(s)$$

$$V_I = V_o + \frac{V_o}{H(s)} \xRightarrow{H(s)=\infty} V_o = V_I$$



V_I
−
V_x
H(s)
ADC
DAC
V_o
D_o



Sample domain

$$y[n] = e[n] + h * (u[n] - y[n])$$

Z-Domain

$$Y(z) = E(z) + H(z)\left[U(z) - Y(z)\right]$$

Signal transfer function

Assume U and E are uncorrelated, and E is zero

$$Y = HU - HY$$

$$STF = \frac{Y}{U} = \frac{H}{1 + H} = \frac{1}{1 + \frac{1}{H}}$$

Noise transfer function

Assume U is zero

$$Y = E + HY \rightarrow NTF = \frac{1}{1 + H}$$

Combined transfer function

$$Y(z) = STF(z)U(z) + NTF(z)E(z)$$

First-Order Noise-Shaping

$$H(z) = \frac{1}{z-1}$$

$$STF = \frac{1/(z-1)}{1+1/(z-1)} = \frac{1}{z} = z^{-1}$$

$$NFT = \frac{1}{1+1/(z-1)} = \frac{z-1}{z} = 1 - z^{-1}$$

$$NFT = 1 - z^{-1}$$

$$z = e^{sT} \overset{s=j\omega}{\rightarrow} e^{j\omega T} = e^{j2\pi f/f_s}$$

$$NTF(f) = 1 - e^{-j2\pi f/f_s}$$

$$= \frac{e^{j\pi f/f_s} - e^{-j\pi f/f_s}}{2j} \times 2j \times e^{-j\pi f/f_s}$$

$$= \sin\frac{\pi f}{f_s} \times 2j \times e^{-j\pi f/f_s}$$

$$|NFT(f)| = \left|2\sin\left(\frac{\pi f}{f_s}\right)\right|$$

$$P_s = A^2/2$$

$$P_n = \int_{-f_0}^{f_0} \frac{\Delta^2}{12}\frac{1}{f_s}\left[2\sin \pi f/f_s\right]^2 dt$$

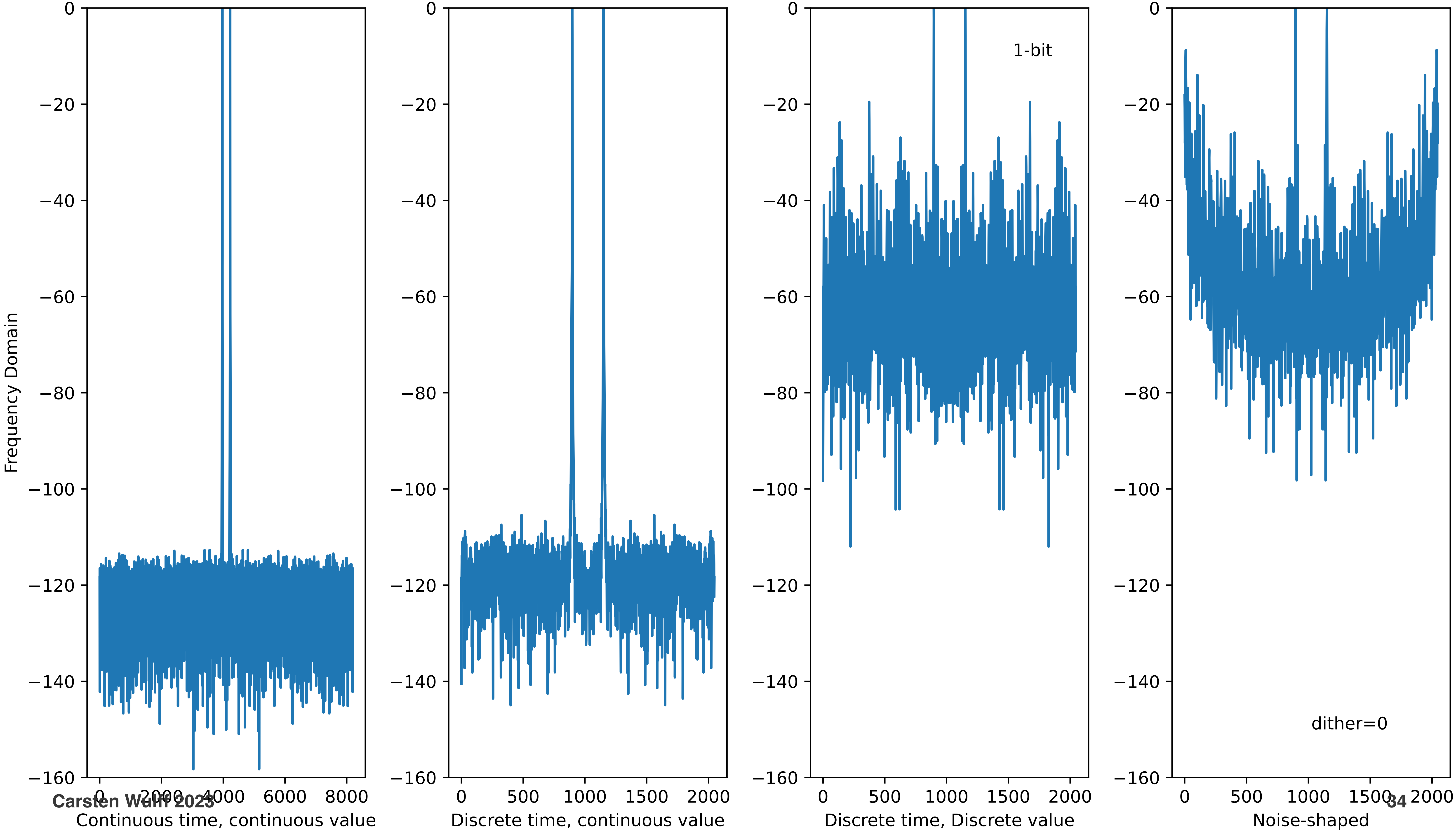
$$\vdots$$

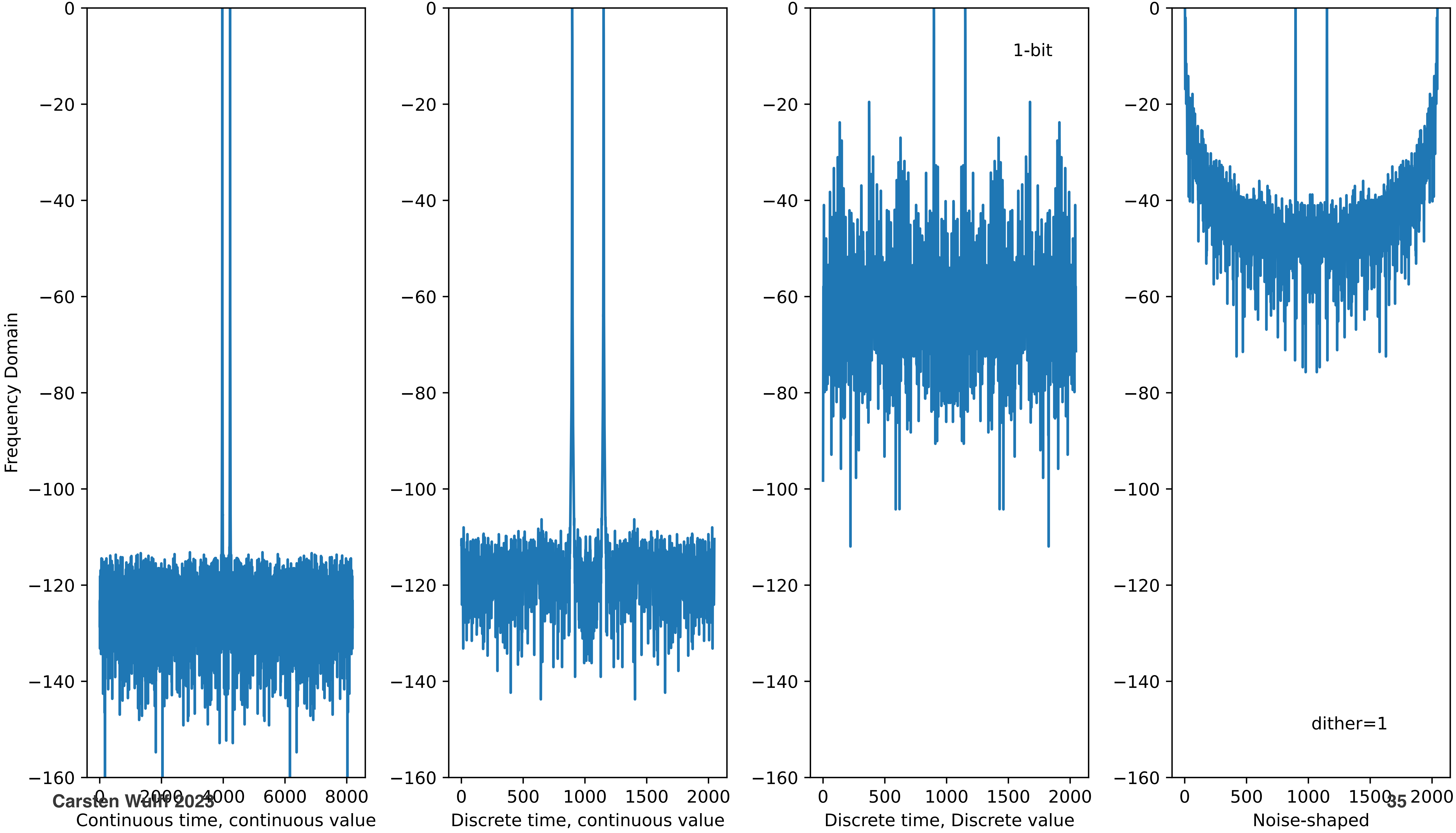
$$SQNR = 6.02B + 1.76 - 5.17 + 30\log(OSR)$$

Assume 1-bit quantizer, what would be the maximum ENOB?

OSR	Oversampling	First-Order	Second Order
4	2	3.1	3.9
64	4	9.1	13.9
1024	6	15.1	23.9

```python
# x_sn is discrete time, continuous value input
dither = 0
M = len(x_sn)
y_sd = np.zeros(M)
x = np.zeros(M)
for n in range(1,M):
    x[n] = x_sn[n-1] + (x_sn[n]-y_sd[n-1])
    y_sd[n] = np.round(x[n]*2**bits
        + dither*np.random.randn()/4)/2**bits
```


Frequency Domain
1-bit
dither=0
Continuous time, continuous value
Discrete time, continuous value
Discrete time, Discrete value
Noise-shaped


Frequency Domain
1-bit
dither=1
Continuous time, continuous value
Discrete time, continuous value
Discrete time, Discrete value
Noise-shaped

Thanks!