Adv. Radio Sci., 15, 163–168, 2017
https://doi.org/10.5194/ars-15-163-2017
© Author(s) 2017. This work is distributed under
the Creative Commons Attribution 3.0 License.

Advances in Radio Science
Open Access Proceedings

Interpolation algorithm for asynchronous ADC-data

Stefan Bramburger[1], **Benny Zinke**[2], **and Dirk Killat**[1]

[1]Microelectronics Department, Brandenburg University of Technology, 03046 Cottbus, Germany
[2]IHP, Im Technologiepark 25, 15236 Frankfurt (Oder), Germany

Correspondence to: Dirk Killat (killat@tu-cottbus.de) and Stefan Bramburger (stefan.bramburger@b-tu.de)

Received: 9 January 2017 – Revised: 23 May 2017 – Accepted: 9 July 2017 – Published: 21 September 2017

Abstract. This paper presents a modified interpolation algorithm for signals with variable data rate from asynchronous ADCs. The Adaptive weights Conjugate gradient Toeplitz matrix (ACT) algorithm is extended to operate with a continuous data stream. An additional preprocessing of data with constant and linear sections and a weighted overlap of step-by-step into spectral domain transformed signals improve the reconstruction of the asycnhronous ADC signal. The interpolation method can be used if asynchronous ADC data is fed into synchronous digital signal processing.

1 Introduction

An approach to improve the energy efficiency of ADCs is their asynchronous operation: The ADC is operated with a variable clock depending on the characteristics of the input signal. The average power consumption of the ADC and the power required for RF data transmission is reduced with a reduction of the average sample rate. Asynchronous ADCs are potentially applicable in stand-alone sensor nodes, biomedicine (Yuan and Lam, 2013) or energy management.

The drawback of an asynchronous operation is the variable data rate which needs to be synchronized to the clock of the subsequent signal processing system by interpolation. Several methods exist for interpolation; some operate in time domain (Neubauer, 2003), others in frequency domain (Feichtinger et al., 1995). The interpolation algorithms differ considerably in complexity and the quality of the reconstructed signal.

The base for the presented interpolation algorithm is the Adaptive weights Conjugate gradient Toeplitz matrix (ACT) algorithm. It operates in the frequency domain and has the drawback of much higher computational effort than interpolation in the time domain, but it converges much faster than other frequency domain based interpolation algorithms (University of California, 2016).

The simulation data is based on a tracking ADC (Fig. 1). The output data rate is variable because it depends on the input signal characteristics. The difference between the current input signal and the output of the I-DAC controls the ADC clock rate. If the difference gets larger the clock rate is increased. The resulting datastream output is used for the interpolation.

This work is organized as follows: Sect. 2 briefly explains the basic principle of the ACT algorithm, Sect. 3 describes the extensions to the ACT algorithm, Sect. 4 presents simulation results and Sect. 5 concludes the paper.

2 ACT-algorithm

The ACT algorithm requires high computational effort because it uses several mathematical methods (Feichtinger et al., 1995). The first is the adaptive weights method (Neubauer, 2003) in which the weights of the points are calculated according to Eq. (1). This input weighting vector $\boldsymbol{w}_j$ is used to calculate the spectral transformation of the time of the samples (Eq. 2). The parameter M denotes the number of spectral lines of the input signal and results from the ratio of the largest time distance between two sample points and the shortest time distance between two sample points. The amplitude values $\boldsymbol{p}(t_j)$ are transformed into spectral domain using the weighting vector $\boldsymbol{w}_j$ (Eq. 3).

$$\boldsymbol{w}_j = \frac{(t_{j+1} - t_{j-1})}{2}\ ;\ \mathbb{J} \in \{1, 2, \ldots, J\} \tag{1}$$

$$\boldsymbol{y}_k = \sum_{j=0}^{J} \boldsymbol{w}_j \cdot e^{-i2\pi k t_j};\ k = 0, 1, \ldots, 2M \tag{2}$$

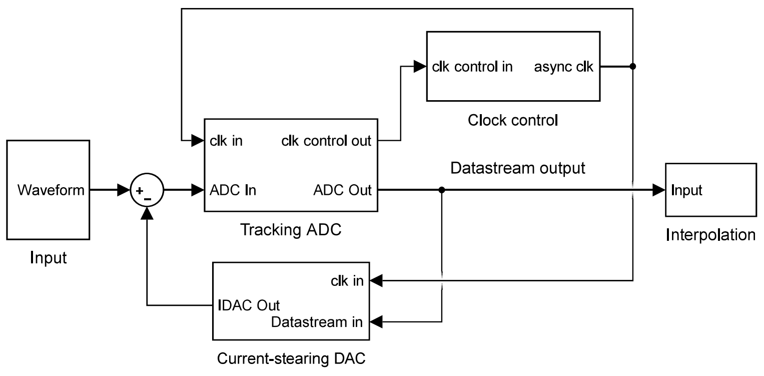

Figure 1. Asynchronous tracking ADC as signal source for interpolation algorithm.

$$\boldsymbol{b}_k = \sum_{j=0}^{J} \boldsymbol{p}\left(t_j\right) \cdot \boldsymbol{w}_j \cdot e^{-i2\pi k t_j};\ |k| \leq M \tag{3}$$

The next step in the ACT algorithm is the construction of the Toeplitz matrix $\mathbf{T}_w$ (Eq. 4). The property of the Toepliz matrix are constant values on each descending diagonal. Additionally the values above the descending main diagonal are complex conjugate of the values below the diagonal. These properties make the calculations in the subsequent iterative process very efficient (University of California, 2016).

$$\mathbf{T}_w = \begin{bmatrix} y_0 & \overline{y_1} & \cdots & \overline{y_{2M-1}} & \overline{y_{2M}} \\ y_1 & y_0 & \cdots & \overline{y_{2M-2}} & \overline{y_{2M-1}} \\ \ldots & \ldots & \ddots & \ldots & \ldots \\ y_{2M-1} & y_{2M-2} & \cdots & y_0 & \overline{y_1} \\ y_{2M} & y_{2M-1} & \cdots & y_1 & y_0 \end{bmatrix} \tag{4}$$

The spectrum of weighted time values $\boldsymbol{y}_k$, the weighted spectrum of amplitude values $\boldsymbol{b}_k$ and the toeplitz matrix are used to solve Eq. (5) for the interpolated spectrum $\boldsymbol{a}$. $\boldsymbol{b}$ is set to $\boldsymbol{b}_k$. This equation will be solved iteratively with the help of the conjugate gradient method (Eq. 6). In combination with the Toeplitz matrix the system converges quickly and after a maximum of $2M+1$ iteration loops the result is available. The initial parameters of the system are these: $\boldsymbol{a}_{n-1}=0$, $\boldsymbol{r}_{n-1}=\boldsymbol{b}_k$ and $\boldsymbol{q}_{n-1}=\boldsymbol{b}_k$.

$$\mathbf{T}_w \cdot \boldsymbol{a} = \boldsymbol{b} \tag{5}$$

$$\begin{aligned} \boldsymbol{a}_n &= \boldsymbol{a}_{n-1} + \frac{\left\langle \boldsymbol{r}_{n-1}, \boldsymbol{q}_{n-1} \right\rangle}{\left\langle \mathbf{T}_w \boldsymbol{q}_{n-1}, \boldsymbol{q}_{n-1} \right\rangle} \cdot \boldsymbol{q}_{n-1} \\ \boldsymbol{r}_n &= \boldsymbol{r}_{n-1} - \frac{\left\langle \boldsymbol{r}_{n-1}, \boldsymbol{q}_{n-1} \right\rangle}{\left\langle \mathbf{T}_w \boldsymbol{q}_{n-1}, \boldsymbol{q}_{n-1} \right\rangle} \cdot \mathbf{T}_w \boldsymbol{q}_{n-1} \\ \boldsymbol{q}_n &= \boldsymbol{r}_n - \frac{\left\langle \boldsymbol{r}_n, \mathbf{T}_w \boldsymbol{q}_{n-1} \right\rangle}{\left\langle \mathbf{T}_w \boldsymbol{q}_{n-1}, \boldsymbol{q}_{n-1} \right\rangle} \cdot \boldsymbol{q}_{n-1} \end{aligned} \tag{6}$$

Finally the interpolated spectrum $\boldsymbol{a}_k$ is transformed back to the time domain with Eq. (7) using an array $\boldsymbol{t}$ with equidistant time steps of the interpolated signal.

$$\boldsymbol{p}(t) = \sum_{k=-M}^{M} \boldsymbol{a}_k \cdot e^{i2\pi k t} \tag{7}$$

3 Extension to ACT algorithm

Frequency based interpolation can be used on signals of a finite period because the number of sample points for the DFT is limited. To deal with a continuous data stream the data has to be split into equally spaced sections. Every section can now be interpolated for itself and the interpolated data stream is constructed by connecting the interpolated sections.

3.1 Smooth transition between sections

By connecting the interpolated sections undesired steps in amplitude can occur at the transitions of the sections. This problem can be solved by overlapping the boundaries of the sections by a certain amount of samples. Further improvement is achieved by using a smoothing function on the overlapping areas. Here the hyperbolic tangent function as proposed in Rädler (2016) is used. Advantages of this transition function is scalability of slope and good convergence to final values. The multiple derivative hyperbolic tangent is continuous, which is advantageous in subsequent signal processing.

The smoothness of the merging of two sections is determined by σ. The higher the value of σ the smoother the transition is. But with a higher σ the function needs more samples for overlapping. Figure 2 shows the used smoothing functions with a total overlap of 100 samples and different grades of smoothness.

The merger of two sections using the smoothing function is exemplified by Eq. (8). The parameter μ determines the transition point between both sections.

$$\begin{aligned} \boldsymbol{p}_{\text{merged}}(t) &= \boldsymbol{p}_2(t) \cdot \left(\frac{1}{2} + \frac{1}{2}\tanh\left(\frac{t-\mu}{\sigma}\right)\right) \\ &+ \boldsymbol{p}_1(t) \cdot \left(1 - \left(\frac{1}{2} + \frac{1}{2}\tanh\left(\frac{t-\mu}{\sigma}\right)\right)\right) \end{aligned} \tag{8}$$

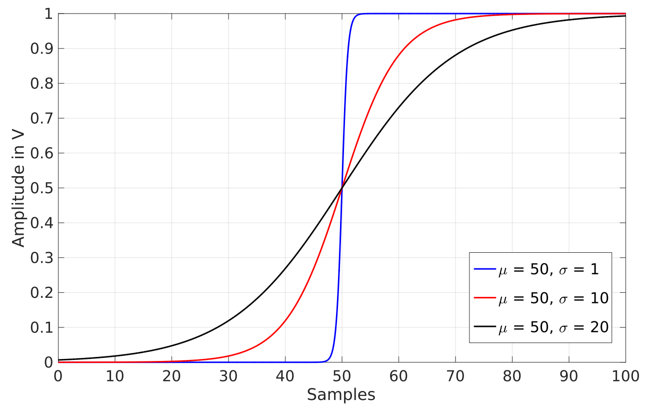

Figure 2. Hyperbolic tangent function for smoothing transitions between sections of samples of data stream.

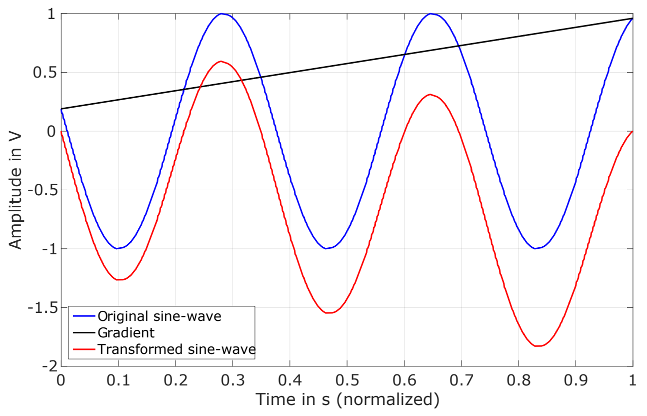

Figure 3. Transformation of input signal of one section to set first and last data point to zero.

3.2 Amplitude shifting function

Another issue of interpolation is the boundary-value problem: The first value and the last value of a section should have the same level. But this occurs either very rarely or not at all for arbitrary input signals. This results in significant interpolation errors at the section boundaries. The proposed solution is a transformation of the input signal by a linear function as described by Eq. (9) before the application of the interpolation algorithm. The whole section of the signal is shifted by the amplitude value of the first point of the signal to zero. In addition a linear function is added so that the last sample point is zero as well.

$$\boldsymbol{p}_{\text{transform}}\left(t_j\right)=\boldsymbol{p}\left(t_j\right)-\vartheta\cdot\left(t_j-t_1\right)-\boldsymbol{p}\left(t_1\right)$$
$$\text{with } \vartheta=\frac{\boldsymbol{p}\left(t_J\right)-\boldsymbol{p}\left(t_1\right)}{t_J-t_1} \tag{9}$$

In Fig. 3 the linear function (gradient) is drawn in black. The linear function is alined to the first and the last value of the input signal (blue). The transformed signal (red) has its first and last point on zero level. After interpolation in the frequency domain the inverse transformation has to be applied to retrieve the original signal.

3.3 Signals with sporadic constant sections

The interpolation algorithm should be suitable for all kind of signal shapes. Critical points are signals with sporadic constant sections, e.g. a square wave signal as shown in Fig. 4. At the left side the interpolated signal (blue) does not represent the original signal (red). On the right side the edges are interpolated well but large overshoots exist. The reason for this difference between both interpolations is the cutoff frequency Ω_M of the input signal. The higher Ω_M the higher the frequencies; respectively more spectral lines M (Eq. 11) are used for the interpolation. To change the number of spectral lines M that are used for calculation the factor κ has to be varied. Factor κ is used to calculate Ω_M in Eq. (10).

Ω_M must be set in relation to the minimum bandwidth of the original signal. Therefore Ω_M is calculated from the maximum sample step size $T_{\max}$ by introducing the factor κ (Eq. 10). This factor has a range between 0 and 1. The higher κ is, the better the interpolation of the slopes of a square wave signal. The number of spectral lines for interpolation depends on Ω_M (Eq. 11).

$$\Omega_M=\kappa\cdot\frac{\pi}{T_{\max}};\ 0<\kappa<1 \tag{10}$$

$$M=\left\lfloor\frac{\Omega_M\cdot N\cdot T_{\min}}{\pi}\right\rfloor+1 \tag{11}$$

The reason for overshoots in the right part of Fig. 4 is the small number of samples from the asynchronous ADC that is used as input for interpolation. An ideal asynchronous ADC produces samples only at transitions of rectangular waveform. The solution is to synthetically fill the constant sections with additional samples. The sections that have to be filled with additional samples before interpolation are detected by the variation of the distance of two successive sample points (Eq. 12). A change of the distance between two successive time steps by a factor greater than τ (Eq. 12), resulting in additional samples being introduced.

$$\left(t_{j+2}-t_{j+1}\right)>\tau\cdot\left(t_{j+1}-t_j\right) \tag{12}$$

The factor τ has to be chosen carefully so that the fill algorithm will be activated during constant sections of a square wave signal or arbitrary signal but not at low frequent sine waves input. The activation depends on the sample rate algorithm of the asynchronous ADC.

4 Results

A comparison of frequency based ACT interpolation and time domain interpolation with linear and cubic splines is demonstrated in Fig. 5. At the peak of the sinus signal the

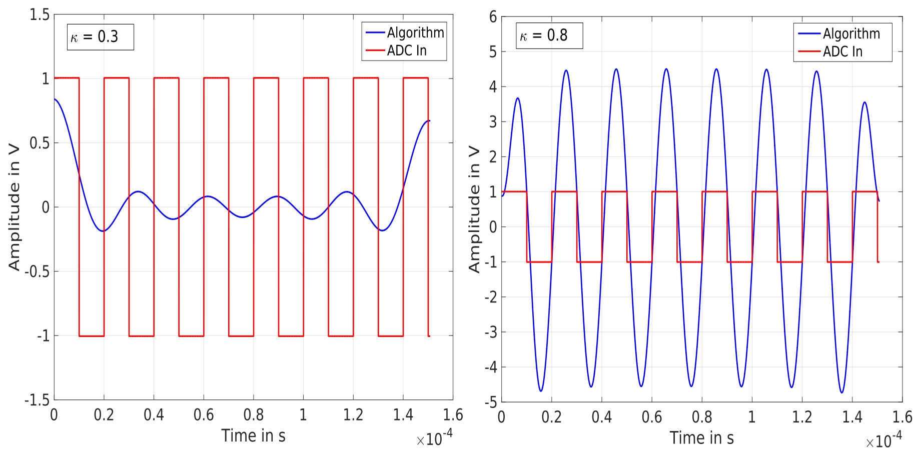

Figure 4. Result of interpolation of rectangular waveform with asynch. ADC samples only at transitions; low cutoff ($\kappa = 0.3$) and high cutoff ($\kappa = 0.8$) frequency for interpolation.

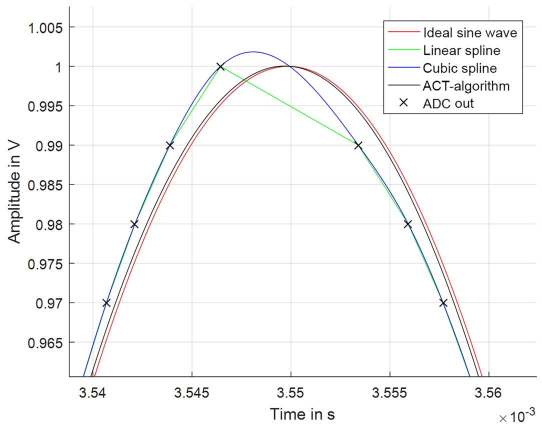

Figure 5. Comparision of time domain and frequency domain based interpolation (ACT) with original signal (red) and asynchronous ADC output (x).

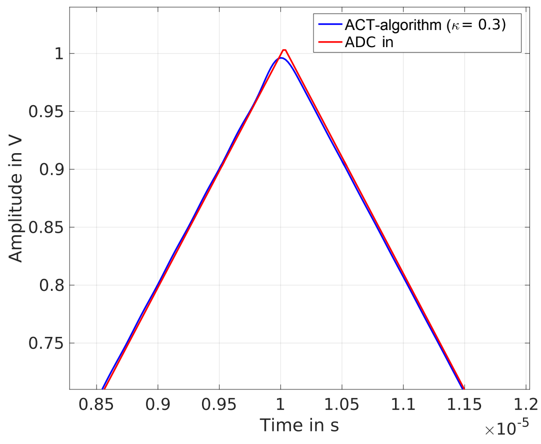

Figure 6. Interpolated triangle waveform from asynch. ADC.

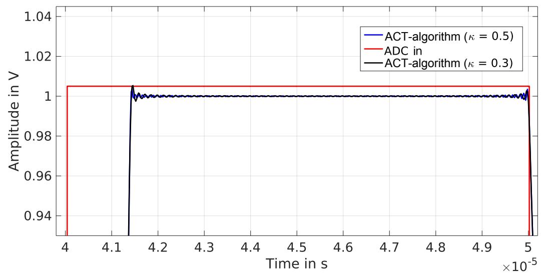

Figure 7. Interpolation of asynch. ADC samples with rect. waveform.

second derivative is at its maximum and the asynchronous ADC produces the largest errors compared to original signal. Therefore the peak of the sinus is well suited to demonstrate the effectiveness of the interpolation algorithm. The original sine wave (red) and the output of the asynchronous ADC ($\Delta q = 0.01$) are given as well. In fact the spline interpolated function is exactly in line with sampled ADC data, but the ACT interpolation result (black) fits best with the original signal.

Figure 6 shows the interpolation of a triangle signal. The falling and rising slopes are interpolated well, but the vertex is flattened because of the reduced sample rate of the asynchronous ADC before the vertex.

The effectiveness of synthetic insertion of additional sample points in a square wave signal is depicted in Fig. 7. The reduced sample rate of the asynch. ADC at the rising slope of the rectangular signal and the quantization levels produces an error that is larger than the interpolation error. The overshoots due to interpolation are reduced to a minimum through the filling of the constant sections. Ringing only exists at the beginning and at the end of the section depending on the bandwidth of the interpolation algorithm.

For a sinusoidal asynchronous ADC input with a frequency of 91 kHz the corresponding spectrum of the inter-

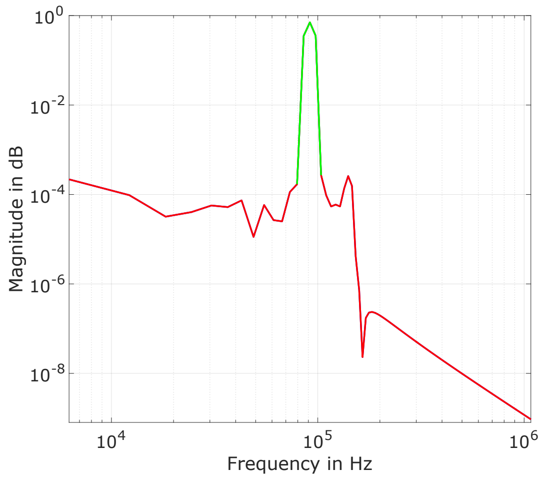

Figure 8. Spectrum of an interpolated sine wave: $f = 91$ kHz, resolution $n = 8$ bit, sample rate of asynch. ADC variable between 2.8 and 25 MS s^{-1}, interpolation bandwidth $\Omega_G = 140$ kHz, frequency resolution $\Delta f = 6.1$ kHz.

polated signal is shown in Fig. 8. By reducing the interpolation cutoff frequency Ω_G to 140 kHz, a full scale resolution of 8 bits and a sample rate varying between a minimum of 2.8 MS s^{-1} and a maximum of 50 MS s^{-1}, an SNDR of 65 dB can be achieved, which is remarkable far above the SNDR of 43 dB that would have been achieved with cubic spline interpolation. The stepsize of the simulated Tracking ADC is 1 or 2 LSB per clock cycle. The results can be interpreted as follows: An interpolation in the frequency domain with interpolation cutoff frequency increases the signal resolution by oversampling. To achieve an optimal SNDR it is important to know the signal waveform that has to be processed. E.g. for a sinus the value of κ could be lower than for a rectangular waveform.

Frequency based interpolation is well suited for filtering of noise (Fig. 9). This can demonstrated by feeding a sinusoidal input with a noise ripple of 10 pW Hz^{-1} into the asynch. ADC input. As the peak value of signal plus noise is limited to 1 V the interpolated sinusoidal signal (blue) has a smaller amplitude than the ideal one (black).

Table 1 gives an overview of the computation time and mean deviations of the ACT algorithm in comparison to standard adaptive weights interpolation and Fourier series expansion using the program *Matlab*. For a sinusoidal input the ACT algorithm interpolates much faster than the other methods. The speed of interpolation depends on the number of points given and the number of spectral lines used for calculation. A square wave signal requires more computation time than a sinusoidal signal. Fourier series expansion does not converge in a reasonable time in the case of a square wave signal. The length of the interpolated signal of the waveforms given in Table 1 is 0.025 ms. This interpolation, performed with *Matlab*, is not suitable for real-time processing at such

Table 1. Computation time and accuracy.

	ACT algorithm	Standard adaptive algorithm	Fourier series expansion
Time sine	0.22 s	31.75 s	1771.13 s
Time rectangular	5.97 s	90.75 s	“∞”
Mean deviation	26 nV	96 nV	96 nV

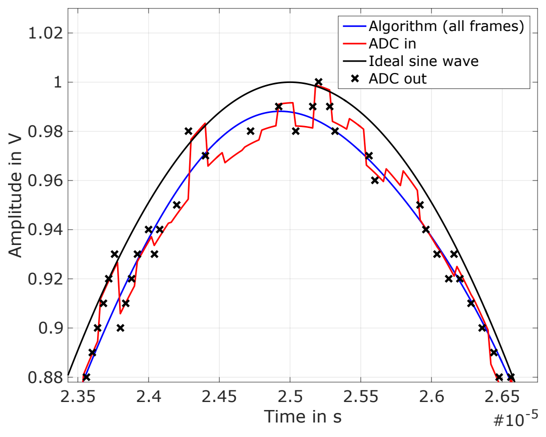

Figure 9. Interpolation result (blue) of sinusoidal ADC input with noise.

a high input sampling rate and signal bandwidth. But for low data rate signals e.g. for biomedical applications or environmental sensors, real-time processing may become feasible.

5 Conclusions

In this work the practical implementation of frequency domain interpolation of asynchronous ADC data using the ACT algorithm (Feichtinger et al., 1995) is demonstrated. The continuous data stream is split into sections for individual interpolation in the frequency domain. The original ACT algorithm is improved in three steps: firstly DC offset and 1st-order linear contribution are removed from the original function. Secondly the individual interpolated sections are reassembled using a smoothing function. Thirdly the problem of periods with reduced sample rate due to asynchronous ADC sampling is eliminated by synthetic insertion of samples before interpolation.

The application of the presented interpolation algorithm requires only a few inputs: The highest sample rate of the ADC and the frequency used for reconstruction. To achieve an optimal interpolation result additional parameters should be known: These are the actual signal frequency and waveform, required for adjustment of the length of the individual sections for interpolation, and the bandwidth of the signal adjusted by factor κ. The ratio κ of the bandwidth of the interpolation algorithm related to the frequency resolution control the amplitude resolution and ratio of noise suppression.

Both the synthetic insertion of samples in the data stream, the optimal segmentation of data stream before spectral interpolation, as well as the optimal adjustment of interpolation bandwidth in relation to the minimum ADC sample rate and signal bandwidth, depend on the algorithm of the asynchronous ADC. A feedback from signal processing analyzing the interpolated signal waveform to the control of interpolation algorithm parameters have the potential for further improvements of interpolation of asynchronous ADC data.

Data availability. Tracking ADC simulation results used to get interpolation results of Figs. 4, 5, 8 and 9 are available in the Supplement.

The Supplement related to this article is available online at https://doi.org/10.5194/ars-15-163-2017-supplement.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. The authors thank the German Research Community (DFG) for funding the project under grant number KI 1585/2-1.

Edited by: Jens Anders
Reviewed by: two anonymous referees

References

Feichtinger, H. G., Gröchenig, K., and Strohmer, T.: Efficient numerical methods in non-uniform sampling theory, Numer. Math., 423–440, 1995.

Neubauer, A.: Irreguläre Abtastung: Signaltheorie und Signalverarbeitung, Springer, Berlin, Heidelberg, 2003.

NI – National Instruments: Schnelle Fourier-Transformation (FFT) und Fensterung, http://www.ni.com/white-paper/4844/de/, last access: 30 December 2016.

Rädler, J.: Smooth transition between functions with tanh(), https://www.j-raedler.de/2010/10/smooth-transition-between-functions-with-tanh/, last access: 30 December 2016.

University of California: A first guided tour on the irregular sampling problem, https://www.math.ucdavis.edu/~strohmer/research/sampling/irsampl.html, last access: 30 December 2016.

Yuan, C. and Lam, Y. Y. H.: A 281-nW 43.3 fJ/conversion-step 8-ENOB 25-kS/s asynchronous SAR ADC in 65 nm CMOS for biomedical applications, IEEE International Symposium on Circuits and Systems (ISCAS2013), 19–23 May 2013, Beijing, China, 622–625, 2013.