DAT116 Lab Report
OMNIBUS

John William Croft

January, 2019

Contents

Foreword

Chapter headings adapted from `https://tex.stackexchange.com/a/163060`.

LAB 1

Sampling

1.2 Preparation

The power of a signal is expressed by the average of the square of the momentary signal value, ie. $P = \text{avg}(f(t)^2)$. In the case of task 3, the signal in question is a pure sine wave of amplitude 0.1. The power of this signal is thus

$$P = \text{avg}(0.1^2 sin^2(\omega t)) = 0.1^2 \cdot \frac{1}{2} = 0.005W \tag{1.1}$$

Expressed in decibels, this power is

$$P_{\text{dB}} = 10 \log_{10} 0.005 = -23.01\text{dB} \tag{1.2}$$

1.3 Uniform sampling of single sine wave

In this task the sample frequency f_s was set to 512 S/s and the simulation time to 1s. A sine generator with f = 7Hz and A = 0.1 was inserted into the model and output plotted against time.

The sine wave was then plotted against frequency to examine the spectral components of the signal, as seen in fig. 1.2. The outlier in this plot is of course the fundamental frequency of the sine wave at 7 Hz, with a power (-23.01dB) that precisely matches the theoretical calculation in the preparation.

The rest of the data points in the plot are what is collectively termed the "noise floor". In this case the signal is ideal, which should be noiseless by definition, but since the model & analysis are digital, a certain amount of rounding-error will occur when representing very small values (negligible in this case), which results in a sort of "rounding noise".

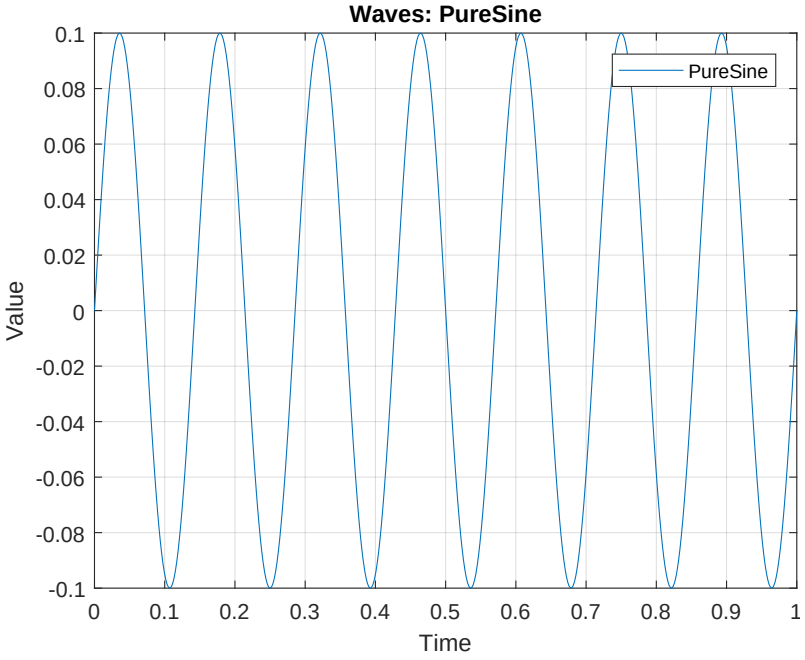


Figure 1.1: Pure sine wave plotted in time domain at 512 S/s

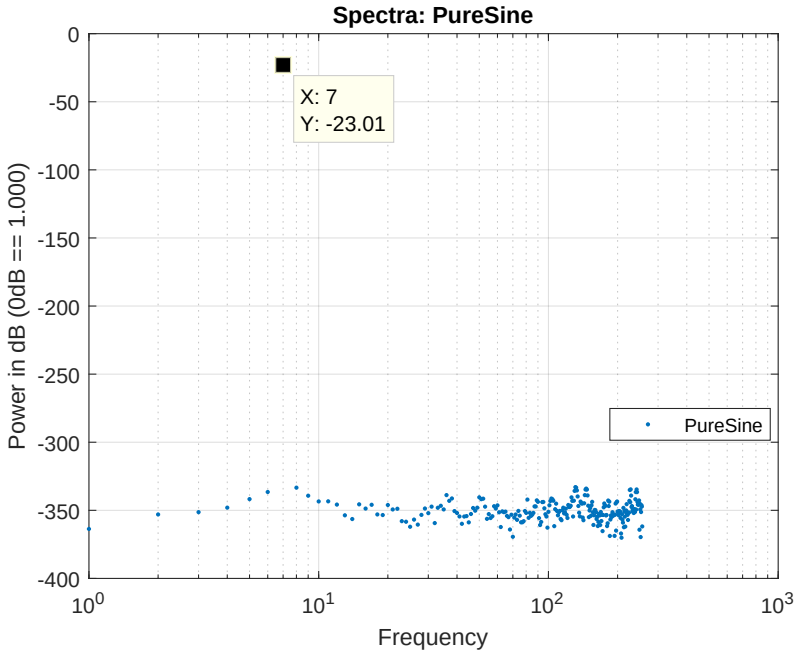


Figure 1.2: Pure sine wave plotted in the frequency domain.

1.4 Multiple sine waves

Four sine wave generators were inserted into the model, all with an amplitude of 0.1 but with varied frequencies of 7Hz, 18Hz, 46Hz and 65Hz respectively. These signals were summed using the `sum` block in `simulink` to construct a compound signal with spectral components at these frequencies. The power spectrum of this compound signal was then examined at different sample rates in order to demonstrate the necessity of an anti-aliasing filter.

Fig. 1.3 shows the signal spectrum with all four components represented with equal power content, as well as the rounding-error noise floor.

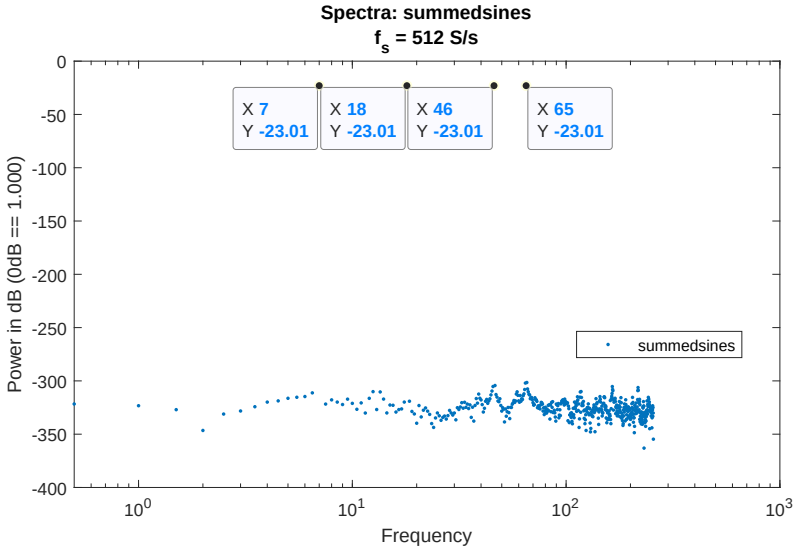


Figure 1.3: Spectral power plot of compound signal at various sample frequencies.

Fig. 1.4 is similar to the previous figure but now the sample frequency is halved and thus the observation window (the frequency content below the Nyquist limit) is also halved. The sine frequency components are, however, still below the Nyquist limit and are thus unaffected.

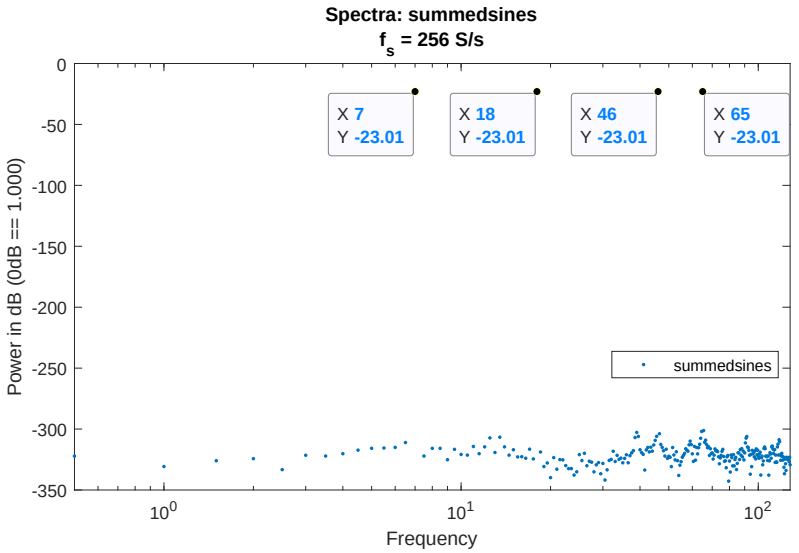


Figure 1.4: Spectral power plot of compound signal at various sample frequencies.

In fig. 1.5 the sampling frequency has been reduced to 128 S/s and the Nyquist limit has corre-

spondingly reduced to 64 S/s, with the consequence that the spectral component at 65Hz is outside the observation window but "mirrored" around this limit at

$$f_s - f_{\text{comp.}} = 128S/s - 65Hz = 63Hz \tag{1.3}$$

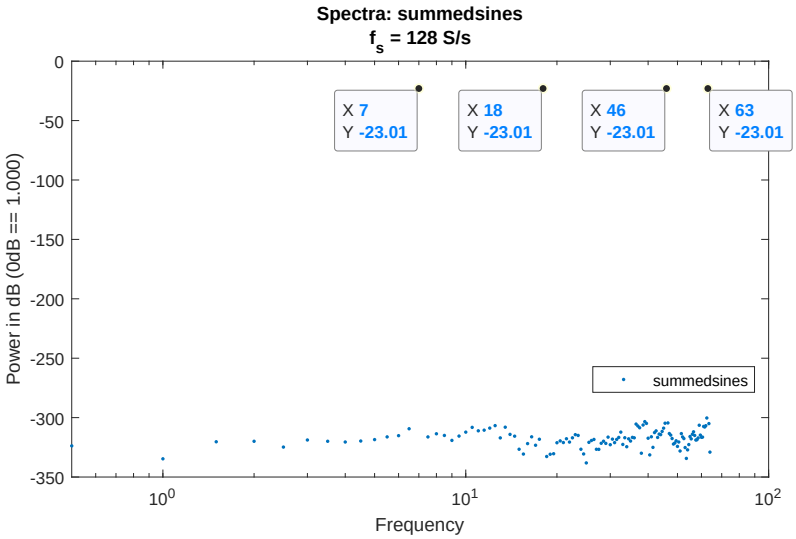


Figure 1.5: Spectral power plot of compound signal at various sample frequencies.

In fig. 1.6 f_s has been further reduced to 64 S/s, giving a Nyquist limit of 32 S/s. The two highest frequency components, 46 Hz and 65 Hz, are now outside of the observation window and are "folded" down to 18 Hz and -1 Hz, respectively, using eq. (1.3). The -1 Hz component is now again outside of the observation window and is thus folded up to +1 Hz around the 0 Hz point (a multiple of f_s). When "folding" occurs it also results in a phase shift of $\pi rad/s$ and so when the original 18 Hz component and the folded 46 Hz component are superimposed, their phase differences cancel each other out, resulting in destructive interference and a final power of 0 (ignoring rounding noise).

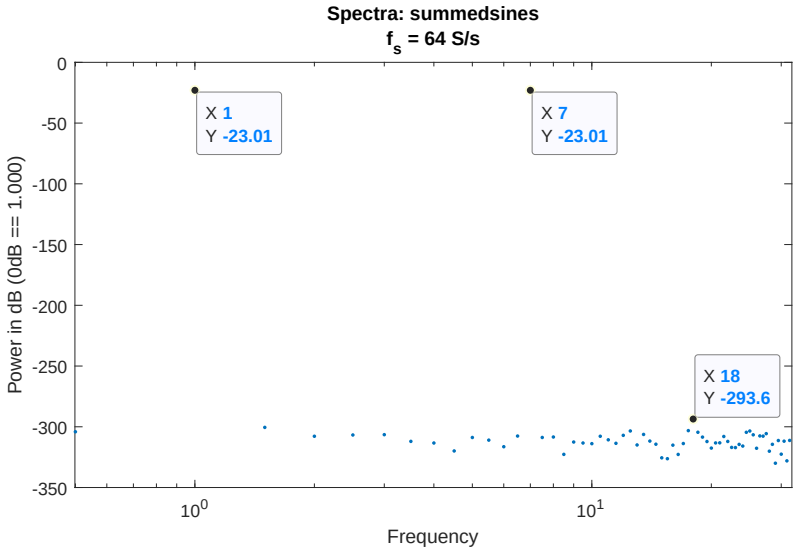


Figure 1.6: Spectral power plot of compound signal at various sample frequencies.

The plot in fig. 1.7 is identical to that of fig. 1.6 with the exception that the 46 Hz component has been phase shifted by $\pi rad/s$. When this component is subsequently folded down around f_s, it is phase shifted again by the same amount, putting it in phase with the 18 Hz component. This has an additive effect, resulting in constructive interference. The power magnitudes are summed at this frequency, giving twice the power relative to the other components, or expressed as decibels $10\log_{10}((2 \cdot 0.1)^2 \cdot \frac{1}{2}) = -16.99dB$.

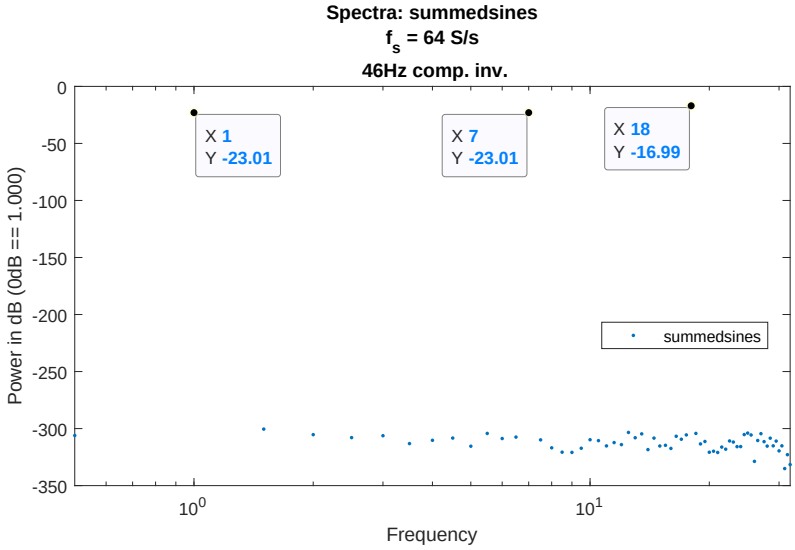


Figure 1.7: Spectral power plot of compound signal at various sample frequencies.

1.5 Anti-alias filtering

In this task a time-continuous, 3$^{\text{rd}}$-order low-pass butterworth filter is inserted into the model after the compound signal described in task 4, using the `Analog Filter Design` block in `simulink`. The filter cutoff-frequency f_c is 23 Hz and the sample frequency of the model is reset to 512 S/s. The simulation length is 1s.

The resulting power spectra of both the filtered and unfiltered signals are shown in fig. 1.8.

Since the Butterworth filter is of the 3$^{\text{rd}}$ order, a roll-off of 18dB per octave is expected. This is achieved with a margin of approximately 1dB, which is due to the fact that the frequencies we are not interested in are not being adequately attenuated, evidenced by the red line at -50dB. This is due to an initial transient in the filter.

To deal with this, the simulation time is extended by an amount that is less than a factor of 2 (in this case, to 1.9) so that only the trailing end of the signal is used, as per the documentation for the `sigspectrum` command.

The resulting spectra are shown in fig. 1.9. The filter response now matches the theoretical response more closely, within 0.2dB.

1.6 Signal-to-noise ratio

A new `simulink` model was created with the same simulation parameters as in the previous task. A sine signal source of amplitude 0.1 and frequency 7Hz as well as a noise source with a variance (and thus power) of 0.005 (see eq. (1.1)) was then inserted into the model. These sources where then summed and viewed in the time and frequency domains using `MATLAB`, as shown in fig. 1.10.

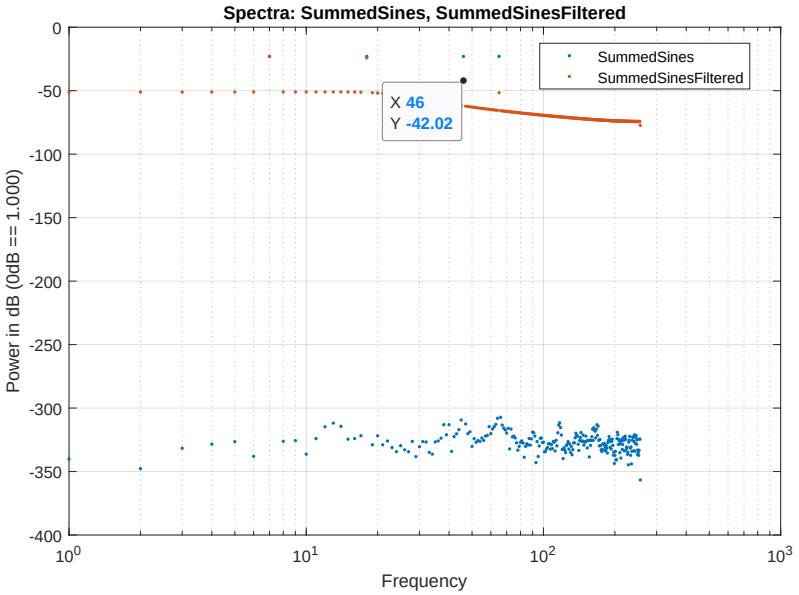


Figure 1.8: Discrete compound signal before and after analog pre-filtering. Filter transient present as a power offset in the pre-filtered signal.

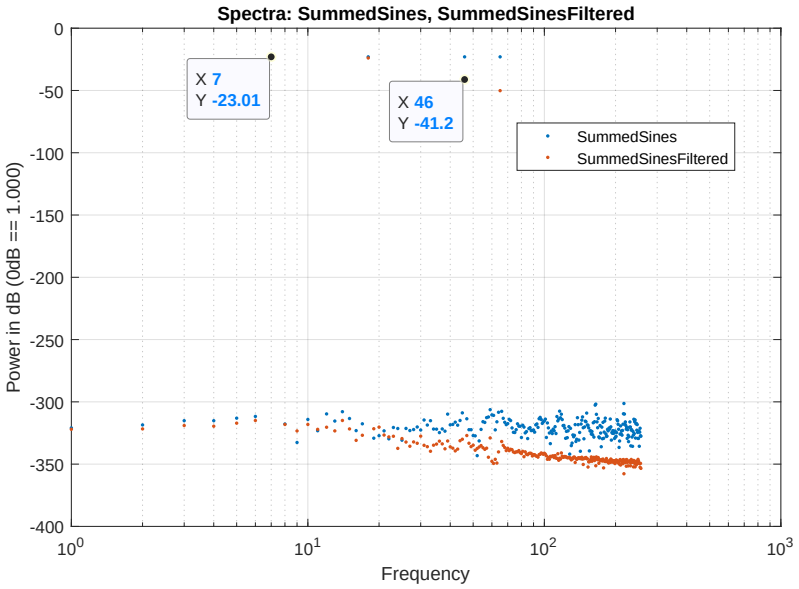


Figure 1.9: Discrete compound signal before and after analog pre-filtering. Initial filter transient suppressed.

The fundamental frequency of the sine signal can be clearly seen in the frequency spectrum at approximately the expected theoretical power. Any discrepancy can be attributed to constructive interference by random noise at that frequency.

The total power of the spectrum can be calculated by taking the sum of all the frequency components, giving 0.01, which agrees with the sum of the sine signal power and the random noise variance. The power of the fundamental frequency component (vector element $f + 1 = 8$) was then calculated at 0.0047, but will vary depending on the generated noise.Finally, the SNR, the ratio of signal power to the power at all other frequencies, was determined to be 0.8753. Expressed in decibels this is -0.5784 dB. This is in line with what is expected, as noise at the fundamental frequency will interfere to give some value around the sine signal power 0.005. There is in fact an

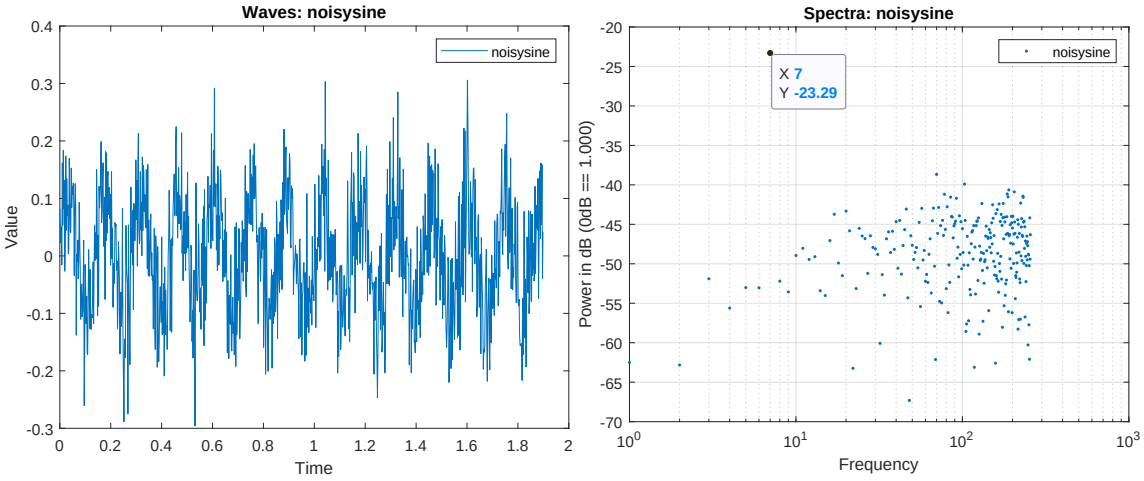


Figure 1.10: Time and frequency plots of sine signal with superimposed guassian noise source of equal power.

equal probability of constructive interference instead, which would give an SNR > 1.

Next, a Butterworth filter of the type used in task 5 was inserted with the following result in fig.

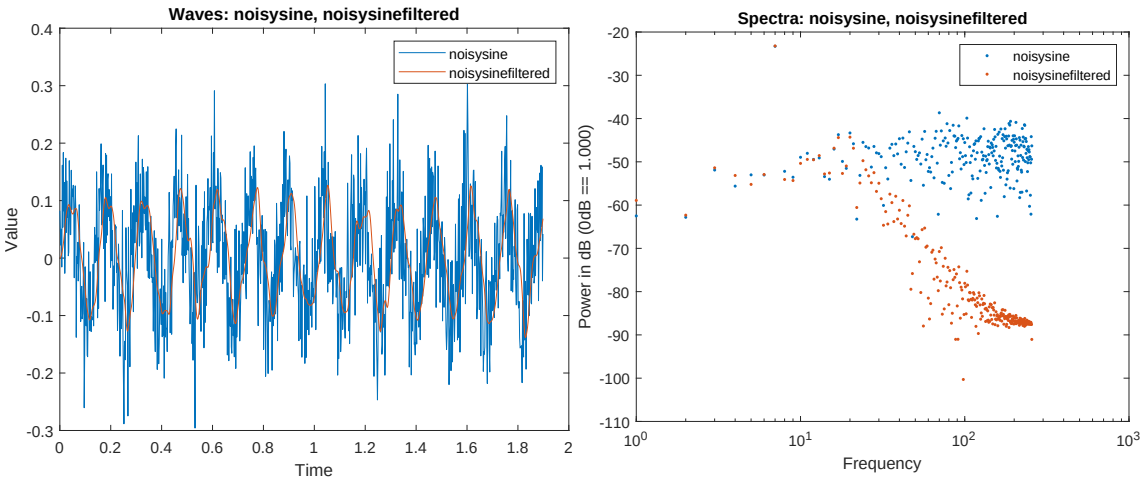


Figure 1.11: Noise-infused sinusoid before and after analog filtering.

The filter initially achieves a noise attenuation of -17.1 dB/octave but the response is clearly not linear but slightly curved at higher frequencies, due to the noise distorting the signal and making it non-periodic (introducing a discontinuity at the edge of the fft sampling interval). At a frequency of 184 Hz (three octaves higher than the cutoff-frequency, or $log_2(f/f_c)$) the noise was attenuated by -34.64 dB. The power of the fundamental frequency (7 Hz, visible near the top of the spectral plot in fig. 1.11) is -23.22 dB, slightly lower than its theoretical value.

By extending the simulation time to 65s (slightly more than a factor of 2), the effect this artefact has on the results can be mitigated.

The response appears to be more linear in this plot and at 184 Hz the signal is attenuated by -49.9 dB, compared to the theoretical value of 54 dB, indicating that the filtering works well even at higher frequencies. The power at the fundamental frequency is now very close to its theoretical value -23.01 dB. This is due to the fact that the filter has reduced the power in the stop band, and since the total power in the spectrum must remain constant, more power is available at the fundamental frequency.

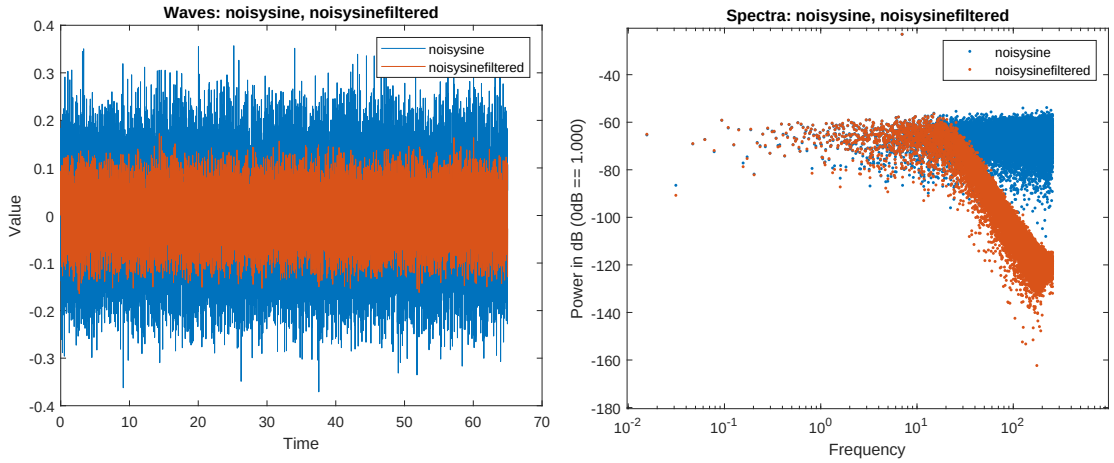


Figure 1.12: Same signal as above with simulation time extended to 65s.

The "noise floor" has been reduced in the longer simulation as, for the same reason as above, the total power in the spectrum can not change and yet there are more datapoints plotted meaning that each individual component of the noise signal must be of a lower magnitude.

The SNR of the unfiltered and filtered signals were -0.0572 dB and 10.2653 dB, respectively. Examining the ratio of the noise power before and after filtering, a 90.7% reduction was observed. This can be compared to the passband power of an ideal filter (f_c/f_{max}) which gives value of 9%, making it a good method of sizing the filter dimensions.

Re-setting the simulation time to 1.9s (as in the previous task) yielded SNR values for the unfiltered and filtered signals of -0.5784 dB and 12.1770 dB, respectively, and a noise power reduction of 94.6%.

This may be misleading, however, as it has previously been shown that HF filtering suffers at short simulation times, and hence the filter may not function realisticallyas expected. The question of increased simulation time, as usual, a tradeoff.

1.7 Reflection Questions

1.7.1

When oversampling a signal, the portion of the sampled frequency spectrum that is taken by the signal band of interest is smaller than it would otherwise be. This in turn means that there is a greater range (up to the Nyquist limit $f_s/2$) in which the digital filter may work, relaxing requirements and potentially improving cut-off performance. The analog pre-sampling filter need then only suppress frequency components above $f_s/2$. Again, the relaxed filter requirements will apply also to the analog filter, and a wide transition band may be used, reducing complexity and costs.

1.7.2

Avoiding initial transients altogether is perhaps not ideal from a pedagogical perspective, as students can simply proceed with other tasks without giving any thought to the underlying method of mitigating discontinuities. This method used has little bearing on this particular lab however, and having to consider methods of obtaining reliable results may only serve to confuse students. The *signal windowing* method would undoubtedly be more complicated to understand and describe as

it would require more theoretical knowledge as well as research into the various MATLAB facilities used to implement it.

LAB 2

Variability and feedback

2.3 Setup and launch

The Cadence environment to be used throughout this lab was initialized using a script provided by the faculty. This made available components from the `DAT116` library. Additionally, a lab-specific library was created and attached to the `cmos65` technology library.

2.4 Using an op-amp macro model

In this task, a macro model of an op-amp (specifically `op1` from the `DAT116` library) was used in conjunction with a resistive voltage-divider to form a negative feedback amplifier. The macro model simulates the behaviour of a generic op-amp without requiring the computationally expensive simulation of a transistor-level implementation.

The resistors `rin` and `rfb` were set to $10\,\mathrm{k\Omega}$ and $50\,\mathrm{k\Omega}$, respectively. This gives a theoretical closed-loop gain of approximately

$$A_{\mathrm{CL}} = -\frac{50\,\mathrm{k\Omega}}{10\,\mathrm{k\Omega}} = -5 \tag{2.1}$$

or

$$A_{\mathrm{CLdB}} = 20\log_{10} 5 = 13.9794\,\mathrm{dB} \tag{2.2}$$

A sine-wave generator of amplitude 0.1 V and frequency 10kHz was set on the amplifier input. Through the `Virtuoso Heirarchy Editor` the design was put into *Analog Mixed-Signal* mode. Additionaly, the simulator `ADE L` was set to use `ams` simulation. The final inverting amplifier schematic is shown in fig. 2.1.

With the schematic and various configurations completed, a transient analysis of the circuit was performed. This consists of generating a netlist that corresponds to the graphical schematic, which is then used by the simulator to compute a given simulation.

The transient plot, seen in fig. 2.2, clearly shows an amplitude gain of -5, as calculated in eq. 2.1. This is not surprising as the model is, in this configuration, mostly ideal.

2.5 Component variability

This task concerned the ability of the amplifier to deal with parameter variations. The sine source AC amplitude was set to 1V and an AC sweep from 0.1Hz to 10MHz was performed, producing the Bode plot seen in fig. 2.3. The plot confirms what we learned in the previous section (at frequencies below 100 kHz).

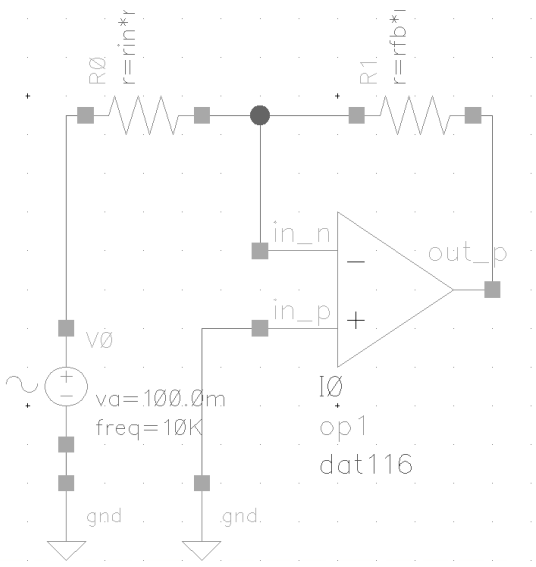


Figure 2.1: The circuit schematic for the negative feedback amplifier as seen in *Cadence Virtuoso*.

Next, a gain plot (which represents the magnitude in dB relative to the input magnitude) was obtained with the use of the `waveform calculator`. Using the tool and the `dB20` function, the existing voltage representation was easily converted to dB. The `waveform calculator` was additionally used to derive the raw gain of the amplifier. Both the closed-loop and open-loop gain are plotted in fig. 2.4. Both of these curves conform to our expectations and one can already observe how the closed-loop gain is limited at high frequencies by the raw gain.

The slope of the raw gain curve was measured in fig. 2.5 to confirm a characteristic roll-off of 20dB/dec. (ie. (117.25dB - 78.05dB)/2 decades = 19,6dB/dec).

Next, we set the OP-amp parameter AOLDC (raw gain at DC) to 100dB and resimulated the design, with the result seen in fig. 2.6. This adjustment affected the raw gain curve by reducing the maximum value though maintiaining it up to a higher frequency before the gain roll-off. The closed-loop gain was unnaffected by the change.

2.6 Multi-component variability

This task demonstrated how various parameter variations may be modeled through the use of parameter sweeps. The first part concerns how a single variable parameter may influence circuit characteristics, the second shows the effects of multiple parameters varying independently of each other and the third part demonstrates how a common ("global") parameter variation may compound the effects of existing variation parameters.

2.6.1 Single-parameter sweep

The static open-loop ("raw") gain parameter of the macro op-amp was replaced with a variable that was subsequently imported into `ADE L` and given a nominal value of 120 dB. The `Parametric Analysis` window of the `ADE L` simulator was then opened in order to give the gain variable a linear sweep range of 20 dB to 120 dB, with a step size of 10 dB.

The resulting plot can be seen in fig. 2.7. Despite different gain values, all of the plots conform to the same gain-bandwidth limit.

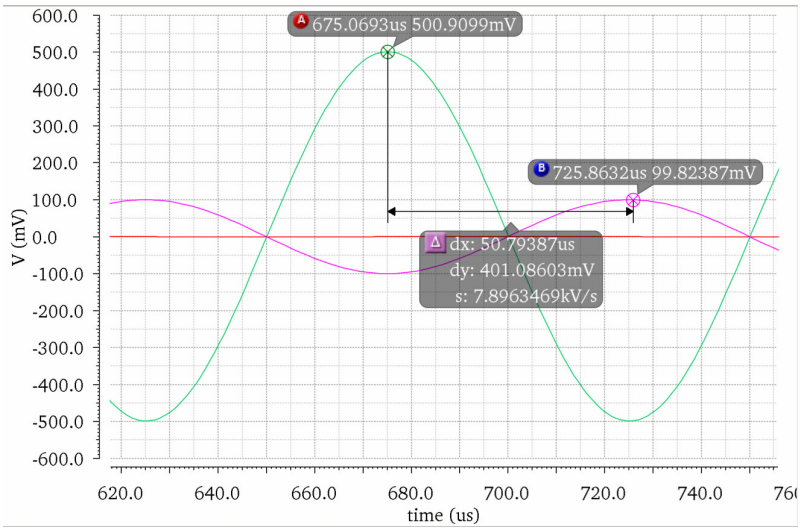


Figure 2.2: Transient analysis of the negative feedback amplifier showing closed-loop gain of -5.

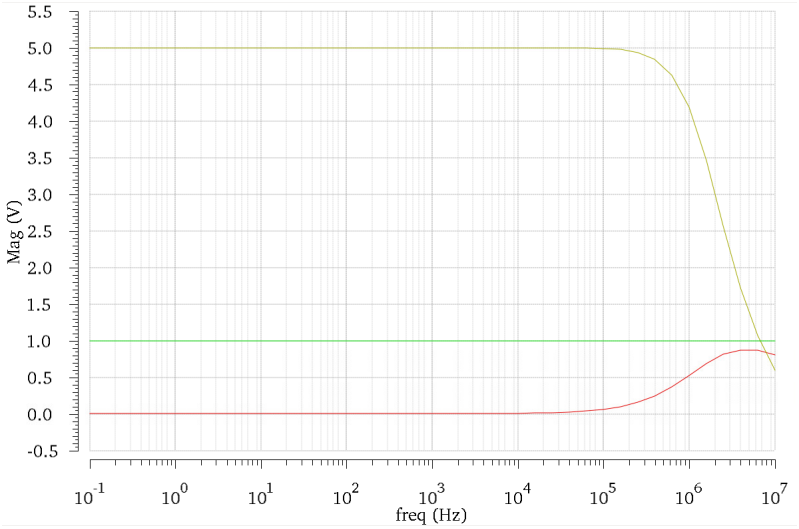


Figure 2.3: Bode plot showing the input (green), output (yellow) and negative amplifier input (red).

When examining the output magnitude of the amplifier, it is clear that the raw gain affects the precision. This is shown in fig. 2.8, where a raw gain of 40dB yields an error margin of less than 10%, whereas a raw gain of 50dB yields and error margin of less than 2%!

This simulation confirms what can easily be calculated by hand: let $\beta = \frac{R1}{R2}$, the *discrepancy* of a negative feedback amplifier is the actual output magnitude as compared to the "ideal" case where gain is infinite.

$$D = \frac{A\beta}{1 + A} \tag{2.3}$$

Low frequency values have been calculated in figure 2.9 using the above formula, showing that relatively little raw gain is needed to achieve high accuracy.

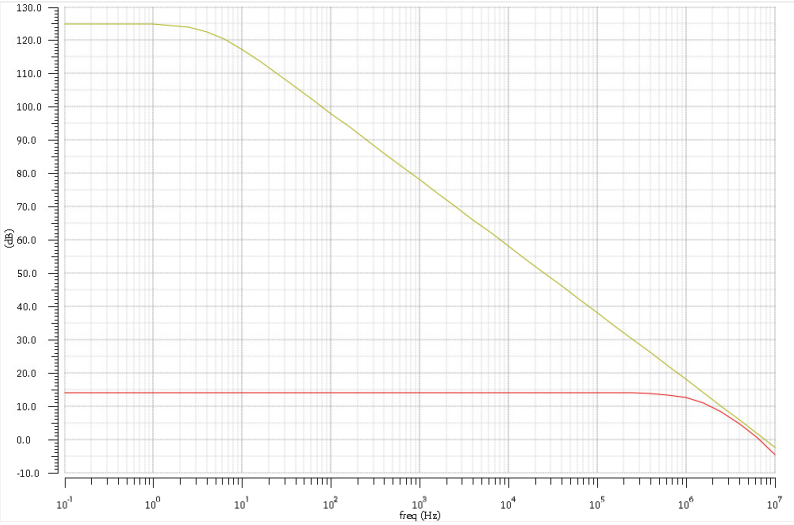


Figure 2.4: Bode plot showing the closed-loop gain (red) and the open-loop (raw) gain (yellow).

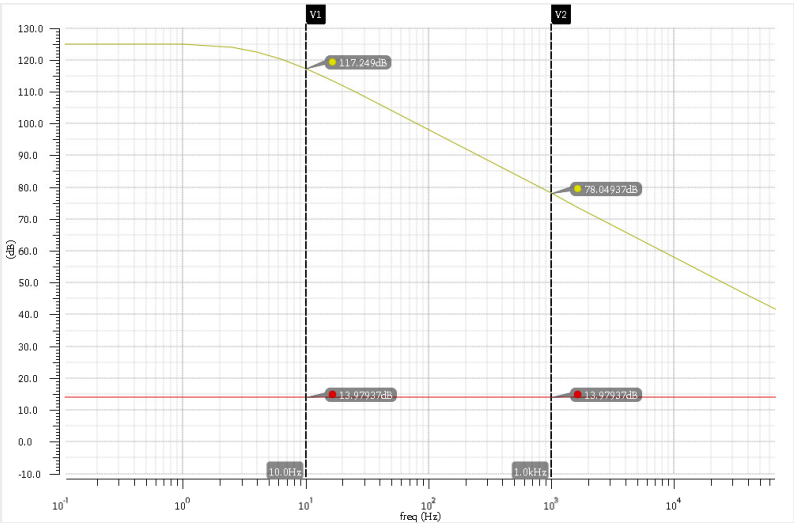


Figure 2.5: Bode plot, the raw gain fall in the range 10Hz to 1kHz

2.6.2 Multi-parameter sweep

In this section two new variables were introduced, replacing the previous values of both resistors with variables named `rin` and `rfb`. The variables were set to the `Range type`, in which steps are spread equally in a span around a specified center (the nominal value). The range in both cases was set to 20% (ie. it can vary 10% in either direction).

A new simulation was performed, with the results plotted in fig. 2.10, and the highest and lowest values were measured to be 6.11V and 4.09V, respectively. This matched the theoretical max and min given below.

$$A_{\mathrm{CL}_{\max}} = 1.1 \cdot 1.1 \cdot 5V = 6.09V$$

and

$$A_{\mathrm{CL}_{\min}} = 0.9 \cdot 0.9 \cdot 5V = 4.05V$$

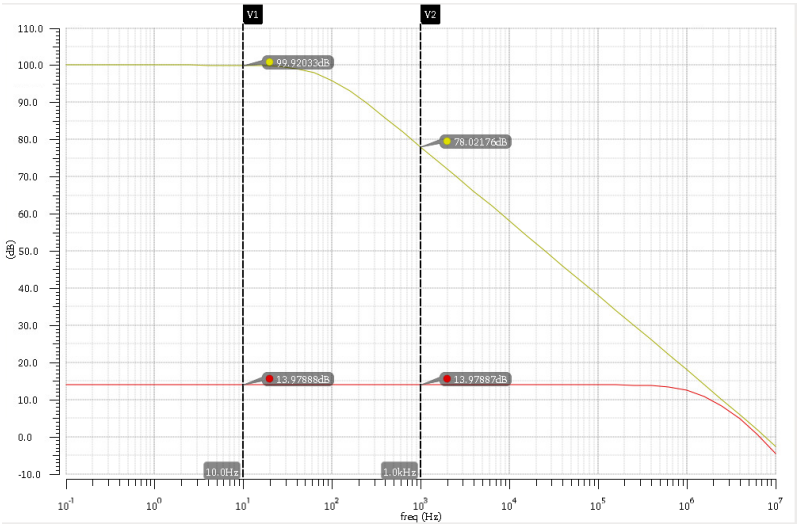


Figure 2.6: Bode plot, the raw gain fall in the range 10Hz to 1kHz with the AOLDC set to 100dB

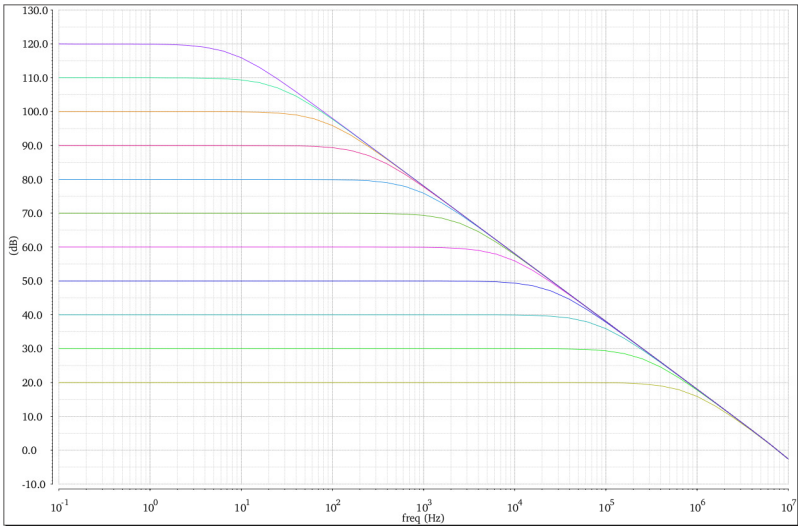


Figure 2.7: The open-loop gain of the opamp at different values of A_{OL}

2.6.3 Parameter co-variation

In this task a "global" parameter `rglob` was introduced, which simply multiplied the existing parameters by a common factor. `rglob` was varied by up to 20% while the other parameters varied by up to 2%. A number of simulation runs on the order of 30-40 was found to be reasonable in order to provide varied cases, while not taking an excessive amount of time.

Due to `rglob` being a common factor, its effect was cancelled out by the feedback network β, sine it is only the *ratio* of the resistors in the network that is important. So despite the large global variation, the actual variation in the output was only

$$A_{\mathrm{CL}_{\max}} = 1.01 \cdot 1.01 \cdot 5V = 5.1005V$$

and

$$A_{\mathrm{CL}_{\min}} = 0.99 \cdot 0.99 \cdot 5V = 4.9005V$$

This is confirmed by the plot in fig. 2.11.

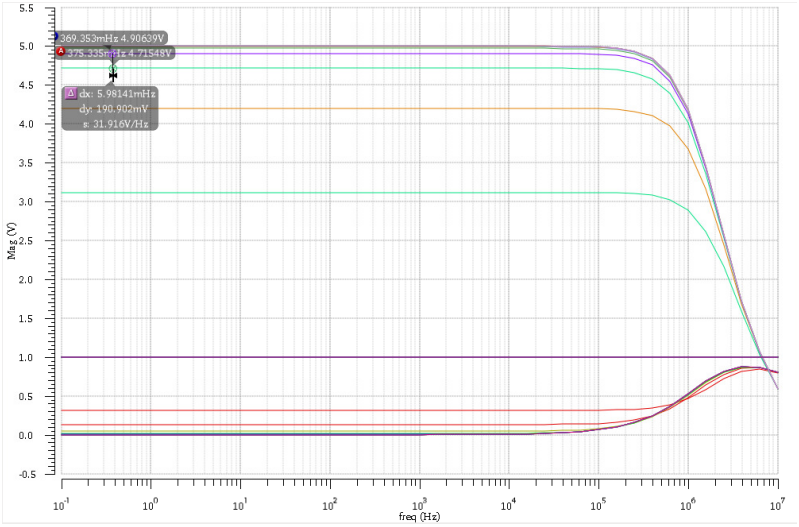


Figure 2.8: Bode plot showing results of A_{OL} sweep.

A_OL	D
120	1
110	0,99998
100	0,99995
90	0,99984
80	0,9995
70	0,99842
60	0,99502
50	0,98443
40	0,95238
30	0,86347
20	0,66667

Figure 2.9: The discrepancy is shown to decrease as A_{OL} increases.

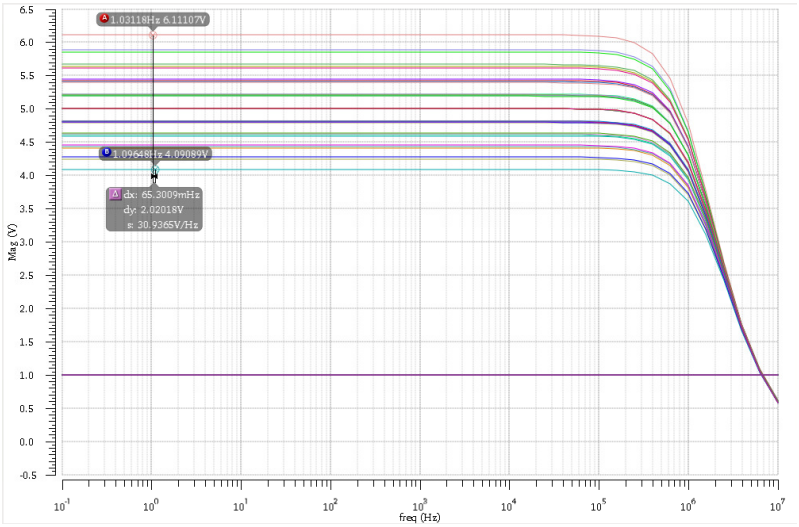


Figure 2.10: Bode plot showing the effects of sweeping the resistance values of the resistors that comprise the feedback network.

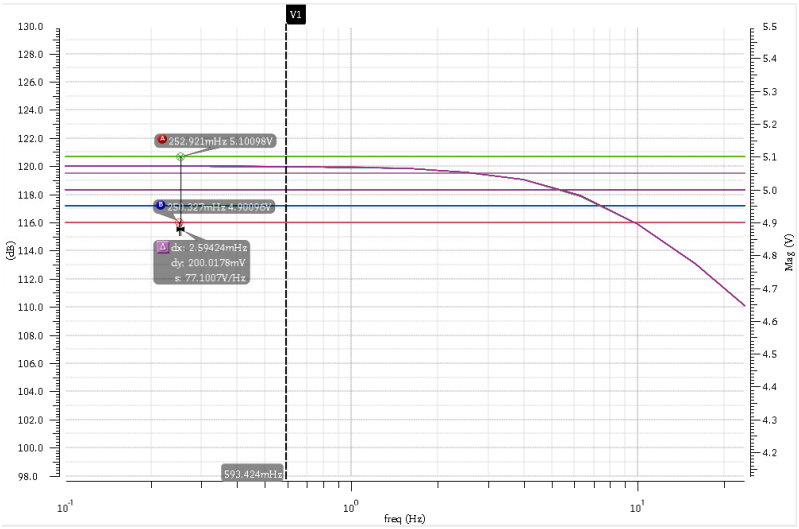


Figure 2.11: Effects of sweeping resistors with a local and global parameter.

2.7 Reflection Questions

2.7.1

Assuming an inverting feedback amplifier with an ideal feedback network (no variation in the resistive divider), the raw gain needed to achieve an error margin of no more than 1% at an intended closed-loop (CL) gain of 10 can be calculated using eq. 2.3.

Rearranging for A,

$$A_{\mathrm{OL}} = D/(\beta(1-D)) \tag{2.4}$$

A CL-gain of 10 corresponds to $\beta = 1/10$, while the discrepancy is 99%. This gives a raw gain $A_{\mathrm{OL}} \approx 10^3$.

For an intended CL-gain of 100, $\beta = 0.01$ and $A_{\mathrm{OL}} \approx 10^4$.

2.7.2

A higher closed-loop gain will require a larger resistor ratio in the feedback network. The larger resistor is more susceptible to the type of parameter variation seen in section 2.6.2.

2.7.3

Given a nominal resistance R_1 and width W_1 for a given standard deviation of random mismatch s_R between two resistors (not necessarily of the same value), Hastings argues that both resistance and width are inversely proportional to the mismatch; that is, if one parameter increases then the other may decrease to obtain the same mismatch and vice-versa.

Thus larger resistance values will require less width for the same matching properties.

Hastings also posits that the level of mismatch is influenced by two constants, the *areal contribution* k_a and the *peripheral contribution* k_p. These constants, depending on which one dominates, will determine the range of mismatch.

Thus for best matching with a minimum areal budget, a higher resistor ratio should be used at each stage.

LAB

3 Quantization and jitter

3.3 Uniform Quantization

The model was set to run for 1s with a fixed-step interval of 1/1024. The sine source was accordingly set to $\omega = 2\pi 5$ with an amplitude of 1. The generated sine signal was subsequently fed into a `1-D LUT` which which uses *breakpoints* to relate the incoming sine value to a discrete value in the table. The breakpoints and associated values were generated by the functions `lutix(lvls)` and `lutdata(lvls)`, respectively.

This effectively discretizes the analog sinusoid to a number of levels (`lvls`) with values set by `lutdata` and which transition at breakpoints whose values are set by `lutix`. En example of this with 15 discrete levels can be seen in fig. 3.1.

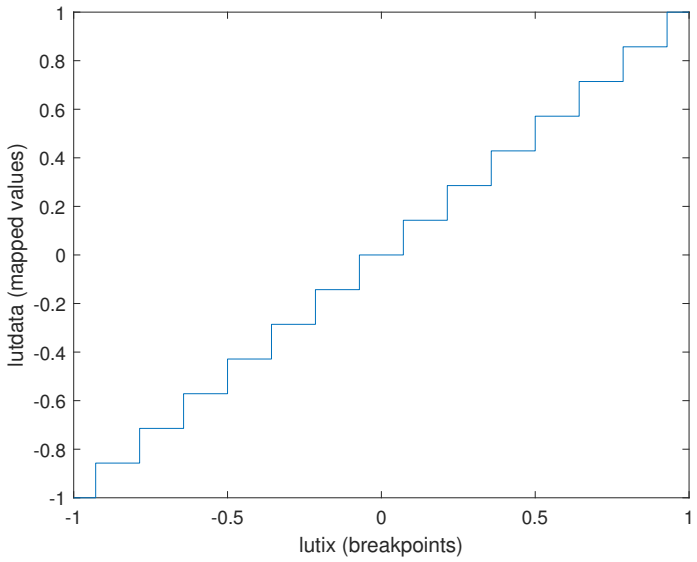
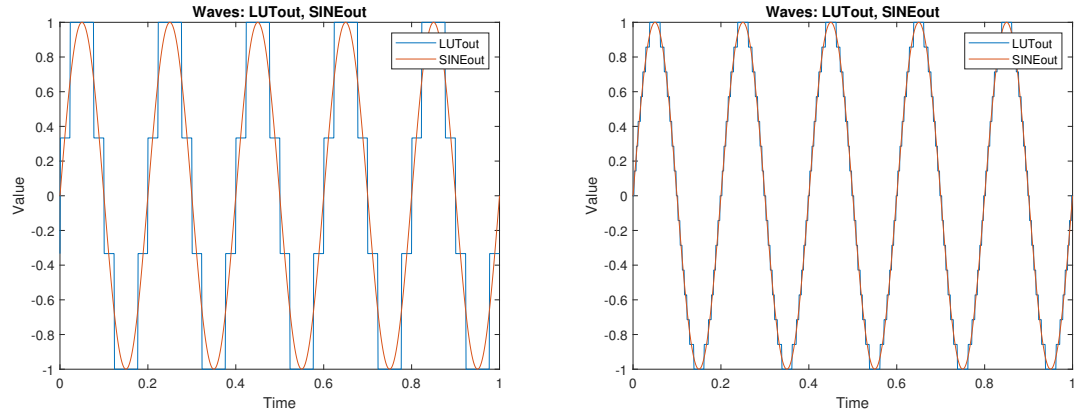


Figure 3.1: `lutdata` w.r.t. `lutix`. This shows how an input signal will be mapped in the range -1<x<1 (input values outisde this range will result in clipping)

In the time-domain, this predictably results in a discontinuous sinusoid, as seen in fig. 3.2, where an increase in levels from 4 to 15 is illustrated.



(a) 4 discrete levels.

(b) 15 discrete levels.

Figure 3.2: Using an LUT to map a pure sine input results in a quantized output with a discrete number of levels.

An example of the spectral content of a quantized signal is given in fig. 3.3. The fundemental frequency component is clearly higher than the rest, but in the presence of significant noise introduced by the quantization process, which can thus be termed "quantization noise". Half of the noise components have a higher power than that of the noise floor, which is characteristic of square-wave type signals.

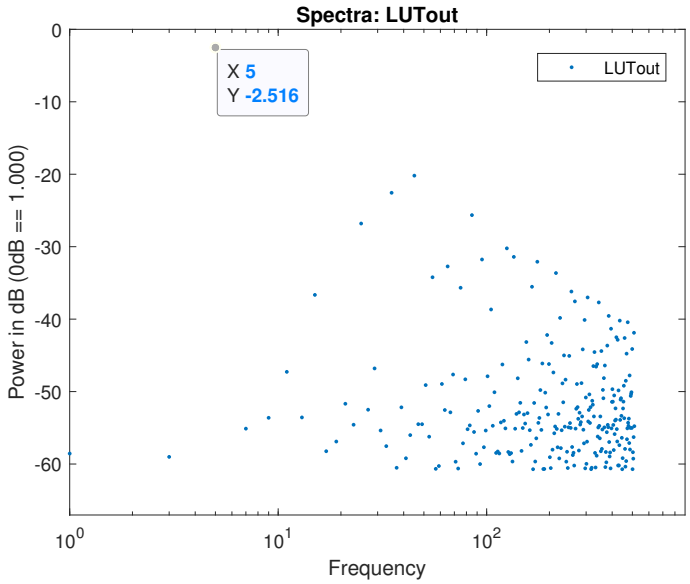


Figure 3.3: Thee frequency spectrum of the 5 Hz sine wave after quantizing to 4 levels. *N.b. "rounding noise"-floor omitted*

From the frequency spectrum the SNR of the quantized signal can be derived. This was done for a range of *bit resolutions* N, from 3 to 14 bits. It was previously mentioned that `lutix(lvls)` and `lutdata(lvls)` take the number of *levels* as an argument; N must therefore be converted to the number of levels they can represent using the simple relationship lvls $= 2^N$.

The resulting SNR for each resolution is plotted in fig. 3.4 and just as expected, the relationship is linear and conforms to the approximate rule-of-thumb (for pure, full-swing sine waves)

$$\mathrm{SNR_{dB}} = 6.02N + 1.76 \tag{3.1}$$

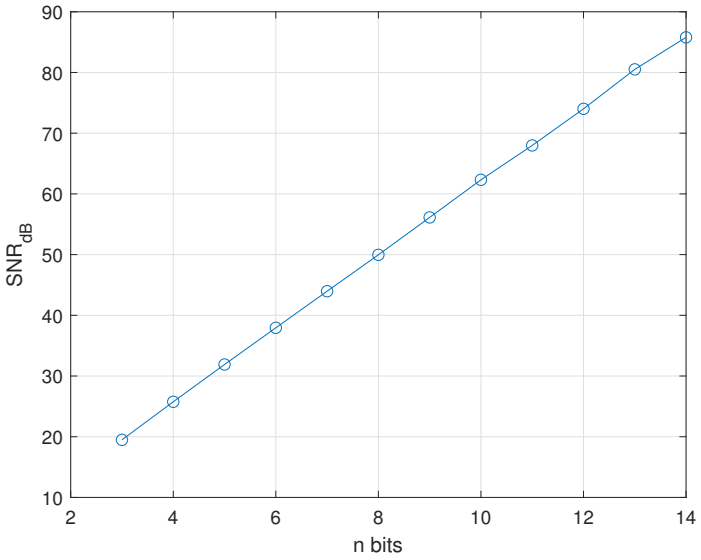


Figure 3.4: SNR increases linearly with the bit resolution N.

3.4 SNR vs input power

This time the model was supposed to vary according to changes of a new variable defining the value of the sine wave amplitude. The number of levels in the quantizer was fixed to 1024, corresponding to a resolution of 10-bits.

The amplitude was varied in order to regulate the input power through the following relationship

$$P_{\mathrm{sine}} = \mathrm{A}^2 \cdot \mathrm{avg}(\sin^2(\omega t)) = \frac{A^2}{2} \tag{3.2}$$

First, using the above equation, the power of the reference input signal, that is, a sine wave of amplitude of 1 can be calculated and is equal to 0.5.

$$A = \sqrt{\frac{10^{\frac{P_{\mathrm{in}}+P_{\mathrm{ref}}}{10}}}{0.5}} \tag{3.3}$$

The input signal is *referred*, that is, normalized, to a full-swing sine signal with a power amplitude of 0.5, or -3.01dB. In logarithmic terms, this corresponds to some form of subtraction between the input and the reference powers so that the sum is the reference power when the input power is zero.

The input power was then swept from -20dB to +3dB, using 0.5dB steps and the result normalized to the reference power. This results in the curve as shown in fig. 3.5. The point at which $\mathrm{P_{in}}$ exceeds $\mathrm{P_{ref}}$ is clearly visible as the the SNR sharply declines. This is due to the input amplitude exceeding the full range of the quantizer (clipping), and the resulting large discrepancy between the input and what can be represented on the output.

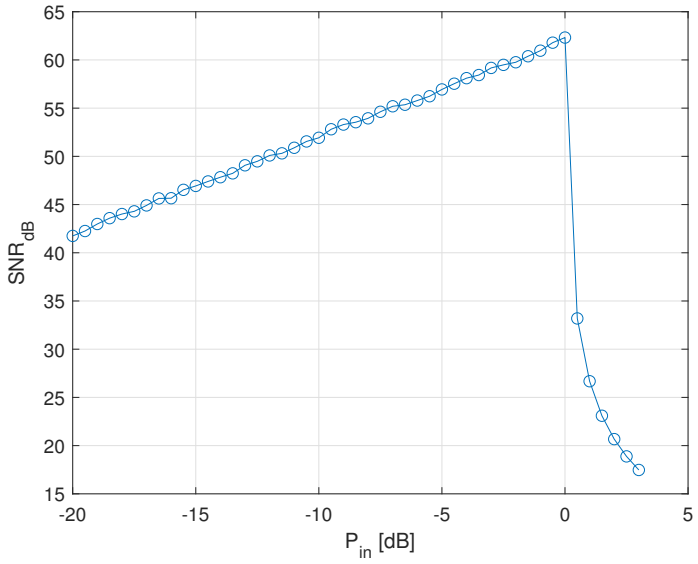


Figure 3.5: P_{in} normalized to full-range sine power.

3.5 Non-uniform quantization

The `1-D LUT` parameters had to be changed in order to simulate the behaviour of the R2R ladder converter. Therefore, instead of using an *analog-like* floating point input values the breakpoints were set to the range of integers varying from -2^{N-1} to $2^{N-1}-1$. The table data was set to values generated with a function `r2r(N)` corresponding to voltage outputs of N bit ladder in range -1 to 1 inclusively.

As the gain should not exceed the breakpoints range, it was set to $2^{N-1}-1$, that, in this case, is equal to 511.

The used function can be extended with another parameter `sigma` defining the standard deviation of ladder resistors. This feature allows to create noisy deviations, with which the model imitate well the differential non-realities - a typical error in the D/A converters, in particular the R2R. In a similar manner to the previous subsection, new plots of SNR in function of normalized signal input were obtained for standard deviations of 2% and 4%.

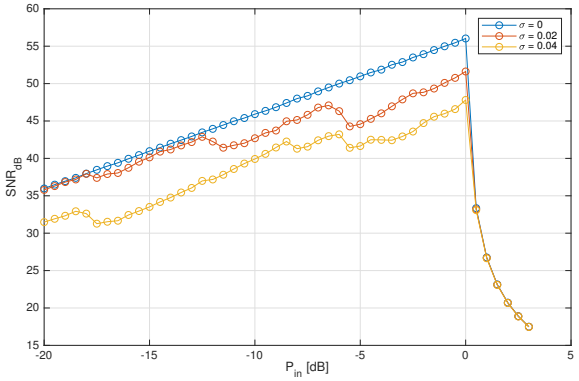


Figure 3.6: SNR in function of normalized P_{in} for various standard deviations.

As expected, the higher sigma is, the more power noise is ergo the SNR in decibels becomes smaller.

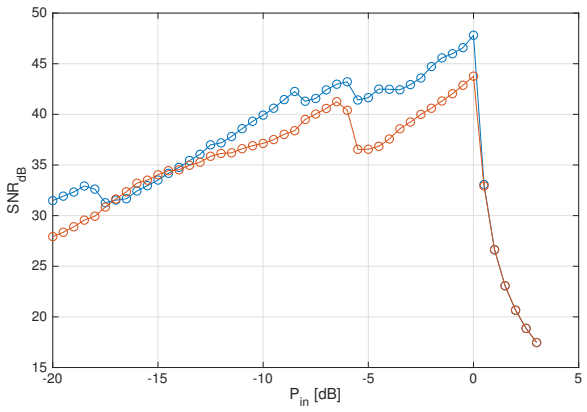


Figure 3.7: SNR in function of normalized P_{in} for two different executions with standard deviation of 0.04.

There is a clear pattern how the sigma factor influences results - curves obtained in disturbed quantization are similar to the pure one but shifted down and slightly inclined. Moreover, their fluctuations get more significant along with a growth of the standard deviation.

What is worth mentioning, the `r2r` function bases on randomly generated deviations and differs with each execution of the function.

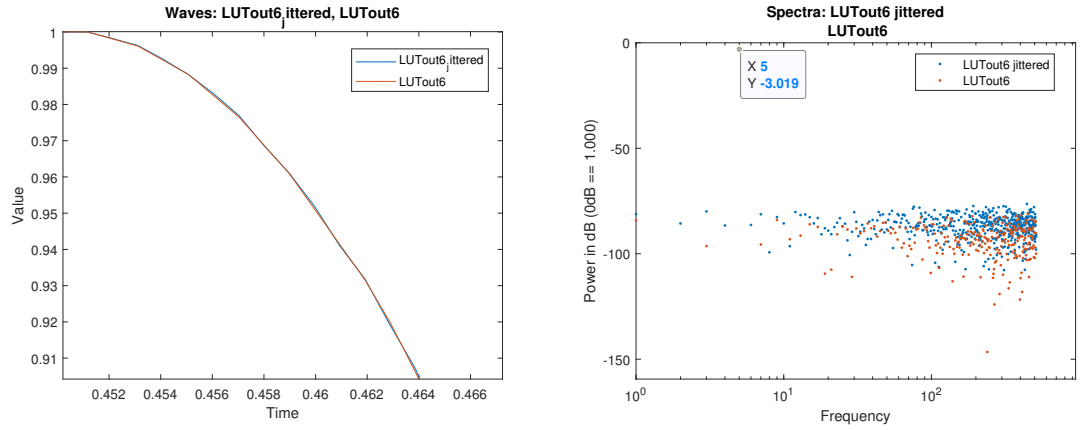
3.6 Sample jitter

In this task sample jitter (sampling that does not occur at precisely uniform time intervals) is modelled. Since the `simulink` model is inherently *fixed-step*, that is to say the sampling intervals can not be directly manipulated. However, this jitter in time results in the derivative of the sampled signal being affected, which is conversely relatively simple to model by using a `Uniform Random Number` generator to scale the *derivative* of the input signal and then adding this offset to the signal itself.

As in the previous task, the input signal is a pure sine wave of amplitude 1 and frequency 5 Hz quantized by a 10-bit *r2r* ADC, while the model uses 1024 time steps and runs for 1 second. In this case, the *r2r* resistors were considered ideal, with no variation.

To simulate the jitter, the `Uniform Random Number` block is set to have a deviation of $\pm 10\%$ of the sampling interval ie. $\pm 0.1/1024$.

Both the "pure" and jitter-affected quantized waveforms are shown in the time and frequency domains in fig. 3.8. Differences in the time domain are nearly imperceptible, but are clearly apparent in the frequency domain, with the "noise floor" occurring at distinctly different levels.



(a) Time domain

(b) Frequency domain

Figure 3.8: Ideal 10-bit quantized sine wave before and after time-jitter.

To show the effect jitter has on signals of higher frequency, the input signal was increased to 50 Hz. A comparison of the frequency spectra the jittered 5Hz and 50Hz signals is illustrated in fig. 3.9. The higher frequency signal has a distinctly higher noise floor, which is to be expected as higher frequency signals by definition have greater derivatives, and so are more sensitive by a time offset. This in turn gives a greater discrepancy between the ideal and jittered signal, increasing apparent "jitter noise".

The SNR for the original 5 Hz signal without jitter is 56.0dB, whereas the jittered signals at 5 Hz andd 50 Hz have an SNR of 52.5dB and 35.1dB, respectively. This is in accordance with the theoretical jitter analysis, which states

$$\mathrm{SNR}_{dB} = -10\log_{10}((2\pi f)2 \cdot \mathrm{avg}((\Delta t)2)) \tag{3.4}$$

In other words, SNR decreases with increased frequency.

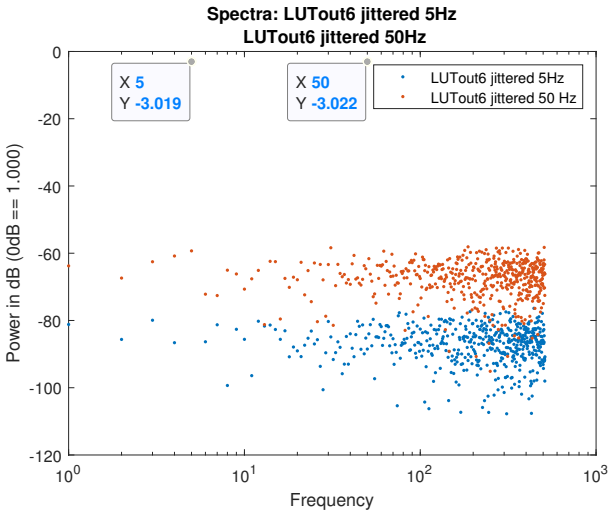
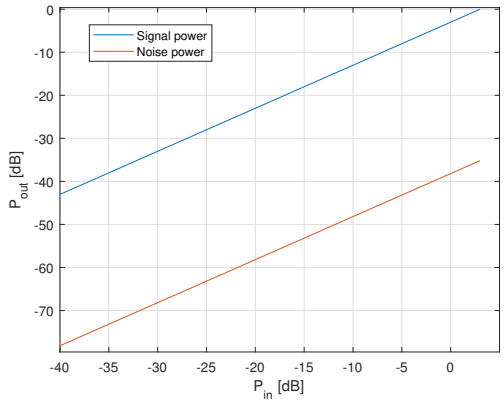
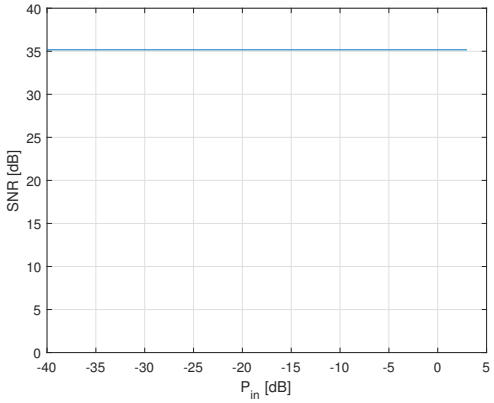


Figure 3.9: Signal spectra for a time-jittered sine wave at 5Hz and 50Hz.

Finally, the power of the 50 Hz input signal was swept from -40dB to +3dB and the output signal and noise powers plotted (normalized to a full scale sine wave, just as in task 4), as shown in fig. 3.10 a). These two curves can then be used to derive a curve representing the SNR, seen in fig. 3.10 b), which appears to be independent of P_{in}.



(a) Powers of fundamental component and noise as function of input power.



(b) SNR as function of input power.

Figure 3.10

3.7 Reflection Questions

3.7.1

Let the signal have an amplitude of 1 (this is arbitrary, the absolute value of the signal is irrelevant). Assuming that there are two noise sources of equal magnitude and a specified SNR of 50 dB, we can express the SNR as the following expression:

$$\mathrm{SNR_{dB}} = 50\mathrm{dB} = 10\log_{10}\left(\frac{1}{N_1 + N_2}\right)$$

where N is the noise *amplitude*, ie $10^{\frac{N\mathrm{dB}}{10}}$.

If one of the noise sources increases in power then the other must decrease in order to maintain the same SNR. Since one of the noise sources is know to be -52 dB relative to the signal, the value of the other noise source can be calculated by rearranging the formula

$$\begin{aligned}\mathrm{SNR_{dB}} = 50\mathrm{dB} &= 10\log_{10}\left(\frac{1}{10^{\frac{-52dB}{10}} + 10^{\frac{-x\mathrm{dB}}{10}}}\right)\\ \Rightarrow x &= -10\log_{10}\left(10^{\frac{-50dB}{10}} - 10^{\frac{-52dB}{10}}\right)\\ &= 54.33\mathrm{dB}\end{aligned}$$

In other words, the other noise source must be -54.33 dB relative to the signal.

3.7.2

The R-2R ladder uses binary scaling and hence higher bits are more significant than lower ones. This leads to a typical DNL plot wherein the output suffers significant deviations when high-significance bits flip. Since larger area is improves matching, the area budget should be allocated in proportion to the significance of the bit that the resistor(s) represents. The area should, in other words, increase along the ladder from LSB to MSB.

LAB 4

Continuous-time filter design

4.3 Filter order

The four classic types of filter were examined in this task and cursorily tested for suitability before proceeding to the final implementation. These are the Butterworth, Chebyshev 1 and 2 and the Cauer filter which each have different characteristics. The simple box-specification in fig. 1 of the Lab PM was entered into MATLAB as two points -2dB at 5 kHz and -35dB at 9.5 kHz. These points define the "corners" that the frequency response of the filter must not cross from above and below, respectively.

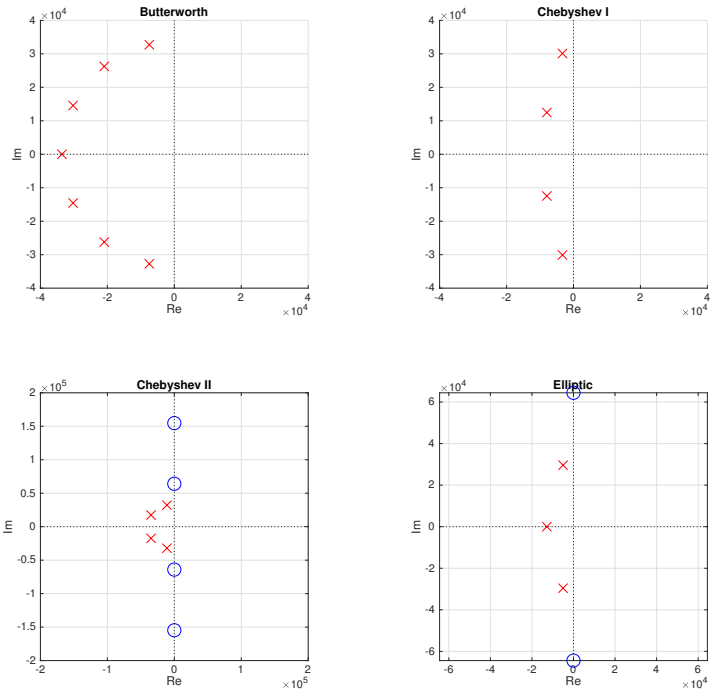


Figure 4.1: Poles (red) and holes (blue) on the s-plane for the minimal implementation of the filter for each classical type.

The lowest order of filter that fulfils the specification as well as the associated cutoff frequency f_c for each type was determined using the functions `buttord`, `cheb1ord`, `cheb2ord` and `ellipord`. Using this information as well information about the level of tolerable ripple as input arguments to the commands `butter`, `cheby1`, `cheby2` and `ellip`, the complex values of the zeroes and poles as well as the gain that can be used to implement filters conforming to the specification was computed. This resulted in the plots seen in fig. 4.1.

The criteria for proceeding to a hardware implementation using a 2^{nd} order Sallen-Key topology was that the filter order should be the lowest available whilst not having any zeroes; as such, the Chebyshev1 implementation was chosen.

4.4 Transfer function selection

Using the zeroes and poles computed in the previous task, the rational, polynomial form could be derived using the function `zp2tf` allowing the use of `freqs` to return the complex transfer function. From this, the magnitude function was derived and plotted in fig. 4.2. Zooming in on the plot confirmed that the original specification has indeed been fulfilled.

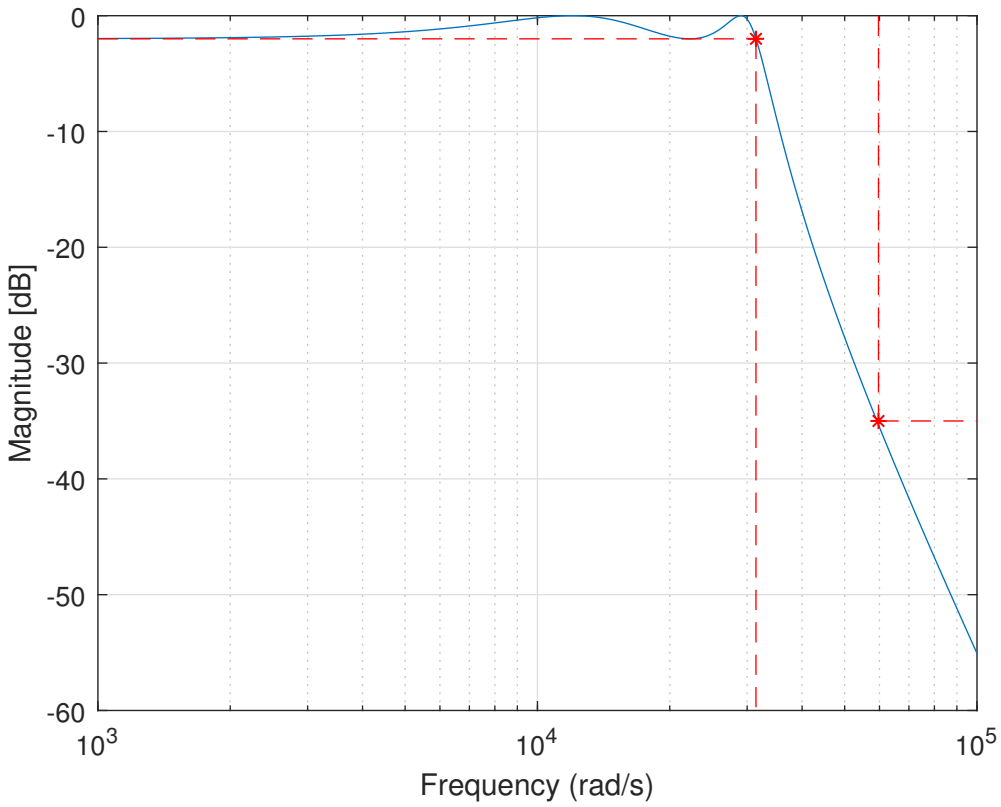


Figure 4.2: Chebyshev1 magnitude response. Box-specification overlaid in red.

4.5 Circuit implementation

The transfer function chosen in the previous subsections contains two complex-conjugate pole pairs (see the Chebyshev1 plot in fig. 4.1). As each of such a pair can be put into practise with a use of a single Sallen-Key topology section, we were able to realize our filter in the form of a two-level cascade. The Sallen-Key topology used in the implementation consists of a basic non-inverting amplifier and two identical resistor-capacitor pairs. The basic non-inverting amplifier contains two resistors of such values that $\frac{R_3}{R_2} + 1$ matches the expected gain of a considered stage. The RC pairs stand for controlling the pole frequency. Both the RC pairs' resistor and the amplitude are calculated using formulas derived from the Sallen-Key topology and based on complex values of poles p_1 and p_2 of a certain pair.

$$R = \frac{1}{C\sqrt{p_1 p_2}} \tag{4.1}$$

$$A = 3 + \frac{p_1 + p_2}{\sqrt{p_1 p_2}} \tag{4.2}$$

In our case, the complex-conjugated pole pairs were $-0.7955 \pm 1.2466i$ and $-0.3295 \pm 3.0095i$, the calculated values of the RC pair resistors were $6.762k\omega$, $3.303k\omega$ and the amplitudes 1.9241,2.7823, respectively.

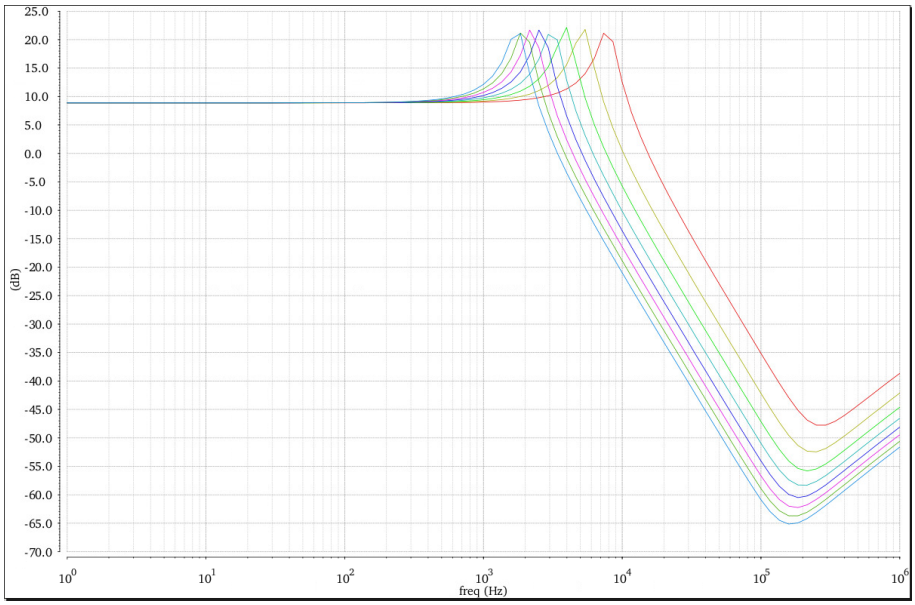


Figure 4.3: $|H(\omega)|$ when sweeping *rbp*.

A testbench was created with a sine source, an `sk` instance (the Sallen-Key circuit described above) and a 10 kΩ load. A `config` view was created to allow the simulator to simulate it as a mixed-signal model using `ams`. As previously stated, the RC pairs on the input to the amplifier control the pole frequency. This is evident when sweeping the value of R, given by the variable *rbp*, in the kΩ range, causing a shift in f_c as illustrated in fig. 4.3.

Likewise, the gain of the opamp was swept, giving the plot in fig. 4.4.

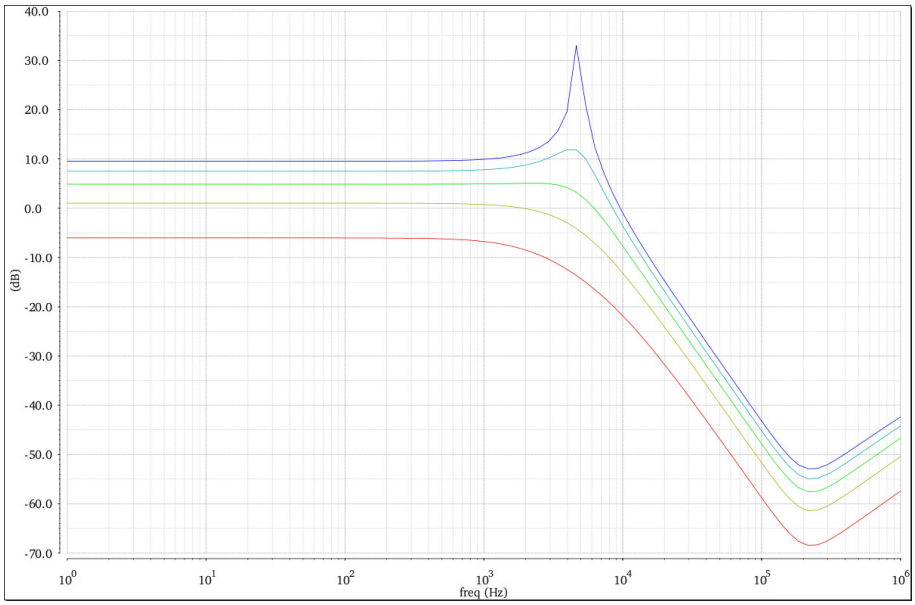


Figure 4.4: $|H(\omega)|$ when sweeping *gain*.

Next, two instances of `sk` with the nominal values calculated previously were cascaded to form the 4^{th} order Chebyshev1 filter. The overall gain was larger than 0 dB and a resistive divider was designed to attenuate the signal. The maximum magnitude was empirically determined to be approximately 6.75 dB, whereas the output of a resistive divider is given by the expression $fracR_2R_1 + R_2$. Solving for R_1 and arbitrarily giving R_2 a value of $10\,\text{k}\Omega$,

$$R_1 = R_2(6.75 - 1) = 57.5\,\text{k}\Omega \tag{4.3}$$

The attenuated output of the final Chebyshev1 filter is shown in fig. 4.5. Zooming in, it becomes apparent the filter is not completely within the bounds of the specification, though the discrepancy was deemed so marginal as to be insignificant.

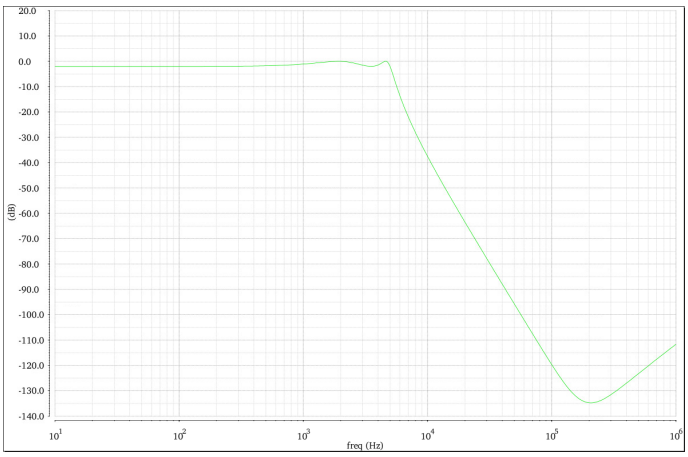


Figure 4.5: $|H(\omega)|$ for the 4^{th} order Chebyshev1 filter after output attenuation.

4.6 Component inaccuracies

In order to test out how the designed filter would behave with inaccuracies which are able to occur in a real production process the Monte-Carlo simulation was held. The schematic was redone with a use of the `sk1` cells in a place of the previous `sk` ones and initialized with the same parameters. The difference between the two filter models is that the latter has additional parameters that can be controlled.

Instead of the usual **ADE L** simulator, the **ADE XL** simulator was used for its inclusion of Monte-Carlo simulation capabilities. After setting up the schematic and simulation parameters, an AC analysis of the same type as previously was performed using the `Debug Test` function in order to do a functional check before proceeding. The main purpose of this more advanced simulator is to be able to change the parameters of components in the filter circuit according to a statistical probability function, provided by the script file `variations.scs`.

The actual Monte-Carlo simulation could then be executed. The statistical model allowed for three different cases, of which two were used: `mismatch` and `all`. The simulator used the statistical definitions to change component parameters between a number of runs, resulting in multiple waveforms being plotted. The mismatch case resulted in the plot in fig. 4.6. In this graph the dominant feature is simply the gain of the amplifier. Since the gain is determined by a ratio of resistors, the response is not particularly sensitive.

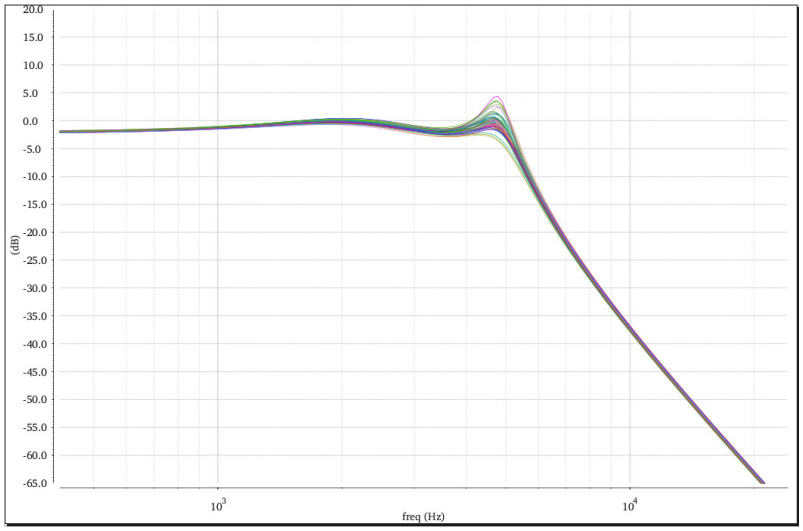


Figure 4.6: Monte Carlo simulation using the *mismatch* case.

In fig. 4.7, however, the capacitance of the circuit has been varied to a greater degree and since RC multiples appear in the expression for the Sallen-Key magnitude response, any global errors are in a sense squared.

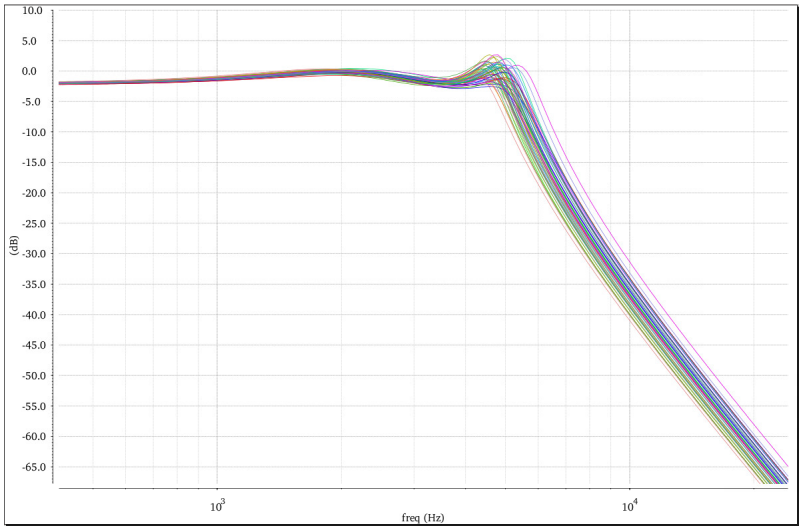


Figure 4.7: Monte Carlo simulation using the *ALL* case.

4.7 Reflection Questions

4.7.1

We can attempt to use fewer values of passive components so that they may be better matched in the *layout* stage. This has costs in that specific values may have to be broken down into multiple "standard values", causing precision to suffer. To increase the precision, more filter stages may then have to be used.

4.7.2

LAB 5 | Nonlinearities in continuous-time circuits

5.4 Cadence-MATLAB data transfer

The OP-amp based inverting amplifier from Lab 2 was launched in Cadence `Virtuoso` and the voltage source parameters were edited to give a frequency of 10 kHz and an amplitude `ampl`. In contrast to earlier experiments, the simulator was explicitly set to use `conservative` accuracy for transient simulations.

The simulator was instructed to simulate a 1ms run, and to give `ampl` a value of 0.1 V. The results were then imported into MATLAB using the function `cds-srr` and then converted from the proprietary Cadence format into a data structure that MATLAB can natively process using the function `cds2sig`. This allowed the simulation results to be analysed in MATLAB, giving the `bar` plot in fig. 5.1.

The spectrum of `bar` conforms to our expectations, as the output of the amplifier has a signal amplitude of 0.5V, which gives a power component of $10\log_{10}(\frac{0.5^2}{2}) \approx -9\,\text{dB}$.

To explore the effects of different simulation tolerance parameters, which control the tradeoff between accuracy and simulation speed, the parameters `reltol` and `restol` were reduced and the simulation repeated, giving the spectral plot `baz` in fig. 5.1. Clearly, there has been a marked improvement in the level of the "noise floor" caused by numerical artifacting (ie. "rounding noise" seen in previous labs), decreasing from -160 dB to -225 dB, a 75 dB improvement.

The cost of this increased simulation accuracy is, however, the computational power required. The default accuracy parameters resulted in 988 transient steps and 35ms of CPU time required. The increased accuracy parameters resulted in 14926 transient steps and 302ms of CPU time. In summary, the simulation time increased tenfold whereas the number of saved plot points in the result increased by a factor 15! Both CPU and memory resources may be strained as a result of high accuracy simulation.

Surprisingly, the SNR of the signal was unaffected by the accuracy difference. Likewise the DC component in both cases occurred at -60.68 dB, far above the noise floor. This DC component occurs because of miniscule currents on the OP-amp inputs. After ommitting the DC component, the SNR rose to 98.29 dB and 160.95 dB for the the low and high resolution signals, respectively. This corresponds fairly well to the difference in noise floor level between the two simulations.

5.5 SNR as function of input level

The simulator **ADE L** was set to sweep the input amplitude logarithmically from 1mV to 10V. The output waveforms are shown in fig. 5.2. Clipping occurs at high input levels (beyond approx. 2.3V). The maximum values at the output, made clearly visible by the clipping, are ± 13.8V. These correspond to the maximum output swing, dependent on the supply voltages. These limits

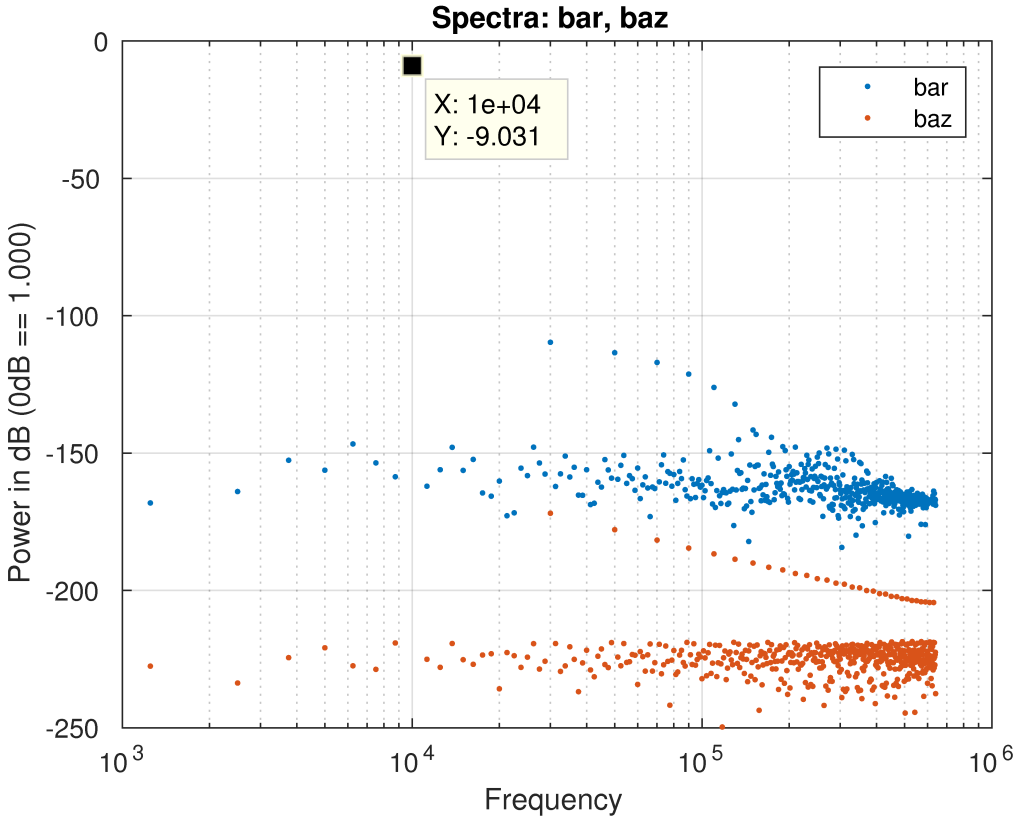


Figure 5.1: Simulations performed (in `Virtuoso`) with difference tolerance settings.

are also given directly as the explicit parameters `VLIMP` and `VLIMN` within the properties of the OPamp model in `Virtuoso`.

The simulation data from all the waveforms was imported into MATLAB and the SNR was computed for each waveform individually before plotting the results against the corresponding input amplitude, shown in fig. 5.3. The increase appears to be logarithmic up until the clipping region, in which it rapidly declines. The fact that the SNR appears to increase at the end of the sweep is simply due to the analysis method employed, which in this case has interpreted a noise component (which is now larger than the signal component!) as the signal of interest. In reality, the SNR continues to decline.

5.6 Soft nonlinearity

The OP-amp model used in the simulation was replaced by a similar model containing a gradual nonlinearity. A new input sweep was performed, this time with the maximum input reduced to 2.5V to avoid clipping, and the SNR relationship plotted in fig. 5.4a as previously. In contrast to the plot without nonlinearities in fig. 5.3, the relationship between input amplitude and SNR is logarithmic at low amplitudes (below 1V) but starts to decline around that point, even without clipping.

To inspect the harmonic content of the signal, the powers of the fundemental component and first nine components were plotted as a function of input amplitude in fig. 5.4b using the function `sigharms`. From this we can observe that the second harmonic appears to dominate. Additionally, higher order harmonics appear to rise faster with input amplitude.

The open-loop gain of the OP-amp model was then reduced to 20 dB and the experiment repeated.

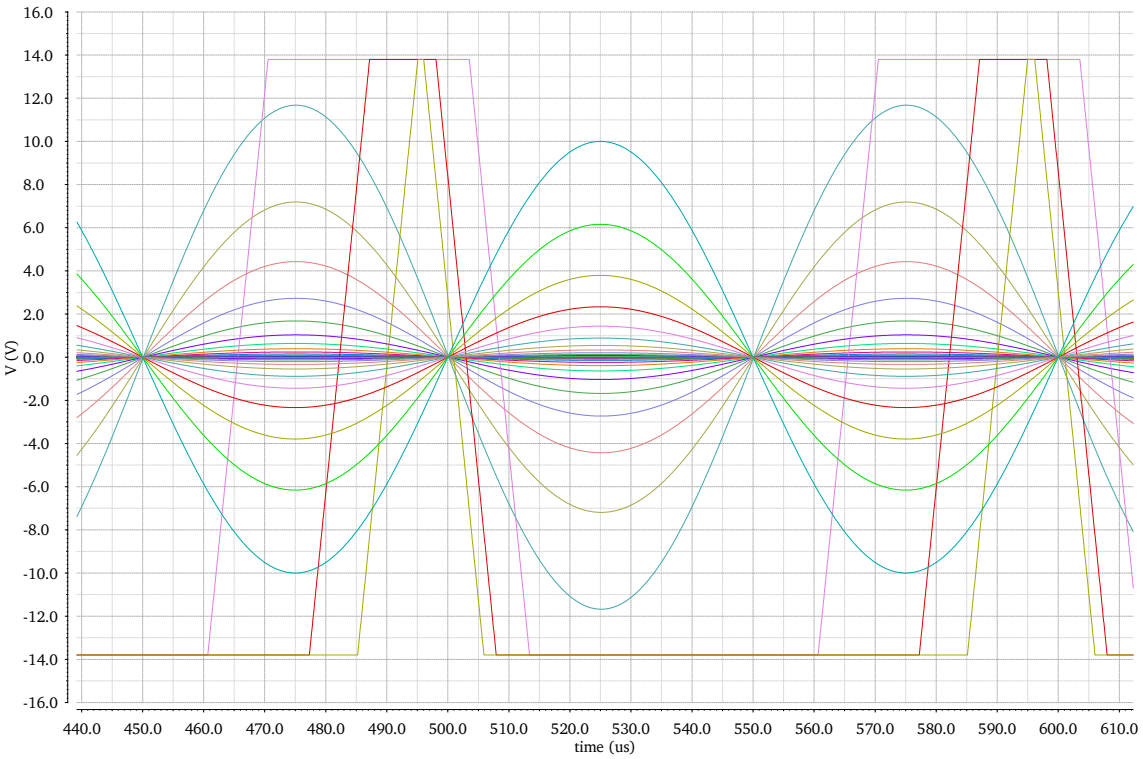


Figure 5.2: Output waveforms as a result of input amplitude sweep.

This resulted in the plots seen in fig. 5.5.

The SNR has maintained its logarithmic relationship at low input amplitudes but now starts to decline earlier, peaking at approx. 61 dB rather than the 68 dB seen previously. The 1^{st} and 2^{nd} harmonics are now much more prominent.

5.7 Reflection Questions

5.7.1

We have seen that a lower gain increases the power of the two first harmonics. Placing such an amplifier first in a cascade would only amplify these harmonics, reducing the overall SNR. Hence, amplifiers with high gain should be placed first and amplifier with low gain placed last.

5.7.2

Distortion cases signal *compression*, which introduces nonlinearities. This will adversely affect the final gain if the nonlinear signal is allowed to propagate through a cascaded amplifier as the nonlinearities will be amplified.

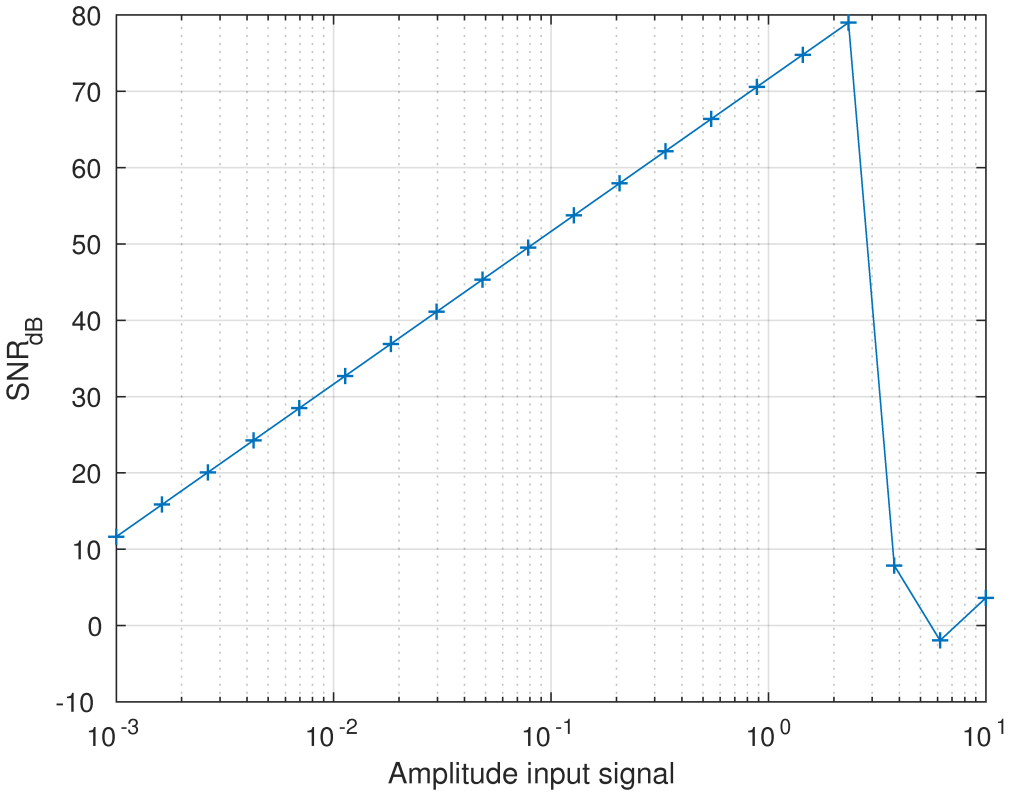
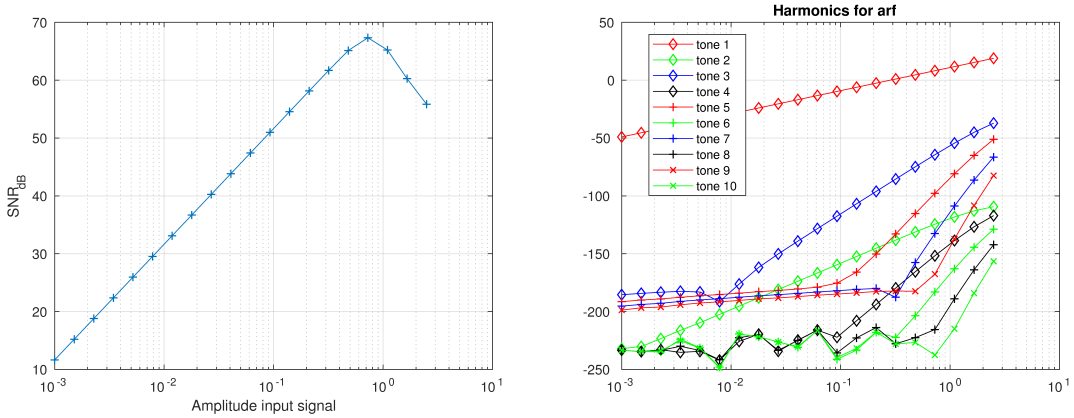
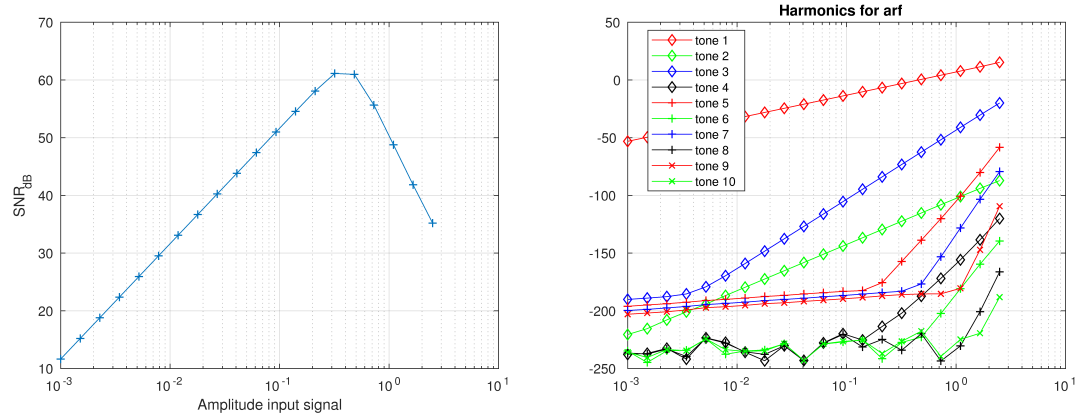


Figure 5.3: SNR vs. input amplitude without nonlinearities.



(a) SNR vs. input amplitude with nonlinearities. (b) Signal harmonics as a function of input amplitude.

Figure 5.4: Input amplitude sweep



(a) SNR vs. input amplitude with nonlinearities. Open loop gain reduced.

(b) Signal harmonics as a function of input amplitude. Open loop gain reduced.

Figure 5.5: Input amplitude sweep with reduced open-loop amplifier gain.

LAB 6 | Oversampling and noise shaping

6.3 Oversampling

A simulink simulation was set up with 1024 samples, 5 full-swing sine periods and a total run-time of 10s. An ideal A/D converter was set up using the same 1D-LUT as in previous labs (using `lutix` and `lutdata`) with 2^7 levels, ie. 7 bits of quantization.

To confirm that the quantized signal is as expected, the output power spectrum was examined to find the signal component at approx. -3 dB at 0.5Hz, with the quantization noise floor at approx. -75 dB. This is illustrated in fig. 6.1.

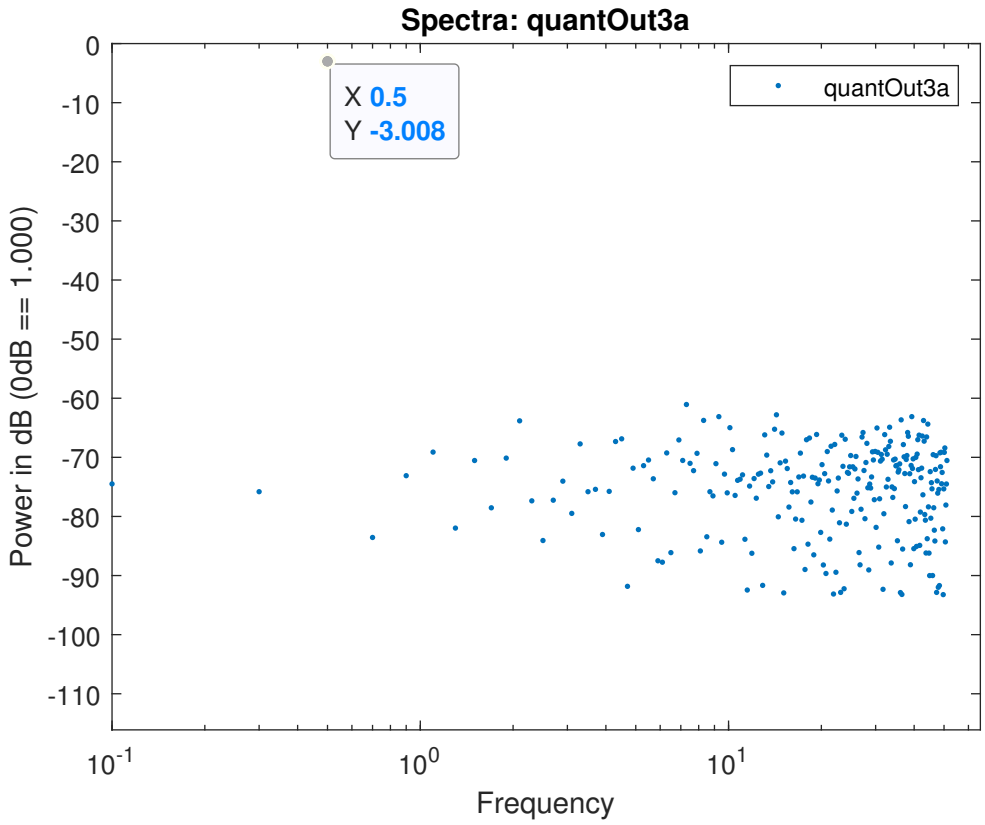


Figure 6.1: Spectral plot of 0.5 Hz sine wave after 7-bit quantization, sampled at 102.4 S/s. Rounding noise omitted.

An SNR of 43.97 dB is obtained as a result of this conversion, which precisely corresponds to the

theoretical expected value using [1, eq. (1.17)].

Next, the signal was *oversampled* by a factor of 2^5, that is, the sample rate f_s is increased by a factor of 32. The spectrum of the oversampled signal is compared to the regularly sampled signal in fig. 6.2. It is clear from the graph that the quantization noise of the oversampled signal is spread throughout the band (ignoring the step-like artifact), with a lower magnitude at singular points than the regularly sampled signal. The SNR of the oversampled signal 43.98 dB, the same as the regularly sampled signal. This is to be expected as f_s is not a dependent variable in SNR calculations.

If we assume the quantization noise to be 'white noise', spread evenly throughout the nyquist interval, then the upper part of the spectrum, beyond the signal of interest, contains only noise. Extracting only the information for frequencies contained in the non-oversampled case (the blue spectrum in fig. 6.2) and ignoring the rest yields a 'perfect' filter. Since the oversampling ratio (OSR) is 32, we therefore expect that the noise is now reduced by a factor of 32. The expected value is thus $43.98\text{dB} + 10\log_{10}(32) = 59.00\text{dB}$. The simulation, however, gives an SNR of 54.31 dB, which can be accounted for by the fact that the quantization noise is not uniform in this case and is in fact higher in the signal band.

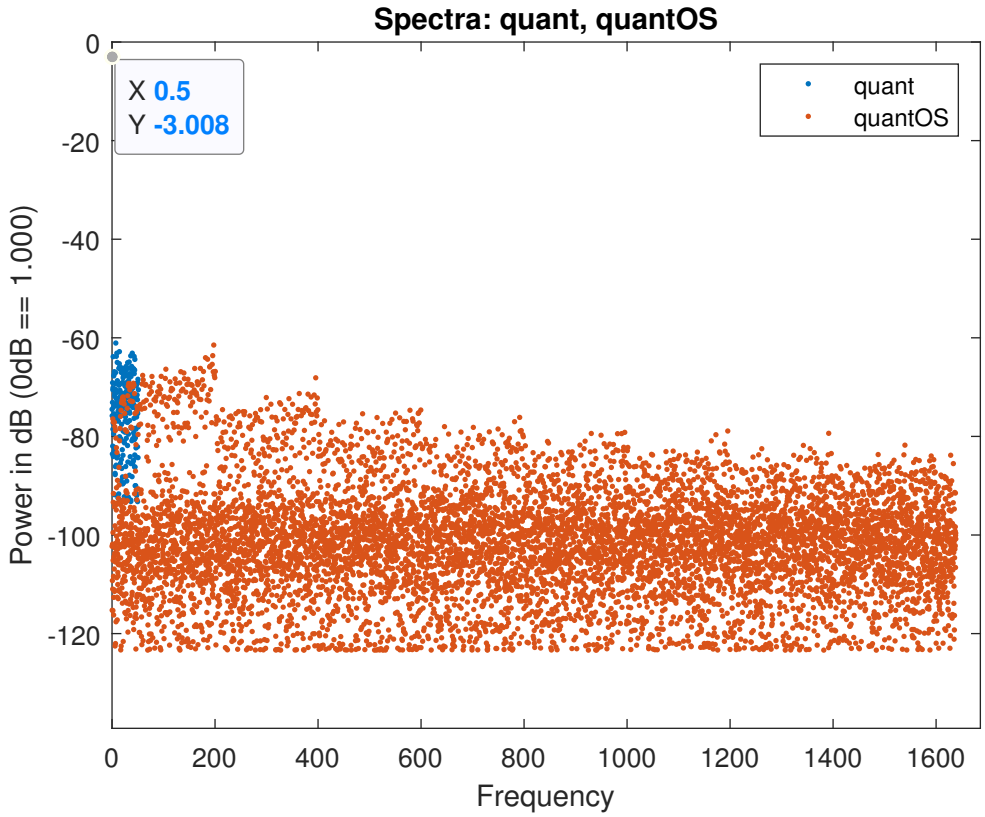


Figure 6.2: Quantized sine wave after regular sampling and oversampling, shown on a linear frequency scale. Rounding noise omitted.

6.4 Single-pole noise shaping

The simulink model was extended with a negative feedback loop and a discrete-time integrator between the feedback and the A/D converter (1D-LUT). An OSR of 32 was used, just as in the previous experiment. The input signal power was reduced by 1 dB from the full-scale sine power.

The output was plotted and the positive extreme of the signal is shown in fig. 6.3. The oscillations occur such that the average value tracks the input. This high frequency oscillation causes the quantization noise, that previously was assumed to be "white" to dominate more at higher frequencies, as seen in fig. 6.4.

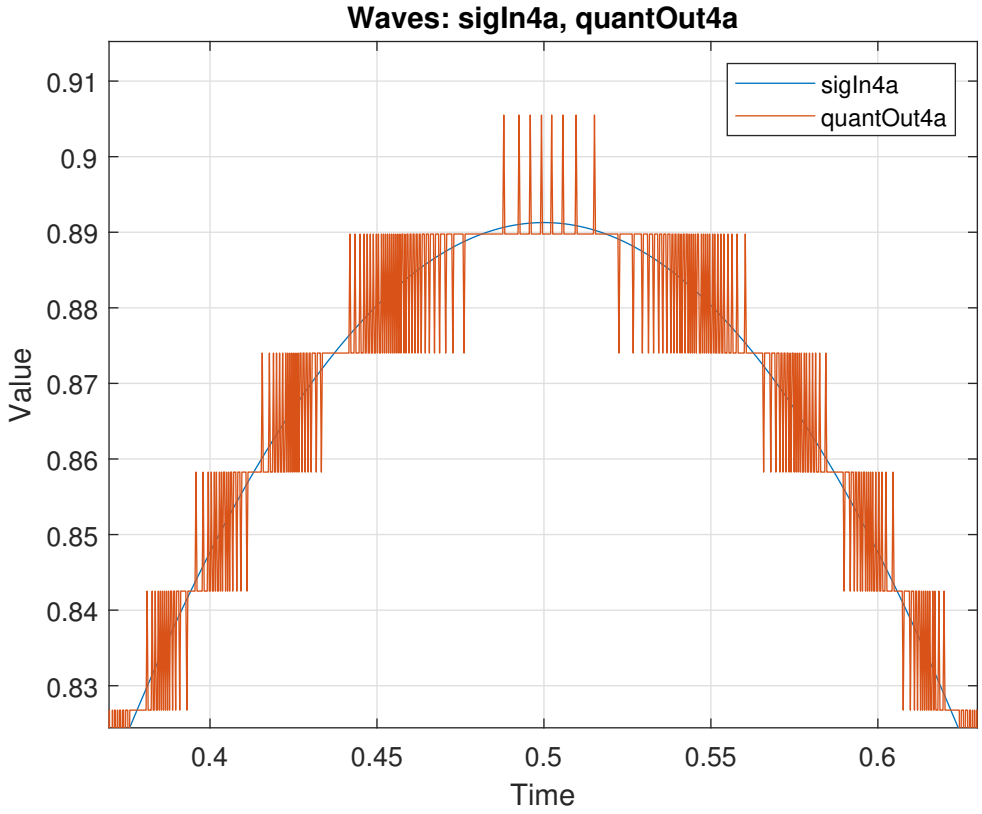


Figure 6.3: The quantized output (red) approximates the input signal by oscillating between the two nearest values defined in the LUT.

As in the previous section, a 'perfect filter' was applied to the noise-shaped output to remove any noise power outside the signal band, yielding an SNR of 82.35 dB, which corresponds within a small margin of error to the expected ideal value of 83.90 given by the expression in [1, eq. (6.16)]. By shaping the noise the SNR has increased by nearly 40 dB!

The feedback loop that we have constructed incorporates a 1^{st}-order integrator, a component that it has in common with many filters of high-pass character. We thus expect to observe a 20 dB/dec rising slope in the spectral plot, which is indeed confirmed by the plot in fig. 6.5, a linear representation of fig. 6.4.

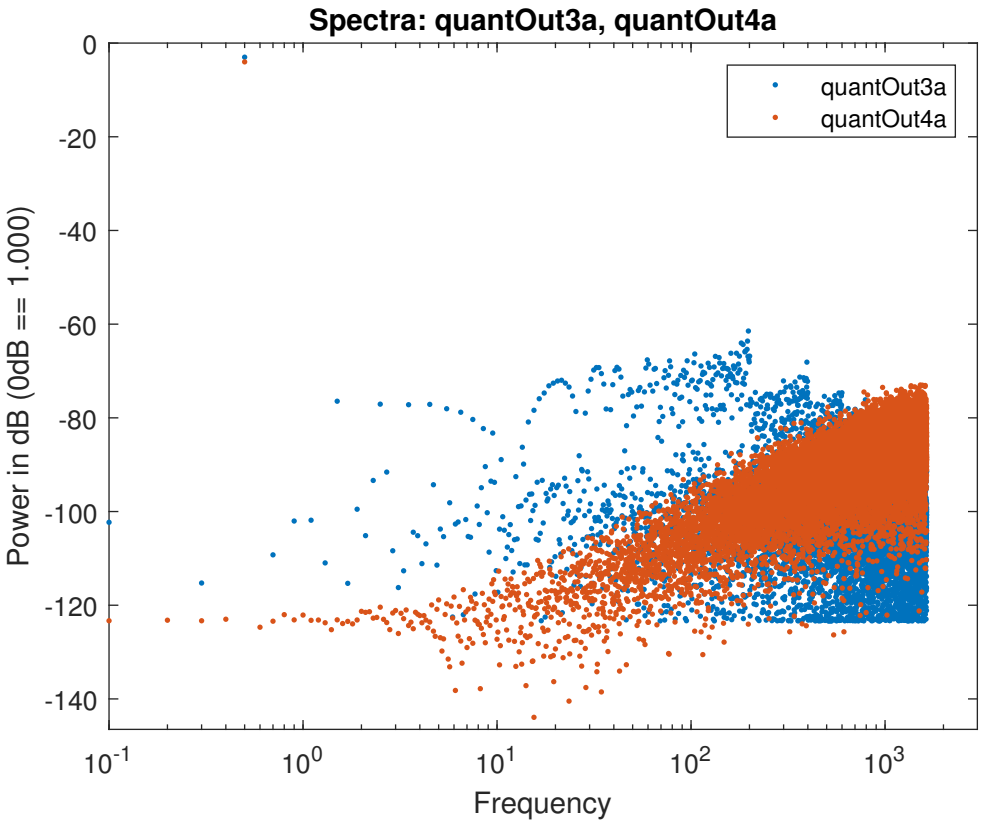


Figure 6.4: Output after negative feedback (red) clearly shows that a large amount of the quantization noise power has been shifted to higher frequencies compared to the oversampled signal without feedback (blue).

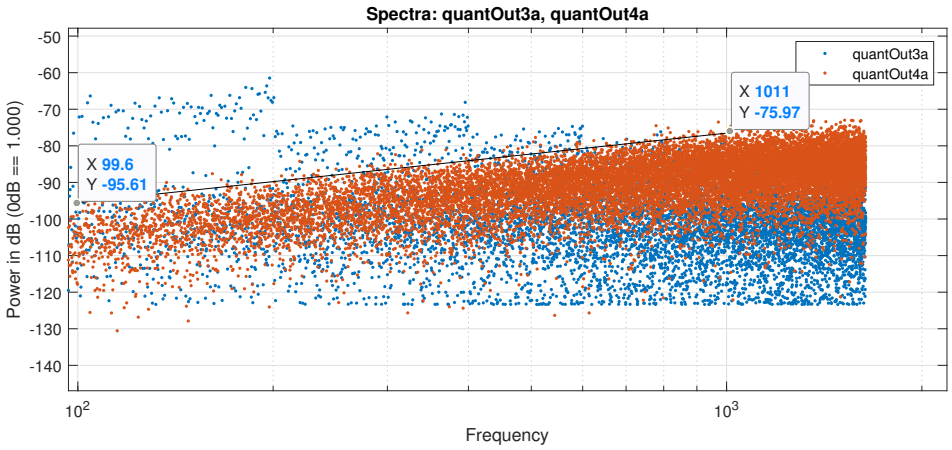


Figure 6.5: The shaped noise (red) has the same slope as a 1^{st}-order integrator (20 dB/dec).

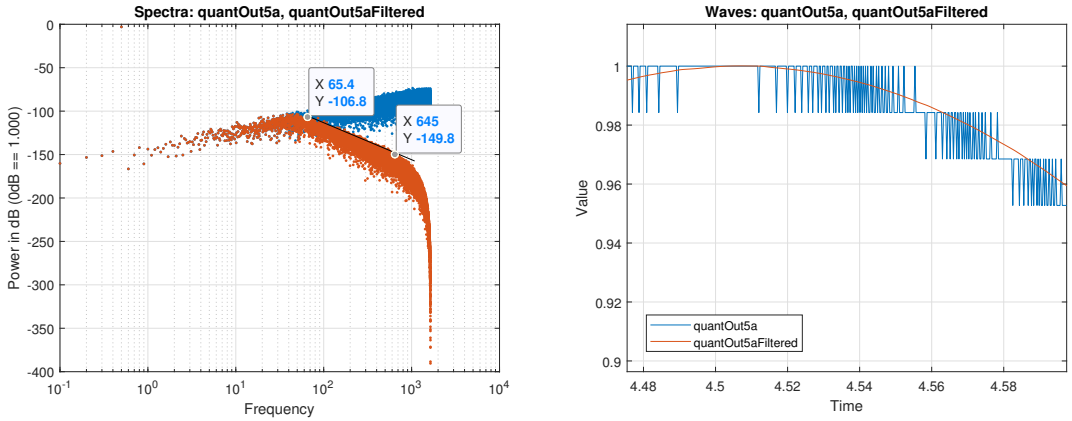
6.5 Filters

The ideal filters used thus far are, of course, not acheivable in reality. In order to design a realizable filter a `Digital Filter Block` was added to the output of the `simulink` model. The filter specified in the design block was a 3 $^{\text{rd}}$-order IIR Butterworth low-pass filter. The cutoff frequency f_c was set to the upper bound of the signal band. This was expedited by setting f_c as a fraction of the

normalized frequency defined by the reciprocal of the OSR, ie. $f_c = 1/OSR$.

The resulting filtered signal spectrum is shown in fig. 6.6a). The filtered signal begins slope downwards at approx. 512 Hz, the edge of the signal band, at a rate of approx. 40 dB/dec. This is to be expected as the 1st order integrator has a positive slope of 20 dB/dec and the 3rd order LP-filter has a negative slope of 60 dB/dec, giving a theoretical net slope of -40 dB/dec.

The transient plot in fig. 6.6b) likewise conforms reasonably to expectation, with most of its observable high frequency content absent.



(a) Filtered and unfiltered signal spectra superimposed. Initial filter transient removed.

(b) Filtered and unfiltered signal transients. Zoomed in.

Figure 6.6: Filtered and unfiltered outputs in red and blue, respectively.

Despite not being perfect, the filter delivers an SNR of 81.18, a reduction of just 1.2 dB compared to the 'perfect' filter!

The filter order was swept (rebuilding the filter each time) and plotted against the recorded SNR in fig. 6.7. The main insight one can gain from the graph is that the filter performance suffers from a case of diminishing returns, with the largest gains being made within the first 3 or 4 orders; beyond that and the complexity of designing a high order filter arguably outweighs the performance benefit. One may also note that the curve seems to converge towards the SNR produced by the 'perfect' filter, meaning that almost ideal performance is achievable with a reasonably low-order filter.

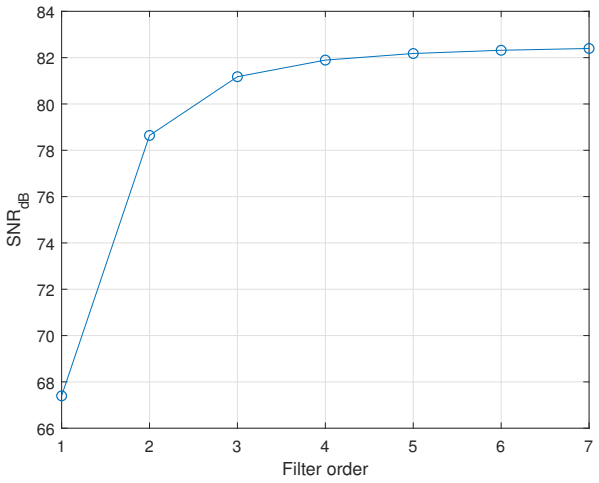


Figure 6.7: SNR as a function of digital filtering of various orders.

6.6 One-bit ΣΔ-converter

The LUT used throughout this lab to model the ADC was modified to map the input to only two levels, essentially creating a comparator. Re-running the simulation as in previous tasks (maintaining an amplitude of 1 dB below full-swing, so as to avoid clipping artifacts) with the 1-bit ADC yielded an SNR of 44.11 dB. This corresponds with the approximate expected value of 47.78 dB (the simulated value will be lower because the source is not quite a full-swing sine wave).

To illustrate the issue of single bit, first order converter susceptibility to limit cycles and spurious tones at low frequencies, the logical extreme of removing the input source entirely and its resulting output before and after filtering is shown in fig. 6.8. The unfiltered output oscillates almost uniformly between the two quantizer levels which, as seen in the filtered output, results in a minor initial transient before averaging out to zero. This is to be expected as

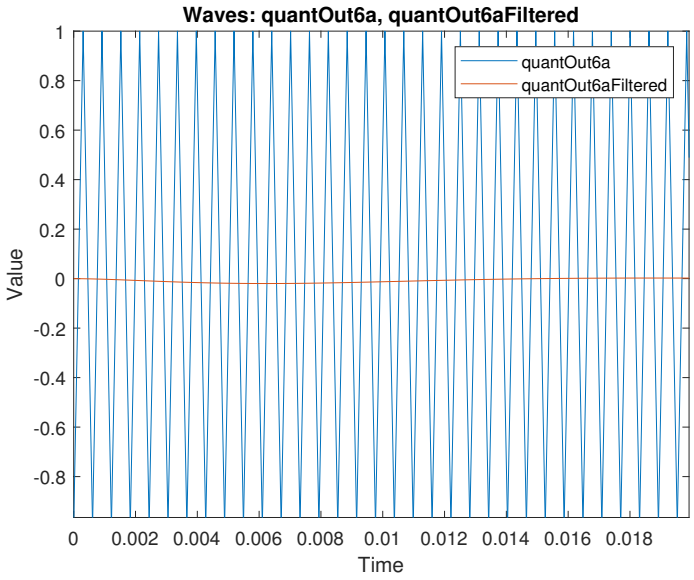
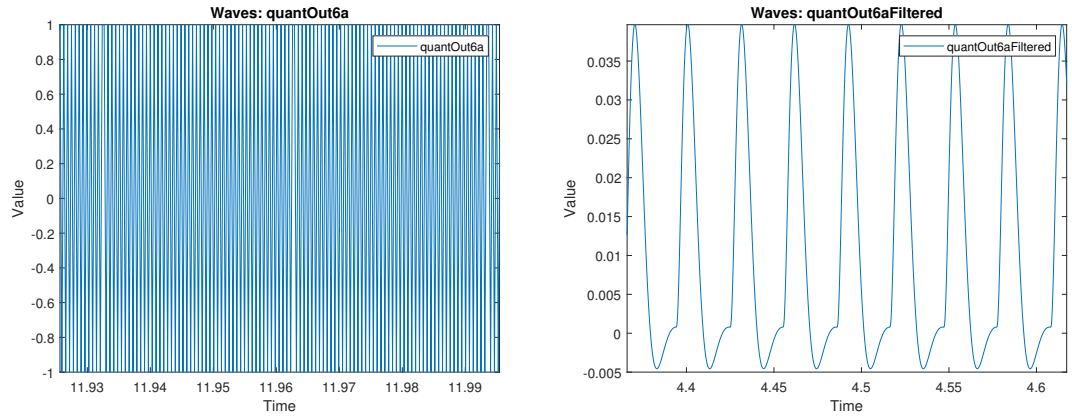


Figure 6.8: Filtered and unfiltered output of 1-bit, 1^{st}-order converter with input source removed (or set to zero).

Applying a DC bias of 0.01 to the input results in the outputs seen in fig. 6.9. The unfiltered output now is not uniform but seems to be "missing" oscillations at regular intervals. It is with this regularity that the "new" component seen in the filtered output oscillates; a frequency of approx. 32 Hz, well within the signal band.



(a) Unfiltered output.

(b) Filtered output.

Figure 6.9: Output from 1-bit, 1st-order converter with 0.01 DC signal on input.

The spectrum of the filtered output confirms the frequency of the limit-cycle and reveals tones at multiples of this frequency, seen in the spectral plot in fig. 6.10.

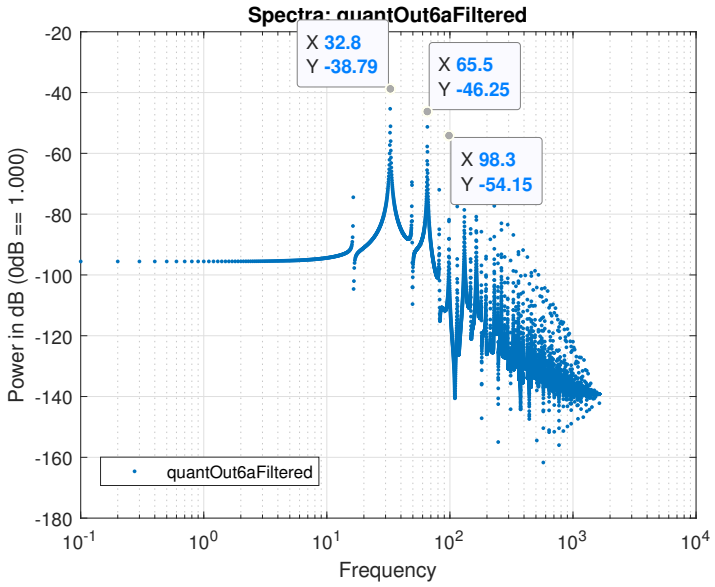
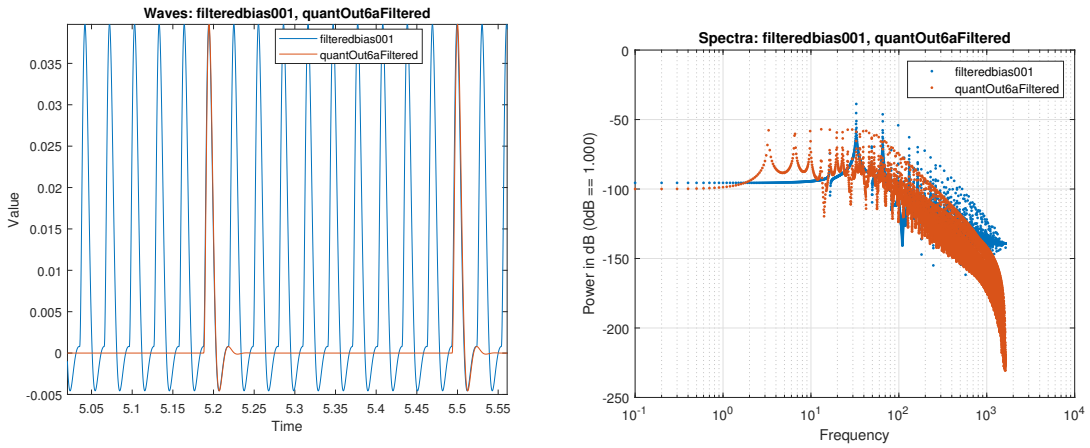


Figure 6.10: Spectral plot of filtered output from 1-bit, 1st-order converter with 0.01 DC signal on input.

Changing the bias level to 0.001, one tenth of the previous level, reveals another fact about the converter in that the limit cycle frequency changes in direct proportion. This can be observed in fig. 6.11a). Unsurprisingly, the lower frequency of the limit cycle for the 0.001 bias level input causes more unwanted tones to enter the signal band, further degrading performance, as seen in fig. 6.11b).

To mitigate the problem of low-frequency limit cycles caused by small DC biases, one may, counter-intuitively, deliberately apply a DC bias level that is certain to cause limit cycles that fall outside the signal band where they can be filtered along with other noise.



(a) Transient plot showing limit cycles.

(b) Spectral plot

Figure 6.11: Filtered outputs of a 1-bit converter with a 0.01 and 0.001 DC bias input, respectively.

6.7 Reflection Questions

For a 1st order converter it is generally more useful to double the OSR than the number of qunatizer levels, if possible, as SNR can be shown to have a stronger positive relationship with the OSR than the ENOB. Maloberti provides a rule-of-thumb in the following equation[1, eq. (6.16)]

$$\mathrm{SNR}_{\Sigma\Delta,1|\mathrm{dB}} = n' \cdot 6.02 + 1.78 - 5.17 + 9.03 \cdot \log_2(\mathrm{OSR})$$

where n' is the number of bits. Doubling the quantizer levels increases n' by 1.

For a 1st order converter with a given quantizer resolution and signal bandwidth, the achievable resolution is limited by the OSR. A high OSR necessitates that all the components in the converter: integrator, DAC, filter etc. are able to handle high frequency operation, thus limiting OSR to the slowest component.

Bibliography

[1] F. Maloberti, *Data Converters*. Springer, 2007, ISBN: 9780387324852.