Reversible integer DCT algorithms

Gerlind Plonka and Manfred Tasche

Affiliations:

Gerlind Plonka
Institute of Mathematics
Gerhard-Mercator-University of Duisburg
D – 47048 Duisburg
Germany

Manfred Tasche
Department of Mathematics
University of Rostock
D – 18051 Rostock
Germany

E–*mail addresses*:
plonka@math.uni-duisburg.de
manfred.tasche@mathematik.uni-rostock. de

Proposed running head: Reversible integer DCT algorithms

Author for correspondence:

Gerlind Plonka
Institute of Mathematics
Gerhard-Mercator-University of Duisburg
D – 47048 Duisburg
Germany
Email: plonka@math.uni-duisburg.de
Telephone: 49 203 379 2677
Fax: 49 203 379 2689

Abstract

Integer DCTs have important applications in lossless coding. An 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. In image compression, the DCT of type II (DCT–II) is of special interest. In this paper we present two new approaches to construct reversible integer DCT–II. Our methods are based on a factorization of the cosine matrix of type II into a product of sparse, orthogonal matrices. Up to some permutations, each matrix factor is a block–diagonal matrix, where every block is an orthogonal matrix of order 2. Hence one has to construct only integer transforms of length 2. The first approach uses expansion factors and rounding–off for the construction of integer transforms of length 2. The second approach works with lifting steps and rounding–off. This allows the construction of new (one– and two–dimensional) integer DCT–II algorithms. For simplicity, the most interesting case $n = 8$ is considered in detail. Further, uniform bounds of the truncation errors are given for all integer DCT–II algorithms proposed in this paper.

Mathematics Subject Classification 2000. 65T50, 65G50, 15A23, 94A08.

Key words. Discrete cosine transform, lossless coding, data compression, factorization of cosine matrix, expansion factors, lifting steps, rounding–off, integer DCT, reversible integer DCT, truncation error, error estimate.

1 Introduction

The discrete cosine transform of type II (DCT–II) has found a wide range of applications in signal and image processing (see [11, 13]), especially in image compression. It has become the heart of international standards in image compression such as JPEG and MPEG (see [1]). In some applications, the input data consist of integer vectors or integer matrices. Then the output of DCT–II algorithm consists no longer of integers. For lossless coding it would be of great interest to be able to characterize the output completely again with integers. Lossless coding schemes are hardly based on integer DCTs which have been studied in recent years (see [3, 6, 9, 14, 16]). Especially, integer DCTs of length 8 and 16 (see [9]) and integer wavelets (see [2]) have been proposed.
A reversible integer DCT–II of length n is a nonlinear, left-invertible mapping which acts on $\mathbb{Z}^n$ and approximates the classical DCT–II of length n. Integer DCT–II possesses some features of the classical DCT–II, whereas its computational cost is not higher than in the classical case.

One method for developing an integer DCT algorithm is based on the idea to approximate the components of the cosine matrix C_n^{II} by dyadic rationals (see e.g. [14]), paying attention that the symmetry relations are kept. This method destroys the orthonormality of the cosine matrix and the challenge is to find an invertible approximation $\tilde{C}_n$ of C_n^{II} such that its inverse $\tilde{C}_n^{-1}$ again only consists of dyadic rationals. For this reason, suitable approximations of the cosine matrix have only been given for special lengths $n = 8$ and $n = 16$.
In [16] a factorization of the transform matrix into products of so–called lifting matrices and simple matrices is applied. Here 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. Then the noninteger entries of the lifting matrices are rounded to dyadic rationals, and the inverse matrix factors are easy to determine. This method has the advantage that it works for arbitrary radix–2 lengths.
Due to the rounding of matrix entries, the methods in [14, 16] can lead to high errors, if one compares the integer DCT output with the classical DCT result, especially if the range for the components of the input vector is large. Explicit error estimates for these algorithms have not been considered.
In this paper, we present two new approaches that lead to reversible integer DCTs. The two approaches are independent of each other. Note that in both cases we are not building integer DCTs in integer arithmetic. Thus the computations are still done with floating point numbers, but the result is guaranteed to be an integer and the invertibility is preserved. In software applications, this should not affect speed, as in many of today's microprocessors floating point and integer computations are virtually equally fast.
Our algorithms are based on a factorization of C_n^{II} into sparse orthogonal matrices of simple structure. By suitable permutations, each matrix factor can be transferred to a block–diagonal matrix, where every block is an orthogonal matrix of order 2. Now the idea for construction of integer DCTs of radix–2 length n is very simple. For each block R_2 of order 2 and for arbitrary $\mathbf{x} \in \mathbb{Z}^2$, find a suitable integer approximation of $R_2\mathbf{x}$ such that this process is reversible.
Using the factorizations of the corresponding cosine matrices in [10], the applied methods can easily be transferred to the DCT–IV and the DCT–I.

The paper is organized as follows. In Section 2 we introduce cosine matrices of type II and IV and we sketch some recent results [10] on the recursive factorization of these matrices into products of sparse, orthogonal matrices. In Section 3, we present two different approaches to integer transforms of length 2. The first approach (see Subsection 3.1) is inspired by the construction of integer wavelet transforms due to [2]. The main result in Theorem 3.4 presents new error estimates for this expansion factor method. A second approach to integer transforms of length 2 is taken in Subsection 3.2. Applying the lifting technique (see [2, 4, 6, 16]) and rounding–off, we construct an integer approximation of $R_2\mathbf{x}$ for arbitrary $\mathbf{x} \in \mathbb{Z}^2$ and estimate the truncation error (see Theorem 3.8).
The results of Section 2 and Section 3 are applied to integer DCT–II of length 8 (in Section 4) and to two–dimensional integer DCT–II of size 8×8 (in Section 5). While we consider the most interesting case $n = 8$ in detail (cf. [7, 5]), our results are directly applicable to reversible integer DCT–II of *arbitrary* radix–2 length. We propose several algorithms for the integer DCT–II and its inverse, which possess low complexity. Further, we estimate the truncation errors of these integer DCT–II algorithms by uniform bounds. Finally, in Section 6 we consider the numerical behaviour of the integer DCT–II. Note that in the JPEG–2000 proposal [8], the use of the integer DCT–II for lossless image coding is recommended.

2 Factorization of cosine matrices

Let $n \geq 2$ be a given integer. In the following, we consider *cosine matrices* of type II and IV with order n which are defined by

$$C_n^{II} := \sqrt{\tfrac{2}{n}} \left(\epsilon_n(j) \cos \tfrac{j(2k+1)\pi}{2n} \right)_{j,k=0}^{n-1}, \tag{2.1}$$

$$C_n^{IV} := \sqrt{\tfrac{2}{n}} \left(\cos \tfrac{(2j+1)(2k+1)\pi}{4n} \right)_{j,k=0}^{n-1}, \tag{2.2}$$

where $\epsilon_n(0) := \sqrt{2}/2$ and $\epsilon_n(j) := 1$ for $j \in \{1, \dots, n-1\}$. In our notation a subscript of a matrix denotes the corresponding order, while a superscript signifies the "type" of the matrix. Observe that these matrices are orthogonal (see e.g. [11], pp. 13 – 14, [12, 13]). The *discrete cosine transforms* of type II (DCT–II) and of type IV (DCT–IV) are linear mappings of $\mathbb{R}^n$ onto $\mathbb{R}^n$, which are generated by C_n^{II} and C_n^{IV}, respectively. In [10], simple split–radix algorithms are proposed for these transforms of radix–2 length, which are based on factorizations of C_n^{II} and C_n^{IV} into products of sparse, orthogonal matrices. In this paper, we want to use these factorizations in order to derive reversible integer DCTs, which are very close to the original DCT and map integer vectors to integer vectors. Naturally, these integer transforms are not longer linear mappings.
Let us recall the factorizations for C_n^{II} and C_n^{IV} from [10]. First, we introduce some notations. Let I_n denote the identity matrix and $J_n := (\delta(j + k - n + 1))_{j,k=0}^{n-1}$ the counteridentity matrix, where δ means the Kronecker symbol. Blanks in a matrix indicate zeros or blocks of zeros. The direct sum of two matrices A, B is defined to be a block diagonal matrix $A \oplus B := \operatorname{diag}(A, B)$. Let $\Sigma_n := \operatorname{diag}((-1)^k)_{k=0}^{n-1}$ be the diagonal sign matrix.
For even $n \geq 4$, P_n denotes the *even–odd permutation matrix* (or 2*–stride permutation*

matrix) defined by

$$P_n \mathbf{x} := (x_0, x_2, \ldots, x_{n-2}, x_1, x_3, \ldots, x_{n-1})^T, \qquad \mathbf{x} = (x_j)_{j=0}^{n-1}.$$

Note that $P_n^{-1} = P_n^T$ is the n_1–stride permutation matrix with $n_1 := n/2$. Further, let $Q_n := (I_{n_1} \oplus J_{n_1})\, P_n$ be a *modified even–odd permutation matrix* with

$$Q_n \mathbf{x} := (x_0, x_2, \ldots, x_{n-2}, x_{n-1}, x_{n-3}, \ldots, x_1)^T.$$

Theorem 2.1 *Let $n \geq 4$ be an even integer.*
(i) *The matrix C_n^{II} can be factorized in the form*

$$C_n^{II} = P_n^T\, (C_{n_1}^{II} \oplus C_{n_1}^{IV})\, T_n(0) \tag{2.3}$$

with the orthogonal matrix

$$T_n(0) := \tfrac{1}{\sqrt{2}} \begin{pmatrix} I_{n_1} & J_{n_1} \\ I_{n_1} & -J_{n_1} \end{pmatrix}.$$

(ii) *The matrix C_n^{IV} can be factorized in the form*

$$C_n^{IV} = P_n^T\, A_n(1)\, (C_{n_1}^{II} \oplus C_{n_1}^{II})\, T_n(1), \tag{2.4}$$

where

$$A_n(1) := \tfrac{1}{\sqrt{2}} \begin{pmatrix} \sqrt{2} & & & \\ & I_{n_1-1} & I_{n_1-1} & \\ & I_{n_1-1} & -I_{n_1-1} & \\ & & & -\sqrt{2} \end{pmatrix} (I_{n_1} \oplus \Sigma_{n_1} J_{n_1})$$

is a modified addition matrix and $T_n(1) := (I_{n_1} \oplus \Sigma_{n_1})\, T_n'(1)$ with the cross–shaped twiddle matrix

$$T_n'(1) = \begin{pmatrix} \cos(\frac{\pi}{8n}) & & & & & & \sin(\frac{\pi}{8n}) \\ & \cos(\frac{3\pi}{8n}) & & & & \sin(\frac{3\pi}{8n}) & \\ & & \ddots & & \iddots & & \\ & & & \cos(\frac{(n-1)\pi}{8n}) \;\; \sin(\frac{(n-1)\pi}{8n}) & & & \\ & & & -\sin(\frac{(n-1)\pi}{8n}) \;\; \cos(\frac{(n-1)\pi}{8n}) & & & \\ & & \iddots & & \ddots & & \\ & -\sin(\frac{3\pi}{8n}) & & & & \cos(\frac{3\pi}{8n}) & \\ -\sin(\frac{\pi}{8n}) & & & & & & \cos(\frac{\pi}{8n}) \end{pmatrix}.$$

The two matrices $A_n(1)$ and $T_n(1)$ are orthogonal.

For a proof of these factorizations we refer to [10]. Note that

$$T_n(0) = (I_{n_1} \oplus (-J_{n_1}))\, Q_n \left(\bigoplus_{k=0}^{n_1-1} R_2(\tfrac{\pi}{4}) \right) Q_n^T, \tag{2.5}$$

$$T_n(1) = (I_{n_1} \oplus \Sigma_{n_1})\, Q_n \left(\bigoplus_{k=0}^{n_1-1} R_2(\tfrac{(2k+1)\pi}{8n}) \right) Q_n^T \tag{2.6}$$

$$A_n(1) = \widetilde{Q}_n\, \Sigma_n \left(I_2 \oplus \bigoplus_{k=1}^{n_1-1} R_2(\tfrac{\pi}{4}) \right) \widetilde{Q}_n^T\, (I_{n_1} \oplus \Sigma_{n_1} J_{n_1}), \tag{2.7}$$

with the *rotation matrix*

$$R_2(\omega) := \begin{pmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{pmatrix}, \tag{2.8}$$

and with $\widetilde{Q}_n := (I_{n_1} \oplus V_{n_1})P_n$, where V_n is the shift matrix determined by

$$V_n \mathbf{x} = (x_1, x_2, \ldots, x_{n-1}, x_0)^T.$$

Hence, up to convenient permutations and changes of sign, the matrices (2.5) – (2.7) can be represented as block–diagonal matrices, where each block is a rotation matrix of order 2. The following constructions of integer DCT–II are based on this essential fact.

3 Integer transforms of length 2

Let now $n = 2^t$ and $n_j := 2^{t-j}$, $j = 0, \ldots, t-1$. If the formulas (2.3) and (2.4) are applied recursively, then we obtain factorizations of the cosine matrices C_n^{II} and C_n^{IV}, where all matrix factors are orthogonal block matrices with blocks being permutation matrices or matrices of the form I_{n_j}, $A_{n_j}(1)$, $T_{n_j}(0)$ and $T_{n_j}(1)$ (see [10]). In particular, all matrix factors are sparse, i.e., they possess two nonzero entries at most in each row and each column. Hence, by suitable permutations, the matrix factors can be transferred to block–diagonal matrices, where each block is an orthogonal matrix of order 2 (see (2.5) – (2.7)).

The main idea to obtain an integer DCT is now as follows. For a given invertible (2×2)-matrix H_2 and for arbitrary $\mathbf{x} \in \mathbb{Z}^2$, find a suitable integer approximation of $H_2\mathbf{x}$ such that this process is reversible. The simple structure of the matrix factors of C_n^{II} implies that we need to find a suitable solution only for orthogonal matrices $H_2 = R_2(\omega)$. In the following we want to present two different approaches to that problem. These two approaches are related to the two approaches given in [2] in order to find invertible wavelet transforms that map integers to integers.

3.1 Integer transforms with expansion factors

We use the following notations. For $a \in \mathbb{R}$ let $\lfloor a \rfloor := \max\{x \le a;\, x \in \mathbb{Z}\}$, $\lceil a \rceil := \min\{x \ge a;\, x \in \mathbb{Z}\}$, and $\{a\} := a - \lfloor a \rfloor \in [0,1)$. Then $\{a\}$ is the *noninteger part* of a. Further let $\mathrm{rd}\, a := \lfloor a + 1/2 \rfloor$ be the integer next to a. We apply these operations to vectors $\mathbf{a} = (a_0,\, a_1)^T \in \mathbb{R}^2$ componentwisely, i.e. $\lfloor \mathbf{a} \rfloor := (\lfloor a_0 \rfloor, \lfloor a_1 \rfloor)^T \in \mathbb{Z}^2$, $\lceil \mathbf{a} \rceil := (\lceil a_0 \rceil, \lceil a_1 \rceil)^T \in \mathbb{Z}^2$, $\mathrm{rd}\,\mathbf{a} := (\mathrm{rd}\, a_0, \mathrm{rd}\, a_1)^T \in \mathbb{Z}^2$, and $\{\mathbf{a}\} := (\{a_0\}, \{a_1\})^T \in [0,1)^2$.
Let $H_2 \in \mathbb{R}^{2\times 2}$ be an invertible matrix of order 2 and $\mathbf{x} = (x_0,\, x_1)^T \in \mathbb{Z}^2$. We look for a vector $\mathbf{y} = (y_0,\, y_1)^T \in \mathbb{Z}^2$ such that $\mathbf{y}$ is a good approximation of $H_2\mathbf{x}$ and

$$\mathbf{x} = \lceil H_2^{-1}(\mathbf{y} - \tfrac{1}{2}\mathbf{1}) \rceil \tag{3.1}$$

with $\mathbf{1} = (1,1)^T$ is satisfied. If H_2 is an expanding matrix then $\mathbf{x}$ will be close to $H_2^{-1}\mathbf{y}$. We denote by $H_2([0,1)^2) := \{H_2\mathbf{r};\, \mathbf{r} \in [0,1)^2\}$ the image of $[0,1)^2$ under the linear mapping generated by H_2.

Lemma 3.1 *Let $H_2 \in \mathbb{R}^{2\times 2}$ be an invertible matrix satisfying the* expansion condition

$$[0,\,1)^2 \subseteq \bigcup_{\mathbf{k}\in\mathbb{Z}^2} (H_2([0,\,1)^2) + \mathbf{k}). \tag{3.2}$$

Then for all $\mathbf{x} \in \mathbb{Z}^2$ there exists a vector $\mathbf{y} \in \mathbb{Z}^2$ approximating $\hat{\mathbf{y}} := H_2\mathbf{x}$ with property (3.1). Such a vector $\mathbf{y} \in \mathbb{Z}^2$ has the form

$$\mathbf{y} = \mathrm{rd}\,(H_2\mathbf{x}) + \mathbf{k}^0,$$

if $\{H_2\mathbf{x} + \frac{1}{2}\mathbf{1}\} \in [0,\,1)^2 \cap (H_2([0,\,1)^2) + \mathbf{k}^0)$ for some $\mathbf{k}^0 = (k_0,\,k_1)^T \in \mathbb{Z}^2$. Further, the error estimates

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &\leq \sqrt{(\tfrac{1}{2} + |k_0|)^2 + (\tfrac{1}{2} + |k_1|)^2}, \\ \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty &\leq \max\{\tfrac{1}{2} + |k_0|,\, \tfrac{1}{2} + |k_1|\} \end{aligned}$$

hold.

Proof. For a fixed $\mathbf{x} \in \mathbb{Z}^2$ we choose $\mathbf{y} = \lfloor H_2\mathbf{x} + \frac{1}{2}\mathbf{1}\rfloor + \mathbf{k}^0$ such that

$$\{H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\} = H_2\mathbf{r} + \mathbf{k}^0$$

with $\mathbf{k}^0 \in \mathbb{Z}^2$ and some $\mathbf{r} \in [0,\,1)^2$. The existence of $\mathbf{k}^0$ and $\mathbf{r}$ is ensured by the expansion condition (3.2). Then we obtain

$$\begin{aligned} \mathbf{y} &= \lfloor H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\rfloor + \mathbf{k}^0 = \lfloor H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\rfloor + \{H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\} - H_2\mathbf{r} \\ &= H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1} - H_2\mathbf{r} \end{aligned}$$

and hence

$$\mathbf{x} = H_2^{-1}(\mathbf{y} - \tfrac{1}{2}\mathbf{1}) + \mathbf{r}.$$

Since $\mathbf{x} \in \mathbb{Z}^2$ and $\mathbf{r} \in [0,\,1)^2$, it follows that

$$\mathbf{x} = \lceil\mathbf{x}\rceil = \lceil H_2^{-1}(\mathbf{y} - \tfrac{1}{2}\mathbf{1}) + \mathbf{r}\rceil = \lceil H_2^{-1}(\mathbf{y} - \tfrac{1}{2}\mathbf{1})\rceil.$$

By $\{H_2\mathbf{x} + \frac{1}{2}\mathbf{1}\} \in [0,\,1)^2$ we can estimate

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 = \|H_2\mathbf{x} - \mathrm{rd}\,(H_2\mathbf{x}) - \mathbf{k}^0\|_2 \leq \sqrt{(\tfrac{1}{2} + |k_0|)^2 + (\tfrac{1}{2} + |k_1|)^2}$$

and

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty \leq \max\{\tfrac{1}{2} + |k_0|,\, \tfrac{1}{2} + |k_1|\}.$$

For special matrices H_2, these simple error estimates can be improved. *q.e.d.*

If an invertible matrix H_2 does not satisfy the expansion condition (3.2), then we apply an *expansion factor* $\alpha > 0$ such that $\alpha\, H_2$ (instead of H_2) satisfies the expansion condition (3.2). Since $H_2([0,\,1)^2)$ is a quadrangle with the area $|\det H_2|$, we can find a suitable expansion factor for each invertible matrix H_2.

Example 3.2 *The butterfly matrix*

$$H_2 := \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$$

satisfies the expansion condition (3.2)*, but the matrix* $R_2(\frac{\pi}{4}) = \frac{1}{\sqrt{2}} H_2$ *does not satisfy* (3.2) (*see Figure* 1)*.*

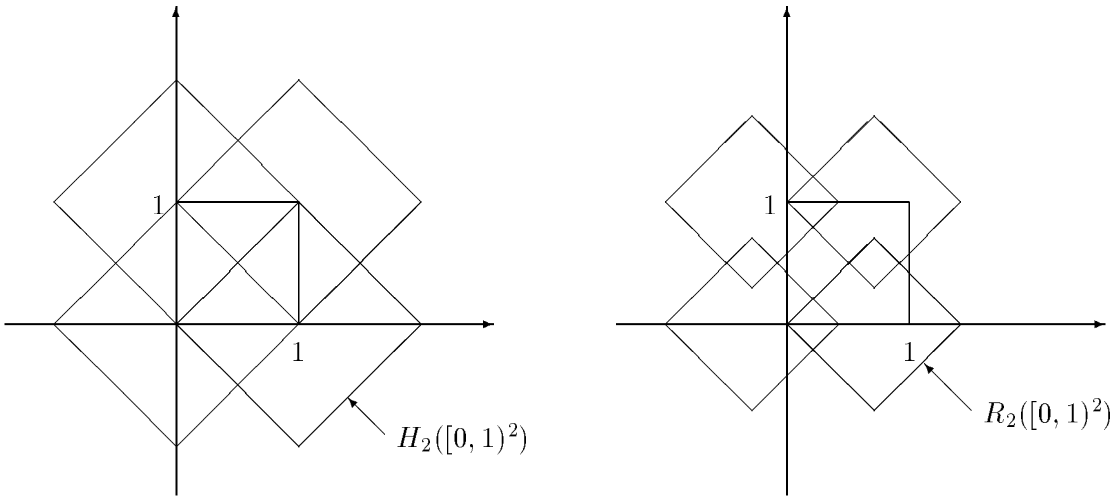

Figure 1. Expansion condition for $H_2 = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$ (left) and $R_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$ (right)

We want to apply the above method to the matrix factors in Theorem 2.1. Let us consider rotation matrices $R_2(\omega)$ (see (2.8)) which appear in $T_n(1)$ with $\omega = \frac{(2k+1)\pi}{8n}$, $k = 0, \ldots, n/2 - 1$ (see (2.6)). Further, the matrices $T_n(0)$ and $A_n(1)$ consist of the matrices $R_2(\frac{\pi}{4})$ (see (2.5) - (2.7)).

The rotation matrices $R_2(\omega)$ are orthogonal, and $R_2(\omega)([0,\, 1)^2)$ is a square with area 1, namely the square $[0,\, 1)^2$ rotated by the angle $-\omega$ around the origin $(0,\, 0)^T$. Hence, the expansion condition (3.2) is not satisfied for $\omega \in (0,\, \frac{\pi}{4}]$ and we need to look for suitable expansion factors. We are especially interested in small expansion factors for $R_2(\omega)$. This is due to the demand that for arbitrary $\mathbf{x} \in \mathbb{Z}^n$, we want to get an integer approximation of $\alpha\, C_n^{II} \mathbf{x}$ with a constant α as small as possible.

Theorem 3.3 *Let* $R_2(\omega)$ *with* $\omega \in [0, \frac{\pi}{4}]$ *be a rotation matrix of the form* (2.8)*. Then*

$$H_2 := (\cos\omega + \sin\omega)\, R_2(\omega)$$

satisfies the condition (3.2)*, i.e., all factors* $\alpha \geq (\cos\omega + \sin\omega)$ *can serve as expansion factors for* $R_2(\omega)$*. Further,* $\alpha = (\cos\omega + \sin\omega)$ *is the smallest possible expansion factor for* $R_2(\omega)$*.*

Proof. We show that (3.2) is satisfied for $H_2 = (\cos\omega + \sin\omega)\, R_2(\omega)$. For this, we only need to prove that for all $\mathbf{z} = (z_0,\, z_1)^T \in [0,\, 1)^2$ there exists a vector $\mathbf{k} \in \mathbb{Z}^2$ such that $\mathbf{r} = H_2^{-1}(\mathbf{z} - \mathbf{k})$ is contained in $[0,\, 1)^2$. We consider the following pairwise disjoint polygons $S_1(\omega)$, $S_2(\omega)$, $S_3(\omega)$ (see Figure 2):

$$\begin{aligned} S_1(\omega) &:= \{\mathbf{z} \in [0,1)^2 : z_0 \geq z_1 \tan\omega\}, \\ S_2(\omega) &:= \{\mathbf{z} \in [0,1)^2 : z_0 < z_1 \tan\omega,\ z_1 < 1 - z_0 \tan\omega\}, \\ S_3(\omega) &:= \{\mathbf{z} \in [0,1)^2 : z_0 < z_1 \tan\omega,\ 1 - z_0 \tan\omega \leq z_1\}. \end{aligned} \tag{3.3}$$

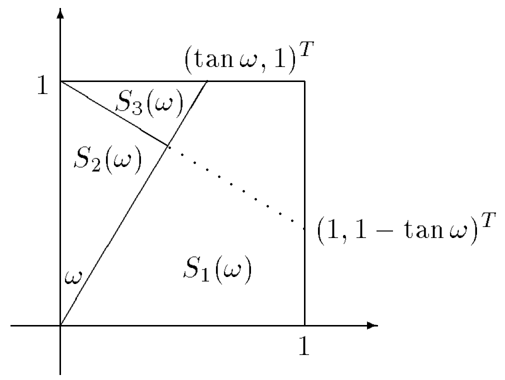

Figure 2. Polygonal partition of $[0,1)^2$ with $\tan\omega = 0.6$

Then we have $S_1(\omega) \cup S_2(\omega) \cup S_3(\omega) = [0,\,1)^2$ such that $S_1(\omega)$, $S_2(\omega)$, $S_3(\omega)$ form a partition of $[0,\,1)^2$. Observe that $H_2^{-1} = (\cos\omega + \sin\omega)^{-1} R_2(\omega)^T$. For $\mathbf{z} \in S_1(\omega)$, choose $\mathbf{k}_1 = (0,0)^T$. Then

$$\mathbf{r} = H_2^{-1}(\mathbf{z} - \mathbf{k}_1) = \tfrac{1}{\cos\omega+\sin\omega} \begin{pmatrix} z_0\,\cos\omega - z_1\,\sin\omega \\ z_0\,\sin\omega + z_1\,\cos\omega \end{pmatrix}$$

is contained in $[0,\,1)^2$. For $\mathbf{z} \in S_2(\omega)$, choose $\mathbf{k}_2 = (-1,0)^T$. Then we obtain

$$\mathbf{r} = H_2^{-1}(\mathbf{z} - \mathbf{k}_2) = \tfrac{1}{\cos\omega+\sin\omega} \begin{pmatrix} z_0\,\cos\omega - z_1\,\sin\omega + \cos\omega \\ z_0\,\sin\omega + z_1\,\cos\omega + \sin\omega \end{pmatrix} \in [0,\,1)^2.$$

Finally, for $\mathbf{z} \in S_3(\omega)$, we choose $\mathbf{k}_3 = (0,1)^T$ and find

$$\mathbf{r} = H_2^{-1}(\mathbf{z} - \mathbf{k}_3) = \tfrac{1}{\cos\omega+\sin\omega} \begin{pmatrix} z_0\,\cos\omega - z_0\,\sin\omega + \sin\omega \\ z_0\,\sin\omega + z_1\,\cos\omega - \cos\omega \end{pmatrix} \in [0,\,1)^2.$$

If the expansion factor α is chosen to be smaller than $\cos\omega + \sin\omega$ and if $H_2' := \alpha\, H_2$, then the point $(1,1)^T$ is not contained in the closure of $H_2'([0,\,1)^2)$ and the three squares $H_2'([0,\,1)^2)$, $H_2'([-1,\,0) \times [0,\,1))$ and $H_2'([0,\,1) \times [1,\,2))$ do not completely cover $[0,1)^2$ (see Figure 3). In this case, the expansion condition (3.2) is not satisfied. *q.e.d.*

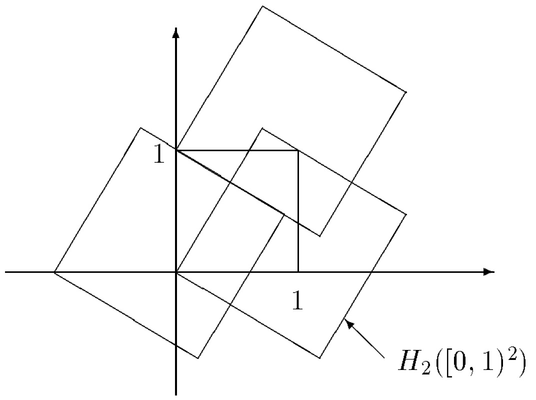

Figure 3. Expansion condition for $H_2 = (\cos\omega + \sin\omega) R_2(\omega)$ with $\tan\omega = 0.6$

Note that $\cos\omega + \sin\omega \le \sqrt{2}$ for all $\omega \in (0,\,\pi/4]$. Applying Theorem 3.3 to the "submatrices" $R_2((2k+1)\pi/(8n))$ of $T_n(1)$ (see (2.6)) we can use the uniform expansion factor $\sqrt{2}$ for all $k = 0, \ldots, n_1 - 1$.

Theorem 3.4 *Let $H_2 := \sqrt{2}\, R_2(\omega)$ with $\omega \in (0, \frac{\pi}{4})$ be a scaled rotation matrix and let $\mathbf{k}_1 = (0,0)^T$, $\mathbf{k}_2 = (-1,0)^T$, $\mathbf{k}_3 = (0,1)^T$.*
Then for arbitrary $\mathbf{x} \in \mathbb{Z}^2$, a suitable integer approximation $\mathbf{y} \in \mathbb{Z}^2$ of $\hat{\mathbf{y}} := H_2\mathbf{x}$ is

$$\mathbf{y} = \mathrm{rd}\,(H_2\mathbf{x}) + \mathbf{k}_j \quad \text{if } \{H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\} \in S_j(\omega), \quad (j \in \{1,\, 2,\, 3\}), \tag{3.4}$$

where $S_j(\omega)$ are given in (3.3). This procedure is reversible for all $\mathbf{x} \in \mathbb{Z}^2$, i.e. (3.1) is valid, and the error estimates

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &\le \tfrac{\sqrt{2}}{2}\sqrt{1 + \sin(2\omega)} \le 1, \\ \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty &\le \tfrac{1}{2}\,(1 + \sin(2\omega)) \le 1 \end{aligned} \tag{3.5}$$

hold.

Proof. By Theorem 3.3, the matrix $H_2 = \sqrt{2}\, R_2(\omega)$ satisfies the expansion condition (3.2). Since $\{H_2\mathbf{x} + \frac{1}{2}\mathbf{1}\} \in [0,\, 1)^2$, there exists a unique index $j \in \{1,\, 2,\, 3\}$ with $\{H_2\mathbf{x} + \frac{1}{2}\mathbf{1}\} \in S_j(\omega)$. Therefore, $\mathbf{y}$ is uniquely defined by (3.4). Applying Lemma 3.1, it follows that (3.1) holds for all $\mathbf{x} \in \mathbb{Z}^2$.
For $\{H_2\mathbf{x} + \frac{1}{2}\mathbf{1}\} \in S_j(\omega)$, we can estimate

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &= \|H_2\mathbf{x} - \lfloor H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\rfloor - \mathbf{k}_j\|_2 = \|\{H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_j\|_2 \\ &\le \sup_{\mathbf{z} \in S_j(\omega)} \|\mathbf{z} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_j\|_2. \end{aligned}$$

Let $\mathbf{p}$ denote the intersection point of the lines $z_0 = (\tan\omega)\, z_1$ and $z_1 = 1 - (\tan\omega)\, z_0$ (see Figure 2). We find that $\mathbf{p} = \frac{1}{2}\,(\sin(2\omega),\, 1 + \cos(2\omega))^T$ and hence

$$\begin{aligned} \sup_{\mathbf{z} \in S_1(\omega)} \|\mathbf{z} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_1\|_2 &= \sup_{\mathbf{z} \in S_1(\omega)} \|\mathbf{z} - \tfrac{1}{2}\mathbf{1}\|_2 = \tfrac{\sqrt{2}}{2}, \\ \sup_{\mathbf{z} \in S_2(\omega)} \|\mathbf{z} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_2\|_2 &= \sup_{\mathbf{z} \in S_2(\omega)} \|\mathbf{z} + \tfrac{1}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix}\|_2 = \|\mathbf{p} + \tfrac{1}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix}\|_2 \\ &= \tfrac{\sqrt{2}}{2}\sqrt{1 + \sin(2\omega)}, \\ \sup_{\mathbf{z} \in S_3(\omega)} \|\mathbf{z} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_3\|_2 &= \sup_{\mathbf{z} \in S_3(\omega)} \|\mathbf{z} - \tfrac{1}{2}\begin{pmatrix} 1 \\ 3 \end{pmatrix}\|_2 \\ &= \max\left\{ \|\tfrac{1}{2}\begin{pmatrix} \sin(2\omega) - 1 \\ \cos(2\omega) - 2 \end{pmatrix}\|_2,\ \|\tfrac{1}{2}\begin{pmatrix} -1 \\ -1 \end{pmatrix}\|_2 \right\} \\ &= \max\{\tfrac{\sqrt{2}}{2}\sqrt{3 - \sin(2\omega) - 2\cos(2\omega)},\ \tfrac{\sqrt{2}}{2}\}. \end{aligned}$$

Since for $\omega \in (0, \pi/4]$ we have $1 + \sin(2\omega) > 3 - \sin(2\omega) - 2\cos(2\omega)$, it follows that

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 \le \tfrac{\sqrt{2}}{2}\sqrt{1 + \sin(2\omega)}.$$

Analogously we find

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty &\le \sup_{\mathbf{z} \in S_j(\omega)} \|\mathbf{z} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_j\|_\infty \\ &\le \max\{\tfrac{1}{2},\ \tfrac{1}{2}(1 + \sin(2\omega)),\ 1 - \tfrac{1}{2}\cos(2\omega)\} = \tfrac{1}{2}(1 + \sin(2\omega)). \end{aligned}$$

This completes the proof. *q.e.d.*

Remark 3.5 *The special values for the errors* $\|\hat{\mathbf{y}}-\mathbf{y}\|_2$ *and* $\|\hat{\mathbf{y}}-\mathbf{y}\|_\infty$ *with* $\hat{\mathbf{y}} = \sqrt{2}\, R_2(\omega)\mathbf{x}$ *via the expansion factor method for* $\omega \in \{\frac{\pi}{8}, \frac{\pi}{16}, \frac{3\pi}{16}\}$ *follow by inserting into formulas* (3.5). *In particular we obtain*

$$\|\hat{\mathbf{y}}-\mathbf{y}\|_2 \le \begin{cases} 0.923879 & \textit{for } \omega = \frac{\pi}{8}, \\ 0.831470 & \textit{for } \omega = \frac{\pi}{16}, \\ 0.980785 & \textit{for } \omega = \frac{3\pi}{16}, \end{cases} \qquad \|\hat{\mathbf{y}}-\mathbf{y}\|_\infty \le \begin{cases} 0.853553 & \textit{for } \omega = \frac{\pi}{8}, \\ 0.691342 & \textit{for } \omega = \frac{\pi}{16}, \\ 0.961940 & \textit{for } \omega = \frac{3\pi}{16}. \end{cases}$$

Note that $\sqrt{2}\, R_2(\frac{\pi}{4})$ *is the butterfly matrix such that* $\hat{\mathbf{y}} = \mathbf{y} = \sqrt{2}\, R_2(\frac{\pi}{4})\mathbf{x}$. *The butterfly operation is left–invertible on* $\mathbb{Z}^2$.

Enlarging the expansion factors we can avoid a case study as in Theorem 3.4.

Theorem 3.6 *Let* $H_2 := 2\, R_2(\omega)$ *with* $\omega \in (0, \frac{\pi}{4})$ *be a scaled rotation matrix and let* $\mathbf{k}_2 = (-1, 0)^T$.
Then for arbitrary $\mathbf{x} \in \mathbb{Z}^2$, *a suitable integer approximation* $\mathbf{y} \in \mathbb{Z}^2$ *of* $\hat{\mathbf{y}} := H_2\mathbf{x}$ *is*

$$\mathbf{y} = \mathrm{rd}\,(H_2\mathbf{x}) + \mathbf{k}_2. \tag{3.6}$$

This procedure is reversible for all $\mathbf{x} \in \mathbb{Z}^2$, *i.e.,* (3.1) *is valid, and the error estimates*

$$\|\hat{\mathbf{y}}-\mathbf{y}\|_2 \le \sqrt{\tfrac{5}{2}}, \qquad \|\hat{\mathbf{y}}-\mathbf{y}\|_\infty \le \tfrac{3}{2}$$

hold.

Proof. By Theorem 3.3, the matrix H_2 satisfies the expansion condition (3.2). Further we have $[0,\, 1)^2 \subseteq (\mathbf{k}_2 + H_2([0,\, 1)^2)$. Using Lemma 3.1, it follows that (3.1) is true for all $\mathbf{x} \in \mathbb{Z}^2$.
By

$$\begin{aligned} \|\hat{\mathbf{y}}-\mathbf{y}\|_2 &= \|H_2\mathbf{x} - \lfloor H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\rfloor - \mathbf{k}_2\|_2 = \|\{H_2\mathbf{x} + \tfrac{1}{2}\mathbf{1}\} - \tfrac{1}{2}\mathbf{1} - \mathbf{k}_2\|_2 \\ &\le \sup_{\mathbf{z}\in[0,1)^2} \|\mathbf{z} + \tfrac{1}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix}\|_2 \le \sqrt{\tfrac{5}{2}} \end{aligned}$$

and

$$\|\hat{\mathbf{y}}-\mathbf{y}\|_\infty = \sup_{\mathbf{z}\in[0,1)^2} \|\mathbf{z} + \tfrac{1}{2}\begin{pmatrix} 1 \\ -1 \end{pmatrix}\|_\infty = \tfrac{3}{2},$$

we obtain the error estimates. *q.e.d.*

Remark 3.7 *1. An integer transform for "submatrices" of* $\sqrt{2}\, T_n(0)$ *(see* (2.5)*) is very simple, since* $\sqrt{2}\, R_2(\omega)$ *contains only* ± 1 *as entries. The same is true for the most "submatrices" of* $\sqrt{2}\, A_n(1)$ *(see* (2.7)*).*
2. In the special case that a "submatrix" is of the form $\sqrt{2}\, I_2$ *(occurring in* $\sqrt{2}\, A_n(1)$*), we can choose*

$$\mathbf{y} = \lfloor \sqrt{2}\,\mathbf{x} + \tfrac{1}{2}\mathbf{1}\rfloor = \mathrm{rd}\,(\sqrt{2}\,\mathbf{x}), \qquad \mathbf{x} \in \mathbb{Z}^2$$

and this process can be reversed by

$$\mathbf{x} = \lceil \tfrac{1}{\sqrt{2}}(\mathbf{y} - \tfrac{1}{2}\mathbf{1})\rceil.$$

Indeed,

$$\lceil \tfrac{1}{\sqrt{2}}(\lfloor \sqrt{2}\,\mathbf{x} + \tfrac{1}{2}\mathbf{1}\rfloor - \tfrac{1}{2}\mathbf{1})\rceil = \lceil \tfrac{1}{\sqrt{2}}(\sqrt{2}\,\mathbf{x} - \{\sqrt{2}\,\mathbf{x} + \tfrac{1}{2}\mathbf{1}\})\rceil = \lceil \mathbf{x} - \tfrac{1}{\sqrt{2}}\{\sqrt{2}\,\mathbf{x} + \tfrac{1}{2}\mathbf{1}\}\rceil = \mathbf{x}.$$

For $\hat{\mathbf{y}} := \sqrt{2}\,\mathbf{x}$*, we obtain the error estimates*

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 \le \tfrac{\sqrt{2}}{2}, \qquad \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty \le \tfrac{1}{2}.$$

3. Note that $2\,R_2(\frac{\pi}{4})$ *is a scaled butterfly matrix. Then for* $\hat{\mathbf{y}} := 2\,R_2(\frac{\pi}{4})\mathbf{x}$ *and* $\mathbf{y} := \mathrm{rd}\,\hat{\mathbf{y}}$ *we obtain the same error estimates as above. This procedure is left–invertible in* $\mathbb{Z}^2$*, since for arbitrary* $\mathbf{x} \in \mathbb{Z}^2$ *from*

$$\mathbf{z} = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}\mathbf{x}, \qquad \mathbf{y} = \mathrm{rd}\,(\sqrt{2}\,\mathbf{z})$$

it follows that $\mathbf{z} = \lceil \frac{1}{\sqrt{2}}(\mathbf{y} - \frac{1}{2}\mathbf{1})\rceil$ *and hence*

$$\mathbf{x} = \tfrac{1}{2}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}\mathbf{z} = \tfrac{1}{2}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}\lceil \tfrac{1}{\sqrt{2}}(\mathbf{y} - \tfrac{1}{2}\mathbf{1})\rceil.$$

3.2 Integer transforms via lifting

Let $s \in \mathbb{R}$ with $s \ne 0$ be given. Then matrices of the form

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

are called *lifting matrices* of order 2 (see [4, 6]). Note that the inverse of a lifting matrix is again a lifting matrix:

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

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

$$R_2(\omega) = \begin{pmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{pmatrix} = \begin{pmatrix} 1 & \tan\frac{\omega}{2} \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ -\sin\omega & 1 \end{pmatrix}\begin{pmatrix} 1 & \tan\frac{\omega}{2} \\ 0 & 1 \end{pmatrix}. \tag{3.7}$$

Note that the above factorization of $R_2(\omega)$ consists of *nonorthogonal* matrix factors. This factorization (see [4]) can be used for construction of integer DCT as follows.

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

$$y_0 = x_0 + \lfloor s\,x_1 + \tfrac{1}{2}\rfloor = x_0 + \mathrm{rd}\,(s x_1), \qquad y_1 = x_1.$$

This transform is invertible and its inverse reads as follows

$$x_0 = y_0 - \lfloor s\, y_1 + \tfrac{1}{2} \rfloor = y_0 - \mathrm{rd}\,(s y_1), \qquad x_1 = y_1.$$

Indeed, we have that

$$y_0 - \lfloor s\, y_1 + \tfrac{1}{2} \rfloor = x_0 + \lfloor s\, x_1 + \tfrac{1}{2} \rfloor - \lfloor s\, x_1 + \tfrac{1}{2} \rfloor = x_0.$$

We obtain

Theorem 3.8 *Let $H_2 := R_2(\omega)$ with $\omega \in (0, \frac{\pi}{4}]$ be a rotation matrix. Then for arbitrary $\mathbf{x} = (x_0, x_1)^T \in \mathbb{Z}^2$, 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 by $y_0 = z_2,\ y_1 = z_1$, where*

$$\begin{aligned} z_0 &:= x_0 + \mathrm{rd}\,(x_1 \tan \tfrac{\omega}{2}), \\ z_1 &:= x_1 + \mathrm{rd}\,(-z_0 \sin \omega), \\ z_2 &:= z_0 + \mathrm{rd}\,(z_1 \tan \tfrac{\omega}{2}). \end{aligned}$$

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

$$\begin{aligned} w_0 &:= y_0 - \mathrm{rd}\,(y_1 \tan \tfrac{\omega}{2}), \\ w_1 &:= y_1 - \mathrm{rd}\,(-w_0 \sin \omega), \\ w_2 &:= w_0 - \mathrm{rd}\,(w_1 \tan \tfrac{\omega}{2}). \end{aligned}$$

Further, the error estimates

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &\le \tfrac{1}{2}(3 + 4\sin\omega + 2\cos\omega + (\tan\tfrac{\omega}{2})^2)^{1/2}, \\ \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty &\le \tfrac{1}{2}(1 + \tan\tfrac{\omega}{2} + \cos\omega) \end{aligned} \tag{3.8}$$

hold.

Proof. The formulas for y_0, y_1 and x_0, x_1 (after left-inverse transform) directly follow by applying the lifting steps to the three matrices in (3.7).
We prove the error estimates (3.8).
1. First we represent the components $\hat{y}_0 - y_0$ and $\hat{y}_1 - y_1$ of $\hat{\mathbf{y}} - \mathbf{y}$ in a convenient way. Let

$$\epsilon_0 := \{x_0 \sin\omega\}$$

denote the noninteger part of $x_0 \sin\omega$, and similarly

$$\epsilon_1 := \{x_1 \tan\tfrac{\omega}{2} + \tfrac{1}{2}\}, \quad \delta_0 := \{\hat{y}_0\} = \{x_0 \cos\omega + x_1 \sin\omega\}, \quad \delta_1 := \{x_1 \cos\omega\}.$$

Hence

$$\{\hat{y}_1\} = \{x_1 \cos\omega - x_0 \sin\omega\} = \{\delta_1 - \epsilon_0\}.$$

Using $(\tan\frac{\omega}{2})\sin\omega = 1 - \cos\omega$, it follows that

$$x_1(1 - \cos\omega) + \tfrac{1}{2}\sin\omega = (x_1 \tan\tfrac{\omega}{2} + \tfrac{1}{2})\sin\omega = (\lfloor x_1 \tan\tfrac{\omega}{2} + \tfrac{1}{2}\rfloor + \epsilon_1)\sin\omega$$

such that

$$\begin{aligned} y_1 &= z_1 = \lfloor (-\sin\omega)\,(\lfloor x_1 \tan\tfrac{\omega}{2} + \tfrac12 \rfloor + x_0) + \tfrac12 \rfloor + x_1 \\ &= \lfloor -x_1\,(1-\cos\omega) + (\epsilon_1 - \tfrac12)\sin\omega - x_0 \sin\omega + \tfrac12 \rfloor + x_1 \\ &= \lfloor x_1 \cos\omega - x_0 \sin\omega + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor = \lfloor \hat{y}_1 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor. \end{aligned}$$

Thus we obtain

$$\hat{y}_1 - y_1 = \hat{y}_1 - \lfloor \hat{y}_1 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor = (\delta_1 - \epsilon_0) - \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor. \quad (3.9)$$

Observing that

$$\begin{aligned} y_1 &= \lfloor \lfloor x_1 \cos\omega \rfloor + \delta_1 - \lfloor x_0 \sin\omega \rfloor - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor \\ &= \lfloor x_1 \cos\omega \rfloor - \lfloor x_0 \sin\omega \rfloor + \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor \end{aligned}$$

and that by $(\cos\omega)\,\tan\frac{\omega}{2} = \sin\omega - \tan\frac{\omega}{2}$,

$$\begin{aligned} &(\lfloor x_1 \cos\omega \rfloor - \lfloor x_0 \sin\omega \rfloor)\,\tan\tfrac{\omega}{2} = (x_1 \cos\omega - x_0 \sin\omega - \delta_1 + \epsilon_0)\,\tan\tfrac{\omega}{2} \\ &= x_1\,(\sin\omega - \tan\tfrac{\omega}{2}) - x_0\,(1-\cos\omega) + (\epsilon_0 - \delta_1)\,\tan\tfrac{\omega}{2}, \end{aligned}$$

we find

$$\begin{aligned} y_0 &= z_2 = \lfloor y_1 \tan\tfrac{\omega}{2} + \tfrac12 \rfloor + \lfloor x_1 \tan\tfrac{\omega}{2} + \tfrac12 \rfloor + x_0 \\ &= \lfloor x_1\,(\sin\omega - \tan\tfrac{\omega}{2}) - x_0\,(1-\cos\omega) \\ &\quad +(\epsilon_0 - \delta_1 + \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor)\,\tan\tfrac{\omega}{2} + \tfrac12 \rfloor + \lfloor x_1 \tan\tfrac{\omega}{2} + \tfrac12 \rfloor + x_0 \\ &= \lfloor x_1 \sin\omega + x_0 \cos\omega + 1 - \epsilon_1 + (\epsilon_0 - \delta_1 + \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor)\,\tan\tfrac{\omega}{2} \rfloor \\ &= \lfloor \hat{y}_0 + 1 - \epsilon_1 + (\epsilon_0 - \delta_1 + \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor)\,\tan\tfrac{\omega}{2} \rfloor. \end{aligned}$$

Hence we get

$$\begin{aligned} \hat{y}_0 - y_0 &= \hat{y}_0 - \lfloor \hat{y}_0 + 1 - \epsilon_1 + (\epsilon_0 - \delta_1 + \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor)\,\tan\tfrac{\omega}{2} \rfloor \\ &= \delta_0 - \lfloor \delta_0 + 1 - \epsilon_1 + (\epsilon_0 - \delta_1 + \lfloor \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega \rfloor)\,\tan\tfrac{\omega}{2} \rfloor. \quad (3.10) \end{aligned}$$

2. Now we can estimate the truncation error in the following way. Putting

$$\mu_0 := \delta_1 - \epsilon_0 + \tfrac12 + (\epsilon_1 - \tfrac12)\sin\omega, \quad \mu_1 := \delta_0 + 1 - \epsilon_1 + (\epsilon_0 - \delta_1 + \lfloor \mu_0 \rfloor)\,\tan\tfrac{\omega}{2},$$

the formulas (3.9) and (3.10) imply

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 = (\delta_1 - \epsilon_0 - \lfloor \mu_0 \rfloor)^2 + (\delta_0 - \lfloor \mu_1 \rfloor)^2 \quad (3.11)$$

and

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty = \max\{|\delta_1 - \epsilon_0 - \lfloor \mu_0 \rfloor|,\, |\delta_0 - \lfloor \mu_1 \rfloor|\}. \quad (3.12)$$

Since ϵ_0, ϵ_1, and δ_1 are contained in $[0,1)$, it follows that $\mu_0 \in (-1,2)$, i.e. $\lfloor \mu_0 \rfloor \in \{-1,\,0,1\}$. Then from

$$\max\{\lfloor \mu_0 \rfloor - \tfrac12 - (\epsilon_1 - \tfrac12)\,\sin\omega,\, -1\} \le \delta_1 - \epsilon_0 < (\lfloor \mu_0 \rfloor + 1) - \tfrac12 - (\epsilon_1 - \tfrac12)\,\sin\omega$$

it follows that

$$-\tfrac{1}{2} - (\epsilon_1 - \tfrac{1}{2})\sin\omega \le \delta_1 - \epsilon_0 - \lfloor\mu_0\rfloor < \tfrac{1}{2} - (\epsilon_1 - \tfrac{1}{2})\sin\omega \tag{3.13}$$

and especially for $\lfloor\mu_0\rfloor = -1$ even

$$0 \le \delta_1 - \epsilon_0 - \lfloor\mu_0\rfloor < \tfrac{1}{2} - (\epsilon_1 - \tfrac{1}{2})\sin\omega. \tag{3.14}$$

Using (3.13), we obtain the estimate for μ_1,

$$\delta_0 + 1 - \epsilon_1 + (-\tfrac{1}{2} + (\epsilon_1 - \tfrac{1}{2})\sin\omega)\,\tan\tfrac{\omega}{2} < \mu_1 \le \delta_0 + 1 - \epsilon_1 + (\tfrac{1}{2} + (\epsilon_1 - \tfrac{1}{2})\,\sin\omega)\,\tan\tfrac{\omega}{2}$$

and equivalently

$$\delta_0 + \tfrac{1}{2} - \tfrac{1}{2}\tan\tfrac{\omega}{2} + (\tfrac{1}{2} - \epsilon_1)\cos\omega < \mu_1 \le \delta_0 + \tfrac{1}{2} + \tfrac{1}{2}\tan\tfrac{\omega}{2} + (\tfrac{1}{2} - \epsilon_1)\cos\omega. \tag{3.15}$$

Since δ_0, $\epsilon_1 \in [0,1)$, we only need to consider the cases $\lfloor\mu_1\rfloor \in \{-1,0,1,2\}$.
3. We estimate the truncation errors (3.11) and (3.12). For $\lfloor\mu_1\rfloor = 0$ we find with (3.11) and (3.13) the error estimates

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 \le \max\{(-\tfrac{1}{2} - (\epsilon_1 - \tfrac{1}{2})\sin\omega)^2, (\tfrac{1}{2} - (\epsilon_1 - \tfrac{1}{2})\sin\omega)^2\} + \delta_0^2 < \tfrac{1}{4}(1+\sin\omega)^2 + 1$$

and

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty < 1.$$

For $\lfloor\mu_1\rfloor = 1$ we find similarly

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 < \tfrac{1}{4}(1+\sin\omega)^2 + (\delta_0 - 1)^2 \le \tfrac{1}{4}(1+\sin\omega)^2 + 1$$

and

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty < 1.$$

For $\lfloor\mu_1\rfloor = 2$ it follows by (3.15) that $\delta_0 > 2 - \frac{1}{2} - \frac{1}{2}\tan\frac{\omega}{2} - (\frac{1}{2} - \epsilon_1)\cos\omega$ and we find

$$\begin{aligned}
\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 &< \tfrac{1}{4}(1+\sin\omega)^2 + (\delta_0 - 2)^2 \\
&< \tfrac{1}{4}(1+\sin\omega)^2 + (\tfrac{1}{2} + \tfrac{1}{2}\tan\tfrac{\omega}{2} + (\tfrac{1}{2} - \epsilon_1)\cos\omega)^2 \\
&\le \tfrac{1}{4}((1+\sin\omega)^2 + (1 + \tan\tfrac{\omega}{2} + \cos\omega)^2) \\
&= \tfrac{3}{4} + \sin\omega + \tfrac{1}{2}\cos\omega + \tfrac{1}{4}(\tan\tfrac{\omega}{2})^2
\end{aligned}$$

and

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty < \tfrac{1}{2}(1 + \tan\tfrac{\omega}{2} + \cos\omega)$$

Finally, for $\lfloor\mu_1\rfloor = -1$ it follows by (3.15) that $\delta_0 < -\frac{1}{2} + \frac{1}{2}\tan\frac{\omega}{2} - (\frac{1}{2} - \epsilon_1)\cos\omega$ and hence

$$\begin{aligned}
\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 &< \tfrac{1}{4}(1+\sin\omega)^2 + (\delta_0 + 1)^2 \\
&< \tfrac{1}{4}(1+\sin\omega)^2 + (\tfrac{1}{2} + \tfrac{1}{2}\tan\tfrac{\omega}{2} - (\tfrac{1}{2} - \epsilon_1)\cos\omega)^2 \\
&\le \tfrac{3}{4} + \sin\omega + \tfrac{1}{2}\cos\omega + \tfrac{1}{4}(\tan\tfrac{\omega}{2})^2
\end{aligned}$$

as before and again

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty < \tfrac{1}{2}(1 + \tan\tfrac{\omega}{2} + \cos\omega).$$

For $\omega \in (0, \frac{\pi}{4}]$ we have

$$1 + \tfrac{1}{4}(1+\sin\omega)^2 < \tfrac{3}{4} + \sin\omega + \tfrac{1}{2}\cos\omega + \tfrac{1}{4}(\tan\tfrac{\omega}{2})^2$$

such that the assertions (3.8) are proved. *q.e.d.*

Remark 3.9 *Let $\hat{\mathbf{y}} := R_2(\omega)\mathbf{x}$ with arbitrary $\mathbf{x} \in \mathbb{Z}^2$ and $\mathbf{y}$ its integer approximation. The special values for the errors $\|\hat{\mathbf{y}} - \mathbf{y}\|_2$ and $\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty$ via the lifting procedure for $\omega \in \{\frac{\pi}{4}, \frac{\pi}{8}, \frac{\pi}{16}, \frac{3\pi}{16}\}$ follow by inserting into formulas* (3.8). *In particular, we obtain*

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 \le \begin{cases} 1.361453 & \textit{for } \omega = \frac{\pi}{4}, \\ 1.266694 & \textit{for } \omega = \frac{\pi}{8}, \\ 1.199128 & \textit{for } \omega = \frac{\pi}{16}, \\ 1.320723 & \textit{for } \omega = \frac{3\pi}{16}, \end{cases} \qquad \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty \le \begin{cases} 1.060660 & \textit{for } \omega = \frac{\pi}{4}, \\ 1.061396 & \textit{for } \omega = \frac{\pi}{8}, \\ 1.039638 & \textit{for } \omega = \frac{\pi}{16}, \\ 1.067408 & \textit{for } \omega = \frac{3\pi}{16}. \end{cases}$$

Comparing with the expansion factor method, the lifting method produces truncation errors which are slightly greater than those found in Theorem 3.4.

The above procedure can easily be applied to the orthogonal "submatrices" of $T_n(1)$ (and of course also to that of $T_n(0)$ and $A_n(1)$, see (2.5) – (2.7)) such that we obtain an integer approximation of $H_2\mathbf{x}$ for $\mathbf{x} \in \mathbb{Z}^2$. Moreover, in this procedure, *no expansion factor* for the matrices H_2 are needed.

4 Integer DCT–II of length 8

Using the methods of Section 2 and Section 3, we want to derive different algorithms for the integer DCT–II of length 8.
An orthogonal factorization of the cosine matrix C_8^{II} looks as follows (see [10]):

$$C_8^{II} = B_8 \begin{pmatrix} I_4 & \\ & A_4(1) \end{pmatrix} \begin{pmatrix} C_2^{II} & & & \\ & C_2^{IV} & & \\ & & C_2^{II} & \\ & & & C_2^{II} \end{pmatrix} \begin{pmatrix} T_4(0) & \\ & T_4(1) \end{pmatrix} T_8(0) \qquad (4.1)$$

with the bit reversal matrix $B_8 := P_8^T(P_4 \oplus P_4)$,

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

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

and with

$$C_2^{II} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad C_2^{IV} = \begin{pmatrix} \cos\frac{\pi}{8} & \sin\frac{\pi}{8} \\ \sin\frac{\pi}{8} & -\cos\frac{\pi}{8} \end{pmatrix} = \Sigma_2 \begin{pmatrix} \cos\frac{\pi}{8} & \sin\frac{\pi}{8} \\ -\sin\frac{\pi}{8} & \cos\frac{\pi}{8} \end{pmatrix}.$$

Let us denote the five *orthogonal* matrix factors of C_8^{II} in (4.1) in this order by

$$C_8^{II} = B_8\, A_8(0,1)\, T_8(0,1,0,0)\, T_8(0,1)\, T_8(0). \qquad (4.2)$$

We want to apply the results of Section 3. Note that the factorization (4.2) implies a fast algorithm for computing the DCT–II of length 8 with 11 multiplications and 29 additions. This algorithm is very similar to that of C. Loeffler et al. [7].

4.1 Algorithms based on the expansion factor method

Let us start with the application of Theorem 3.4. For that we consider instead of (4.2) the factorization

$$4\,C_8^{II} = B_8\,(\sqrt{2}\,A_8(0,1))\,(\sqrt{2}\,T_8(0,1,0,0))\,(\sqrt{2}\,T_8(0,1))\,(\sqrt{2}\,T_8(0)).$$

Using the subsets $S_1(\omega)$, $S_2(\omega)$, $S_3(\omega)$ defined in Theorem 3.3, and $\mathbf{k}_1 = (0,0)^T$, $\mathbf{k}_2 = (-1,0)^T$, $\mathbf{k}_3 = (0,1)^T$, we obtain the following algorithm for integer DCT–II of length 8, namely an integer approximation for $\hat{\mathbf{y}} := 4\,C_8^{II}\mathbf{x}$ with given $\mathbf{x} \in \mathbb{Z}^8$:

Algorithm 4.1 [Integer DCT–II algorithm A]
Input: $\mathbf{x} \in \mathbb{Z}^8$.

1. Compute $\mathbf{x}^{(1)} := \sqrt{2}\,T_8(0)\,\mathbf{x}$.
2. Compute $\tilde{\mathbf{x}}^{(2)} := \sqrt{2}\,T_8(0,1)\,\mathbf{x}^{(1)}$.
   For $j = 0,1,2,3$ put $x_j^{(2)} := \tilde{x}_j^{(2)}$.
   If $(\{\tilde{x}_4^{(2)} + \frac{1}{2}\}, \{-\tilde{x}_7^{(2)} + \frac{1}{2}\})^T \in S_l(\frac{\pi}{16})$, $(l = 1,2,3)$, then put
   $$(x_4^{(2)}, -x_7^{(2)})^T := (\mathrm{rd}\,(\tilde{x}_4^{(2)}), \mathrm{rd}\,(-\tilde{x}_7^{(2)}))^T + \mathbf{k}_l.$$
   If $(\{\tilde{x}_5^{(2)} + \frac{1}{2}\}, \{\tilde{x}_6^{(2)} + \frac{1}{2}\})^T \in S_l(\frac{3\pi}{16})$, $(l = 1,2,3)$, then put
   $$(x_5^{(2)}, x_6^{(2)})^T := (\mathrm{rd}\,(\tilde{x}_5^{(2)}), \mathrm{rd}\,(\tilde{x}_6^{(2)}))^T + \mathbf{k}_l.$$
3. Compute $\tilde{\mathbf{x}}^{(3)} := \sqrt{2}\,T_8(0,1,0,0)\,\mathbf{x}^{(2)}$.
   For $j = 0,1,4,5,6,7$ put $x_j^{(3)} := \tilde{x}_j^{(3)}$.
   If $(\{\tilde{x}_2^{(3)} + \frac{1}{2}\}, \{-\tilde{x}_3^{(3)} + \frac{1}{2}\})^T \in S_l(\frac{\pi}{8})$, $(l = 1,2,3)$, then put
   $$(x_2^{(3)}, -x_3^{(3)})^T := (\mathrm{rd}\,(\tilde{x}_2^{(3)}), \mathrm{rd}\,(-\tilde{x}_3^{(3)}))^T + \mathbf{k}_l.$$
4. Compute $\tilde{\mathbf{x}}^{(4)} := \sqrt{2}\,A_8(0,1)\,\mathbf{x}^{(3)}$.
   Put $\mathbf{x}^{(4)} := \mathrm{rd}\,\tilde{\mathbf{x}}^{(4)}$.
5. Compute $\mathbf{y} := B_8\,\mathbf{x}^{(4)}$.

Output: $\mathbf{y} \in \mathbb{Z}^8$ integer approximation of $\hat{\mathbf{y}} = 4\,C_8^{II}\mathbf{x}$.

Algorithm 4.1 requires 18 multiplications, 32 additions, 12 rounding operations (rounding to the next integer), 6 comparisons and (possibly) 3 additions of 1. Expressions of the form $\{x + \frac{1}{2}\} = |x - \mathrm{rd}\,x|$ can be obtained by one substraction. Observe further, that 6 multiplications (with $\sqrt{2}$) are due to the multiplication with $\sqrt{2}\,A_8(0,1)$ in step 4. Hence, the arithmetical complexity can be improved by taking different expansion factors for the matrices. The factorization

$$4\sqrt{2}\,C_8^{II} = B_8\,(I_4 \oplus \sqrt{2}I_4)\,A_8(0,1))\,(\sqrt{2}\,T_8(0,1,0,0))\,(2\sqrt{2}I_4 \oplus \sqrt{2}I_4)\,T_8(0,1))\,(\sqrt{2}\,T_8(0)).$$

yields an algorithm with only 14 multiplications and 4 shifts. Unfortunately, the overall expansion factor 4 is enlarged to $4\sqrt{2}$.
The inverse integer DCT–II of length 8 is very simple. Let here $\mathbf{1} := (1,1,1,1,1,1,1,1)^T$.

Algorithm 4.2 [Inverse integer DCT–II algorithm A]
Input: $\mathbf{y} \in \mathbb{Z}^8$ (integer approximation of $4\,C_8^{II}\mathbf{x}$ computed by Algorithm 4.1).

1. Compute $\mathbf{y}^{(1)} := B_8 \mathbf{y}$.
2. Compute $\mathbf{y}^{(2)} := \lceil \frac{1}{\sqrt{2}} A_8(0,1)^T (\mathbf{y}^{(1)} - \frac{1}{2}\mathbf{1}) \rceil$.
3. Compute $\mathbf{y}^{(3)} := \lceil \frac{1}{\sqrt{2}} T_8(0,1,0,0)^T (\mathbf{y}^{(2)} - \frac{1}{2}\mathbf{1}) \rceil$.
4. Compute $\mathbf{y}^{(4)} := \lceil \frac{1}{\sqrt{2}} T_8(0,1)^T (\mathbf{y}^{(3)} - \frac{1}{2}\mathbf{1}) \rceil$.
5. Compute $\mathbf{x} := \lceil \frac{1}{\sqrt{2}} T_8(0)^T (\mathbf{y}^{(4)} - \frac{1}{2}\mathbf{1}) \rceil$.

Output: $\mathbf{x} \in \mathbb{Z}^8$ original vector of Algorithm 4.1.

Enlarging the expansion factors due to Theorem 3.6, we can use the following factorization

$$4\sqrt{2}\,C_8^{II} = B_8\,(\sqrt{2}A_8(0,1))\,(2I_4 \oplus \sqrt{2}I_4)T_8(0,1,0,0)\,(\sqrt{2}I_4 \oplus 2I_4)T_8(0,1)\,(\sqrt{2}T_8(0)).$$

In this case we need to replace the steps 2 and 3 of Algorithm 4.1 suitably and obtain:

Algorithm 4.3 [Integer DCT–II algorithm B]
Input: $\mathbf{x} \in \mathbb{Z}^8$.

1. Compute $\mathbf{x}^{(1)} := \sqrt{2}\,T_8(0)\,\mathbf{x}$.
2. Compute $\tilde{\mathbf{x}}^{(2)} := (\sqrt{2}I_4 \oplus 2I_4)\,T_8(0,1)\,\mathbf{x}^{(1)}$.
   For $j = 0,1,2,3$ put $x_j^{(2)} := \tilde{x}_j^{(2)}$. Further, put
   $$(x_4^{(2)}, x_7^{(2)})^T := (\mathrm{rd}\,(\tilde{x}_4^{(2)}) - 1, \mathrm{rd}\,(\tilde{x}_7^{(2)}))^T,\ (x_5^{(2)}, x_6^{(2)})^T := (\mathrm{rd}\,(\tilde{x}_5^{(2)}) - 1, \mathrm{rd}\,(\tilde{x}_6^{(2)}))^T.$$
3. Compute $\tilde{\mathbf{x}}^{(3)} := (2I_4 \oplus \sqrt{2}I_4)\,T_8(0,1,0,0)\,\mathbf{x}^{(2)}$.
   For $j = 4,5,6,7$ put $x_j^{(3)} := \tilde{x}_j^{(3)}$.
   Put
   $$(x_0^{(3)}, x_1^{(3)})^T := (\mathrm{rd}\,(\tilde{x}_0^{(3)}), \mathrm{rd}\,(\tilde{x}_1^{(3)}))^T, \quad (x_2^{(3)}, x_3^{(3)})^T := (\mathrm{rd}\,(\tilde{x}_2^{(3)}) - 1, \mathrm{rd}\,(\tilde{x}_3^{(3)}))^T$$
4. Compute $\tilde{\mathbf{x}}^{(4)} := \sqrt{2}\,A_8(0,1)\,\mathbf{x}^{(3)}$ and put $\mathbf{x}^{(4)} := \mathrm{rd}\,\tilde{\mathbf{x}}^{(4)}$.
5. Compute $\mathbf{y} := B_8\,\mathbf{x}^{(4)}$.

Output: $\mathbf{y} \in \mathbb{Z}^8$ integer approximation of $\hat{\mathbf{y}} = 4\sqrt{2}\,C_8^{II}\mathbf{x}$.

This algorithm needs 20 multiplications, 26 additions, 14 rounding operations and three subtractions of 1. Again the 6 multiplications in step 4 can be partially avoided by taking a slightly changed factorization (and enlarging the overall expansion factor to 8). The inverse integer DCT–II algorithm for Algorithm 4.3 is very similar to Algorithm 4.2, one only needs to replace the inverse factorization matrices suitably. Let us consider the truncation errors of these algorithms in the worst case. We obtain

Theorem 4.4 *Let* $\mathbf{x} \in \mathbb{Z}^8$ *be an arbitrary vector. Using the integer* DCT–II *Algorithm* 4.1, *we obtain an integer approximation* $\mathbf{y} \in \mathbb{Z}^8$ *of* $\hat{\mathbf{y}} = 4\, C_8^{II}\mathbf{x}$ *satisfying*

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &\le 3\sqrt{1 + \tfrac{\sqrt{2}}{2}} + \sqrt{\tfrac{3}{2}} \approx 5.144434, \\ \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty &\le 3.306564. \end{aligned}$$

Using the integer DCT–II *Algorithm* 4.3, *we obtain an integer approximation* $\mathbf{y} \in \mathbb{Z}^8$ *of* $\hat{\mathbf{y}} = 4\sqrt{2}\, C_8^{II}\mathbf{x}$ *satisfying*

$$\begin{aligned} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 &\le 2\sqrt{10} + \sqrt{7} + \sqrt{\tfrac{3}{2}} \approx 10.195052 \\ \|\hat{\mathbf{y}} - \mathbf{y}\|_\infty &\le 6. \end{aligned}$$

Proof. 1. We consider Algorithm 4.1. Let us denote the preliminary results of the exact DCT–II of length 8 by

$$\begin{aligned} \hat{\mathbf{x}}^{(1)} &:= \sqrt{2}\, T_8(0)\, \mathbf{x}, \\ \hat{\mathbf{x}}^{(2)} &:= \sqrt{2}\, T_8(0,1)\, \hat{\mathbf{x}}^{(1)}, \\ \hat{\mathbf{x}}^{(3)} &:= \sqrt{2}\, T_8(0,1,0,0)\, \hat{\mathbf{x}}^{(2)}, \\ \hat{\mathbf{x}}^{(4)} &:= \sqrt{2}\, A_8(0,1)\, \hat{\mathbf{x}}^{(3)}. \end{aligned}$$

We estimate the *truncation error*

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 = \|B_8(\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)})\|_2 = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_2.$$

Let $\mathbf{e}^{(s)} := \mathbf{x}^{(s)} - \tilde{\mathbf{x}}^{(s)}$ denote the truncation error of a single step $s \in \{1,\ 2,\ 3,\ 4\}$, where the preliminary results $\mathbf{x}^{(s)}$ and $\tilde{\mathbf{x}}^{(s)}$ are defined in Algorithm 4.1, and in particular $\tilde{\mathbf{x}}^{(1)} = \mathbf{x}^{(1)}$. Then we see that $\mathbf{e}^{(1)} = \mathbf{0}$. By Theorem 3.4 we obtain

$$\|\mathbf{e}^{(2)}\|_2 \le \sqrt{\tfrac{1}{2}(2 + \sin\tfrac{\pi}{8} + \sin\tfrac{3\pi}{8})} = \sqrt{1 + \sin\tfrac{\pi}{4}}, \qquad \|\mathbf{e}^{(3)}\|_2 \le \tfrac{1}{\sqrt{2}}\sqrt{1 + \sin\tfrac{\pi}{4}}.$$

By Remark 3.7 we can estimate $\|\mathbf{e}^{(4)}\|_2 \le \sqrt{\frac{3}{2}}$. Note that $\hat{\mathbf{x}}^{(1)} = \mathbf{x}^{(1)}$. Then it follows that

$$\begin{aligned} \|\hat{\mathbf{x}}^{(2)} - \mathbf{x}^{(2)}\|_2 &\le \|\hat{\mathbf{x}}^{(2)} - \tilde{\mathbf{x}}^{(2)}\|_2 + \|\tilde{\mathbf{x}}^{(2)} - \mathbf{x}^{(2)}\|_2 \\ &\le \sqrt{2}\, \|T_8(0,1)\|_2\, \|\hat{\mathbf{x}}^{(1)} - \mathbf{x}^{(1)}\|_2 + \|\mathbf{e}^{(2)}\|_2 = \|\mathbf{e}^{(2)}\|_2, \\ \|\hat{\mathbf{x}}^{(3)} - \mathbf{x}^{(3)}\|_2 &\le \|\hat{\mathbf{x}}^{(3)} - \tilde{\mathbf{x}}^{(3)}\|_2 + \|\tilde{\mathbf{x}}^{(3)} - \mathbf{x}^{(3)}\|_2 \\ &\le \sqrt{2}\, \|T_8(0,1,0,0)\|_2\, \|\hat{\mathbf{x}}^{(2)} - \mathbf{x}^{(2)}\|_2 + \|\mathbf{e}^{(3)}\|_2 \le \sqrt{2}\, \|\mathbf{e}^{(2)}\|_2 + \|\mathbf{e}^{(3)}\|_2 \end{aligned}$$

and finally

$$\begin{aligned} \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_2 &\le \|\hat{\mathbf{x}}^{(4)} - \tilde{\mathbf{x}}^{(4)}\|_2 + \|\tilde{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_2 \\ &\le \sqrt{2}\, \|A_8(0,1)\|_2\, \|\hat{\mathbf{x}}^{(3)} - \mathbf{x}^{(3)}\|_2 + \|\mathbf{e}^{(4)}\|_2 \\ &\le 2\, \|\mathbf{e}^{(2)}\|_2 + \sqrt{2}\, \|\mathbf{e}^{(3)}\|_2 + \|\mathbf{e}^{(4)}\|_2 < 3\sqrt{1 + \tfrac{\sqrt{2}}{2}} + \sqrt{\tfrac{3}{2}}. \end{aligned}$$

2. Now we estimate the truncation error of Algorithm 4.1 componentwisely. By $x_j^{(s)}$ and $\hat{x}_j^{(s)}$ we denote the components of $\mathbf{x}^{(s)}$ and $\hat{\mathbf{x}}^{(s)}$, respectively. Step 1 of Algorithm 4.1

yields $x_j^{(1)} = \hat{x}_j^{(1)}$ $(j = 0, \ldots, 7)$. After step 2 we obtain $x_j^{(2)} = \hat{x}_j^{(2)}$ $(j = 0, \ldots, 3)$ and by Remark 3.5

$$\begin{aligned} \max\{|x_4^{(2)} - \hat{x}_4^{(2)}|,\, |x_7^{(2)} - \hat{x}_7^{(2)}|\} &\le 0.691342, \\ \max\{|x_5^{(2)} - \hat{x}_5^{(2)}|,\, |x_6^{(2)} - \hat{x}_6^{(2)}|\} &\le 0.961940. \end{aligned}$$

After step 3 we get $x_j^{(3)} = \hat{x}_j^{(3)}$ $(j = 0,\, 1)$ and by Remark 3.5 it follows that

$$\max\{|x_2^{(3)} - \hat{x}_2^{(3)}|,\, |x_3^{(3)} - \hat{x}_3^{(3)}|\} \le 0.853553.$$

Further we estimate

$$\begin{aligned} |x_4^{(3)} - \hat{x}_4^{(3)}| &\le |x_4^{(2)} - \hat{x}_4^{(2)}| + |x_5^{(2)} - \hat{x}_5^{(2)}| \\ &\le 0.691342 + 0.961940 = 1.653282 \end{aligned}$$

and analogously $|x_j^{(3)} - \hat{x}_j^{(3)}| \le 1.653282$ $(j = 5,\, 6,\, 7)$. Finally, step 4 yields by Remark 3.7 that $|x_j^{(4)} - \hat{x}_j^{(4)}| \le 0.5$ $(j = 0,\, 1)$. For $j = 2,\, 3$ we get that

$$\begin{aligned} |x_j^{(4)} - \hat{x}_j^{(4)}| &\le |x_j^{(4)} - \sqrt{2}\, x_j^{(3)}| + \sqrt{2}\, |x_j^{(3)} - \hat{x}_j^{(3)}| \\ &\le 0.5 + \sqrt{2} \cdot 0.853553 \approx 1.707106. \end{aligned}$$

For $j = 4,\, 7$ we obtain analogously

$$|x_j^{(4)} - \hat{x}_j^{(4)}| \le 0.5 + \sqrt{2} \cdot 1.653282 \approx 2.838094.$$

For $j = 5,\, 6$ we estimate

$$|x_j^{(4)} - \hat{x}_j^{(4)}| \le |x_5^{(3)} - \hat{x}_5^{(3)}| + |x_7^{(3)} - \hat{x}_7^{(3)}| \le 3.306564.$$

Thus we obtain $\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_\infty \le 3.306564$.

3. For Algorithm 4.3, we estimate the truncation error in like manner. With

$$\begin{aligned} \hat{\mathbf{x}}^{(1)} &:= \sqrt{2}\, T_8(0)\, \mathbf{x}, \qquad \hat{\mathbf{x}}^{(2)} := (\sqrt{2} I_4 \oplus 2 I_4)\, T_8(0,1)\, \hat{\mathbf{x}}^{(1)}, \\ \hat{\mathbf{x}}^{(3)} &:= (2 I_4 \oplus \sqrt{2} I_4)\, T_8(0,1,0,0)\, \hat{\mathbf{x}}^{(2)}, \qquad \hat{\mathbf{x}}^{(4)} := \sqrt{2}\, A_8(0,1)\, \hat{\mathbf{x}}^{(3)} \end{aligned}$$

we find by Theorem 3.6 and Remark 3.7 that

$$\mathbf{e}^{(1)} = \mathbf{0}, \quad \|\mathbf{e}^{(2)}\|_2 \le \sqrt{5}, \quad \|\mathbf{e}^{(3)}\|_2 \le \sqrt{\tfrac{7}{2}}, \quad \|\mathbf{e}^{(4)}\|_2 \le \sqrt{\tfrac{3}{2}}$$

and hence

$$\|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_2 \le \sqrt{2}\, (2\|\mathbf{e}^{(2)}\|_2 + \|\mathbf{e}^{(3)}\|_2) + \|\mathbf{e}^{(4)}\|_2 < 2\sqrt{10} + \sqrt{7} + \sqrt{\tfrac{3}{2}}.$$

Using componentwise estimation of the truncation error, we obtain by Theorem 3.6 and Remark 3.7 that

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_\infty \le 6.$$

This completes the proof. *q.e.d.*

4.2 Algorithms based on the lifting method

We propose two further algorithms using integer transforms via lifting, and estimate the truncation errors in the worst case.
Using the factorization (4.2) of C_8^{II}, we give an algorithm, where lifting is applied to all orthogonal matrix factors of C_8^{II} in (4.2):

Algorithm 4.5 [Integer DCT–II algorithm C]
Input: $\mathbf{x} \in \mathbb{Z}^8$.

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

$$\begin{aligned}
\mathbf{z}^{(0)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{8})\mathbf{w}^{(1)}) + \mathbf{w}^{(0)},\\
\mathbf{z}^{(1)} &:= \mathrm{rd}\,(((-\sin\tfrac{\pi}{4})\mathbf{z}^{(0)}) + \mathbf{w}^{(1)},\\
\mathbf{z}^{(2)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{8})\mathbf{z}^{(1)}) + \mathbf{z}^{(0)}.
\end{aligned}$$

   Put $\mathbf{x}^{(1)} := ((\mathbf{z}^{(2)})^T, -(\mathbf{z}^{(1)})^T)^T$.

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

$$\begin{aligned}
\mathbf{z}^{(0)} &:= \mathrm{rd}\,(((\tan\tfrac{\pi}{8})\, I_2 \oplus \tan\tfrac{\pi}{32} \oplus \tan\tfrac{3\pi}{32})\mathbf{w}^{(1)}) + \mathbf{w}^{(0)},\\
\mathbf{z}^{(1)} &:= \mathrm{rd}\,(((-\sin\tfrac{\pi}{4})\, I_2 \oplus (-\sin\tfrac{\pi}{16}) \oplus (-\sin\tfrac{3\pi}{16}))\mathbf{z}^{(0)}) + \mathbf{w}^{(1)},\\
\mathbf{z}^{(2)} &:= \mathrm{rd}\,(((\tan\tfrac{\pi}{8})\, I_2 \oplus \tan\tfrac{\pi}{32} \oplus \tan\tfrac{3\pi}{32})\mathbf{z}^{(1)}) + \mathbf{z}^{(0)}.
\end{aligned}$$

   Put $\mathbf{x}^{(2)} := (z_0^{(2)}, z_1^{(2)}, -z_0^{(1)}, -z_1^{(1)}, z_2^{(2)}, z_3^{(2)}, z_3^{(1)}, -z_2^{(1)},))^T$.

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

$$\begin{aligned}
\mathbf{z}^{(0)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{8} \oplus \tan\tfrac{\pi}{16} \oplus (\tan\tfrac{\pi}{8})\, I_2)\mathbf{w}^{(1)}) + \mathbf{w}^{(0)},\\
\mathbf{z}^{(1)} &:= \mathrm{rd}\,(((-\sin\tfrac{\pi}{4}) \oplus (-\sin\tfrac{\pi}{8}) \oplus (-\sin\tfrac{\pi}{4})\, I_2)\, \mathbf{z}^{(0)}) + \mathbf{w}^{(1)},\\
\mathbf{z}^{(2)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{8} \oplus \tan\tfrac{\pi}{16} \oplus (\tan\tfrac{\pi}{8})\, I_2)\, \mathbf{z}^{(1)}) + \mathbf{z}^{(0)}.
\end{aligned}$$

   Put $\mathbf{x}^{(3)} := (z_0^{(2)}, -z_0^{(1)}, z_1^{(2)}, -z_1^{(1)}, z_2^{(2)}, -z_2^{(1)}, z_3^{(2)}, -z_3^{(1)})^T$.

4. For $j = 0, \ldots, 4$ put $x_j^{(4)} := x_j^{(3)}$ and $x_6^{(4)} := x_7^{(3)}$. Compute

$$\begin{aligned}
z_0 &:= \mathrm{rd}\,(x_7^{(3)}\, \tan\tfrac{\pi}{8}) + x_5^{(3)}, \quad x_7^{(4)} := -\mathrm{rd}\,(-z_0\, \sin\tfrac{\pi}{4}) - x_7^{(3)},\\
x_5^{(4)} &:= \mathrm{rd}\,(-x_7^{(4)}\, \tan\tfrac{\pi}{8}) + z_0.
\end{aligned}$$

5. Compute $\mathbf{y} := B_8\, \mathbf{x}^{(4)}$.

Output: $\mathbf{y} \in \mathbb{Z}^8$ integer approximation of $\hat{\mathbf{y}} = C_8^{II}\mathbf{x}$.

The arithmetical complexity of Algorithm 4.5 is very high, we need 39 multiplications, 39 additions and 39 rounding operations. This high arithmetical complexity is due to the fact that matrix factors containing the rotation matrices $R_2(\frac{\pi}{4})$ (as e.g. $T_8(0)$) are computed by more expensive lifting steps. The inverse integer DCT algorithm for Algorithm 4.5 simply follows by going backward and taking the inverse lifting procedure of Theorem

3.8. Since integer transform via lifting does not need expansion factors of matrices, we are able to give an integer approximation of $C_8^{II}\mathbf{x}$ (instead of $\alpha\, C_8^{II}\mathbf{x}$) in Algorithm 4.5. The truncation error of this algorithm can be estimated by

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_2 \le \|\mathbf{e}^{(1)}\|_2 + \|\mathbf{e}^{(2)}\|_2 + \|\mathbf{e}^{(3)}\|_2 + \|\mathbf{e}^{(4)}\|_2 \approx 9.385895,$$

where

$$\begin{aligned}
\|\mathbf{e}^{(1)}\|_2 &\le 2\sqrt{h(\tfrac{\pi}{4})} \approx 2.722905,\\
\|\mathbf{e}^{(2)}\|_2 &\le \sqrt{2\,h(\tfrac{\pi}{4}) + h(\tfrac{\pi}{16}) + h(\tfrac{3\pi}{16})} \approx 2.624752,\\
\|\mathbf{e}^{(3)}\|_2 &\le \sqrt{3\,h(\tfrac{\pi}{4}) + h(\tfrac{\pi}{8})} \approx 2.676784,\\
\|\mathbf{e}^{(4)}\|_2 &\le \sqrt{h(\tfrac{\pi}{4})} \approx 1.361453,
\end{aligned}$$

and $h(t) := \frac{3}{4} + \sin t + \frac{1}{2}\cos t + \frac{1}{4}(\tan\frac{t}{2})^2$. Here the notations are defined analogously as in Theorem 4.4. Using Remark 3.9, we obtain by componentwise error estimation that

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_\infty \le 7.386088.$$

This truncation error is comparable with that of the Algorithm 4.3. An alternative for integer DCT–II of length 8 with reasonably small expansion factor and small truncation error may be a mixed version of the two methods. Finally we propose an algorithm, where the expansion factor method as well as the lifting method are used. Based on the factorization

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

we apply lifting to the submatrix $T_4(1)$ of $T_8(0,1)$, to the submatrix $C_2^{II} \oplus C_2^{IV}$ of $T_8(0,1,0,0)$ and to $A_8(0,1)$. For the other matrices we use the expansion method.

Algorithm 4.6 [Integer DCT–II algorithm D]
Input: $\mathbf{x} \in \mathbb{Z}^8$.

1. Compute $\mathbf{x}^{(1)} := \sqrt{2}\,T_8(0)\,\mathbf{x}$.

2. Put $\mathbf{w}^{(0)} := (x_0^{(1)}, x_1^{(1)}, x_2^{(1)}, x_3^{(1)})^T$, $\mathbf{w}^{(1)} := (x_4^{(1)}, x_5^{(1)})^T$, $\mathbf{w}^{(2)} := (x_7^{(1)}, x_6^{(1)})^T$.
   Compute $\mathbf{z} := \sqrt{2}\,T_4(0)\mathbf{w}^{(0)}$ and

$$\begin{aligned}
\mathbf{z}^{(0)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{32} \oplus \tan\tfrac{3\pi}{32})\mathbf{w}^{(2)}) + \mathbf{w}^{(1)},\\
\mathbf{z}^{(1)} &:= \mathrm{rd}\,(((-\sin\tfrac{\pi}{16}) \oplus (-\sin\tfrac{3\pi}{16}))\mathbf{z}^{(0)}) + \mathbf{w}^{(2)},\\
\mathbf{z}^{(2)} &:= \mathrm{rd}((\tan\tfrac{\pi}{32} \oplus \tan\tfrac{3\pi}{32})\mathbf{z}^{(1)}) + \mathbf{z}^{(0)}.
\end{aligned}$$

   Put $\mathbf{x}^{(2)} := (\mathbf{z}^T,\, z_0^{(2)}, z_1^{(2)}, z_1^{(1)}, -z_0^{(1)})^T$.

3. Put $\mathbf{w}^{(0)} := (x_0^{(2)}, x_2^{(2)})^T$, $\mathbf{w}^{(1)} := (x_1^{(2)}, x_3^{(2)})^T$, $\mathbf{w}^{(2)} := (x_4^{(2)}, x_5^{(2)}, x_6^{(2)}, x_7^{(2)})^T$.
   Compute $\mathbf{z} := (\sqrt{2}\,C_2^{II} \oplus \sqrt{2}\,C_2^{II})\mathbf{w}$ and

$$\begin{aligned}
\mathbf{z}^{(0)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{8} \oplus \tan\tfrac{\pi}{16})\mathbf{w}^{(1)}) + \mathbf{w}^{(0)},\\
\mathbf{z}^{(1)} &:= \mathrm{rd}\,(((-\sin\tfrac{\pi}{4}) \oplus (-\sin\tfrac{\pi}{8}))\mathbf{z}^{(0)}) + \mathbf{w}^{(1)},\\
\mathbf{z}^{(2)} &:= \mathrm{rd}\,((\tan\tfrac{\pi}{8} \oplus \tan\tfrac{\pi}{16})\mathbf{z}^{(1)}) + \mathbf{z}^{(0)}.
\end{aligned}$$

   Put $\mathbf{x}^{(3)} := (z_0^{(2)}, -z_0^{(1)}, z_1^{(2)}, -z_1^{(1)},\, \mathbf{z}^T)^T$.

4. For $j = 0, \ldots, 4$ put $x_j^{(4)} := x_j^{(3)}$ and $x_6^{(4)} := x_7^{(3)}$. Compute

$$\begin{aligned} z_0 &:= \mathrm{rd}\,(x_7^{(3)} \tan \tfrac{\pi}{8}) + x_5^{(3)}, \quad x_7^{(4)} := -\mathrm{rd}\,(-z_0 \sin \tfrac{\pi}{4}) - x_7^{(3)}, \\ x_5^{(4)} &:= \mathrm{rd}\,(-x_7^{(4)} \tan \tfrac{\pi}{8}) + z_0. \end{aligned}$$

5. Compute $\mathbf{y} := B_8\,\mathbf{x}^{(4)}$.

Output: $\mathbf{y} \in \mathbb{Z}^8$ integer approximation of $\hat{\mathbf{y}} = 2\,C_8^{II}\mathbf{x}$.

Algorithm 4.6 needs only 15 multiplications, 31 additions and 15 rounding operations, where the overall expansion factor is 2. Hence, its arithmetical complexity is nearly optimal, keeping in mind that best DCT–II algorithms for $n = 8$ need 11 multiplications and 29 additions without counting the scaling by $2\sqrt{2}$ (see [7]). For the truncation error produced by Algorithm 4.6 we find in the same manner as before with $h(t) = \frac{3}{4} + \sin t + \frac{1}{2}\cos t + \frac{1}{4}(\tan \frac{t}{2})^2$

$$\begin{aligned} \mathbf{e}^{(1)} &= \mathbf{0}, \quad \|\mathbf{e}^{(2)}\|_2 \le (h(\tfrac{\pi}{16}) + h(\tfrac{3\pi}{16}))^{1/2} \approx 1.783877, \\ \|\mathbf{e}^{(3)}\|_2 &\le (h(\tfrac{\pi}{4}) + h(\tfrac{\pi}{8}))^{1/2} \approx 1.859588, \quad \|\mathbf{e}^{(4)}\|_2 \le h(\tfrac{\pi}{4})^{1/2} \approx 1.361453, \end{aligned}$$

and

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2 = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_2 \le \sqrt{2}\,\|\mathbf{e}^{(2)}\|_2 + \|\mathbf{e}^{(3)}\|_2 + \|\mathbf{e}^{(4)}\|_2 \approx 5.743824.$$

Using Remark 3.9, we obtain by componentwise error estimation that

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_\infty = \|\hat{\mathbf{x}}^{(4)} - \mathbf{x}^{(4)}\|_\infty \le 4.040473.$$

These worst case error estimates are comparable with that of Algorithm 4.1.

5 Two–dimensional integer DCT–II of size 8×8

The two–dimensional (2–d) DCT–II has important applications in image compression (JPEG, MPEG). Therefore we extend our results of Section 4 to the 2–d integer DCT–II. Let $X \in \mathbb{Z}^{8\times 8}$ be given. Then the 2–d DCT–II of size 8×8 of X is defined by $C_8^{II}\,X\,(C_8^{II})^T$. Let $\hat{Y} = 4\,C_8^{II}\,X\,(4\,C_8^{II})^T$. The simple *row–column method* for computing of $\hat{Y}$ is based on the observation

$$\hat{Y} = (4\,C_8^{II}\,X)\,(4\,C_8^{II})^T = \hat{Z}\,(4\,C_8^{II})^T = (4\,C_8^{II}\,\hat{Z}^T)^T$$

with $\hat{Z} := 4\,C_8^{II}\,X$. Now we compute integer approximations of $\hat{Z}$ and $\hat{Y}$ by Algorithm 4.1.

Algorithm 5.1 [2–d integer DCT–II algorithm]
Input: $X = (\mathbf{x}_0, \ldots, \mathbf{x}_7) \in \mathbb{Z}^{8\times 8}$.

1. For $k = 0, \ldots, 7$ compute the integer approximation $\mathbf{z}_k$ of $\hat{\mathbf{z}}_k := 4\,C_8^{II}\mathbf{x}_k$ by Algorithm 4.1.

2. Set $Z := (\mathbf{z}_0, \ldots, \mathbf{z}_7)$ and $(\mathbf{u}_0, \ldots, \mathbf{u}_7) := Z^T$.

3. For $k = 0, \ldots, 7$ compute the integer approximation $\mathbf{v}_k$ of $\hat{\mathbf{v}}_k := 4\, C_8^{II} \mathbf{u}_k$ by Algorithm 4.1.

4. Form $Y := (\mathbf{v}_0, \ldots, \mathbf{v}_7)^T$.

Output: $Y \in \mathbb{Z}^{8\times 8}$ integer approximation of $\hat{Y} = 16\, C_8^{II} X\, (C_8^{II})^T$.

The inverse 2–d integer–DCT–II of size 8×8 follows now immediately:

Algorithm 5.2 [Inverse 2–d integer DCT–II algorithm]
Input: $Y \in \mathbb{Z}^{8\times 8}$ (integer approximation of $16\, C_8^{II} X\, (C_8^{II})^T$ computed by Algorithm 5.1).

1. Form $(\mathbf{v}_0, \ldots, \mathbf{v}_7) := Y^T$.

2. For $k = 0, \ldots, 7$ compute the vector $\mathbf{u}_k \in \mathbb{Z}^8$ via Algorithm 4.2 such that $\mathbf{v}_k$ is the integer approximation of $4\, C_8^{II} \mathbf{u}_k$ by Algorithm 4.1.

3. Set $Z := (\mathbf{u}_0, \ldots, \mathbf{u}_7)^T$ and $(\mathbf{z}_0, \ldots, \mathbf{z}_7) := Z$.

4. For $k = 0, \ldots, 7$ compute the vector $\mathbf{x}_k \in \mathbb{Z}^8$ via Algorithm 4.2 such that $\mathbf{z}_k$ is the integer approximation of $4\, C_8^{II} \mathbf{x}_k$ by Algorithm 4.1.

Output: $X := (\mathbf{x}_0, \ldots, \mathbf{x}_7) \in \mathbb{Z}^{8\times 8}$ original matrix of Algorithm 5.1.

Let us consider the (worst case) truncation error of Algorithm 5.1 estimated in the Frobenius norm.

Theorem 5.3 *Let $X \in \mathbb{Z}^{8\times 8}$ be an arbitrary matrix. Using Algorithm 5.1, the resulting integer approximation $Y \in \mathbb{Z}^{8\times 8}$ of $\hat{Y} = 16\, C_8^{II}\, X\, (C_8^{II})^T$ satisfies the error estimate*

$$\|\hat{Y} - Y\|_F < 30\, \sqrt{2 + \sqrt{2}} + 10\, \sqrt{3} \approx 72.753280.$$

Proof. By Theorem 4.4, we know that the computed vectors $\mathbf{z}_k$ $(k = 0, \ldots, 7)$ in step 1 of Algorithm 5.1 satisfy the estimate

$$\|\mathbf{z}_k - \hat{\mathbf{z}}_k\|_2^2 < \left(3\, \sqrt{1 + \tfrac{\sqrt{2}}{2}} + \sqrt{\tfrac{3}{2}}\right)^2 .$$

Summing up, this yields

$$\|Z - \hat{Z}\|_F^2 = \textstyle\sum_{k=0}^{7} \|\mathbf{z}_k - \hat{\mathbf{z}}_k\|_2^2 < 8\, \left(3\, \sqrt{1 + \tfrac{\sqrt{2}}{2}} + \sqrt{\tfrac{3}{2}}\right)^2$$

with the matrix $\hat{Z} := (\hat{\mathbf{z}}_0, \ldots, \hat{\mathbf{z}}_7)$. Hence,

$$\|Z - \hat{Z}\|_F < 6\, \sqrt{2 + \sqrt{2}} + 2\sqrt{3}.$$

Set $(\hat{\mathbf{u}}_0, \ldots, \hat{\mathbf{u}}_7)^T := \hat{Z}^T$ and $\tilde{\mathbf{v}}_k := 4\, C_8^{II} \hat{\mathbf{u}}_k$ $(k = 0, \ldots, 7)$. Further let $\mathbf{v}_k$ $(k = 0, \ldots, 7)$ be the computed vectors in step 3 of Algorithm 5.1. Applying again Theorem 4.4, we get

$$\|\mathbf{v}_k - \tilde{\mathbf{v}}_k\|_2^2 < \left(3\, \sqrt{1 + \tfrac{\sqrt{2}}{2}} + \sqrt{\tfrac{3}{2}}\right)^2$$

and hence

$$\|Y - \tilde{Y}\|_F < 6\sqrt{2+\sqrt{2}} + 2\sqrt{3}$$

with $Y := (\mathbf{v}_0, \ldots, \mathbf{v}_7)^T$ and $\tilde{Y} := (\tilde{\mathbf{v}}_0, \ldots, \tilde{\mathbf{v}}_7)^T = Z\,(4\,C_8^{II})^T$. Since the Frobenius norm is unitarily invariant, we have $\|\tilde{Y} - \hat{Y}\|_F = 4\,\|Z - \hat{Z}\|_F$. The Frobenius norm is consistent such that we can estimate

$$\|Y - \hat{Y}\|_F \leq \|Y - \tilde{Y}\|_F + \|\tilde{Y} - \hat{Y}\|_F \leq 30\sqrt{2+\sqrt{2}} + 10\sqrt{3}.$$

This completes the proof. *q.e.d.*

Remark 5.4 *Instead of Algorithm* 4.1*, we also can use another integer* DCT–II *algorithm in the row–column method. Considering for example* $\hat{Y} := (2\,C_8^{II})\,X\,(2\,C_8^{II})^T$ *for a given integer matrix* $X \in \mathbb{Z}^{8\times 8}$*, we can use Algorithm* 4.6 *to find an integer approximation* Y *of* $\hat{Y}$*. In the worst case, we even obtain the error estimate*

$$\|\hat{Y} - Y\|_F < 48.737963$$

by the same method as in the proof of Theorem 5.3*. This much better error estimate is especially due to the smaller expansion factor in Algorithm* 4.6*.*

6 Numerical results

We want to apply the four algorithms proposed in Section 4 and compare them regarding their truncation errors and their arithmetical complexity. The following examples already show the different behaviour of the Algorithms 4.1 (A), 4.3 (B), 4.5 (C), and 4.6 (D) in Section 4.
Let $\mathbf{x} \in \mathbb{Z}^8$ be a given integer vector. Let $\mathbf{y}_\circ$ denote the result of the integer DCT–II algorithm $\circ$ with $\circ \in \{A, B, C, D\}$. Further, let $\hat{\mathbf{y}}_A = 4\,C_8^{II}\mathbf{x}$, $\hat{\mathbf{y}}_B = 4\sqrt{2}\,C_8^{II}\mathbf{x}$, $\hat{\mathbf{y}}_C = C_8^{II}\mathbf{x}$, $\hat{\mathbf{y}}_D = 2C_8^{II}\mathbf{x}$ be the exact vectors after applying (scaled) DCT–II of length 8. In the following tables we give the components of exact vectors $\hat{\mathbf{y}}_\circ$ (rounded to 3 decimal places) and the components of $\mathbf{y}_\circ$ for 3 examples of $\mathbf{x}$.
1. Let $\mathbf{x} := (100, 100, 100, 100, 0, 0, 0, 0)^T$.

$\hat{\mathbf{y}}_A$	565.685	512.583	0.000	−179.995	0.000	120.269	0.000	−101.959
$\mathbf{y}_A$	566	512	0	−180	0	120	0	−102
$\hat{\mathbf{y}}_B$	800.000	724.902	0.000	−254.551	0.000	170.086	0.000	−144.192
$\mathbf{y}_B$	800	721	−1	−254	0	170	0	−144
$\hat{\mathbf{y}}_C$	141.421	128.146	0.000	−44.999	0.000	30.067	0.000	−25.490
$\mathbf{y}_C$	141	130	0	−45	0	30	0	−26
$\hat{\mathbf{y}}_D$	282.843	256.292	0.00	−89.998	0.000	60.134	0.000	−50.980
$\mathbf{y}_D$	283	256	0	−90	0	61	0	−51

For the Euklidian errors we obtain

$$\begin{aligned} \|\hat{\mathbf{y}}_A - \mathbf{y}_A\|_2 &= 0.716236, \quad \|\hat{\mathbf{y}}_B - \mathbf{y}_B\|_2 = 4.071106, \\ \|\hat{\mathbf{y}}_C - \mathbf{y}_C\|_2 &= 1.969911, \quad \|\hat{\mathbf{y}}_D - \mathbf{y}_D\|_2 = 0.926969. \end{aligned}$$

The absolute errors in the components can be seen from the table.

2. Let $\mathbf{x} := (1,2,3,4,5,6,7,8)^T$.

$\hat{\mathbf{y}}_A$	50.912	−25.769	0.000	−2.694	0.000	−0.804	0.000	−0.203
$\mathbf{y}_A$	51	−28	0	−3	0	−1	0	−1
$\hat{\mathbf{y}}_B$	72.000	−36.443	0.000	−3.810	0.000	−1.136	0.000	−0.287
$\mathbf{y}_B$	72	−40	−1	−4	0	0	0	0
$\hat{\mathbf{y}}_C$	12.728	−6.442	0.000	−0.673	0.000	−0.201	0.000	−0.051
$\mathbf{y}_C$	11	−7	0	−1	0	−1	0	0
$\hat{\mathbf{y}}_D$	25.456	−12.885	0.000	−1.347	0.000	−0.402	0.000	−0.101
$\mathbf{y}_D$	25	−13	0	−1	0	−1	0	0

For the Euklidian errors we find

$$\begin{aligned} \|\hat{\mathbf{y}}_A - \mathbf{y}_A\|_2 &= 2.398267, \quad \|\hat{\mathbf{y}}_B - \mathbf{y}_B\|_2 = 3.880764, \\ \|\hat{\mathbf{y}}_C - \mathbf{y}_C\|_2 &= 2.011088, \quad \|\hat{\mathbf{y}}_D - \mathbf{y}_D\|_2 = 0.842357. \end{aligned}$$

3. Let $\mathbf{x} := (-30,-94,-112,60,26,-79,27,38)^T$.

$\hat{\mathbf{y}}_A$	−231.931	−358.004	−49.220	−38.915	497.803	205.456	−288.821	−13.655
$\mathbf{y}_A$	−232	−358	−49	−39	498	205	−288	−14
$\hat{\mathbf{y}}_B$	−328.000	−506.294	−69.607	−55.034	704.000	290.559	−408.454	−19.311
$\mathbf{y}_B$	−328	−509	−71	−55	704	291	−409	−18
$\hat{\mathbf{y}}_C$	−57.983	−89.501	−12.305	−9.729	124.451	51.364	−72.205	−3.414
$\mathbf{y}_C$	−58	−90	−13	−9	125	52	−72	−4
$\hat{\mathbf{y}}_D$	−115.966	−179.002	−24.610	−19.457	248.902	102.728	−144.410	−6.827
$\mathbf{y}_D$	−116	−179	−24	−20	249	103	−144	−7

For the Euklidian errors we obtain

$$\begin{aligned} \|\hat{\mathbf{y}}_A - \mathbf{y}_A\|_2 &\approx 1.048921, \quad \|\hat{\mathbf{y}}_B - \mathbf{y}_B\|_2 \approx 3.387218, \\ \|\hat{\mathbf{y}}_C - \mathbf{y}_C\|_2 &\approx 1.534718, \quad \|\hat{\mathbf{y}}_D - \mathbf{y}_D\|_2 \approx 0.974273. \end{aligned}$$

We consider the behaviour of the errors $\|\hat{\mathbf{y}}_\circ - \mathbf{y}_\circ\|_2$ and $\|\hat{\mathbf{y}}_\circ - \mathbf{y}_\circ\|_\infty$ generated by the four algorithms $\circ \in \{A, B, C, D\}$ in more detail . As input vectors we use 1000 random vectors in $\mathbb{Z}^8$ with entries in the range $[-1023, 1024]$. We compute the r–th–quantiles for $r = \frac{j}{10}$, $j = 1, \ldots, 10$ for each algorithm. After sorting the errors of 1000 resulting vectors, the r–th–quantile is the smallest value that separates the errors into two parts; $1000r$ of the sorted errors are less than the quantile value, the other $1000\,(1-r)$ errors are greater than the quantile. The 1–th–quantile is the maximal error occurring. In the following tables the r-th quantiles are rounded to three decimal places.

Alg.	r=0.1	r=0.2	r=0.3	r=0.4	r=0.5	r=0.6	r=0.7	r=0.8	r=0.9	r=1.0
A	1.184	1.398	1.584	1.733	1.949	2.189	2.379	2.578	2.821	4.006
B	2.726	3.026	3.194	3.366	3.517	3.688	3.837	4.029	4.237	5.004
C	1.220	1.413	1.534	1.652	1.771	1.876	2.003	2.125	2.296	3.097
D	0.888	1.012	1.110	1.191	1.276	1.353	1.426	1.521	1.656	2.438

Table 1. r-th quantiles for the error $\|\hat{\mathbf{y}}_\circ - \mathbf{y}_\circ\|_2$ *with* $\circ \in \{A, B, C, D\}$

Alg.	r=0.1	r=0.2	r=0.3	r=0.4	r=0.5	r=0.6	r=0.7	r=0.8	r=0.9	r=1.0
A	0.764	0.930	1.061	1.211	1.360	1.503	1.657	1.858	2.119	3.735
B	1.998	2.268	2.452	2.672	2.848	3.010	3.180	3.409	3.677	4.534
C	0.744	0.883	0.958	1.039	1.108	1.194	1.278	1.397	1.576	2.345
D	0.535	0.631	0.697	0.759	0.822	0.894	0.966	1.070	1.245	2.270

Table 2. r-th quantiles for the error $\|\hat{\mathbf{y}}_\circ - \mathbf{y}_\circ\|_\infty$ *with* $\circ \in \{A, B, C, D\}$

The numerical results show that Algorithm 4.6 (D) is most favourable and outperforms the other algorithms. Algorithm 4.6 possesses very small worst case truncation errors and provides suitable integer approximations for the DCT–II of length 8, as seen in the numerical tests. The average Euclidian error of this algorithm is less than 1.3 and the average maximum error is even smaller than 1. That means, in most cases algorithm D provides in each component one of the two nearest integers to the exact DCT component value. Taking the arithmetical complexity into account, Algorithm 4.6 is most recommended.

Finally, let us look at the 2–d DCT–II applying Algorithm 5.1 in one example. Now by A we denote Algorithm 5.1, where Algorithm 4.1 (A) is used. By D we denote the modified Algorithm 5.1, where we apply Algorithm 4.6 (D) instead of Algorithm 4.1. Let X be an input matrix of order 8, Y_A resp. Y_D are the 2–d integer DCT–II of X computed by method A resp. D, and $\hat{Y}_A$, $\hat{Y}_D$ are the corresponding exact 2–d DCT–II of X, where each entry is rounded to the next integer. For

$$X := \begin{pmatrix} 11 & 16 & 21 & 25 & 27 & 27 & 27 & 27 \\ 16 & 23 & 25 & 28 & 31 & 28 & 28 & 28 \\ 22 & 27 & 32 & 35 & 30 & 28 & 28 & 28 \\ 31 & 33 & 34 & 32 & 32 & 31 & 31 & 31 \\ 31 & 32 & 33 & 34 & 34 & 27 & 27 & 27 \\ 33 & 33 & 33 & 33 & 32 & 29 & 29 & 29 \\ 34 & 34 & 33 & 35 & 34 & 29 & 29 & 29 \\ 34 & 34 & 33 & 33 & 35 & 30 & 30 & 30 \end{pmatrix}$$

we obtain that

$$\hat{Y}_A = \begin{pmatrix} 3770 & -17 & -193 & -83 & 34 & -27 & -43 & 21 \\ -361 & -280 & -100 & -51 & -46 & -1 & 7 & -19 \\ -175 & -148 & -25 & 24 & 3 & -15 & -9 & -1 \\ -113 & -31 & 4 & 23 & 14 & -1 & -1 & 5 \\ -10 & -13 & 24 & 25 & -2 & -11 & 10 & 20 \\ 28 & -3 & 26 & -5 & -12 & 24 & 17 & -16 \\ -21 & -6 & -5 & -23 & -8 & 28 & 17 & -12 \\ -42 & 25 & -60 & -30 & 30 & 19 & -9 & -7 \end{pmatrix},$$

$$Y_A = \begin{pmatrix} 3772 & -18 & -194 & -82 & 35 & -24 & -44 & 21 \\ -378 & -272 & -105 & -53 & -44 & -3 & 8 & -13 \\ -175 & -157 & -25 & 23 & 0 & -13 & -11 & -6 \\ -112 & -28 & 4 & 24 & 16 & 0 & 0 & 6 \\ -13 & -13 & 23 & 27 & -1 & -9 & 10 & 20 \\ 27 & -3 & 27 & -8 & -16 & 24 & 18 & -17 \\ -21 & -6 & -7 & -23 & -7 & 27 & 17 & -10 \\ -33 & 20 & -58 & -28 & 30 & 20 & -11 & -11 \end{pmatrix},$$

and

$$\hat{Y}_D = \begin{pmatrix} 943 & -4 & -48 & -21 & 9 & -7 & -11 & 5 \\ -90 & -70 & -25 & -13 & -11 & 0 & 2 & -5 \\ -44 & -37 & -6 & 6 & 1 & -4 & -2 & 0 \\ -28 & -8 & 1 & 6 & 4 & 0 & 0 & 1 \\ -2 & -3 & 6 & 6 & 0 & -3 & 2 & 5 \\ 7 & -1 & 6 & -1 & -3 & 6 & 4 & -4 \\ -5 & -1 & -1 & -6 & -2 & 7 & 4 & -3 \\ -10 & 6 & -15 & -7 & 7 & 5 & -2 & -2 \end{pmatrix},$$

$$Y_D := \begin{pmatrix} 942 & -5 & -49 & -19 & 9 & -8 & -12 & 6 \\ -92 & -70 & -26 & -13 & -10 & 2 & 2 & -5 \\ -43 & -37 & -8 & 5 & 1 & -3 & -2 & 0 \\ -32 & -4 & -1 & 5 & 5 & 0 & 0 & 2 \\ -2 & -6 & 5 & 6 & 0 & -2 & 3 & 6 \\ 4 & -3 & 6 & 0 & -2 & 4 & 3 & -3 \\ -3 & -1 & -2 & -5 & -2 & 7 & 5 & -3 \\ -12 & 6 & -14 & -7 & 8 & 4 & -1 & -2 \end{pmatrix}.$$

We get the truncation errors

$$\|\hat{Y}_A - Y_A\|_F \approx 27.211073, \qquad \|\hat{Y}_D - Y_D\|_F \approx 10.240275.$$

The above example is taken from [15]. Further examples show that Algorithm 5.1 together with Algorithm 4.6 (D) is considerably better.

Acknowledgement. The authors would like to thank S. Dekel for very instructive remarks improving the paper.

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

[5] E. Feig and S. Winograd, Fast algorithms for the discrete cosine transform, IEEE Trans. Signal Process. **40** (1992), 2174 – 2193.

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

[7] 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.

[8] M.W. Marcellin, M.J. Gormish, A. Bilgin, and M.P. Boliek, An overview of JPEG–2000, Proc. Data Compression Conf., 2000, pp. 523 – 541.

[9] W. Philps, Lossless DCT for combined lossy/lossless image coding, Proc. IEEE Internat. Conf. Image Process., vol. 3, 1998, pp. 871 – 875.

[10] G. Plonka and M. Tasche, Split–radix algorithms for discrete trigonometric transforms, Preprint, Gerhard–Mercator–Univ. Duisburg, 2002.

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

[12] U. Schreiber, Fast and numerically stable trigonometric transforms (in German), Thesis, Univ. of Rostock, 1999.

[13] G. Strang, The discrete cosine transform, SIAM Rev. **41** (1999), 135 – 147.

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

[15] G.K. Wallace, The JPEG still picture compression standard, Comm. ACM **34** (1991), 32–44.

[16] Y. Zeng, L. Cheng, G. Bi, and A.C. Kot, Integer DCTs and fast algorithms, IEEE Trans. Signal Process. **49** (2001), 2774 – 2782.