PART III - Subbands

1 PREFACE

1.1 NOTES

Read Part I first.

I apologise for endless repetition of the words "that will be explained later". There is no way to explain FSAF in a linear, consecutive way, as a narrative. Practically every part of it is tightly interconnected with others, and it can not be explained on its own, so a reader will have hard time to understand it.

The order of explanation is not the order of design. Finding Perfect Reconstruction (PR) synthesis filer for a given analysis filter is easy (if it exists - not always), and non-essential. But we start discussion with it due to reasons we'll understand later.

The names of chapters with pictures include a reference [3xy] to a doc_p3xy.m script which was used to generate these pictures.

1.2 SUMMARY

- The "plant" to be identified by a per-subband adaptive filter must be *band-limited*, otherwise the meaning of all the rest is not unquestionable. The definition of "*band-limited*" is not simple.
- A digital delta-function in full band RIR is translated into an infinite non-casual spread in subbands. The shape of the spread is defined by *IN(t)* and *OUT(t)* filters and denoted by DSF.
- In some cases, DSF can be predicted analytically from *IN(t)* and *OUT(t)* filters. In general case, DSF can be reconstructed from statistical modelling.
- IO aliasing shall be adequately accounted for, for it puts a strict limit to convergence.
- Under-modelling of sub-band RIR translations is a non-negligeable factor. Non-causal extension of sub-band RIR translation may be necessary.
- Even if there is no AWGN at OUT (microphone), there always is a noise at residual whenever IN is present regardless of adaptive filter convergence state.
- Full-band RIR can be easily reconstructed from per-subband RIR translations (back and forth, really) using DCF (which is a function orthogonal to the DSF).
- Equalization of sub-sampled IN is crucial for the adaptive filter performance.
- …especially for RLS-class adaptive algorithms which otherwise diverge.
- For the chosen *OUT(t)* prototype filter, there must be a reasonable bi-orthogonal *RES(t)* synthesis filter.
- Analysis / synthesis filters do not have to be symmetric. The minimal phase filters are not without problems.
- FSAF architectures might be quite elaborated.

2 OUT/RES CO-DESIGN

- The resampling ratio, R, is assumed here to be equal to integer 2 because otherwise it's a mess.
- Input comes in blocks of M real numbers, the output is M+1complex subbands, except that the first and the last subbands are real, so we get K=2*M numbers out of M on the input.
- The transformation is classical DFT, not GDFT:
  - The first subband samples 0…Fs/M components, and aliases the rest by reflection so that Fs/M+δ -> Fs/M-δ, 2*Fs/M+δ -> δ, etc

- The second subband samples 0….2*Fs/M components, and aliases the rest by wrapping around, so that 2*Fs/M+δ -> δ, same for m*2Fs/M+ δ.
- The third subband samples Fs/M…3*Fs/M components and moves then into – Fs/M …+Fs/M, and aliases the rest by wrapping around so that 3*Fs/M+δ -> -Fs/M+δ
- The fourth, sixth, eighth, etc subband follows the pattern of the 2nd subband.
- The fifth, seventh, etc subband follows the pattern of the 3nd subband.
- The last subband follows the pattern of the 1st subband

- The transformation uses the same prototype filter for all subbands, of length of L*M; even L >=6.
- The transformation is of perfect reconstruction type. The inevitable errors must be kept below 16bit linear PCM quantization level (-98dB) so that it's much easier to debug and test for bit-exactness.
- The prototype filters are usually bi-orthogonal and non-symmetric.

2.1 BASICS

Mathematical theories are often explained from the view point of abstract mathematics, from the depth of a black hole. However, most of modern mathematics we use today was born from and with modern mechanics, and no formula was used in vain. Every equation had profound mechanical sense and could be demonstrated and/or visualized, and there was no magical formalism involved. Newton's laws and proofs were based on geometry, not on abstract algebra (although he could do both, afaik). I also believe that there is no sense in writing equations without understanding of their physical meaning.

2.1.1 FIR Multirate signal processing

There is no such a thing as sample-based multirate processing. Multirate signal processing is essentially a block-based processing. The length of a block is commonly denoted as "***M***".

Every sample inside each block shall be treated in a "democratic" way. The invariance to the sample's offset (inside this block) is THE most important property of proper multirate processing. There could be "internal" differences but there could not be externally-visible, full-band time or frequency domain differences. Otherwise, the underlying LTI condition breaks and everything falls apart.

- For a regular "full-band" FIR-based LTI system, we "knock" with a digital δ-function to observe its Impulse Response (IR) which is equal to that *FIR(1:1:end).*
- In a multirate FIR-based LTI system, there are two FIR filters: on analysis (ANA) stage and on synthesis (SYN) stage.
  - For a multirate FIR-based analysis stage, we "knock" with ***M*** consecutive digital δ-functions to observe ***M*** sub-sequences of Impulse Responses (IR) which are equal to FIR_{ANA}*(**m**:**M**:end),* where ***m*** is the offset of those δ-functions within the input block of ***M*** samples.
  - In other words, multirate analysis' filtering and down-sampling produces offset-specific sub-sequences of FIR_{ANA}.
  - With suck knocking, FIR_{ANA} become identifiable, even for a "black box" multi-rate system.
  - We can always reconstruct FIR_{ANA} by observing these sub-sequences as *fanas(**L**,**M**)* and combining them as:

    for m =1:M, firin(m:M:end)=fanas(:,m);end

  - How can we observe these sub-sequences? That depends on the type of transform used. If the transform is a standard DFT, then all we need is to look at is the first, real-valued DC band. In other cases, the "observability" may be quite challenging as we may need to "un-modulate".

[TBD diagram]

- The synthesis is a mirrored analysis FIR, with its own *fsyns(**L**,**M**)*; The output of "clear-channel" multirate processing, for each offset ***m,*** is (sorry for oversimplification) the convolution of their sub-sequences:

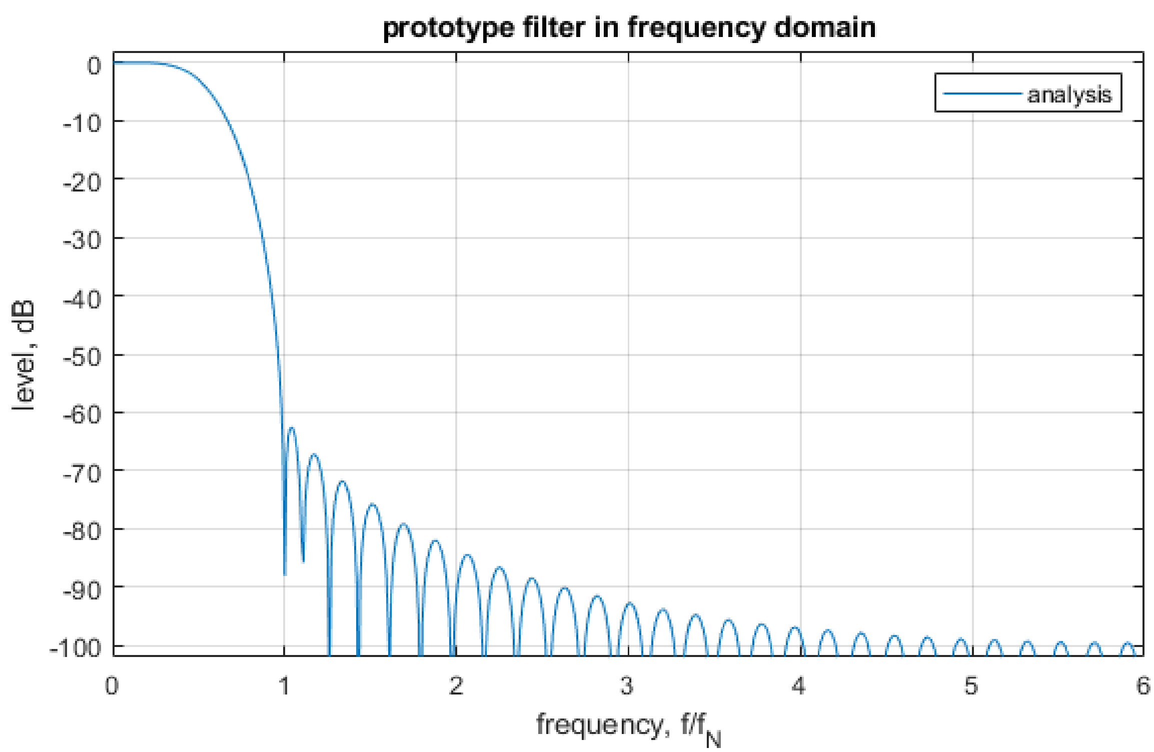
  out(m)=conv(fanas(:,m), fsyns(:,m));

If we require Perfect Reconstruction (PR) multirate processing, we have to ensure that *fanas(:,m)* and *fsyns(:,m)* are $\delta(\tau)$-correlated for each offset ***m***, with the same τ. In a multirate jargon, we say that analysis and synthesis functions are bi-orthogonal.

Note that we did not mention any frequency filtering properties. Indeed, PR FB itself works for with any filters, frequency wise. The adaptive filter, however, does not.

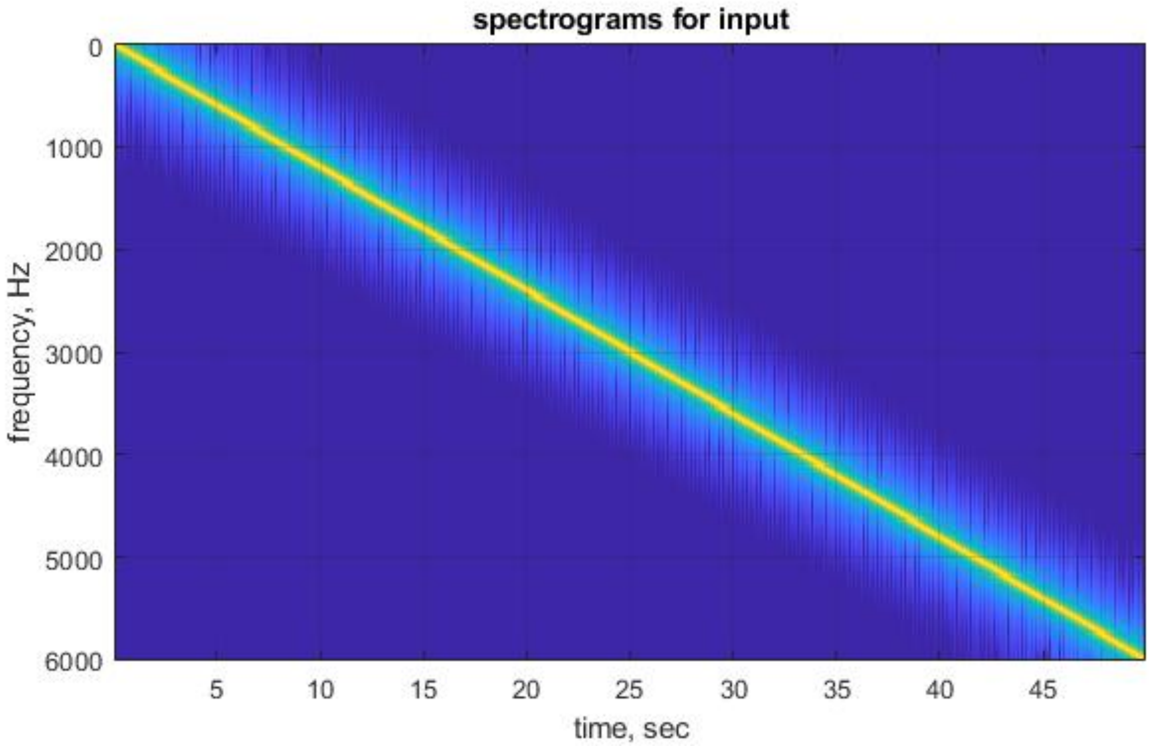
2.1.2 Frequency domain [312]

From adaptive filter's point of view, analysis FIR shall be a very good LPF, with minimum signal in the stopband because adaptive filter is fully uncapable of distinguishing between passband and aliased stopband. We need about 2x oversampling (over critical) to let only stopband to become aliased.

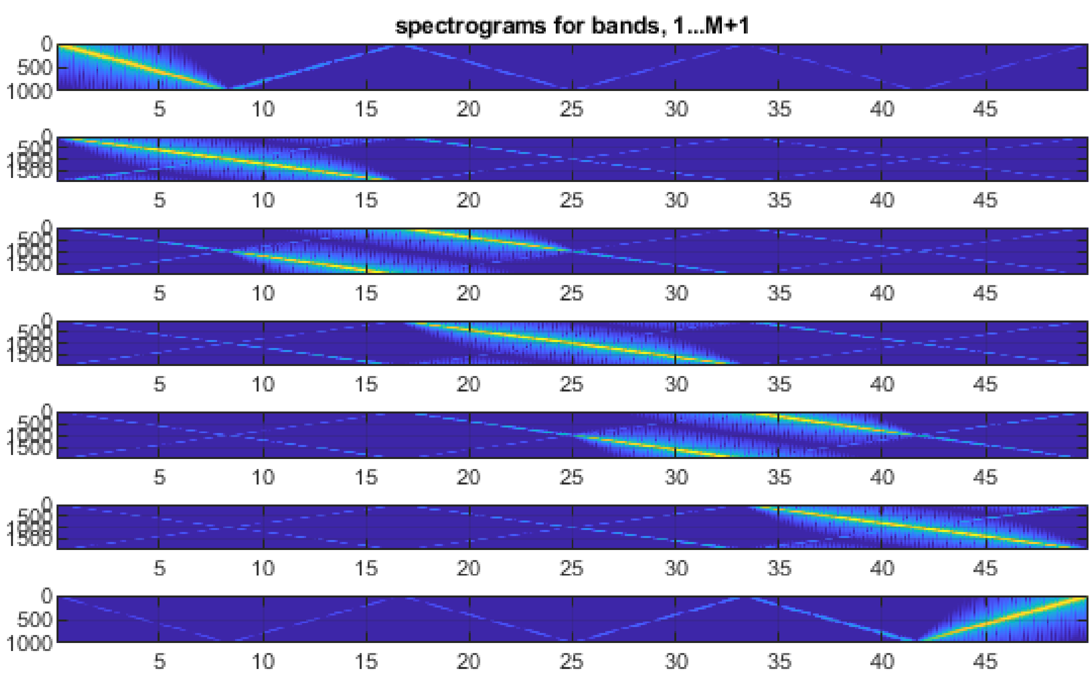


TBD: more on bandpass sampling, I/Q bandpass sampling, aliasing.

In the case of 2x oversampling, M=6:



And outputs of analysis filterbank, for bands 1…M+1 (or 0…M):



For 1st and last subbands, frequencies reflect. For all others, with complex sub-sampling, frequencies wrap around a cylinder as a spiral (much alike the famous DNA), from 0 to 2kHz (0…2*π) which becomes 0, and so on. For even bands, center frequency is at 1kHz. For odd: at 0 (or 2kHz).

2.1.3 Time domain: 2x-oversampling

R=1: L*2-1 equations for L variables?

R=2: L-1 equations for L variables.

2.1.4 Perfect reconstruction

In the context of multirate signal processing, for DFT-based prototype-modulated FB designs, a pair of prototype filters $f(t), h(t)$ is called by-orthogonal wherever perfect reconstruction can be achieved with these filters:

$\sum_{m=-\infty}^{\infty} f(n-mM)h(mM-n+sK) = \delta(s)$; for all n and s [Rabiner & Croshiere, (7.16)]

As explained above, this terminology is misleading. The difference from standard functional analysis definitions is that instead of convolving the functions f(.) and h(.), their subsequences are convolved, and all such subsequences must correlate with the same $\delta(s)$ [= 1 for s=0; and 0 otherwise].

For K=2*M, this falls apart onto M independent linear systems of L-1 equations with L unknowns. We do not have to deal with large hard-to-comprehend system of equations. Instead, we can easily SVD factorise such small matrices, look at their spectra, optimise them, etc.

Let's define subsequences:

$$f_n = [f(n)\ \ f(n+M) \ldots f(n+mM) \ldots f(n+(L-1)M]'$$

$$g_n = [g(n)\ \ g(n+M) \ldots g(n+mM) \ldots g(n+(L-1)M]'; where\ g = h(end{:}-1{:}1);$$

Then, for example, when L = 6:

$$f_n(5)g_n(1) + f_n(6)g_n(2) + 0 * g_n(3) + 0 * g_n(4) + 0 * g_n(5) + 0 * g_n(6) = 0;$$

$$f_n(3)g_n(1) + f_n(4)g_n(2) + f_n(5)g_n(3) + f_n(6)g_n(4) + 0 * g_n(5) + 0 * g_n(6) = 0;$$

$$f_n(1)g_n(1) + f_n(2)g_n(2) + f_n(3)g_n(3) + f_n(4)g_n(4) + f_n(5)g_n(5) + f_n(6)g_n(6) = 1;$$

$$0 * g_n(1) + 0 * g_n(2) + f_n(1)g_n(3) + f_n(2)g_n(4) + f_n(3)g_n(5) + f_n(4)g_n(6) = 0;$$

$$0 * g_n(1) + 0 * g_n(2) + 0 * g_n(3) + 0 * g_n(4) + f_n(1)g_n(5) + f_n(2)g_n(6) = 0;$$

Thus, we have a matrix equation $A(f_n)g_n = b$; where the vector b=zeros(L-1,1) except for "1" at row v, i.e. $b(v) = 1$. For symmetric filters, $v = L/2$. The resulting analysis-synthesis filterbank latency is

$$\tau = (2v-1)M;$$

For asymmetric low(er) latency filterbank design, move the "1" towards 1st row, $1 \le v < L/2$. It does not have to be in the middle, in the general case.

The $A(f_n)$ matrix for L = 6:

$f_n(5)$	$f_n(6)$				
$f_n(3)$	$f_n(4)$	$f_n(5)$	$f_n(6)$		
$f_n(1)$	$f_n(2)$	$f_n(3)$	$f_n(4)$	$f_n(5)$	$f_n(6)$
		$f_n(1)$	$f_n(2)$	$f_n(3)$	$f_n(4)$
				$f_n(1)$	$f_n(2)$

For L=8, we have 7 rows with full middle row.

A(L/2, 1:end) = f_n*';*

The upper rows (3,2,1) is shifted left by 2, 4, and 6 correspondingly:

if row < L/2, A(row, 1:end-R)=A(row+1, 1+R:end); end;

The lower rows (5,6,7) is shifted right in the same manner:

if row > L/2, A(row, 1+R:end)=A(row-1, 1:end-R); end;

There are infinite number of solutions due to the number of rows = number of columns – 1, which is a degree of freedom. We can perform pseudo-inverse (by SVD or other means) to obtain the minimal-norm synthesis filter. Unfortunately, most $f(n)$, forming a good LP filter, have somewhat awkward $g(n)$ with minimal norm. To improve it, we can use the existing degree of freedom.

We take the $g_n\ (k)$ which has maximum absolute value, and move it to the right side. The resulting square *(L-1)*(L-1)* system has an solution. The rest is optimising the sub-vector $g\ ((k-1)*M+n)\ ;\ n=(1{:}M)$ so that the resulting $g(t)$ has the optimal frequency response in a particular sense. Such optimization is not too time-consuming because we can perform SVD factorisation, examine singular value spectrum, and invert the $A(f_n)$ only once, before calling optimization methods.

We can also look at the $A(f_n)$ as a strange case of system identification, with each second observation is missing. Or, a case when σ_t^2 alternates between 0 and ∞. We know quite a bit of what *g(t)* and *DFT(g(t))* shall look like. Thus, we can use ReRLS (ReLS / WRLS) instead of formal optimization.

2.1.5 Near Perfect Reconstruction

We use only SVD for solving such poorly defined linear systems. The SV inversion and thresholding are performed according to a generalised Wiener $z=s^{p-1}/(thr^p+s^p)$; to "smoothly" avoid division by zero.

If $s >> thr$: then $z=1/s$;

If $s << thr$, then $z=(s/thr)^{p-1}/thr$;

If $p == 2$ (default) then $z=(r/s)$ where $r=s^2/(thr^2+s^2)$;

If $p\to\infty$ we get "standard" SV thresholding… which appropriateness [to FSAF] isn't undebatable.

The real sub-band "perfect reconstruction" errors are limited by the floating-point implementations. For "double" precision, it's about 300dB; for "single": 145dB. The reconstruction errors grow with *thr* increase, however while the "single" floating-point implementation errors are outside of dynamic range of modern ADC the reconstruction shall be pronounced "perfect".

There are people believing in universal MP3-like "thresholds of audibility". I am not one of them. Too many of my friends with N generations of musicianship in their families call it "cellophane radio" and I consciously refrain from using subband filter banks with reconstruction errors above -100dB. There may be a way to design a FB with totally inaudible distortions above -100dB but I don't know how to ensure it. Thus, the NPR range here is from -145 to -100dB.

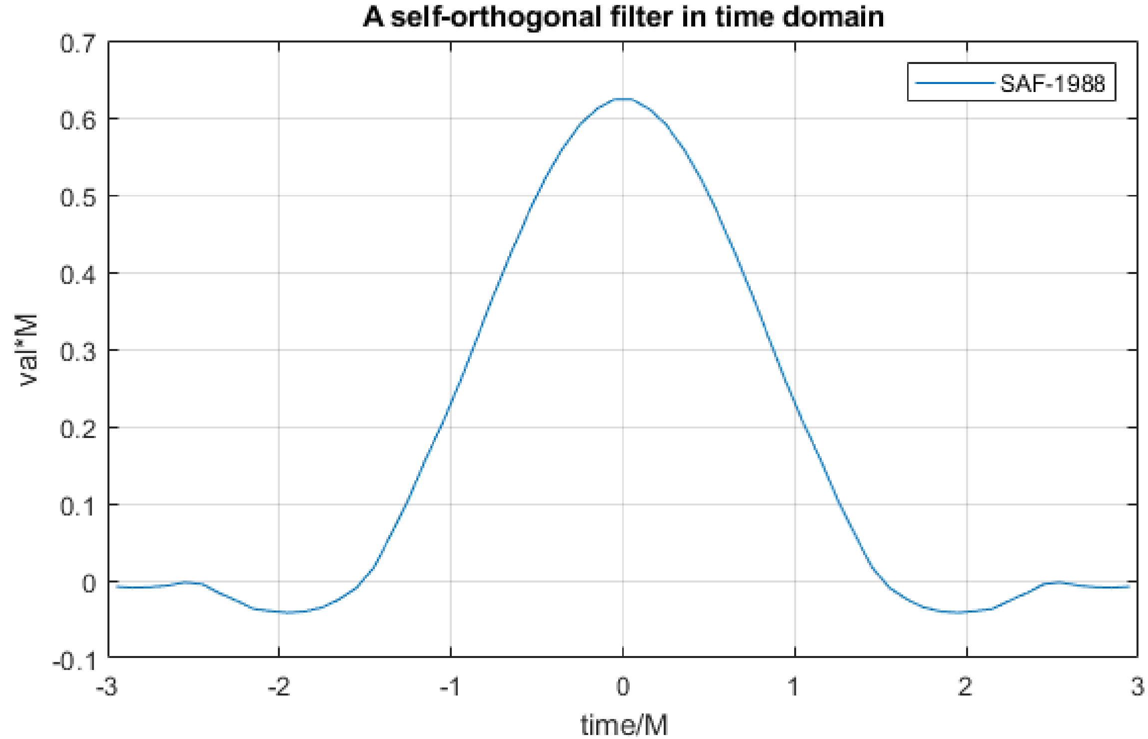
More about allowed time and frequency distribution of errors is TBD.

2.1.6 Examples

First, we'll illustrate the design process on the example of short prototype filter for wide subbands. Let's start with M=10 and L=6. Higher values of M and L will be discussed later.

2.2 SAF1988 FILTERBANK [331]

The traditional design is based on re-using the same filter for IN, OUT, and RES filter banks, which requires a filter which is bi-orthogonal to itself.



This is not the only self-orthogonal filter for given M & L. There are others, with lower sidelobes but worse crest-factor frequency-wise, and vice versa. All of them share a few basic properties.

The self-orthogonal filters are weird: the sidelobes are negative, with zero crossings at ±M*1.5 and ±M*2.5. This is easy to explain: according to the orthogonality equation no.1 and symmetry, for n=M/2:

$$h_1(n)h_1(n) + h_2(n)h_2(n) = 0$$

$$h_1^2(n) + h_2^2(n) = 0; -\!-\!\gg h_1(n) = 0 \cap h_2(n) = 0$$

Analogously, we can show that for n=0:

$h_1(0) = -\frac{h_3(0)}{h_2^2(0)}$; where *$h_1(0)$ = h(M*0), $h_2(0)$ = h(M*1), $h_3(0)$ = h(M*2)*, and *h(0) ~h(1)*

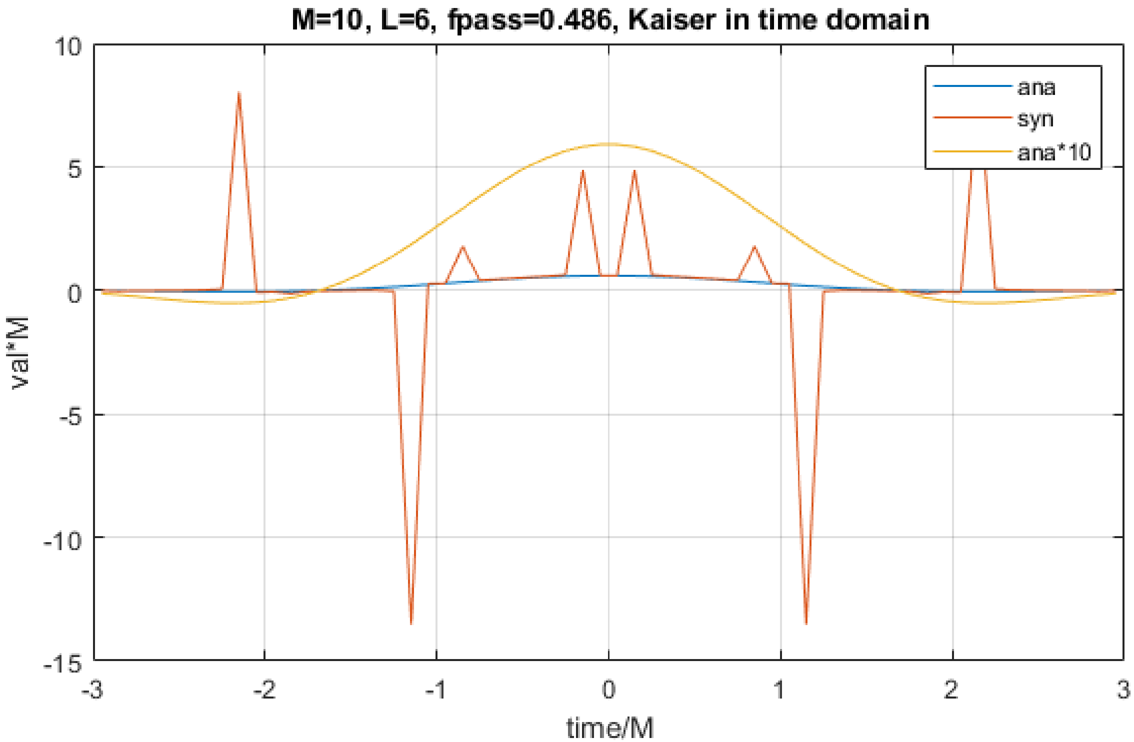
which means that if main lobe is positive, then the second sidelobe must be negative, that both sidelobes are negative, etc. Of course, such filter may not have a good frequency response. It makes me wonder why self-orthogonal-filter-based designs stayed with us for so long …

2.3 FSAF FILTERBANK

We know that some analysis prototype filters "work" while others "don't work", and that we can't tell "who is who" by just looking at them, they all look "innocent".

2.3.1 An example of flaky design [332]

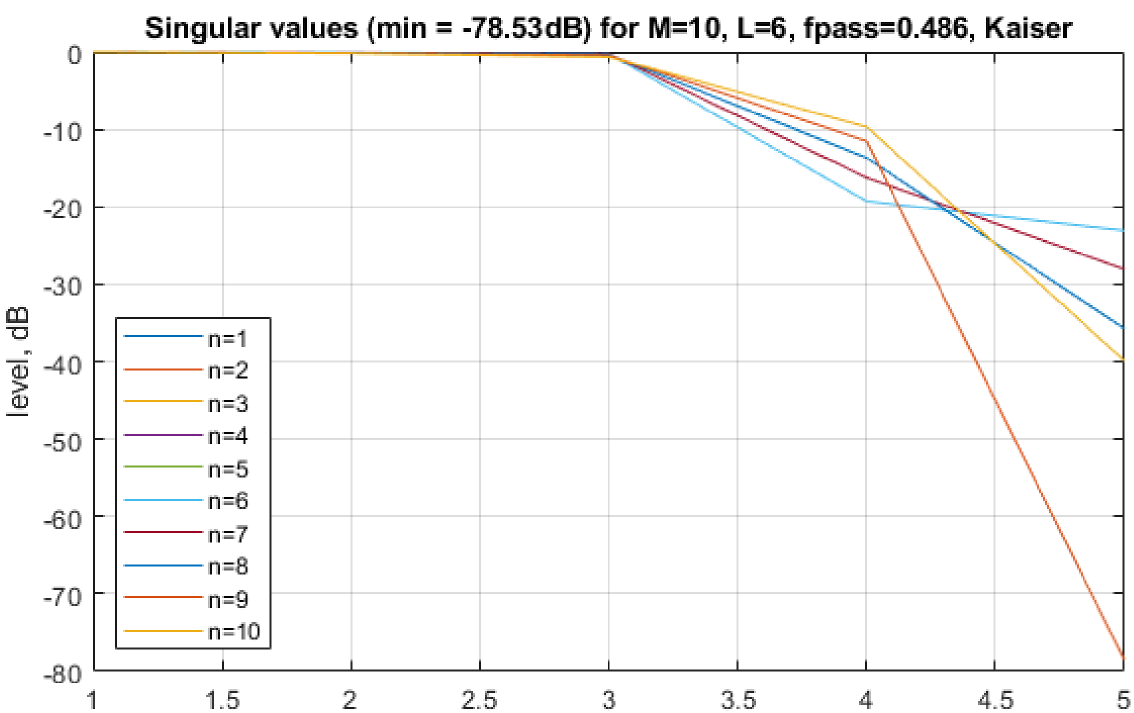
Suppose we'd like to start with an analysis prototype filter having first null at *1/M*, -3dB passband on 0.486/M, derived from a Kaiser window with *fir1()* function, using *fsaf_mkio.out_make(…)* to figure out *Wn* and β, given the passband and stopband. The analysis filter looks fine, but the corresponding minimal norm SVD synthesis filter is horrible:



The outliers, in time domain, produced by the f_n subsequence with $n=9$ (and 2, symmetrically).

2.3.2 Eigenvalue spread at the roots of the problem [332]

The same subsequence in SVD values is also sharply different from the rest:



2.3.3 Robustness Diagram [332,333]

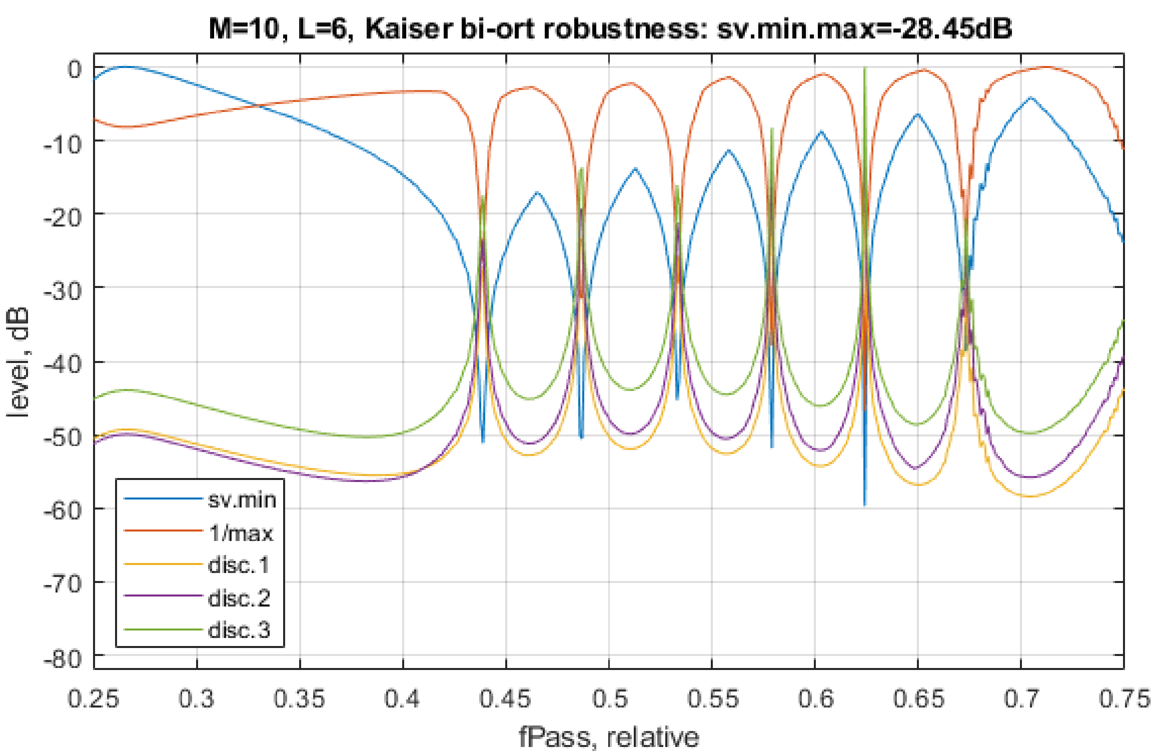
Thus, the design of FSAF FB must start with understanding of the underlying spectral (eigen) distribution. Otherwise, the design of entire FSAF is on shaky ground, and it is not worthy of proceeding any further because such SAF will fall apart sooner or later regardless of how advanced other parts of SAF design are.

Unfortunately, I am not aware of formal, analytical ways of performing such robustness analysis, and alas, we can not assume robustness of an idea however splendid it looks when it first comes to mind. Therefore, the check for FSAF FB robustness shall be numerical and explicit.

The class *fsaf_svcheck.m* is designed to perform this check and display results in a digestible form. A typical use case:

```
sv=fsaf_svcheck(M,L);
wintype='Kaiser';
range=(0.25:0.001:0.75); % range of fPass
sv.plot(1,wintype,range); % figure(1)
```

The supported parametric window types are *Tukey, Gauss and Kaiser* as *name(M*L,β)*. For *Hann, Hamming, Blackman, BlackmanHarris, Bohman* and *Parzen* the *β* parameter is used as *name(M*L).^β.* The usage of such operation is a bit unconventional. The *range* of *fPass* values is self-explanatory.

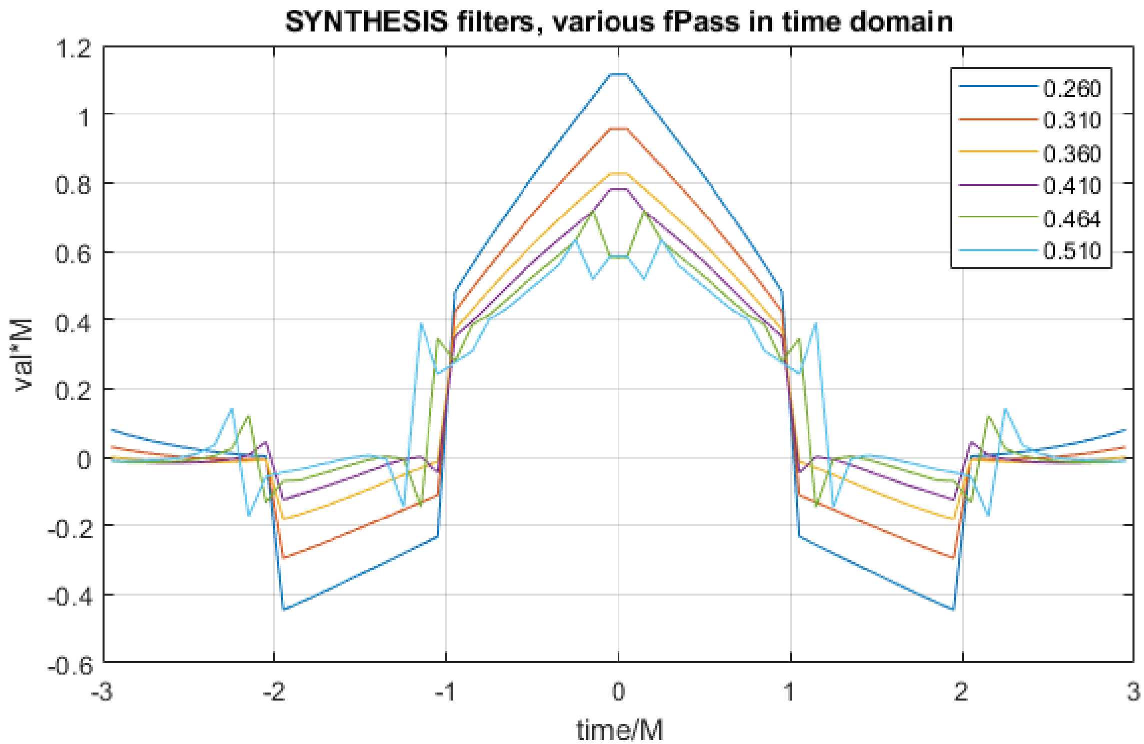


- x-axis is -3dB passband width *fPass*, relative to *fStop* = 1/M.
- "sv.min" curve is the singular value distribution spread for the bi-orthogonality condition system of linear equation for the analysis filter with *fPass*, i.e. the amplitude of minimal singular value for the worst subsequence, relative to the best case of SVD (the number in the title line).
- "1/max" curve is the *max()-min()* of corresponding minimal-norm bi-orthogonal filter, with low dB values corresponding to large outliers and flaky design.

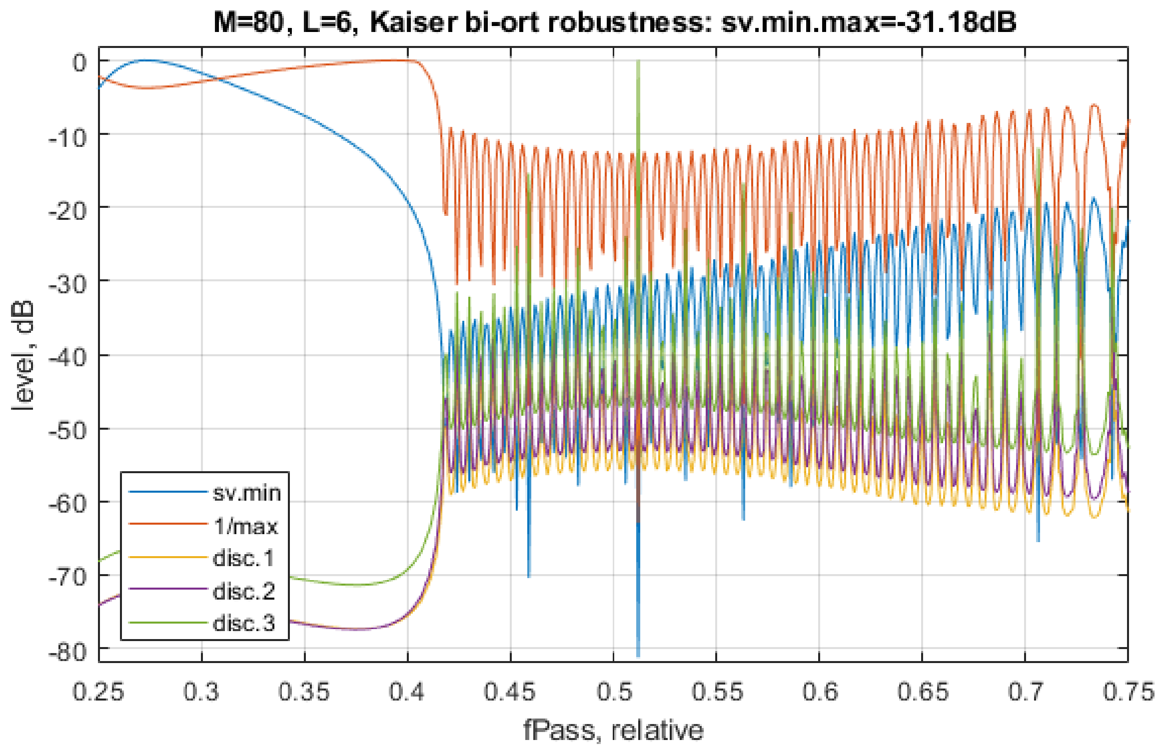
- "disc.n" is *max()-min()* of n-th difference, indicative of discontinuity is the n-th derivative of minimal-norm bi-orthogonal filter which characterise the level of frequency-domain sidelobes.
- display in dB format allows inspecting very small, near zero values.

The curves are different for each window type. The spikes (very low values of sv.min) mean that the rank of the bi-orthogonality condition system is less than L-1, there are more degrees of freedom, and we need to look for other ways to design a perfect reconstruction FB. It might be quite possible – but I don't know how.

At first, we could think that we can freely choose the *fPass* values in the valleys between spikes but it is not so.



The resulting synthesis prototype filters for *fPass* less than then first spike look good, and those for *fPass* in the valleys above the first spike do not look good. Moreover, the robustness diagram for high M shows that there is a principal difference between analysis / synthesis filters for *fPass* less than the first spike (here ~0.42 for *Kaiser* – based *fir1()* designs) and the rest:



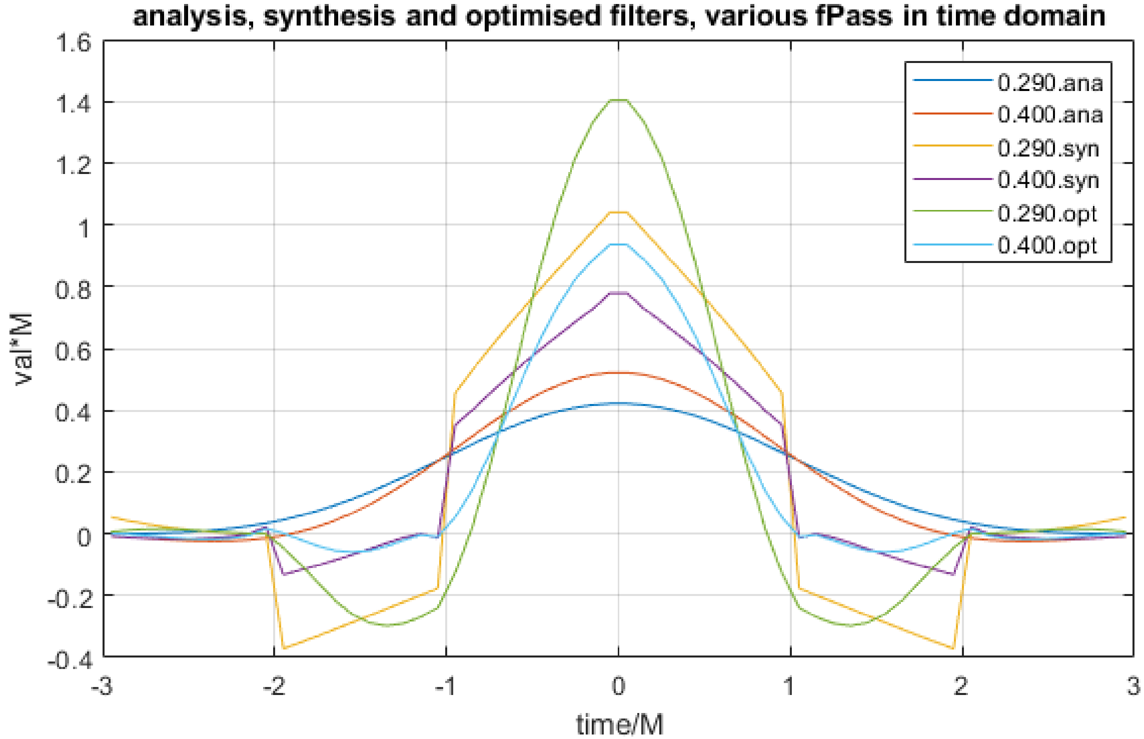
The high-*fPass* analysis *Kaiser-window* filter based filterbank does not lead to a good synthesis filter and is not robust to M resampling. Generally, all window types are equal but some types are more equal than others [for certain ranges of *fPass*].

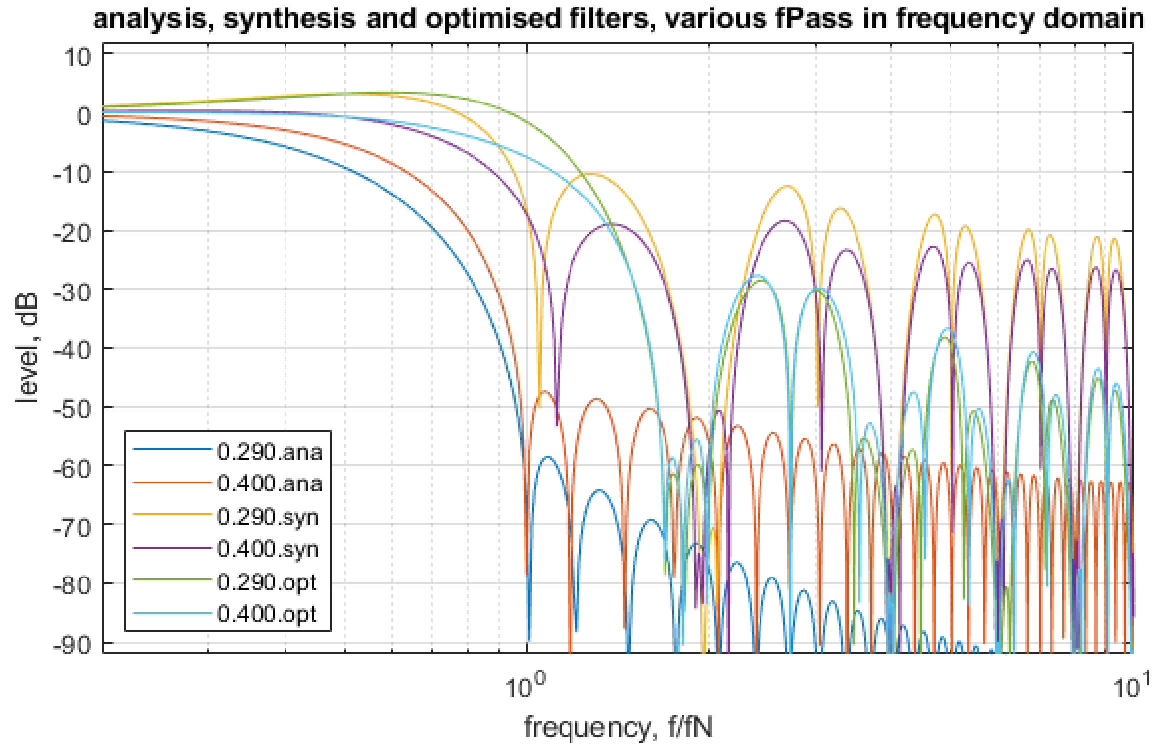
2.3.4 Minimise sidelobes [334]

We can use the existing degree of freedom to optimise the synthesis filter to have lower sidelobes.

The optimality criterion weights frequency response proportional to the *frequency.^sld* (Side Lobe Decay), thus enforcing continuity of the *(sld-1)*th time-domain derivative $\frac{d^k g(t)}{dt^k}$ where *k=sld-1*;

The selectable *(0: Fc/M)* part of frequency response is omitted allowing for wider mainlobe and passband overshoot because both are fine - in moderation.





The synthesis filters, corresponding to the analysis filters with *fPass* less than of the first spike are also easily optimizable. Others are not.

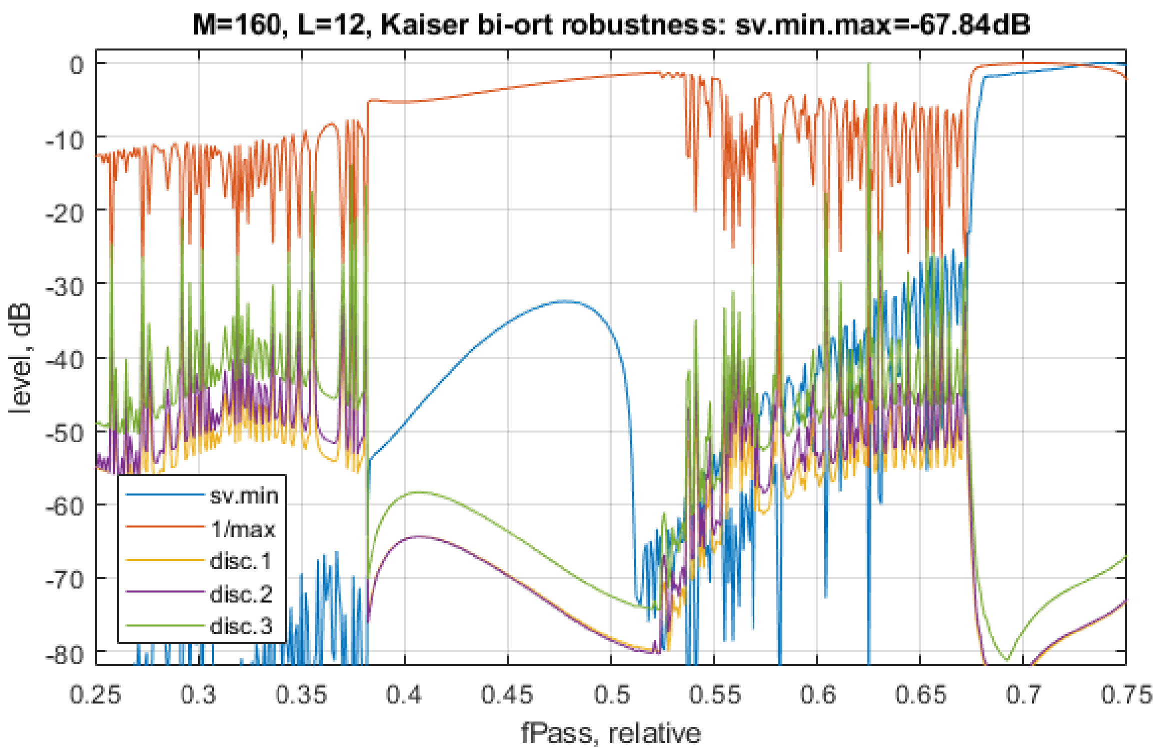
The synthesis filter does not need to have a null at f_N/M: why would it? The impact of being extra-wide could, potentially, become noticeable when radically different gain is applied to subbands, as for noise reduction or selective echo suppression; however, no one complained so far, in 15 years, over millions of "professionally" installed channels of a rebranded AEC. The band-edge effect also needs to accounted for, depending on the adaptive algorithm, if the synthesis filter is a really poor LP filter.

2.3.5 NPR to condition robustness [335]

Filterbanks based on low values of L, like L=6, suffer from limitations and deficiencies typical for short filters in general. Longer filters can be used to overcome those deficiencies but, as usual, they introduce new problems.

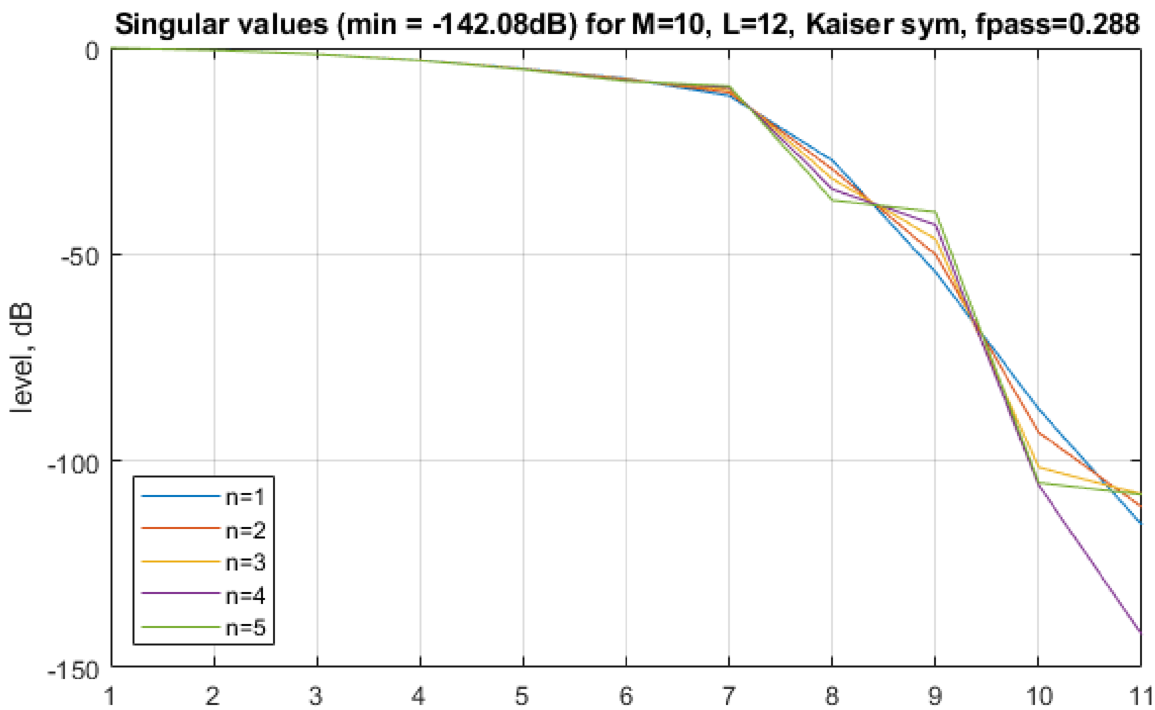
The spread of singular values for longer filters is much larger, 100+dB (like on the picture below). Partially, that happens because the values at the tails of the time response are leaning towards zero much closer. The first and the last equations of bi-orthogonality condition system contain only two last sub-samples of the *n*-th (of *1:M*) sub-sequence, and both are likely to be very small.

The spread of singular values becomes so high than we may need to regularize the system anyway. The synthesis filter will not be exactly bi-orthogonal, and the FB will become Near Perfect Reconstruction (NPR), and we will have to check for reconstruction errors explicitly.

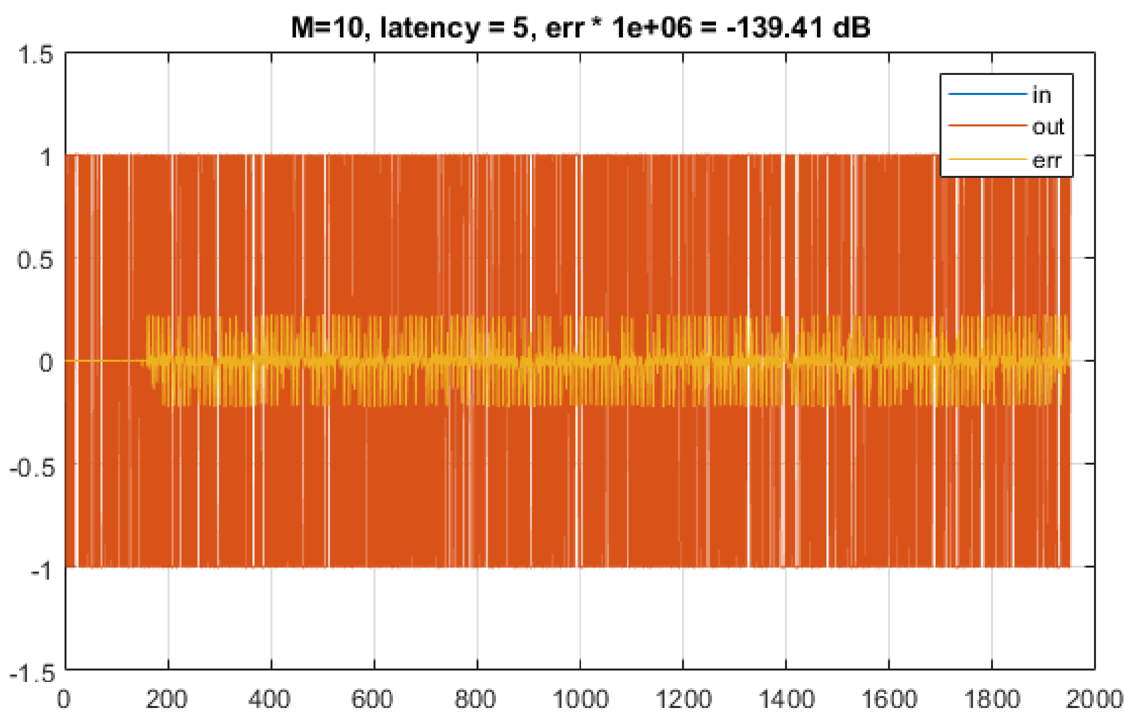


The *fsaf_mksyn.pr_test(…)* and *fsaf_mksyn.pr_plot(…)* help in understanding the reconstruction errors for a given pair of analysis / synthesis filters.

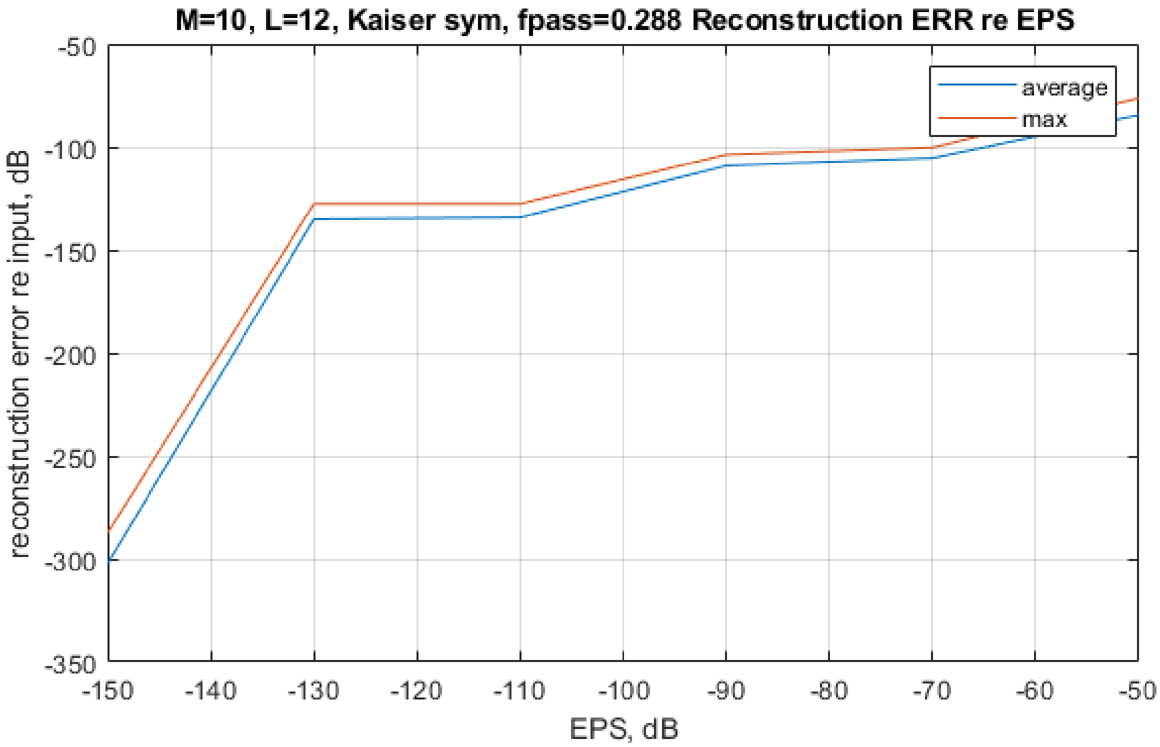
When we use regularization, we become less constrained in the choice of *fPass.* Here is a singular value domain of a spike case:



The reconstruction errors look like:



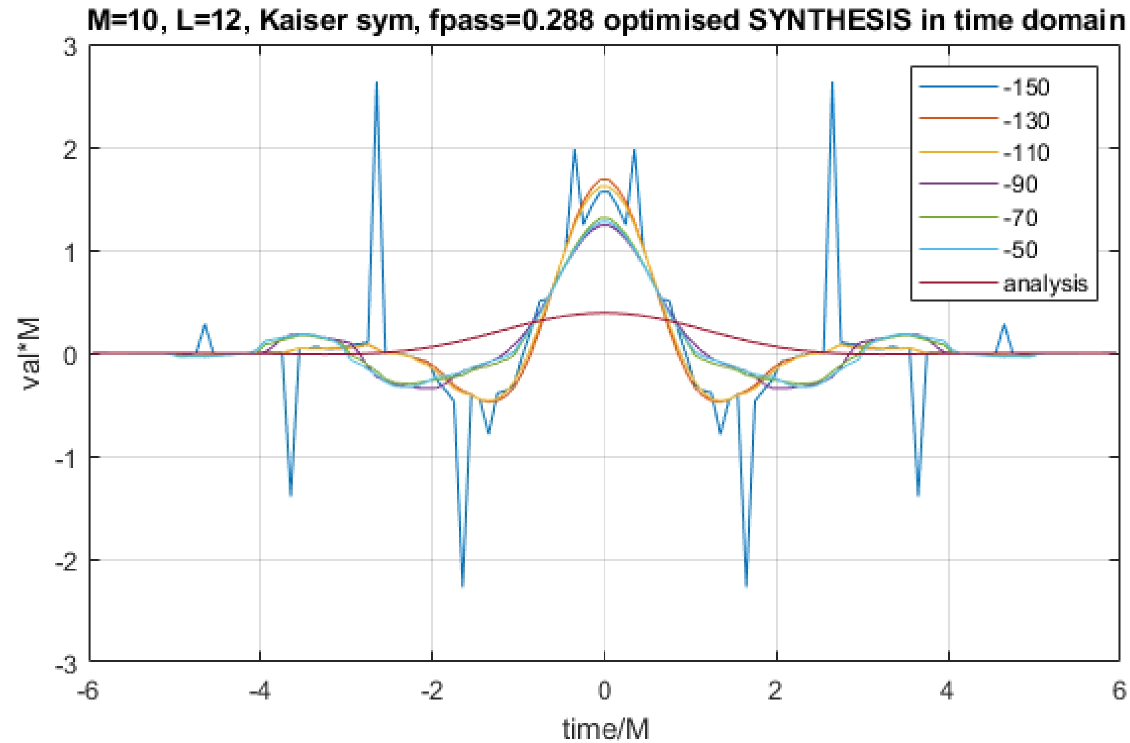
...and vary in level for different SVD thresholds:



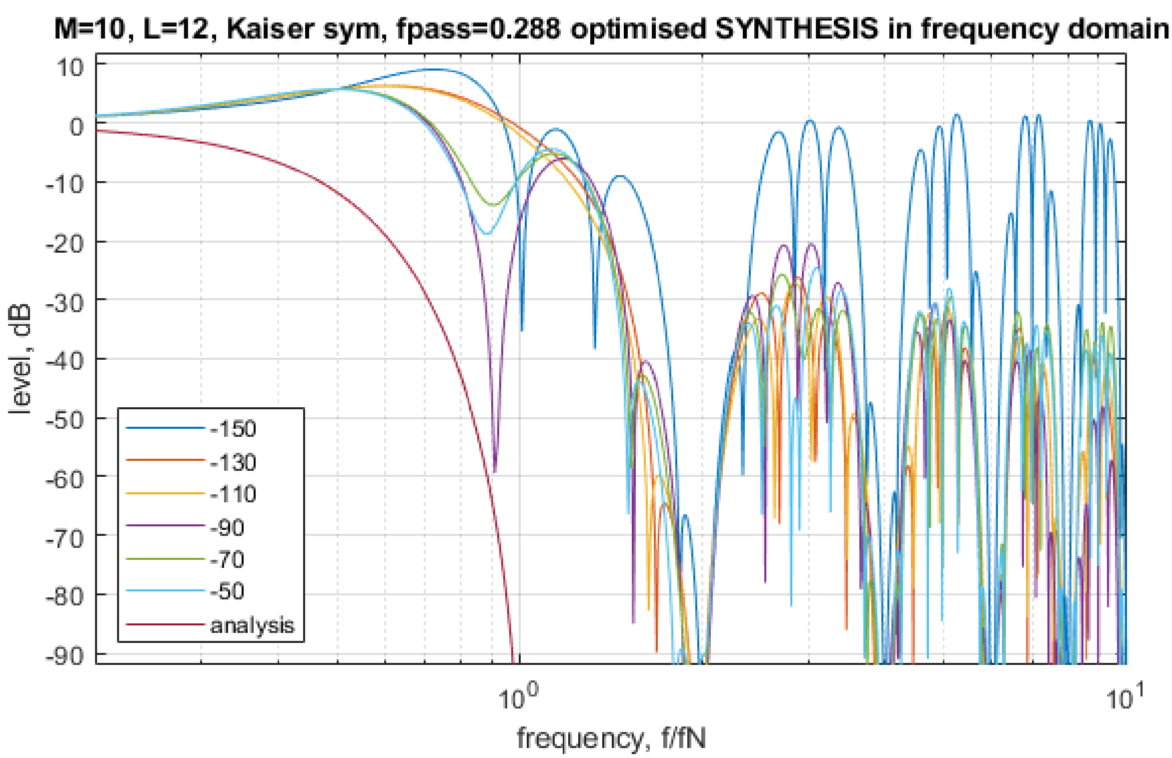
If the SVD threshold is less than the smallest singular value, the reconstruction errors are only due to limited numerical precision of about 50 bits (300/6.02 ~ 50) which corresponds to MATLAB's double precision calculations. For regular floating point with 23bit mantissa, lowering the achievable reconstruction error below -125 dB (which is also SNR of the best audio codecs currently) is untrivial. If so, then striving for theoretically true PR is of limited practical sense.

In this design, reconstruction error is less than -100dB for SVD threshold of -70dB (3.16e-4). Remember that 98dB corresponds to 16bit resolution, and perfect reconstruction in practical terms. Thus, enforcing SVD regularization does not destroy FSAF FB, if done properly and tested for. If we need to use even higher SVD's EPS, we need to make sure that synthesis filter decays sufficiently sharply (in frequency domain), to hide reconstruction errors under masking threshold.

The same case in time-domain, for different regularization thresholds:



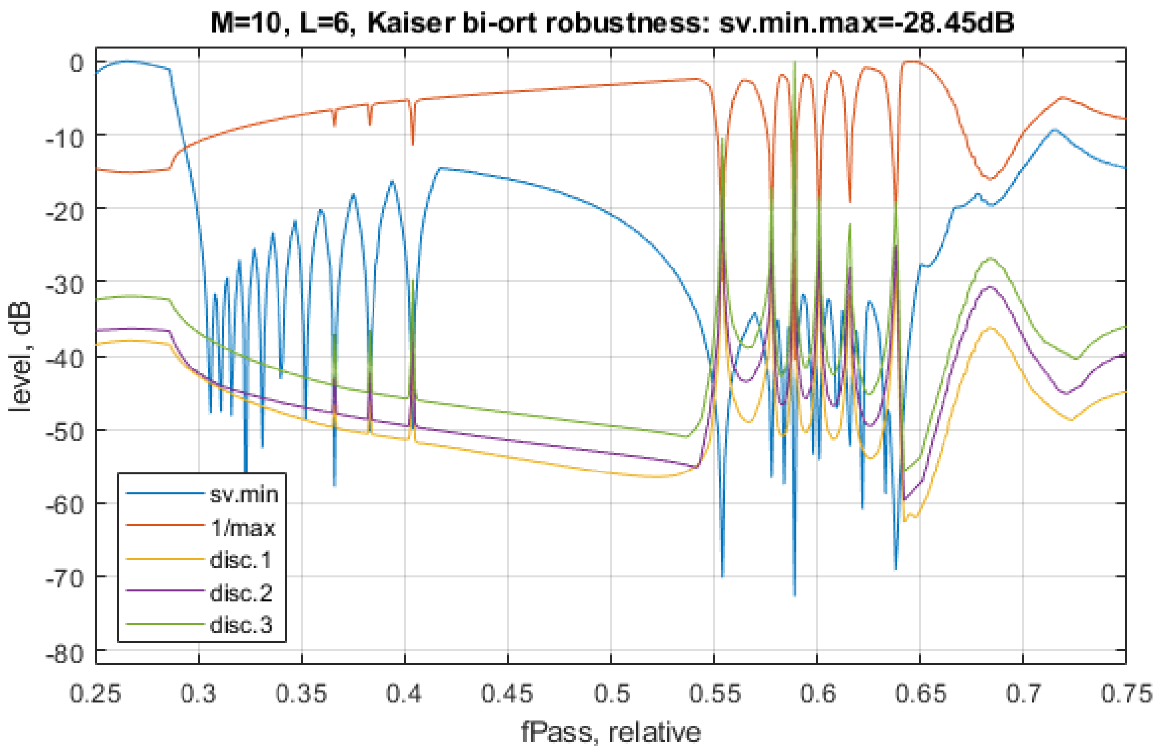
The corresponding frequency domain plot:



Besides much worse spectrum and must-have regularization, the longer analysis / synthesis filters introduce another problem: longer processing latency. To minimise the processing latency of FSAF FB, we can convert the analysis filter minimal phase form, and then find the corresponding synthesis filter for a Lower filterbank Delay (LD).

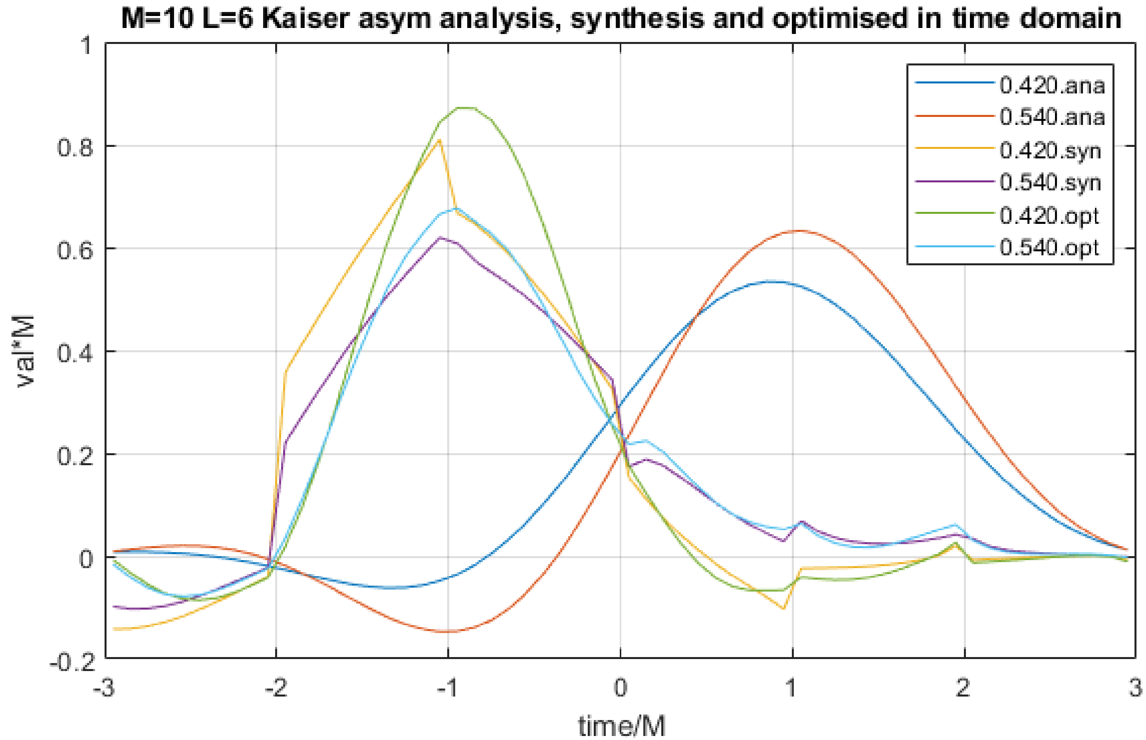
2.3.6 LD versions of Short FB [336]

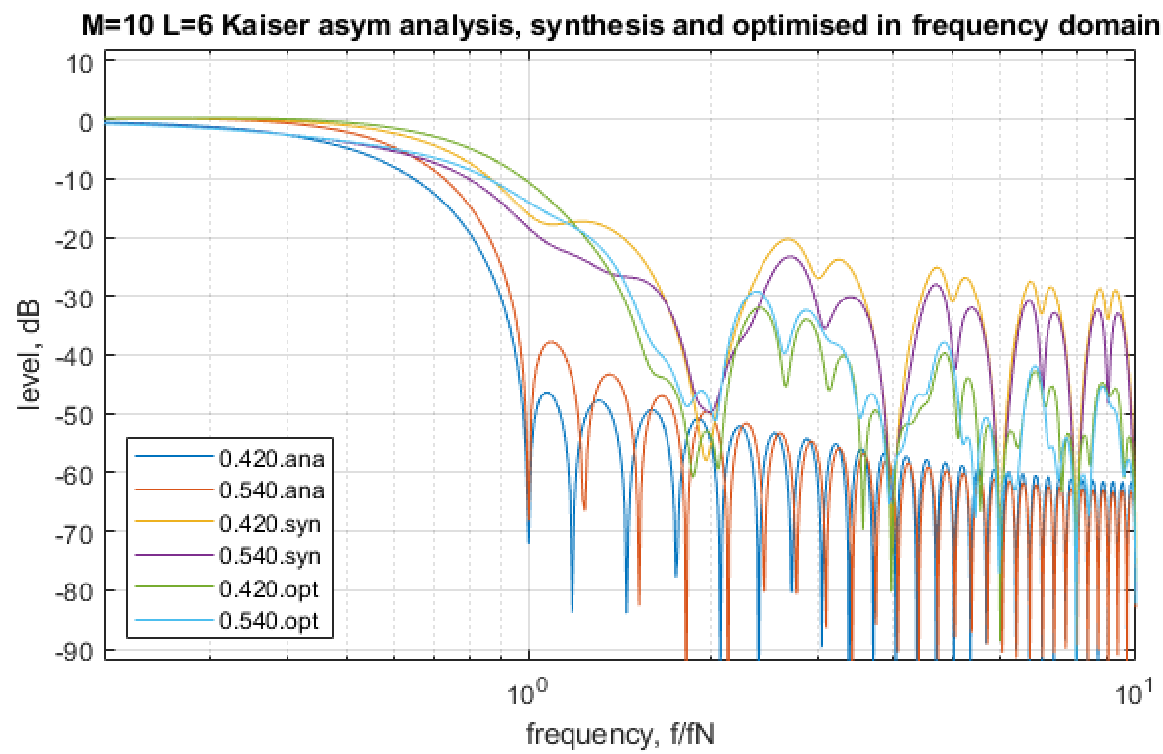
The old-fashioned method is to use *polystab()* MATLAB function, which has its quirks, and, in particular, it is limited by the number of zeros it can resolve (which are very tight here). For low M and L values, like {M=10; L=6}, it happily works. The robustness diagram shows that there are three stable regions: [0.0…0.29], [0.42…0.54] and [0.65…0.75] where regularization and NPR are not required:



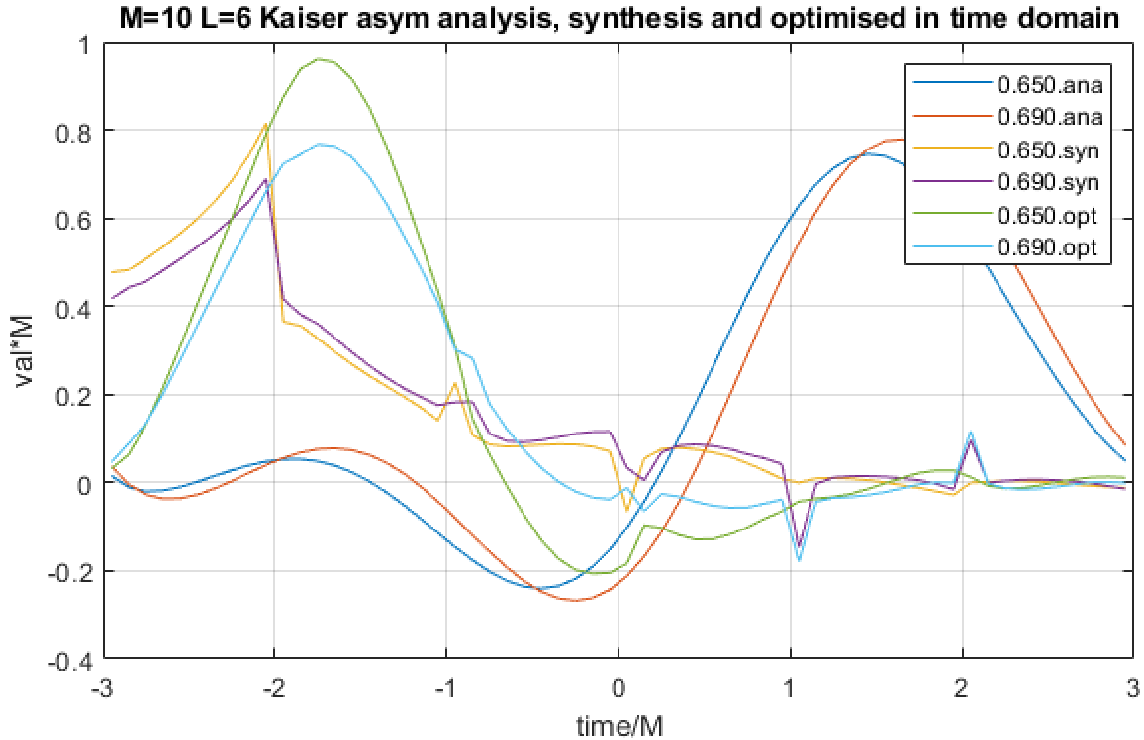
The first region [0.0…0.29] corresponds to essentially symmetrical case (latency of 5 frames of M samples), when entire function is above zero.

The middle region [0.42…0.54] is well suited for operating with algorithmic latency of 3 frames of M samples, when “1” is in the second row of “b” vector.

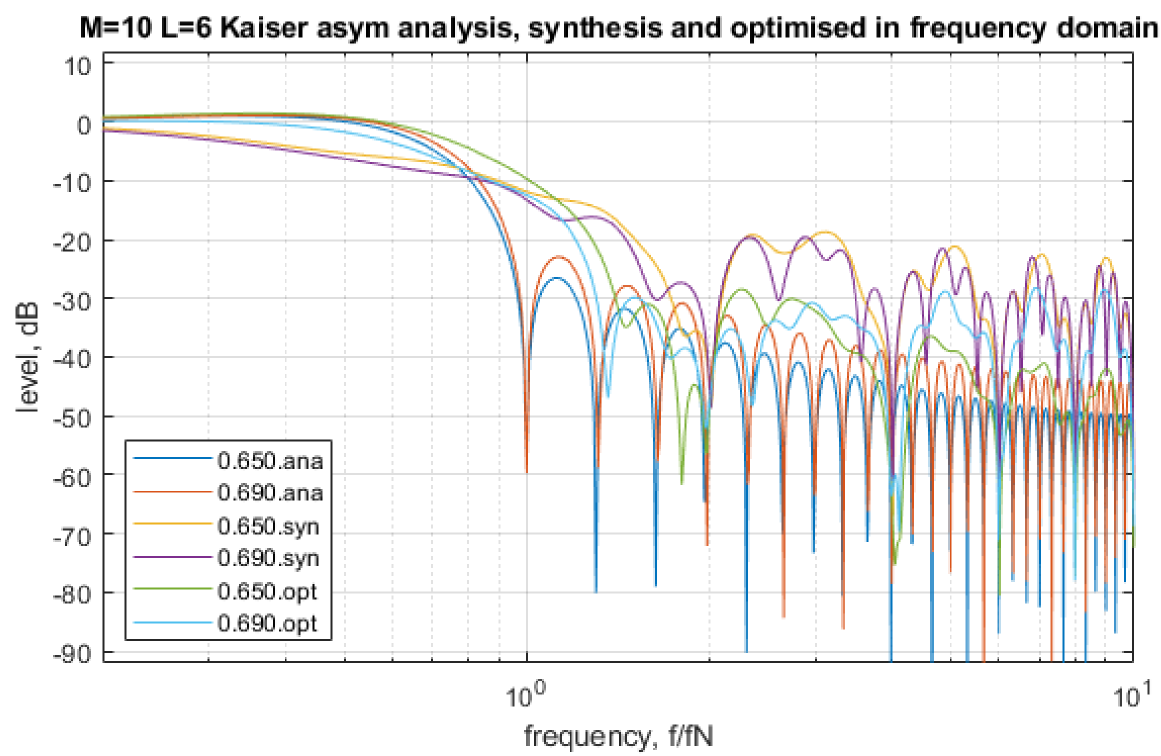




The last region [0.65…0.75] is well suited for operating with algorithmic latency of 1 frame of M samples, when “1” is in the 1st row of “b” vector.



Obviously, the case with delay = 1 frame requires a completely different analysis filter, with sharper upfront – and higher, slower decaying sidelobes, consecutively. The lower is the delay of PR filterbank, the higher sidelobes must be, which prevents good convergence of per-subband adaptive algorithm (to be demonstrated later).



2.3.7 LD versions of Long FB [337]

We don't have to be constrained to short filters, L=6 or so. We can use any feasible L, as far as asymmetric filterbank's overall latency is acceptable. MIPS will be essentially the same as for Short FB if we use Weighted Over-Lap Add (WOLA) operations. This way, we can easily construct very nice analysis filters with low sidelobes (and we will see later why we need them).

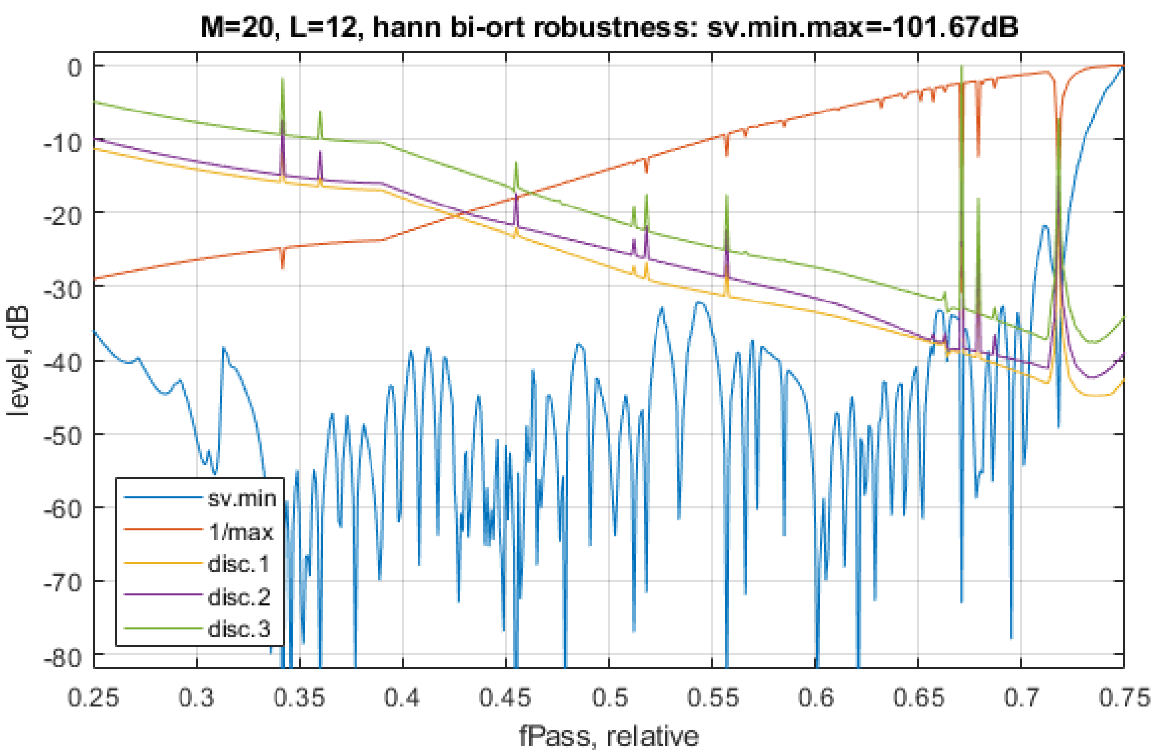
The old-fashioned method to use *polystab()* MATLAB function may give unpredictable and unstable results if the length of analysis filter, L*M, is > 64 (depending on the version of MATLAB) either due to large L or M or both. There is a *minphase()* function in MatLab but I could not make it work reliably. We can overcome this deficiency by

- down-sampling the symmetric analysis prototype,
- converting it to minimal delay,
- up-sampling it back.

The `function fmin = minphase_make(o,fir,Nroots,ButOrder)` performs this sequence. The `Nroots` shall be < 64, generally, = M*L/an_integer. The aliased components after up-sample can be suppressed by Butterworth filter of order `ButOrder`. The impact of resampling on singular value distribution shall not be understated.

Alternatively, we can start with an (already minimal phase) windowed IIR filter response.

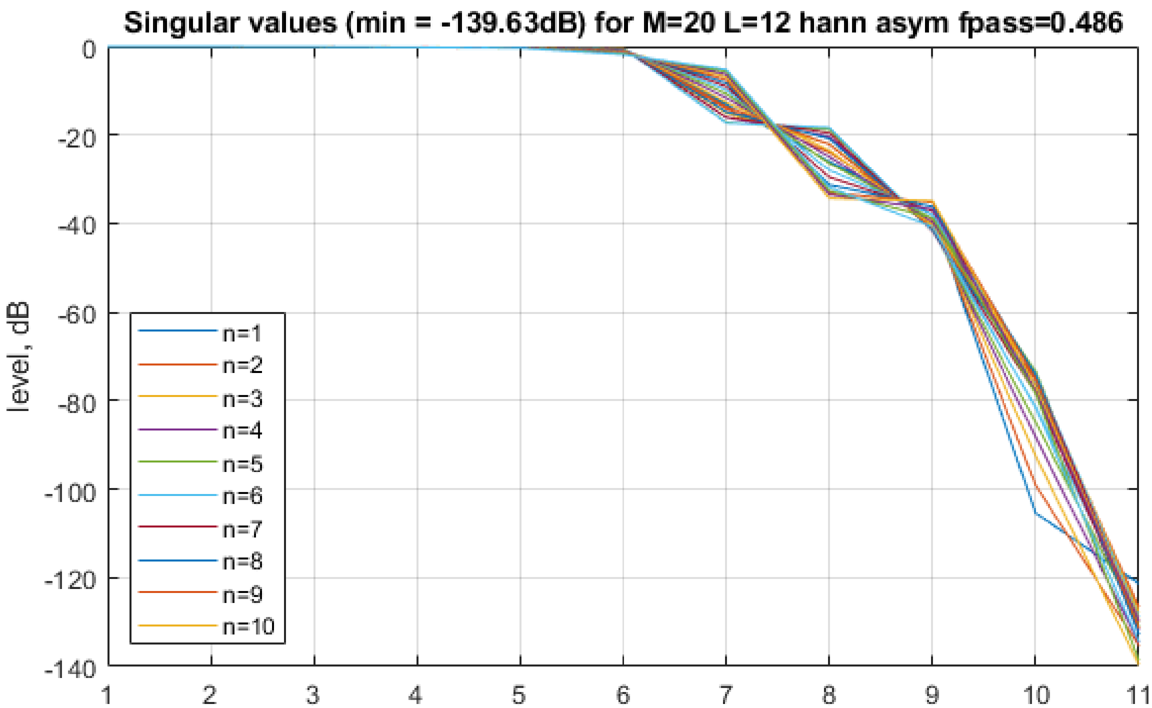
First, we need to check for unfortunate lack of robustness by examining singular values of the bi-orthogonal conditioning system of linear equation for a minimal-norm synthesis filter, and conclude that regularization is a must.



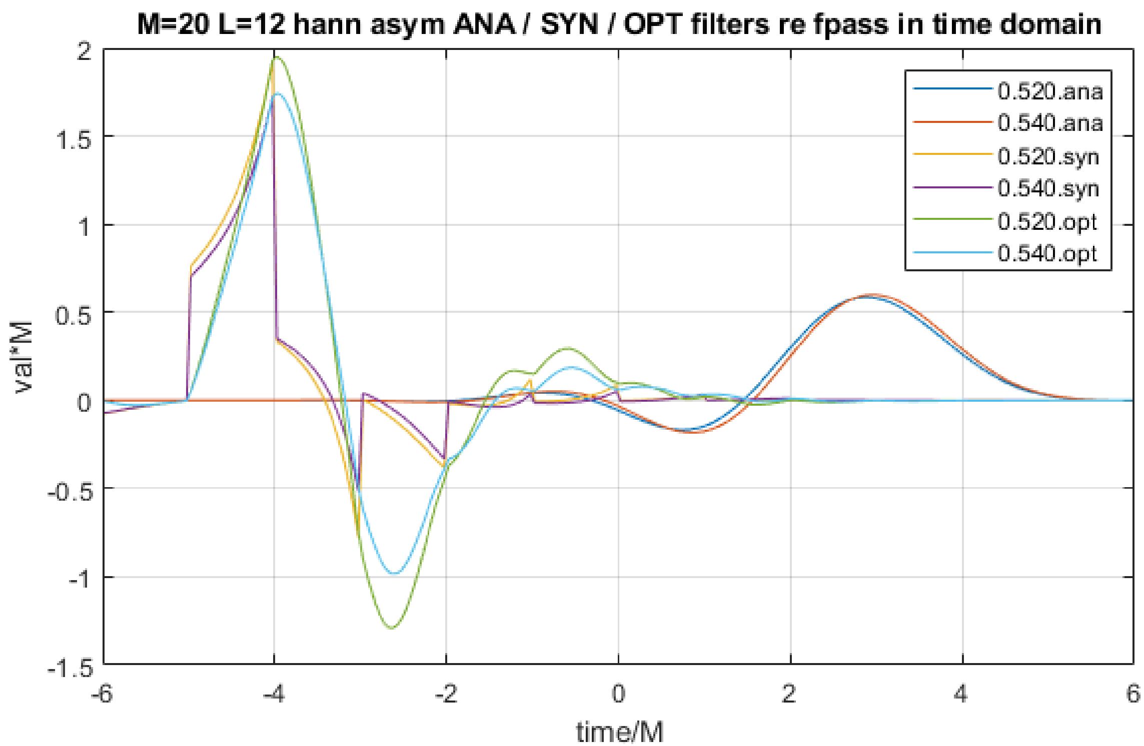
Large spread of eigen / singular values is a dark side of having a good LP analysis filter. It grows exponentially with L, and it's unavoidable:

- A good LP filter must have very small discontinuities in the function and derivatives;

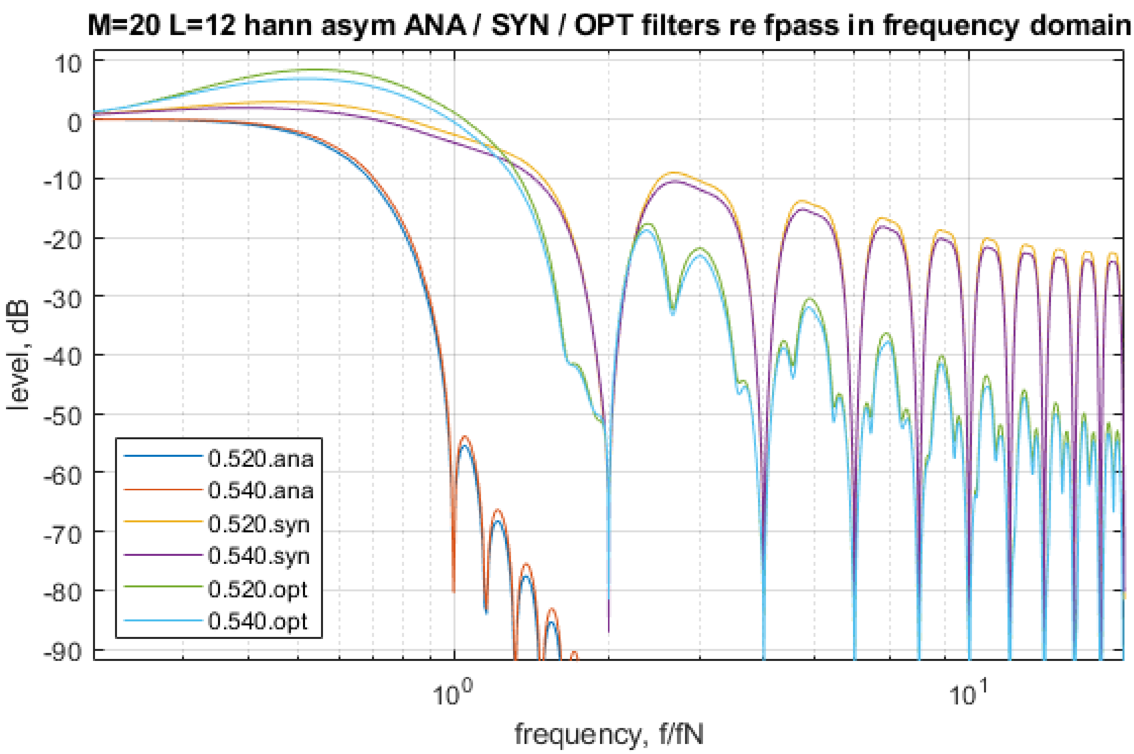
- Therefore, the tails of a good LP filter are almost zero, proportionally to the sidelobe suppression and the spectral decay;
- Therefore, the bi-orthogonality system of equation has the first and last rows containing values from the very tail of LP analysis filter;
- Therefore, the spectrum likely decays as fast (or faster) as the time-domain sidelobes of the analysis filter



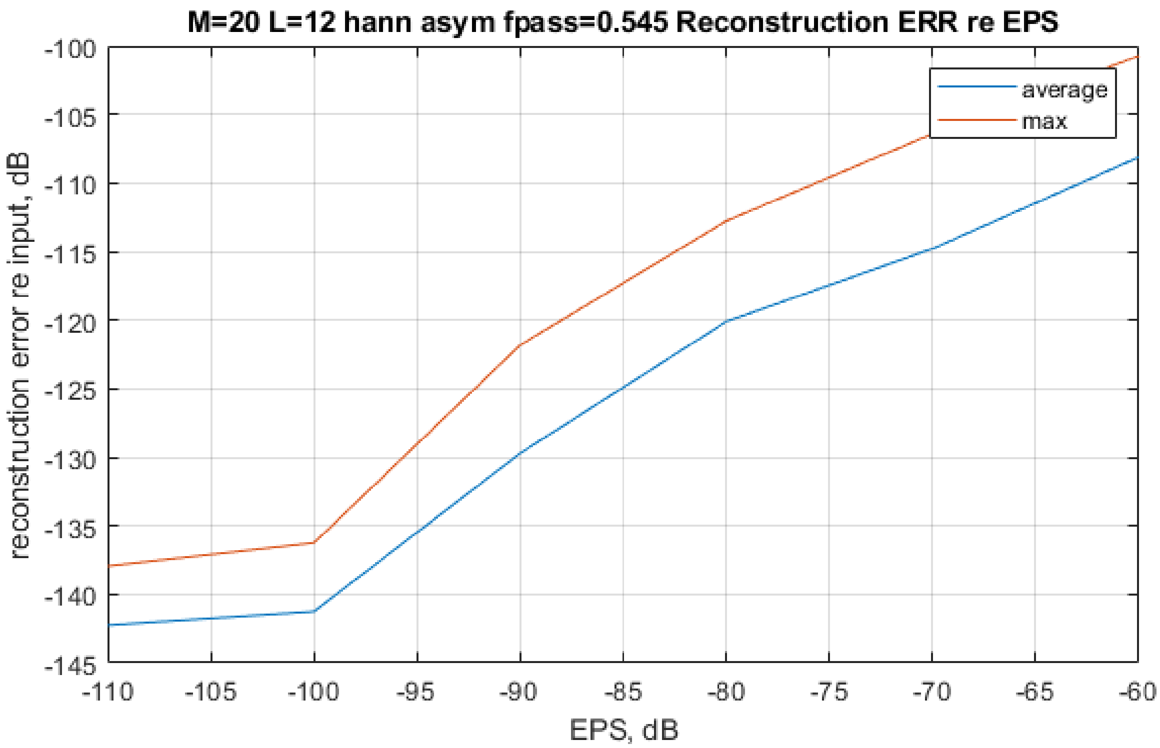
Then we choose an appropriate SVD threshold and find much nicer (relative to L=6) analysis & synthesis filters for algorithmic delay of 3 frames of M samples. Analysis, synthesis and optimised synthesis filters in time domain:



… and in frequency domain:



… and verify that reconstruction errors well are under control for reasonable SVD threshold / EPS:



The longer L filters give a lot of freedom in the FB design. We need to note that it is not problematic to design an exchange algorithm for much smoother synthesis filters – but it is simply unnecessary.

2.3.8 JTF-LS for solving singular systems of linear equations [313]

The described approach involves optimization routines and, in general, Optimization Toolbox. The internals of the fminsearch(.) and fminunc(.) are out of user control and may differ for different versions of MATLAB. We formulate a criterion for optimization (weighted norm of frequency response' sidelobes) and provide a first approximation. We trust optimization algorithms but we don't know what it will converge to. It's hard to figure out what we shall change, and how, to cause the result to move in a desired direction. Optimization time and resources grow with M*L quite sharply.

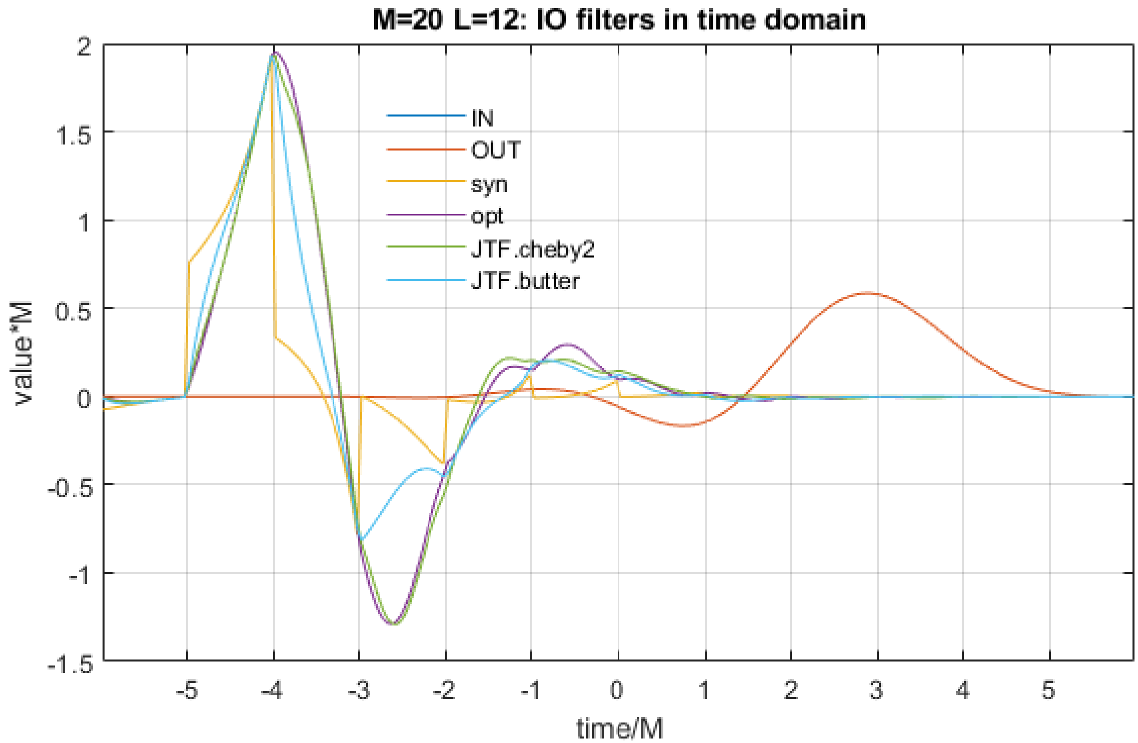
The ReLS initialization approach from Part II shall work just fine for solving our underdetermined / singular system of linear equations:

$$g = (A^TA + \sigma^2 D_0^{-1})^{-1}A^Tb; \text{ for } \sigma^2\text{->}0$$

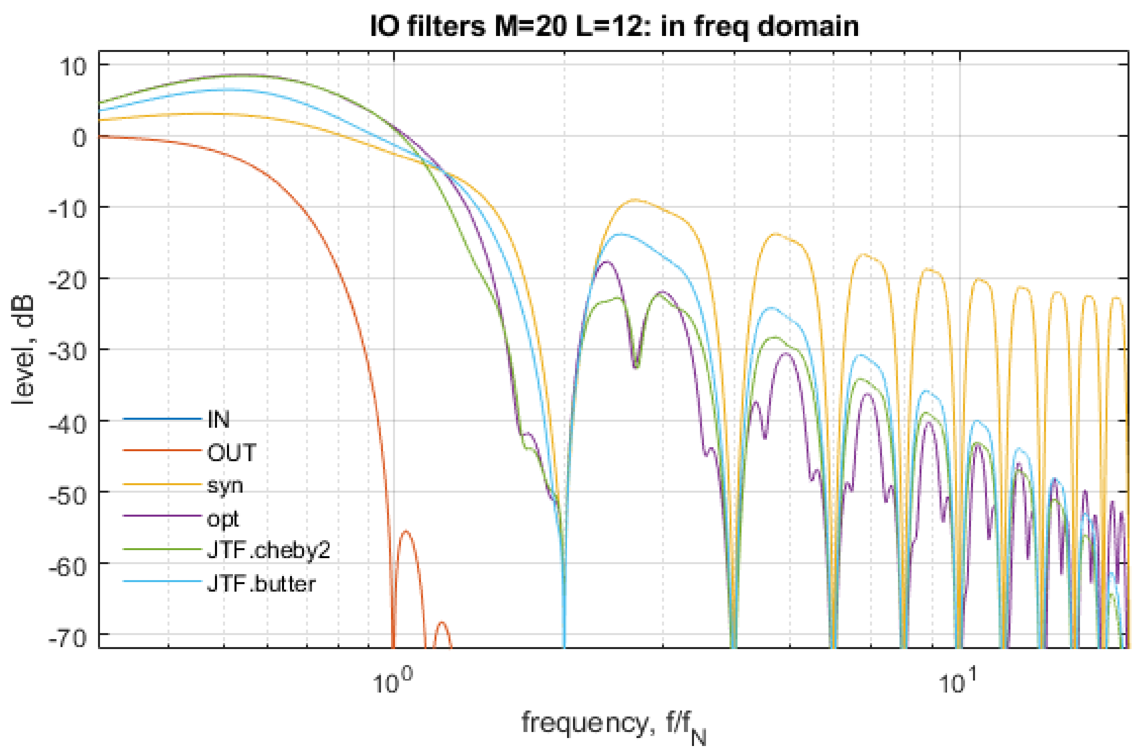
Instead of formulating an optimization criterion, we can provide a simple meaningful D_0^{-1} using Joint Time Frequency initialization, and improve it iteratively based on *g* (see *f2w()* function). JTF is a tool to formalize our (informal) knowledge about desired properties of the solution which makes a bit more sense than indirectly formulating an optimization criterion, and takes much less time and resources.

Below is a comparison of "old" vs "new" approach, on the [discussed above] example of LD filterbank with M=20, L=12, algorithmic latency=3*M, *fir1(M*L, 0.52/M, hann(M*L))* based, NPR with distortions -110dB. JTF-LS controls the desired frequency response of synthesis filterbank by *cheby2(3,20,1.5/M);*

The major difference, relative to the "old" approach, is that the adjusting parameters of D_0^{-1} became quite straight-forward: changing *Rs* and *Wn* in *cheby2(n, Rs, Wn)* makes sidelobes of JTF-conditioned synthesis filter go up or down, start earlier or later - as expected. Indeed, σ^2 mostly impacts FB reconstruction error without affecting the synthesis filter shape much. That's what is should be by theory but the details how it happens are not clear yet.



JTF-LS.cheby2(…) is intentionally made similar to the old solution, to show that what was done a hard way – can be done easier. The *cheby2(…)* is not "the only way", there are other filters that accomplish the task with similar (or dissimilar, like *butter(2,0.9/M)* here) results.



The JTF-LS iterative initializing appears to be on par with SVD threshold conditioning[1] – which is in line with other ReLS findings.

2.4 SUMMARY [303]

While designing [Nearly] Perfect Reconstruction analysis / synthesis filterbank for Subband Adaptive Filtering, we'd be better off keeping in mind that the subband adaptive filter does not "know" anything about PR and aliasing cancelation.

Subband adaptive filter has to deal with aliasing noise as regular noise. Say, near sidelobes level is -35dB, and the level of signal in the next subband is 20dB higher than in this subband. That results in SNR of only 15dB, which means that the adaptive filter either can not converge well enough, or it will diverge, or it will perceive this noise as RIR change, or as double talk – and any of these scenarios is not good.

Short and Wide FB (L=6, M=8…20) does not provide enough MIPS savings to become practical. However, it's an excellent starting point in FSAF design.

Short and Narrow FB (L=6, M>100) do not work well for adaptive filtering. If the M-normalised analysis filter is short, the sidelobes are high and decay relatively slowly. Then each subband will accumulate lots of aliasing noise (from other parts of the spectrum), and … see next chapter for details.

LD Long and Narrow FB (L>10, M>100) is often the best case for real-world applications. Let's assume that RT_{60} is less than 500ms, and we can get away with echo tail about 300ms (). For 16kHz sampling, 320 sub-bands of 50Hz wide, the subband representation of RIR takes about 16 samples. If so, we can use WRLS because both memory (~1/M) and MIPS (~$1/M^2$) requirements decline sharply with M.

However, all that is nice… if we can afford long algorithmic latency over analysis-synthesis filterbank. With latency of 3 frames, we end up with minimum 60ms delay. We may have to drop to 160 sub-bands of 100Hz wide. We'll need at least 30 taps of complex sub-band adaptive filter which may be prohibitively high number for applying WRLS. There is a cardinal solution by open-loop delayless (OLD) implementation but you have to read the entire document.

However, the demonstrations to follow are concentrated on the examples of LD Long and Wide FB (L>10, M=10). The reason is simplicity. Technically, the process of conversion from low to high M is tedious but conceptually simple. We need to start with a low M, like 8 or 10. If the design does not work for low M, something is wrong. Then, we slowly and carefully up-sample both the asymmetric analysis and the synthesis filters, adjusting SVD's thresholds appropriately, to be used as the starting point for optimisation in bi-orthogonality, and controlling the result at each iteration (which may easily take hours of computations). A structural example code can be found at the end of doc_p303.m.

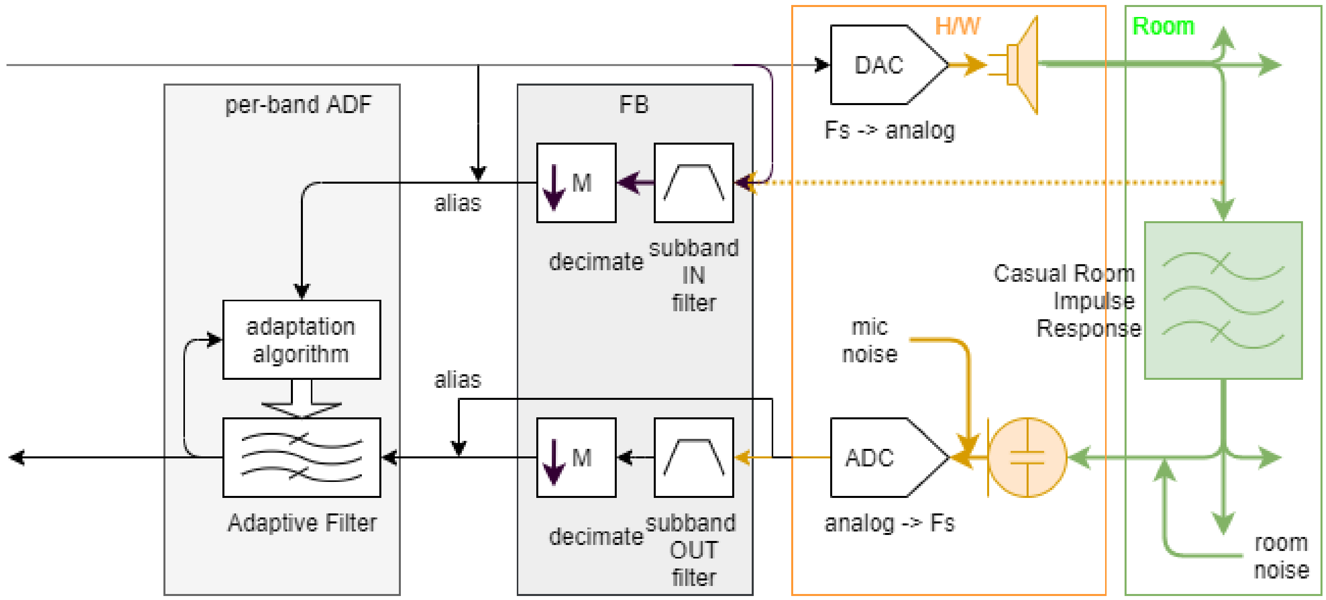
3 OUT/IN CO-DESIGN

3.1 BASICS

Let's forget for a moment about bi-orthogonality condition, singular value spectral spread, reconstruction error, etc, and consider designing IN and OUT prototype filters "alone", for the sake of proper adaptive filter functioning. *IN(t)* and *OUT(t)* do not have to be of the same length, nor belong to the same "family", nor strictly symmetric or asymmetric.

---

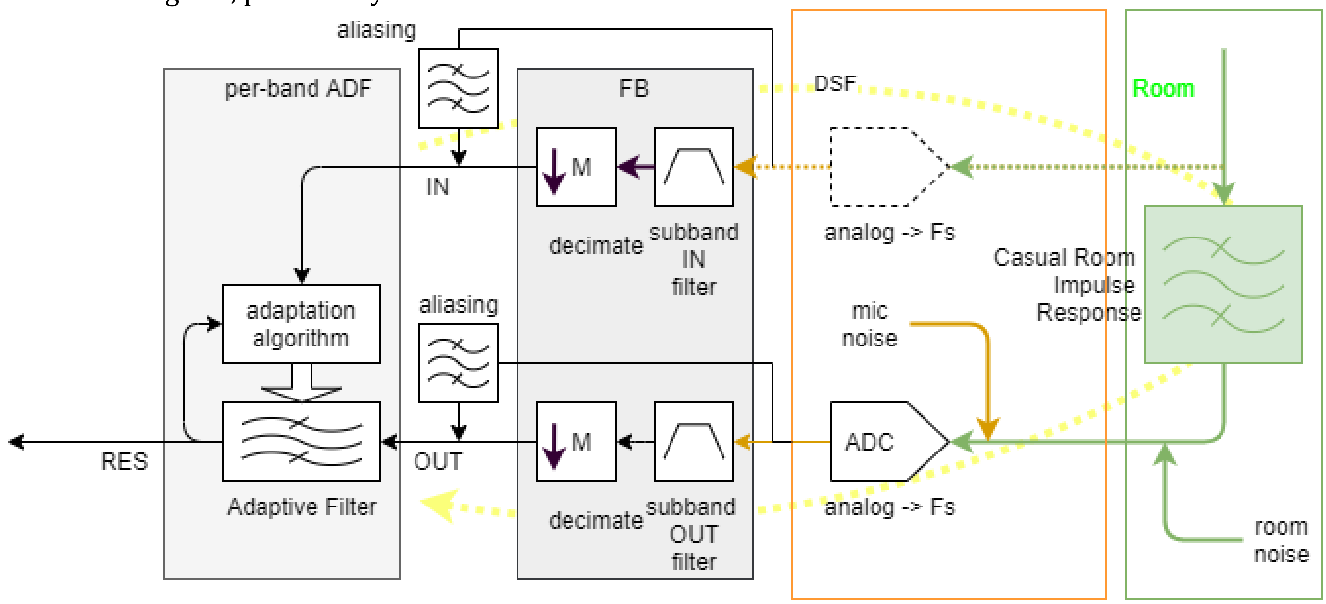
[1] We can apply SVD thresholding on the top of ReLS – but the results are neither predictable nor intuitive.



The adaptive filter in each subband "does not know" about perfect reconstruction, nor about existence of synthesis filterbank. The key consideration is that the adaptive filter "sees" a shadow of

{decimate^{-1} * (IN filter)$^{-1}$ * DAC * (continuous time RIR) * ADC * (Out filter) * (decimate)}

… <yellow> chain from <IN> to <OUT>, projected onto the subband cave's wall as observable subsampled IN and OUT signals, polluted by various noises and distortions:



Now we shall consider the difficulties in recovering of the ideal <yellow> chain from incomplete observation, which we have to keep in mind while designing FSAF.

3.2 Aliasing

The IN and OUT subband signals have all other frequencies aliased onto 'main' passband in each subband.

Let's consider the excitation on white, frequency uniform, noise, and the 1st DC subband. It's not worse for other subbands. Any excessive level of signal above flat in stopband area will contribute to correspondingly excessive aliasing, accounting for the stopband attenuation.

The OUT signal aliases itself as if the frequency response has been folded, by reflecting the remnants of the spectrum until exhausted.

[diagram TBD]

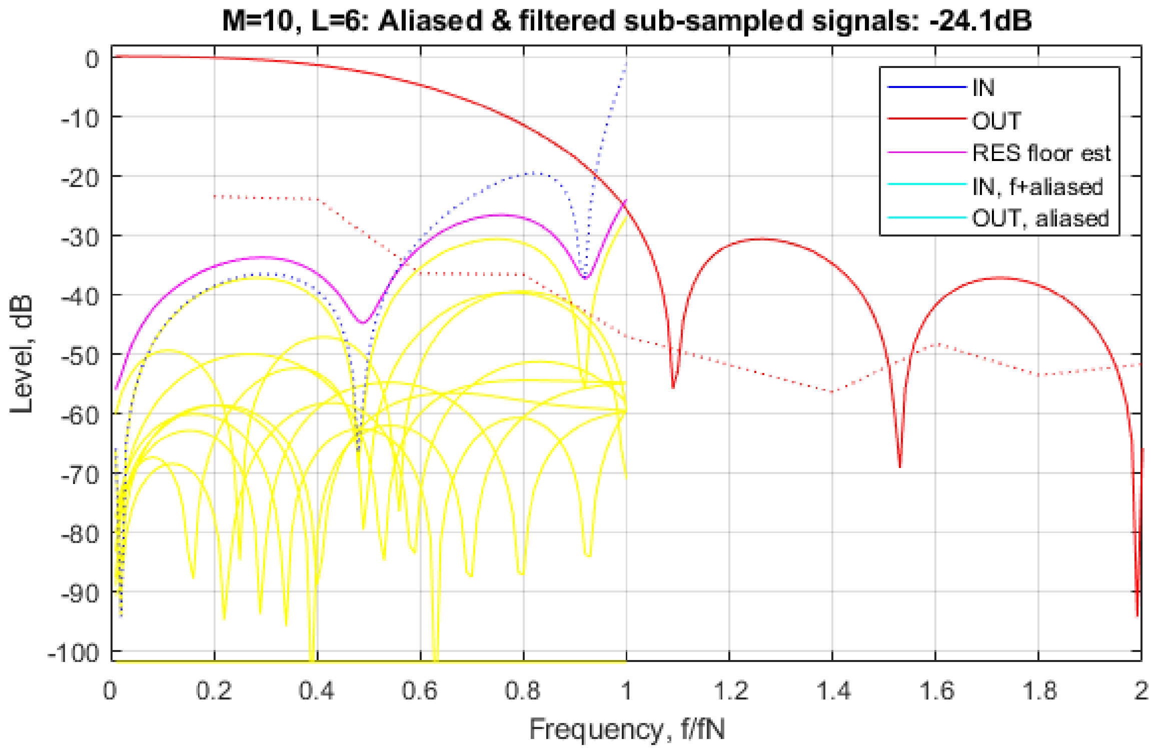
The IN signal is aliased in the same way but the impact of IN aliasing onto residual spectrum should be calculated by adding *OUT(f)/IN(f)*. On a frequency f < 1, we observe *db(IN(1+(1-f))) – db(IN(f)) + db(OUT(f));* Thus, that limits how wide we can make IN filter, relative to OUT. We can afford quite a bit of IN aliasing (except for non/under-regularised RLS algorithms, see later) as far as it does not exceed OUT aliasing too badly. Ideal IN filter is a brick-wall with passband being a Wiener filter against aliased stop-band.

The convergence of adaptive algorithms can be fast if and only if the residual error signal is above noise. When the error goes below noise, convergence becomes $\sim {}^{1}/_{\sqrt{t}}$, and we need doubling of observation time for each 3dB of additional convergence (in the best case). Even if there is no AWGN on the OUT, there will be noise on the RES – the under-modelled and under-cancelled IN signal, on the level of IN aliasing, shaped by the synthesis filterbank. Thus, the IN/OUT aliasing imposes a practical limit on the FSAF convergence, and we have to use a Long LD FB, as the previous chapter advised.

The figures below assume white noise as excitation, with around 0dB of mean RIR(f). The figures below are 'theoretical', modelling will come later.

The dotted red line shows the amount of Inter-Band Leakage (IBL), which is aliasing from near-by subbands, the first dot for total, and the rest in the order of distance from a subband. The farther a pair of subbands is, the less they should affect each other.

3.2.1 Aliasing for SAF1988 [304]



The middle of subband is relatively ok, but the band edge is quite poor. The inter-band leakage is high: only ~50 dB for far-away subbands.

3.2.2 Aliasing for Short FB [304]

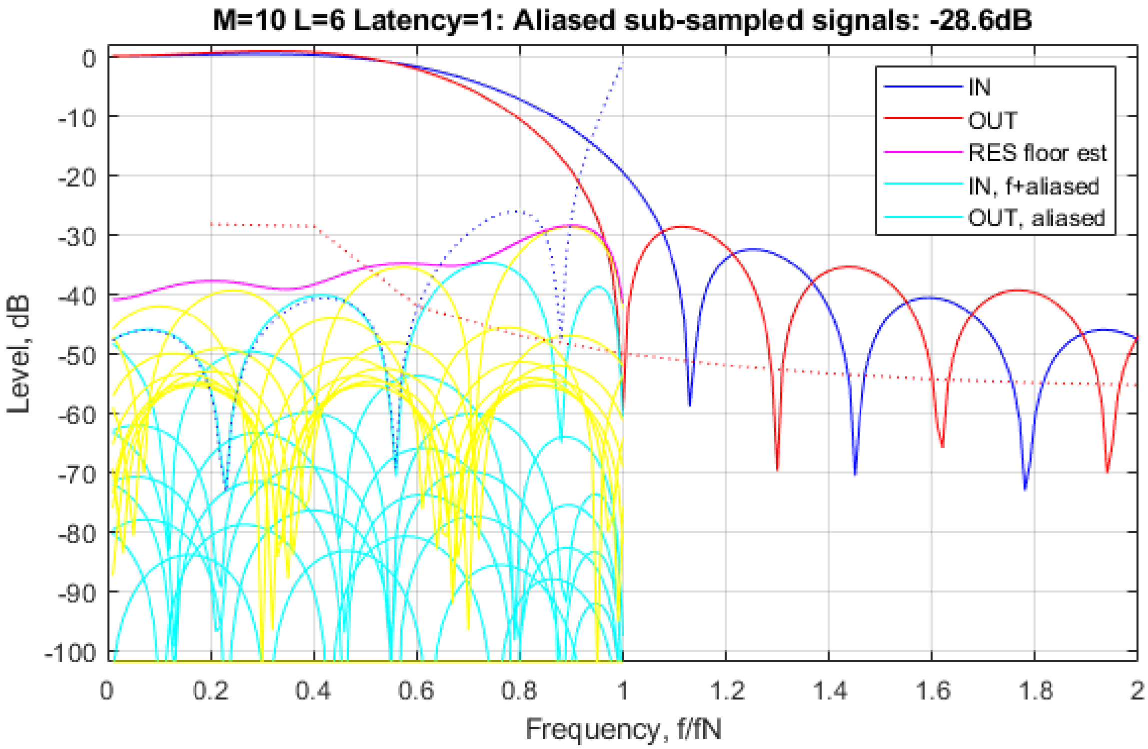
The level of aliasing is the lowest for symmetric filters with low *fPass* and, correspondingly, sidelobes.

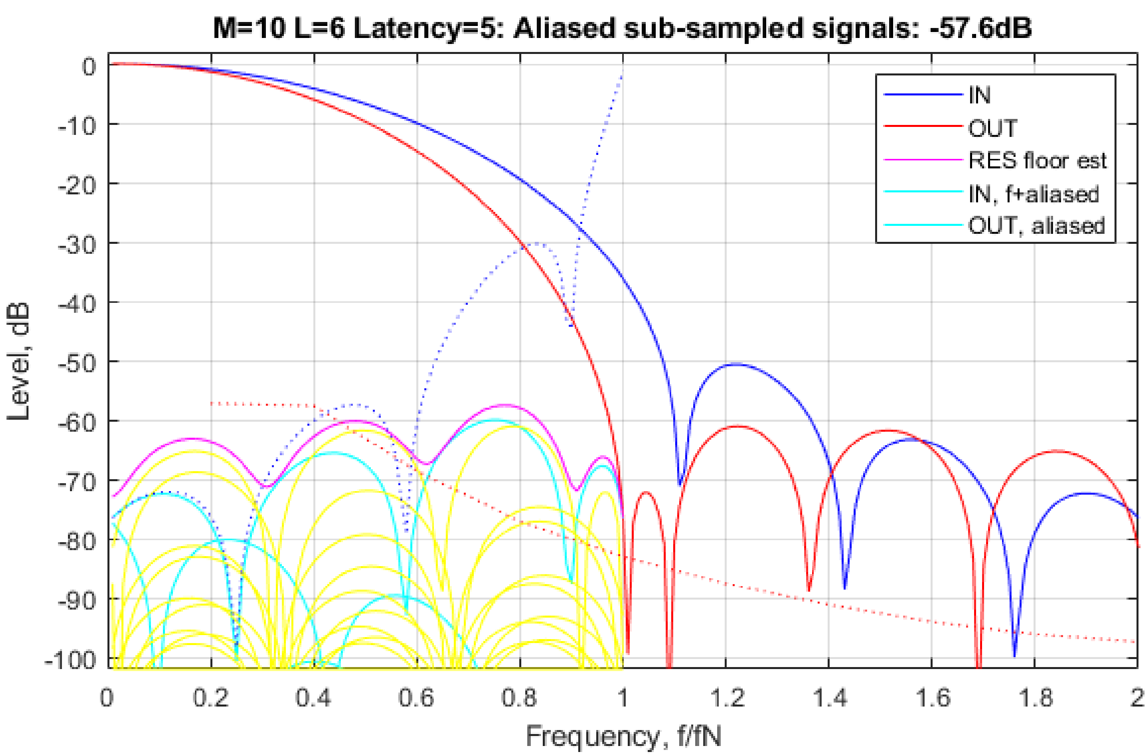
If the *OUT(f)* filter is chosen to be as narrow as possible (but still with reasonable synthesis filter), the aliasing drops to about -60dB, uniformly across subband, and far IBL go down to ~ -90…-100dB.

If *OUT(f)* filter is chosen to be QMF-ish as above, then the amount of aliasing is around -45-ish in the middle of a subband. Aliasing may mask excessive residual echo due to RIR change and interfere with convergence. Far IBL worsens to about -70dB.

The LD versions, for both Short and Long FB, pose a significant challenge: it's not so easy to create even steeper "brick-wall" LP filter with sharp transition and low sidelobes. Historically, IN filter has been chosen of a double length, relative to *OUT(t)/RES(t)* filters, for most applications.

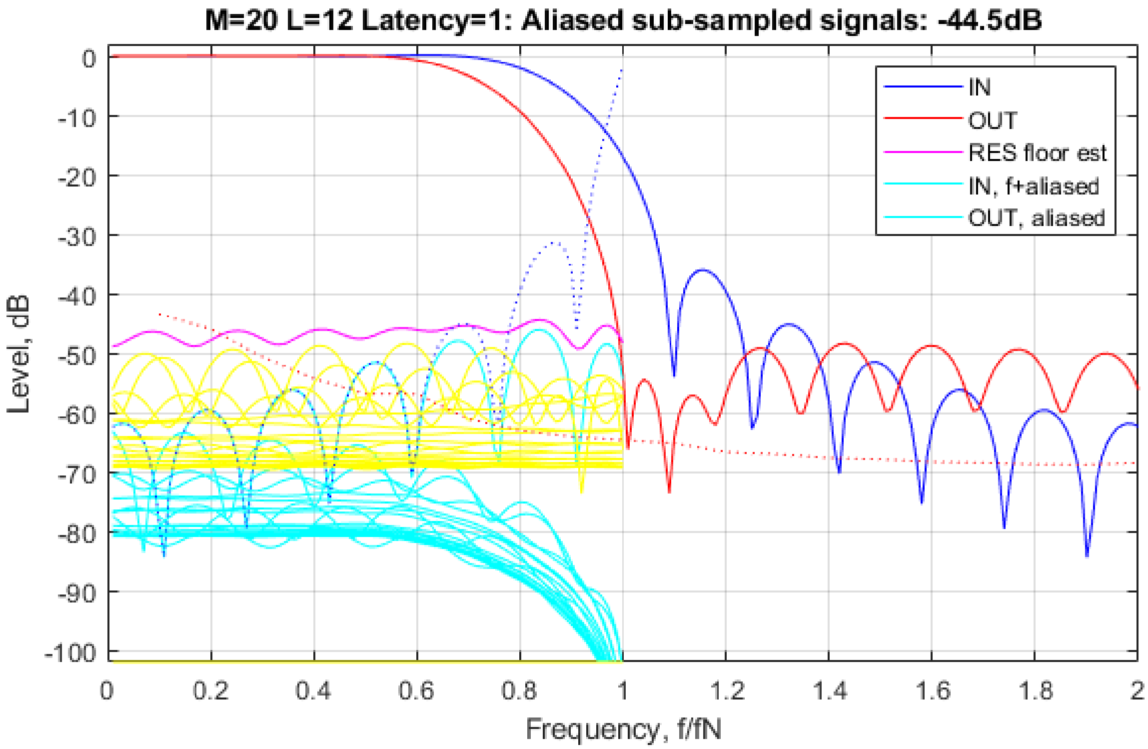
Let's provide here a couple of examples, for an asymmetric design with the shortest latency of 1 frame of M samples, and a "standard" symmetric design:

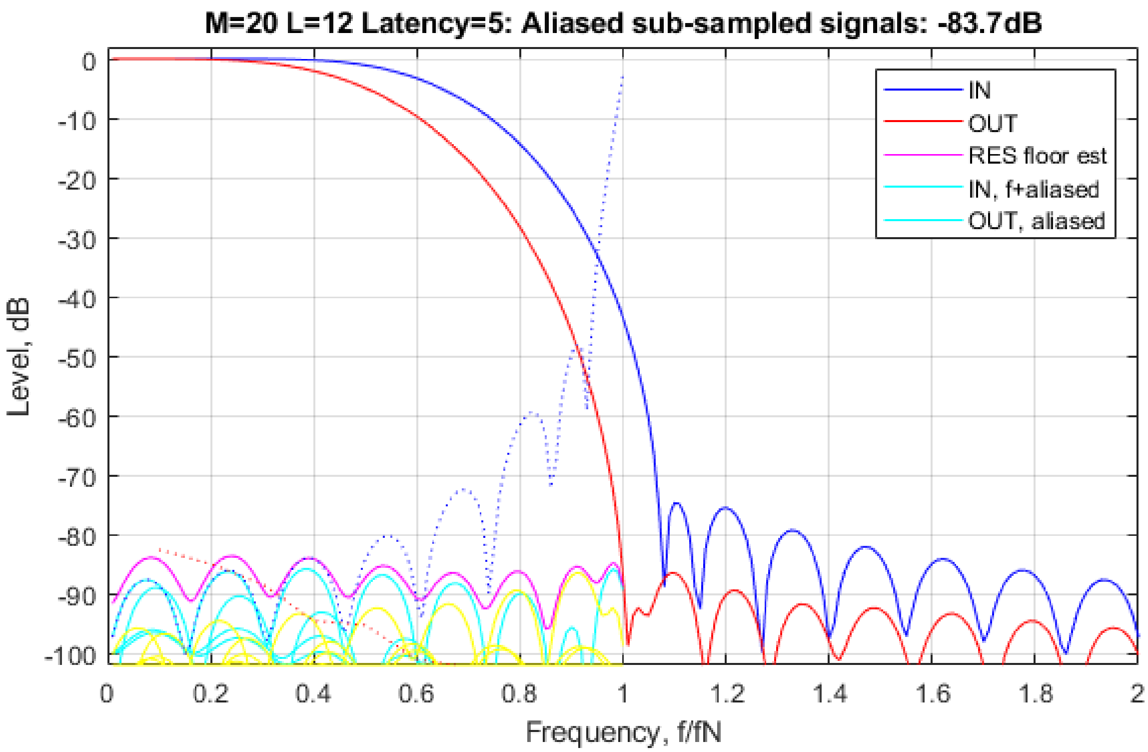




3.2.3 Aliasing for Long FB [304]

The longer filters allow to decrease the “noise floor”, due to sidelobes and aliasing, quite significantly - relative to the short L=6 filters for the same frame length M and same latency.





This is one of many factors of why Long LD FB are most practical.

3.3 Delta-function Spread Function (DSF)

Understanding of Delta-function Spread Function (DSF) is central to Subband Adaptive processing. It is a formalised representation of the <yellow> signal chain from the 3.1 Basics chapter.

[F]SAF shall be seen as observable, and therefore, identifiable multirate signal processing "black-box":

- A delta-function response in full-band is somehow translated into a spread in sub-band sub-sampled time domain.
- This spread is infinite (because it is finite frequency-wise), and goes into both causal and non-casual directions
- The shape of this spread depends on the position of "knocking" delta-function within M-long interval.
- The shape of this spread does not depend on the M-long shift
- The observed spreads, the internal responses on "knocks", are combinable into a FIR_{ANA}-like filter.

We define DSF in same way we define prototype FIR_{ANA}-like filters (for integer R): DSF is a full-band time domain representation of a "continuum" the from which these spreads of sub-band delta-function translations are sampled.

It's not quite clear how to find "exact" DSF analytically, for the given pair of arbitrary low-pass IN and OUT prototype filters (except that *dsf(t)= sinc(t)* for equal *IN(t)* and *OUT(t)*).

1) We can try to find DSF "experimentally", following a simple algorithm (essentially the same as in Basics 2.1.1):

   Choose LRIR > L to cover DSF sufficiently well

   Allocate aaDSF(LRIR,M);

   for m=1:M

   - *simulate full-band aRIR(1:M*LRIR) as a delta-function at the position (m+M*LRIR/2)*
   - *generate aIn as white Gaussian noise,*
   - *get aOut = filter(aRIR,1,aIN); No AWGN.*
   - *apply sub-band analysis to both aIn and aOut and retain DC sub-band only.*
   - *apply LMS … RLS to find out the aaDSF(:,m) m-th subsequence.*

   end

   *reshape the aaDSF(LRIR,M) to DSF(M*LRIR,1) by interleaving the subsequences.*

2) We can also try a simple *ifft(OUT(f)./IN(f))* with *OUT(f) =0* after 1/M. Practically, it's using either *fir2 /firls / firpm* / etc for a finite-length approximation of DSF, which is [ideally] a LP filter with f_{stop}=1/M and infinitely low sidelobes. Different functions and weighting will produce different results. So far it appears that *firls()* agrees with experimental data pretty well… but I do not know which one is "the best" and the objective optimality criterion. This DSF will ne noted in the following figures as low-cased "dsf".
3) (TBD) We can also try to find such *DSF(t)* so *conv(IN(t), DSF(t)) = OUT(t)*; if not exactly but as close as possible.

For FSAF to function properly, we need a compact DSF(t): with sidelobes that decay much faster than *sinc()* sidelobes. We know that $IN(f)^{-1}.*OUT(f)$ discontinuities in the n_{th} derivative [anywhere] stipulates sidelobe decay of f^{n-1}. Therefore, we need to lower $IN(f)^{-1}.*OUT(f)$ band-edge discontinuities of the function itself and its derivatives, as much as we can.

Obviously, *IN(f)* filter can not have zeros in the 1/M passband.

3.3.1 DSF considerations

There are some important notes:

- For a given *OUT(t)* filter, the *IN(t)* filter shall be chosen so that DSF is the most compact, i.e. the DSF sidelobes drop quickly and sidelobes' floor is low. We can design IN filter to achieve minimal aliasing, or to form most compact DSF (in a particular meaning). I doubt that we can do both and I believe that it's more important to have a compact DSF than very low aliasing noise floor
- The choice of adaptive algorithm may have serious impact on the *IN(.)* filter design.
- Ideally, all adaptive algorithms, from LMS to RLS, and everything in between, must converge to the same DSF, and in reasonable time.
- Convergence properties shall be invariable to the "m" offset index of wideband delta-function.
- In the absence of AWGN on OUT, the per-sub-band converged RESidual error spectrum shall be more or less uniform and independent of "m" offset index.
- Convergence shall be fast and the limit shall be low
- DSF must be robust to excitation, small changes in design and implementation imperfections
- DSF analysis shall aid in finding the "best" $IN(.)^{-1}*OUT(.)$ filters, together with most appropriate adaptive algorithm, given various optimality criteria and constrains which are hard to formalize completely.

…and a few less important notes:

- There is no need to process other subbands above DC. The DSF is the same for all sub-bands for prototype-modulated FB.
- Do not worry too much when RLS does not converge well. If so, this pair of $IN(.)^{-1}*OUT(.)$ filters is of no good anyway.
- The IN per-subband equalization (possibly, duplicated in OUT and reversed in RES) shall be reasonable and should not introduce artifacts.

3.3.2 DSF by SA-LMS

We'll start with SA-LMS, a version of LMS as the adaptive algorithm which lowers step size according to Stochastic Approximation (SA) rule o{*1/t*} ; $t \to \infty$, more precisely expressed as

$$\mu(t) = \mu_0 \frac{\tau}{\tau + \max(0, t - \Delta)};$$

$$\tau = \tau_0 L_{adf};$$

Where L_{adf} is the length of subband ADaptive Filter (ADF) in [complex] samples.

τ_0 is spectral normalization, which is usually chosen in the range 1.0 (white excitation) … 2.0 (a bit colored excitation). Here it is often set to 1.1 to ensure statistically optimal asymptotic SA-LMS convergence for nearly WGN as excitation and an AWGN approximation of aliased signal here. The configuration parameter fsaf_adf.sstau is the inverse of τ_0 to make zero a meaningful default (SA is effectively off, $\mu(t) = \mu_0$).

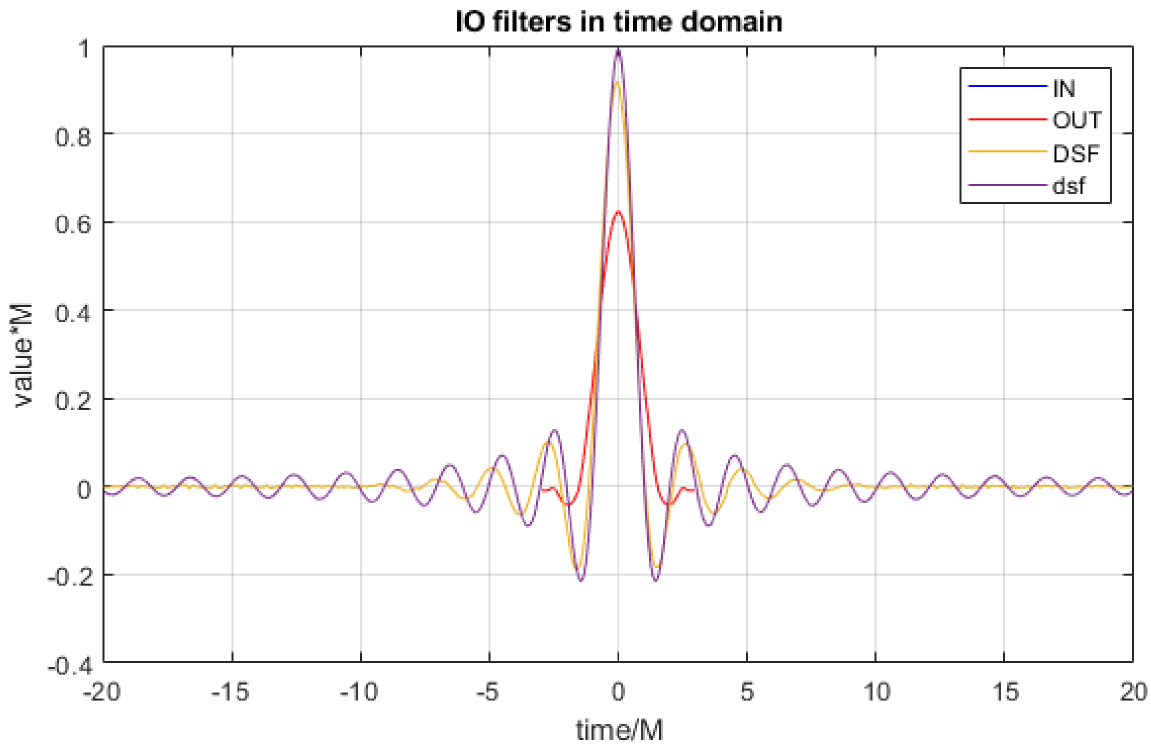
Δ is the time required to reach noise floor on maximal step size μ_0 (usually $\mu_0 = 1$). Δ can be estimated by L_{adf}·(-aliasing_level_dB) /(μ_0 · 5.2dB).

Although using BLUE RLS may look as much more appropriate choice for finding *DSF* quickly, precisely and reliably… it's not so simple. The FSAF RLS is an advanced topic and will be discussed later.

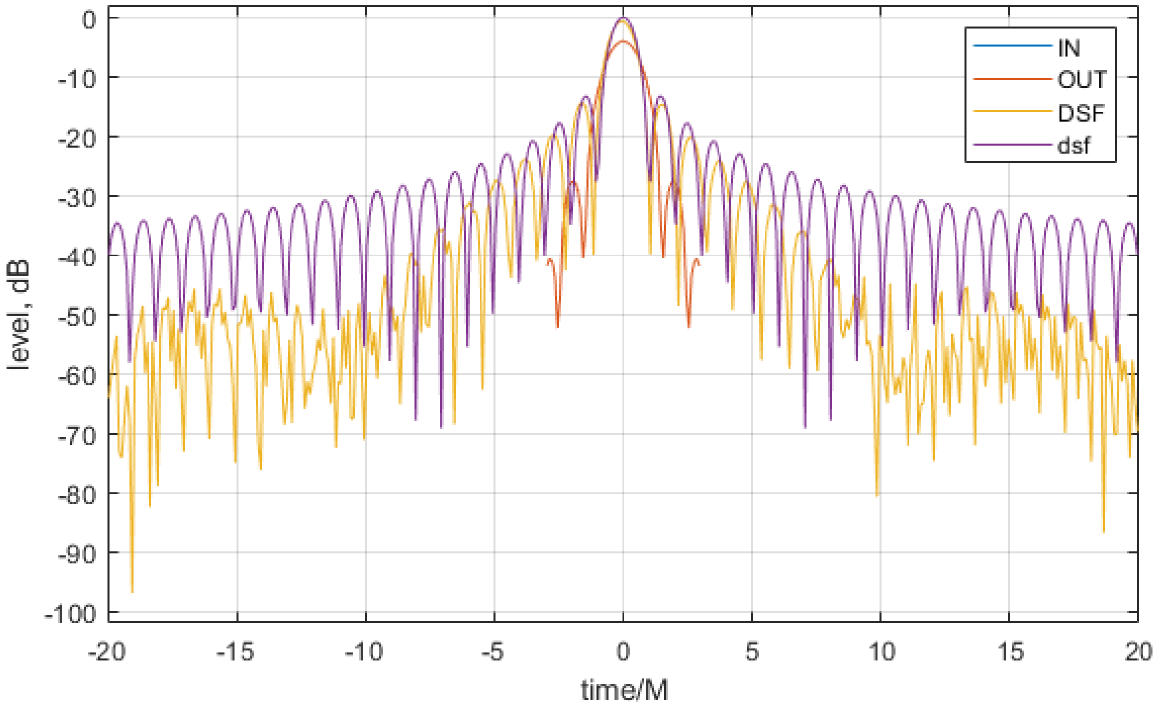
Although using non-recursive methods like properly Weighted LS, accounting for all second order effects, may look as much more appropriate choice for finding *DSF(t)*, it would hide the details of real-life adaptive algorithms' convergence. There is not much sense in a $IN(t)^{-1}* OUT(t)$ design with a great *DSF(t)* if there are no algorithms that could converge to that *DSF(t)*.

3.3.3 DSF for SAF1988 [305]

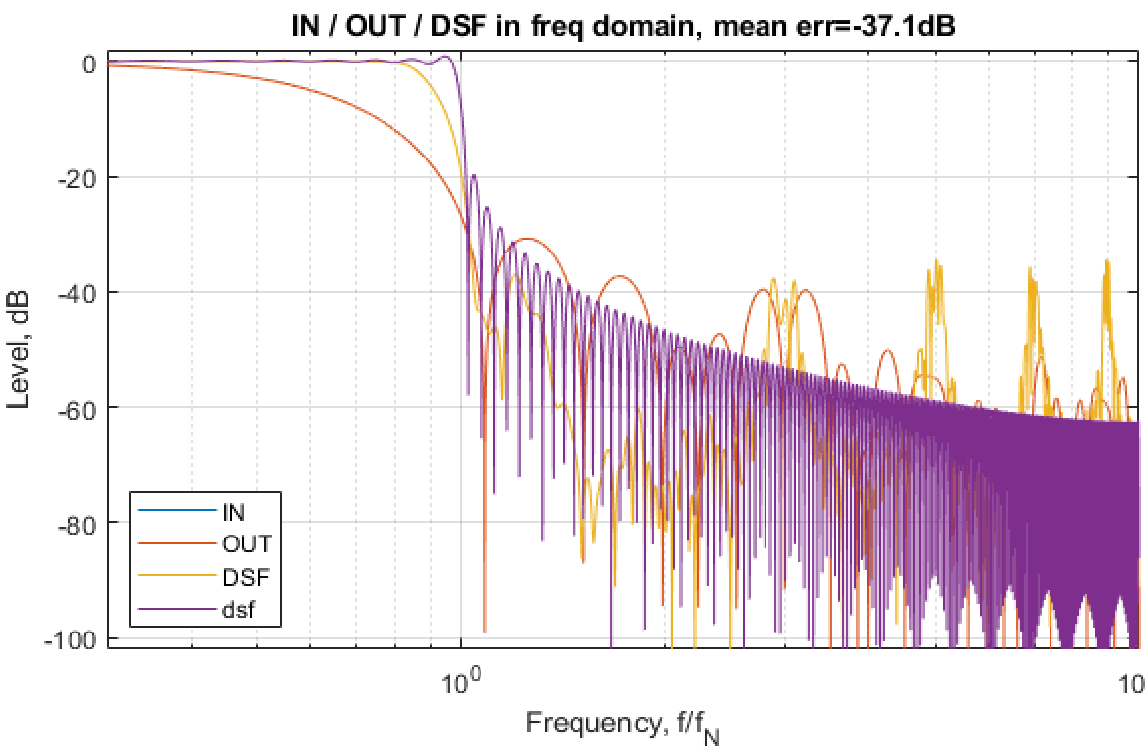
Here we analyse what the asymptotically optimal SA-LMS converges to, on excitation of white noise, when we know that it should be *sinc(t/M)* : (IN filter)$^{-1}$ -> up/down sampling… -> (OUT filter=IN filter).



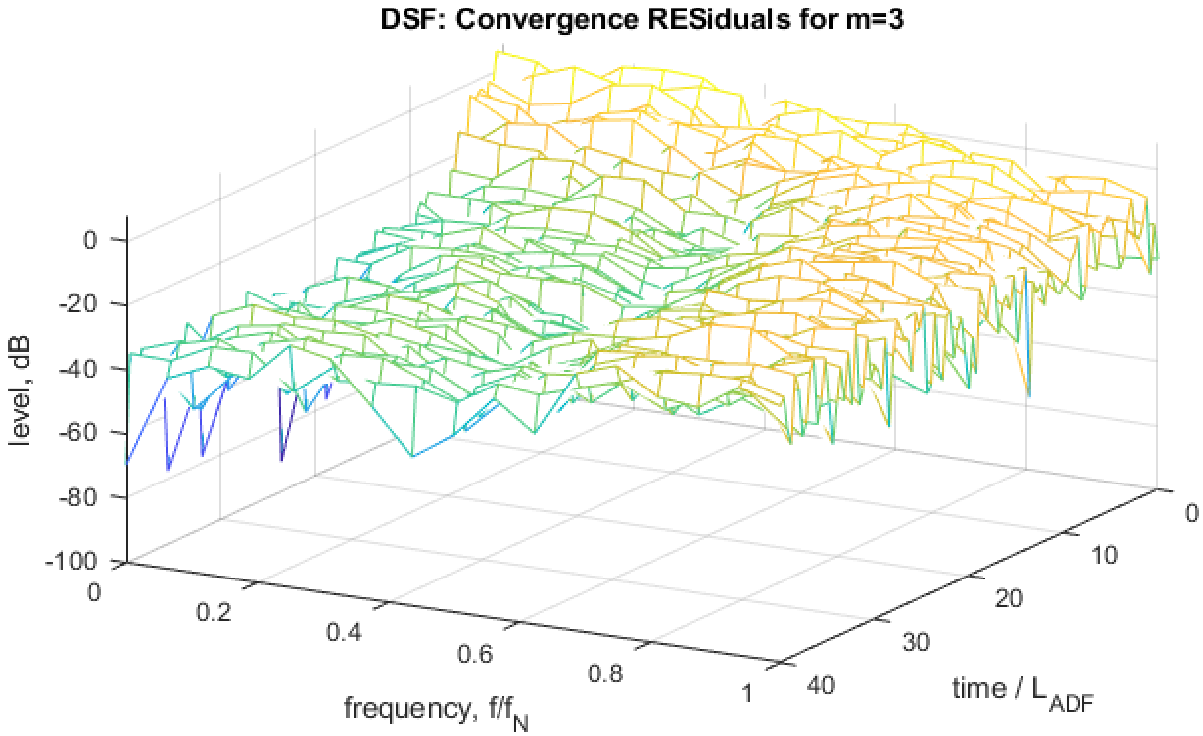
… and accompanying picture in dB scale to see the minute details:



LMS does not converge the yellow "DSF" to the true *sinc(t)* - while *firls()* predicts it properly (magenta "dsf"), with zeros at correct places, and with correct decay $1/\delta_f$, from -13dB down).



The predicted "dsf" shines with proper Gibbs effects, while LMS's "DSF" does not. It follows "dsf" only till $\sim 0.8{*}f_N$, and has high false spurs, about -35dB, on and around odd multiples (1, 3, 5, …), in full agreement with baseband sampling theory, of f_N in stopband.



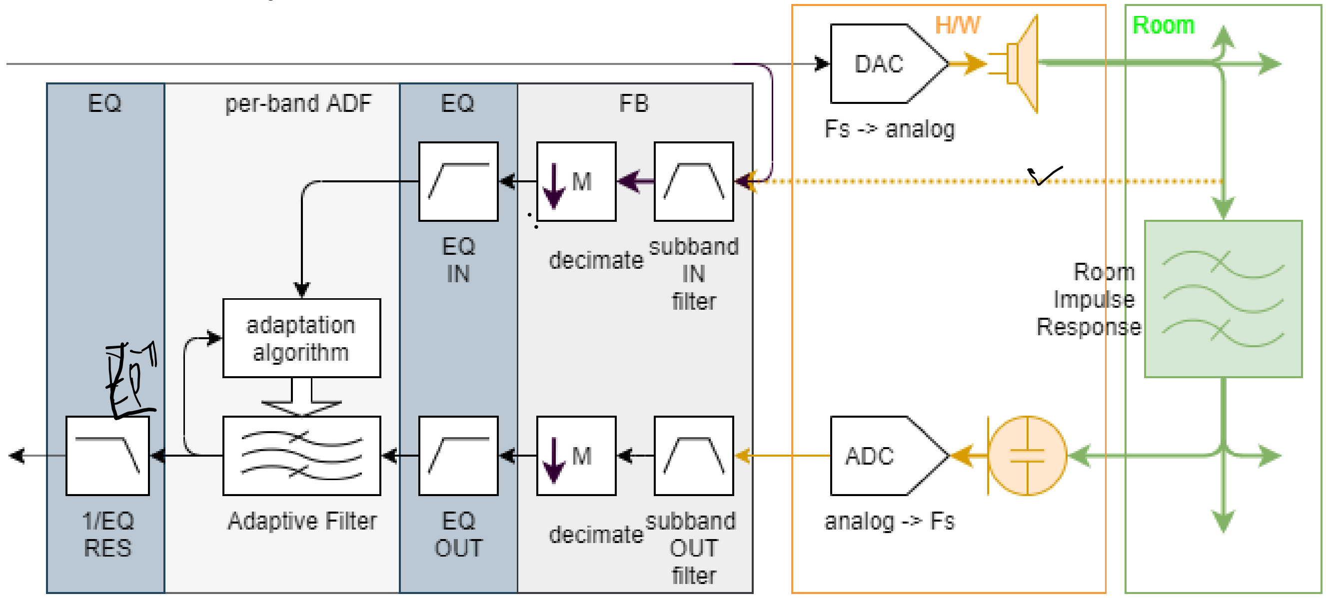
The residuals remain poor, at approximately -25dB, in good agreement with aliasing analysis, for any offset index m, except for m==M, which will be discussed soon.

3.4 IV / EQUALISING IN/OUT [305]

We know that the IN signal is invariably shaped by the *IN(.)* filter, which that causes all kinds of ill problems for all adaptive algorithms, LMS … RLS and everything in between.

These distortions are known in the time of FSAF FB design, right after *IN(t)* and *OUT(t)* have been designed. Therefore, we can compensate for them by adding a per-subband equalising filter (EQ), which “mirrors” the attenuation inserted by *IN(f)*, so that $\|db(EQ(f)) + db(IN(f))\| < thr$;

The same EQ is applied to both IN and OUT, with reversing it on RES. In the theory of System Identification, such EQ is known under the name "Instrumental Variable".



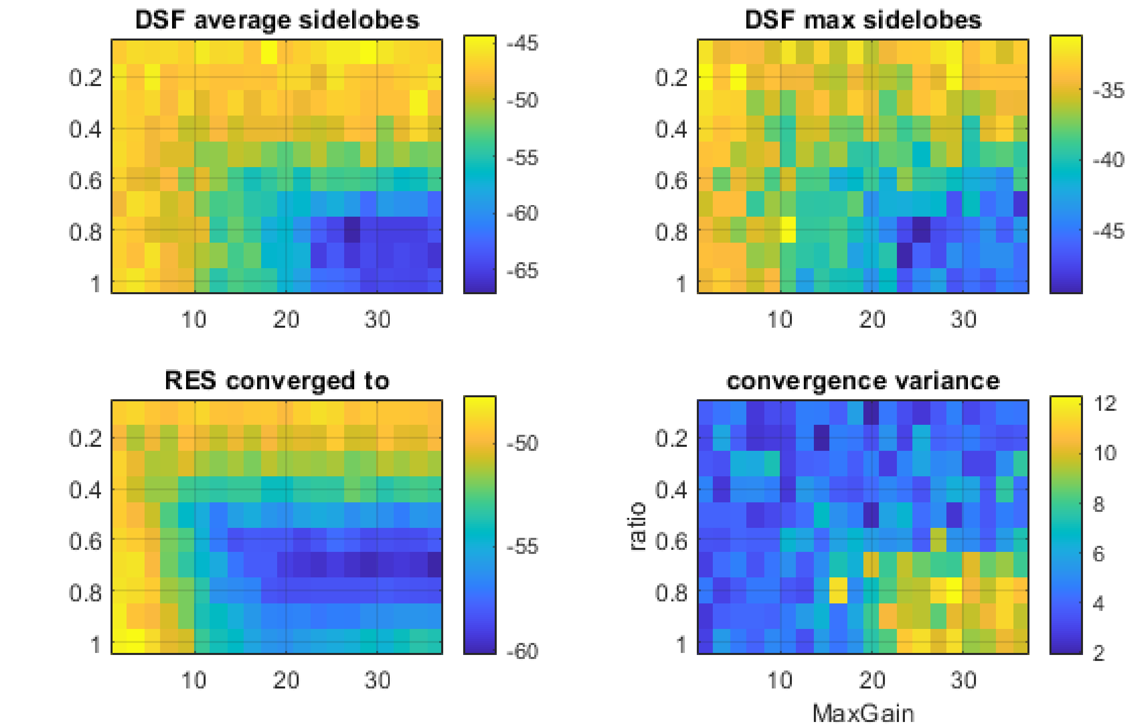
The FSAF preferable way here is to implement the EQ is IIR (of 5th SOS order) because it's easier to form an inverse EQ^{-1} on the output.

The parameters of EQ are adjusted manually for each pair of *IN(t), OUT(t)*, according to the results of modelling. The two main parameters of equalization are EQ ratio (how closely $EQ(f)$ follows $IN(f)^{-1}$) and max gain of EQ. For simplicity here, it is set to $\{-IN(f_N)-6dB\}$ 6dB below max IN filter attenuation – which is Wiener-inspired.

On one hand, for SA LMS, the choice of EQ determines each and every property of the SA LMS convergence process. On another hand, the choice of EQ depends, very heavily, on the choice of adaptation algorithm and its regularization.

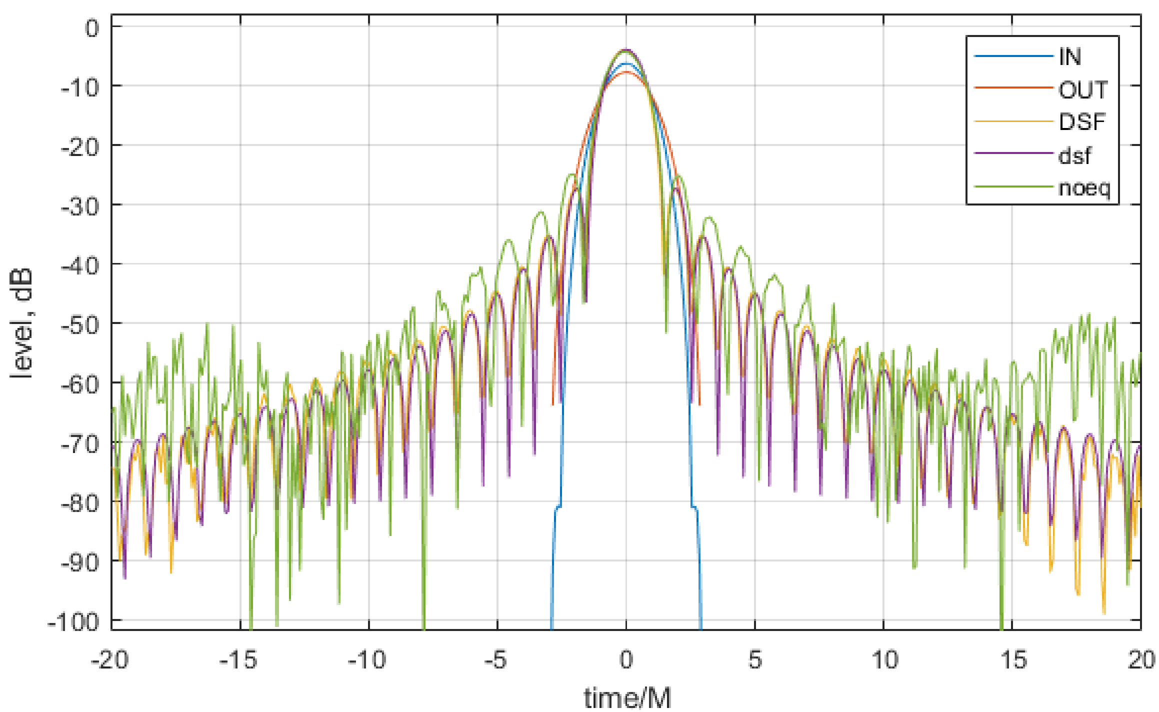
While adding EQ, we need to reverse EQ from high-frequency pre-emphasis into low-frequency pre-emphasis for even bands. The reason is the way bandpass sampling of the complex signals is done. The subsampled frequency wraps around in an alternating order inside odd and even sub-bands. The frequency always wraps around in a circle placing f_N*(1,3,5,...) at 180˚ and f_N*(0,2,4,...) at 0˚. Therefore, the 2nd subband, from 0 to 2*f_N, has its center at 180˚ but 3rd subband, from 1*f_N to 3*f_N, has its center at 0˚, etc.

Here are typical plots for EQ tuning: the average and maximal sidelobes of DSF frequency response in the stopband, and the convergence of residual noise – we can't have both at minimum at the same time:

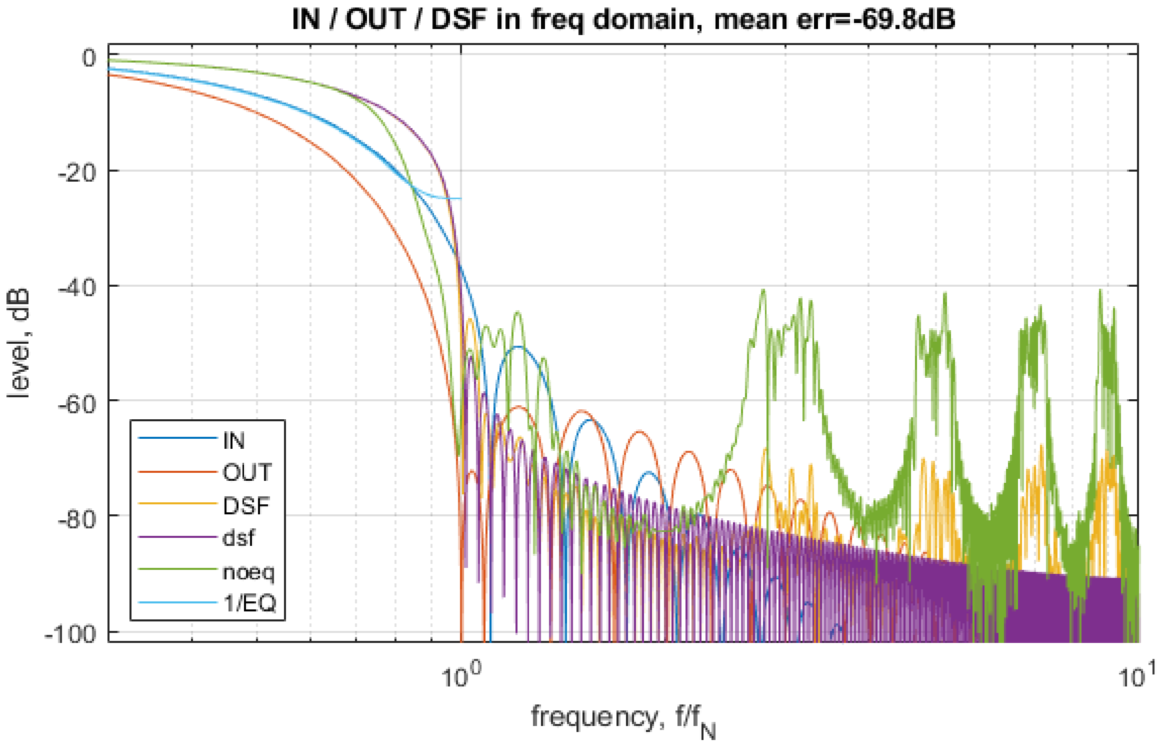


3.4.1 DSF for Short FB + EQ [305]

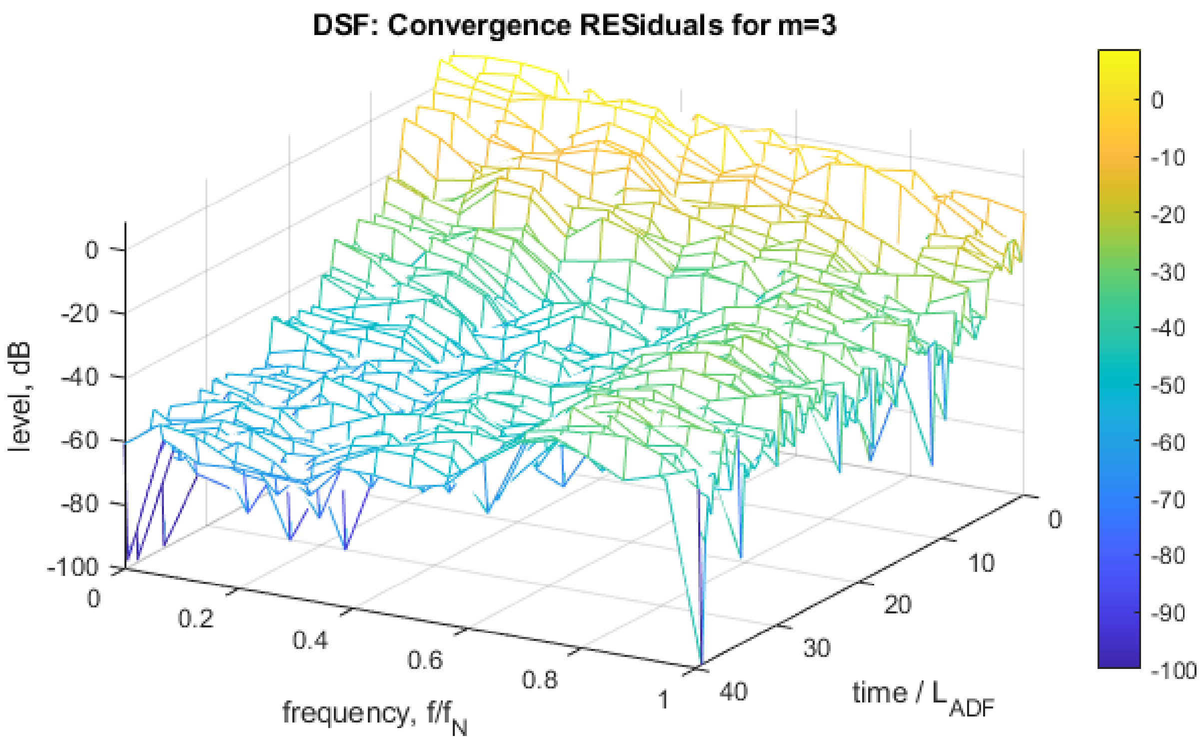
Let's display the EQ effect on the example of Short symmetric FB with L=6 (which enforces latency=5 frames of M samples). We omit time-domain linear scale graph due to its little informativity:



We can easily see that most problems in SAF1988 with LMS were due to the lack of EQ. The reasons for that omission were and still are unclear to me. With EQ enacted, conversion of "DSF" to the predicted "dsf" is obvious. In frequency domain, max error is about -50dB, at $f=(1\pm\delta)*f_N$, and -70dB elsewhere.



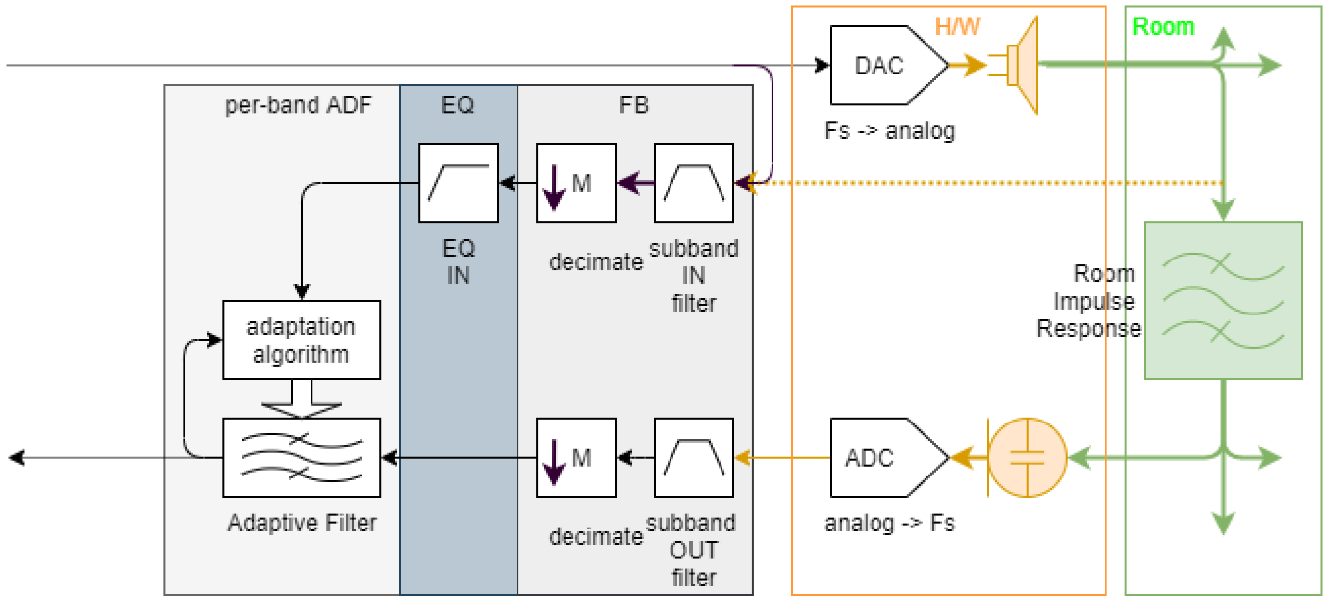
SA LMS converges quickly to the aliasing level divided by EQ^{-1}(which is -EQ in dB domain):



After EQ^{-1} is applied to RES, the residual level shall drop to the predicted aliasing level [exactly].

3.5 EQUALIZING IN ONLY

We can expand the meaning of DSF by including EQ inside the adaptation process, by removing mutually compensating OUT EQ and RES EQ^{-1} (or, by applying EQ to IN only). We'll still be able to predict its shape – but only if EQ is implemented as symmetric FIR (which is not true here).

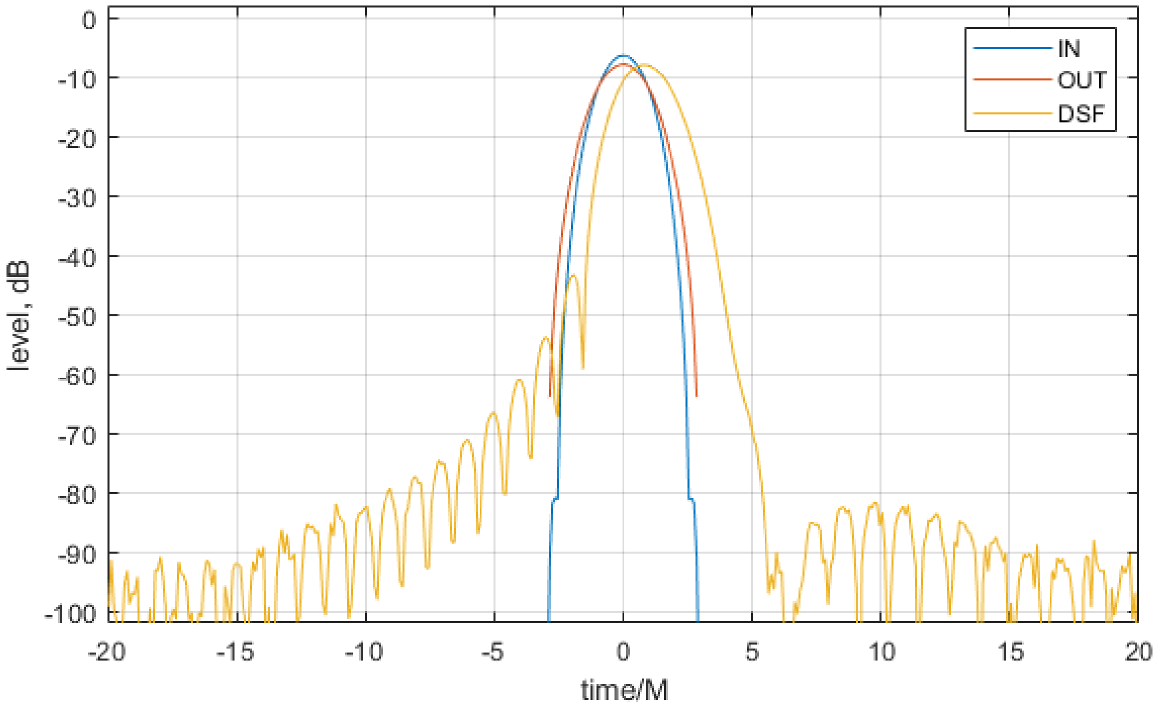


With this approach, we maximally untangle the IN and OUT processing in FSAF, with significant benefits. When *EQ(f)* become very close to the $IN(f)^{-1}$, the *DSF(f)* becomes very close to the *OUT(f),* and that makes *DSF(t)* as compact as possible for a given design.

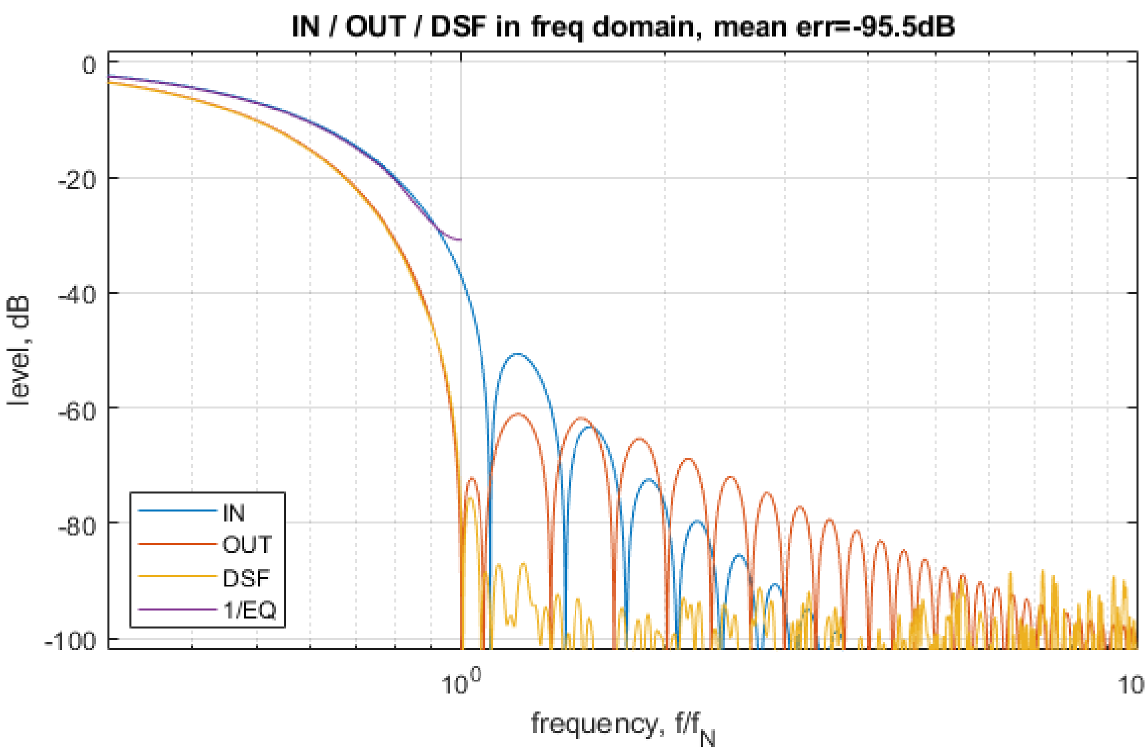
Ideally, *EQ(f)* IN(f)* shall form Wiener's filter for signal re aliasing distortions. Thus, having -6dB attenuation at the 1/M makes some sense.

3.5.1 DSF for Short FB + EQIN [305]

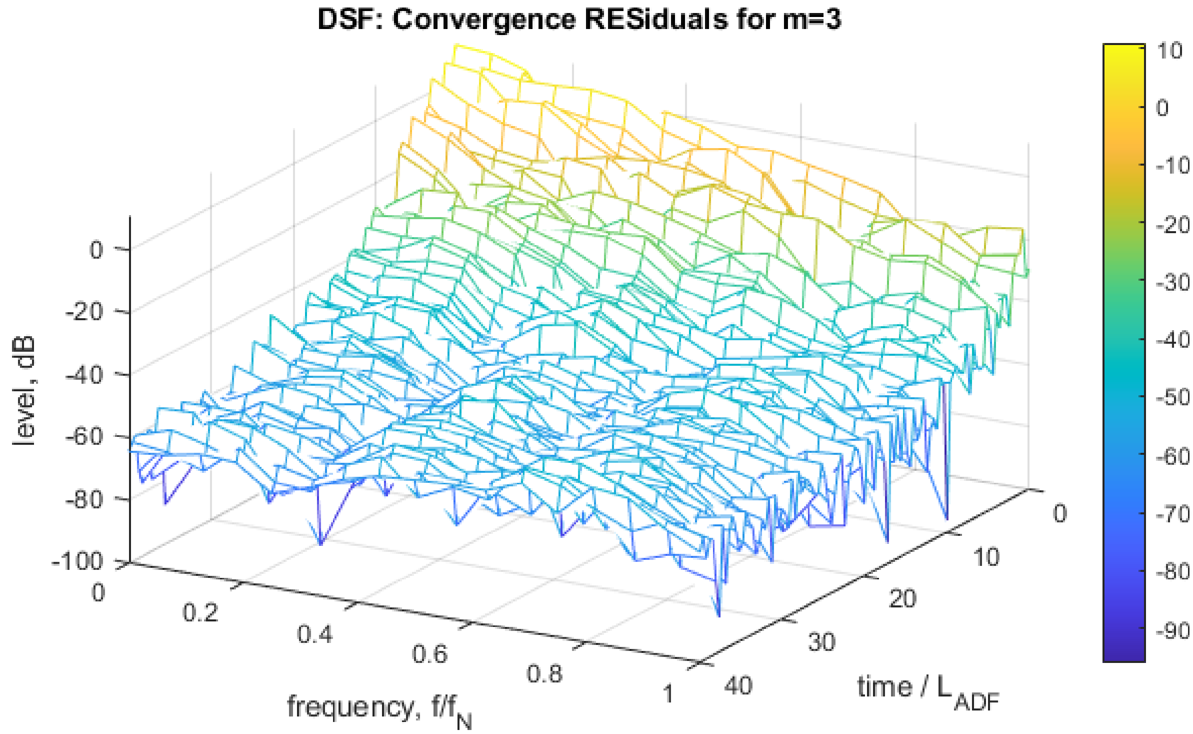
The same FB design as in previous chapter results in a more compact DSF, with time-domain sidelobes below -70dB re peak:



The frequency-domain noise floor of convergence is much cleaner:

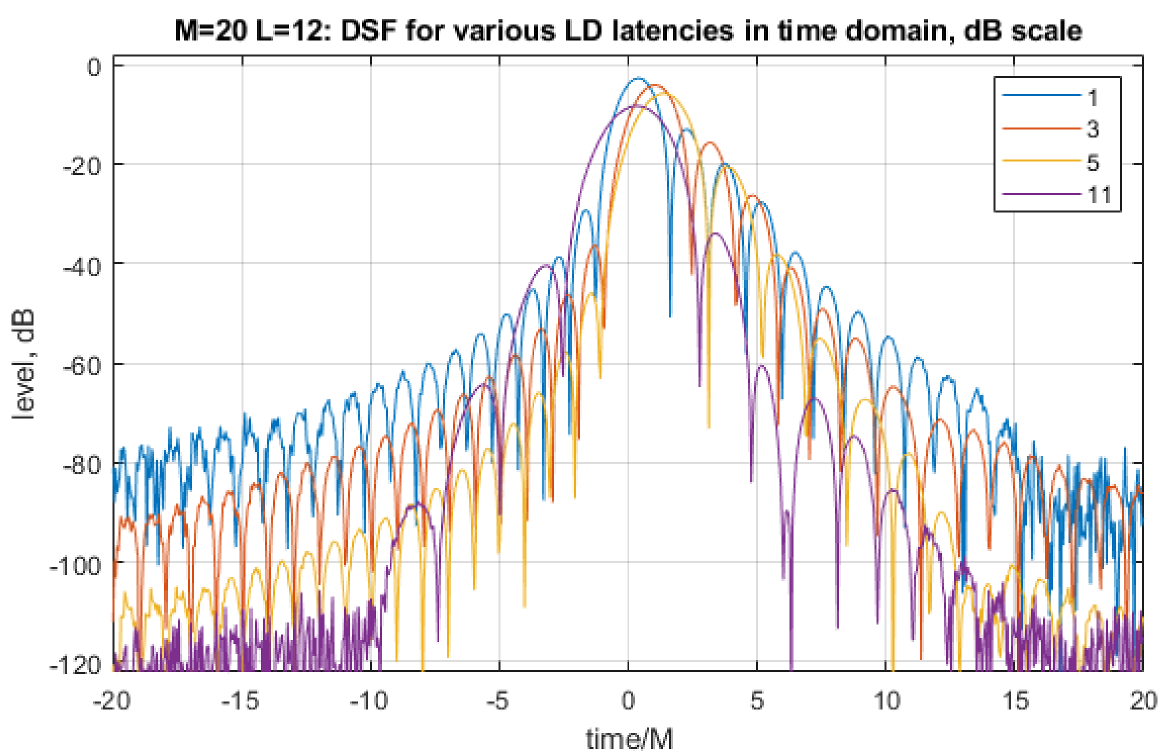


And the residual error spectrum exactly corresponds to the aliasing noise:



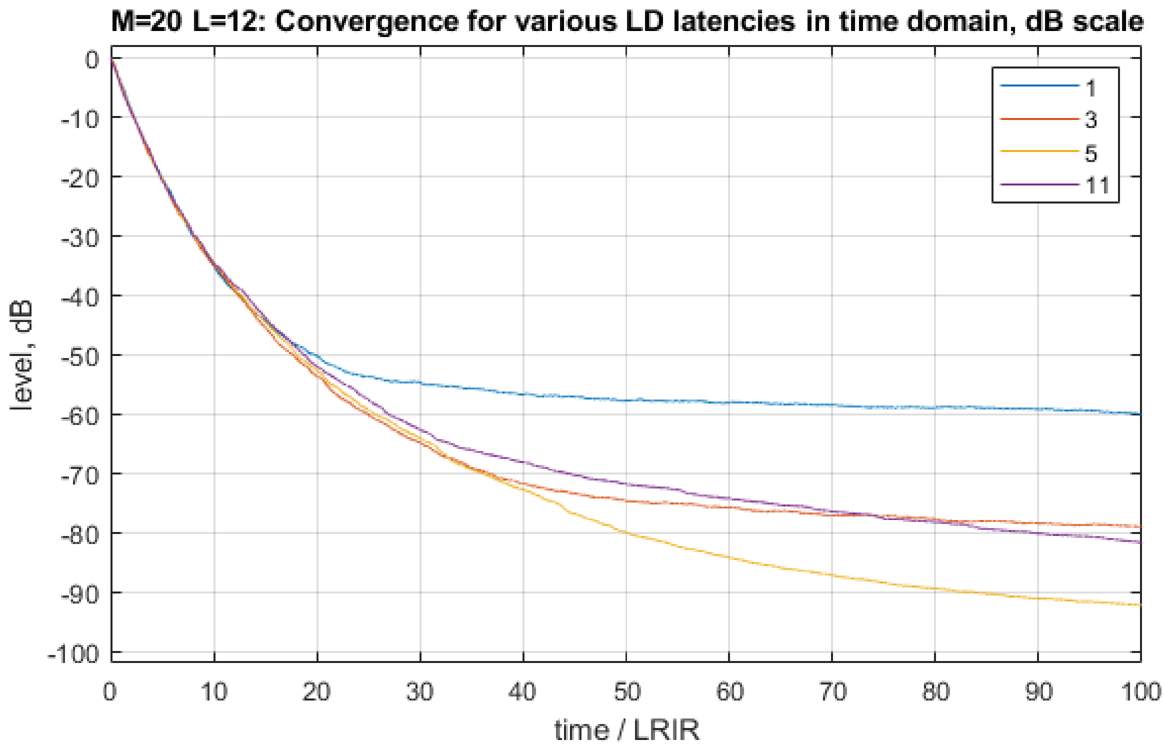
3.5.2 DSF for LD Long FB + EQIN [306]

Here we analyse DSF for L=12 with latencies of 1,3,5, and 11 frames, the same baseline designs we discussed in the aliasing analysis.



The design with algorithmic latency of 11 frames is symmetric, others are of LD kind and not symmetric. The LD DSFs become more compact and move to the right with increase in algorithmic latency. Thus, decreasing FB latency may not be so beneficial for FSAF as it looks.

The convergence of DSF also depends on the amount of DSF tails which stretch beyond ±LRIR/2. If a DSF does not decrease fast enough, it can not converge deep enough due to under-modelling effects.



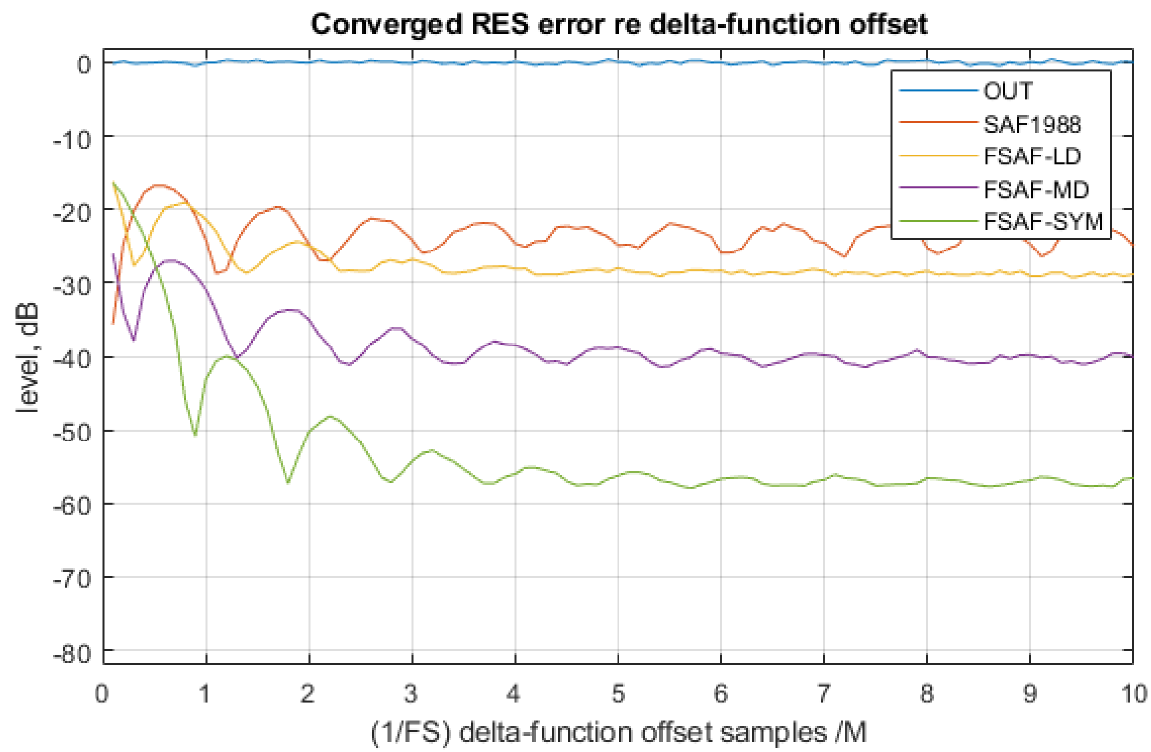
It's tempting to decrease MSE by $\sqrt{M}$ with filtering out the noise in the region *[f_N...M*f_N]* but the meaning of this operation is not fully clear.

3.6 Impact of DSF under-modelling

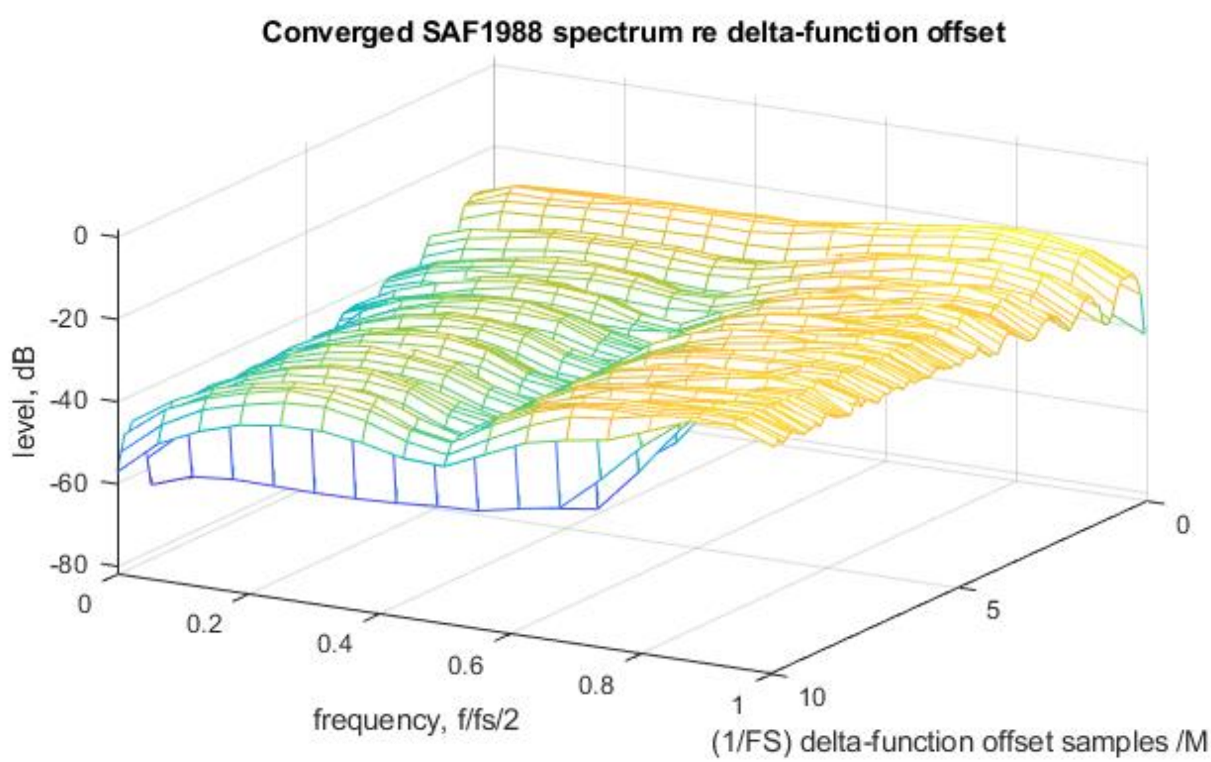
The results of neglecting of non-casual DSF spread need to be demonstrated explicitly. We can move the delta function in the full-band RIR emulation from Δt = 0 to the middle of LRIR, and observe how well SA LMS converges for each delay value (testing that a multirate system is invariant to a "knock" offset).

3.6.1 SAF1988 vs FSAF [307]

Here we compare the traditional SAF design, with two identical PR filters, to the proposed FSAF design with the same filter length and varying algorithmic latency (LD = 1 frame, MD = 3 frames, SYM =5 frames, same as for SAF1988:

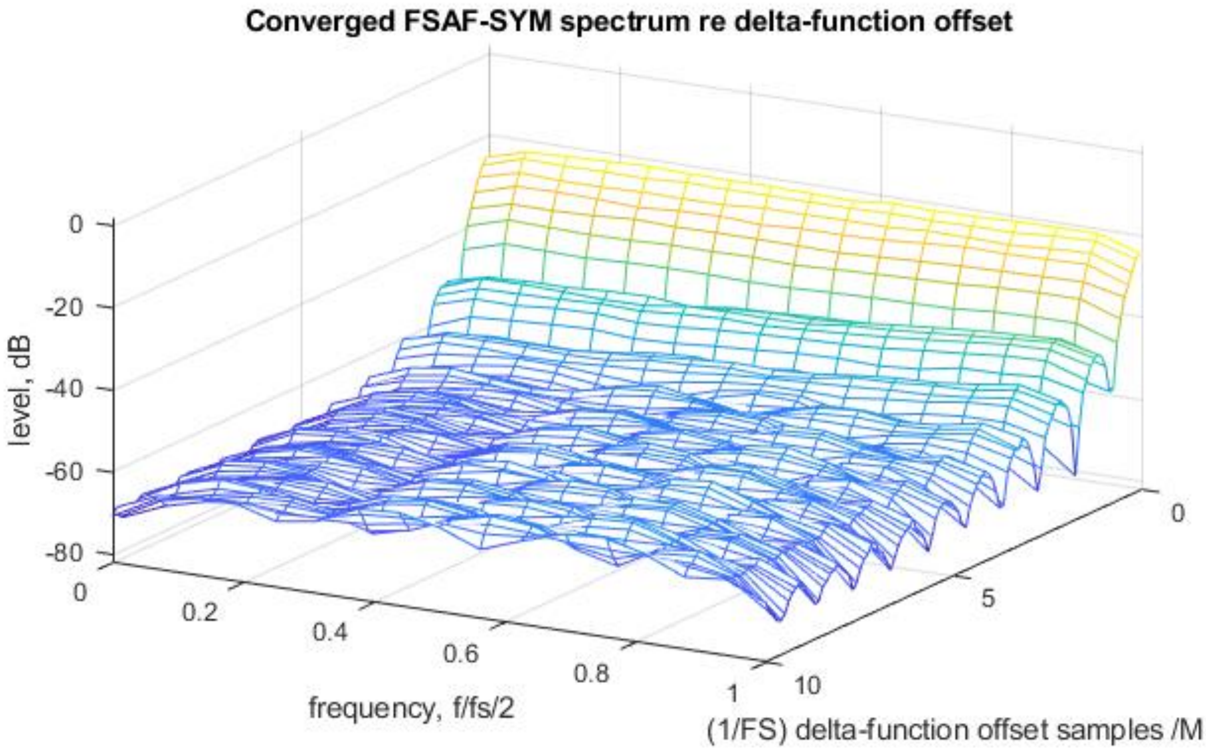


However, that's not the entire story. The RES error spectrum for SAF1988 is very heavy towards the band edge but that noise is also much attenuated during synthesis so that the resulting full-band RES is about 10dB lower.



Despite being lower, it suffers from the lack of uniformity even more. The full-band RES error is very low for pulses if and only if such pulses are at τ=M*k samples. The same full-band RES error is high (~-33dB) otherwise. Thus, the RIR pulse of mechanical coupling between loudspeaker and microphone can be cancelled very well, but the echo corresponding to the remainder of RIR, which comes, say, 10+ms later at arbitrary times, can not be cancelled well (especially first reflections) no matter which adaptive algorithm we are using and how "hard" we try[2].

For FSAF symmetric case with the same latency, the RES error is much lower:

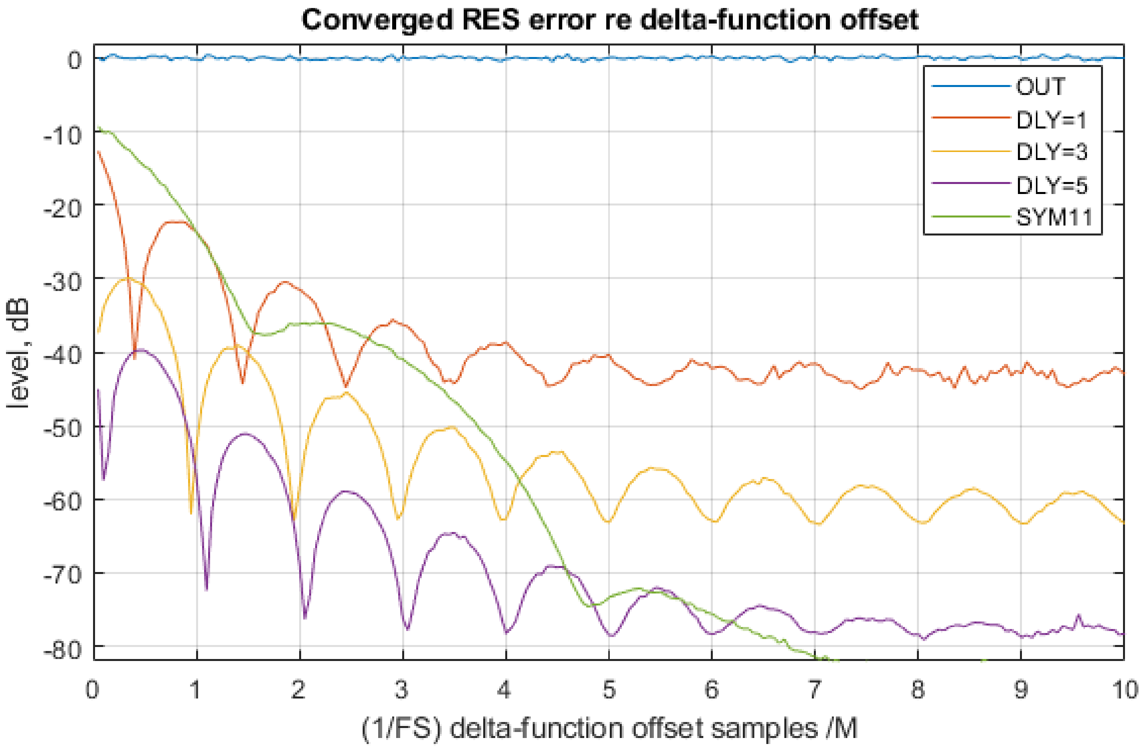


…and more or less uniform, it does not depend much on the specific pulse timing. However, it requires that a microphone signal is delayed at least 1.6*M samples relative to speaker signal, if cancellation of at least 45dB is required (2.4*M if 50dB). That usually not a problem due to double-buffering delay interfacing hardware.

[2] I wonder what have been done in areas like geology / seismic research where people routinely probe the signal in 2 (or more) points and compute the TF / IR between them… should they have essentially the same problems as in SAF-1988? The 1st ADC antialiasing filter had to be wider than 2nd ADC's to minimize the problems.

3.6.2 FSAF Long FB + EQIN [307]

The impact of under-modelling on LD Long FB + EQIN is quite complicated:



The minimal latency that needs to exist between microphone and speaker signals to get 50dB of cancellation is only 0.8*M samples for DLY=5 case but 2.7*M samples for DLY=3. Therefore, if IO latency is minimal, and M is high, the case of DLY=5 is better because we would have to add extra delay of almost 2*M to the microphone signal for DLY=3.

The symmetric filter case needs the highest microphone-speaker latency to achieve its potential. Most likely we need to add extra delay in this case.

The extreme case of DLY=1 is implementable only for LD Long FB, and it can not achieve a high degree of RIR adaptive cancellation. The problem of low-delay subband adaptive filtering is not solvable by using LD filterbanks. The solution however exists and will be discussed in the chapters to follow.

This additional "look-ahead" latency due to non-casual DSF spread, required for an adaptive filter to achieve a desired level of echo/reverberation/etc cancellation, puts a practical limit on the minimal width of subband filters. If we need to add 5 non-casual taps to ADF, there is little if any sense to use very high M so that subband $L_{ADF}=L_{RIR}(RT_{60}, F_s)/M$ becomes comparable to 5 taps. If we want L_{ADF} to be about 300ms and have >= 8 taps, then we shall not go below 25Hz and 40ms frames, and 50Hz / 20ms for 15 taps (+5 taps per non-casual DSF extension, less if operating system IO latency is worse than DSP-like double-buffered).

3.6.3 Full-band Residual noise

TBD

3.7 WRLS FOR FSAF

Alas, the straight-forward subband implementations of a standard RLS are inherently unstable.

3.7.1 Basics

Standard WRLS is based on several assumptions - which are broken for audio sub-band adaptive filtering:

- System is perfectly linear
- No under-modelling
- Noise is present only on output
- Noise is white
- Noise variance is known

$x_t = [s(t); x_{t-1}(1{:}\, L_{RIR} - 1)]$; where $s(t)$ is down-sampled sub-band-filtered speaker signal.

$z_t = D_t x_t$; projection vector

${\nu_t}^2 = {x_t}^H z_t = {x_t}^H D_t x_t$; expected residual error variance

$\mu_t = {\nu_t}^2/({\nu_t}^2 + {\Sigma_t}^2)$; step size

$h_{t+1} = h_t + \mu_t z_t (y_t - {x_t}^H h_t)/{x_t}^H z_t$; filter vector update

$D_{t+1} = D_t - \frac{\mu_t D_t x_t {z_t}^H}{{x_t}^H z_t} = D_t(I - \mu_t \frac{x_t {z_t}^H}{{x_t}^H z_t})$; dispersion matrix update

Also, because RLS is a recursive implementation of LS, and assuming a stationary noise level:

$$D_t^{-1} = D_0^{-1} + \sum_{k=1}^{t} \frac{x_k x_k^H}{\Sigma_k^2} = D_0^{-1} + \Sigma^{-2} \sum_{k=1}^{t} x_k x_k^H = D_0^{-1} + \Sigma^{-2} S(t);$$

D_t^{-1} quickly progresses to become the *s(t)* excitation's autocorrelation matrix because Fisher's $\Phi(t)$ elements Φ_{rc} becomes $S_{|r-c|}$, depending only on the absolute value of relative delay between row and column indices. The excitation *s(t)* passes through *IN(t)* filter and thus the operation $z_t = D_t x_t$ becomes synonymous to filtering excitation through a squared inverse $IN(f)^{-2}$, which also serves as a pre-emphasis for the aliased input = noise on input.

There are several commonly used approaches to improve robustness of RLS (beside those discussed by Golub and Van Loan):

- Use forgetting factor λ to WRLS
- Adding a small, $\delta \ll 0$, diagonal to D_t in Kalman manner

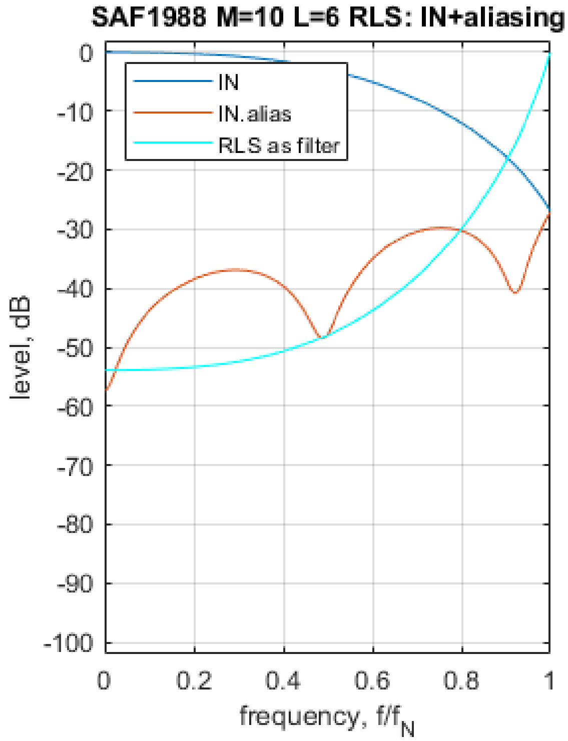
$$D_{t+1} = D_t \left(\lambda^{-1} I - \mu_t \frac{x_t {z_t}^H}{{x_t}^H z_t} \right) + \delta I;$$

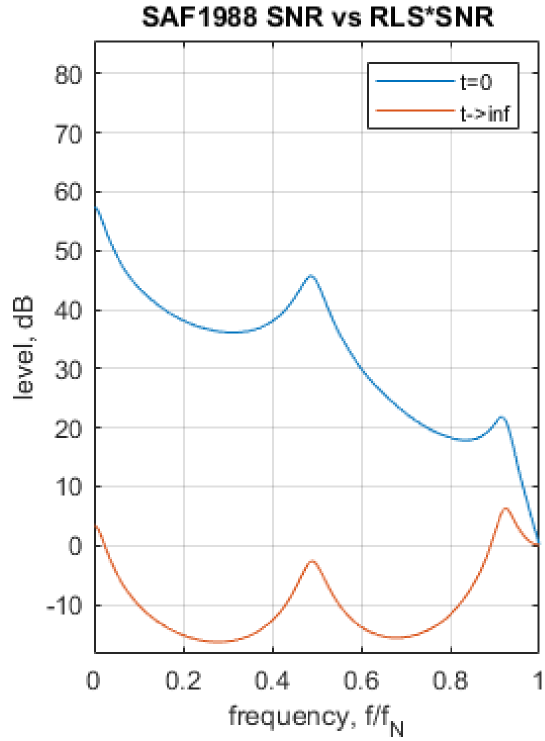
- Ensure that $trace(D_t) \geq \theta$ with appropriate θ

$$D_{t+1} = D_{t+1} + I * max(\theta - trace(D_{t+1}), 0)/L_{RIR};$$

3.7.2 Pictures for SAF1988 [309]

Let's illustrate this wordy narrative with a picture:



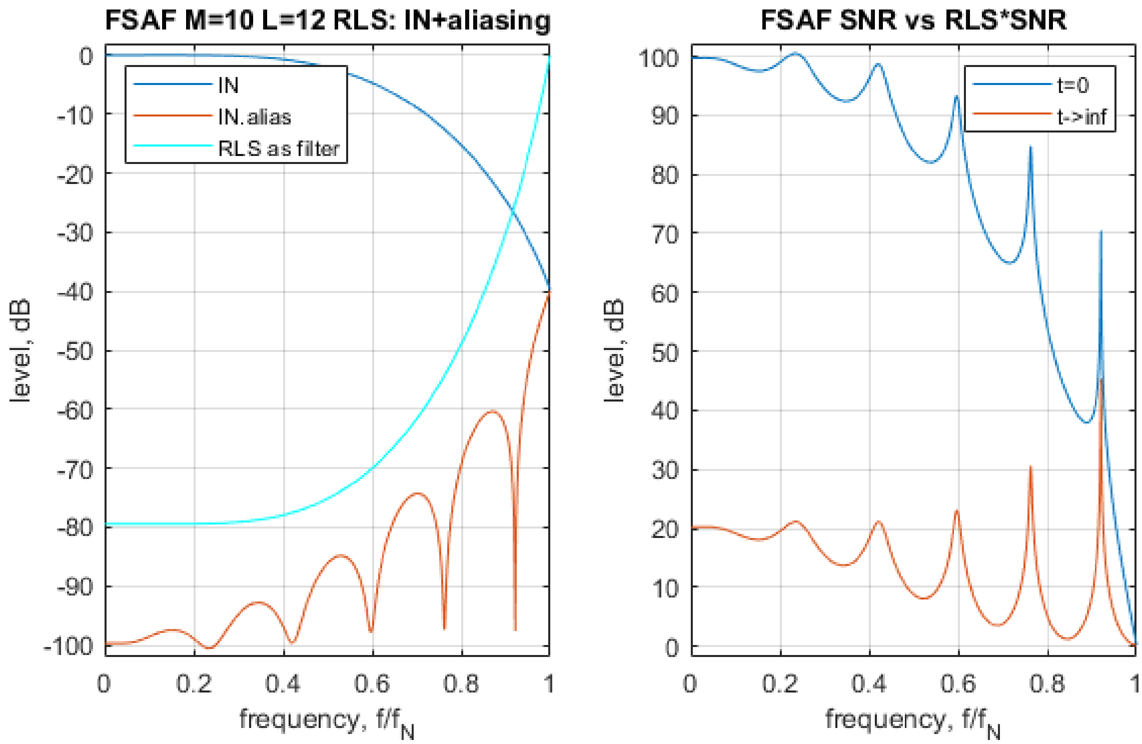


On the left, we can see the subband passband and aliased signals, and the filter which WRLS uses to create projection vector $z_t = D_t x_t$, after a few L_{RIR} iterations on WGN. On the right, we can see SNR in the beginning, when D_t is a diagonal matrix, and after these few L_{RIR} iterations.

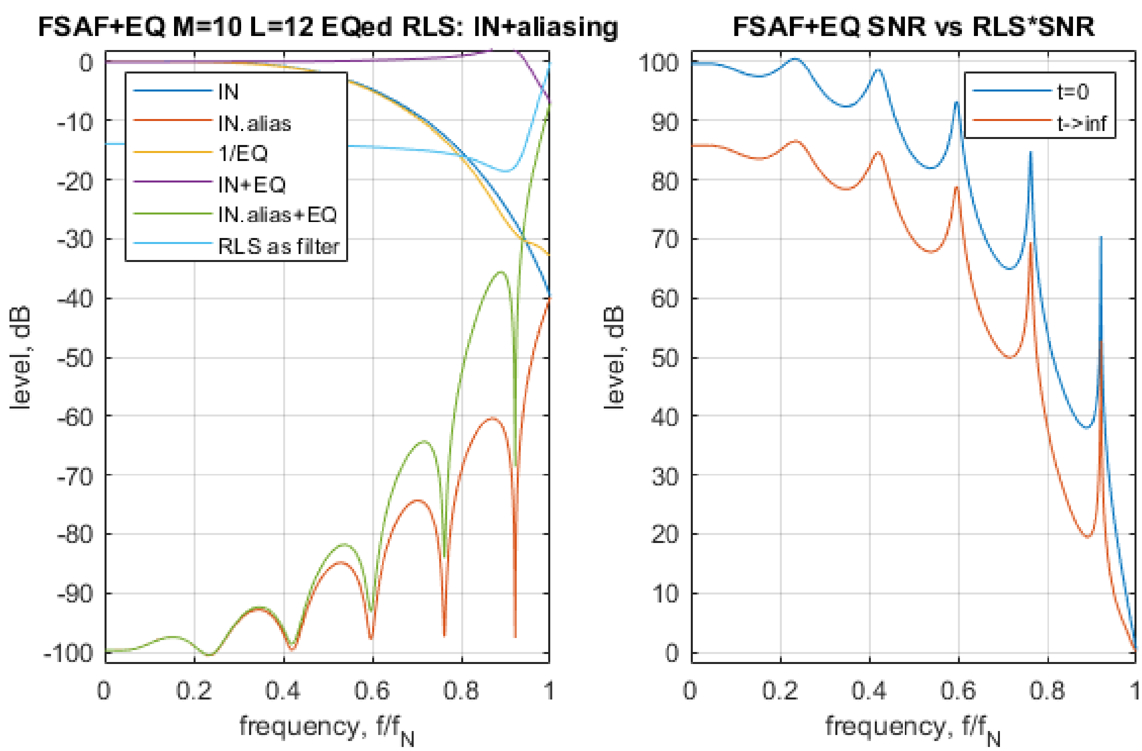
We see that sub-band WRLS converges to diverge, to suppress the 'true' signal to or below "noise on input" level. Then, the error signal $e_t = (y_t - {x_t}^H h_t)$ is still calculated with high SNR input x_t but the projection vector z_t becomes badly distorted. As the result, WRLS tends to diverge. The process is unstable and stochastic and therefore each realization is unpredictable.

3.7.3 Pictures for FSAF [309]

The situation does not improve cardinally if we use standard WRLS after FSAF FB.



The SNR is better for all iterations, but still marginal and WRLS still can diverge. The change comes with EQ:

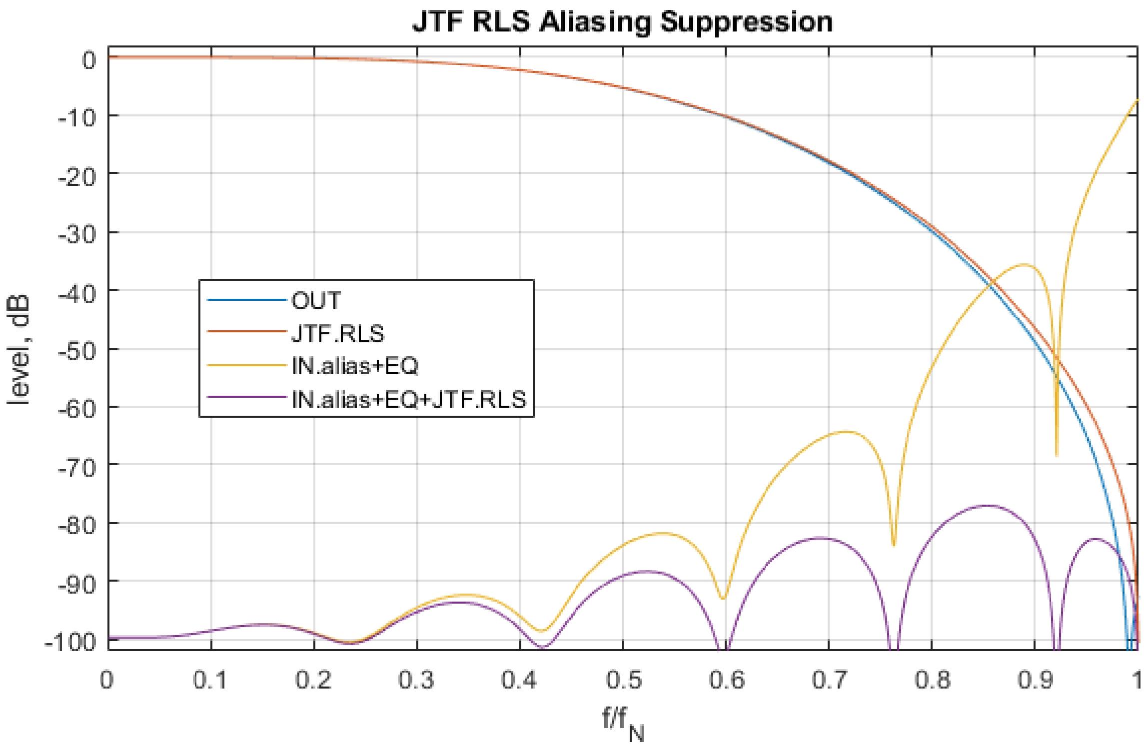


For WRLS, a carefully designed EQIN becomes necessary - but not sufficient.

From observing the graphs above, we also conclude that *OUT(f)* filter with fast sidelobe decay is strongly preferred, to ensure good SNR (more exactly, signal – aliasing ratio) in the entire passband.

3.7.4 Properly Initialized JTF-RLS [309]

We apply Joint Time Frequency initialization to ensure that the system identified is band-limited. As we know from Part II, for any vector v: $v'D_t v <= v'D_0 v$, including v corresponding to Fourier transform on the band edge, $e^{-j\pi k}$. When we constrain JTF-RLS to the shape of OUT filter, we automatically suppress the aliased input signal leaking from the adjacent bands, which both improves JTF-RLS convergence and lowers the chances of RLS diverging.



Of course, the frequency domain initialization of off-diagonal D_0 shall swap LPF / HPF mirroring in even bands, in the same manner as EQ.

JTF-RLS also must be hardened by adopting all applicable known regularization approaches, and by adjusting step-size according to the aliased energy. Alas, we can never be absolutely sure that sub-band JTF-RLS remains unconditionally stable, so Multiple Models (AFMM / IMM / etc) must be used. As minimum, save the known-good h_t, D_t from time to time, and keep checking for divergence. Whenever it happens, reinitialise JTF-RLS to the last "golden" solution. However, have in mind that you are utterly unlikely to invent an algorithm which has not already been invented, tried and tested by scientists who have been working on MM-Kalman radar/sonar target tracking.

3.8 DSF Robustness

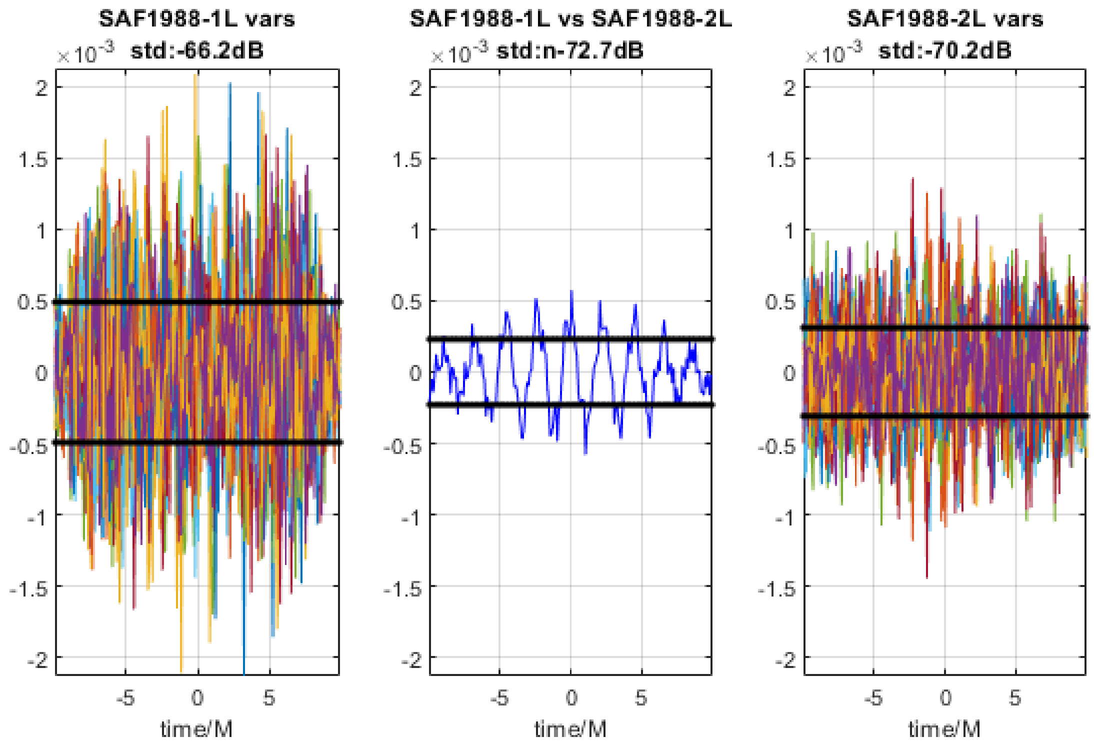
The robustness of IN/OUT codesign can be seen from many different viewpoints. Here we discuss how to

- check if an adaptive algorithm converges to the same DSF on different realizations of excitation
- check if an adaptive algorithm converges to the same subset of DSF if limited by ADF length
- check if different adaptive algorithms converge to the same DSF

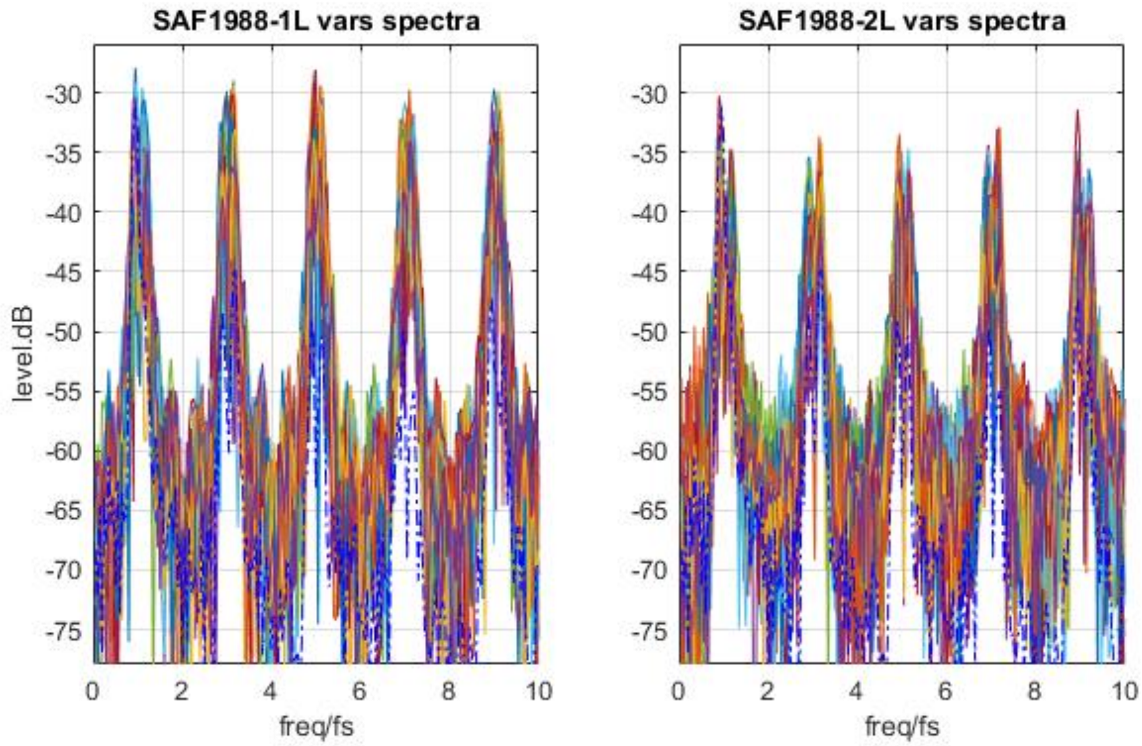
By no means this list is exhaustive or conclusive.

3.8.1 SAF1988: noise + L_{RIR} [308]

The first 2 checks are combined in one test. Here we design *IN(t)* and *OUT(t)* filters, and find DSFs of the length $L_{RIR}*1$ and $L_{RIR}*2$, running the same adaptive algorithms for $B_{LKS}*L_{RIR}*k$. We repeat it R_{UNS} times to have R_{UNS} realizations. Then we take a central part of the second DSF ensemble, of the $L_{RIR}*1$ length, and compare ensembles to their respective means, and the means between themselves, both in time and frequency domains.

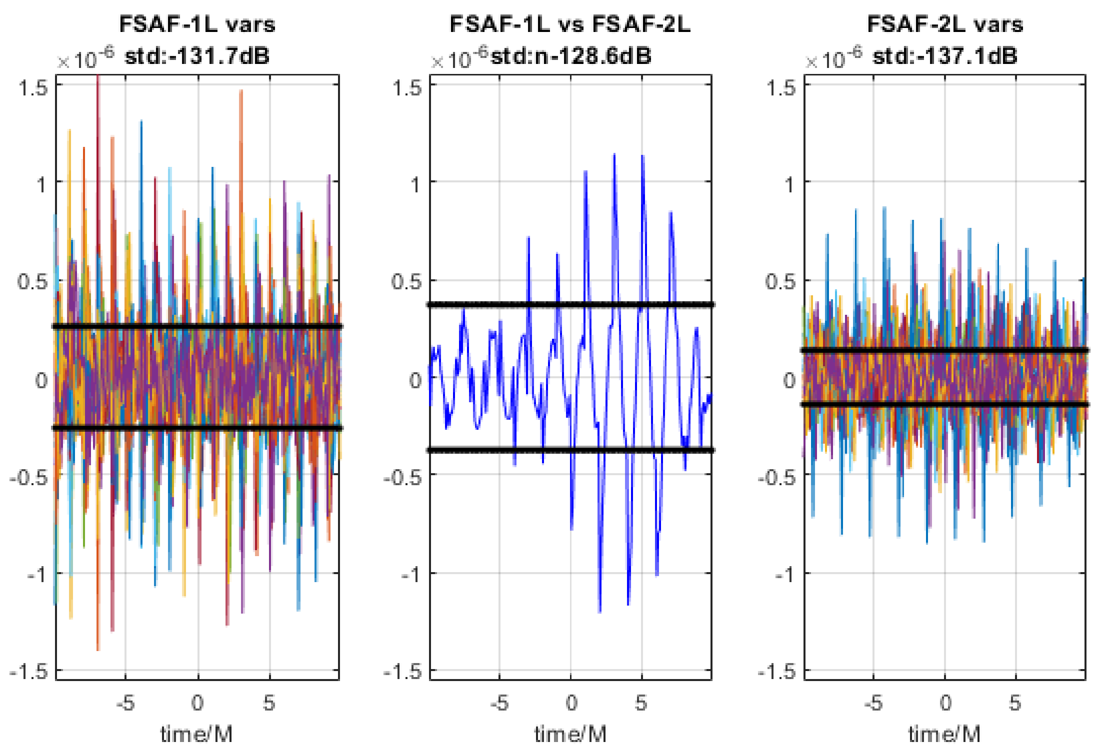


For SAF1988 designs, the same algorithm converges to quite different solutions depending on white noise realisations as excitation, and averages depend on the allowed spread width. We can never be sure to what SAF1988 converges and how it is related to theory.

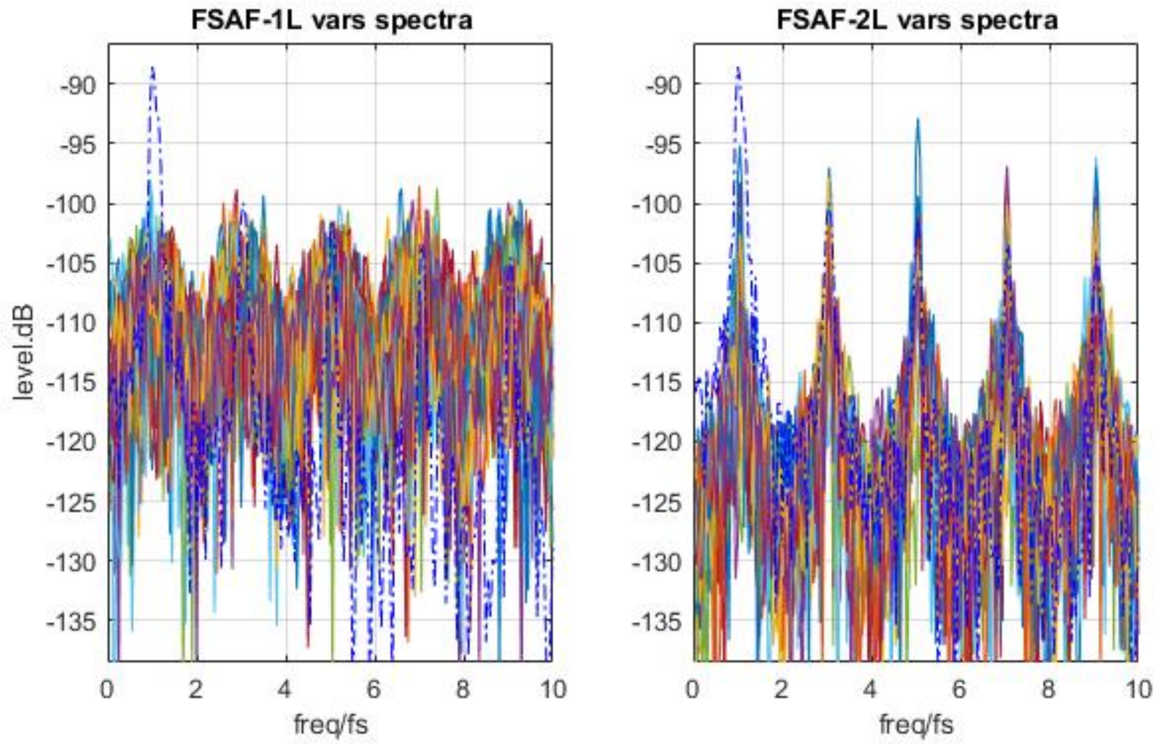


Frequency-wise, this robustness deficiency is centered at the band-edge (not *fs* but f_N), and thus replicated at odd multiples of f_N.

3.8.2 FSAF: noise + L_{RIR} [308]

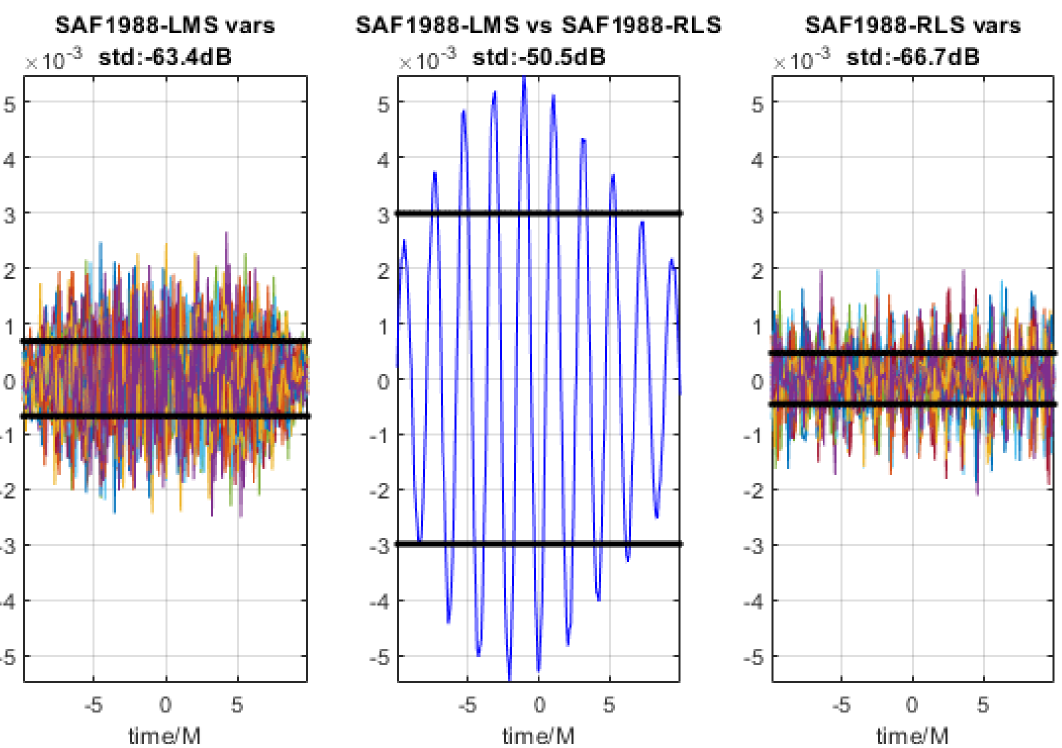


For FSAF based designs (like M=10, L=12, IL=3, EQ is on, EQOUT is off), the same deficiency exists (DSF spread is still infinite) but its impact is 60dB less. In other words, the FSAF's DSF is 1000 times more compact. For some applications, that may mean the impact is negligeable.

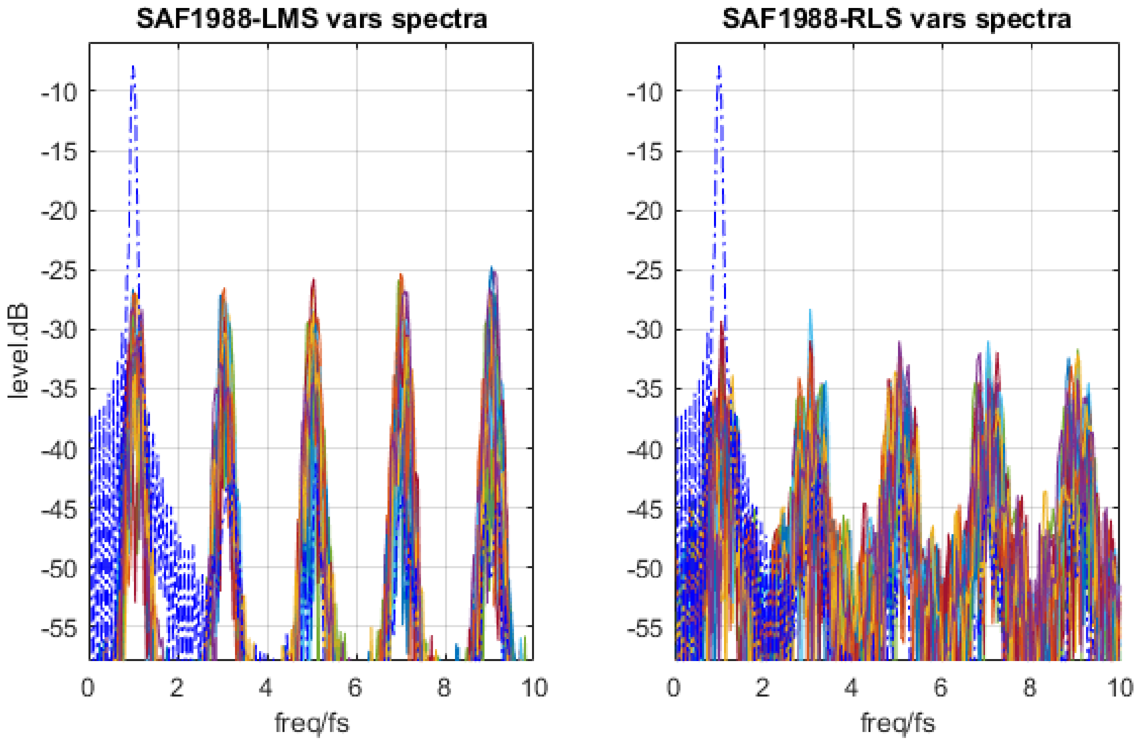


3.8.3 SAF1988: RLS vs LMS [310]

In a good IN/OUT codesign, any converging algorithm, like SA-LMS and hopefully RLS, shall converge to the same DSF.



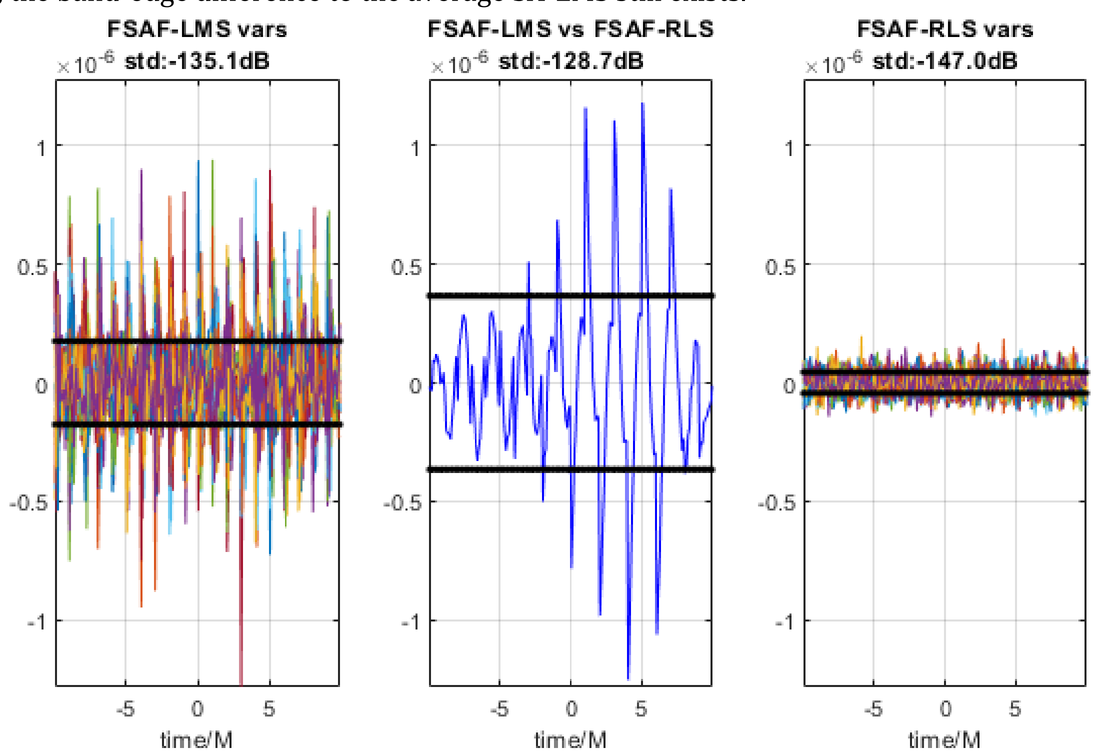
In SAF1988, that's certainly not the case. No SA-LMS nor RLS converges particularly well due to under-modelling.



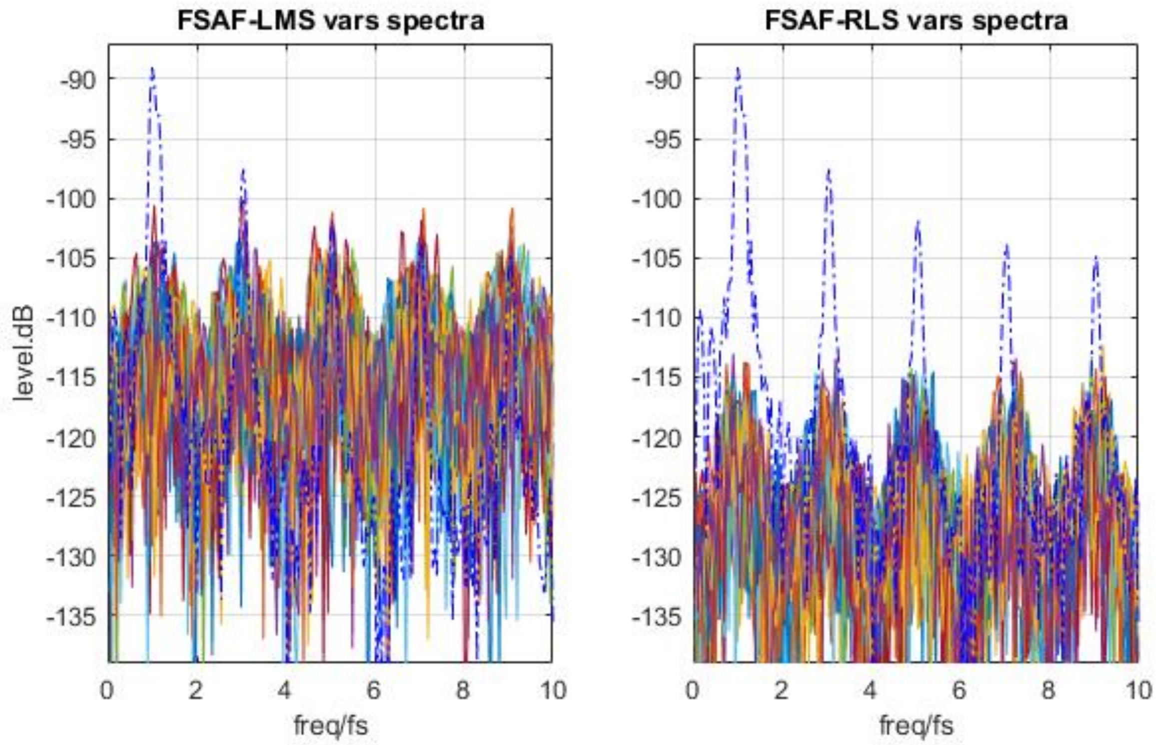
The spectrum of inter-algorithm difference (dashed blue) is concentrated on band-edge.

3.8.4 FSAF EQ: RLS vs LMS [310]

As before, FSAF vs SAF1988 gains about 60dB of fidelity. Additionally, here RLS shines with its precision. However, the band-edge difference to the average SA-LMS still exists.



In frequency domain, we may notice that SA-LMS with meaningful step control parameters is not too bad in comparison to the optimal RLS, and that the inter-algorithm difference shrinks from -8dB for SAF1988 to -88dB for FSAF:



Generally, while SAF1988 problems are evident to the naked eye, we need a 1000x microscope to notice analogous FSAF problems.

3.9 SUMMARY

The DSF has the same importance for understanding of subband adaptive filtering as the prototype function for perfect reconstruction filterbank. The omission of any notion of DSF in the initial proposal is as regretful as omissions of D_0, discussed in the Part II, which is equivalent to throwing the baby out with the bath water.

4 RIR RECONSTRUCTION: OPEN LOOP DELAYLESS (OLD) FSAF

4.1 BASICS

Obviously, to convert a sub-band translation (of a full-band RIR) back to full-band, we need to find a DCF (Delta-function Compress Function) which is bi-orthogonal to already well-known DSF (Delta-function Spread Function), using exactly the same method as for analysis – synthesis filters.

Obviously, "true" perfect reconstruction is impossible due to infinite spread of DSF, so we are discussing only NPR here.

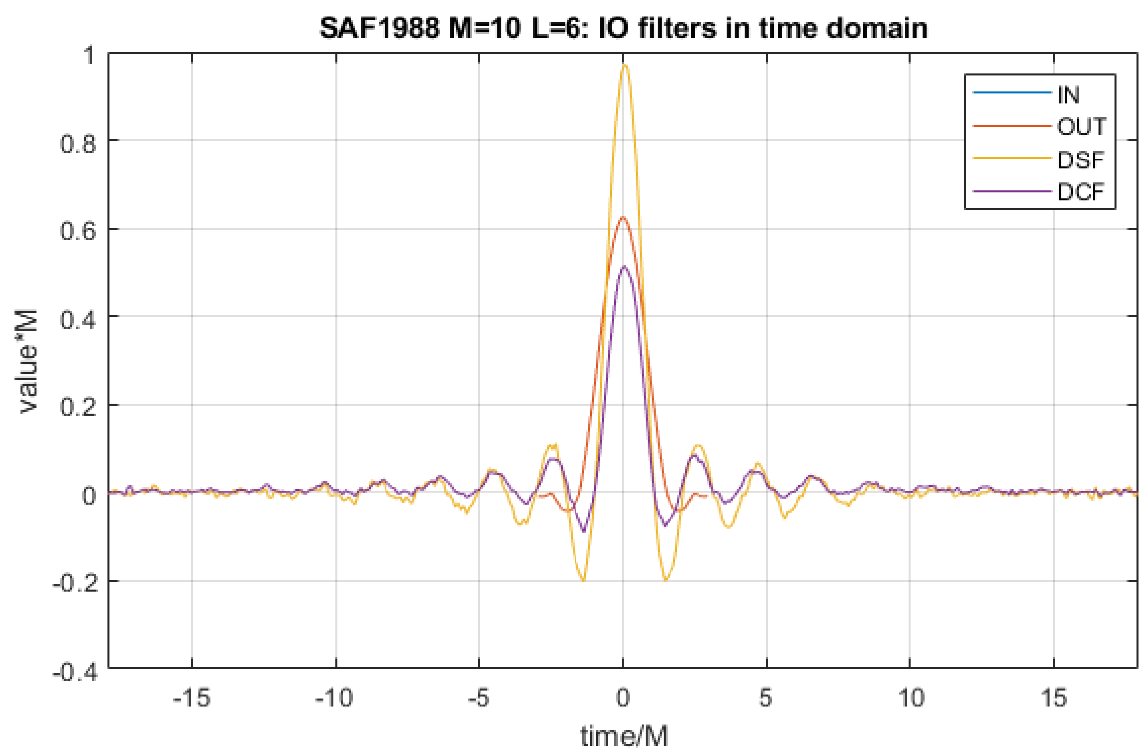
Everywhere below LDSF=36 frames (of M samples long).

4.2 OPEN LOOP DELAYLESS SAF1988 LMS

4.2.1 L=6, IL=0, BLKS=1000 [311]

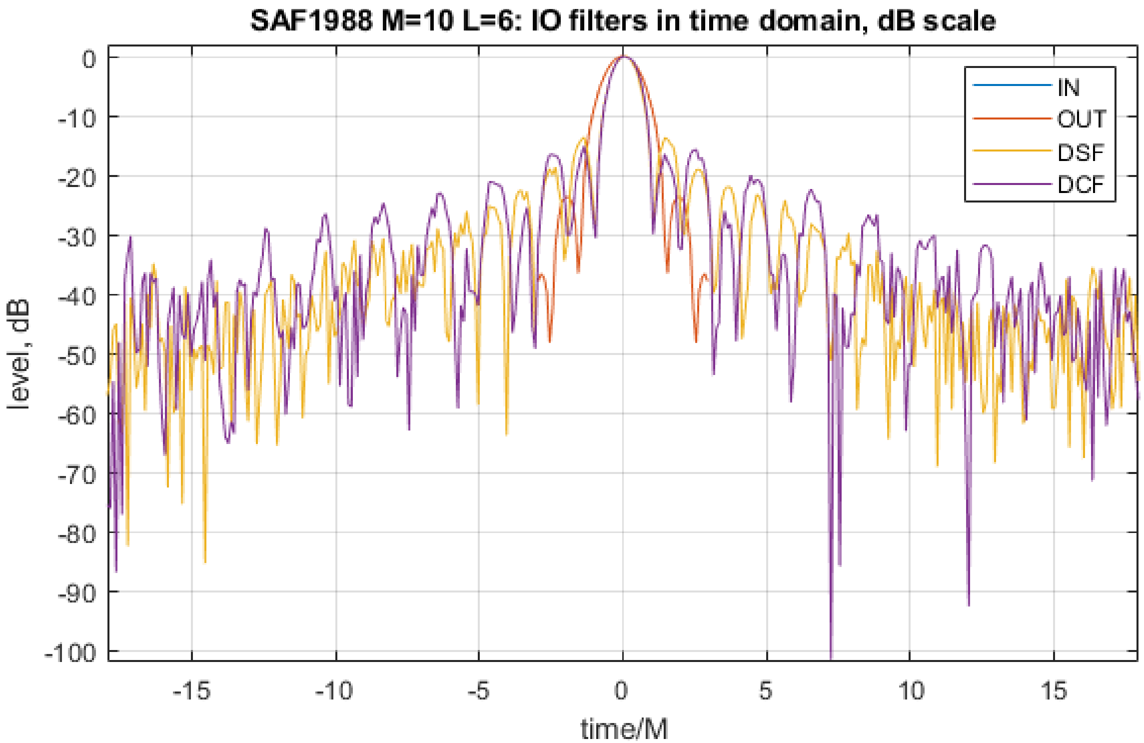
Obviously, DCF for SAF1988 is a [almost] the same *sinc(t)*, i.e., *DCF(t)=0.5*DSF(t)=0.5*sinc(t/M)*. The existing alternative approaches which have been published by various authors since 1995 are either convoluted ways to compute *0.5*sinc(t/M)* or wrong[3]. DSF for SAF1988 is not compact, so the DCF for *sinc(t)* shall be also quite long to provide a reasonably low reconstruction error. As shown above, we'd also have to use the same LMS to artificially ensure algorithmic robustness: an adaptive algorithm shall converge to the same DSF for each *Fs*-sample of full-band RIR.

Lots of time was given for LMS to converge: $1000*L_{ADF}$ frames.

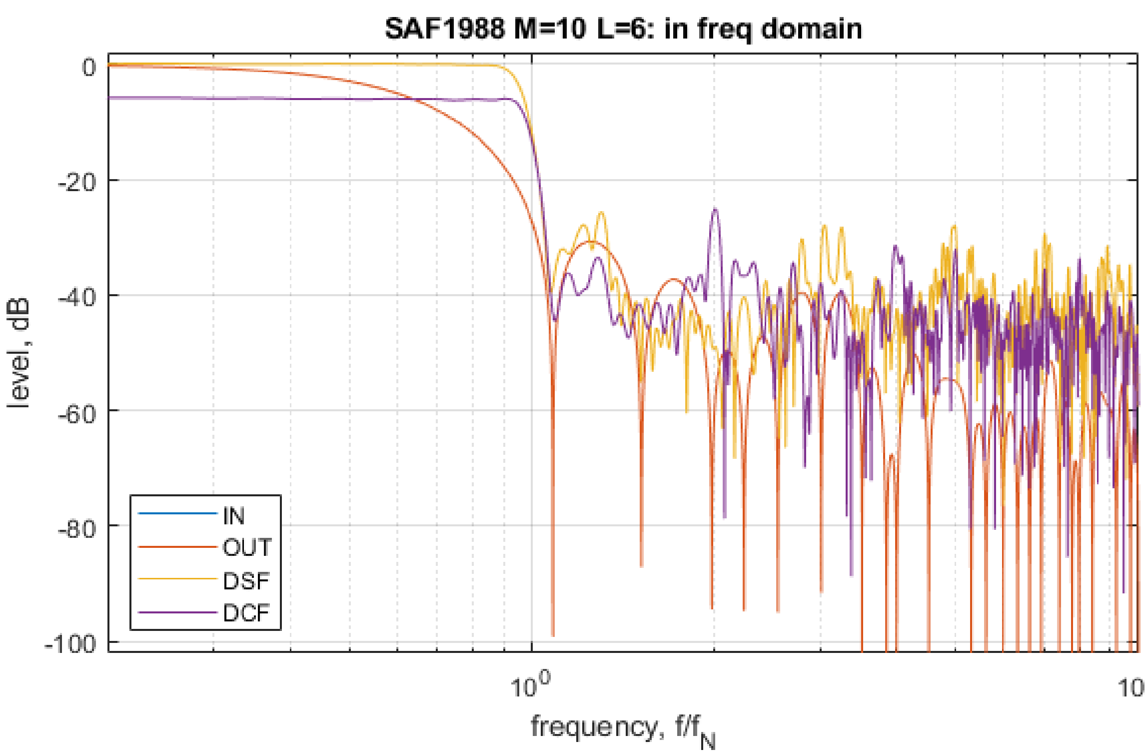


[3] Irrational belief in supernatural capabilities of written-down words and signs (whether the author does or does not have any idea of what s/he is doing) can be traced back to pre-dynastic Egypt.

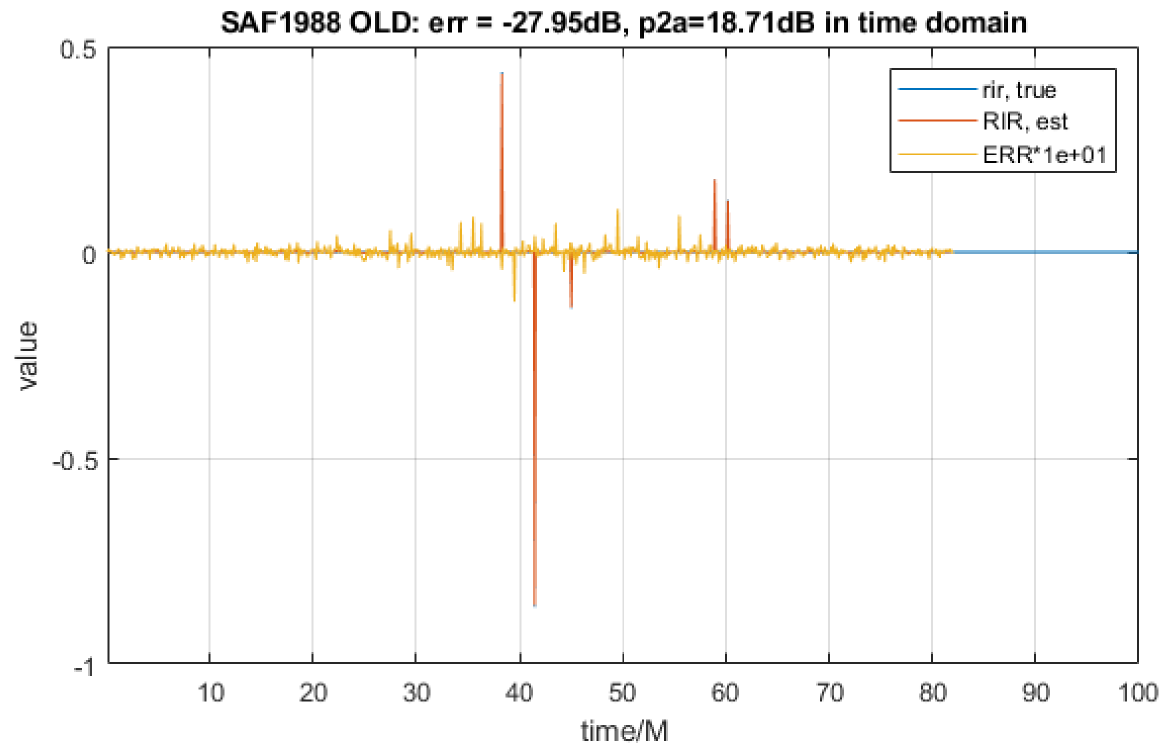
Here overall latency for rir1->DSF->DCF->rir2 was set to usual L_{DSF}-1so DCF's *sinc(t)* is shifted right by 1 frame. The same figure in dB scale:



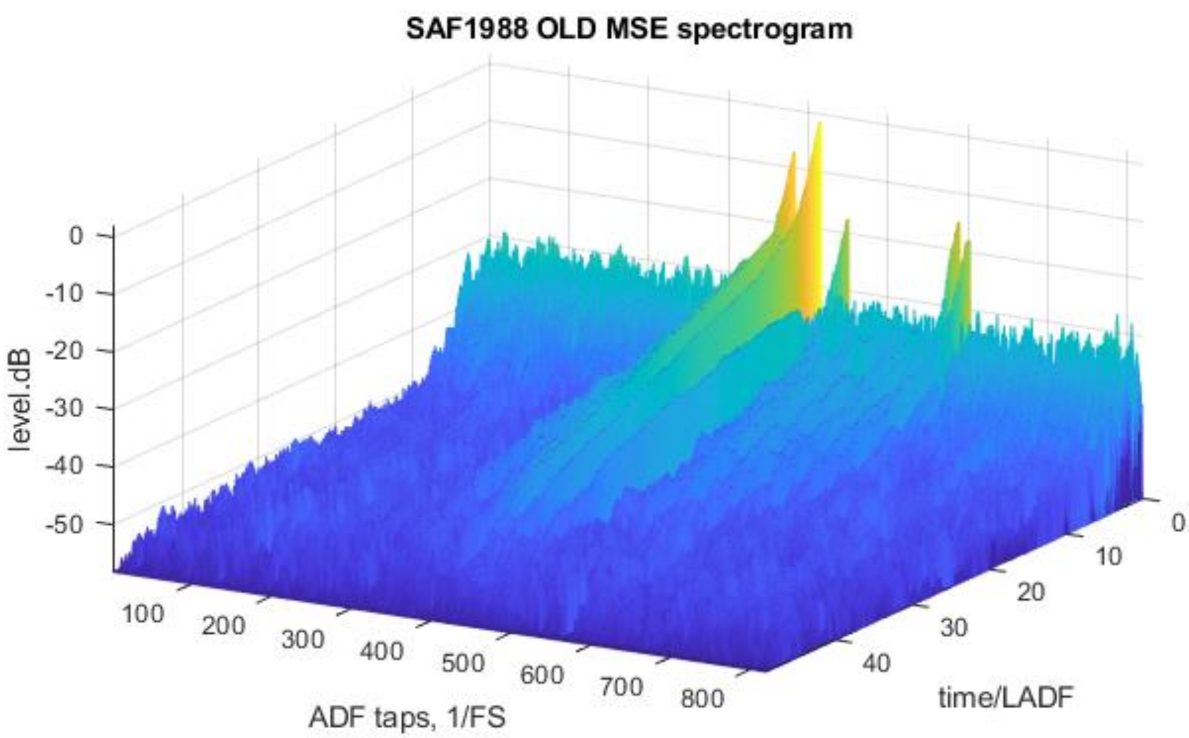
And the corresponding frequency domain:



The reconstruction error of just DSF->DCF, taken alone, is about -45dB. The error of RIR recovery is higher:



The convergence is expectedly slow due to the LMS spectral deficiency, as discussed in the Part II:



4.3 IN/OUT CODESIGN FOR OPEN LOOP DELAYLESS FSAF

It's assumed as self-evident that:

- The target for OLD Reconstruction Distortion (OLD-RD) shall be meaningful and account for real-world signal and noise levels and cadencies. There is very little sense to design a nearly perfect reconstruction OLD FSAF if in reality RIR varies quickly and wildly, noise is nigh and we can not get MSE less than $\sqrt{var(\hat{h}_t)}$ due to Cramer-Rao Bounds (CRB) being a pseudo-inverse of above-mentioned Fisher Matrix Φ_t over maximum quasi-stationary duration, no mater what approach we use.
- We need both DSF and DCF to be compact, so *fPass=0.5* for *OUT(f).*
- There is no sense in short IN/OUT filters if they produce less compact DSF/DCF.
- LD versions of Long FB would appear most promising for reconstructing the ever-changing RIR with minimal latency
- RLS with EQ (IN only, without OUT equalization) appear to be most promising for design and for run-time use.
- Frame size, *M*, is likely to be very large to lower overall MIPS, and per-subband L_{ADF} is short.

Thus, the major design parameters are:

- *L* = length of *IN(t), OUT(t)* filters, normalized to *M*.
- L_{DSF} = length of *DSF(t), DCF(t)* filters, normalized to *M.* Due to the reasons not understood, L must be an integer multiply of 4
- *EPS* = SVD threshold for solving DSF-> DCF equations.
- LD latency
- EQ's Max Gain
- RLS robustness parameters

The major design optimization criteria are:

- Reconstruction Distortion, average
- Reconstruction Distortion's peak to average ratio
- Reconstruction Latency due to IN/OUT/DSF/DCF length

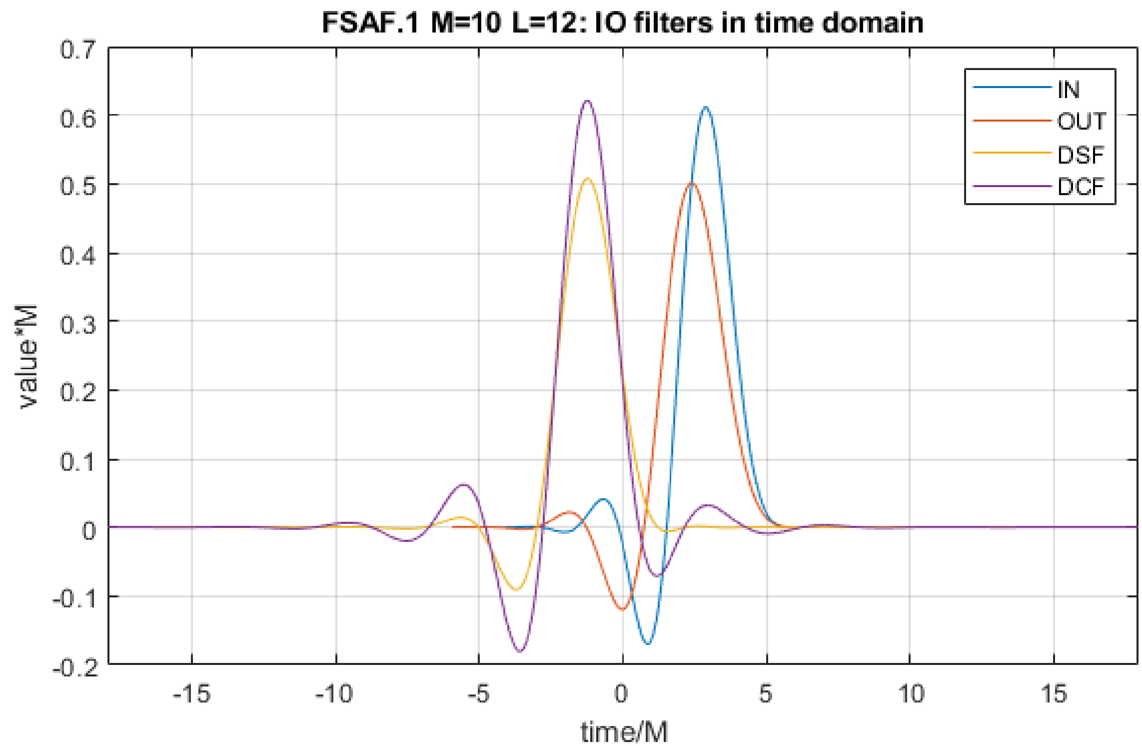
4.4 Open Loop Delayless RLS FSAF examples

In the examples below:

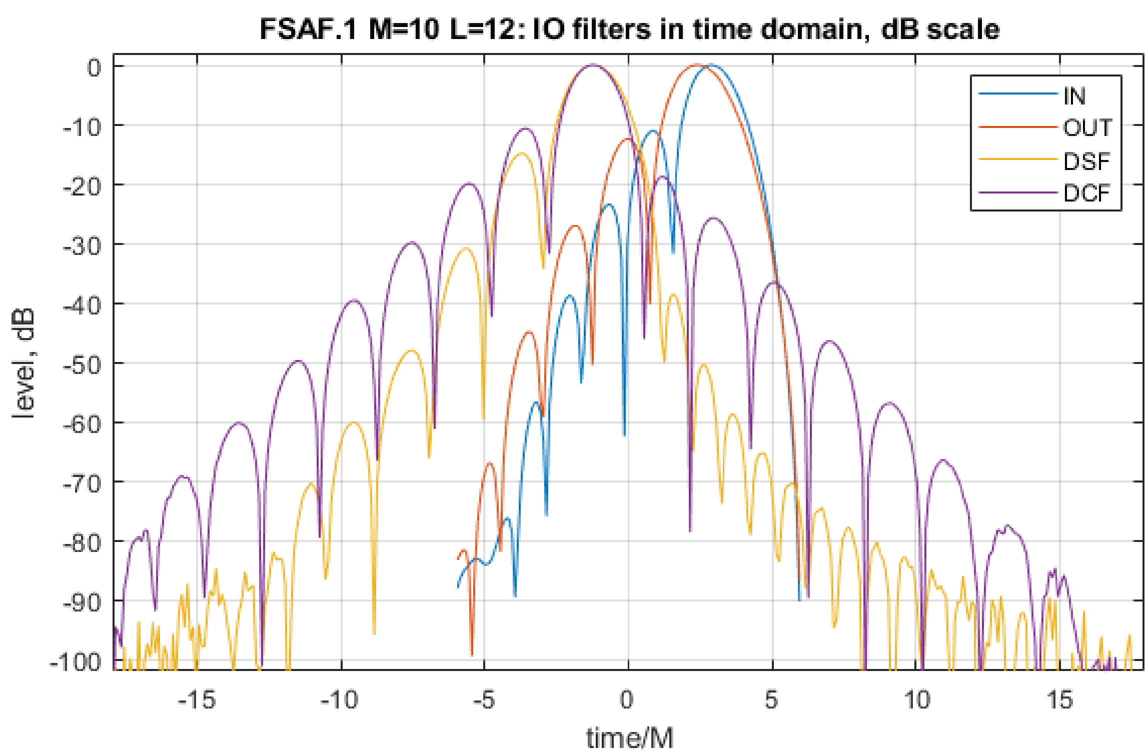
- *M =10*
- Test RIR is an *zeros(100*M,1)* vector with 5 pulses of random value (WGN) at random (uniform) locations
- SVD *EPS* is chosen manually 3e-4 … by subjective aesthetics.
- BLKS=41

4.4.1 L=12, IL=3 [311]

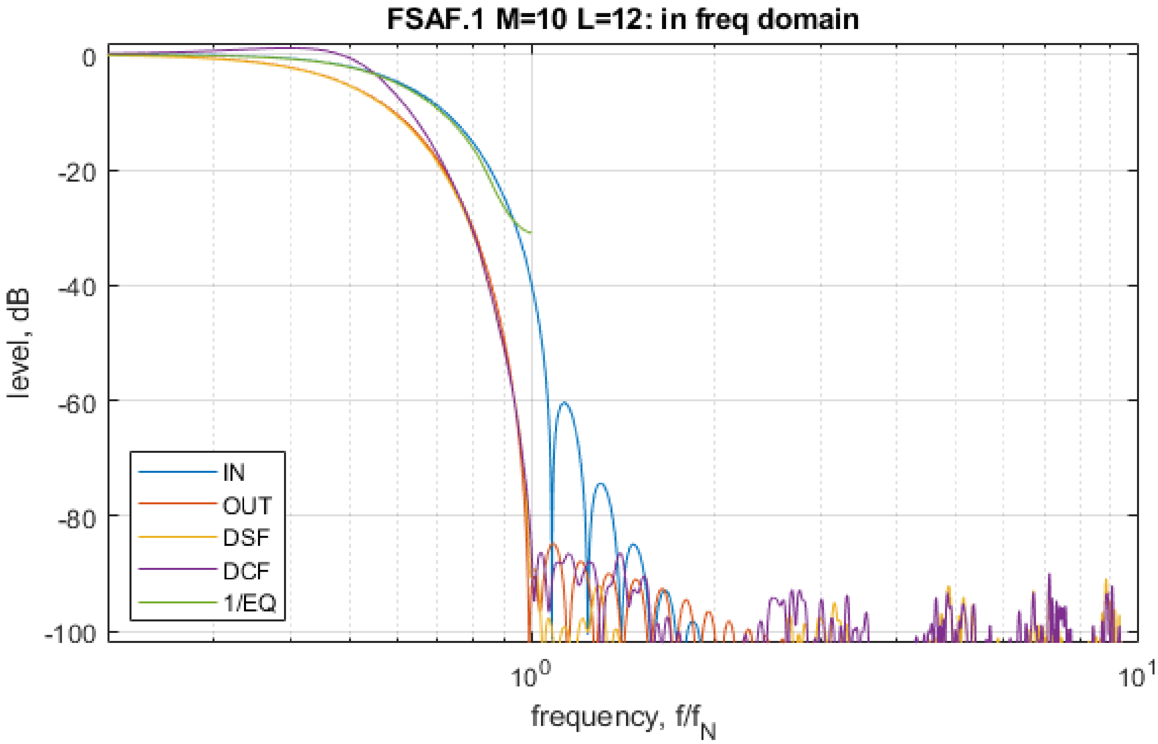
In this example we use an *OUT(f)* filter with *fPass=0.43*. We can see that DSF is significantly more compact than DCF, which is not what we want:



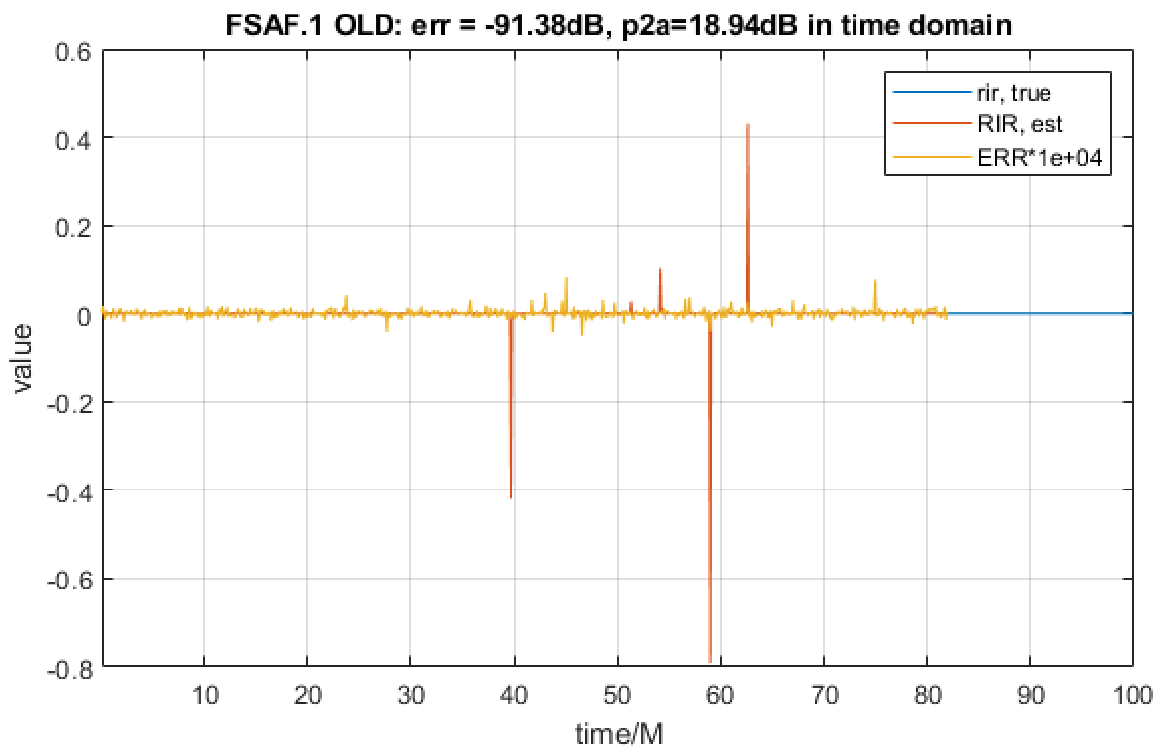
And



And

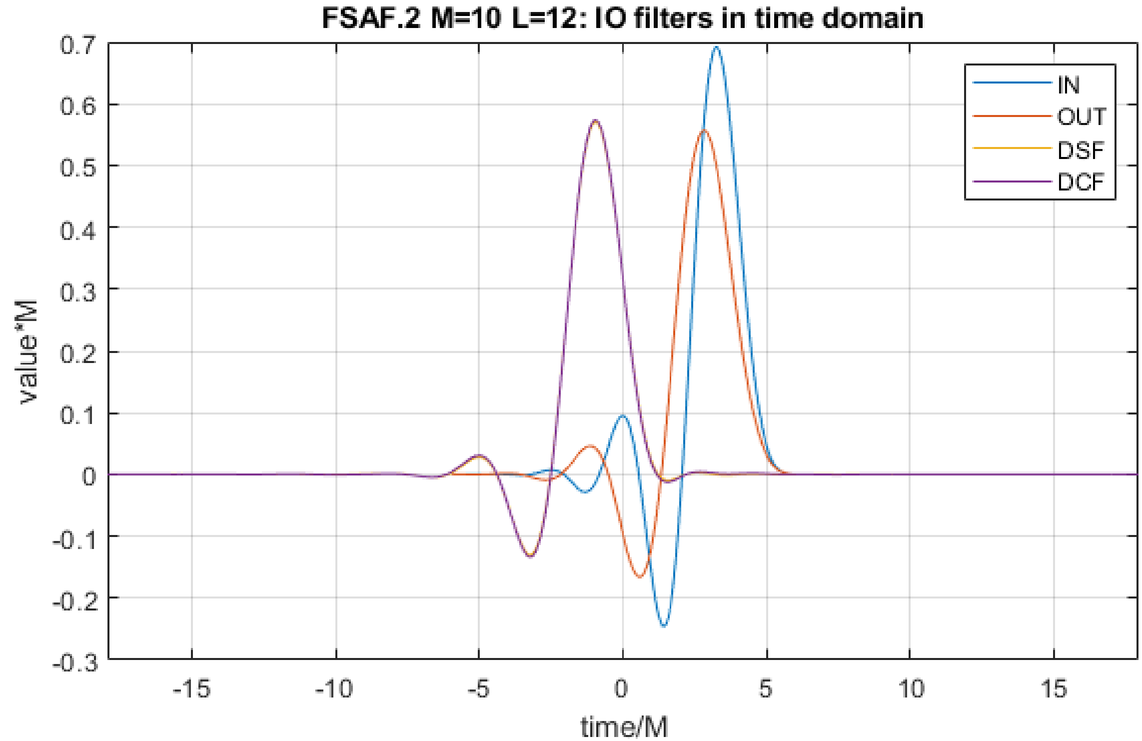


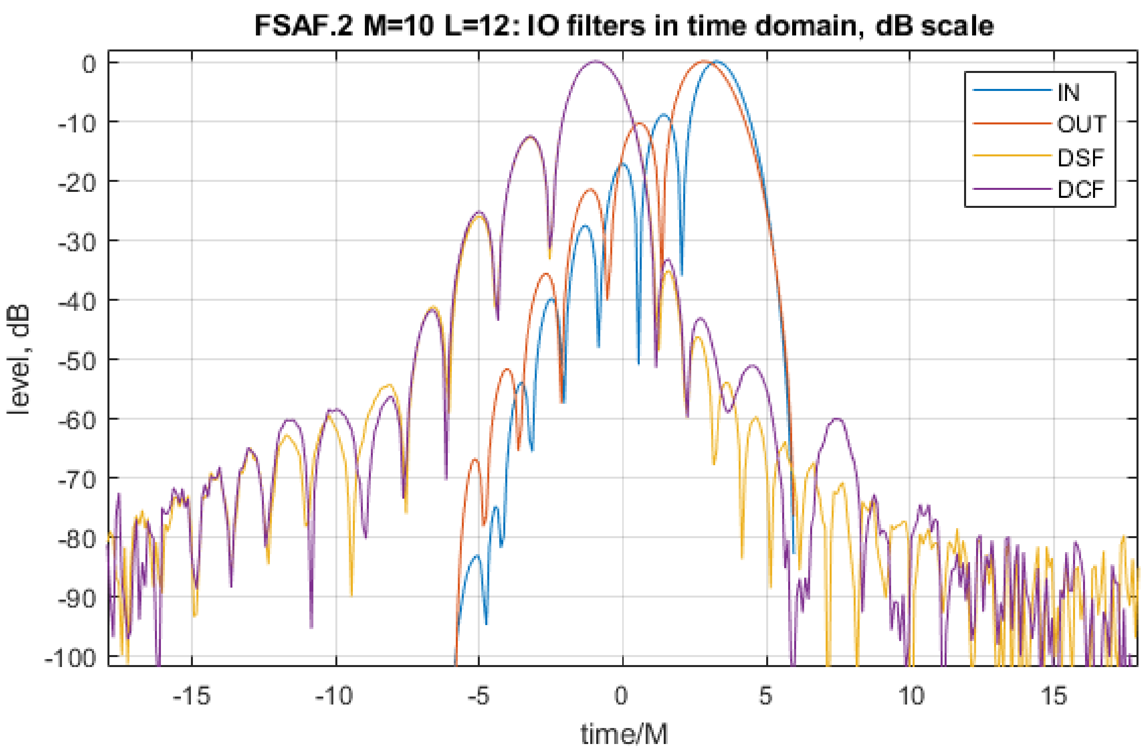
So, the RIR reconstruction resulting error (quite flat spectrum) is less than -90dB:



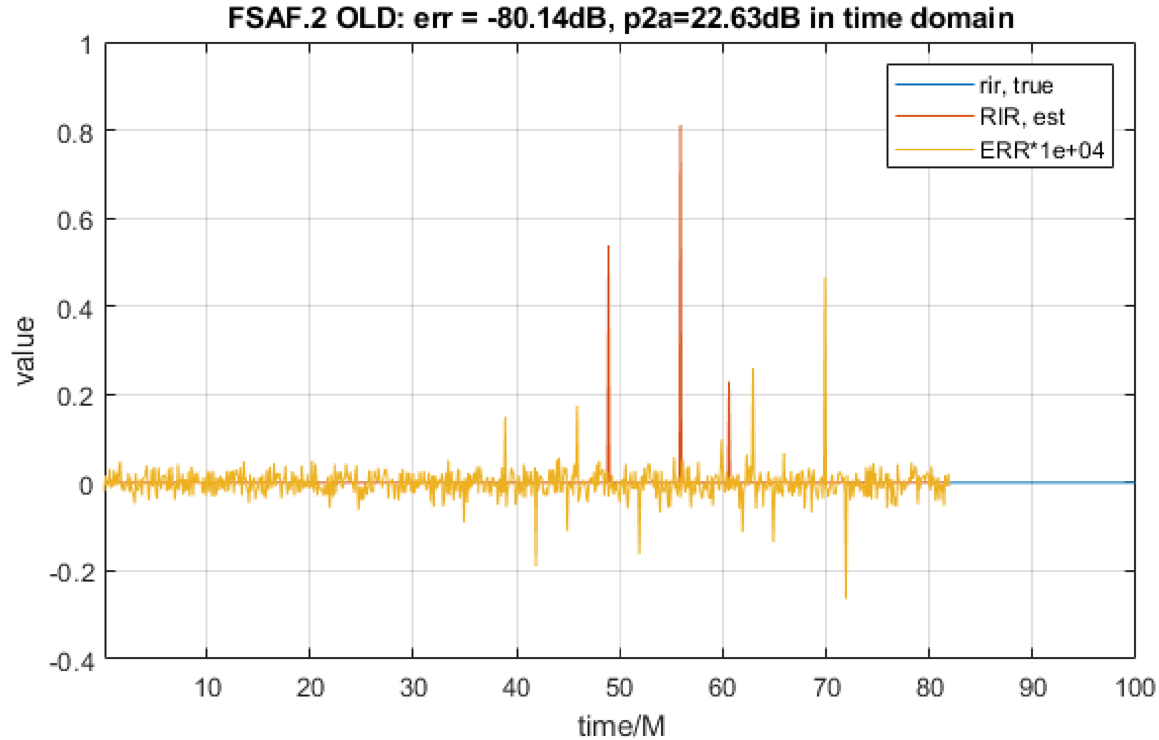
4.4.2 L=12, IL=4 [311]

When the OUT(f) filter is QMF-ish, i.e. fPass = 0.5, DCF looks as a mirror reflection of DSF (to make it more obvious, DSF is flipped) :



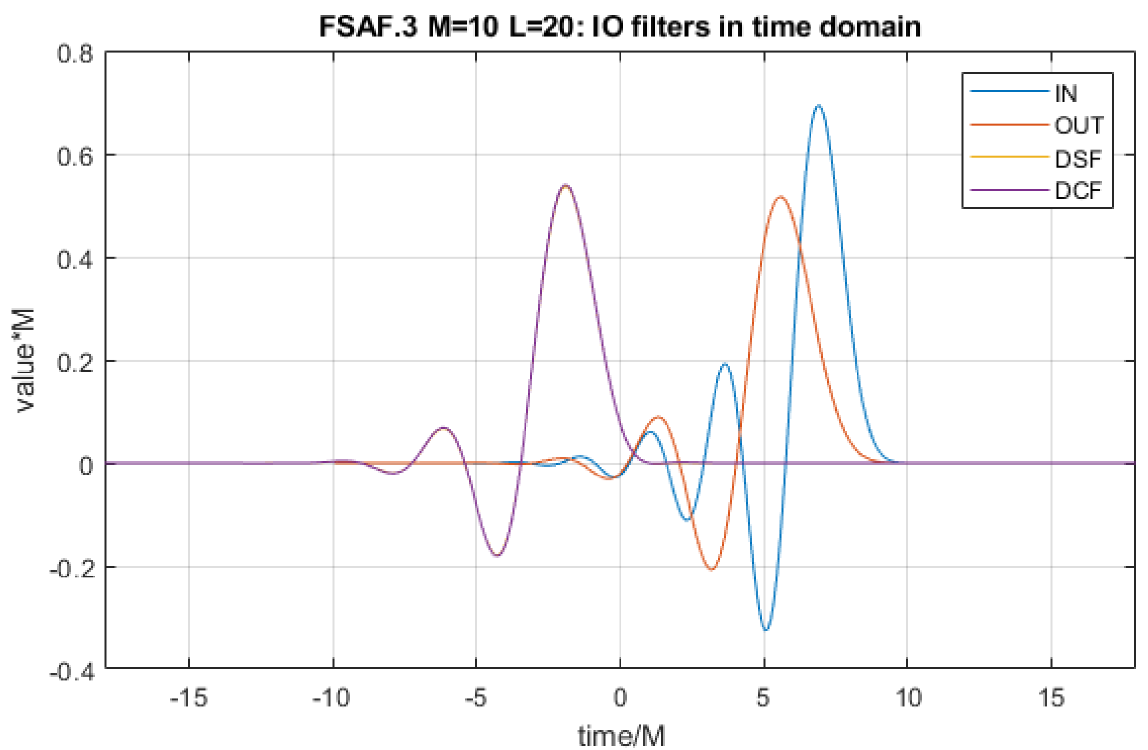


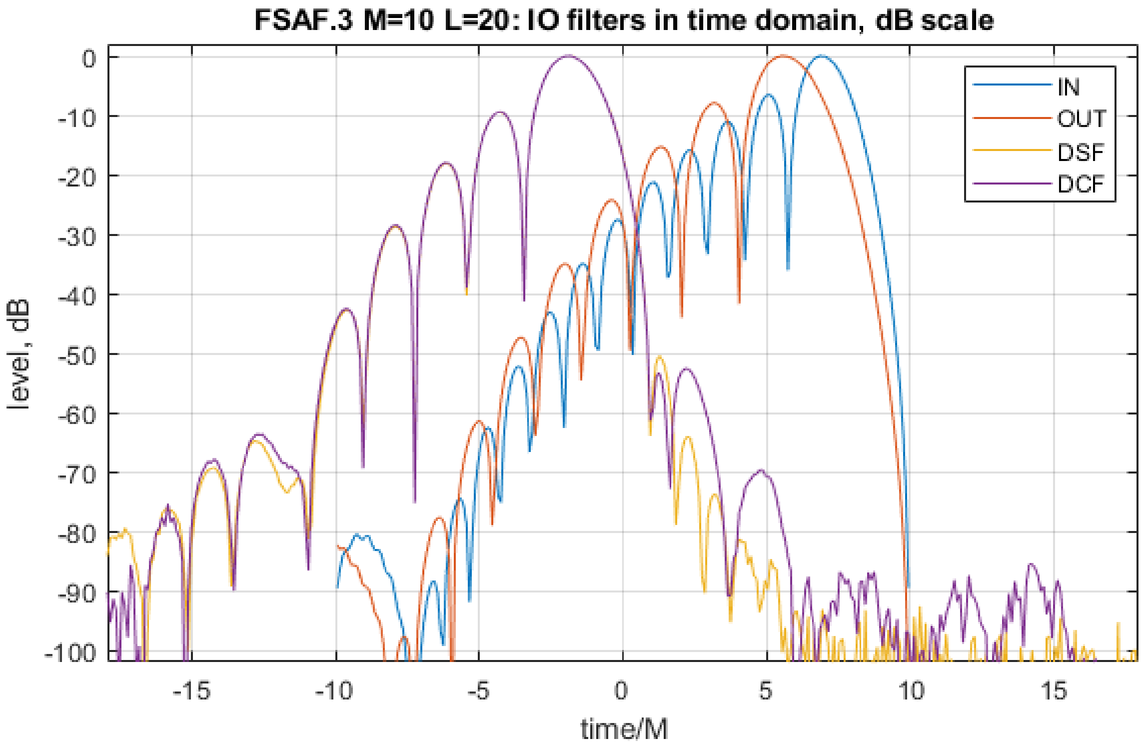
The RIR reconstruction results are slightly worse because a wider filter has higher sidelobes and more artifacts:



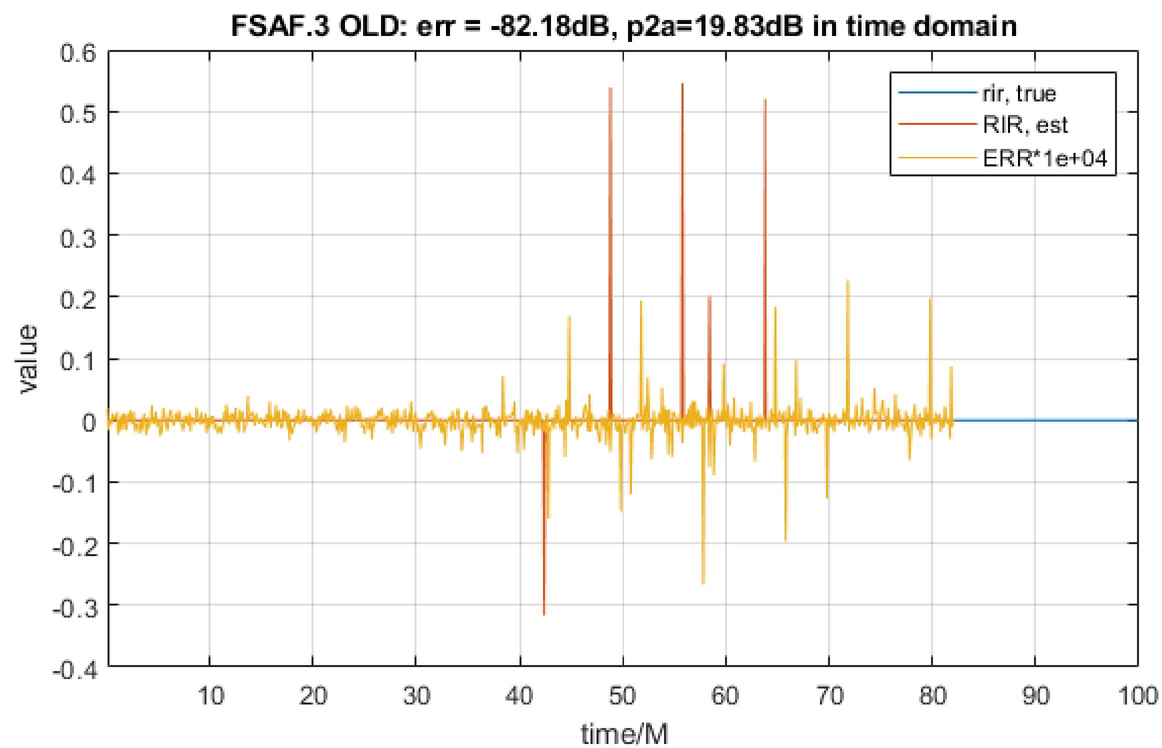
4.4.3 L=20, IL > 0 [311]

Using a longer, more QMF-ish filter with lower sidelobes as *OUT(f)* makes *DSF(t)* resemble *OUT(f)* closer, *DCF(t)* to mirror them *DSF(t),* and improves RIR reconstruction:

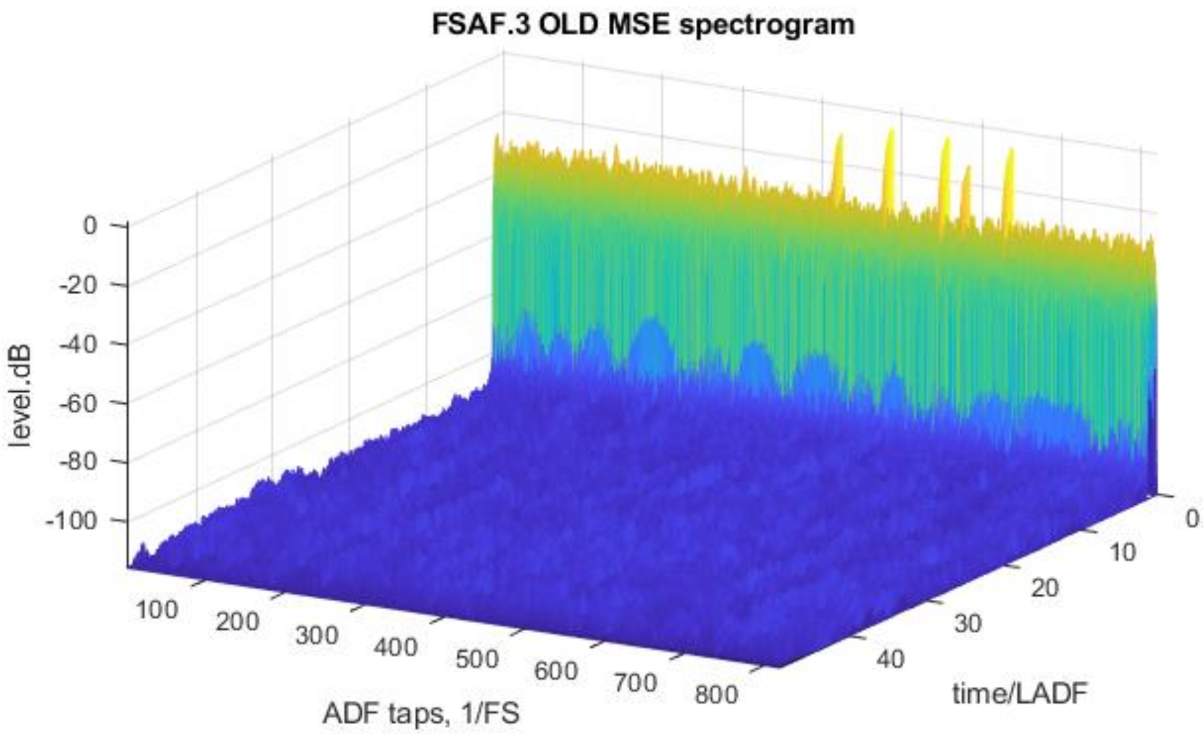




We notice that "look-ahead" delay is minimal, 2 frames only, in this design if the required precision does not exceed 50dB. The 2-frame delay already happens in the double-buffered IO, so no extra delay is required.



The process of convergence is fast, typical for RLS:

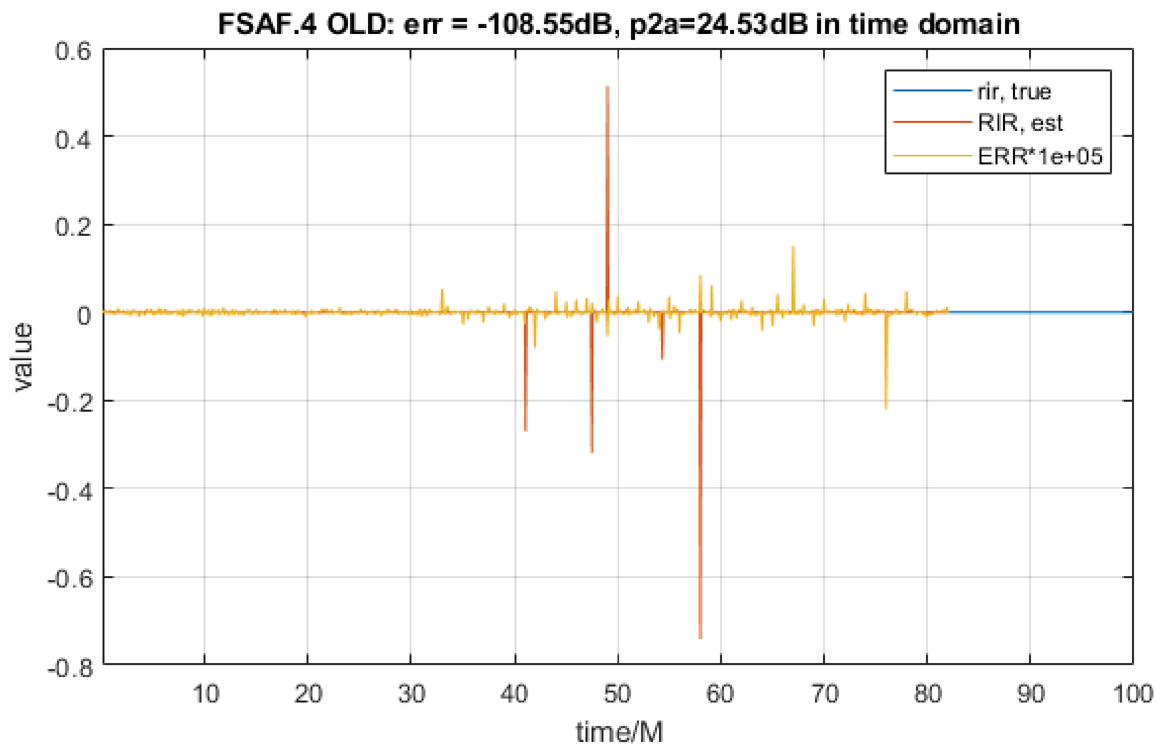


Essentially, RLS converges in a L_{ADF} samples. Adding time does not improve precision much because RLS convergence is limited by under-modelling artifacts, not by noise averaging.

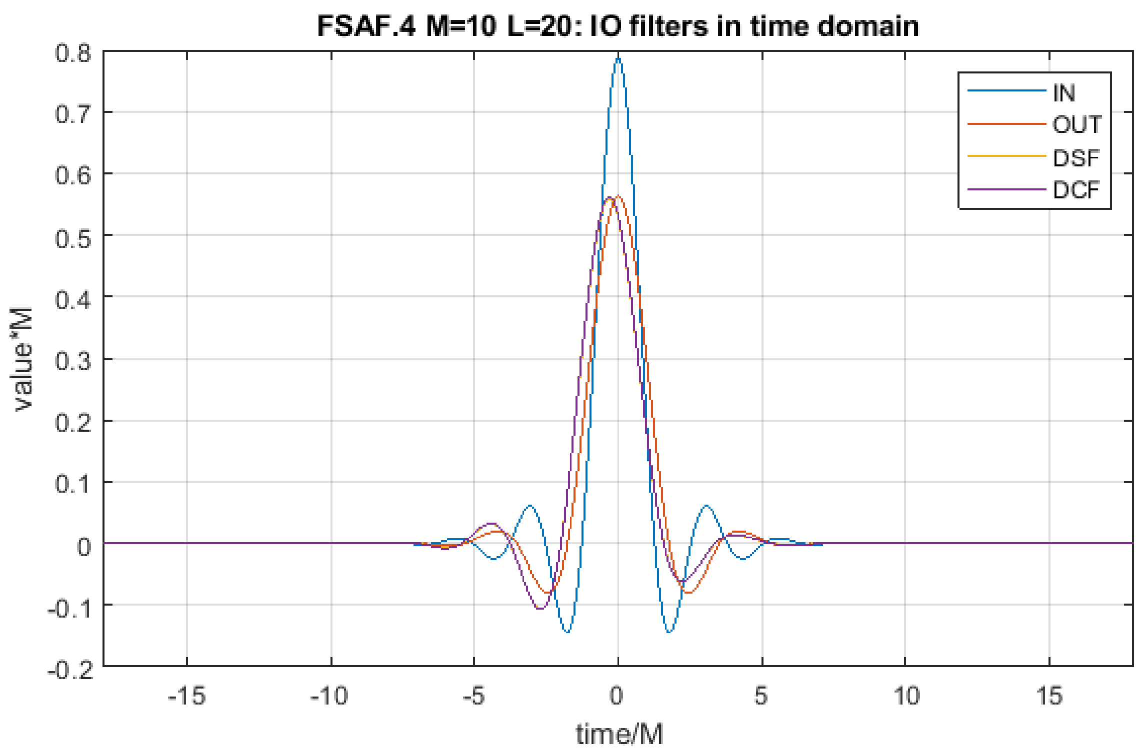
4.4.4 L=20, IL=L/2 [311]

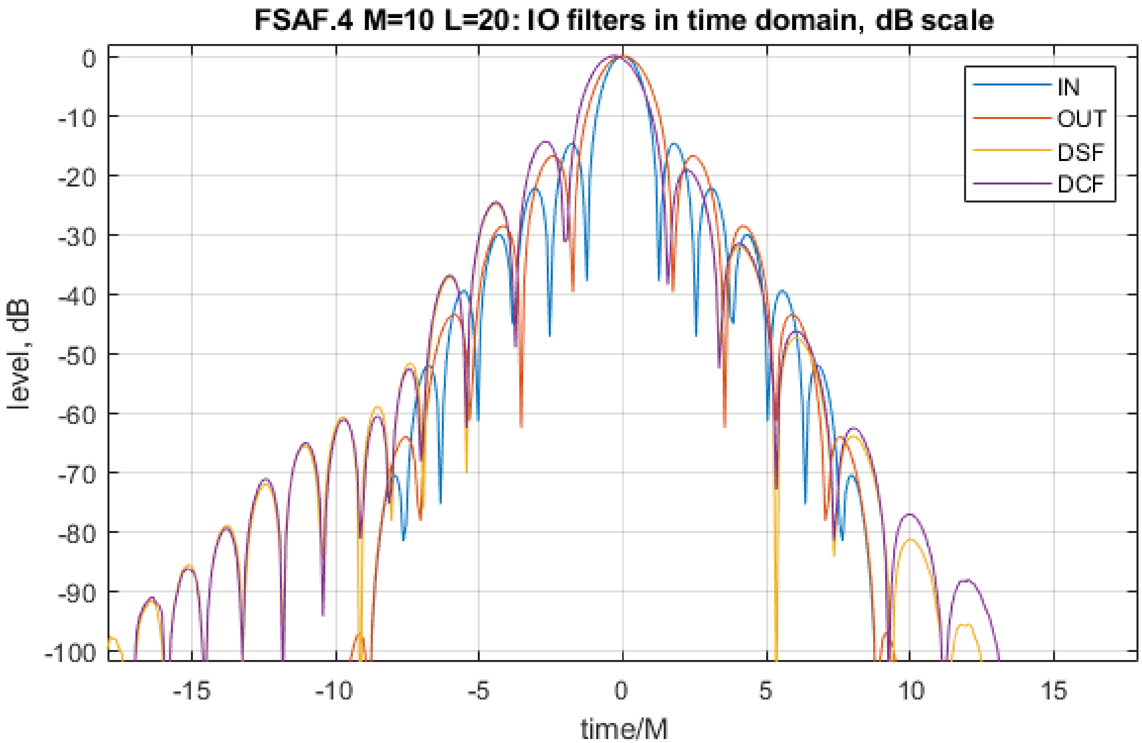
The quality of RIR reconstruction could be limited by the artifacts of imperfect reshaping of *IN(t) / OUT(t)* filters from symmetric into minimal phase form. If we keep the filters symmetric, we may achieve about

20…30dB of precision improvement:



The largest contributor to distortions is a spike of about 2e-6 relative amplitude following the originator $M{*}L_{DSF}/2$ samples later. The filters:



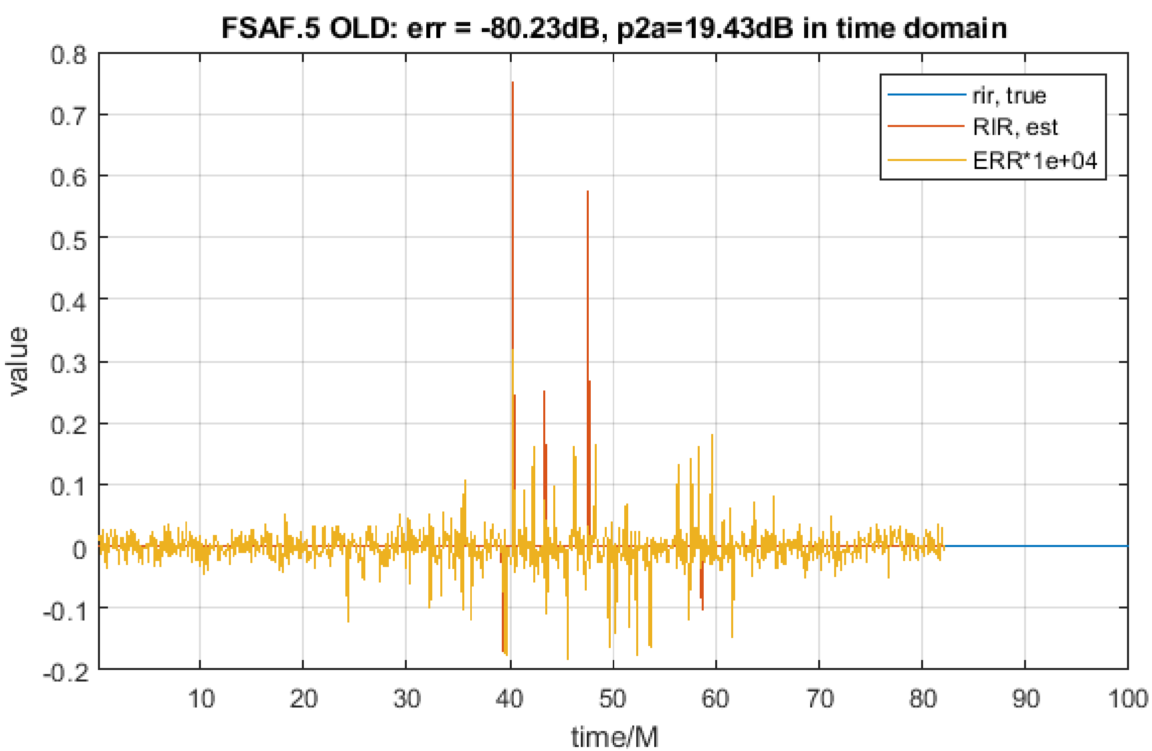


4.5 Open Loop Delayless SA-LMS FSAF examples

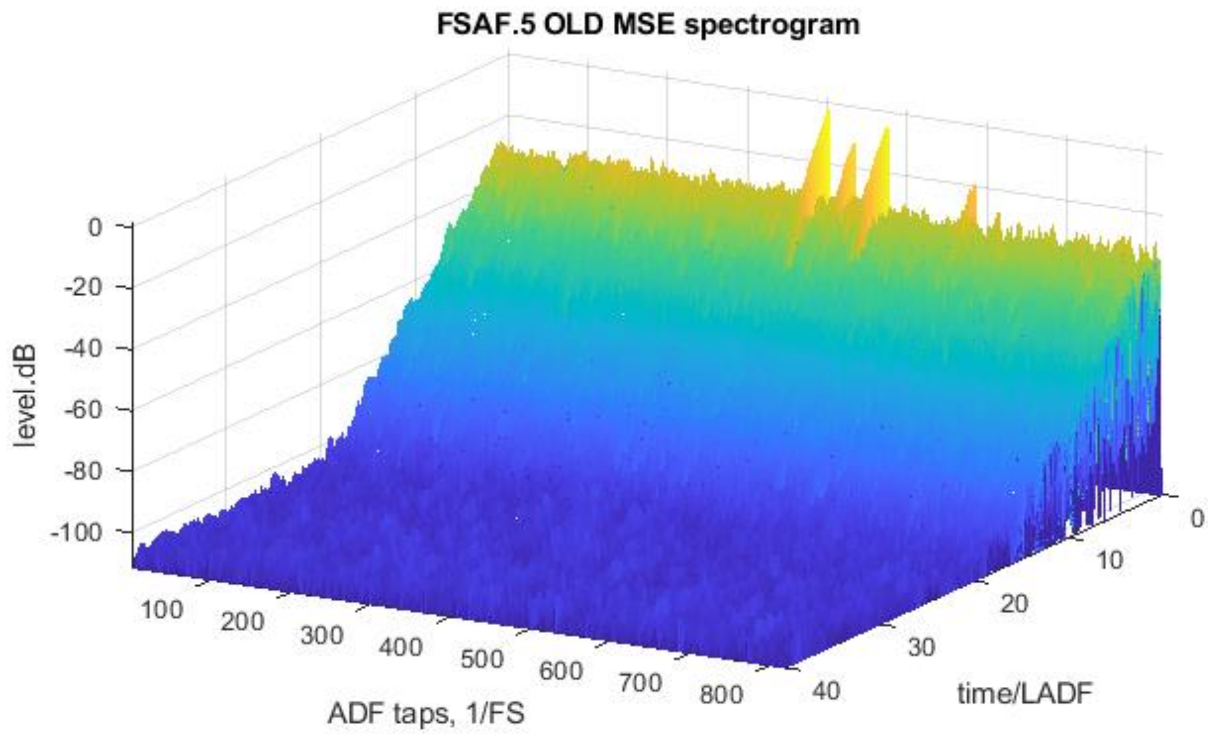
Same LDSF=36, BLKS=41 below.

4.5.1 L=24, IL>0 [311]

The SA-LMS filter is absolutely useful if subbands are so narrow that the signal inside becomes sufficiently white. SA-LMS can converge to about the same precision:

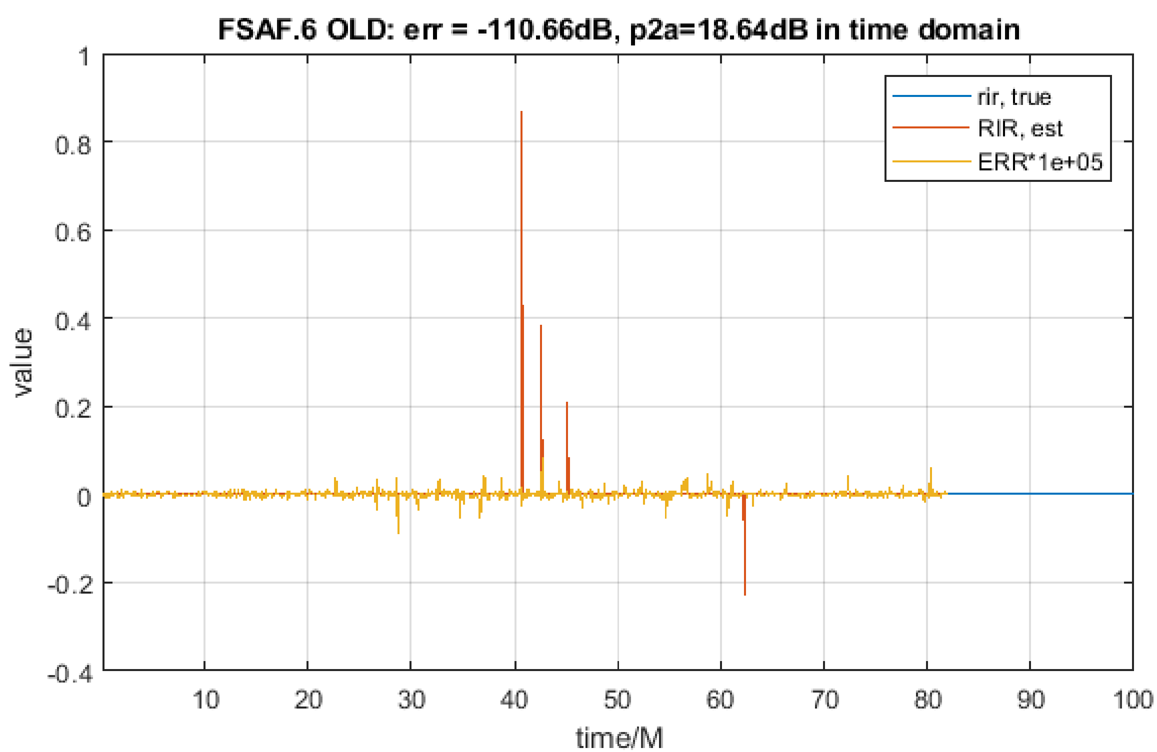


…but it takes much longer:

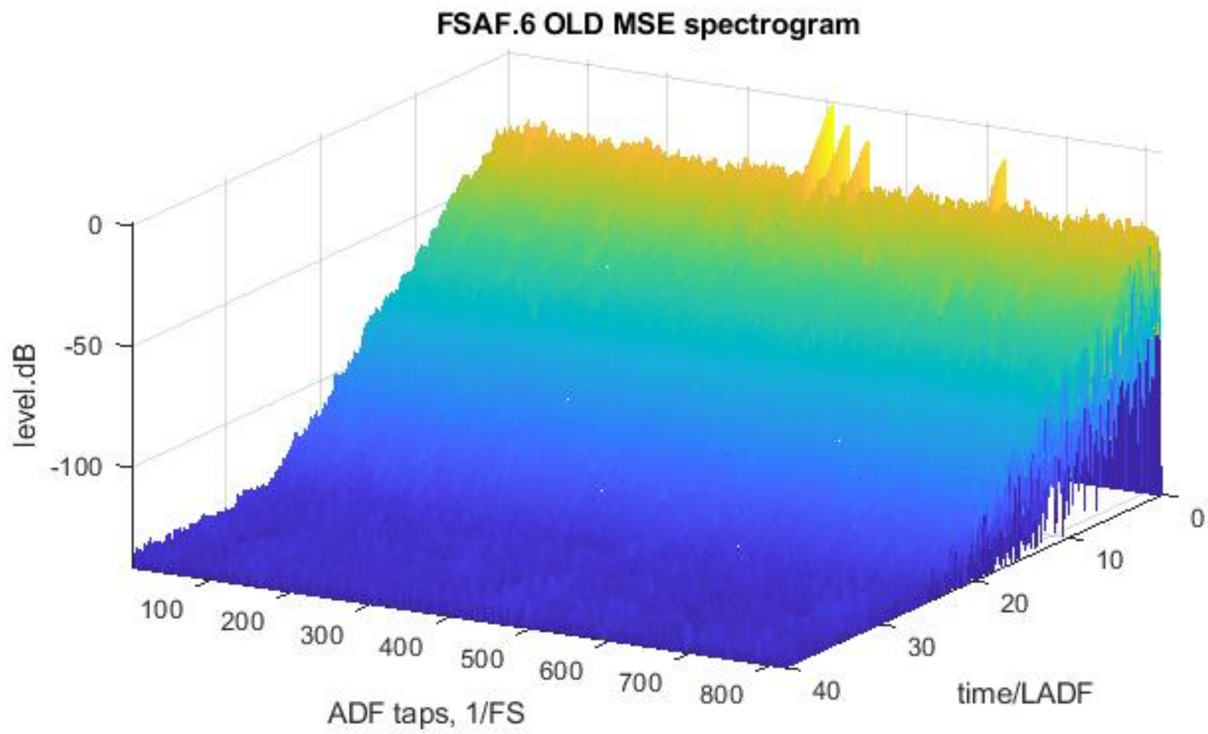


4.5.2 L=24, IL=L/2 [311]

A better pair of symmetric *IN(t) / OUT(t)* filters with fast sidelobe decay and, therefore, lower aliasing level also improves convergence of SA-LMS:



… and, as usual, convergence of LMS is significantly slower:



It's far from clear why RIR reconstruction errors are so uneven, with M-periodic repeated spikes but without (almost) usual white gaussian noise.

4.6 WHY F[AST]SAF?

Thus, we see that FSAF resembles FFT in the following meaning:

- Instead of identifying a very long LTI system with L_{RIR} long impulse response:
- Using *M*-band DFT-based multirate filterbank with *R=2* oversampling over the critical
- An appropriately equalized appropriately regularized subband adaptive algorithm (LMS, xDLS, RLS)
- With length of $L_{ADF} = (L_{RIR} + o(1/\delta))/M$
- We can identify this LTI in subbands using
  - $\sim 1/M$ (for sub-band LMS vs full-band LMS), or
  - $\sim 1/M^2$ (for sub-band RLS vs full-band RLS), or
  - $\sim 1/M^3$ (for sub-band DLS vs full-band RLS [because of pre-whitening effect of narrow-band FSAF])

    less MIPS in real time.
- and convert the result back to full-band with reconstruction error < δ
- with essentially the same convergence speed as a full-band BLUE or even faster, as Best Linear Biased Estimator.

4.7 SUMMARY

The Open Loop Delayless SAF requires good understanding of DSF.

✓ Remember that even if the FSAF adaptation is MIPS-savvy, the full-band sample-based filtering (non-adaptive, just filtering) may require near-infinite MIPS.

5 CONCLUSIONS

As we have already discussed, the main improvements of sub-band adaptive filtering over full-band adaptive filtering:

- Significantly lower MIPS, *~1/M* for LMS vs LMS, *~1/M²* for RLS vs RLS, and *~1/M³* for DLS vs RLS
- Improved ability to operate on excitation with poorly defined spectrum, improving ~M

…come, mainly, on the cost of

- Algorithmic latency, *~M*L*
- Even for LD versions, *~M*(5* + non-casual DSF spread compensation) latency
- Higher memory usage (opposite for subband RLS vs full-band RLS).

The increased algorithmic latency is often prohibitive for real-time applications. Are there any ways to work around it?

5.1 NESTED FSAF

One approach to deal with latency could be to nest FSAF:

- Create an FSAF with Long LD FB, and moderate M – number of subband so that the overall algorithmic latency conforms to the functional requirements for the application
- Inside each m-th subband, no adaptative algorithms are used to estimate *RIR(:,m).*
- Inside each m-th subband, the corresponding *RIR(:,m)* estimate is produced by a nested OLD FSAF which operates on thinly sliced narrow subbands where whiteness is [best] ensured and MIPS are so low that we an use RLS (if we have enough memory).
- Thus, the delayed response of the internal nested FSAF will be compensated by the vastly increased convergence speed of RLS.

The problems are, of course, with additional R*R=4 oversampling, high memory usage, duplication of adaptation in overlapping internal subbands, etc.

5.2 SPLIT-FUNCTION FSAF

Another approach to deal with latency could be to split adaptation and processing functions of FSAF:

- Create an OLD FSAF, RLS based, with Long LD FB, and as high M (number of subbands) as we can allow, to the point where excitation's whiteness is ensured and MIPS are as low as it gets.
- Create a processing FSAF with low M, so that algorithmic latency conforms to the functional requirements for the application.
- Add a translation layer which, using DSF, splits full-band *RIR(:)* into processing sub-band translations. May be, it could be optimized to bypass full-band RIR presentation.
- The additional operations of processing FSAF (noise reductions, gating, NLP) can be guided (or aimed) by weighted guidance from the OLD FSAF.
- A processing FSAF may be replaced by other means of fast low-latency convolution.

The ways of arranging FSAFs are bound only by algorithm developer's fantasy.

5.3 SUMMARY

The FSAF is a tightly interwoven knot of techniques from different domains which works in the real world, with complicated systems, with very long IRs.

You may have noticed that there was no magic formula nor any "optimal" solution nor any guaranteed safe/ robust design proposed. All these terms belong to the Program Control domain, which is not applicable to FSAF, which is imperfect (even in theory), overly complicated, robust and precise. FSAF approach itself is based on multi-level feedback loops, controllability, observability and testability i.e., it's adaptive.