International Journal of Chemistry, Mathematics and Physics (IJCMP)
[Vol-6, Issue-5, Sep-Oct, 2022]
https://dx.doi.org/10.22161/ijcmp.6.5.3
ISSN: 2456-866X

Application of the Transpositions matrix for obtaining n-dimensional rotation matrices

Ognyan Ivanov Zhelezov, Valentina Markova Petrova

Department of Information Technology, Nikola Vaptsarov Naval Academy, Varna, Bulgaria

Received: 09 Sep 2022; Received in revised form: 01 Oct 2022; Accepted: 07 Oct 2022; Available online: 30 Oct 2022

Abstract— *This article proposes an algorithm for generation of N-dimensional rotation matrix R, N=m+n, $m=2^p$, $n=2^q$, p,q ∈ [2, 4, 8] which rotates given N-dimensional vector X in the direction of coordinate axis x_1 Algorithm uses block diagonal matrix, composed by Transpositions matrices. As practical realization article gives Matlab code of functions, which creates Householder and Transpositions matrices and V matrix for given n-dimensional vector X.*

Keywords— ***Applied Mathematics, Mathematics of computing, Mathematical analysis, Numerical analysis, Computations on matrices.***

I. INTRODUCTION

This paper proposes an algorithm for creating an n-dimensional rotation matrix V, which rotates a given vector X in the direction of the coordinate axis x_1 using block diagonal Transpositions (Trs) matrices. Matrices that rotate a given vector X in the direction of a given coordinate axis are used in calculations such as QR decomposition and in obtaining a rotation matrix according to the NRMG algorithm [2].

II. ROTATION OF N-DIMENSIONAL VECTOR X TO THE DIRECTION OF AXIS X_1

The rotation of an N-dimensional vector X to the direction of one of the coordinate axes is a basic operation of the NRMG algorithm for obtaining an n-dimensional rotation matrix [2] and for QR decomposition [9] of a matrix A into a product A = QR of an orthogonal matrix Q and an upper triangular matrix R.

Definition 1: We will call a V matrix or V_X matrix an orthogonal matrix that rotates a given vector X in the direction of the x_1 coordinate axis.

To obtain a V matrix for dense vectors, the method using Householder reflections is considered the most effective [7],[13], while for sparse vectors the method using Givens rotations may be more effective. However, for vectors of dimensions 2, 4 and 8, the method using a Transpositions matrix is the most effective [1]. The following is a description of these three methods for obtaining an orthogonal matrix V that rotates a given vector X to the direction of given coordinate axis (e.g. axis x_1).

2.1 OBTAINING AN V MATRIX USING HOUSEHOLDER REFLECTION

According to the transformation of Householder [13] if X and Y are vectors with the same norm there exists an orthogonal symmetric matrix P, which rotates vector X to the direction of vector Y such that

$$\begin{aligned} &Y = PX \text{ where } P = I - W.W^T \\ &\text{and } W = (X - Y)/\|X - Y\| \end{aligned} \quad (1)$$

In particular, if Y is unit vector $Y=[1,0, \dots 0]^T$, we get

$$\begin{aligned} &Y = P\overline{X} \text{ where } P = I - W.W^T \\ &\text{and } W = (\overline{X} - Y)/\|\overline{X} - Y\| \\ &\overline{X} = X/\|X\| \end{aligned} \quad (2)$$

then matrix P will rotates vector X to the direction of the coordinate axis x1 Matrix P is matrix of reflection (not a rotation) because det(P) = -1, (which gives the name to method)

The following example shows the Matlab code of the function Householder(X,k) that creates and returns orthogonal matrix V which rotates given vector X to the

direction of the coordinate axis x_k using Householder transformation.

```
function V= Householder (X, k)
  m=length(X);
  k=fix(k);
  v=X;
  v(k)=X(k)- norm(X);
  s=norm(v);
if s~=0
  w=v/s;
  V=eye(m)-2*w*w';
else
  V= eye(m);
end
```

Example 1 Code of Matlab function, which returns orthogonal matrix V, that rotates vector X to the direction of the coordinate axis x_k using Householder transformation

As it can be seen, function Householder(x,k) performs two normalization of vectors and calculates elements of matrix $w.w^T$, which needs n^2 floating points multiplications.

2.2 OBTAINING AN V MATRIX USING GIVENS ROTATIONS

Rotation of given vector X in the direction of one of coordinate axes (e.g. axis x1) can be performed by subsequent multiplications by Givens matrices [2] as follows:

$$X_{(N-1)} = \prod_{k=N-1}^{1} G(k, k+1, \theta_k).X = M_X.X = [r_X, 0, ...0, 0]^T \quad .(3)$$

where G(k, k+1, θk) is Givens matrix, which rotates vector in coordinate plane (k, k+1) by angle θk and r_X is the norm of vector X.

Givens matrices G(k, k+1, θk), k = N-1, n-2, … 1 are defined as follows [4],[5]:

$$G(k, k+1, \theta_k) = \begin{vmatrix} 1 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \iddots & \vdots & \vdots \\ 0 & 0 & \cdots & C_k & -S_k & \cdots & 0 & 0 \\ 0 & 0 & \cdots & S_k & C_k & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \iddots & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 \end{vmatrix} \quad (4)$$

where θk is the angle of rotation and coefficients Ck=cos(θk) and Sk=sin(θk) appears at the intersections of k-th and k+1 -th rows and columns.

The target of multiplication by Givens matrix G(k, k+1, θk) is to set to zero coordinate $\bar{x}_{k+1}$. It is easy to find that equation $\bar{x}_{k+1} = 0$ and equations (5) are satisfied simultaneously when sin(θk) and cos(θk) are calculated using formulas (5) [2]:

$$\begin{cases} \sin(\theta_k) = -\dfrac{x_{k+1}}{\sqrt{x_k^2 + x_{k=1}^2}}, \\ \cos(\theta_k) = \dfrac{x_k}{\sqrt{x_k^2 + x_{k=1}^2}} \text{ if } x_k^2 + x_{k=1}^2 > 0 \\ \sin(\theta_k) = 0, \cos(\theta_k) = 1 \text{ if } x_k^2 + x_{k=1}^2 = 0 \end{cases} \quad (5)$$

Thus searched matrix V_X can be calculated as multiplication of Givens matrices

$$G(N-1, N-2, \theta_{N-1}), G(N-2, N-3, \theta_{N-2}), ..., G(1,2,\theta_1)$$

as follows:

$$V_X = \prod_{k=1}^{N-1} G(N-k, N-k+1, \theta_{N-k}) \quad (6)$$

where coefficients of rotation sin(θk) and cos(θk) are calculated using formulas (5). Calculation of angles of two-dimensional rotations θk really is not needed. If $x_k^2 + x_{k+1}^2 = 0$ then corresponding Givens matrix is equal to Identity matrix G(k, k+1, θk) = I – no rotation.

2.3 OBTAINING AN V MATRIX USING TRANSPOSITIONS_MATRIX

Definition 2 Transpositions matrix (Tr matrix) [1] is square matrix with dimension $n=2^m$, $m \in N$, which rows are permutations of elements x_k, $k \in [0, n-1]$ of given n-dimensional vector X and the values of elements $Tr_{p,q}$ of matrix are obtained from the elements of given vector as follows:

$$Tr_{p,q} = X_{p \oplus q}, \; p, q \in [0,1,..., n-1] \quad (7)$$

where $\oplus$ denotes operation 'bitwise exclusive or' (XOR) [15],[16],[17].

Definition 3 Transpositions matrix with mutually orthogonal rows (Trs matrix) [1],[2] is square matrix with dimension n=2,4 or 8, which is obtained as Hadamard product [10], (denoted by ○) of Tr matrix and n-dimensional Hadamard matrix whose rows (except the first one) are rearranged relative to the rows of Sylvester-Hadamard matrix [10] in order $R=[1, r_2, \ldots r_n]^T$, $r_2, \ldots,$

$r_n \in [2,n]$ for which the rows of the resulting Trs matrix are mutually orthogonal.

By definition, Trs(X) is a matrix whose rows are mutually orthogonal and contain transpositions of a given vector X, while the first row contains the values of X without inversions. Therefore, if ||X||=1 then the multiplication of Trs(X/||X||) by X gives unit vector to the direction of the coordinate axis x_1. The preparation of an orthogonal Trs matrix from a Tr matrix and a Sylvester-Hadamard matrix for a given vector X is given by the formula:

$$\begin{aligned} &\mathrm{Trs}(X)=\mathrm{Ts}(\overline{X})\circ H(P) \\ &\overline{X}=X/\|X\| \end{aligned} \tag{8}$$

where:

- ○ denotes operation Hadamard product of matrices [13],
- H(P) is n-dimensional Hadamard matrix, which rows are interchanged against the Sylvester-Hadamard matrix in given order $P=[1, r_2, \dots r_n]^T$, $r_2, \dots, r_n \in [2,n]$ for which the rows of the resulting Trs matrix are mutually orthogonal.
- X is the vector from which the elements of Tr matrix are derived.

The formula below shows obtaining of 4-dimensional Trs matrix for a vector $X=[x1,x2, x3, x4]^T$ and for H(P) rows ordering $P=[1\ 2\ 4\ 3]^T$.

$$\mathrm{Trs}(X)=H(P)\circ \mathrm{Tr}(X)= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \end{pmatrix} \circ \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ x_2 & x_1 & x_4 & x_3 \\ x_3 & x_4 & x_1 & x_2 \\ x_4 & x_3 & x_2 & x_1 \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ x_2 & -x_1 & x_4 & -x_3 \\ x_3 & -x_4 & -x_1 & x_2 \\ x_4 & x_3 & -x_2 & -x_1 \end{pmatrix} \tag{9}$$

It is important to note, that the ordering P of Hadamard matrix's rows (against the Sylvester-Hadamard matrix) does not depend on the vector X but only on its dimension.

An algorithm for creating Trs matrices for an arbitrary vector X for dimensions 2, 4 and 8 is proposed in [1].

The following example shows the code of a Matlab function that creates and returns Trs matrix which rotates given vector X of dimension n = 2, 4, or 8 in the direction of coordinate axis x_1.

```
function V = TrsMatrix(X)
H2=[1 1; 1 -1];H4=[1 1 1 1; 1 -1 1 -1; 1 -1 -1 1; 1 1 -1 -1];
H8=[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];
N=length(X); V=zeros(N);
if N==1 V=X;
else
if N==8 H=H8;
  else if N==4 H=H4;
    else if N==2 H=H2;
      else error('length of X must be 2,4 or 8');
end; end; end;
 for col=1:N
  step=col-1;  %Step of displacements
  for row=1:N    % Loop for rows
   new_row =bitxor(row-1, step)+1;
if H(new_row,col)==1 %set Trs value
   V(new_row,col)=X(row);
else
   V(new_row,col)=-X(row);
 end; end; end; end
```

Example 2 Matlab function, that creates and returns a Trs matrix for a given vector X of size n = 1, 2, 4, or 8.

As it can be seen, function TrsMatrix(X) obtains elements of matrix V, which needs n^2 bitwise multiplications of the indices of the vector's elements.

III. COMPARISON OF THE COMPUTATIONAL EFFICIENCY OF THE METHODS FOR CREATING AN V MATRIX

We will compare the computational efficiency of the methods for creating a matrix that rotates an n-dimensional vector in the direction of the x1 coordinate axis (V matrix) using Householder Reflection and using Transpositions_matrix.

3.1 THE COMPUTATIONAL EFFICIENCY OF THE METHOD THAT USES HOUSEHOLDER REFLECTION

Rotation of given vector X in the direction of one of coordinate axes (e.g. axis x_1) can be performed by subsequent multiplications by Givens matrices [2] as follows:

As can be seen from formula (2), to obtain a matrix P (which for us is the required matrix V) the vector Y must be a unit vector $Y = [1,0,\dots, 0]^T$. The calculation of the matrix R includes the calculation of the norm of the vector

X, the norm of the vector W = X / || X || - Y and the calculation of the elements of the matrix $W.W^T$, which needs n^2 multiplications. Other computational operations due to their lower weight can be ignored. So, for the volume C_H of computational operations when using Householder Reflection we can write the following estimate:

$$C_H = 2 \text{ norms} + n^2 \text{ multiplications} \quad \text{.....(10)}$$

3.2 THE COMPUTATIONAL EFFICIENCY OF THE METHOD THAT USES TRANSPOSITIONS MATRIX

As can be seen from formula (6), obtaining a Transpositions matrix Trs (X) for a given vector X with dimension 2, 4 or 8 includes the following operations:

- normalization of the vector X
- obtaining a Transpositions matrix $\mathrm{Tr}\left(\overline{X}\right)$ for vector $\overline{X} = X/\|X\|$.
- element-by-element multiplication (Hadamard product) of the matrix $\mathrm{Tr}\left(\overline{X}\right)$ by a matrix H(P) having the same dimension.

As can be seen from the definition of the Trs matrix (2,3), its production includes n^2 bitwise multiplications of the indices of the vector elements, and (since the matrix elements H(P) have values =1 and -1), the element-by-element multiplication of the matrix $\mathrm{Tr}\left(\overline{X}\right)$ by given matrix H(P) having the same dimension. These element-by-element multiplications are actually a change in the sign of the elements of the matrix $\mathrm{Tr}\left(\overline{X}\right)$, for which the corresponding elements of the matrix H(P) have a value of -1. Thus, for the volume CTrs of computational operations when using Transpositions_matrix, we can record the following estimate

$$CTrs = 1 \text{ norm} + n^2 \text{ bitwise multiplications} + n^2 \text{ changes of sign} \quad (11)$$

Since bitwise multiplications and changes of sign of integer values takes less microprocessors cycles than cycles of floating point multiplication, the comparison of the volumes of the computational operations CH and CTrs shows that

$$CTrs < CH \quad (12)$$

and therefore the method for creating a 4- and 8-dimensional orthogonal matrix R that rotates a given vector X in the direction of the x_1 coordinate axis that uses Transpositions_matrix has higher performance than the method that uses Householder Reflection

3.3 THE COMPUTATIONAL EFFICIENCY OF THE METHOD THAT USES BLOCK DIAGONAL TRANSPOSITIONS MATRICES

In [1] the possibility of using Transpositions matrices to obtain an orthogonal matrix that rotates a given N-dimensional vector X in the direction of a given coordinate axis (i.e. axis x_1) is considered. Block diagonal Transpositions matrices are used for this purpose. This paper examines the effectiveness of a method that uses a block-diagonal matrix Trb, consisting of two blocks of Trs matrices. When multiplying the block-diagonal matrix Trb by the specified vector X, it is obtained a vector Y1, which contains two non-zero elements. By multiplying the vector Y1 with a Givens matrix, is obtained the vector Y2, which is in the direction of a coordinate axis x_1. Obtaining this result can be represented as follows:

$$\begin{aligned} Y2 &= G(1,m+1,\theta).Y1 = G(1,m+1,\theta).Trb.X \\ &= G(1,m+1,\theta).\begin{bmatrix} Trs(X1/\|X1\|) & 0 \\ 0 & Trs(X2/\|X2\|) \end{bmatrix}.X \\ &= R.X = \left[\|X\|,0\ldots,0\right]^T \end{aligned} \quad (13)$$

where:

- X=[X1,X2]T is given vector of dimension N ≤ 16.
- X1, X2 are parts ot input vector X=[X1,X2] of dimensions m, n, N=m+n, m=2p, n=2q, p,q [2, 4, 8]
- Trs(X1/||X1||) and Trs(X2/||X2||) are Transpositions matrices [1]
- G(1,m+1,θ) is Givens matrix, which rotates vector Y1 in the direction of axis x1.

3.3.1 ALGORITHM FOR CALCULATING V MATRIX USING BLOCK DIAGONAL T_{RB} MATRIX:

The algorithm for calculating V matrix using block diagonal Trb matrix, includes the following operations:

Step 1: Split the given vector X into vectors X1 and X2 with dimensions 2, 4 or 8.

Step 2: From the given vector $X = [X1, X2]^T$ a vector $\overline{X}$ is obtained. The two parts of vector $\overline{X}$ - the vectors $\overline{X}1$ and $\overline{X}2$, are calculates by vectors X1 and X2 as it is given in formulas:

$$\begin{aligned} \overline{X}1 &= X1/\|X1\| \\ \overline{X}2 &= X2/\|X2\| \end{aligned} \quad (14)$$

Step 3: The matrix V is obtained as follows:

$$V = G_n(1, m+1, \theta).Trb$$
$$= G_n(1, m+1, \theta).\begin{bmatrix} Trs(\overline{X}1) & 0 \\ 0 & Trs(\overline{X}2) \end{bmatrix} \quad (15)$$

where:

- Gn(1,m+1, θ) is Givens matrix, which has coefficients

$$C = \|X1\|/\|X\|,\ S = \|X2\|/\|X\| \quad (16)$$

- Trb is block diagonal matrix, which consists blocks $Trs(\overline{X}1)$ and $Trs(\overline{X}2)$ as it is given above.

Multiplying the matrix V obtained by formula (15) by the vector X gives as a result vector $Y=[\|X\|,0,\ldots,0]^T$. The matrix V calculated by formula (15) has the form:

$$V = \left[\begin{array}{cccc|ccc} c.\overline{x}_1 & c.\overline{x}_2 & \ldots & c.\overline{x}_m & -s.\overline{x}_{m+1} & -s.\overline{x}_{m+2} & \ldots & -s.\overline{x}_n \\ \overline{x}_2 & -\overline{x}_1 & \ldots & -\overline{x}_{m-1} & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \overline{x}_m & \overline{x}_{m-1} & \ldots & -\overline{x}_1 & 0 & 0 & \ldots & 0 \\ \hline s.\overline{x}_1 & s.\overline{x}_2 & \ldots & s.\overline{x}_m & c.\overline{x}_{m+1} & c.\overline{x}_{m+2} & \ldots & c.\overline{x}_n \\ 0 & 0 & \ldots & 0 & \overline{x}_{m+2} & -\overline{x}_{m+1} & \ldots & -\overline{x}_{n-1} \\ \ldots & \ldots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & \overline{x}_n & \overline{x}_{n-1} & \ldots & -\overline{x}_{m+1} \end{array}\right]$$

Fig. 1 Scheme for computing the V matrix

As can be seen from Fig. 1, obtaining the V matrix involves obtaining two Trs matrices of sizes m and n-m, respectively, and a total of 2n multiplications with the coefficients of C and S of the Givens matrix. In addition, to calculate the coefficients of C and S of the Givens matrix, it is necessary to calculate the norm ||X|| of the given vector X. Thus, for the volume C_R of computational operations when using block diagonal Transpositions matrices using formula (11), we can record the following estimate

C_R = 2.(1 norm + $n^2/4$ bitwise multiplications + $n^2/4$ changes of sign) +1 norm

and consequently

C_R = 3 norms+ $n^2/2$ bitwise multiplications + $n^2/2$ changes of sign (17)

A comparison of estimates of the volume of calculations using Householder Reflection and using block diagonal Transpositions matrices shows that the volume of calculations is approximately the same $C_R \approx C_H$. In some cases, when using a more efficient vector norm calculation algorithm, using block diagonal Transpositions matrices may be a better choice.

Theorem: V matrix, created using algorithm, described above, is matrix of rotation.

Proof: The V matrix is represented as a product of a Givens matrix and a Trb matrix $V=G_n(1,m+1,\theta).Trb$ (15). Since the Givens matrix is a rotation matrix, for the V matrix to be a matrix of rotation it suffices to prove that the Trb matrix is a matrix of rotation. For this purpose, it is necessary to prove that Trb is an orthogonal matrix (i.e. $Trb.Trb^T=I$, where I is identity matrix) and that det(Trb)=1.

First, for $Trb.Trb^T$= we get:

$$Trb.Trb^T = \begin{bmatrix} Trs(\overline{X}1) & 0 \\ 0 & Trs(\overline{X}2) \end{bmatrix}\begin{bmatrix} Trs(\overline{X}1)^T & 0 \\ 0 & Trs(\overline{X}2)^T \end{bmatrix}$$
$$= \begin{bmatrix} Trs(\overline{X}1).Trs(\overline{X}1)^T & 0 \\ 0 & Trs(\overline{X}2).Trs(\overline{X}2)^T \end{bmatrix}$$

whence, taking of consideration that Transpositions matrices are orthogonal matrices we get $Trb.Trb^T=I$.

Second, since the matrix Trb is a block diagonal matrix whose blocks are Transpositions matrices $Trs(\overline{X}1)$ and $Trs(\overline{X}2)$, then for the determinant of the matrix Trb we get $det(Trb) = det(Trs(\overline{X}1)).det(Trs(\overline{X}2))$, whence, taking into account that Transpositions matrices are matrices of reflection [1], i.e. $det(Trs(\overline{X}1)) = -1$ and $det(Trs(\overline{X}2)) = -1$, we get det(Trb)=1, Q.E.D.

The following example shows the code of a Matlab function that creates and returns orthogonal matrix which rotates a given vector X in the direction of the coordinate axis x_1 using block diagonal Transpositions (Trs) matrices.

```
function V = OrthogonalTrs(X1,X2)
 n1 = length(X1); n2 = length(X2);
 if n2 ~= 1 && n1 ~= 2 && n1 ~= 4 && n1 ~= 8
    error('Dimension of X1 have to be 1,2,4 or 8');
 end;
 if n2 ~= 1 && n2 ~= 2 && n2 ~= 4 && n2 ~= 8
    error('Dimension of X2 have to be 1,2,4 or 8');
 end;
 end;
 X=[X1;X2]; %Full input vector
 normX=norm(X); %Norms of vectors
 norm1=norm(X1); norm2=norm(X2);
 X1n = X1/ norm1;
 X2n = X2/ norm2;
 Trs1 = TrsMatrixs(X1n); %Transpositions matrices
 Trs2 = TrsMatrixs(X2n);
```

```
V = [Trs1 zeros(n1,n2); zeros(n2,n1) Trs2]; %Orthogonal Trs matrix
C=norm1/normX; S=norm2/normX; %Coeficients of Givens rotation
V(1,1:n1) = C* X1n';
V(n1+1,1:n1) = -S* X1n';
V(1, n1+1:n1+n2) = S* X2n';
V(n1+1, n1+1:n1+n2) = C* X2n';
```

Example 3 Matlab function, that creates orthogonal matrix, which rotates a given vector X in the direction of the coordinate axis x_1 (V matrix) using block diagonal Transpositions (Trs) matrices.

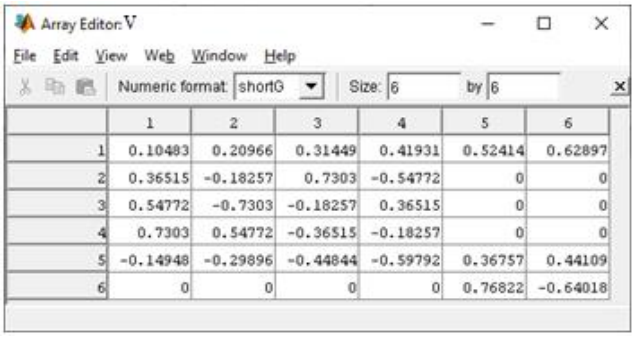
As it can be seen, function OrthogonalTrs(X1,X2) calculates norms of input vectors X1 and X2, obtain V matrix as block diagonal matrix, the blocks of which are Transpositions (Trs) matrices. The rows number *1* and *n1+1* of block diagonal matrix are replaced by the result of multiplication of vectors X1 and X2 by the coefficients C and S of Givens matrix, calculated as it is given in formula (16).

IV. NUMERICAL EXPERIMENTS

Executing the Matlab code below yields a V matrix using the OrthogonalTrs function. Obtained V matrix rotates given vector $X=(1,2,3,4,5,6)^T$ in the direction of coordinate axis x_1. The vector X is split into vectors $X1=(1,2,3,4)^T$ and $X=(5,6)^T$ of dimensions 4 and 2. When executing the following Matlab code

```
X=[1 2 3 4 5 6]';
X1=[1 2 3 4]'; X2=[5 6]';
V = OrthogonalTrs(X1,X2);
Y=V*X;
```

is obtained the following orthogonal V matrix:



	1	2	3	4	5	6
1	0.10483	0.20966	0.31449	0.41931	0.52414	0.62897
2	0.36515	-0.18257	0.7303	-0.54772	0	0
3	0.54772	-0.7303	-0.18257	0.36515	0	0
4	0.7303	0.54772	-0.36515	-0.18257	0	0
5	-0.14948	-0.29896	-0.44844	-0.59792	0.36757	0.44109
6	0	0	0	0	0.76822	-0.64018

Fig. 2 An orthogonal V matrix obtained by the OrthogonalTrs function. V matrix rotates the vector $X=(1,2,3,4,5,6)^T$ along the x_1 coordinate axis.

As a result of multiplying the obtained matrix V by the specified vector X, a vector Y is obtained, which is in the direction of the coordinate axis x_1.

V. CONCLUSION

The proposed algorithm for creating an N-dimensional orthogonal matrix V, which rotates a given vector X of dimension $N = m + n$, $m=2^p$, $n=2^q$, $p,q \in [2, 4, 8]$ in the direction of the x_1 coordinate axis using Transpositions matrices (Trs) allows to obtain similar performance compared to the computational efficiency of the method that uses Householder Reflection under the above vector dimension limits. The matrix, obtained by the proposed algorithm, can be useful in performing some operations in linear algebra such as QR decomposition of matrices.

REFERENCES

[1] O. I. Zhelezov, 2019, A Special Case of Symmetric Matrices and Their Applications, American Journal of Computational and Applied Mathematics , Vol. 9 No. 4, pp. 110-118. doi: 10.5923/j.ajcam.20190904.03.

[2] O. I. Zhelezov, 2021, An Algorithm for Generating N-Dimensional Rotation Matrix . Current Topics on Mathematics and Computer Science Vol. 6, pp. 15–28. https://doi.org/10.9734/bpi/ctmcs/v6/3301F

[3] H. G. Golub, J. M. Ortega, 1993, Scientific Computing and Introduction with Parallel Computing, Academic Press, Inc., San Diego.

[4] G. A. Korn, T. M. Korn, 1961, Mathematical Handbook for Scientists and Engineers (1st ed.), New York: McGraw-Hill. pp. 55–79.

[5] H. Friedberg, A. Insel, L. Spence, 1997, Linear Algebra (3rd ed), Prentice Hall.

[6] D. A. Harville, 1997, Matrix Algebra from Statistician's Perspective, Softcover.

[7] J. R. Silvester, 2000, Determinants of block matrices. The Mathematical Gazette, 84, 460 - 467..

[8] G. H. Golub, C. F. Van Loan, 1996, Matrix Computations, 4rd edition. Johns Horkins University Press, Baltimore

[9] M. Cosnard, Y. Robert. Complexity of parallel QR factorization. J. ACM 33, 4 (August 1986), 712-723. DOI=10.1145/6490.214102, 1986.

[10] A. Hedayat, W. D. Wallis, 1978, Hadamard matrices and their applications. Annals of Statistics. 6 (6): 1184–1238. doi:10.1214/aos/1176344370.

[11] G. Strang, 2006, Linear Algebra and its Applications, Thomson Learning Ink, pages 69-135, ISBN 0-03-010567.

[12] St. Roman, Advanced Linear Algebra, second ed., 2005 Springer-Verlag, New York. pages 59-85, ISBN: 978-1-4757-2180-5

[13] J. E. Gentle, 2007, Matrix Algebra: Theory, Computations, and Applications in Statistics, Springer, page 180, ISBN 978-0-387-70872-0

[14] [14] I. R. Shafarevich, A. Remizov, 2013, Linear Algebra and Geometry, Springer, pages 133-160, ISBN 978-3-642-30993-9
[15] [15] S. R. Davis C++ for dummies. IDG Books Worldwide, 2000.
[16] [16] Scott, N. R. , Computer Number Systems and Arithmetic, Prentice Aall, Englewood Cliffs, N.J. 1985
[17] Victor P. Nelson, H. Troy Nagle, Bill D. Carroll, and J. David Irwin, Digital Logic Circuit Analysis and Design, Prentice Hall, Inc., 1995
[18] Gentle, J. E. "QR Factorization." §3.2.2 in Numerical Linear Algebra for Applications in Statistics. Berlin:Springer-Verlag, pp. 95-97, 1998.
[19] https://www.agner.org/optimize/instruction_tables.pdf Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD