NON-UNIFORM FILTER DESIGN IN THE LOG-SCALE

Brigitte Bidégaray-Fesquet[1]*, Laurent Fesquet*[2]

[1]Laboratoire Jean Kuntzmann,
CNRS, Université Joseph Fourier, Grenoble INP, Université Pierre Mendès-France, France
[2]Laboratoire TIMA,
CNRS, Grenoble INP, Université Joseph Fourier, France
Emails: {Brigitte.Bidegaray, Laurent.Fesquet}@imag.fr

ABSTRACT

We propose a filtering technique which takes advantage of the possibility of using a very low number of samples for both the signal and the filter transfer function thanks to non-uniform sampling. Following some previous work where we sampled non-uniformly already existing filter transfer functions, we propose now to design directly new filters in the frequency domain. They will be linear with respect to the log-scale. This approach also leads to a summation formula which plays the role of the discrete convolution for classical FIR filters, but with a very reduced number of samples which yields a good efficiency.

***Keywords*—** Non-uniform sampling, filtering, filter design.

1. INTRODUCTION

Reducing the power consumption of mobile systems – such as cell phones, sensor networks and many other electronic devices – by one to two orders of magnitude is extremely challenging but will be very useful to increase the system autonomy and reduce the equipment size and weight. In order to reach this goal, this paper proposes a solution applicable to filtering which completely rethinks the signal processing theory and the associated system architectures.

Today the signal processing systems uniformly sample analog signals (at Nyquist rate) without taking advantage of their intrinsic properties. For instance, temperature, pressure, electro-cardiograms, speech signals significantly vary only during short moments. Thus the digitizing system part is highly constrained due to the Shannon theory, which fixes the sampling frequency at least twice the input signal frequency bandwidth. It has been proved in [5] and [6] that Analog-to-digital Converters (ADCs) using a non equi-repartition in time of samples leads to interesting power savings compared to Nyquist ADCs. A new class of ADCs called A-ADCs (for Asynchronous ADCs) based on level-crossing sampling (which produces non-uniform samples in time) [2, 3] and related signal processing techniques [1, **?**] have been developed.

This work has been supported by a MathSTIC Project of the Grenoble University.

A previous work [4] suggested an important change in the filter design consisting in sampling analog signals and filter transfer functions non-uniformly in time and frequency respectively. Instead of using predefined filters (e.g. Butterworth, Chebyshev,...) which are designed to be uniformly discretized in time, we define here directly new filters which are designed in the loglog scale and originally thought to be used in the frequency domain.

As in [4], non-uniform sampling leads to an important reduction of the weight-function coefficients. We can describe very steep filters with an extremely low number of samples. Combined with a non-uniform level-crossing sampling technique performed by an A-ADC, this approach drastically reduces the computation load by minimizing the number of samples and operations, even if they are more complex.

2. PRINCIPLE AND NOTATIONS

For a large class of signal, non-uniform sampling leads to a reduced number of samples, compared to a Nyquist sampling. This feature has already been used in [1] to design non-uniform filtering techniques based on interpolation. In this work the authors however used a classical (uniform) filter, that is a usual discretization in time of the impulse response.

Then we took advantage of the fact that the filter transfer function (the Fourier transform of the impulse response) is a very smooth function with respect to frequency, and chose to sample the transfer filter in the Fourier domain and approximate it using linear interpolation [4].

Now we go one level up in the the design of filters which are usually calibrated in the loglog scale. We directly define the filter as a linear by part function in the loglog scale.

The initial signals are supposed to be analog ones. The signal which we want to filter is given in the time domain and is denoted by $s(t)$. The filter transfer function is given in the frequency domain and is denoted by $H(\omega)$. The result of the filtering process $x(t)$ is then theoretically the convolution of $s(t)$ with the impulse response $h(t)$ which is the inverse Fourier

transform of $H(\omega)$:

$$\begin{aligned} x(t) &= \int_{-\infty}^{+\infty} h(t-\tau)s(\tau)d\tau, \qquad (1)\\ h(t) &= \frac{1}{2\pi}\int_{-\infty}^{+\infty} H(\omega)e^{-i\omega t}d\omega. \end{aligned}$$

2.1. Signal sampling and linear interpolation

The input signal is sampled using a level crossing scheme yielding samples $(s_n, \delta t_n)$ consisting of an amplitude of the signal (s_n) and the delay (δt_n) since the previous sample. Only the form of these samples are important for the algorithm and not the technique (level crossing) used to obtained them. To give results or describe algorithms we will use the sample times defined as $t_n = t_0 + \sum_1^n \delta t_{n'}$ but computations will be performed using only the time intervals δt_n. We denote by $I_n = [t_{n-1}, t_n]$ the time intervals.

To approximate the theoretical integral formula (1), we form a new analog function from the samples using linear interpolation, which yields

$$\bar{s}(t) = \sum_n [a_n + b_n t]\chi_{I_n},$$

where χ_{I_n} denotes the indicator function of the interval I_n. The coefficients a_n and b_n can be expressed in terms of s_n, s_{n-1}, t_n and δt_n.

In fact these formulae cover the piecewise constant case (only take $b_n = 0$) in three possible forms: constant on intervals I_n or nearest neighbor interpolation, with a possible need to modify the definition of t_n and δt_n in the algorithms.

2.2. Filter description in the loglog scale

Very often filter transfer functions are represented in the loglog scale and show almost linear features in some parts. Let us define a filter which is piecewise linear in this scale. Given some an increasing sequence of frequencies ω_k, we define the frequency intervals $J_k = [\omega_{k-1}, \omega_k]$. Defining the filter in the loglog scale consists in defining $L(\omega) = \ln(H(\omega))$ as a piecewise linear function of the variable $\ln(\omega)$. More precisely, on the interval " $\ln(J_k)$", $L(\omega)$ can be approximated by $\overline{L}(\omega) = \lambda_k^0 + \lambda_k^1 \ln(\omega)$ where the coefficients λ_k^0 and λ_k^1 are defined in function of the amplitudes H_k and the frequencies ω_k by

$$\lambda_k^1 = \frac{\ln(H_k) - \ln(H_{k-1})}{\ln(\omega_k) - \ln(\omega_{k-1})} \text{ and } \lambda_k^0 = \ln(H_k) - \lambda_k^1 \ln(\omega_k).$$

Comming back to the initial frequency domain, on interval J_k we have an approximation on $H(\omega)$ by

$$\begin{aligned} \overline{H}(\omega) &= \exp(\overline{L}(\omega)) = \exp(\lambda_k^0 + \lambda_k^1 \ln(\omega)) = \exp(\lambda_k^0)\omega^{\lambda_k^1}\\ &= H_k \omega_k^{-\lambda_k^1}\omega^{\lambda_k^1} = H_k \left(\frac{\omega}{\omega_k}\right)^{\lambda_k^1}. \end{aligned}$$

On the first interval $]-\infty, \ln(\omega_1)]$ the slope is necessarily zero: $\lambda_1^1 = 0$.

2.3. Digital filter

The digital filter then consists in computing (possibly) for all time

$$\begin{aligned} \overline{x}(t) &= \int_{-\infty}^{+\infty} \bar{h}(t-\tau)\bar{s}(\tau)d\tau,\\ \overline{h}(t) &= \frac{1}{2\pi}\int_{-\infty}^{+\infty} \bar{H}(\omega)e^{-i\omega t}d\omega. \end{aligned}$$

To be able to have explicit formulae for the integrals, as we did in [4] we need to have $\lambda_k^1 \in \mathbb{Z}$. In this paper, we will deal more precisely with low-pass filters and therefore choose negative slopes $(-\lambda_k^1 \in \mathbb{N})$. This is not very restrictive, and we are able to fulfill most templates.

Now $\overline{h}(t)$ can be split as a sum of elementary impulse responses $\sum_{k=1}^K h_k(t)$. A first step to describe the algorithm is therefore first to compute the $h_k(t)$ integrating over frequency intervals. Then the convolution equals

$$\begin{aligned} \overline{x}(t) &= \int_{-\infty}^{\infty} \overline{h}(t-\tau)\overline{s}(\tau)d\tau\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\int_{t_{n-1}}^{t_n} h_k(t-\tau)(a_n + b_n\tau)d\tau\\ &= \sum_{n=1}^{N}\left\{(a_n + b_n t)\sum_{k=1}^{K} h_{nk}(t) - b_n\sum_{k=1}^{K}\hat{h}_{nk}(t)\right\}, \end{aligned}$$

where we have to compute the integrals over time

$$h_{nk}(t) = \int_{t-t_n}^{t-t_{n-1}} h_k(\tau)d\tau \text{ and } \hat{h}_{nk}(t) = \int_{t-t_n}^{t-t_{n-1}} h_k(\tau)\tau d\tau.$$

The next section is devoted to the construction of the elementary contributions $h_{nk}(t)$ and $\hat{h}_{nk}(t)$.

3. ELEMENTARY CONTRIBUTIONS

3.1. Elementary impulse responses

We first compute the elementary impulse responses $h_k(t)$:

$$h_k(t) = \frac{1}{2\pi}\int_{\omega_{k-1}}^{\omega_k}\left(\frac{\omega}{\omega_k}\right)^{\lambda_k^1}\left(H_k e^{i\omega t} + H_k^* e^{-i\omega t}\right)d\omega,$$

which reads as $h_k(t) = \alpha_k h_k^\alpha(t) + \beta_k h_k^\beta(t)$ where

$$\begin{aligned} h_k^\alpha(t) &= \int_{\omega_{k-1}}^{\omega_k} \omega^{\lambda_k^1}\cos(\omega t)d\omega,\\ h_k^\beta(t) &= \int_{\omega_{k-1}}^{\omega_k} \omega^{\lambda_k^1}\sin(\omega t)d\omega \end{aligned}$$

and $\alpha_k = \Re(H_k)/\pi\omega_k^{\lambda_k^1}$ and $\beta_k = \Im(H_k)/\pi\omega_k^{\lambda_k^1}$.

3.1.1. Computation of the elementary contributions

The filtered signal is therefore

$$\begin{aligned}\overline{x}(t) &= \sum_{n=1}^{N}(a_n+b_nt)\sum_{k=1}^{K}(\alpha_k h^{\alpha}_{nk,m_k}(t)+\beta_k h^{\beta}_{nk,m_k}(t)) \\ &\quad -\sum_{n=1}^{N} b_n \sum_{k=1}^{K}(\alpha_k \hat{h}^{\alpha}_{nk,m_k}(t)+\beta_k \hat{h}^{\beta}_{nk,m_k}(t)),\end{aligned}$$

where, denoting $m_k = \lambda_k^1$

$$\begin{aligned}h^{\alpha}_{nk,m_k}(t) &= \int_{t-t_n}^{t-t_{n-1}}\int_{\omega_{k-1}}^{\omega_k} \omega^{m_k}\cos(\omega\tau)d\omega d\tau, \\ h^{\beta}_{nk,m_k}(t) &= \int_{t-t_n}^{t-t_{n-1}}\int_{\omega_{k-1}}^{\omega_k} \omega^{m_k}\sin(\omega\tau)d\omega d\tau, \\ \hat{h}^{\alpha}_{nk,m_k}(t) &= \int_{t-t_n}^{t-t_{n-1}}\int_{\omega_{k-1}}^{\omega_k} \omega^{m_k}\cos(\omega\tau)\tau d\omega d\tau, \\ \hat{h}^{\beta}_{nk,m_k}(t) &= \int_{t-t_n}^{t-t_{n-1}}\int_{\omega_{k-1}}^{\omega_k} \omega^{m_k}\sin(\omega\tau)\tau d\omega d\tau.\end{aligned}$$

We notice that it will be possible using the difference operator

$$\begin{aligned}\Delta_{n,k}(t)(f(\tau,\omega)) &= f(t-t_n,\omega_k)-f(t-t_{n-1},\omega_k) \\ &\quad -f(t-t_n,\omega_{k-1})+f(t-t_{n-1},\omega_{k-1}).\end{aligned}$$

With this notation

$$\begin{aligned}h^{\alpha}_{nk,m_k}(t) &= \Delta_{n,k}(t)(f^{\alpha}_{m_k}(\tau,\omega)), \\ h^{\beta}_{nk,m_k}(t) &= \Delta_{n,k}(t)(f^{\beta}_{m_k}(\tau,\omega)), \\ \hat{h}^{\alpha}_{nk,m_k}(t) &= \Delta_{n,k}(t)(\hat{f}^{\alpha}_{m_k}(\tau,\omega)), \\ \hat{h}^{\beta}_{nk,m_k}(t) &= \Delta_{n,k}(t)(\hat{f}^{\beta}_{m_k}(\tau,\omega)).\end{aligned}$$

We easily compute the values for $m_k = 0$

$$\begin{aligned}f^{\alpha}_0(\tau,\omega) &= -\operatorname{Si}(\omega\tau), \\ f^{\beta}_0(\tau,\omega) &= -\operatorname{Cin}(\omega\tau), \\ \hat{f}^{\alpha}_0(\tau,\omega) &= \tau\operatorname{cosc}(\omega\tau), \\ \hat{f}^{\beta}_0(\tau,\omega) &= \tau\operatorname{sinc}(\omega\tau),\end{aligned}$$

and $m_k = 1$ (where ω cannot be zero since the first filter sample corresponds to $m_1 = 0$)

$$\begin{aligned}f^{\alpha}_1(\tau,\omega) &= \frac{1}{\omega}[\sin(\omega\tau)-\omega\tau\operatorname{Ci}(\omega\tau)], \\ f^{\beta}_1(\tau,\omega) &= \frac{1}{\omega}[-\cos(\omega\tau)-\omega\tau\operatorname{Si}(\omega\tau)], \\ \hat{f}^{\alpha}_1(\tau,\omega) &= \frac{1}{2\omega^2}[\cos(\omega\tau)+\omega\tau\operatorname{sinc}(\omega\tau]-\omega^2\tau^2\operatorname{Ci}(\omega\tau)), \\ \hat{f}^{\beta}_1(\tau,\omega) &= \frac{1}{2\omega^2}[-\omega\tau\cos(\omega\tau)+\sin(\omega\tau)-\omega^2\tau^2\operatorname{Si}(\omega\tau)].\end{aligned}$$

The formulae uses special functions. Besides the well known sine cardinal $\operatorname{sinc}(x) = \sin(x)/x$ and cosine cardinal $\operatorname{cosc}(x) = (\cos(x)-1)/x$ function, there are also the less common sine integral function defined by

$$\operatorname{Si}(x) = \int_0^x \sin(y)\frac{dy}{y},$$

and cosine integral functions

$$\begin{aligned}\operatorname{Ci}(x) &= -\int_x^{\infty}\cos(y)\frac{dy}{y}, \\ \operatorname{Cin}(x) &= \int_0^x (1-\cos(y))\frac{dy}{y}.\end{aligned}$$

To attain larger m, we can (a) derive similar formulae for all $m \in \mathbb{N}$ or (b) decompose every integer slope filter, in products of filters with slopes 0 or -1. Choice (b) is simpler to implement since you only have to implement simple filters and sequence them. Choice (a) is more computationally efficient.

A simple calculation shows that the elementary contributions can be derived thanks to induction. This is given by formula (2). In this formula $F(x) = \operatorname{Si}(x)$ and $G(x) = \operatorname{Ci}(x)$ if m is even and $F(x) = \operatorname{Ci}(x)$ and $G(x) = \operatorname{Si}(x)$ is m is odd. The coefficients $P_m(x)$, $Q_m(x)$, $\hat{P}_m(x)$, $\hat{Q}_m(x)$ are polynomials, and $C_m = m!$ and $\hat{C}_m = (m-1)!(m+1)$ are integers. The polynomials are computed via induction:

$$\begin{aligned}P_m(x) &= xQ_{m-1}(x), \\ Q_m(x) &= (m-1)! - xP_{m-1}(x), \\ \hat{P}_m(x) &= (m-1)! + xP_m(x), \\ \hat{Q}_m(x) &= xQ_m(x).\end{aligned}$$

4. EXAMPLE

To show the filtering results via this algorithm we use a frequency-sweep input signal

$$s(t) = 0.9(\cos(2\pi t^2)+1).$$

This signal is plotted with the solid line in Figure 1. The signal has first been sampled using a 3-bit asynchronous analog to digital converter with $[0, 1.8]$ range which performs level crossing sampling, yielding the 88 samples plotted with stars in Figure 1. Two different filters have been applied. The cut-off frequency is $\omega_1 = 4\pi$, and we define a -5 slope for the first filter between ω_1 and $\omega_2 = 30\pi$ and -10 for the second filter. The result is plotted on Figure 1. As awaited when the cut-off frequency is reached the subsequent signal is more damped with the order 10 filter.

5. CONCLUSION

As in [4], we have defined a filtering technique for non-uniform signals which involves a sampled filter in the frequency domain. To sample the signal we needed about 10 samples to sample correctly a filter in [4], where the filter was supposed to be linear in frequency. Here we suppose linearity but in the log scale both in amplitude and frequency. This corresponds to usual definitions

$$
\begin{aligned}
f_m^{\alpha}(\tau,\omega) &= \frac{1}{C_m\omega^m}\left(P_m(\omega\tau)\cos(\omega\tau)+Q_m(\omega\tau)\sin(\omega\tau)-(-1)^{[m/2]}\omega^m\tau^m F(\omega\tau)\right),\\
f_m^{\beta}(\tau,\omega) &= \frac{1}{C_m\omega^m}\left(-Q_m(\omega\tau)\cos(\omega\tau)+P_m(\omega\tau)\sin(\omega\tau)+(-1)^{[(m+1)/2]}\omega^m\tau^m G(\omega\tau)\right),\\
\hat{f}_m^{\alpha}(\tau,\omega) &= \frac{1}{\hat{C}_m\omega^{m+1}}\left(\hat{P}_m(\omega\tau)\cos(\omega\tau)+\hat{Q}_m(\omega\tau)\sin(\omega\tau)-(-1)^{[m/2]}\omega^{m+1}\tau^{m+1}F(\omega\tau)\right),\\
\hat{f}_m^{\beta}(\tau,\omega) &= \frac{1}{\hat{C}_m\omega^{m+1}}\left(-\hat{Q}_m(\omega\tau)\cos(\omega\tau)+\hat{P}_m(\omega\tau)\sin(\omega\tau)+(-1)^{[(m+1)/2]}\omega^{m+1}\tau^{m+1}G(\omega\tau)\right).
\end{aligned}
\tag{2}
$$

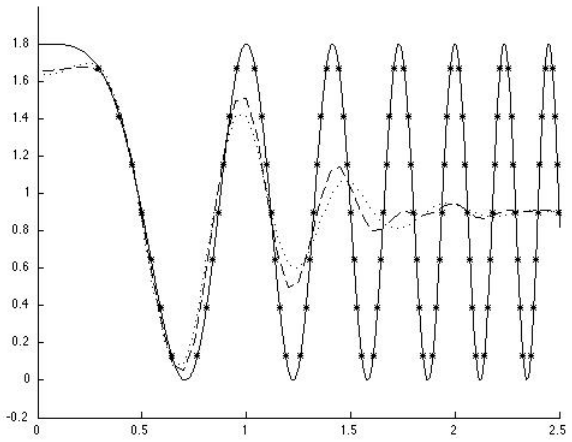

Fig. 1. Filtering of a frequency-sweep signal. Initial continuous signal: solid line; non-uniform samples: stars; filtered signal with an order 5 filter: dashed line; filtered signal with an order 10 filter: dotted line.

of templates for filter design. Besides, this enables to correctly represent a low pass filter with two frequency samples as in our example. This induces a reduction of the complexity in the algorithm and therefore a reduction of the activity and consumption in future hardware implantations, making such an approach suitable for mobile applications.

In the MATLAB implementation used to produce the example, the order of the filter has a very low impact on the complexity of the algorithm, since the evaluation of the sine integrals and cosine integrals are very penalizing. Using a look-up table for these evaluations should prove useful for this implementation and absolutely necessary for an hardware implantation. Even with this problem fixed, the extra-complexity of a high-order filter compared to a low-order one lies only in the evaluation of the polynomials in (2). Computing with high-order filters is therefore affordable for practical applications, all the more as we have a very low number of non-uniform time samples to process.

6. REFERENCES

[1] Fabien Aeschlimann, Emmanuel Allier, Laurent Fesquet, and Marc Renaudin. Asynchronous FIR filters: Towards a new digital processing chain. In *10th International Symposium on Asynchronous Circuits and Systems, Async'04*, pages 198–206, Hersonisos, Crete, April 2004. IEEE.

[2] Filipp Akopyan, Rajit Manohar, and Alyssa B. Apsel. A level-crossing flash asynchronous analog-to-digital converter. In *12th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'06)*, pages 11–22, Grenoble, France, March 2006.

[3] Emmanuel Allier, Gilles Sicard, Laurent Fesquet, and Marc Renaudin. A new class of asynchronous A/D converters based on time quantization. In *9th International Symposium on Asynchronous Circuits and Systems, Async'03*, pages 196–205, Vancouver, Canada, May 2003. IEEE.

[4] Brigitte Bidégaray-Fesquet and Laurent Fesquet. A fully non-uniform approach to FIR filtering. In Laurent Fesquet and Bruno Torrésani, editors, *8th International Conference on Sampling Theory and Applications (SampTa'09)*, Marseille, France, 2009.

[5] Jon W. Mark and Terence D. Todd. A nonuniform sampling approach to data compression. *IEEE Transactions on Communications*, 29(1):24–32, January 1981.

[6] Necip Sayiner, Henrik V. Sorensen, and Thayamkulangara R. Viswanathan. A level-crossing sampling scheme for A/D conversion. *IEEE Transactions on Circuits and Systems II*, 43(4):335–339, April 1996.