Consistent Scaling for the Inverse Chirp-Z Transformation

David Veit[1], Michael Vorderderfler[1], Michael Gadringer[1] and Erich Leitgeb[1]

Abstract – **The Chirp-Z transformation is a versatile tool for frequency-time domain transformation. Though this versatility introduces a non-consistent scaling of the transformation products for which the standard implementation of the transformation does not account for. In this contribution we present a scaling factor that can be applied to the signal to achieve a consistent scaling for different time and frequency vectors. We analyze the behavior of the Chirp-Z transformation in case of an interpolation of the time domain data and compare the results to the Discrete Fourier Transformation. Further, we discuss why this approach is of practical use for calculating time domain signals from data captured with a vector network analyzer.**

Keywords – **frequency-domain analysis, time-domain analysis, time-frequency transformation, chirp-Z transformation, inverse chirp-Z transformation**

I. Introduction

Chirp-Z Transformation (CZT) is a generalized form of the Discrete Fourier Transformation (DFT) and can be used for efficient and versatile frequency-time-domain conversions. In contrast to the DFT, target frequency and time vectors can be supplied for the transformation. This allows efficient conversion of sparse signals and interpolation in a single transformation step [1] [2] [3].

Another big advantage of the Inverse Chirp-Z Transformation (ICZT) is that the input frequency vector does not need to start at DC (0 Hz) to result in a real valued time domain signal, assuming a symmetric spectrum. This is of practical importance as vector network analyzers (VNAs) are usually not able to perform measurements down to DC. If one wants to apply the Inverse Discrete Fourier Transformation (IDFT) to measurement data of a VNA usually the spectrum is extrapolated to DC. For this the frequency vector generated by the VNA needs to be chosen in a way to result in a sample exactly at DC, if it is extended with equally spaced samples. Interpolation in the frequency domain can help with data not meeting this requirement.

Another method often seen is to simply neglect the imaginary part of the time domain signal. In the authors' opinion this is not a proper way to handle this issue, as information of the signal is intentionally thrown away, which makes a conversion back to the original spectrum impossible, although the error produced by this method is small if the lowest frequency sample is close to DC. The ICZT does not suffer from this problem. But with the possibility to supply target frequency and time vectors comes a problem which is often not mentioned in literature. If an ICZT of the same spectrum is performed using different target time vectors the time domain signal will be scaled differently.

To understand why this happens please think about the following example. For this example we use an implementation of the CZT which conserves the energy of the signal on both sides of the transformation. For the transformation we use two time vectors with same start and end time, but different sampling frequencies. In this example the magnitude of the time domain signal with lower sample rate will be higher as the same energy is distributed over less samples. In most cases this is not what we want to achieve. The situation becomes even more complicated if you consider that also the start and end times of the vectors could be different, and the input spectra might also use different frequency vectors. To deal with this problem we present an implementation of the ICZT which provides an additional scaling factor as output of the transformation in Section II. By multiplying the scaling factor with the time domain signal one can correct for the above mentioned effect, while the transformation itself uses the known implementations from [1] [2] [3]. At this point it needs to be mentioned that [4] showed, that the combination of CZT and ICZT produces numerical errors depending on the provided parameters for the transformation. If one is aware of this, the resulting error can be kept low by choosing appropriate parameters for the transformation, allowing practical use with only little impact due to numerical errors of the transformation.

In Section III we compare the behavior of the DFT and CZT with scaling factor using a test signal which is sparse in time. We also show how the ICZT can be employed to perform an interpolation of the time domain signal directly in the transformation. In Section IV we shortly conclude our findings.

II. Implementation of CZT and ICZT

In this section we present the equations used to perform the CZT and ICZT with the mentioned scaling factor. To simplify the equations of the CZT and ICZT we assume that the transformation is performed along the unit circle in the Z-plane, as it is the case for the DFT. For the purpose of transforming VNA measurement data into time domain this is a valid assumption and has the benefit that for correctly chosen time and frequency vectors the CZT and DFT produce the same result. Information on this can be found in [1]. The equations (1) and (2) show that the time and frequency vectors used for the transformation do not need to start at zero, while (3) shows the CZT and (4) the ICZT along the unit circle.

$$\underline{t} = t[n] = nT + T_0 \quad n = 0 \dots N-1 \tag{1}$$

$$\underline{f} = f[k] = k\Delta f + f_0 \quad k = 0 \dots M-1 \tag{2}$$

The authors[1] are with the Institute of Microwave and Photonic Engineering at Graz University of Technology, 8010 Graz, Austria.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\,2\pi\,t[n]\,f[k]} \tag{3}$$

$$x[n] = \frac{1}{\sqrt{NM}} \sum_{k=0}^{M-1} X^*[k] \cdot e^{j\,2\pi\,t[n]\,f[k]} \tag{4}$$

The implementation of the CZT in (5) is done according to (3). In addition to (4) our Matlab® implementation of the ICZT also returns a scaling factor A as shown in (6) and applies a normalization according to the used time window as shown in (7). The normalization factor A_t depends on the duration of the time vector t and the alias free time T_{alias}. Like in the frequency domain a signal will also repeat in the time domain if the time vector exceeds T_{alias}. The magnitude scaling factor A is calculated in (10) using the length of the time vector N, the length of the frequency vector M, the time vector dependent scaling factor A_t and the frequency vector dependent scaling factor A_f.

$$\underline{X} = myCZT\left(\underline{t}, \underline{x}, \underline{f}\right) \tag{5}$$

$$\left[\underline{x'}, A\right] = myICZT\left(\underline{f}, \underline{X}, \underline{t}\right) \tag{6}$$

$$x'[n] = A_t \cdot \frac{1}{\sqrt{NM}} \sum_{k=0}^{M-1} X^*[k] \cdot e^{j\,2\pi\,t[n]\,f[k]} \tag{7}$$

$$A_t = \sqrt{\frac{t[N-1] - t[0]}{T_{alias}}} \tag{8}$$

$$T_{alias} = \frac{1}{\Delta_f} = \frac{1}{f[1] - f[2]} \tag{9}$$

$$A = \sqrt{N/M} \cdot A_t^{-1} \cdot A_f \tag{10}$$

$$A_f = \frac{f[M-1] - f[0]}{f[M-1]} \tag{11}$$

The following code lines show the Matlab® implementation of the function myICZT() and myCZT().

```
function [X] = myCZT(t, x, f)
T = mean(t(2:end) - t(1:end-1));
Fs = 1 / T;
if max(abs(f)) > Fs/2
   warning('Specified frequency vector exceeds alias free range.')
end
f = f(:)';
x = x(:);
t = t(:);
%% Calculate transformation
x_matrix = repmat(x, 1, length(f));
X = sum( x_matrix .* exp(-1i * 2*pi * t * f ) ,1);
X = X(:);
end

function [x, A] = myICZT(f, X, t)
f = f(:);
f_low = min(abs(f));
f_high = max(abs(f));
X = X(:);
t = t(:)';
f = [-f(end:-1:1); f];
X_two_sided = [conj(X(end:-1:1)); X];
% Generate time vector
T_alias = 1 / abs(f(2) - f(1));
if nargin < 3 || length(t) <= 1
   Td = 1/(max(abs(f))*2);
   t = 0:Td:T_alias;
end
% Check time vector
if t(end) > T_alias + eps
   warning('Provided time vector exceeds alias free time.')
end
A_t = sqrt((t(end)-t(1)) / T_alias) ;
% Calculate transformation
X_matrix = repmat(X_two_sided, 1, length(t));
x = conj(1/sqrt(length(t)*length(f)) * …
sum( X_matrix .* exp(1i * 2*pi * f * t) ,1));
% Correct power error due to time window
x = x * A_t;
 % magnitude scaling factor
A_f = (f_high - f_low) / f_high
A = sqrt(length(t)/length(f)) / A_t * A_f;
end
```

III. Comparison of Results for DFT and CZT

In this section we show results obtained by using the implementations of the CZT and ICZT in (6) and (5) and compare them to the DFT and IDFT. For this we used the test signals shown in Fig. 1, which are generated by multiplying a 1 ns rectangular window with a 10 GHz carrier as shown in (12) to (14). The sampled signal $s[n]$ uses a sampling frequency (F_s) of 28 GHz.

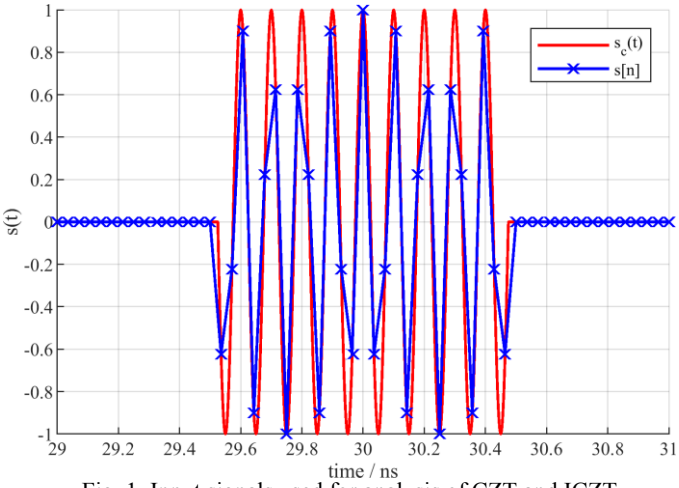

Fig. 1. Input signals used for analysis of CZT and ICZT.

$$s_c(t) = w(t) \cdot cos(2\pi\, t\, f_c) \tag{12}$$

$$w(t) = \begin{cases} 1 & 29.5\ ns < t < 30.5\ ns \\ 0 & otherwise \end{cases} \tag{13}$$

$$\underline{s} = s[n] = s_c(nT) = s_c(n\ 1/F_s) \tag{14}$$

First we needed to calculate the complex spectrum of the signals in order to test the ICZT. For this we used the built in Fast Fourier Transformation (FFT) of Matlab® (15) and our implementation of the CZT (16). If the target frequency vector f is chosen to be identical to the one of the FFT, the CZT and FFT yield identical results as shown in Fig. 2. Using the calculated spectrum we can now apply the ICZT to get back the time domain signal. If the same time vector is used the ICZT yields the original signal s as shown in Fig. 3 and (17), even without the magnitude scaling factor A, which is equal to one for this case.

For (18) we used a different time vector t'', which uses the

same number of samples but starts at 29.2 ns and ends at 30.8 ns. Because we use the same number of samples for a shorter period of time the ICZT now inherently performs an interpolation of the time domain signal which is visualized in Fig. 3. As transformation and interpolation is performed in the same step this is a very efficient implementation, and it can be reversed using the corresponding inverse function with the same time and frequency vectors. Of course only signal components which were not thrown away due to the selected time vector can be restored. In our example the time t'' vector only removed zero samples, leading to no loss of information.

To achieve the correct magnitude for the interpolated signal s'', the signal in Fig. 3 was multiplied by A''. The matching frequency domain representation was calculated in (19). Please note that for this s'' was divided by A'' before the transformation. As the CZT is a linear transformation the scaling could be done at a different stage, but we chose to apply it to the time domain signal before the transformation. The result of (19) is plotted in Fig. 2. You can observe that due to the interpolation the sampling frequency of the signal is much higher than for the original signal, but because the number of points is the same, the frequency spacing between points in the spectrum increased. By using this interpolation, the signal s'' in Fig. 3 looks very similar to the analog signal $s(t)$. Only at the edges of the rectangular window the signal could not follow the quick change because of the missing signal components at higher frequencies.

$$\underline{S}_{FFT} = FFT\{\underline{s}\} \tag{15}$$

$$\underline{S}_{CZT} = myCZT\{\underline{t}, \underline{s}, \underline{f}\} \tag{16}$$

$$\underline{s}' = myICZT\left(\underline{f}, \underline{S}_{CZT}, \underline{t}\right) = \underline{s} \tag{17}$$

$$[\underline{s}'', A''] = myICZT(\underline{f}, \underline{S}_{CZT}, \underline{t}'') \tag{18}$$

$$\underline{S}''_{CZT} = myCZT\left(\underline{t}'', \frac{\underline{s}''}{A''}, \underline{f}''\right) \tag{19}$$

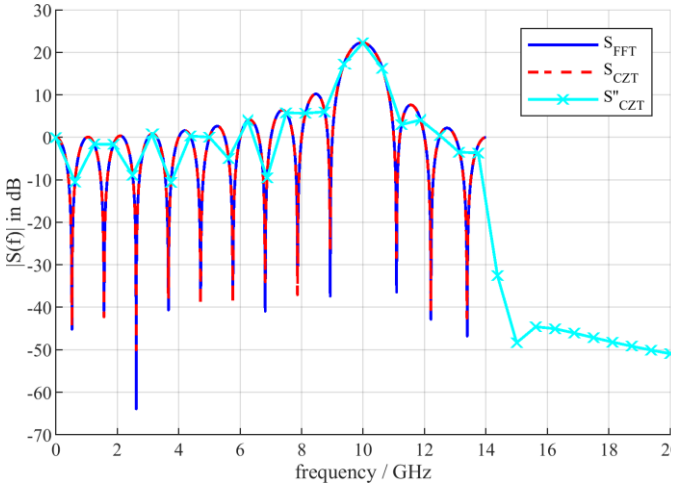

Fig. 2 Frequency domain representation of the sampled test signal using Fast Fourier Transformation (blue line), CZT (dashed red line) and CZT of the time windowed signal, which was created using the ICZT on the FFT result of the sampled signal.

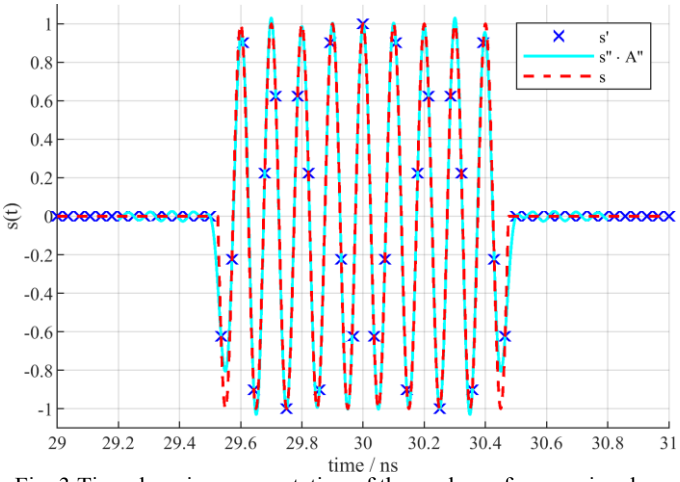

Fig. 3 Time domain representation of the analog reference signal (red), a signal calculated using ICZT with the original time vector $\underline{t}$ (blue) and a signal calculated using a truncated time vector with higher sample rate $\underline{t}''$ (cyan).

Until now we only showed transformations for spectra starting at DC, which could as well be transformed using the IDFT. In Fig. 4 and Fig. 5 we show the impact of applying the ICZT to a spectrum ranging from 3 GHz to 14 GHz. The calculation of the blue spectrum in Fig. 4 is shown in (20) and (21), and simply is the truncated spectrum of the original sampled signal. By applying the ICZT in (22) a real valued time domain signal can be calculated from this spectrum, which is shown as cyan colored line in Fig. 5. Using the same time and frequency vectors as in (19) the frequency domain representation of this signal (S'_{NoDC}), which now contains frequency components down to DC, is calculated in (23) and is also presented in the spectrum plot. What happens if a frequency vector extending over the alias free range is supplied for the CZT is presented in (24) and Fig. 4. For this the original signal s was transformed into frequency domain using the frequency vector of the interpolated signal which causes aliasing. The same occurs for the ICZT if a time vector exceeding the alias free time is supplied for the transformation. A practical implementation of the CZT and ICZT should provide you with a warning in such a case.

$$f_{NoDC} = f|_{3\ GHz<f<14\ GHz} \tag{20}$$

$$\underline{S}_{NoDC} = S_{FFT}|_{f_{NoDC}} \tag{21}$$

$$[\underline{s}_{NoDC}, A_{NoDC}] = myICZT\left(\underline{f}_{NoDC}, \underline{S}_{NoDC}, \underline{t}''\right) \tag{22}$$

$$\underline{S}'_{NoDC} = myCZT\left(\underline{t}'', \frac{\underline{s}_{NoDC}}{A_{NoDC}}, \underline{f}''\right) \tag{23}$$

$$\underline{S}''_{NoDC} = myCZT\left(\underline{t}, \underline{s}, \underline{f}''\right) \tag{24}$$

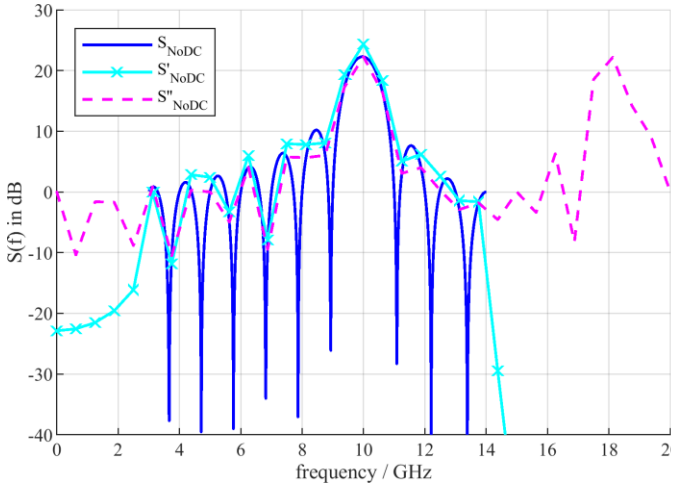

Fig. 4 Plot of a spectrum with missing lower frequency components (blue), the spectrum of a signal after the missing components where extrapolated using ICZT (cyan) and spectrum of the original signal s with extrapolation of higher frequency components above the alias free range using CZT (dashed magenta).

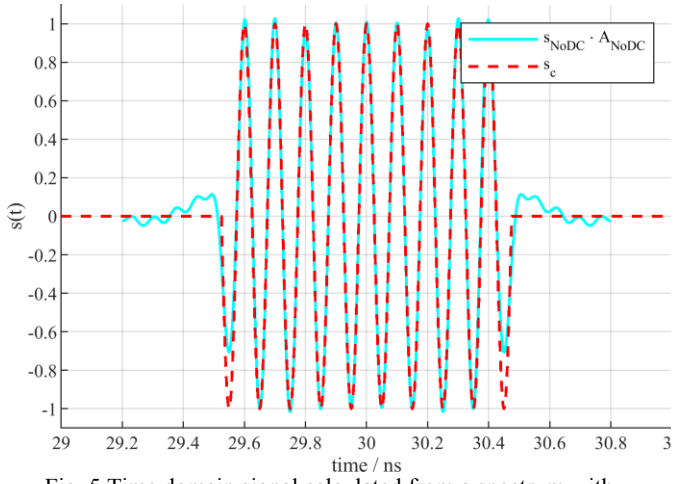

Fig. 5 Time domain signal calculated from a spectrum with missing lower frequency components using ICZT (cyan) and reference signal (red).

IV. Conclusion

In this contribution we present a method for normalization of the time domain signals generated by the ICZT based on the provided frequency and time vectors. Without this normalization a transformation of the same signal with different target time vectors would lead to differently scaled time signals. By multiplying the presented normalization factor with the time domain signal a correct scaling can be achieved. For this method it is mandatory that the provided time and frequency vectors contain equally spaced samples, which is not a formal requirement for the CZT or ICZT itself. For example, two spectra with different frequency ranges and frequency spacing could be combined using ICZT. In such a case the mentioned normalization factor will not be correct and should be omitted. At this point we would also like to mention that if the time and frequency vectors do not change, the scaling factor will also not change, allowing a relative comparison of the transformation results without applying any scaling factors.

V. Acknowledgement

This work was funded by the Austrian Research Promotion Agency (FFG) under the research project UB-Smart (No.: 859475).

References

[1] D. Frickey, "Using the Inverse Chirp-Z Transform for Time-Domain Analysis of Simulated Radar Signals," in *International Conference on Signal Processing Applications and Technology (ICSPAT 94)*, 1994.

[2] L. R. Rabiner, R. W. Schafer and C. M. Rader, "The chirp z-transform algorithm and its application," *The Bell System Technical Journal,* vol. 48, no. 5, pp. 1249-1292, 1969.

[3] W. Yiding, W. Yirong and H. Jun, "Application of inverse chirp-z transform in wideband radar," in *International Geoscience and Remote Sensing Symposium (IGARSS 2001)*, Sydney, Australia, 2001.

[4] V. Sukhoy and A. Stoytchev, "Numerical error analysis of the ICZT algorithm for chirp contours on the unit circle," *Nature,* vol. Article Number: 4852 (2020), 2020.