A Lattice Basis Reduction Approach for the Design of Quantized FIR Filters

Nicolas Brisebarre, Silviu-Ioan Filip, Guillaume Hanrot

▶ **To cite this version:**

Nicolas Brisebarre, Silviu-Ioan Filip, Guillaume Hanrot. A Lattice Basis Reduction Approach for the Design of Quantized FIR Filters. 2016. hal-01308801v1

HAL Id: hal-01308801

https://inria.hal.science/hal-01308801v1

Preprint submitted on 28 Apr 2016 (v1), last revised 22 Feb 2018 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Lattice Basis Reduction Approach for the Design of Quantized FIR Filters

Nicolas Brisebarre, Silviu-Ioan Filip and Guillaume Hanrot

***Abstract*—Many applications of finite impulse response (FIR) digital filters impose strict format constraints for the filter coefficients. Such requirements increase the complexity of determining optimal designs for the problem at hand, making exact approaches intractable even for moderately high filter degrees. We introduce a fast and efficient heuristic, which is based on the computation of good nodes for polynomial interpolation and Euclidean lattice basis reduction. Most of the time, it returns quantized FIR filters of extremely good quality.**

***Index Terms*—BKZ algorithm, Euclidean Lattice, finite wordlength, FIR coefficient quantization, FIR digital filters, HKZ algorithm, lattice basis reduction, LLL algorithm, minimax approximation**

I. Introduction

THE efficient design of FIR filters has been an active research topic in Digital Signal Processing (DSP) for several decades now. The problems that are most prevalent in practice are related to the discrete nature of the representation space used for storing the filter coefficients (usually fixed-point or floating-point formats).

In general, an initial design procedure like the well-known Parks-McClellan [1] formulation of the Remez exchange algorithm [2] is applied in order to retrieve a set of "infinite precision" coefficients, which are frequently computed as 64-bit floating-point values. When, for example, FIR filters are used inside specialized DSP processors or field-programmable gate arrays (FPGAs), the coefficient constraints are very strict, usually requiring the use of fixed-point values of much smaller bit size. Even very recent FPGAs that come with floating-point DSP blocks [3] only offer support for 32-bit floating-point arithmetic. Simply rounding the initial 64-bit precision coefficients to their nearest representations in these imposed formats can give rise to suboptimal filters. In many cases, the quality of these naive results can be significantly improved, especially when fixed-point values are used.

As a result, the literature on the FIR quantization problem is quite vast. Typical constraints are that the coefficients be representable as signed b-bit integer values or that they have low complexity, such as them being sums of only a small number of power-of-two terms. This second requirement is especially important when one wants to construct low complexity multiplierless filtering circuits.

N. Brisebarre is with CNRS, Laboratoire LIP, ÉNS Lyon, INRIA, Université Claude Bernard Lyon 1, Lyon, France.
E-mail: nicolas.brisebarre@ens-lyon.fr
S.-I. Filip and G. Hanrot are with ÉNS Lyon, Laboratoire LIP, CNRS, INRIA, Université Claude Bernard Lyon 1, Lyon, France.
E-mail: silviuioan.filip@ens-lyon.fr, guillaume.hanrot@ens-lyon.fr

In both cases, finding a set of *optimal* quantized coefficients proves to be computationally expensive for high degree filters (larger than, say 60), with exact solvers making heavy use of (mixed) integer linear programming (MILP) techniques [4]–[11]. To successfully cope with larger degrees, faster heuristics have also been proposed. For example, in the case of b-bit fixed-point coefficients, the approach introduced in [12], which employs error spectrum shaping techniques, is routinely used inside MATLAB$^{\text{TM}}$ for FIR quantization. The telescoping rounding approach described in [13] also seems to produce good results, while for low complexity designs, the greedy randomized heuristic from [14] is applicable to both FIR and IIR specifications.

The rest of this article will be focused on introducing a novel heuristic for designing FIR filters with fixed-point and/or floating-point coefficients. It is based on previous work for doing machine-efficient polynomial approximation of functions [15] combined with techniques that yield families of very good interpolation nodes [16]–[21] and turns out to be quite robust, especially when dealing with high degree designs. We also make available an open source C++ implementation of our approach[1]. Since it produces very good quantizations, our code should also be able to help accelerate exact solvers based on MILP.

We begin Section II with a precise statement of the problem we want to solve. Section III introduces important facts about euclidean lattices and machine-efficient polynomial approximation. They are later used in Section IV, where we give a detailed account of our approach. Examples and a comparison to other methods are discussed in Section V, followed by concluding remarks in Section VI.

A very preliminary and partial version of this paper was published in [22].

II. The finite wordlength setting

We start by recalling some well-known ideas related to linear-phase FIR filter design [23], [24]. Such a process is typically carried out in the frequency domain. In case of an N-tap causal filter with real valued impulse response $\{h[n]\}$, the corresponding frequency response we want to optimize is

$$\begin{aligned} H(\omega) &= \sum_{k=0}^{N-1} h[k] e^{-i\omega k} \\ &= G(\omega) e^{i\left(\frac{L\pi}{2} - \frac{N-1}{2}\omega\right)} \end{aligned}$$

[1] see https://github.com/sfilip/fquantizer

where G is a real valued function and L is 0 or 1. Traditionally, there are four types of FIR filters considered in practice, which depend on the parity of the filter length N and on the symmetry of h (positive for $L = 0$ and negative for $L = 1$). We have:

Type 1: Positive symmetry and odd length

$$G(\omega) = \sum_{k=0}^{n} a_k \cos(k\omega), \tag{1}$$

$n = (N-1)/2, a_0 = h[n]$ and $a_k = 2h[n-k], k = 1, \ldots, n$.

Type 2: Positive symmetry and even length

$$G(\omega) = \sum_{k=1}^{n} b_k \cos((k-1/2)\omega), \tag{2}$$

$n = N/2$ and $b_k = 2h[n-k], k = 1, \ldots, n$.

Type 3: Negative symmetry and odd length

$$G(\omega) = \sum_{k=1}^{n} c_k \sin(k\omega), \tag{3}$$

$n = (N-1)/2, c_k = 2h[n-k], k = 1, \ldots, n$ and $h[n] = 0$.

Type 4: Negative symmetry and even length

$$G(\omega) = \sum_{k=1}^{n} d_k \sin((k-1/2)\omega), \tag{4}$$

$n = N/2$ and $d_k = 2h[n-k], k = 1, \ldots, n$.

When using an algorithm like Parks-McClellan, it is convenient to notice that all four cases can be treated uniformly. We can express $G(\omega)$ as $Q(\omega)P(\omega)$, where $P(\omega)$ is of the form given in (1). Depending on the filter type, $Q(\omega)$ will be either $1, \cos(\omega/2), \sin(\omega)$ or $\sin(\omega/2)$. The resulting minimax approximation problem can be stated as:

Problem 1 (Equiripple (or minimax) FIR filter design)**.** *Let Ω be a compact subset of $[0, \pi]$ and $D(\omega)$ an ideal frequency response, continuous on Ω. For a given **filter degree** $n \in \mathbb{N}$, we want to determine $P(\omega) = \sum_{k=0}^{n} p_k \cos(\omega k)$ such that the weighted error function $E(\omega) = W(\omega)\left(D(\omega) - P(\omega)\right)$ has **minimum** uniform norm*

$$\|E(\omega)\|_{\infty,\Omega} = \sup_{\omega\in\Omega} |E(\omega)|,$$

with the weight function W continuous and positive over Ω.

Remark 1. *If we consider the change of variable $x = \cos(\omega)$, $P(\omega)$ is in fact a polynomial of degree at most n in $\cos(\omega)$ expressed in the basis of Chebyshev polynomials $(T_k)_{0\leqslant k\leqslant n}$ of the first kind,* i.e.,

$$P(\omega) = \sum_{k=0}^{n} p_k \cos(\omega k) = \sum_{k=0}^{n} p_k T_k(\cos(\omega)) = \sum_{k=0}^{n} p_k T_k(x).$$

In the sequel, it will sometimes be convenient to see our problems in the language of polynomial approximation. When this happens, x will be the approximation variable and $X \subseteq [-1, 1]$ the transformed domain associated to Ω.

A. The quantization problem

There are simple formulas to pass between the coefficients of G, P and the values of h, while Q plays a part in how W and D are chosen (we refer the reader to [25, Sec. 2] for the specifics). Despite this, for quantization purposes, it is more convenient to work with expressions (1)-(4) directly, since the values of h are just scaled versions of the coefficients of G. The finite wordlength versions of Problem 1 that are of interest here and in practice prove to be much harder to solve in general, with [10], [11] suggesting that they might be NP-complete.

For fixed-point quantization we can basically consider the impulse response coefficients to be scaled integers. If we want to use b-bit fixed-point values, we view them under the form m/s, where m is an integer in the range $I_b = [-2^{b-1}, 2^{b-1}]$ and s is a *fixed* scaling factor, which, in many cases, is a power of 2. The new version of Problem 1 we are addressing is:

Problem 2 (Integer minimax approximation)**.** *Consider Ω, D, n and W defined as in Problem 1. Let $\varphi_0, \ldots, \varphi_n$ be real valued functions, continuous over $[0, \pi]$. Determine a set of integer values $(m_k)_{0\leqslant k\leqslant n}$ from I_b, such that the error*

$$\sup_{\omega\in\Omega} \left| W(\omega) \left(\sum_{k=0}^{n} m_k \varphi_k(\omega) - D(\omega) \right) \right| \tag{5}$$

*is **minimal**.*

In case of type I filters, if we take $\varphi_0(\omega) = 1/s$ and $\varphi_k(\omega) = 2\cos(k\omega)/s$ for $k = 1, \ldots, n$, then we simply have

$$h[k] = m_{n-k}/s, \qquad k = 0, \ldots, n.$$

By looking at (2)-(4), we can arrive at similar formulations for the basis functions $(\varphi_k)_{0\leqslant k\leqslant n}$ and h for type 2 to 4 filters. Expressed as an MILP, Problem 2 becomes:

$$\begin{aligned}
\text{minimize} \quad & \delta \\
\text{subject to} \quad & W(\omega)\left(\sum_{k=0}^{n} m_k \varphi_k(\omega) - D(\omega) \right) \leqslant \delta, \omega \in \Omega, \\
& W(\omega)\left(D(\omega) - \sum_{k=0}^{n} m_k \varphi_k(\omega) \right) \leqslant \delta, \omega \in \Omega, \\
& \delta > 0, \quad m_k \in I_b \cap \mathbb{Z}, k = 0, \ldots, n.
\end{aligned}$$

In practice, Ω is generally replaced with a finite discretization Ω_d. If this set is sufficiently dense, then one hopes that the obtained solution will be close enough to the one over Ω.

The scaling factor s, often referred to as the filter gain, can also be a design parameter. Optimizing for s as well generally improves the quality of the quantization error (5). The caveat is that finding the best possible s is nontrivial [10], [26], [27].

More general quantization problems where coefficients have different formats from one another (be it fixed-point of floating-point) can also be expressed using the framework of Problem 2, with the difference being that each coefficient will have its own power of 2 scaling factor $s_k, k = 0, \ldots, n$. Nevertheless, for simplicity, in the sequel we will only consider fixed-point quantization examples with an input scaling factor s.

B. Practical implementation: transition bands issues

Despite the fact that transition regions (*i.e.*, the intervals between two successive bands from Ω) are generally unconstrained, in practice, the amplitude of P should not present any overshoots on these intervals [23, Sec. 7.8.3]. Even though such issues occur sometimes, they do not seem to be *actively* considered in existing quantization methods, whereas with our approach they are easily handled (see Section V-B).

III. Euclidean lattices & machine-efficient polynomial approximation

One main ingredient to address Problem 2 is the adaptation of ideas that prove successful in the framework of polynomial approximation for elementary function evaluation routines. We briefly recall them in Section III-B. Central to this approach is the use of a beautiful and versatile tool from Geometry of Numbers, namely Euclidean lattices, and the rich algorithmic results they benefit from, which we now introduce.

A. Lattice basis reduction overview – the SVP/CVP problems

Our main tool for finding a good solution to Problem 2 will be lattice basis reduction. The relationship between lattice basis reduction techniques and ILP has an old and rich history [28], [29]. It is thus natural to use ILP in the present context. We shall however use lattice basis reduction in a more naive way, to find small values of a discrete and modified version of Eq. (5) where Ω is replaced by a finite set; this turns out to be a good approximation to the actual quantization problem.

For the sake of completeness, we briefly review the needed facts concerning lattices and lattice basis reduction algorithms.

Definition 1. A lattice of $\mathbb{R}^n$ is a discrete subgroup of $\mathbb{R}^n$; equivalently, a lattice $L \subset \mathbb{R}^n$ is the set of integer linear combinations of a family $(\mathbf{b}_1, \dots, \mathbf{b}_d)$ of $\mathbb{R}$-linearly independent vectors of $\mathbb{R}^n$. We shall then say that $(\mathbf{b}_i)_{1 \leqslant i \leqslant d}$ is a **basis** of L, and that $d \leqslant n$ is the **dimension** of L.

In the sequel, we shall endow $\mathbb{R}^n$ with an abstract norm $\|\cdot\|$. In practice, we shall only use the two standard norms: $\|\cdot\|_2$ (Euclidean norm, $\|\mathbf{x}\|_2 = \left(\sum_{i=1}^n x_i^2\right)^{1/2}$) and $\|\cdot\|_\infty$ (supremum norm, $\|\mathbf{x}\|_\infty = \max_{i=1}^n |x_i|$).

Proposition 1. *The families $B = (\mathbf{b}_i)_{1 \leqslant i \leqslant d}$, $C = (\mathbf{c}_i)_{1 \leqslant i \leqslant d}$ are two bases of the same lattice, given in (column) matrix form as elements of $\mathbb{R}^{n \times d}$, if and only if there exists $U \in \mathrm{GL}_d(\mathbb{Z})$ such that $C = BU$. As a consequence, the quantity $(\det C^t C)^{1/2} = (\det B^t B)^{1/2}$, is independent of the basis and is associated to the lattice itself – we shall call it the* volume *of the lattice and denote it by* $\mathrm{vol}\, L$.

This implies that, for $d \geqslant 2$, a lattice has infinitely many bases. Among those bases, the ones which are made up of short and somewhat orthogonal vectors are usually considered as being good, whereas the other ones are deemed bad.

1) Lattice basis reduction & the SVP problem: It is common that interesting lattice problems are formulated in a bad basis, while the knowledge of a good basis produces a solution. *Lattice basis reduction* provides algorithms to compute a good basis given a bad one. Our case is quite typical here.

TABLE I
Lattice basis reduction algorithms

Algorithm	Approximation factor	Time
LLL_δ [32]	$(\delta^2 - 1/4)^{-(d-1)/2}$	$\mathrm{poly}\,(d)$
BKZ_β [33], [34]	$\approx \beta^{d/\beta}$	$\mathrm{poly}\,(d) 2^{O(\beta)}$
HKZ [35]	1	$2^{O(d)}$

A celebrated theorem of Minkowski [30, Chap. 2] offers some information regarding short lattice vectors:

Theorem 1 (Minkowski)**.** *Let $L \subset \mathbb{R}^n$ be a d-dimensional lattice, and $B = \{\mathbf{x} \in \mathbb{R}^n / \|\mathbf{x}\| \leqslant 1\}$ the unit ball with respect to the norm $\|\cdot\|$; then there is a $\mathbf{v} \in L - \{0\}$ such that $\|\mathbf{v}\| \leqslant 2\left(\frac{\mathrm{vol}\, L}{\mathrm{vol}\, B}\right)^{1/d}$.*

Unfortunately, the proof of this theorem eventually rests on some kind of pigeonhole principle, so is uneffective. From a computational point of view, given a basis $(\mathbf{b}_1, \dots, \mathbf{b}_d)$ of L as input, we want to find a shortest non-zero vector from L; also known as the shortest vector problem (SVP).

In this subsection we will focus on the Euclidean norm, since, in practice, an important body of *practical* work regarding SVP is concerned with the $\|\cdot\|_2$ setting. Let us start by pointing out that the decision version of this problem has been shown [31] to be hard under randomized reductions for the Euclidean norm. Dealing with lattice basis reduction and/or the SVP problem is thus a matter of compromises. A popular (yet oversimplified) rule of thumb in the community is that for any value of the parameter $k \in [1, n]$, one can solve (given the current state of the art) an SVP instance for the Euclidean norm within an approximation factor of $2^{O(d/k)}$ in complexity $O(2^k \mathrm{poly}\,(d))$. This rule of thumb is supported by Table I, which shows the performances, in terms of worst-case approximation factor / worst-case complexity, of the three most popular algorithms for lattice basis reduction.

We will not describe these algorithms nor their complete output (they actually produce a full basis) – this is a rather technical topic and is out of the scope of this paper; we refer the reader to [30] for a detailed discussion of lattice basis reduction algorithms. Note however that the LLL algorithm depends on a real parameter $1/4 < \delta < 1$, while the BKZ algorithm depends on an integer parameter $\beta \in [2, d]$, the blocksize. For $\beta = 2$, BKZ is akin to LLL, whereas for $\beta = d$, BKZ is identical to HKZ.

Table I discusses *worst-case* complexities and approximation factors. The average case of LLL is also studied experimentally in [36]; roughly speaking, on average LLL retains a c^d approximation factor, but the constant c drops down to roughly 1.05. It seems highly plausible that similar facts hold for BKZ (see [37]). **In our setting though**, it should be pointed out that the performances of those algorithms, both in terms of approximation factor and of complexity, seem significantly better than expected – HKZ-reducing a a random lattice of dimension greater than 60 is, as of today, a considerable task, while we were able to do it on an instances of dimension up to 101 coming from the problem under study in less than half a minute (see Sections IV-C and V).

2) The CVP problem & Kannan's embedding: We now turn our attention to the so-called closest vector problem (CVP): given a d-dimensional lattice L of $\mathbb{R}^d$ and a target point $\mathbf{t} \in \mathbb{R}^d$, we are looking for a vector $\mathbf{x} \in L$ which minimizes $\|\mathbf{x} - \mathbf{t}\|$. This is again, an NP-hard problem, for the $\|\cdot\|_2$ and $\|\cdot\|_\infty$ norms [38]. Again, among the wide variety of work concerning the CVP problem at large, the situation of the $\|\cdot\|_2$ norm is more promising, since several efficient approximation algorithms do exist. We shall thus again focus on the $\|\cdot\|_2$ norm.

When facing a CVP instance, one may choose to resort to exact, exponential-time approaches. However, good heuristics also exist; in particular, it is possible to relate a CVP instance to a lattice basis reduction question, using the so-called Kannan embedding technique [39]. Given a basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of L and the target vector $\mathbf{t}$, we form the $d+1$-dimensional lattice $\tilde{L}$ of $\mathbb{R}^{d+1}$ generated by the vectors $\tilde{\mathbf{b}}_i := (\mathbf{b}_i, 0)$ and $\tilde{\mathbf{t}} := (\mathbf{t}, \gamma)$, where γ is a real parameter to be chosen later on.

If $\mathbf{x}$ is a short vector of the lattice $\tilde{L}$, there exist $x_1, \ldots, x_d, x_{d+1}$ such that $\mathbf{x} = \sum_{i=1}^{d} x_i \tilde{\mathbf{b}}_i + x_{d+1}\tilde{\mathbf{t}}$; hence

$$\|\mathbf{x}\|_2^2 = \|\sum_{i=1}^{d} x_i \mathbf{b}_i + x_{d+1}\mathbf{t}\|_2^2 + \gamma^2 x_{d+1}^2. \tag{6}$$

Since $\mathbf{x}$ is assumed to be short (for instance, a vector in a reduced basis of $\tilde{L}$), $\|\sum_{i=1}^{d} x_i \mathbf{b}_i + x_{d+1}\mathbf{t}\|_2^2$ is small. If $x_{d+1} = \pm 1$, $\mp \sum_{i=1}^{d} x_i \mathbf{b}_i \in L$ is close to the vector $\mathbf{t}$.

The key question is how to choose γ such that we get $x_{d+1} = \pm 1$. If we take $\gamma \geqslant \max_{i=1}^{n} \|\mathbf{b}_i\|_2$, as soon as $|x_{d+1}| \neq 0$ we'll have $\|\mathbf{x}\|_2 \geqslant \gamma \geqslant \|\mathbf{b}_i\|_2$ for all i. Hence, as reducing bases tend to produce shorter vectors, if $\mathbf{c}_1, \ldots, \mathbf{c}_d$ is a reduced basis of L, we expect that for all $j \leqslant d$, $\|\mathbf{c}_j\|_2 < \gamma$.

Now, let $\tilde{\mathbf{c}}_1, \ldots, \tilde{\mathbf{c}}_{d+1}$ be a reduced basis of $\tilde{L}$. As any vector $\mathbf{x}$ with a nonzero last coordinate will have $\|\mathbf{x}\|_2 > \|\mathbf{c}_j\|_2$ for all j, we expect that $\tilde{\mathbf{c}}_j = (\mathbf{c}_j, 0)$ for $j \leqslant d$. Hence $\mathbf{c}_{d+1}$ has to have a nonzero last coordinate. As γ is large, we assume this last coordinate to be ± 1, otherwise the term $\gamma^2 x_{d+1}^2$ in Eq. (6) is large and it is likely that some shorter vector will be found by the reduction process. Hence, from $\tilde{\mathbf{c}}_{d+1}$ we'll often be able to deduce a vector close to $\mathbf{t}$.

In our experiments, the (heuristic) choice $\gamma = \max_{i=1}^{n} \|\mathbf{b}_i\|_2$ yielded a vector with $x_{d+1} = \pm 1$ in all cases.

We note that Kannan's description in [39] is somewhat different, since it deals with another context and needs the target vector $\mathbf{t}$ to be rather close to L – an assumption we cannot make about our current setting.

B. Approximating functions with polynomials

Evaluating mathematical functions on a computer, be it in software or hardware, has always been a key subject of study in Computer Arithmetic [40]. In practice, we frequently replace the target function with good polynomial approximations of it with respect to the uniform norm.

There are two main reasons why such an approach is used. The first is that polynomials can be evaluated very efficiently on the underlying hardware, only requiring the use of addition and multiplication operations. Secondly, by the Stone-Weierstrass theorem [40, Ch. 3.2, Thm. 6.2], any real-valued function $f \in \mathcal{C}([a, b])$ can be approximated arbitrarily well on $[a, b]$ by a polynomial $p \in \mathbb{P}(\mathbb{R})$, provided it has a sufficiently high degree. In practice, analogously to our treatment of Problem 1, we can use the Remez algorithm to compute a target degree n minimax approximation $p_n^\star$ of f over $[a, b]$.

Unfortunately, we are faced again with the same issue as in Section II. In order to use any polynomial to approximate f inside a processor, its coefficients *must* be machine numbers, representable using one of the available floating-point and/or fixed-point formats. The naive approach (*i.e.,* rounding of the Remez-obtained coefficients) can be suboptimal.

A general method for computing machine-efficient polynomial approximations is presented in [41]. It consists of carefully constructing a rational polytope which contains all optimal solutions as integer points. A solution is then retrieved with a scan of all candidate integer coordinate points inside the polytope. In terms of scalability, the method seems to be limited to small degrees. This led to the work of [15]. It uses a very natural idea: machine-coefficient polynomials have a structure that is extremely close to that of a euclidean lattice. By a clever discretization of $[a, b]$, the authors of [15] solve a CVP problem with respect to the $\|\cdot\|_2$ norm. The results they obtain give rise to machine-coefficient polynomials which are in many cases optimal, if not extremely close to optimal. The method scales to much larger degrees than the approach from [41].

Because of this promising behavior, in the next two sections we will present the extension of this approach to the FIR quantization problem. The focus will be on the aspects which are particular to filter design.

IV. Lattice-based filter design

The idea behind our approach to solving Problem 2 is to search for an approximation $P^\star(\omega) = \sum_{k=0}^{n} m_k \varphi_k(\omega)$ that fits the given precision constraints and that behaves similarly to an optimal filter $P(\omega)$ with real coefficients that we got, for instance, using the exchange algorithm [1], [2]. Although the process is based on heuristics and is by no means guaranteed to give a positive result, in practice it yields very satisfactory results, while also being fast and robust.

We start by discretizing the approximation domain: that is, pick $\ell \geqslant n + 1$ points $\omega_0, \ldots, \omega_{\ell-1}$ in Ω (and consequently the points $x_i = \cos(\omega_i), i = 0, \ldots, \ell - 1$ from X). We want that the vectors

$$m_0 \underbrace{\begin{pmatrix} W(\omega_0)\varphi_0(\omega_0) \\ W(\omega_1)\varphi_0(\omega_1) \\ \vdots \\ W(\omega_{\ell-1})\varphi_0(\omega_{\ell-1}) \end{pmatrix}}_{\mathbf{b}_0} + \cdots + m_n \underbrace{\begin{pmatrix} W(\omega_0)\varphi_n(\omega_0) \\ W(\omega_1)\varphi_n(\omega_1) \\ \vdots \\ W(\omega_{\ell-1})\varphi_n(\omega_{\ell-1}) \end{pmatrix}}_{\mathbf{b}_n}$$

and $\mathbf{v} = (W(\omega_0)P(\omega_0) \cdots W(\omega_{\ell-1})P(\omega_{\ell-1}))^t$ be as close as possible to each other, meaning find $(m_0, \ldots, m_n) \in \mathbb{Z}^{n+1}$ that minimize

$$\|m_0\mathbf{b}_0 + \ldots m_n\mathbf{b}_n - \mathbf{v}\|_2, \tag{7}$$

which is a CVP instance.

By solving (7), we hope to obtain a quantized filter for which $\|W(\omega)(P^\star(\omega) - P(\omega))\|_{\Omega,\infty}$ and ultimately $\|W(\omega)(P^\star(\omega) -$

$D(\omega))\|_{\Omega,\infty}$ are as small as possible. The use of the $\|\cdot\|_2$ norm inside (7), as opposed to the more suitable $\|\cdot\|_\infty$ one is mitigated, as indicated in Section III-A, by the existence of well-tuned, practical approximation algorithms in the Euclidean setting. The solving of the exact $\|\cdot\|_\infty$ problem, on the other hand, requires a super-exponential algorithm [39] (it is an NP-hard problem), making it quickly intractable. Since in $\mathbb{R}^\ell$, $\|\cdot\|_\infty \leqslant \|\cdot\|_2 \leqslant \sqrt{\ell}\|\cdot\|_\infty$, we expect in general that small vectors with respect to the $\|\cdot\|_2$ norm are also small when considering $\|\cdot\|_\infty$. Taking ℓ with minimal value $n+1$ hence helps [15].

A. Suitable discretization grids

Choosing appropriate points ω_i (or x_i) is critical to obtaining very good quantization results. The preferred choices in [15] are the zeros of $E(\omega)$ when first solving Problem 1 on a closed interval $X = [a, b]$ and appropriately scaled Chebyshev nodes $x_i = \frac{a+b}{2} + \frac{b-a}{2}\cos\left(\frac{(n-i)\pi}{n}\right), i = 0, \ldots, n$. The nice numerical properties of such point sets in the context of polynomial approximation on a closed interval are well-known (see for instance [42, Ch. 16] and the references therein).

In a filter design setting however, Ω (and consequently X) are routinely a union of two or more closed intervals. While taking the zeros of $E(\omega)$ together with the band edges of Ω gives good results in many cases, using equidistant nodes $\omega_i = \frac{i\pi}{n}, i = 0, \ldots, n$ inside $[0, \pi]$ (which map to Chebyshev nodes x_i inside $[-1, 1]$) does not produce the same quality results as in [15].

Nevertheless, other point sets can sometimes lead to improvements. A good example is that of the Fekete points of X when doing (weighted) polynomial approximation. In order to define them, let $w\mathbb{P}_n(\mathbb{R}) = \operatorname{span}_{\mathbb{R}}\{w(x)T_0(x), \ldots, w(x)T_n(x)\}$ be the space of weighted polynomials with degree at most n, where $w \in \mathcal{C}(X)$ is a positive weight function. Here, we actually have $w(x) = W(\arccos(x))$, where W is the weight function used in the statements of Problems 1 and 2. If we take a set of interpolation nodes $T = \{t_0, \ldots, t_n\} \subset X$ and consider the Vandermonde-like matrix

$$V(t_0, \ldots, t_n) = [v_{ij}] := [w(t_i)T_j(t_i)],$$

then the interpolation operator at the elements of T in Lagrange form has basis functions

$$\ell_i(x) = \frac{\det V(t_0, \ldots, t_{i-1}, x, t_{i+1}, \ldots, t_n)}{\det V(t_0, \ldots, t_{i-1}, t_i, t_{i+1}, \ldots, t_n)}, \qquad i = 0, \ldots, n \tag{8}$$

and is defined as

$$\mathcal{L}_T f(x) = \sum_{i=0}^{n} f(t_i)\ell_i(x).$$

Its operator norm (called *Lebesgue constant*), satisfies [43, Thm. 4.3]

$$\Lambda_{X,T,w} = \|\mathcal{L}_T\| = \max_{x\in X}\sum_{i=0}^{n}|\ell_i(x)|$$

and measures the quality of a set T for doing interpolation over X: for every $f \in \mathcal{C}(X)$ we have [43, Thm. 3.1]

$$\|f - \mathcal{L}_T f\|_{\infty,X} \leqslant (1 + \Lambda_{X,T,w})\operatorname{dist}_X(f, w\mathbb{P}_n(\mathbb{R})).$$

Fekete points are the elements of a set $T \subset X$ which maximize the absolute value in the denominators of Eq. (8), thus ensuring that their Lebesgue constant is kept small.

In case of polynomial approximation over $[-1, 1]$, Chebyshev nodes are, for practical purposes, no different than Fekete points, since they have a Lebesgue constant upper bounded by $\frac{2}{\pi}\log(n+1) + 1$, a value which is extremely close to the minimal value attainable by a set of nodes (see, for instance [42, Thm. 15.2]). In contrast, equidistant points in $[-1, 1]$ have an associated constant larger than $\frac{2^{n-2}}{n^2}$, making them extremely bad choices for doing interpolation, with many approximations actually diverging in the asymptotic regime (the classic Runge's phenomenon [44] in Numerical Analysis).

When working on multi-interval domains, we do not have simple extensions to Chebyshev nodes, even though we expect the Lebesgue constant for Fekete points over X to be $O(\log n)$ [19] as well. Unfortunately, exact computation of Fekete points is difficult in general. Still, *approximate* versions of such nodes, called Approximate Fekete Points (AFP), can be determined relatively easily using a greedy approach based on a column pivoting QR factorization [16], [18]–[20]. The method works as follows:

(1) Choose an *appropriate* discretization $Y = \{y_0, \ldots y_N\} \subset X$, $N \geqslant n+1$ by applying the theory of admissible and weakly admissible meshes introduced in [17]; we refer the reader to [19], [21], [45] for examples on how to construct such meshes;
(2) Form the $(N+1) \times (n+1)$ matrix

$$V = [v_{ij}] := [w(y_i)T_j(t_i)],$$

where $0 \leqslant i \leqslant N, 0 \leqslant j \leqslant n+1$ and each column of V^t corresponds to an element of Y;
(3) We want to extract from V^t the $(n+1)\times(n+1)$ submatrix of maximal volume; this is an NP-hard problem [46], but the greedy algorithm introduced in [18] works well in practice. Following the discussion from [16, Sec. 2], this procedure selects the columns of V^t like so (for the precise QR formulation, see for instance [19, Sec. 1]):
   (i) $\mathbf{v}_1$ is the column of V^t of maximum euclidean norm;
   (ii) Having chosen $\mathbf{v}_1, \ldots, \mathbf{v}_k$, the $(k+1)$-st column $\mathbf{v}_{k+1}$ is picked such that the volume generated by $\mathbf{v}_1, \ldots, \mathbf{v}_k, \mathbf{v}_{k+1}$ is as large as possible.

The resulting submatrix (call it $V_{\max}$) has columns which form a basis for the lattice we use in our quantization algorithm. Based on steps (i)–(ii), these basis vectors are chosen such that they are almost orthogonal to one another. Empirically, this means that a lattice basis reduction algorithm like LLL, applied to $V_{\max}$, will often run fairly quickly (when compared to the worst possible execution times), even for very large designs.

Remark 2. *We have also used this approximate Fekete points approach as a numerically robust way of initializing the exchange algorithm [45, Sec. 4.1–4.2].*

Remark 3. *Another alternative of good discretization grids is to take them according to the so-called equilibrium distribution for weighted minimax approximation on X. Articles like [47]–[49] touch on this aspect, with [48], [49] being particularly*

focused on the implications to FIR filter design problems. Computation generally involves the use of tools for numerical conformal mapping. We have not tested this idea here because it is more involved to set up than the approximate Fekete points method and it is not as flexible either, since it deals only with piecewise constant weighting functions.

B. Solving the resulting CVP problem and a refinement trick

The approximate CVP heuristic approach described in Section III-A2 that uses Kannan's embedding transforms (7) into a lattice basis reduction problem. Concretely, it will determine a vector $\mathbf{m} = \sum_{k=0}^{n} m_k \mathbf{b}_k$, which is hopefully close to $\mathbf{v}$, but also a reduced basis $(\mathbf{c}_0, \ldots, \mathbf{c}_n)$ of $(\mathbf{b}_0, \ldots, \mathbf{b}_n)$ (by applying the LLL, BKZ or HKZ algorithm). We also expect the quantization resulting from the m_k coefficients to be close to the solution of Problem 2.

Since the $\mathbf{c}_i$ vectors are usually short with respect to the $\|\cdot\|_2$ norm (and consequently with the $\|\cdot\|_\infty$ norm as well), we can use them to search in the vicinity of $\mathbf{m}$ with the goal of potentially improving the quality of our quantization.

In the function approximation setting, this idea is described in [50, p. 128]. We thus propose the following strategy:

(1) Let $\mathbf{m}'$ successively denote the vectors $\mathbf{m} + \varepsilon_i \mathbf{c}_i + \varepsilon_j \mathbf{c}_j$, where $\varepsilon_i, \varepsilon_j \in \{0, \pm 1\}, t_1, t_2 \in \mathbb{N}, 0 < t_1 < t_2 \leqslant n$ and $0 \leqslant i \leqslant t_1 < j \leqslant t_2$. The values of t_1 and t_2 can be chosen by the user.

(2) Express each $\mathbf{m}'$ as $\sum_{k=0}^{n} m'_k \mathbf{b}_k$ and pick $(m'_0, \ldots, m'_n) \in \mathbb{Z}^{n+1}$ for which

$$\left\| W(\omega) \left(\sum_{k=0}^{n} m'_k \varphi_k(\omega) - D(\omega) \right) \right\|_{\Omega,\infty} \tag{9}$$

is minimal, as final quantization vector.

C. Computation time in practice

In general, improvements are only visible when $t_1 < 4$. Taking a small t_1 also helps keep the cost of this vicinity search from dominating the quantization process. For instance, addressing (7) using the LLL algorithm and Kannan's embedding takes time $O(n^5 l \log^3 B)$, where B is the $\|\cdot\|_2$ norm of the largest vector $\mathbf{b}_i, i = 0, \ldots, n$ of the initial lattice basis. Since we are dealing with very well behaved bases with respect to reduction algorithms (see discussion from Section IV-A), this cost is usually a very pronounced overestimation of the actual runtime. Table II illustrates this, with quite reasonable execution times even for degree $n = 800$ problems. The specifications used are given in Tables III and V. We used the `fplll` C++ library implementations [51] of the LLL, BKZ and HKZ algorithms.

To compute each error of the form in (9) we use a Chebyshev-proxy approach [52], which takes $O(n^2)$ time. Already when considering $O(n^2)$ points in the vicinity of $\mathbf{m}$, the neighboring search is generally slightly more expensive than performing the associated basis reduction.

D. Other remarks

We consider it useful to summarize the heuristics that could possibly affect the quality of the method proposed in this section:

(a) transforming the initial optimal filtering problem into a discrete one at ℓ values;

(b) use of the Euclidean norm when solving (7), even though the solution we are hoping to retrieve should be optimal with respect to infinity norm over Ω;

(c) the lattice basis reduction approaches we use provide results that are hopefully close to the actual CVP solution, but they are not necessarily optimal.

Despite these apparent shortcomings, the testing we have done (see Section V) offers plenty of reasons to be optimistic.

We note that there are two other issues, typical to FIR quantization, which affect the quality of the final result. A first one is that, as opposed to the setting of [15], we have to deal with a weight function W when constructing the basis vectors used in (7). Iteratively changing the values of $W(\omega)$ for the entries in $\mathbf{b}_i$ and the target $\mathbf{v}$ which correspond to discretization points located inside the band of Ω where the approximation error is largest, can sometimes improve the overall quantization quality. The effect of slightly increasing the weighting values on worst error bands is that, for the new quantization, the error there generally decreases. If the weight change is not too large, then we can also expect that the quantization error over Ω also decreases. The second item of interest is how to treat transition bands. Simply ignoring them can pose problems, even in the real case [53]. We will come back to these aspects in Sections V-A and V-B, respectively, where we cover ideas on how to handle them in practice.

V. Experimental results

To illustrate our ideas combining lattice basis reduction (Section III-A1), Kannan's embedding (Section III-A2) and good families of points for doing interpolation (Section IV-A), we will use as measuring bar the telescoping rounding approach introduced in [13] and the MATLAB™ DSP Toolbox implementation of the error shaping algorithm from [12]. We use both discretization approaches mentioned in Section IV-A and take the best result. On some of the examples, we will use the passband ripple $20 \log_{10} \left(\frac{1+\delta_p}{1-\delta_p} \right)$ and stopband attenuation $-20 \log_{10} \delta_s$ to measure the quality of our outputs, where δ_p and δ_s correspond to the unweighted approximation errors on the passband and stopband, respectively.

We start with the type I FIR filter specifications given in [13, Table 1], which we reproduce for convenience, in Table III. For instance, A35/8 denotes a design problem adhering to specification A, of length $N = 35$ (meaning a degree $n = 17$ approximation), $b = 8$ bits used to store the filter coefficients (sign bit included) and a $s = 2^{b-1}$ scaling factor.

The results are highlighted in Table IV. The first column lists the problem specification, whereas the second one gives the real-valued coefficient minimax error computed using the Parks-McClellan algorithm. All remaining columns represent approximation errors of the transfer function for various coefficient quantization strategies. The third column errors are

TABLE II
TIMINGS (SECONDS) FOR LATTICE BASIS REDUCTIONS APPEARING IN DIFFERENT QUANTIZATION PROBLEMS

Filter	Coefficient bit size b	n	Discretization grid	LLL	BKZ	HKZ
A	8	17	Zeros+band edges	0.00078	0.00099	0.0012
A	8	17	AFP	0.00056	0.00088	0.00094
A	8	22	Zeros+band edges	0.0011	0.0017	0.0019
A	8	22	AFP	0.0011	0.0017	0.0021
A	21	62	Zeros+band edges	0.024	0.034	0.091
A	21	62	AFP	0.021	0.035	0.11
F	12	100	Zeros+band edges	0.066	0.093	0.16
F	12	100	AFP	0.066	0.10	0.11
F	16	200	Zeros+band edges	0.54	0.75	N/A
F	16	200	AFP	0.54	0.75	N/A
F	16	400	Zeros+band edges	5.28	7.92	N/A
F	16	400	AFP	5.33	7.25	N/A
F	18	800	Zeros+band edges	68.54	85.54	N/A
F	18	800	AFP	66.91	122.54	N/A

TABLE III
FILTER SPECIFICATIONS CONSIDERED IN [13]

Filter	Bands	$D(\omega)$	$W(\omega)$
A	$[0, 0.4\pi]$	1	1
	$[0.5\pi, \pi]$	0	1
B	$[0, 0.4\pi]$	1	1
	$[0.5\pi, \pi]$	0	10
C	$[0, 0.24\pi]$	1	1
	$[0.4\pi, 0.68\pi]$	0	1
	$[0.84\pi, \pi]$	1	1
D	$[0, 0.24\pi]$	1	1
	$[0.4\pi, 0.68\pi]$	0	10
	$[0.84\pi, \pi]$	1	1
E	$[0.02\pi, 0.42\pi]$	1	1
	$[0.52\pi, 0.98\pi]$	0	1

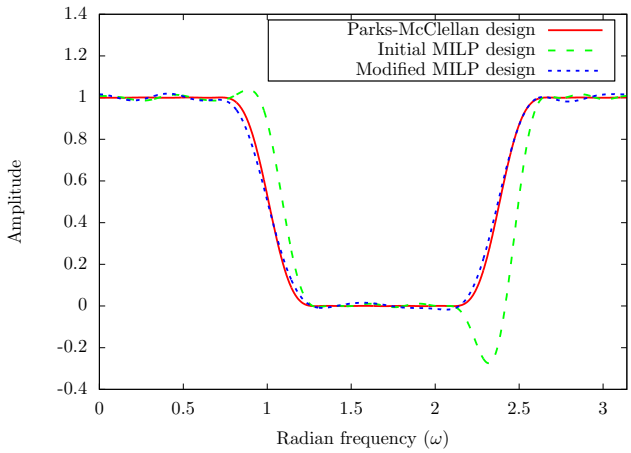

Fig. 1. Bandstop design exemplifying possible transition band problems when performing coefficient quantization. The specification is C45/8 from [13].

optimal in the sense that most of them (*i.e* those corresponding to the problems without a † or ‡ tag) are obtained using an MILP solver. Problems marked with † (of degree $n = 62$) are those where the exact solver was stopped after a certain time limit and the result given is the best one found so far. The two specifications marked with ‡ also have time-limited MILP values in [13], but during our lattice-based computations we were able to find quantizations with smaller errors, which are given here. Column four lists the errors obtained by simply rounding the real-valued minimax coefficients to their closest values in the imposed quantization format, whereas the fifth column gives the best errors from using the two coefficient telescoping rounding approach of [13, Sec. 4]. Since we were not able to reproduce most of them using our tools, these values are reported here in italic. The last three columns show the lattice-based quantization errors when applying the LLL, BKZ and HKZ basis reduction algorithms, respectively. We used a default block size of 8 for the BKZ reduction executions.

Lattice-based quantization generally outperforms telescoping rounding and gives results which are very close, if not equal, to the optimal ones. We remark, in particular, the *favorable* behavior of the LLL algorithm in all 15 test cases, especially when compared to the more expensive reductions obtained with the BKZ and HKZ algorithms.

A. Adaptive weighting

As noted in Section IV-D, in four of the instances presented in Table IV, we were able to obtain better results by *adaptively* increasing the initial weighting values of discretization nodes located inside frequency bands where the approximation error was largest. One such case is the lowpass design A125/21. When we used uniform unit weighting for all the interpolation nodes, the initial LLL quantization error was $1.397 \cdot 10^{-5}$. This value represented the passband error, while the stopband error was a slightly smaller $1.369 \cdot 10^{-5}$. Running the LLL reduction algorithm with a new passband weight of 1.5 reduces the overall starting error to $1.216 \cdot 10^{-5}$. This new value corresponds to the stopband, while for the passband we got $1.124 \cdot 10^{-5}$. Further improving this initial guess with the local search using short lattice vectors decreased the error to what is given in Table IV. For the time being, we are not able to fully automate this refinement process. Nevertheless, a filter designer can very quickly experiment with different weighting possibilities and pick the best one. More examples are given in the test file accompanying the code.

TABLE IV
QUANTIZATION ERROR COMPARISON FOR THE FILTER SPECIFICATIONS GIVEN IN [13]

Filter	Minimax E	Optimal E_{opt}	Naive rounding E_{naive}	Telescoping E_{tel}	LLL reduction E_{LLL}	BKZ reduction E_{BKZ}	HKZ reduction E_{HKZ}	Adaptive weighting
A35/8	0.01595	0.02983	0.03266	*0.03266*	**0.02983**	**0.02983**	**0.02983**	no
A45/8	$7.132 \cdot 10^{-3}$	0.02962	0.03701	*0.03186*	**0.030555**	**0.030555**	**0.030555**	no
A125/21†	$8.054 \cdot 10^{-6}$	*$1.077 \cdot 10^{-5}$*	$1.620 \cdot 10^{-5}$	*$1.179 \cdot 10^{-5}$*	$\mathbf{1.155 \cdot 10^{-5}}$	$1.304 \cdot 10^{-5}$	$1.222 \cdot 10^{-5}$	yes
B35/9	0.05275	0.07709	0.15879	***0.07854***	**0.07854**	**0.07854**	**0.07854**	yes
B45/9	0.02111	0.05679	0.11719	*0.06641*	**0.06148**	0.06368	0.06746	no
B125/22†	$2.498 \cdot 10^{-5}$	*$2.959 \cdot 10^{-5}$*	$6.198 \cdot 10^{-5}$	*$3.293 \cdot 10^{-5}$*	$\mathbf{3.243 \cdot 10^{-5}}$	$3.274 \cdot 10^{-5}$	$3.274 \cdot 10^{-5}$	no
C35/8	$2.631 \cdot 10^{-3}$	0.01787	0.04687	***0.01787***	**0.01787**	0.01917	0.01917	yes
C45/8	$6.709 \cdot 10^{-4}$	0.01609	0.03046	*0.02103*	**0.01609**	0.01897	0.01897	no
C125/21‡	$1.278 \cdot 10^{-8}$	$1.564 \cdot 10^{-6}$	$8.203 \cdot 10^{-6}$	*$2.1 \cdot 10^{-6}$*	$\mathbf{1.761 \cdot 10^{-6}}$	$1.822 \cdot 10^{-6}$	N/A	no
D35/9	0.01043	0.03252	0.12189	*0.03368*	**0.0329**	0.03623	0.03623	no
D45/9	$2.239 \cdot 10^{-3}$	0.02612	0.10898	*0.02859*	**0.02714**	0.02805	0.02805	no
D125/22‡	$4.142 \cdot 10^{-8}$	$1.831 \cdot 10^{-6}$	$3.425 \cdot 10^{-5}$	*$2.16 \cdot 10^{-6}$*	$\mathbf{1.831 \cdot 10^{-6}}$	$1.933 \cdot 10^{-6}$	$1.869 \cdot 10^{-6}$	no
E35/8	0.01761	0.03299	0.04692	***0.03404***	**0.03404**	**0.03404**	**0.03404**	no
E45/8	$6.543 \cdot 10^{-3}$	0.02887	0.03571	*0.03403*	**0.02971**	0.03007	0.03281	no
E125/21†	$7.884 \cdot 10^{-6}$	*$1.034 \cdot 10^{-5}$*	$1.479 \cdot 10^{-5}$	***$1.127 \cdot 10^{-5}$***	$1.133 \cdot 10^{-5}$	$1.192 \cdot 10^{-5}$	$1.160 \cdot 10^{-5}$	yes

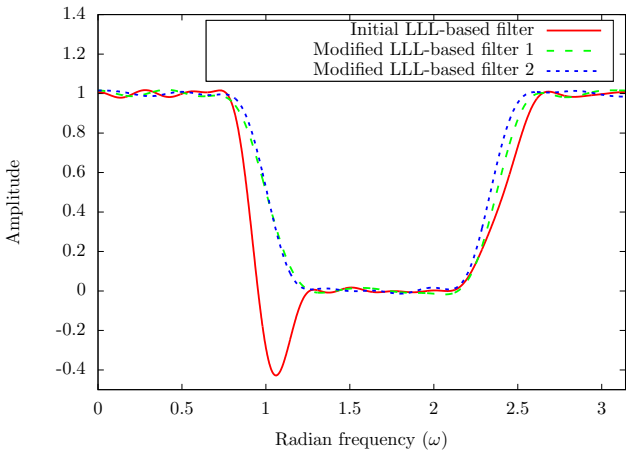

Fig. 2. The effect of iteratively removing transition band anomalies when doing lattice basis reduction-based quantization

B. Transition band problems

According to Section II-A, when designing FIR minimax filters with real-valued coefficients, the transfer function of the final design can present undesired overshoots inside transition regions. This can happen for specifications with more than two target frequency ranges, like bandstop and bandpass filters. Such problems seem even more likely to occur in the finite wordlength setting, even though articles like [12], [13] do not seem to consider this. Take for instance the bandstop design C45/8 (unit weighting everywhere, passbands $[0, 0.24\pi]$, $[0.84\pi, \pi]$ and stopband $[0.4\pi, 0.68\pi]$).

Fig. 1 shows that an optimal quantization solving Problem 2 can present ripples in both transition regions, even though the Parks-McClellan designed filter does not. A *monotonic* behavior is usually preferred instead. Since the functions that appear in practice are differentiable, we can *enforce* monotonicity by adding sign constraints on the derivative of the transfer function inside all transition regions. In terms of an MILP formulation, the modifications are minor. For our current example, adding them increases the quantization error from 0.01609 to 0.01916. The new frequency response is also given in Fig. 1.

In light of this behavior, we believe that quantization procedures should *explicitly* require frequency response checks on transition regions. One way to remove such anomalies from our lattice-based designs consists of iteratively refining the quantized results by including problematic points in the interpolation process. For instance, in case of design C45/8, the starting LLL-reduced filter (error 0.02079) has a spike in the first transition band, as shown in Fig. 2. By adding the local extrema of such spikes to the list of interpolation nodes and reapplying the lattice basis reduction routine, we generally remove them. Performing the short vector local search on this new lattice-based quantization gives the two modified frequency responses from Fig. 2. The first one is in fact the same filter as what was obtained by solving the MILP version of the quantization routine with monotonicity constraints. The second one is the optimal filter for specification C35/8 (hence, error 0.01787). Although this last frequency response is *not entirely* monotonous inside the second transition band (a very small ripple can be noticed), this design can be considered acceptable, since the transition band response there does not grow past the maximum passband amplitude. Specifications C125/21 and D125/22 had similar problems. Another possible solution is to add a pair of interpolation nodes for each problematic transition interval. We noticed that if these points are picked close to the transition region endpoints, then they generally reduce the risk of overshoots, before and after the local search.

C. Other examples

Consider now a lowpass filter specification similar to that of [12, Sec. 5]. We want to design a degree $n = 34$ type I filter with passband $[0, 0.4\pi]$ and stopband $[0.6\pi, \pi]$ in such a way that all coefficients are stored using 16 bits of precision and the stopband attenuation of the final design is maximized. Inside MATLAB™, we can first construct a real-valued coefficient filter and then use the error shaping algorithm from [12]:

```
d = fdesign.lowpass('N,Fp,Fst,Ast',68,0.4,0.6,120);
Hd = design(d,'equiripple');
Hq = maximizestopband(Hd,16,'Ntrials',100);
```

The first two lines design an equiripple filter with a 120 dB target stopband attenuation, whereas the last one performs the

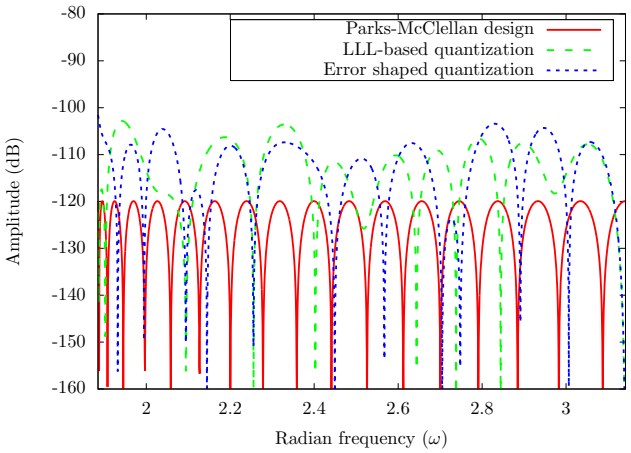

Fig. 3. Stopband attenuation for the minimax design of a given lowpass specification, alongside the 16-bit quantizations of the error shaped approach and our LLL-based algorithm.

TABLE V
HIGH DEGREE TYPE I FIR SPECIFICATIONS

Filter	Bands	$D(\omega)$	$W(\omega)$
F	$[0, 0.2\pi]$	0	10
	$[0.205\pi, \pi]$	1	1
G	$[0, 0.4\pi]$	1	1
	$[0.405\pi, 0.7\pi]$	0	10
	$[0.705\pi, \pi]$	1	1
H	$[0, 0.45\pi]$	0	10
	$[0.47\pi, \pi]$	1	1
I	$[0, 0.2\pi]$	0	10
	$[0.21\pi, \pi]$	1	1
J	$[0, 0.25\pi]$	0	10
	$[0.27\pi, 0.7\pi]$	1	1
	$[0.72\pi, \pi]$	0	10
K	$[0, 0.25\pi]$	0	10
	$[0.3\pi, 0.7\pi]$	1	1
	$[0.75\pi, \pi]$	0	10

actual quantization. Since the `maximizestopband` routine is stochastic, we execute it 100 times and take the best result, which has a 101.54 dB attenuation on the stopband. The LLL basis reduction approach on the same starting minimax design gives us a larger 102.83 dB attenuation. We were able to obtain this result by adaptively increasing the stopband weight when performing the lattice basis reduction. These responses are shown in Fig.3. The cost of having a better stopband behavior for our LLL-quantized filter is a larger passband ripple (0.046 dB as opposed to 0.013 dB for the error shaping design).

Up to this point, we have mostly showed examples of moderate order ($n < 65$), but, as indicated in Section IV-C, our euclidean lattice-based routine can also scale efficiently to much larger degrees. This is evident in the results from Table VI, which adhere to the specifications from Table V. The scaling factor used is again $s = 2^{b-1}$. Execution time is in general very reasonable, with the largest filter (equiripple frequency response + quantization) taking less than 600 seconds to design on an Intel i7-3687U CPU running a 64-bit Linux-based system and a `g++` version 5.3.0 C++ compiler.

VI. CONCLUSION

We have developed a novel approach for designing machine-number coefficient FIR filters based on an idea previously introduced in [15], which transforms the design problem using the language of euclidean lattices. The values obtained show that the method is extremely robust and competitive in practice. It frequently produces results which are close to optimal and/or beats other heuristic approaches.

There are several directions of research we are currently pursuing. The first is integrating this method into a FIR filter design suite for FPGA targets. The text [54] is a first attempt in this direction. IIR filter quantization using euclidean lattices is another idea. There are several difficulties which have to be overcome in the rational IIR setting:

- nonlinearity of the transfer function;
- the approximation domain switches from Ω to a subset of the unit circle;
- ensuring stability of the transfer function (control the position of poles).

Last but not least, for critical applications, a certificate validating the quality of the filters we provide could prove useful.

REFERENCES

[1] T. W. Parks and J. H. McClellan, "Chebyshev Approximation for Nonrecursive Digital Filters with Linear Phase," *IEEE Transactions on Circuit Theory*, vol. 19, no. 2, pp. 189–194, March 1972.

[2] E. Remes, "Sur le calcul effectif des polynomes d'approximation de Tchebichef," *Comptes rendus hebdomadaires des séances de l'Académie des Sciences*, no. 199, pp. 337–340, 1934.

[3] M. Langhammer and B. Pasca, "Floating-Point DSP Block Architecture for FPGAs," in *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ser. FPGA '15. New York, NY, USA: ACM, 2015, pp. 117–125.

[4] Y. C. Lim, S. Parker, and A. Constantinides, "Finite word length FIR filter design using integer programming over a discrete coefficient space," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 30, no. 4, pp. 661–664, Aug 1982.

[5] Y. C. Lim and S. Parker, "FIR filter design over a discrete powers-of-two coefficient space," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 31, no. 3, pp. 583–591, Jun. 1983.

[6] Y. C. Lim, R. Yang, D. Li, and J. Song, "Signed power-of-two term allocation scheme for the design of digital filters," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 5, pp. 577–584, May 1999.

[7] O. Gustafsson and L. Wanhammar, "Design of linear-phase FIR filters combining subexpression sharing with MILP," in *The 2002 45th Midwest Symposium on Circuits and Systems, 2002. MWSCAS-2002.*, vol. 3, Aug 2002, pp. 9–12.

[8] D. Shi and Y. J. Yu, "Design of Linear Phase FIR Filters With High Probability of Achieving Minimum Number of Adders," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 1, pp. 126–136, Jan 2011.

[9] D. M. Kodek, "Design of optimal finite wordlength FIR digital filters using integer programming techniques," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 28, no. 3, pp. 304–308, Jun. 1980.

[10] ——, "Performance limit of finite wordlength FIR digital filters," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2462–2469, Jul. 2005.

[11] ——, "LLL Algorithm and the Optimal Finite Wordlength FIR Design," *IEEE Transactions on Signal Processing*, vol. 60, no. 3, pp. 1493–1498, Mar. 2012.

[12] J. Nielsen, "Design of linear-phase direct-form FIR digital filters with quantized coefficients using error spectrum shaping," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 7, pp. 1020–1026, Jul 1989.

TABLE VI
HIGH DEGREE QUANTIZATION RESULTS

Filter	Degree n	Coefficient bit size b	Minimax $E^\star$	Naive rounding E_{naive}	LLL reduction E_{LLL}
F	800	18	$8.64 \cdot 10^{-4}$	$3.31 \cdot 10^{-3}$	$\mathbf{1.24 \cdot 10^{-3}}$
G	600	16	$4.78 \cdot 10^{-3}$	$1.24 \cdot 10^{-2}$	$\mathbf{6.73 \cdot 10^{-3}}$
H	250	16	$1.67 \cdot 10^{-4}$	$5.79 \cdot 10^{-3}$	$\mathbf{1.71 \cdot 10^{-3}}$
I	300	18	$4.67 \cdot 10^{-3}$	$6.56 \cdot 10^{-3}$	$\mathbf{4.91 \cdot 10^{-3}}$
J	180	16	$1.77 \cdot 10^{-3}$	$6.19 \cdot 10^{-3}$	$\mathbf{3.11 \cdot 10^{-3}}$
K	150	21	$2.99 \cdot 10^{-6}$	$7.68 \cdot 10^{-5}$	$\mathbf{2.15 \cdot 10^{-5}}$

[13] D. M. Kodek and M. Krisper, "Telescoping rounding for suboptimal finite wordlength FIR digital filter design," *Digital Signal Processing*, vol. 15, no. 6, pp. 522–535, 2005.

[14] J. Skaf and S. P. Boyd, "Filter Design With Low Complexity Coefficients," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3162–3169, Jul. 2008.

[15] N. Brisebarre and S. Chevillard, "Efficient polynomial L^∞-approximations," in *18th IEEE Symposium on Computer Arithmetic, 2007. ARITH '07*, Jun. 2007, pp. 169–176.

[16] L. P. Bos and N. Levenberg, "On the calculation of approximate Fekete points: the univariate case," *Electronic Transactions on Numerical Analysis*, vol. 30, pp. 377–397, 2008.

[17] J.-P. Calvi and N. Levenberg, "Uniform approximation by discrete least squares polynomials," *Journal of Approximation Theory*, vol. 152, no. 1, pp. 82–100, May 2008.

[18] A. Sommariva and M. Vianello, "Computing approximate Fekete points by QR factorizations of Vandermonde matrices," *Computers & Mathematics with Applications*, vol. 57, no. 8, pp. 1324–1336, Apr. 2009.

[19] ——, "Approximate Fekete points for weighted polynomial interpolation," *Electronic Transactions on Numerical Analysis*, vol. 37, pp. 1–22, 2010.

[20] L. P. Bos, J.-P. Calvi, N. Levenberg, A. Sommariva, and M. Vianello, "Geometric weakly admissible meshes, discrete least squares approximations and approximate Fekete points," *Mathematics of Computation*, vol. 80, no. 275, pp. 1623–1638, 2011.

[21] S. De Marchi, F. Piazzon, A. Sommariva, and M. Vianello, "Polynomial Meshes: Computation and Approximation," in *Proceedings of the 15th International Conference on Computational and Mathematical Methods in Science and Engineering*, 2015, pp. 414–425.

[22] N. Brisebarre, S.-I. Filip, and G. Hanrot, "De nouveaux résultats sur la synthèse de filtres RIF," in *Proc. 25ème colloque du Groupement de Recherche en Traitement du Signal et des Images (GRETSI), Lyon*, 2015.

[23] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, ser. Prentice-Hall Signal Processing Series. Prentice Hall, 2010.

[24] P. Prandoni and M. Vetterli, *Signal Processing for Communications*. Taylor & Francis, 2008.

[25] J. H. McClellan, T. W. Parks, and L. Rabiner, "A computer program for designing optimum FIR linear phase digital filters," *IEEE Transactions on Audio and Electroacoustics*, vol. 21, no. 6, pp. 506–526, Dec 1973.

[26] Y. Lim, "Design of discrete-coefficient-value linear phase FIR filters with optimum normalized peak ripple magnitude," *IEEE Transactions on Circuits and Systems,*, vol. 37, no. 12, pp. 1480–1486, Dec 1990.

[27] D. M. Kodek, "Design of optimal finite wordlength FIR digital filters," in *Proc. of the European Conference on Circuit Theory and Design, ECCTD '99.*, vol. 1, Aug 1999, pp. 401–404.

[28] H. W. Lenstra, Jr., "Integer programming with a fixed number of variables," *Mathematics of Operations Research*, vol. 8, no. 4, pp. 538–548, 1983.

[29] K. Aardal and F. Eisenbrand, "The LLL algorithm and integer programming," in *The LLL Algorithm - Survey and Applications*, 2010, pp. 293–314. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02295-1_9

[30] P. Q. Nguyen and B. Vallée, Eds., *The LLL Algorithm - Survey and Applications*, ser. Information Security and Cryptography. Springer, 2010. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02295-1

[31] M. Ajtai, "The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract)," in *Proceedings of the 30th Symposium on the Theory of Computing (STOC 1998)*. ACM, 1998, pp. 284–293.

[32] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász, "Factoring polynomials with rational coefficients," *Math. Ann.*, vol. 261, pp. 515–534, 1982.

[33] C. P. Schnorr, "A hierarchy of polynomial lattice basis reduction algorithms," *Theoretical Computer Science*, vol. 53, pp. 201–224, 1987.

[34] G. Hanrot, X. Pujol, and D. Stehlé, "Analyzing blockwise lattice algorithms using dynamical systems," in *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, 2011, pp. 447–464.

[35] R. Kannan, "Improved algorithms for integer programming and related lattice problems," in *Proceedings of the 15th Symposium on the Theory of Computing (STOC 1983)*. ACM, 1983, pp. 99–108.

[36] P. Nguyen and D. Stehlé, "LLL on the average," in *Proceedings of the 7th Algorithmic Number Theory Symposium (ANTS VII)*, ser. LNCS, vol. 4076. Springer, 2006, pp. 238–256.

[37] Y. Chen and P. Q. Nguyen, "BKZ 2.0: Better lattice security estimates," in *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, 2011, pp. 1–20. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-25385-0_1

[38] P. van Emde Boas, "Another NP-complete problem and the complexity of computing short vectors in a lattice," University of Amsterdam, Department of Mathematics, Netherlands, Tech. Rep. 8104, 1981. [Online]. Available: https://staff.fnwi.uva.nl/p.vanemdeboas/

[39] R. Kannan, "Minkowski's convex body theorem and integer programming," *Mathematics of Operations Research*, vol. 12, no. 3, pp. 415–440, Aug 1987.

[40] J.-M. Muller, *Elementary Functions: Algorithms and Implementation*. Birkhäuser, 2006.

[41] N. Brisebarre, J.-M. Muller, and A. Tisserand, "Computing machine-efficient polynomial approximations," *ACM Transactions on Mathematical Software (TOMS)*, vol. 32, no. 2, pp. 236–256, 2006.

[42] L. N. Trefethen, *Approximation Theory and Approximation Practice*. Society for Industrial and Applied Mathematics, 2013.

[43] M. J. D. Powell, *Approximation Theory and Methods*. Cambridge University Press, 1981.

[44] C. Runge, "Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten," *Zeitschrift für Mathematik und Physik*, vol. 46, no. 224–243, p. 20, 1901.

[45] S.-I. Filip, "A robust and scalable implementation of the Parks-McClellan algorithm for designing FIR filters," *to appear in ACM Transactions on Mathematical Software (TOMS)*, 2016. [Online]. Available: https://hal.inria.fr/hal-01136005

[46] A. Çivril and M. Magdon-Ismail, "On selecting a maximum volume sub-matrix of a matrix and related problems," *Theoretical Computer Science*, vol. 410, no. 47-49, pp. 4801–4811, Nov. 2009.

[47] M. Embree and L. N. Trefethen, "Green's Functions for Multiply Connected Domains via Conformal Mapping," *SIAM Review*, vol. 41, no. 4, pp. 745–761, Dec 1999.

[48] J. Shen and G. Strang, "The asymptotics of optimal (equiripple) filters," *IEEE Trans. on Signal Processing*, vol. 47, pp. 1087–1098, 1999.

[49] J. Shen, G. Strang, and A. J. Wathen, "The potential theory of several intervals and its applications," *Appl. Math. Opt.*, vol. 44, pp. 67–85, 2001.

[50] S. Chevillard, "Évaluation efficace de functions numériques. outils et exemples," Ph.D. dissertation, École Normale Supérieure de Lyon, 2009. [Online]. Available: http://www-sop.inria.fr/members/Sylvain.Chevillard/

[51] M. Albrecht, S. Bai, D. Cadé, X. Pujol, and D. Stehlé, "fplll-4.0, a floating-point LLL implementation," available at http://perso.ens-lyon.fr/damien.stehle.

[52] J. P. Boyd, "Finding the Zeros of a Univariate Equation: Proxy Rootfinders, Chebyshev Interpolation, and the Companion Matrix," *SIAM Review*, vol. 55, no. 2, pp. 375–396, 2013.

[53] L. Rabiner, J. Kaiser, and R. W. Schafer, "Some Considerations in the Design of Multiband Finite-Impulse-Response Digital Filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 22, no. 6, pp. 462–472, Dec 1974.

[54] N. Brisebarre, F. de Dinechin, S.-I. Filip, and M. Istoan, “Automatic generation of hardware FIR filters from a frequency domain specification,” *submitted*, 2016. [Online]. Available: https://hal.inria.fr/hal-01308377