

Appendix

Introduction: why simulations?

Computer simulations offer a sensible way to understand the operation of D to A and A to D converters more deeply. A clear distinction must be made however between transistor- and system-level simulations. Transistor-level simulations analyze the detailed behavior of circuits. Running them is very costly in terms of computer time because the calculations rely upon very large numbers of time samples, each resolving a large number of transistors. System-level simulations, on the contrary, globalize the behavior and therefore do not suffer from the same drawback.

The MATLAB program available on the website is intended for system-level simulation. The principles underlying the associated converter simulation toolbox are explained below. The text concludes with a few examples to illustrate how to carry out converter ‘software experiments’.

www.oup.co.uk/best.textbooks/engineering/jespers

1. ‘Analog’ and ‘digital’ representations

The interpretation of simulation results rests mostly on graphs, like linear and logscaled plots. Since converters combine analog and digital data, the representation of the latter is entrusted to their analog counterparts. Thus, binary coded words are represented by 2^N discrete DC levels as illustrated by the quantization function `qtz(in,N)`, whose input ‘in’ is ‘analog’ (continuous) and output ‘digital’ (discrete). Consider the MATLAB program below:

```
clear
N = 3;                               % number of bits
in = -1.5:.01 : 1.5;                 % input vector
out = qtz(in,N);                     % output vector
stairs(in,out); grid                 % plot
```

The plot of Fig. A.1 represents the transfer characteristic of a 3-bit quantizer versus the continuous analog input with eight discrete plateaus illustrating the set of all the output words of the quantizer (unless specified, dynamic ranges are normalized between minus and plus one).

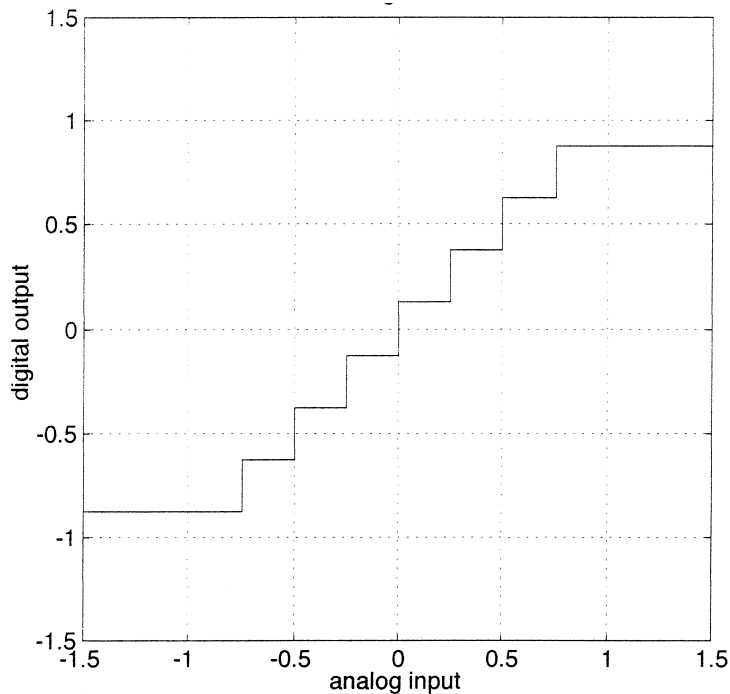


Fig. A.1.

2. The input data

Testing converters requires appropriate input data (generally a linear ramp or a pure sine wave). The linear ramp is used in order to assess the DC characteristics, like the transfer function and INL–DNL plots whereas a pure sine wave is used to evaluate the AC performances, like in the code density test or spectral signature.

In the next paragraphs we consider how to input ‘analog’ and ‘digital’ data to A to D and D to A converters.

2.1 Input data for A to D converters

The input data required to test A to D converters are generally linear input ramps and sinusoidal signals. A linear ramp can be obtained by means of the MATLAB command

```
in = linspace(lower bound, upper bound, number of samples);
```

‘Analog’ sine waves can be invoked similarly by means of MATLAB instructions. We recommend however the function `sinput(St,Np,P,D)`, which is intended especially for spectral signature evaluation. `Sinput` generates a matrix consisting of `St` columns, each containing a sine wave whose magnitude is normalized to one. Each column contains the same number ‘`Np`’ of entire sine wave periods but with

230 | Appendix

random phases. The number of rows represents the number of input samples N_s accordingly to the equation below:

$$N_s = 2^{(P)} + D$$

'D' adds D more samples to the set already defined by 'P' to allow eventual transients to die out before spectral analysis. It is strongly recommended to make use of this variable when systems with state variables are being considered, like Delta-Sigma converters. Making D equal to 50 or 100 is generally enough to avoid the deterioration of the **ffts**. This delay is transparent for it is erased automatically once spectra are being evaluated. The number of samples is then a power of two. When not specified, D is automatically equal to zero.

The example below shows a set of instructions that defines a matrix Y consisting of 5 columns, each 1024 samples long representing 2 periods of the same sine wave with random phases (Fig. A.2):

```
clear
St = 5; Np = 2; P = 10;
Y = sinput(St, Np, P);
Ns = 2^(P); time = 1: Ns;
plot(time, Y)
```

2.2 Input data for D to A converters

The simulation of D to A converters requires a little more caution since D to A converters are controlled by coded input words. A look-up table is used for this purpose.

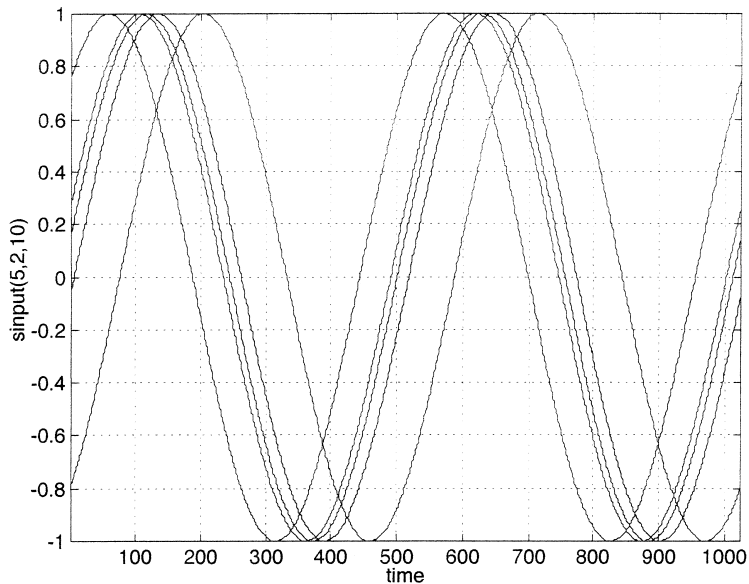


Fig. A.2.

Look-up tables can be implemented by combining the instructions **binm**(N) or **trm**(N) with **binw**(Typ,N,sigm). The instructions **binm** and **trm** generate matrices or tables representing the ordered sets of all possible binary codes that can be represented by *N*-bit words. The **binm** instruction applies to single quadrant *binary* coded words whereas **trm** lists *redundant signed digit* (RSD) words (the latter is used widely in this toolbox for it removes automatically DC pedestals, easing **fft** calculations). The instruction **binw**(Typ,N,sigm) which is described further, generates a column vector *N* rows long that represents the weights in descending order of any binary coded D to A converter described by the variables *Typ*, *N* and *sigm*. When **binm** or **trm** is multiplied by the column vector **binw**, the results is the look-up vector of the D to A converter.

In the example below the binary and RSD look-up tables w1 and w2 of a perfect 3-bit D to A scaled converter are generated.

```
clear
N = 3;
w1 = binm(N)*binw(2,N,0);
w2 = trm(N)*binw(2,N,0);
w = [ w1 w2]
```

The result is:

```
w =
    0          - 0.8750
   0.1250     - 0.6250
   0.2500     - 0.3750
   0.3750     - 0.1250
   0.5000      0.1250
   0.6250      0.3750
   0.7500      0.6250
   0.8750      0.8750
```

In the example below, we apply a 'digital' two periods sine wave consisting of 256 samples to the converter above using the look-up vector **w2**. The 'digital' sine wave is generated first starting from the analog **sinput** instruction, which is followed by a **qtz** instruction that generates the 'analog' counterpart of the digital input sine wave. The next instruction is a linear transform changing the discrete outputs of the quantizer into addresses *z*, which access the look-up table 'w2'. The 'analog' output of the converter is the result of instruction **w2(z)**:

```
clear
N = 3;
% generation of the look-up table
w2 = trm(N)*binw(2,N,0);
% 'analog' input sine wave
St = 1; Np = 2; P = 8; in = sinput(St,Np,P);
% quantized sine wave
q = qtz(in,N);
% address conversion
z = 2^(N-1)*(q+1) + .5;
```

232 | Appendix

```

% output
out = w2(z, :);
% graph of the 'analog' input before the quantizer and converter
output
X = 1:2^(P); plot(X,[ in out] )

```

The outcome is illustrated in Fig. A.3.

The same procedure may be used to produce a 'digital' ramp to evaluate the converter static transfer characteristic but this is not very useful since look-up tables are transfers functions by themselves.

3. Available conversion functions

A survey of the available conversion functions is presented below.

3.1 D to A converters

Three types of scalars are considered: *R2R*, *binary unit-elements* and *thermometer* scalars. The unit-elements may either represent unit-capacitors or unit-current source.

The instruction that generates the weight scale of any binary weighted D to A converter is: **binw** (Typ,N,sigm). Typ defines the kind of scalar, 1 for an R2R and 2 for a unit-element binary scalar. N represents the number of bits. Errors affecting the resistors or the unit-elements are controlled by the variable *sigm* that determines

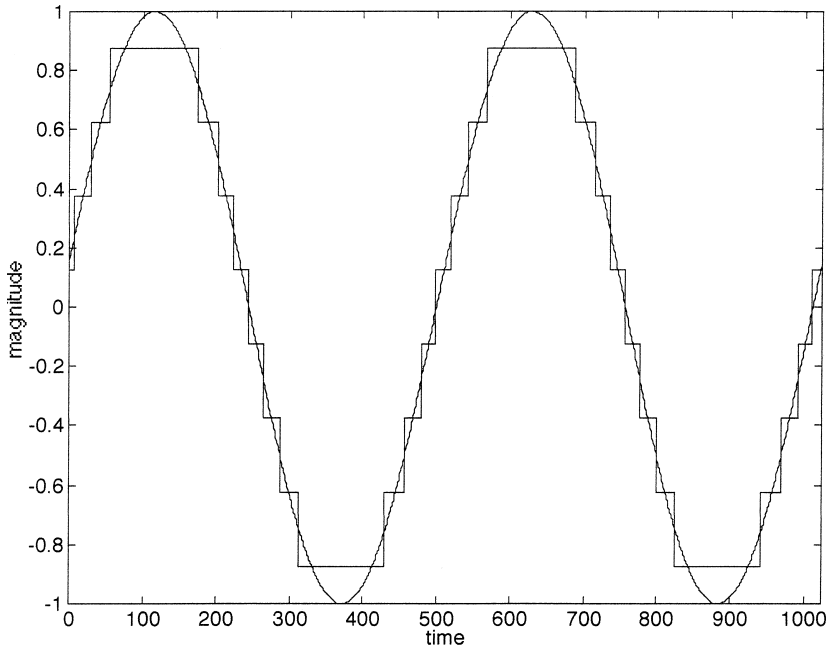


Fig. A.3.

the standard deviation of the Gaussian distribution of the errors. For instance, the instruction:

```
z = binw(2,8,0.01)
```

generates a column vector **z** which represents the binary weights in descending order of an 8-bit unit-elements binary scalar whose unit-elements exhibit errors of 1% standard deviation with respect to their nominal values.

```
z = 0.5001
    0.2497
    0.1249
    0.0627
    0.0312
    0.0157
    0.0080
    0.0039
```

It is possible to compare the impact of mismatches affecting R2R and unit-element converters. In the first case, the precision of the resistors must be almost the same as that of the converter to comply with the alleged resolution, whereas in the second, large numbers of unit-elements to implement the MSBs improve the overall accuracy.

Multiplying **binm** or **trm** by **binw** generates the look-up table of any D to A converter described by **binw**. An example is shown below:

```
clear
N = 6; d = 2^(-N);
Typ = 2; % unit-elements binary scaled conv
sigm = 0.2; % stand dev of every unit-element
Y = trm(N)*binw(Typ,N,sigm); % two-quadrant look-up table
X = -1 + d: 2*d: 1-d; % input codes
plot(X,Y,'+'); grid; axis('square')
```

The actual transfer characteristic is shown in Fig. A.4.

Notice that in the above example, the input X is equivalent to the cascading of a **linspace** and a **qtz** instruction.

The instruction **da1**(in,N,Typ,sigm,S,sigmS) has been introduced in order to bypass most of the above instructions. It computes the analog output of any N-bit scaled D to A converter or segment converter whose digital input is given by the vector or matrix 'in'. 'Typ' defines the type of scaler like above and 'sigm' the standard deviation of every scaler element. The two other variables are optional. They are specified when segment converters are considered. The number of segments is equal to 2^S and the standard deviation of the references defined by 'sigmS'. When omitted, S is equal to zero and the number of segments equal to one so that the converter resumes to a simple binary scaled converter.

For instance, the program at the end of Section 2.2, when it is rewritten as shown below, produces the same plot as the one depicted in Fig. A.3. The instruction **da1** takes care automatically of the generation of the look-up table, quantization of the sine wave and address conversion.

```
clear
N = 3;
% 'analog' input sine wave
St = 1; Np = 2; P = 8; in = sinpwt(St,Np,P);
```

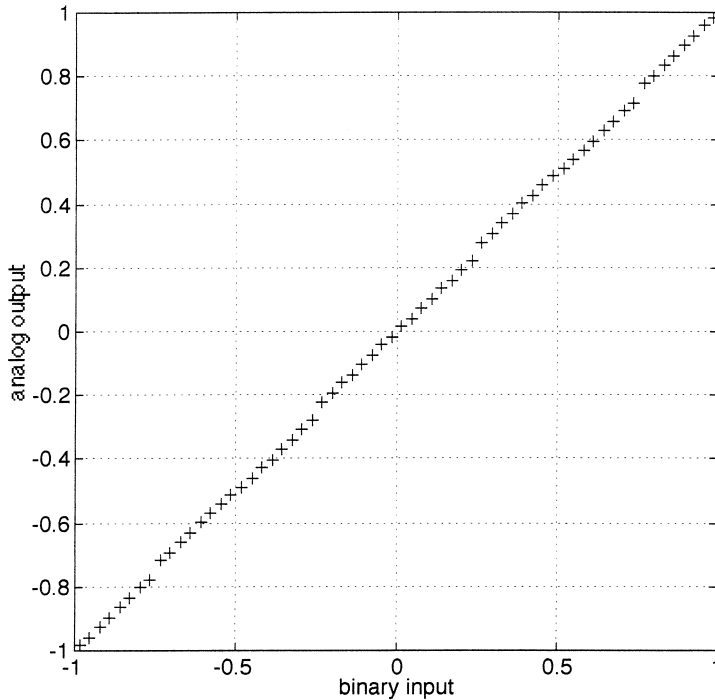


Fig. A.4.

```
% 'digital' output
out = da1 (in,N,2,0);
% graph of the 'analog' input before the quantizer and converter output
X = 1:2^(P); plot (X,[ in out] )
```

The instruction **da2**(in,N,sigm) is intended for N-bit scaled D to A converters whose unit-elements are selected according to a thermometer code instead of binary law. This type of converter exhibits a transfer characteristic without DNL errors in practice. The input is represented by the variable 'in' like above, 'N' being the number of bits and 'sigm' the standard deviation of the unit-elements.

3.2 A to D converters

Various A to D conversion functions are available. They may be divided into three categories, *serial converters* like algorithmic converters, *Delta-Sigma converters* and *multi-step converters*.

Algorithmic converters

The instruction **algor1**(in,N,A,offOpA,offComp) computes the digital output of any N-bit cyclic or algorithmic converter. Impairments can be introduced by means of appropriate choices of the gain A (nominally equal to 2), as well as the Op. Amp. and comparator offsets.

The instruction **algor2**(in,N,A,Vmin,Vmax,offOpA) simulates the behavior of RSD cyclic converters. Besides the variables 'A' and 'offOpA', which have the same meanings as above, the comparison levels 'Vmin' and 'Vmax' can be adjusted separately (the nominal values are -0.25 and $+0.25$ respectively, V_{ref} being always equal to 1). The RSD converter ignores the impact of transition position errors upon the INL and DNL. Reversing the sign of the input after the first cycle compensates moreover the global offset of the transfer characteristic.

Delta-Sigma converters

The three basic noise shapers shown in Fig. 7.13 can be implemented by means of the instruction **dtsgX**(in,N,G,B,sigmAD,sigmDA) where X is equal to 1, 2 or 3, accordingly to the order of the noise shaper. The quantizer consists of an A to D converter followed by a D to A converter whose number of bits is fixed by the variable N. The data outputted by the A to D converter are the actual output data.

Various parameters can be modified in order to evaluate the impact of errors on the overall performances of noise shapers. 'G' defines the feedback loop gain of the accumulators. These are the discrete counterparts of continuous time integrators. When G is smaller than 1, the accumulators are similar to integrators having gains equal to $1/(1 - G)$. The resulting 'leakage' from the finite gain of the Op Amps increases the low frequency noise. The variable 'B' controls the symmetrical saturation limits of the integrator located before the quantizer. B offers a means to evaluate the consequences of saturation of the loop filter. The variables 'sigmAD' and 'sigmDA' model the quantizer imperfections. As suggested by the names, the first affects the A to D conversion, the second the D to A. The A to D converter is a flash converter, which consists of a resistive divider ladder followed by a bank of ideal comparators. Impairments are introduced in the string of resistors that form the reference divider by means of the 'sigmAD' variable. The D to A converter is a scaled converter whose unit-elements are controlled by 'sigmDA' in the same way as the **binw** instruction. It is possible to verify that the A to D converter impairments produce unnoticeable effects upon the spectral content of the noise shaper whereas impairments affecting the D to A converter introduce harmonics and increase the noise floor in the baseband. Those effects disappear of course when the quantizer counts only 1 bit.

The duration of the input signals should always be long enough to overcome the artefacts that affect the fft's. A good figure of the variable 'P' defining the sample length 2^P is 12 to 14 (remember however that the duration of the computations increases exponentially with 'P'). The variable D should be not less than 50 to avoid transients.

Multi-step converters

Multi-step converters are built by means of the instructions **recycl**(in,M,cycl,imp,ext) or **recyclRSD**(in,M,cycl,imp,ext). Each instruction defines a bloc consisting of an M-bit flash converter and an M-bit multiplying D to A converter (MDAC). The input signal 'in' is applied to the flash converter whose output drives the MDAC. The latter is a unit-elements capacitive D to A converter, which outputs the difference between the input and the 'analog' counterpart of coded words delivered by the flash converter. This difference is multiplied by 2^M before being recycled as many times as fixed by the variable 'cycl'.

Impairments may be introduced by means of the vector **imp**, which consists of [sigmR sigmComp sigmC offOpA]. The flash converter discrimination levels are

236 | Appendix

controlled by means of a resistive divider and a bank of comparators. 'sigmR' defines the standard deviation of the divider resistances. The standard deviation of the comparator offsets are set by 'sigmComp'. The MDAC operates like a scaled D to A capacitive converter. Therefore its errors are determined by means of the variable 'sigmC' (C stands for unit-capacitors). Since the Op Amp gain 2^M is controlled by the ratio representing the sum of all the MDAC capacitances over a single unit-capacitor, the interstage gain error is an implicit result of sigmC. This is why no interstage gain control is provided. The Op Amp offset is controlled by 'offOpA'.

The instructions **reflash** (N,sigmR,sigmComp,ext) and **remdac**(N,sigmC,ext) define respectively the comparison levels of the flash converter and the transfer function of the MDAC.

Since new sets of impairments are calculated every time these instructions are invoked, one should take care to call them before the cycling algorithm is initiated to keep the converter unchanged during the computations.

Transition position errors are corrected if the flash converter dynamic range is extended beyond Vref. This can be achieved by means of the variable 'ext', equal to 1 when the dynamic range of the flash is bound to ± 1 , and 2 when its limits are extended to ± 2 .

Two types of recycling converters are available: the non-restoring one and the RSD. The instructions **recycl**, **reflash** and **remdac** are duplicated for the RSD mode simply by adding the characters **RSD** to the above instructions.

4) Testing tools

A number of tools are available for evaluation of the impact of defects on the performances of D to A and A to D converters. These are controlled by instructions enabling them to perform display functions, spectral analysis and code density tests.

Visualization

The usual MATLAB instructions suffice to display data. The instruction **visut**(q) shortens writing programs for it infers the horizontal scale automatically from the length of the data 'q' and plots as many curves as there are columns in the matrix 'q'.

Static characteristics

The instructions below are used to display static characteristics.

The instruction **datrsf**(q,N,S) displays the transfer characteristic(s) and INL–DNL curve(s) of N-bit D to A converters whose output data are described by the column vector or matrix 'q'. The INL and DNL characteristics are plotted after the pause. The vertical scales are set automatically (green lines $\pm 1/2$ LSB, red lines ± 1 LSB). 'S' defines the log of segments number when considered. When not specified, S is equal to zero.

The instruction **adtrsf**(in,q) displays the static transfer characteristic of A to D converters (except Delta–Sigma) and INL–DNL plots after the pause. The column vector 'in' is the analog ramp applied to the converter and 'q' the corresponding output. Midpoints are shown as long as the resolution is low to avoid the risk of cluttering, the linear regression derived from these points is always displayed however. The most extreme points of the transfer characteristic are never taken into account for linear regression computations. Neither saturation nor the overall offset of the transfer characteristic influence the 'gain' of the converter in this manner. The magnitude of the

gain derived from the linear regression is printed in the command window in the same time as the overall offset of the transfer characteristic. The same kind of flexibility is not offered for D to A converters because their transfer characteristics are always normalized before characteristics are plotted.

Dynamic characteristics

Two kinds of dynamic tests are currently available: spectral analysis and code density test.

The instruction **avrsptr**(q) computes the spectrum averaged over the 'St' columns of the data contained in the matrix 'q'. Making 'St' equal to 5 or 10 reduces the amount of random noise superimposed on the individual spectra. It is still possible to look at the individual spectra nevertheless by making use of the instruction **sptr**(q).

Spectral tests are relevant for both D to A and A to D converters, whereas code density tests are restricted to A to D converters exclusively. The instruction **coden**(Ampl,q) displays the code density test and compares the result to the ideal probability density of the input sine wave.

5) Examples

The list below illustrates a number of examples that can be performed with the accompanying toolbox. All the examples are invoked via instructions **explxx.m** where xx stands for one of the file numbers below.

expl01.m Static characteristics of scaled and segment D to A converters

This file displays the static transfer characteristic as well as INL and DNL curves of scaled and segment converters. The INL–DNL characteristics are needed when the number of bits N is larger than 8, because the fine granularity of the static transfer characteristic does not allow for the resolution of individual steps anymore. It is recommended that you start with a scaled converter making 'S' and 'sigmS' equal to zero before considering segment converters. Making use of 'St' offers the possibility to compare the characteristics of several converters having distinct impairments but the same sigm's.

The performances of R2R and unit-elements D to A converters can easily be compared. Those of R2R converters are worse than those of unit-element converters with similar *sigm*'s. The R2R sigma must match the class of the converter whereas unit-elements tolerate sigmas one order of magnitude larger. Hence, the choice is either small numbers of precise resistors or large numbers of less precise unit-elements.

For segment converters, mismatches affecting the references that control the segments are easily spotted. The INL tends to look like a broken line, which reflects the impact of 'sigmS'. The DNL is always a lot better for it reflects the DNL of the internal lower-resolution converter only. To improve the INL, the tolerances of the current sources must be drastically narrowed. Mismatches should be at least one order of magnitude smaller than those of the internal converter. Notice that when a larger number of segments with a lower resolution are considered, but the resolution is kept the same, the INL improves slightly. The chances that segment references balance each other to a larger extent are enhanced indeed (assuming of course that errors obey a zero median Gaussian distribution).

Unit-elements thermometer coded D to A converters may be tested also. As expected, their DNL is always excellent whichever INL.

exmpl02.m Histograms and cumulative curves of INL and DNL

This file is intended to give a more precise picture of the performances of scaled and segment converters. A substantially larger number of converters is considered (St may be equal to 100 or even larger). The max, INL and DNL errors are recorded for every converter. Their histograms are displayed versus the tolerances expressed in LSBs plotted horizontally and, after the pause, the cumulative curves.

The interaction between the INL and the DNL depends upon the type of converter considered. In segment converters the DNL is decoupled from the INL. Thermometer coded converters always exhibit excellent DNL performances.

exmpl03.m Spectral analysis of scaled and segment D to A converters

This file is about the dynamic performances of scaled and segment D to A converters. Spectra offer a different but very efficient way to assess the impact of imperfections on the performances of converters. An increase of the background noise with respect to quantization noise means that the DNL may be the limiting factor as far as accuracy, whereas harmonics point towards the INL. It is possible to evaluate intermodulation products by inputting two sine waves with distinct but close frequencies instead of one.

exmpl04.m Static characteristics, INL and DNL of A to D converters

The transfer characteristic and INL–DNL curves (obtained after the pause) of A to D converters are displayed in this file (except for Sigma–Delta converters). Only one characteristic is plotted at a time because the computation times that are needed are longer than those of D to A converters.

Algorithmic converters

The variable ‘cycl’ that defines the number of times the algorithm is supposed to run represents also the number of bits when the converter is ideal. The impact of gain mismatch and offsets can be analyzed. In RSD converters, when Vmin and Vmax are changed from zero to non-zero values, the single comparison level step splitting of converters running the same number of cycles is clearly visible.

Flash converters

Defects result from mismatches of the resistors defining the reference divider and offsets of the comparators. Although fast by essence, flash converters are slow once they are simulated by means of computers for their behavior is analyzed step by step. The computation time is unnoticeable below 10 bits however.

Recycling converters

Transition position errors and transition magnitude errors may be introduced by means of the variables sigmR, sigmComp and sigmC. The transition errors are compensated when the flash converter dynamic range is extended. When ‘ext’ is equal to 2 the dynamic range widens till ± 2 whereas when ‘ext’ is equal to 1, the dynamic range resumes to ± 1 .

The display shown in exmpl05.m illustrates the way errors affect the behavior of the flash converter and the MDAC. It is advised to run this file before the others to understand the mechanisms controlling accuracy, especially the auto-correction algorithm.

The impact of errors affecting the MDAC depends greatly on the number of bits M resolved per cycle. Errors are divided indeed by the interstage gain 2^M . One can check this feature by comparing the performances of two converters with different numbers of cycles and bits but with the same resolution.

exmpl05.m Two-step A to D cyclic converter waveforms

This file displays the output of the flash converter after each cycle as well as the reconstructed output code words. Errors affecting the reference divider introduce transition position errors that are easily spotted since the transitions do not coincide with the equally spaced vertical grid. The amplified differences between the input data and output of the flash converter encompass a scale wider than resolvable by the flash. Once the scale is expanded ($\text{ext} = 2$), saturation is avoided however.

exmpl06.m Spectral signature of A to D converters

This file, which is the A to D counterpart of *exmpl03.m*, considers algorithmic, flash and recycling converters (for Delta-Sigma noise shapers, the reader should refer to *exmpl09.m*).

To perform an intermodulation test, use two sine waves instead of one and remember that the converter input dynamic range should not exceed the range between -1 and $+1$ to avoid the saturation of the quantizer.

The instruction **harm** lists items such as the magnitude of the fundamental, the quantization noise floor density, the SFDR and the magnitudes of the main harmonics.

exmpl07.m Code density test of A to D converters

The code density test can be performed considering the same converters as those listed in *exmpl04.m*. The input is a pure analog sine wave whose probability density curve is plotted in the same graph as the code density test. The ratio of the two curves is displayed below.

Artefacts affect the code density test when the number of samples and bins fixed by the resolution are commensurable. The variable P defining the number of samples should always exceed the number of bits by at least 4.

exmpl08.m SNR versus magnitude of A to D converters

This file applies to the same converters as those listed in example 04 (Sigma-Delta converters are treated separately). The plot, which represents the Signal-to-Noise ratio of the output data versus the magnitude of the input signal, is currently used to evaluate the Effective Number of Bits (ENOB) as well as the dynamic range. The input signal is the usual **sinput** instruction, every column being pre-multiplied by a distinct coefficient accordingly to a log-scaled law.

It is interesting to compare the SNR plots of single- and two-level algorithmic converters. Owing to their additional bit, the SNR curve of the ideal RSD converter lies always 6 dB above that of the ideal single-level converter running the same number of cycles.

In the single-level converter, gain mismatch shifts the SNR curve downwards, parallel to itself, whereas in the RSD converter the SNR is not affected at low level but bends gradually while the input increases. The RSD converter is therefore more or less similar to a compander.

exmpl09.m Spectral content of noise shapers

This test brings to the fore some of the main features of noise shapers, namely the shape of noise density spectra versus the order of the noise shaper and the number of quantization bits. The three first noise shapers are those depicted in Fig. 7.13. The last is the fourth order noise shaper described in the paper of Chao, K. C.H., Nadeem, S., Lee, W.L., Sodini, Ch., in the *IEEE Trans CAS* **37**, (3), March 1990, 309–18.

It is possible to introduce impairments like Op Amp leakage and saturation as well as errors affecting the A to D and D to A internal converters in the three first noise shapers. When the gain G of the accumulators is low (e.g. 0.95, which is equivalent to an Op Amp gain of 20) the low frequency noise floor increases. Mismatches in the D to A converter ($\text{sigmDA} = 0.001$) introduce low frequency noise and harmonic distortion that is clearly visible when the order of the noise shaper is large (typ. 3) and the number of samples exceeds 2^{12} . Very larger sigmAD s are needed to produce similar performance degradation for the A to D converter does not belong to the feedback branch. When single bit quantizers are considered, sigmAD and sigmDA do not affect the performances at all.

exmpl10.m Magnitude of signals after integrators and quantizer

The way impairments affect the three dtsgX converters of *exmpl09.m* is easily understood if time domain plots representing the signals delivered by the integrators and quantizer are contemplated. It is recommended to start this example with a multi-bit noise shaper (N being equal to 3 or 4) for easing the interpretations.

Noise shapers tend to increase the high frequency quantization noise and lower the amount of low frequency noise. For instance in the 3d order noise shaper, more noise is being superimposed on the sine waves as we move from the first to the third integrator.

With small numbers of quantization bits, e.g. one, the amount of quantization noise is large. Consequently the integrators tend to output large signals that may lead to saturation. A single-bit noise shaper whose input sine wave magnitude is close to 0.5 saturates almost immediately. Long sequences of data stuck at +1 or -1 are experienced as a result. Owing to the large signals before the quantizer, a lot of time can be needed before re-entering the dynamic range of the latter. This introduces low frequency components that produce a large quantity of harmonics. It is possible to shorten somehow the duration of the saturated periods by keeping the dynamic range of the signal applied to the quantizer within more narrow limits. Another way to enlarge the dynamic range is to extend the quantizer dynamic range slightly beyond plus and minus V_{ref} .

exmpl11.m SNR versus magnitude of noise shapers

Same test as in *exmpl08.m* but the noise power density is integrated over the baseband N_c instead of the half-sampling frequency. This is a simple way to simulate the operation of an ideal boxcar decimator. The optional variable N_c fixes the so-called cut-off frequency. Beware of the fact that if N_c is too small, few spectral contributions enter into the spectral power count and the erratic character of the SNR plot increases. A large N_c improves the SNR plot but requires more computation time to keep the OSR unchanged.

When N_c is not specified the baseband is automatically equal to the sampling frequency divided by 2.

exmpl012.m Second order accumulate-and-dump decimator

The spectral signature after a sinc^2 filter is shown in this example. The file makes use of the data stored in the QUANT file after running the noise shaper considered in `exmpl09.m`. There is no need thus to repeat `exmpl09.m` when re-running this example to consider the impact of the downsampling rate of the decimator.

The decimated output signal is shown after the pause.

exmpl013.m Robertson plot

This file shows the Robertson plot of single- or multi-level cyclic (algorithmic) converters. The input is a scalar. The test is intended to illustrate how errors affect the performances.