Worst-case error analysis of lifting-based fast DCT-algorithms

Author for correspondence:

Miriam Primbs

Institut für Mathematik

Universität Duisburg-Essen

D-47048 Duisburg

Germany

E-mail: Miriam.Primbs@math.uni-duisburg.de

Telephone: ++49 203 379 1318

Abstract

Integer DCTs have a wide range of applications in lossless coding, especially in image compression. An integer-to-integer DCT of radix-2-length n is a nonlinear, left-invertible mapping which acts on $\mathbb{Z}^n$ and approximates the classical discrete cosine transform (DCT) of length n. All known integer-to-integer DCT-algorithms of length 8 are based on factorizations of the cosine matrix C_8^{II} into a product of sparse matrices and work with lifting steps and rounding-off. For fast implementation one replaces floating point numbers by appropriate dyadic rationals. Both, rounding and approximation leads to truncation errors. In this paper we consider an integer-to-integer transform for (2×2) rotation matrices and give estimates of the truncation errors for arbitrary approximating dyadic rationals. Further, using two known integer-to-integer DCT-algorithms, we show examplarily how to estimate the worst-case truncation error of lifting based integer-to-integer algorithms in fixed-point arithmetic, whose factorizations are based on (2×2) rotation matrices..

Key words Data compression, discrete cosine transform, error estimate, factorization of the cosine matrix, fast algorithm, fast multiplierless transform, fixed-point arithmetic, integer-to-integer DCT, lifting steps, lossless coding, reversible integer-to-integer DCT rounding-off, truncation error.

1 Introduction

The discrete cosine transform of type II (DCT-II) is defined as

$$\hat{\mathbf{y}} = C_n^{II}\,\mathbf{x} \quad \text{with} \quad C_n^{II} := \sqrt{\frac{2}{n}}\left(\epsilon_n(j)\cos\frac{j(2k+1)\pi}{2n}\right)_{j,k=0}^{n-1} \in \mathbb{R}^{n\times n},$$

where $\epsilon_n(0) := \sqrt{2}/2$ and $\epsilon_n(j) := 1$ for $j \in \{1,\dots,n-1\}$ and $\mathbf{x} \in \mathbb{R}^n$. The DCT-II has a wide range of applications in signal and image processing and is incorporated in the international standards JPEG and MPEG (see [1, 20, 21]). In particular, the DCT-II of length 8 is most commonly used and that is why our main interest also relates to this case. In some applications the input vector (or input matrix in the two-dimensional case) consists of integers, while the output data of DCT-II are no longer of integer form. Observe that the procedure of rounding an output vector of DCT-II to the next integer vector in lossy signal compression is not invertible. For lossless coding it would be of great interest to be able to characterize the output completely again with integers, but lossless coding schemes are hardly based on integer DCT-II which have been studied in recent years (see [3, 4, 5, 6, 9, 10, 14, 15, 17, 19, 22, 23]). Thus we are very interested in algorithms that deliver also output data consisting of integers. We denote those algorithms as integer-to-integer DCT-II algorithms and define it as follows.

An integer-to-integer DCT-II of length n is a nonlinear, left-invertible integer-to-integer mapping that approximates the classical DCT-II, whereas its computational cost is not higher than in the classical case.

In current literature we find several approaches to develop new integer DCT-II algorithms, decreasing the arithmetical complexity of the transform. Most of these approaches are based on special factorizations of the cosine matrix C_n^{II} (see [4, 6, 7, 8, 9, 11, 17, 18, 23]), where the matrix factors are simple matrices and lifting matrices. A lifting matrix is a matrix whose diagonal elements are 1, and only one nondiagonal element is nonzero. Simple matrices are permutation matrices or sparse matrices whose nonzero entries are only integers or half integers. The inverses of these matrices are easy to determine. Many factorizations are

based on the rotation matrix $R_2(\omega) = \begin{pmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{pmatrix}$ of order 2, which in turn can be factorized in three lifting matrices (see for $n = 8$ [4, 5, 9, 11, 16, 17, 18, 22]). Most interesting for practical purpose is the DCT-II of length 8, on which we also focus in this paper.

There are two important aspects concerning lifting-based DCT-II algorithms. Firstly, they cannot only be used for lossy but also for lossless coding. Including rounding-off in a lifting step leads to a so called integer lifting step, which is invertible and maps integers to integers. Thus, any lifting-based DCT-II algorithm can be used for lossless coding as well. This is also mentioned or used in [4, 5, 8, 9, 11, 16, 17, 18, 22, 23]. Secondly, replacing the floating point numbers in the lifting steps by appropriate dyadic rationals leads to fast algorithms, which keep the special symmetric relations of C_8^{II} into account and also remains invertible (see [4, 5, 11, 22, 23]).

Besides trial-and-error approaches how to allocate and choose the dyadic rationals in the lifting steps in [4, 11, 22, 23], in [5] we find a quasi-coordinate descent algorithm with adder constraint for systematically finding the most appropriate binary coefficients approximating the exact DCT. There, the used performance measure is the mean square error (MSE), which depends on the variance of the Gaussian input data. Further, the Minimum Adder Representation [5] of the numerators of the dyadic rationals is used to minimize the arithmetical complexity in implementation. An interesting analytic approach is presented in [4], based on a special WHT (Walsh-Hadamard Transform) factorization of the cosine matrix C_8^{II} [20], being also used in our paper. Chen, Oraintara and Nguyen determined optimal values, which minimize the statistical MSE of the algorithm if being used in the occurring integer lifting steps. These floating point numbers depend on the variance of the input data, which are again supposed to be Gaussian. Although these values are only used for analysis purpose, they are a valuable tool for performance comparisons.

However, little attention has been paid to the issue of analysis for the errors caused by both approximation and rounding, apart from the MSE considerations mentioned above. In [8] we find estimates for error bounds in infinity norm for arbitrary TERM (Triangular

Elementary Reversible Matrix) factorizations. Unfortunately, it is not directly applicable to the factorizations which are based on rotation matrices $R_2(\omega)$ and it does not use special characteristics of the current transform. In [17, 18] Plonka and Tasche propose a left-invertible integer-to-integer transform in floating point arithmetic which approximates the classical DCT-II very well. The underlying factorization of C_8^{II} (presented for arbitrary radix-2 length n in [17]) is based on the rotation matrix $R_2(\omega)$. The case $n = 8$ is very similar to the well known Loeffler factorization in [12]. We also use this factorization in our paper. For each block $R_2(\omega)$ of order 2 and for arbitrary $\mathbf{x} \in \mathbb{Z}^2$, one finds suitable integer approximation of $R_2(\omega)\,\mathbf{x}$ such that this process is left-invertible. In [17] there is presented a detailed, componentwise error analysis for the truncation error of integer approximation of $R_2(\omega)\,\mathbf{x}$ and the exact value. Using these results Plonka and Tasche derive componentwise worst case estimates for the whole transform. This method of estimating can be adopted to any integer-to-integer DCT-II algorithm in floating point arithmetic, that is based on rotation matrices. Unfortunately, these estimates cannot be used to treat dyadic approximation in the lifting steps as well.

In this paper we approximate each block $R_2(\omega)$ of order 2 for arbitrary $\mathbf{x} \in \mathbb{Z}^2$ by a suitable integer approximation of $R_2(\omega)\,\mathbf{x}$ such that this process is left-invertible and works in fixed-point arithmetic (see also [4, 5, 11, 22]). As always, this is achieved by approximating the floating point numbers in the lifting steps by dyadic rationals. Further, for this process we present componentwise worst-case error estimates depending on the range of the input data and approximation quality . This is really new, because error estimates have not yet been considered for algorithms in fixed-point arithmetic. Furthermore, we analyze two concrete algorithms for the integer-to-integer DCT of length 8, choosing special dyadic rationals for approximation. We show how to determine the componentwise error estimates. This is representative for the application of our estimates to arbitrary integer-to-integer DCT-II algorithms in fixed-point arithmetic, based on rotation matrices of order 2. A numerical comparison between these two algorithms and some of their modifications is finally presented.

The paper is organized as follows. In section 2 we present two factorizations of the cosine matrix C_8^{II} into permutation matrices and rotation matrices of the form $R_2(\omega)$. In section 3 we factorize $R_2(\omega)$ in lifting matrices and apply the lifting technique and rounding-off to this factorization (see [7, 11, 23]). Further, we present estimates for the truncation errors occurring for the integer approximation of $R_2(\omega)\mathbf{x}$ ($\mathbf{x} \in \mathbb{Z}^2$) during the integer lifting and approximation process. These estimates depend on the range of the input vector $\mathbf{x}$ and the approximation quality in the lifting matrices. In section 4 we apply these results to the factorizations of C_8^{II} and present worst case error estimates for the associated integer-to-integer DCT-II algorithms in fixed-point arithmetic. Finally, in section 5 we consider a comparison between the floating-point integer-to-integer DCT-II algorithms and some of their possible fixed-point versions.

2 Factorizations of the cosine matrix C_8^{II}

In this paper we consider two factorizations of the cosine matrix C_8^{II} into permutation matrices and rotation matrices. The first has been proposed in [17, 18] and reads

$$C_8^{II} = B_8 \begin{pmatrix} I_4 & \\ & A_4(1) \end{pmatrix} \begin{pmatrix} C_1^{II} & & & \\ & C_2^{IV} & & \\ & & C_2^{II} & \\ & & & C_2^{II} \end{pmatrix} \begin{pmatrix} T_4(0) & \\ & T_4(1) \end{pmatrix} T_8(0), \tag{1}$$

where B_8 is the bitreversal matrix, which maps $\mathbf{x} = (x_j)_{j=0}^7$ to $B_8\mathbf{x} = (x_0, x_4, x_2, x_6, x_1, x_5, x_3, x_7)^T$,

$$A_4(1) = \frac{1}{\sqrt{2}} \begin{pmatrix} \sqrt{2} & & & \\ & 1 & & 1 \\ & 1 & & -1 \\ & & \sqrt{2} & \end{pmatrix}, \; T_4(1) = \begin{pmatrix} \cos\frac{\pi}{16} & & & \sin\frac{\pi}{16} \\ & \cos\frac{3\pi}{16} & \sin\frac{3\pi}{16} & \\ & -\sin\frac{3\pi}{16} & \cos\frac{3\pi}{16} & \\ \sin\frac{\pi}{16} & & & -\cos\frac{\pi}{16} \end{pmatrix},$$

$$T_4(0) = \frac{1}{\sqrt{2}} \begin{pmatrix} I_2 & J_2 \\ I_2 & -J_2 \end{pmatrix}, \quad T_8(0) = \frac{1}{\sqrt{2}} \begin{pmatrix} I_4 & J_4 \\ I_4 & -J_4 \end{pmatrix}$$

and

$$C_2^{II} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad C_2^{IV} = \begin{pmatrix} \cos\frac{\pi}{8} & \sin\frac{\pi}{8} \\ \sin\frac{\pi}{8} & -\cos\frac{\pi}{8} \end{pmatrix}.$$

Here I_n denotes the identity matrix of order n and J_n the counter identity matrix of order n, defined by $J_n := (\delta_{n-1-i,j})_{i,j=0}^{n-1}$. As in [18] we denote the five orthogonal matrix factors of C_8^{II} in (1) in this order by

$$C_8^{II} = B_8\, A_8(0,1)\, T_8(0,1,0,0)\, T_8(0,1)\, T_8(0).$$

Note that this factorization coincides up to simple permutation with the well known Loeffler factorization (see [12]). Nevertheless, we choose this special presentation to make our result better comparable to these in [17, 18]. Note, that in [17] one finds a factorization of C_n^{II} for arbitrary radix-2 length n, which agrees with ours for the case $n = 8$.

For $A \in \mathbb{R}^{k\times l}$, $B \in \mathbb{R}^{n\times m}$ we define $A \oplus B := \mathrm{diag}(A, B) = \begin{pmatrix} A & \\ & B \end{pmatrix} \in \mathbb{R}^{(k+n)\times(l+m)}$. Choosing the expansion factor 2, we then obtain the factorization

$$2C_8^{II} = B_8\, A_8(0,1)\, (I_4 \oplus \sqrt{2}I_4)\, T_8(0,1,0,0)\, (\sqrt{2}I_4 \oplus I_4)\, T_8(0,1)\, \sqrt{2}\, T_8(0), \tag{2}$$

which has been proposed in [17, 18].

The second factorization we want to apply can be found in [4, 20]. It reads

$$2\sqrt{2}\, C_8^{II} = B_8 \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & U_{22} & \\ & & & U_{44} \end{pmatrix} B_8 H_w, \tag{3}$$

where B_8 is again the bitreversal matrix and H_w denotes the Walsh Hadamard matrix defined by

$$H_w := \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{pmatrix},$$

$$U_{22} := \begin{pmatrix} \cos(\frac{\pi}{8}) & \sin(\frac{\pi}{8}) \\ -\sin(\frac{\pi}{8}) & \cos(\frac{\pi}{8}) \end{pmatrix}$$

and finally

$$U_{44} := \begin{pmatrix} \cos(\frac{7\pi}{16}) & & & -\sin(\frac{7\pi}{16}) \\ & \cos(\frac{3\pi}{16}) & -\sin(\frac{3\pi}{16}) & \\ & \sin(\frac{3\pi}{16}) & \cos(\frac{3\pi}{16}) & \\ \sin(\frac{7\pi}{16}) & & & \cos(\frac{7\pi}{16}) \end{pmatrix} T \begin{pmatrix} \cos(\frac{3\pi}{8}) & & & -\sin(\frac{3\pi}{8}) \\ & \cos(\frac{3\pi}{8}) & -\sin(\frac{3\pi}{8}) & \\ & \sin(\frac{3\pi}{8}) & \cos(\frac{3\pi}{8}) & \\ \sin(\frac{3\pi}{8}) & & & \cos(\frac{3\pi}{8}) \end{pmatrix}$$

with

$$T := \begin{pmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{pmatrix}.$$

Observe, that in contrast to the first factorization, the scaling factor is $2\sqrt{2}$ instead of 2.

3 Integer-to-integer transform via lifting in rotation matrices

The integer-to-integer DCT-II-algorithms we consider in this paper are based on the factorizations (2) and (3) of the cosine matrix C_8^{II} presented in section 2. They only consist of permutation matrices and rotation matrices $R_2(\omega)$ and $\tilde{R}_2(\omega)$ of order 2 with angle $\omega \in (0, \frac{\pi}{2}]$, which are defined by

$$R_2(\omega) = \begin{pmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{pmatrix},$$

$$\tilde{R}_2(\omega) = \begin{pmatrix} \cos\omega & -\sin\omega \\ \sin\omega & \cos\omega \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} R_2(\omega) \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{4}$$

More concretely, the factorization (2) needs 5 rotations $R_2(\omega)$ with $\omega \in \{\frac{\pi}{16}, \frac{\pi}{8}, \frac{\pi}{4}, \frac{3\pi}{16}\}$, where $\omega = \frac{\pi}{4}$ is used twice. The factorization (3) needs $R_2(\frac{\pi}{8})$ and four rotations $\tilde{R}_2(\omega)$ with $\omega \in \{\frac{3\pi}{16}, \frac{3\pi}{8}, \frac{7\pi}{16}\}$.

In this section, we first consider the problem of how to find a good integer approximation of $R_2(\omega)\mathbf{x}$, where $\mathbf{x} \in \mathbb{Z}^2$, which can be carried out in fixed point arithmetic. We start with some notations. Let $s \in \mathbb{R}$ with $s \neq 0$ be given. Then matrices of the form

$$\begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ s & 1 \end{pmatrix}$$

are called *lifting matrices* of order 2. The inverse of a lifting matrix is again a lifting matrix, and we have

$$\begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & -s \\ 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ s & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 0 \\ -s & 1 \end{pmatrix}.$$

For $a \in \mathbb{R}$ let $\lfloor a \rfloor := \max\{x \leq a; x \in \mathbb{Z}\}$. Further, let $\mathrm{rd}(a) := \lfloor a + \frac{1}{2} \rfloor$ be the next integer to a and $\{a\} = a - \lfloor a \rfloor$ the non-integer part of a.

A *lifting step* of the form

$$\hat{\mathbf{y}} = \begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix} \mathbf{x}$$

with $\mathbf{x} = (x_0, x_1)^T \in \mathbb{Z}^2$ can be approximated by $\mathbf{y} = (y_0, y_1)^T \in \mathbb{Z}^2$ with

$$y_0 = x_0 + \mathrm{rd}(\, sx_1), \quad y_1 = x_1.$$

This transform is invertible and its inverse reads

$$x_0 = y_0 - \mathrm{rd}(\, sy_1), \quad x_1 = y_1,$$

which is not difficult to prove (see [2]). This scheme is called integer lifting scheme and has also been applied in [4, 5, 7, 8, 9, 11, 17, 18, 23].

Every rotation matrix $R_2(\omega)$ of order 2 can be represented as a product of three lifting matrices with

$$R_2(\omega) = \begin{pmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{pmatrix} = \begin{pmatrix} 1 & \tan\frac{\omega}{2} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\sin\omega & 1 \end{pmatrix} \begin{pmatrix} 1 & \tan\frac{\omega}{2} \\ 0 & 1 \end{pmatrix}.$$

In order to develop an algorithm in fixed-point arithmetic, the main idea is to approximate the trigonometric values $\tan\frac{\omega}{2}$ and $\sin\omega$ by dyadic rationals, i.e. by numbers a, b of the form $a = \frac{\beta_a}{2^n}$, $b = \frac{\beta_b}{2^n}$ with $\beta_a, \beta_b, n, \in \mathbb{N}$. This idea has also been used in [4, 5, 6, 11, 23], to avoid floating point multiplications. Afterwards, we apply three integer lifting steps to the factorization

$$\tilde{H}_2 := \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -b & 1 \end{pmatrix} \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1-ab & 2a-a^2b \\ -b & 1-ab \end{pmatrix}. \tag{5}$$

We obtain the following estimates.

Theorem 3.1 *Let $H_2 := R_2(\omega)$ with $\omega \in (0, \frac{\pi}{2}]$ be a rotation matrix.*

Further, let $a = a(\omega) = \frac{\beta_1}{2^n} \geq 0$ and $b = b(\omega) = \frac{\beta_2}{2^n} \geq 0$, $\beta_1, \beta_2, n \in \mathbb{N}$ be given, with

$$|\tan\frac{\omega}{2} - a| \leq 2^{-j} \quad \textit{and} \quad |\sin\omega - b| \leq 2^{-j}$$

for some fixed $j \in \mathbb{N}$. Then for arbitrary $\mathbf{x} = (x_0, x_1)^T \in (-2^k, 2^k]^2 \cap \mathbb{Z}^2$, a suitable integer approximation $\mathbf{y} = (y_0, y_1)^T \in \mathbb{Z}^2$ of $\hat{\mathbf{y}} = H_2\mathbf{x}$ is given, with $y_0 = z_2$ and $y_1 = z_1$, where

$$z_0 := x_0 + \mathrm{rd}\ (x_1\, a), \quad z_1 := x_1 + \mathrm{rd}\ (-z_0\, b), \quad z_2 := z_0 + \mathrm{rd}\ (z_1\, a).$$

The procedure is left-invertible. The left-inverse reads $x_0 = w_2$, $x_1 = w_1$, where

$$w_0 := y_0 - \mathrm{rd}\ (y_1\, a), \quad w_1 := y_1 - \mathrm{rd}\ (-w_0\, b), \quad w_2 := w_0 - \mathrm{rd}\ (w_1\, a).$$

Further, the error estimates

$$|\,\hat{y}_0 - y_0| \;\leq\; \left(2 + a + a^2 + \sin\omega(1 + a + \tan\frac{\omega}{2})\right) 2^{k-j} + \frac{1}{2}(2 + a - ab), \tag{6}$$

$$|\,\hat{y}_1 - y_1| \;\leq\; (1 + a + \sin\omega)\, 2^{k-j} + \frac{1}{2}(b + 1) \tag{7}$$

hold.

Proof: Replacing the trigonometric values $\tan\frac{\omega}{2}$ and $\sin\omega$ by the dyadic rationals a and b and using the factorization

$$H_2 = \begin{pmatrix} 1 & \tan\frac{\omega}{2} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\sin\omega & 1 \end{pmatrix} \begin{pmatrix} 1 & \tan\frac{\omega}{2} \\ 0 & 1 \end{pmatrix}$$

we obtain the approximation matrix

$$\tilde{H}_2 = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -b & 1 \end{pmatrix} \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 - ab & 2a - a^2 b \\ -b & 1 - ab \end{pmatrix}. \tag{8}$$

The formulas for y_0, y_1, x_0 and x_1 directly follow by applying the lifting step approximation to the three matrices.

We now estimate the componentwise errors. First we consider the truncation error $|\hat{y}_0 - y_0|$ of the first component. For the exact value we have

$$\hat{y}_0 = (\cos\omega)\, x_0 + (\sin\omega)\, x_1$$

and for the calculated value we find

$$\begin{aligned} y_0 = z_2 &= z_0 + \lfloor a z_1 + \frac{1}{2} \rfloor \\ &= z_0 + \lfloor a(x_1 + \lfloor -b z_0 + \frac{1}{2} \rfloor) + \frac{1}{2} \rfloor \\ &= z_0 + \lfloor a(x_1 - b z_0 + \frac{1}{2} - \epsilon_1) + \frac{1}{2} \rfloor, \end{aligned}$$

setting $\epsilon_1 := \{-b z_0 + \frac{1}{2}\}$. With $\epsilon_2 := \{a(x_1 - b z_0 + \frac{1}{2} - \epsilon_1) + \frac{1}{2}\}$, it follows that

$$\begin{aligned} y_0 &= z_0 + a(x_1 - b z_0 + \frac{1}{2} - \epsilon_1) + \frac{1}{2} - \epsilon_2 \\ &= z_0(1 - ab) + a x_1 + \frac{a}{2} - a\epsilon_1 + \frac{1}{2} - \epsilon_2 \\ &= (x_0 + \lfloor a x_1 + \frac{1}{2} \rfloor)(1 - ab) + a x_1 + \frac{a}{2} - a\epsilon_1 + \frac{1}{2} - \epsilon_2. \end{aligned}$$

Finally we put $\epsilon_3 := \{a x_1 + \frac{1}{2}\}$ and receive

$$\begin{aligned} y_0 &= (x_0 + a x_1 + \frac{1}{2} - \epsilon_3)(1 - ab) + a x_1 + \frac{a}{2} - a\epsilon_1 + \frac{1}{2} - \epsilon_2 \\ &= (1 - ab)\, x_0 + (2a - a^2 b)\, x_1 + 1 - \frac{ab}{2} + \frac{a}{2} - a\epsilon_1 - \epsilon_2 - \epsilon_3(1 - ab). \end{aligned}$$

Consequently,

$$\begin{aligned} |\hat{y}_0 - y_0| &\le |\cos\omega - (1 - ab)||x_0| + |\sin\omega - (2a - a^2 b)||x_1| \\ &\quad + |1 - \frac{ab}{2} + \frac{a}{2} - a\epsilon_1 - \epsilon_2 - \epsilon_3(1 - ab)| \\ &= I + II + III, \end{aligned}$$

where

$$\begin{aligned} I &:= |\cos\omega - (1 - ab)||x_0|, \\ II &:= |\sin\omega - (2a - a^2 b)||x_1|, \\ III &:= |1 - \frac{ab}{2} + \frac{a}{2} - a\epsilon_1 - \epsilon_2 - \epsilon_3(1 - ab)|. \end{aligned}$$

We estimate these three terms separately. For the first term, observing that $\cos\omega = 1 - \sin\omega \tan\frac{\omega}{2}$, we find

$$\begin{aligned} I = |\cos\omega - (1-ab)||x_0| &= |1 - \sin\omega\tan\frac{\omega}{2} - 1 + ab||x_0| \\ &= |ab - \sin\omega\tan\frac{\omega}{2}||x_0| \\ &\leq (|ab - a\sin\omega| + |a\sin\omega - \sin\omega\tan\frac{\omega}{2}|)|x_0| \\ &\leq (a|b - \sin\omega| + \sin\omega|a - \tan\frac{\omega}{2}|)|x_0| \\ &\leq (a + \sin\omega)\, 2^{k-j}. \end{aligned}$$

For the second term we obtain, using $\sin\omega = 2\tan\frac{\omega}{2} - \tan^2(\frac{\omega}{2})\sin\omega$,

$$\begin{aligned} II = |\sin\omega - 2a + a^2 b||x_1| &= |2\tan\frac{\omega}{2} - \tan^2(\frac{\omega}{2})\sin\omega - 2a + a^2 b||x_1| \\ &\leq (2|\tan\frac{\omega}{2} - a| + |a^2 b - \tan^2(\frac{\omega}{2})\sin\omega|)|x_1| \\ &\leq (2^{1-j} + |a^2 b - a^2\sin\omega| + |a^2\sin\omega - \tan^2(\frac{\omega}{2})\sin\omega|)|x_1| \\ &\leq (2^{1-j} + a^2|b - \sin\omega| + \sin\omega|a - \tan\frac{\omega}{2}||a + \tan\frac{\omega}{2}|)|x_1| \\ &\leq (2^{1-j} + a^2 2^{-j} + \sin\omega(a + \tan\frac{\omega}{2})2^{-j})2^k \\ &\leq (2 + a^2 + \sin\omega(a + \tan\frac{\omega}{2}))2^{k-j}. \end{aligned}$$

Since $\epsilon_1, \epsilon_2, \epsilon_3 \in [0, 1)$ we get

$$1 - \frac{ab}{2} + \frac{a}{2} - a\epsilon_1 - \epsilon_2 - \epsilon_3(1-ab) \in [-\frac{1}{2}(2 + a - ab), \frac{1}{2}(2 + a - ab)],$$

and thus for the third term

$$III = |1 - \frac{ab}{2} + \frac{a}{2} - a\epsilon_1 - \epsilon_2 - \epsilon_3(1-ab)| \leq \frac{1}{2}(2 + a - ab).$$

Hence we get for the truncation error in the first component

$$|\hat{y}_0 - y_0| \leq \left(2 + a + a^2 + \sin\omega(1 + a + \tan\frac{\omega}{2})\right) 2^{k-j} + \frac{1}{2}(2 + a - ab).$$

Furthermore, we need to estimate the error of the second component. For the exact value we have

$$\hat{y}_1 = -(\sin\omega)\, x_0 + (\cos\omega)\, x_1.$$

For the calculated value we get

$$\begin{aligned} y_1 = z_1 &= x_1 + \lfloor -bz_0 + \frac{1}{2} \rfloor \\ &= x_1 - bz_0 + \frac{1}{2} - \epsilon_1 \\ &= x_1 - b(x_0 + \lfloor ax_1 + \frac{1}{2} \rfloor) + \frac{1}{2} - \epsilon_1, \end{aligned}$$

setting $\epsilon_1 := \{-bz_0 + \frac{1}{2}\}$ again. With $\epsilon_3 := \{ax_1 + \frac{1}{2}\}$ we also find

$$y_1 = (1 - ab)\, x_1 - bx_0 - \frac{b}{2} + \epsilon_3 b + \frac{1}{2} - \epsilon_1.$$

Thus we have for the truncation error in the second component

$$\begin{aligned} |\hat{y}_1 - y_1| &\leq |\, b - \sin\omega |\, |x_0| + |\, \cos\omega - (1 - ab)|\, |x_1| + |\, \frac{b}{2} - \frac{1}{2} + \epsilon_1 - \epsilon_3 b| \\ &= I + II + III. \end{aligned}$$

For the first term we obtain

$$I := |\, b - \sin\omega |\, |x_0| \leq 2^{k-j}.$$

We estimate the second term in analogy to the first term of the first component and receive

$$II := |\, \cos\omega - (1 - ab)|\, |x_1| \leq (a + \sin\omega) 2^{k-j}.$$

For the third term III, observing that $\epsilon_1, \epsilon_3 \in [0, 1)$, we obtain

$$III := |\frac{b}{2} - \frac{1}{2} + \epsilon_1 - \epsilon_3 b| \leq \frac{1}{2}(b + 1).$$

Altogether for the truncation error of the second component we find the estimate

$$|\hat{y}_1 - y_1| \leq (1 + a + \sin\omega) 2^{k-j} + \frac{1}{2}(1 + b).$$

□

Remark 3.2 *Considering the error estimates (6) and (7), we see that controlling the input vector* $\mathbf{x}$ *(k fixed) and improving the approximation (j large) leads to arbitrary small error estimates for the first two terms, due to the fact that the terms within the brackets are bounded.*

In [17, 18] Plonka and Tasche presented the following estimates for integer lifting applied to the above factorization of the rotation matrix without replacing the trigonometric values by dyadic rationals:

$$\begin{aligned} |\,\hat{y}_0 - y_0| &\leq \frac{1}{2}(1 + \tan\frac{\omega}{2} + \cos\omega), \\ |\,\hat{y}_1 - y_1| &\leq \frac{1}{2}(1 + \sin\omega). \end{aligned}$$

The second terms $\frac{1}{2}(\,2 + a - ab)$ *and* $\frac{1}{2}(\,1 + b)$ *represent exactly these estimates, where the trigonometric values are replaced by the approximating dyadic rationals.*

Example 3.3 *Let* $\hat{\mathbf{y}} := R_2(\omega)\mathbf{x}$ *with arbitrary* $\mathbf{x} = (x_0, x_1)^T \in (-2^k, 2^k]^2 \cap \mathbb{Z}^2$ *and* $\mathbf{y}$ *the integer approximation computed by the procedure in Theorem 3.1. For* $\omega \in \{\frac{\pi}{4}, \frac{\pi}{8}, \frac{\pi}{16}, \frac{3\pi}{16}\}$ *we choose for* $\tan\frac{\omega}{2}$ *and* $\sin\omega$ *15-bit and 8-bit approximations. Further we control the input vectors* $\mathbf{x}$*, choosing different values for* k*. The special choice of* ω *and* k *depends on the factorizations (2) of* $2\,C_8^{II}$ *in section 2. Inserting in (6) and (7), we obtain the following table, which presents the proposed dyadic rationals and their estimates.*

ω	k	j	2^8a	2^8b	$\lvert\hat{y_0} - y_0\rvert$	$\lvert\hat{y_1} - y_1\rvert$	j	$2^{15}a$	$2^{15}b$	$\lvert\hat{y_0} - y_0\rvert$	$\lvert\hat{y_1} - y_1\rvert$
$\frac{\pi}{4}$	10	12	106	181	2.0302	1.3838	16	13573	23170	1.1213	0.8867
$\frac{\pi}{4}$	9	12	106	181	1.5454	1.1187	16	13573	23170	1.0910	0.8701
$\frac{\pi}{8}$	9	11	51	98	1.7550	1.0869	17	6517	12539	1.0722	0.6975
$\frac{\pi}{16}$	8	10	25	50	1.6244	0.9208	16	3327	6393	1.0487	0.6026
$\frac{3\pi}{16}$	8	9	78	142	2.7133	1.7075	18	9940	18205	1.0706	0.7796

Table 1: Dyadic rationals and estimates for 8-bit and 15-bit approximation PART I.

Obviously, improving the quality of the approximation enables us to get closer to the estimates in [17, 18], but that also increases the used arithmetical capacity of the procedure in Theorem 3.1.

Concerning the second factorization (3) from section 2, we will need the following estimates in the next section.

ω	k	j	2^8a	2^8b	$\lvert\hat{y_0}-y_0\rvert$	$\lvert\hat{y_1}-y_1\rvert$	j	$2^{15}a$	$2^{15}b$	$\lvert\hat{y_0}-y_0\rvert$	$\lvert\hat{y_1}-y_1\rvert$
$\frac{\pi}{8}$	10	11	51	98	2.4485	1.4824	17	6517	12539	1.0831	0.7037
$\frac{3\pi}{8}$	10	9	171	237	11.5697	6.1466	16	21895	30274	1.1078	1.0024
$\frac{3\pi}{16}$	11	9	78	142	14.2314	8.2184	18	9940	18205	1.0931	0.7923
$\frac{7\pi}{16}$	11	11	210	251	7.0915	3.7913	16	26892	32138	1.1980	1.07793

Table 2: Dyadic rationals and estimates for 8-bit and 15-bit approximation PART II.

4 Integer-to-integer DCT-II algorithms of length 8

In this section we discuss the over-all worst case error estimates for the integer-to-integer DCT-II algorithms based on both factorizations in section 2. This is exemplary for all integer-to-integer DCT-II algorithms that use a rotation based factorization of the cosine matrix C_8^{II}. Since the implementation of a lifting based integer-to-integer DCT-II algorithm in fixed-point arithmetic in respect of a given factorization is well known, we dispense with detailed algorithms here. For the first factorization (2) one finds a detailed presentation of the associated integer-to-integer floating-point algorithm in [18]. We denote this algorithm in [18] with PTfl. Replacing the trigonometric values by the appropriate dyadic rationals at any one time for 8-bit or 15-bit approximation from table 1, we receive two algorithms which we denote with PT8 and PT15.

Analogously, we get the algorithms CONfl, CON8 and CON15 based on the second factorization (3) in section 2 and the dyadic numbers from table 2. PT and CON are here the initials of the authors of [17, 18] respectively [4, 5]. Note, that the choice of the "k"s in Table 1 and table 2 is adequate to integer input vectors with the range $(-2^7, 2^7]^8 \cap \mathbb{Z}^8 = (-128, 128]^8 \cap \mathbb{Z}^8$.

Now analyze the worst case errors of the algorithms PT8, PT15, CON8 and CON15 comparing the resulting integer vector $\mathbf{y}$ with the exact result $\hat{\mathbf{y}} = 2\,C_8^{II}\mathbf{x}$ for the PT algorithms

and $\hat{\mathbf{y}} = 2\sqrt{2}\, C_8^{II}\mathbf{x}$ for the CON algorithms for arbitrary $\mathbf{x} \in (-128, 128]^8 \cap \mathbb{Z}^8$.

Theorem 4.1 *Let $\mathbf{x} \in (-128, 128]^8 \cap \mathbb{Z}^8$ be an arbitrary vector of integers. Using algorithms PT8, PT15, CON8 and CON15 the resulting integer approximations $\mathbf{y}$ of $\hat{\mathbf{y}} = 2\, C_8^{II}\mathbf{x}$ respectively $\hat{\mathbf{y}} = 2\sqrt{2}\, C_8^{II}\mathbf{x}$ satisfy the following error estimates.*

	PT8	PT15	CON8	CON15
$\lvert\hat{y}_0 - y_0\rvert$	2.0302	1.0910	0	0
$\lvert\hat{y}_1 - y_1\rvert$	4.3377	2.1194	15,3771	2.3973
$\lvert\hat{y}_2 - y_2\rvert$	1.7550	1.0722	2,4485	1.0830
$\lvert\hat{y}_3 - y_3\rvert$	6.3095	3.3627	19,7569	2.2412
$\lvert\hat{y}_4 - y_4\rvert$	1.1187	0.8701	0	0
$\lvert\hat{y}_5 - y_5\rvert$	6.9560	3.5792	27,2661	2.5711
$\lvert\hat{y}_6 - y_6\rvert$	1.0869	0.6975	1,4825	0.7037
$\lvert\hat{y}_7 - y_7\rvert$	2.6283	1.3821	16,3378	2.3600
$\lVert\hat{\mathbf{y}} - \mathbf{y}\rVert_2$	10.9761	5.8399	40,5629	4.9617
$\lVert\hat{\mathbf{y}} - \mathbf{y}\rVert_\infty$	6.9560	3.5792	27,2661	2,5711

Proof: We show how to estimate the worst case error for PT15 representively. Before starting, let us fix some notations. Based on the factorization (2) from section 2, we set $\mathbf{x}^{(1)} := T_8(0)\mathbf{x}$. With $\mathbf{x}^{(2)}$ we denote the integer approximation of $T_8(0,1)\mathbf{x}^{(1)}$, which is calculated as follows:

Set $\mathbf{w}^{(0)} := (x_0^{(1)}, x_1^{(1)}, x_2^{(1)}, x_3^{(1)})^T$, $\mathbf{w}^{(1)} := (x_4^{(1)}, x_5^{(1)})^T$, $\mathbf{w}^{(2)} := (x_7^{(1)}, x_6^{(1)})^T$.

Compute $\mathbf{z} := \sqrt{2}\, T_4(0)\mathbf{w}^{(0)}$ and

$$\begin{aligned}
\mathbf{z}^{(0)} &:= \mathrm{rd}\left(\left(\frac{3327}{2^{15}} \oplus \frac{2485}{2^{13}}\right)\mathbf{w}^{(2)}\right) + \mathbf{w}^{(1)},\\
\mathbf{z}^{(1)} &:= \mathrm{rd}\left(\left(-\frac{6393}{2^{15}} \oplus -\frac{18205}{2^{15}}\right)\mathbf{z}^{(0)}\right) + \mathbf{w}^{(2)},\\
\mathbf{z}^{(2)} &:= \mathrm{rd}\left(\left(\frac{3327}{2^{15}} \oplus \frac{2485}{2^{13}}\right)\mathbf{z}^{(1)}\right) + \mathbf{z}^{(0)}.
\end{aligned}$$

Set $\mathbf{x}^{(2)} := (\mathbf{z}^T, z_0^{(2)}, z_1^{(2)}, z_1^{(1)}, -z_0^{(1)})^T$. Analogously, $\mathbf{x}^{(3)}$ and $\mathbf{x}^{(4)}$ denote the calculated integer approximations of $T_8(0,1,0,0)T_8(0,1)T_8(0)\mathbf{x}$
and $A_8(0,1)T_8(0,1,0,0)T_8(0,1)T_8(0)\mathbf{x}$ (see factorization (2)). Please note that $\mathbf{y} = B_8\mathbf{x}^{(4)}$ yields, because in the last step no rounding occurs.
It is easy to see that with $\mathbf{x} \in (-128, 128]^8 \cap \mathbb{Z}^8$ we get

$$\mathbf{x}^{(1)} \in (-2^8, 2^8]^8 \cap \mathbb{Z}^8,\ \mathbf{x}^{(2)} \in (-2^9, 2^9]^8 \cap \mathbb{Z}^8 \quad \text{and} \quad \mathbf{x}^{(3)} \in (-2^{10}, 2^{10}]^8 \cap \mathbb{Z}^8.$$

Note, that it is not necessary to know which range $\mathbf{x}^{(4)}$ has, because the last step only consists of permutation and uses no approximating transform.

Thus we can use the estimates of Example 3.3. We give a detailed explanation for the estimates of the error in the first and fifth component, the others follow analogously.
For the first component we have $y_0 = x_0^{(4)} = x_0^{(3)}$ and thus we receive

$$|\hat{y}_0 - y_0| = |\hat{y}_0 - x_0^{(4)}| = |\hat{y}_0 - x_0^{(3)}|.$$

Let us now set $\tilde{x}_0^{(3)} = \frac{1}{\sqrt{2}}(x_0^{(2)} + x_1^{(2)})$ in order to consider the error which is made in the first component during the third step. That leads us to

$$|\hat{y}_0 - y_0| \leq |\hat{y}_0 - \tilde{x}_0^{(3)}| + |\tilde{x}_0^{(3)} - x_0^{(3)}| = |\hat{y}_0 - \frac{1}{\sqrt{2}}(x_0^{(2)} + x_1^{(2)})| + |\tilde{x}_0^{(3)} - x_0^{(3)}|.$$

(We dispense with this intermediate step in the analyze of the fifth component, later.)

The first term contains the complete error, that is made until the second step, while the second term describes the first component error made in the third step. Since $\mathbf{x}^{(2)} \in (-2^9, 2^9]^8 \cap \mathbb{Z}^8$, we can use the estimate of the first component of Example 3.3 for $\omega = \frac{\pi}{4}$, $k = 9$, $a = \frac{13573}{2^{15}}$ and $b = \frac{23170}{2^{15}}$ and obtain

$$|\hat{y}_0 - y_0| \leq |\,\hat{y}_0 - \frac{1}{\sqrt{2}}(x_0^{(2)} + x_1^{(2)})| + 1.090961.$$

Further

$$|\,\hat{y}_0 - \frac{1}{\sqrt{2}}(x_0^{(2)} + x_1^{(2)})| \quad = \quad |\hat{y}_0 - \frac{1}{\sqrt{2}}(x_0^{(1)} + x_1^{(1)} + x_2^{(1)} + x_3^{(1)})|$$

$$
\begin{aligned}
&= |\hat{y}_0 - \frac{1}{\sqrt{2}} \sum_{j=0}^{7} x_j| \\
&= 0,
\end{aligned}
$$

where $\mathbf{x} = (x_j)_{j=0}^{7}$ is the input vector. This yields the following estimate for the overall worst case error in the first component

$$|\hat{y}_0 - y_0| \leq 1.090961.$$

We also consider representively the estimate of the error made in the fifth component, which is most complicated to determine. We find

$$
\begin{aligned}
|\hat{y}_5 - y_5| &= |\hat{y}_5 - x_5^{(4)}| \leq |\hat{y}_5 - \frac{1}{\sqrt{2}}(x_5^{(3)} + x_7^{(3)})| + 1.121262 \\
&\leq |\hat{y}_5 - \frac{1}{\sqrt{2}}(x_4^{(2)} - x_5^{(2)} + x_6^{(2)} - x_7^{(2)})| + 1.121262 \\
&\leq |\hat{y}_5 - \frac{1}{\sqrt{2}}(\cos\frac{\pi}{16}(x_4^{(1)} + x_7^{(1)}) + \sin\frac{\pi}{16}(x_7^{(1)} - x_4^{(1)}) \\
&\quad + \cos\frac{3\pi}{16}(-x_5^{(1)} + x_6^{(1)}) + \sin\frac{3\pi}{16}(-x_6^{(1)} - x_5^{(1)}))| \\
&\quad + \frac{1}{\sqrt{2}}(1.048745 + 0.602587 + 1.070623 + 0.779586) + 1.121262 \\
&\approx 3.597225.
\end{aligned}
$$

The estimates for PT8 follow completely analogous, the same is true for CON8 and CON15, applying the results from Table 2 to the second factorization (3) in section 2. Note, that in factorization (3) we find the rotation matrix $\tilde{R}_2(\omega)$ instead of $R_2(\omega)$. Due to the transform (4) it is easy to see, that the estimates in Theorem 3.1 are also valid for $\tilde{R}_2(\omega)$.

□

Remark 4.2 *Note, that the error bounds for CON8 are very large compared with those of PT8. Due to the fact that the actual numerical behavior definitely is not that bad, as can be seen from the numerical results in the next section, let us have a closer look to our worst case error bounds in Theorem 4.1. Especially the first terms in both estimates crucially depend on k. From the proof of Theorem 4.1 we see, that the range of the intermediate*

results is very important for estimating the overall error. While for the PT algorithms we find $\mathbf{x}^{(1)} \in (-2^8, 2^8]^8 \cap \mathbb{Z}^8$ *if* $\mathbf{x} \in (-2^7, 2^7]^8 \cap \mathbb{Z}^8$ *for the CON algorithms we obtain for* $\tilde{\mathbf{x}}^{(1)} = H_\omega \mathbf{x}$ *that* $\tilde{\mathbf{x}}^{(1)} \in (-2^{10}, 2^{10}]$. *We see that the range of the intermediate vectors rapidly increases already at the first step. That is why in Table 2 k has to be chosen that large. Unfortunately, this leads to high error bounds. Limiting the approximation to 4, 3 or 2 bits would enforce this effect. Generally, we can say that our estimates deliver better over-all worst-case error estimates for algorithms which are based on factorizations similar to that in [12, 17, 18]. Nevertheless, also for the CON8 algorithm we can present valid over-all worst-case estimates, which perhaps could be improved by taking the special structure of the factorization (3) into account.*

5 Numerical results and final comments

In this final section, we examine and compare the numerical behavior of the integer-to-integer DCT-algorithms PTfl, PT8, PT15, CONfl, CON9 and CON15P based on factorizations (2) and (3) above. For this purpose we examine whether these algorithms approximate the exact DCT satisfactorily. Further, we compare the floating-point integer-to-integer versions PTfl and CONfl with the fixed-point versions PT8, PT15, CON8 and CON15.

We consider for 10000 random vectors, which are uniformly distributed in $(-128, 128]^8 \cap \mathbb{Z}^8$, the r-th-quantiles of the errors $\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty$ and $\|\hat{\mathbf{y}} - \mathbf{y}\|_2$ for $r = \frac{j}{10}$, $j = 1, \ldots, 10$. Recall, that for algorithms PTfl, PT15 and PT8 the integer vector $\mathbf{y}$ is the computed integer approximation of $\hat{\mathbf{y}} = 2C_8^{II}\mathbf{x}$ whereas for algorithms CONfl, CON15 and CON8 $\mathbf{y}$ is the integer approximation for $\hat{\mathbf{y}} = 2\sqrt{2}\, C_8^{II}\mathbf{x}$. After sorting the errors of the 10000 resulting vectors, the r-th-quantile is the smallest value that separates the errors into two parts; $10000r$ of the sorted errors are less than or equal to the quantile value, the other 10000(1-r) errors are greater than the quantile. The quantile for $r = 1.0$ is the maximal error occurring. In

the following table the r-th-quantiles are rounded to three decimal places:

r	PTfl	PT15	PT8	CONfl	CON15	CON8
0.1	0.545	0.546	0.569	0.504	0.507	0.563
0.2	0.623	0.621	0.647	0.603	0.604	0.684
0.3	0.699	0.699	0.734	0.679	0.679	0.783
0.4	0.769	0.765	0.802	0.740	0.740	0.861
0.5	0.836	0.835	0.879	0.801	0.801	0.958
0.6	0.904	0.902	0.969	0.860	0.860	1.064
0.7	0.995	0.989	1.069	0.914	0.913	1.177
0.8	1.099	1.096	1.191	1.016	1.021	1.320
0.9	1.270	1.257	1.369	1.167	1.167	1.537
1.0	1.970	1.970	2.246	1.595	1.594	2.692

Table 3: Quantiles in maximum norm

r	PTfl	PT15	PT8	CONfl	CON15	CON8
0.1	0.894	0.894	0.928	0.770	0.774	0.847
0.2	1.021	1.018	1.054	0.890	0.889	1.004
0.3	1.112	1.110	1.166	0.972	0.973	1.126
0.4	1.201	1.196	1.252	1.064	1.062	1.242
0.5	1.289	1.282	1.359	1.141	1.133	1.349
0.6	1.380	1.375	1.445	1.211	1.209	1.486
0.7	1.471	1.470	1.552	1.290	1.291	1.629
0.8	1.587	1.586	1.682	1.397	1.397	1.804
0.9	1.719	1.719	1.846	1.550	1.541	2.035
1	2.307	2.307	2.826	2.099	2.099	3.638

Table 4: Quantiles in Euclidian norm

For a huge number of input vectors (more than 60 %) the maximum componentwise error is smaller than 1. In these cases the algorithms compute one of the two integers being next to the exact DCT-II result in every component.

Comparing PTfl, PT15 and PT8, we see that PT15 even slightly outperforms PTfl. Further, we recognize the PT8 quantiles not to be that far away from those of PTfl and PT15.

Comparing CONfl with CON15, we observe that the associated quantiles are quiet similar, so that using CON15 instead of CONfl is admissible. Also CON8 approximates the exact DCT-II in fairly appropriate mode, but compared with PT8 it produces worse quantiles, which can be seen especially from the worst Euclidian error occurring. This is surprising, due to the fact that CONfl and CON15 outperform PTfl and PT15, an observation which is also represented in the worst case estimates above. We suppose this behavior to be caused by the

two different factorizations from Section 2, underlying the algorithms (Compare Remark 4.2). The first factorization seems to be more advantageous concerning the use of fairly inaccurate dyadic approximation.

Finally, we conclude that one can use the integer-to-integer DCT-II algorithms PT15, PT8, CON15 and CON8 in fixed-point arithmetic instead of the integer-to-integer algorithms PTfl, CONfl in floating point arithmetic, without loosing to much exactness in approximating the exact scaled DCT-II. For $j = 8$ we prefer algorithm PT8, whereas for $j = 15$ we propose algorithm CON15.

Acknowledgment: I would like to thank the referees for their constructive remarks on this paper, and in particular for pointing out the interesting ideas in [4, 5].

References

[1] V. Bhaskaran and K. Konstantinides: *Images and Video Compression Standards: Algorithms and Architectures*, Kluwer, Boston, 1997.

[2] A. R. Calderbank, I. Daubechies, W. Sweldens and B. L. Yeo: *Wavelet transform that map integers to integers*, Appl. Comput. Harmon. Anal. **5** (1998), 332 - 369.

[3] W. K. Cham and P. C. Yip: *Integer sinusoidal transforms for image processing*, Internat. J. Electron. **70** (1991), 1015 - 1030.

[4] Y.-J. Chen, S. Oraintara and T. Q. Nguyen: *Integer discrete cosine transform (IntDCT)*, Preprint 2000. See also http://surflets.mit.edu/∼yrchen/Research/.

[5] Y.-J. Chen, S. Oraintara, T. D. Tran, K. Amaratunga and T. Q. Nguyen: *Multiplierless approximation of transforms using lifting scheme and coordinate descent with adder constraint*, Proc. IEEE Internat. Conf. Acoust. Speech Signal Process.,Vol. **3**, (2002), 3136 - 3139.

[6] L. Z. Cheng, H. Xu and Y. Luo: *Integer discrete cosine transform and its fast algorithm*, Electron. Lett. **37** (2001), 64 - 65.

[7] I. Daubechies and W. Sweldens: *Factoring wavelet transforms into lifting steps*, J. Fourier Anal. Appl. **4** (1998), 247 - 269.

[8] P. Hao: *Matrix factorizations for reversible integer mapping*, IEEE Trans. Signal Process. **49** (2001), 2314 - 2324.

[9] K. Komatsu and K. Sezaki: *Reversible discrete cosine transform*, Proc. IEEE Internat. Conf. Acoust. Speech Signal Process., 1998, 1769 - 1772.

[10] K. Komatsu and K. Sezaki: *2D lossless discrete cosine transform*, Proc. IEEE Internat. Conf. Image Process., 2001, 466 - 469.

[11] J. Liang and T.D. Tran: *Fast multiplierless approximations of the DCT with the lifting scheme*, IEEE Trans. Signal Process. **49** (2001), 3032 - 3044.

[12] C. Loeffler, A. Lightenberg and G. Moschytz:*Practical fast 1-d DCT algorithms with 11 multiplications*, Proc. IEEE Internat. Conf. Acoust. Speech Signal Process., vol.2 (1989), 988 - 991.

[13] M. W. Marcellin, M. J. Gormish, A. Bilgin and M. Boliek: *An overview of JPEG-2000*, Proc. Data Compression Conf., 2000, 523 - 541.

[14] W. Philips: *Lossless DCT for combined lossy/lossless image coding*, Proc. IEEE Internat. Conf. Image Process., Vol. **3**, 1998, 871 - 875.

[15] G. Plonka: *A global method for invertible integer DCT and integer wavelet algorithms*, Appl. Comput. Harmon. Anal. **16** (2004), 90 - 110.

[16] G. Plonka and M. Tasche: *Fast and numerically stable algorithms for discrete cosine transforms*, Linear Alg. Appl., to appear.

[17] G. Plonka and M. Tasche: *Reversible integer DCT algorithms*, Appl. Comput. Harmon. Anal. **15** (2003), 70 - 88.

[18] G. Plonka and M. Tasche: *Integer DCT-II by lifting steps*, International Series in Numerical Mathematics Vol. 145 (W. Haußmann, K. Jetter, M. Reimer, J. Stöckler, eds.), Birkhäuser, Basel, 2003, 235 - 252.

[19] M. Primbs: *Integer DCT-II-Algorithmen (in German)*, Diploma thesis, Institute of Mathematics, Universität Duisburg, 2003.

[20] K. R. Rao and P. Yip: *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, Boston 1990.

[21] G. Strang: *The discrete cosine transform*, SIAM Rev. **41** (1999), 135 - 147.

[22] T.D. Tran: *The BinDCT: Fast Multiplierless approximation of the DCT*, IEEE Signal Process. Lett. **7** (2000), 141 - 144.

[23] Y. Zeng, L. Cheng, G. Bi and A.C. Kot: *Integer DCT´s and fast algorithms*, IEEE Trans. Signal Process. **49** (2001), 2774 - 2782.