Two-dimensional fast Fourier transform

Batterfly in analog of Cooley-Tukey algorithm

V.S. Tutatchikov
Institute of Space and Information Technology
Siberian Federal University
Krasnoyarsk, Russia
vtutatchikov@mail.ru

One- and two-dimensional (2D) fast Fourier transform (FFT) algorithms has been widely used in digital processing. 2D discrete Fourier transform is reduced to a combination of one-dimensional FFT for all coordinates due to the increased complexity and the large amount of computation by increasing dimension of the signal. This article provides the butterfly of analog Cooley-Tukey algorithm, which requires less complex operations of additional and multiplication than the standard method, and runs 1.5 times faster than analogue in Matlab.

Keywords—fast Fourier transform; butterfly of Cooley-Tukey algorithm

I. INTRODUCTION

One- and two-dimensional (2D) fast Fourier transform (FFT) algorithms has been widely used in digital processing [1]. 2D discrete Fourier transform is reduced to a combination of one-dimensional FFT for all coordinates [2] due to the increased complexity and the large amount of computation by increasing dimension of the signal. This article provides the butterfly of a general Cooley-Tukey algorithm analog, which requires less complex operations of additional and multiplication than the standard method [3]. Shown results testing of the resulting algorithm in two-dimensions in comparison with the standard algorithm in Matlab [4].

II. 1D ALGORITHM DESCRIPTION

Let us have a look at the signal f, which is an n-dimensional periodic signal with a period $N = 2^S$ of coordinate with values in a complex space. The counts are given as $f(x)$, where x take values $0,1,...,2^S-1$. The discrete Fourier transformation (DFT) $F(y)$ for the signal $f(x)$ is given in the formula:

$$F(y)=\sum_{x=0}^{2^s-1} f(x)\cdot e^{\frac{2\pi ixy}{2^s}} \tag{1}$$

where y take values $0,...,2^s-1$.

In general transform (1) by dividing coordinate x into even and odd components:

$$\begin{aligned} F(y) &= \sum_{x=0}^{2^s-1} f(x)\cdot e^{\frac{2\pi ixy}{2^s}} = \sum_{x_1=0}^{2^{s-1}-1} f(2x_1)\cdot e^{\frac{2\pi i 2x_1 y}{2^s}} + \\ &+ \sum_{x_1=0}^{2^{s-1}-1} f(2x_1+1)\cdot e^{\frac{2\pi i(2x_1+1)y}{2^s}} = \\ &= \sum_{x_1=0}^{2^{s-1}-1} f(2x_1)\cdot e^{\frac{2\pi i x_1 y}{2^{s-1}}} + \sum_{x_1=0}^{2^{s-1}-1} f(2x_1+1)\cdot e^{\frac{2\pi i x_1 y}{2^{s-1}}} e^{\frac{2\pi iy}{2^s}} = \\ &= g_0(x_1)+g_1(x_1)e^{\frac{2\pi iy}{2^s}} = g_0(x_1)+g_1(x_1)W_s^y \end{aligned} \tag{2}$$

where $W_s^y = e^{\frac{2\pi iy}{2^s}}$, $g_i(x_1)=\sum_{x_1=0}^{2^{s-1}-1} f(2x_1+i)\cdot e^{\frac{2\pi i x_1 y}{2^{s-1}}}$, $i=0,1$ - 1D Fourier transform with 2^{s-1} elements of subsignals f that contain the components with even and odd indexes, respectively.

One can show that multiplier $e^{\frac{2\pi iy}{2^s}}$ in (3) possesses the property of symmetry with respect to 2^{s-1}, i.e. for $y_1 = 0:2^{s-1}-1$ and y_1+2^{s-1} we have:

$$\begin{aligned} & e^{\frac{2\pi i(y_1+2^{s-1})}{2^s}} = e^{\frac{2\pi iy_1}{2^s}} e^{\frac{2\pi i 2^{s-1}}{2^s}} = e^{\frac{2\pi iy_1}{2^s}} e^{\pi i} = -e^{\frac{2\pi iy_1}{2^s}}, \\ & W_s^{y_1+2^{s-1}} = W_s^{y_1} W_s^{2^{s-1}} = W_s^{y_1} W_1^1 = -W_s^{y_1} \end{aligned} \tag{3}$$

Then formula (2) with regard to (3) takes the form:

$$\begin{aligned} F(y_1) &= g_0(x_1)+g_1(x_1)W_s^{y_1} \\ F(y_1+2^{s-1}) &= g_0(x_1)-g_1(x_1)W_s^{y_1} \end{aligned} \tag{4}$$

Equation (4) can be visualized as a “butterfly” in figure 1 [5].

Let us to calculate the number of operations. Butterfly (4) takes one complex multiplication and two complex operations of addition. This expansion procedure can be repeated as long as there will be only 2 elements. In general, when period signal f is $N=2^s$, then require $N/2 \log_2 N$ complex multiplications and $N\log_2 N$ additions.

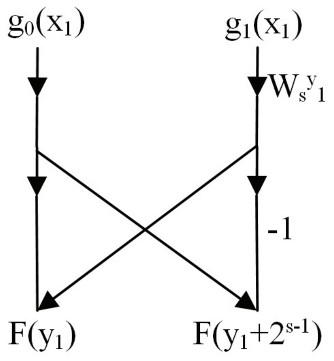


Fig. 1. butterfly of Cooley-Tukey algorithm

III. 2D ALGORITHM DESCRIPTION

The standard way to calculate the two-dimensional FFT - a consistent application of the one-dimensional FFT first for each column, then each row. In this case, the number of operations required. In [4] described two-dimensional Fourier transform butterfly on analog Cooley-Tukey algorithm. Let's consider this scheme.

Let us have a look at the signal f, which is an two-dimensional periodic signal with a period 2^s of both coordinates with values in a complex space. The counts are given as $f(k,t)$, where k,t take values $0,1,...,2^s-1$. The discrete Fourier transformation (DFT) $F(p,m)$ for the signal $f(k,t)$ is given in the formula:

$$F(p,m)=\sum_{k=0}^{2^s-1}\sum_{t=0}^{2^s-1} f(k,t)\cdot e^{\frac{2\pi i(kp+tm)}{2^s}} \tag{5}$$

where p,m take values $0,...,2^s-1$.

2D DFT can be computed by a combination of one-dimensional FFT. To this end, one computes F in the following form:

$$\begin{aligned} F(p,m)&=\sum_{k=0}^{2^s-1}\left[\sum_{t=0}^{2^s-1} f(k,t)e^{\frac{2\pi itm}{2^s}}\right]\cdot e^{\frac{2\pi ikp}{2^s}} = \\ &=\sum_{k=0}^{2^s-1} G(k,m)\cdot e^{\frac{2\pi ikp}{2^s}} \end{aligned} \tag{6}$$

The sum in square brackets in (6) represents the computation of a one-dimensional DFT, for example, by rows; then the outer sum represents the computation of a one-dimensional DFT by columns (figure 1). To calculate the butterfly in Figure 1 will require 4 complex multiplications and 8 complex additions. In general, if the dimension of the signal f is $N\times N, N=2^s$ samples, then require $N^2\log_2 N$ complex multiplications and $2N^2\log_2 N$ additions.

Thus after calculating the rows of one-dimensional FFTs require additional time to transpose matrix and one-dimensional FFT computation has columns, or need to change the algorithm for data columns, which is also time-consuming. We modify the formula (5) for calculating the two-dimensional FFT by analog of Cooley-Tukey algorithm.

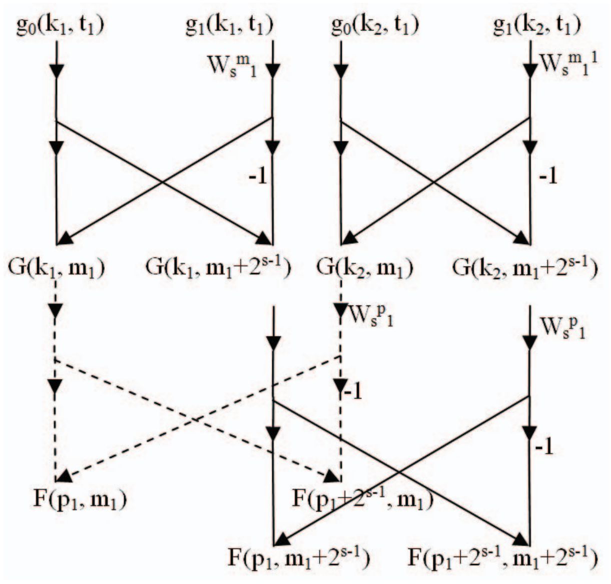


Fig. 2. standard 2D FFT algorithm

Let us transform this formula by dividing both coordinates into even and odd components:

$$\begin{aligned} F(p,m)&=\sum_{k=0}^{2^s-1}\sum_{t=0}^{2^s-1} f(k,t)\cdot e^{\frac{2\pi ikp}{2^s}} e^{\frac{2\pi itm}{2^s}} = \\ &=\sum_{k_1=0}^{2^{s-1}-1}\sum_{t_1=0}^{2^{s-1}-1} f(2k_1,2t_1)\cdot e^{\frac{2\pi i2k_1p}{2^s}} e^{\frac{2\pi i2t_1m}{2^s}} + \\ &+\sum_{k_1=0}^{2^{s-1}-1}\sum_{x_2^1=0}^{2^{s-1}-1} f(2k_1+1,2t_1)\cdot e^{\frac{2\pi i(2k_1+1)p}{2^s}} e^{\frac{2\pi i2t_1m}{2^s}} + \\ &+\sum_{k_1=0}^{2^{s-1}-1}\sum_{t_1=0}^{2^{s-1}-1} f(2k_1,2t_1+1)\cdot e^{\frac{2\pi i2k_1p}{2^s}} e^{\frac{2\pi i(2t_1+1)m}{2^s}} + \\ &+\sum_{k_1=0}^{2^{s-1}-1}\sum_{t_1=0}^{2^{s-1}-1} f(2k_1+1,2t_1+1)\cdot e^{\frac{2\pi i(2k_1+1)p}{2^s}} e^{\frac{2\pi i(2t_1+1)m}{2^s}} = \\ &= g_{00}(k_1,t_1)+g_{10}(k_1,t_1)W_s^p+g_{10}(k_1,t_1)W_s^m + \\ &+ g_{11}(k_1,t_1)W_s^pW_s^m \end{aligned} \tag{7}$$

where $g_{ij}(k_1,t_1)=\sum_{k_1=0}^{2^{s-1}-1}\sum_{t_1=0}^{2^{s-1}-1} f(2k_1+i,2t_1+j)\cdot e^{\frac{2\pi ik_1p}{2^{s-1}}} e^{\frac{2\pi it_1m}{2^{s-1}}}$,

$i,j=0,1$ - 2D Fourier transform with $2^{s-1}\times 2^{s-1}$ elements of subsignals f that contain the all components with even and odd indexes, respectively, $W_s^p = e^{\frac{2\pi ip}{2^s}}$.

Then formula (7) with regard to (3) takes the form:

$$\begin{aligned}
&F(p_1,m_1) = g_{00}(k_1,t_1) + g_{10}(k_1,t_1)W_s^{p_1} + \\
&+ g_{01}(k_1,t_1)W_s^{m_1} + g_{11}(k_1,t_1)W_s^{p_1+m_1}, \\
&F(p_1+2^{s-1},m_1) = g_{00}(k_1,t_1) - g_{10}(k_1,t_1)W_s^{p_1} + \\
&+ g_{01}(k_1,t_1)W_s^{m_1} - g_{11}(k_1,t_1)W_s^{p_1+m_1}, \\
&F(p_1,m_1+2^{s-1}) = g_{00}(k_1,t_1) + g_{10}(k_1,t_1)W_s^{p_1} - \\
&- g_{01}(k_1,t_1)W_s^{m_1} - g_{11}(k_1,t_1)W_s^{p_1+m_1}, \\
&F(p_1+2^{s-1},m_1+2^{s-1}) = g_{00}(k_1,t_1) - \\
&- g_{10}(k_1,t_1)W_s^{p_1} - g_{01}(k_1,t_1)W_s^{m_1} + \\
&+ g_{11}(k_1,t_1)W_s^{p_1+m_1}.
\end{aligned} \tag{8}$$

where the expression $W_s^{p_1+m_1} = W_s^{p_1}W_s^{m_1}$ allows you to really save one complex multiplication in the last term in each expression (8), and we instead of multiplying of two exponents multiplication put one with the sum of indicators.

Simplified expression (8) can be represented in figures 3-4, the generic version of the figure 5.

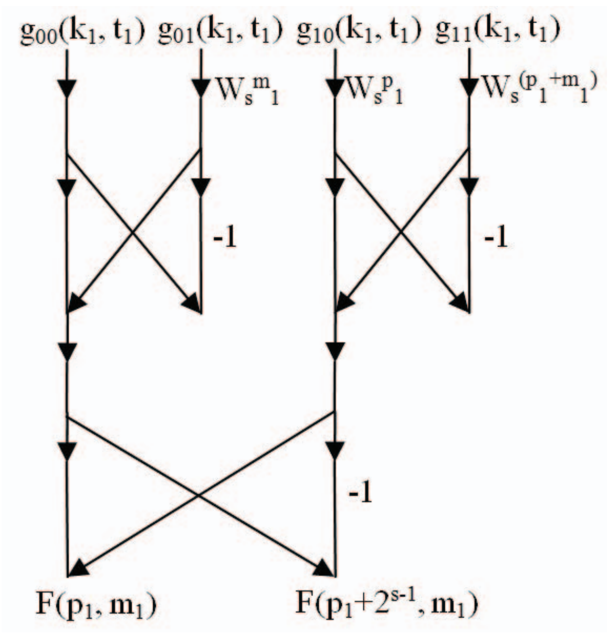
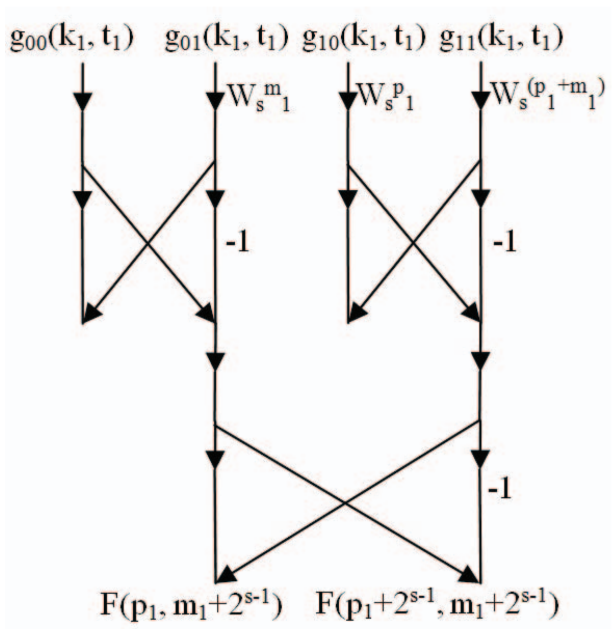


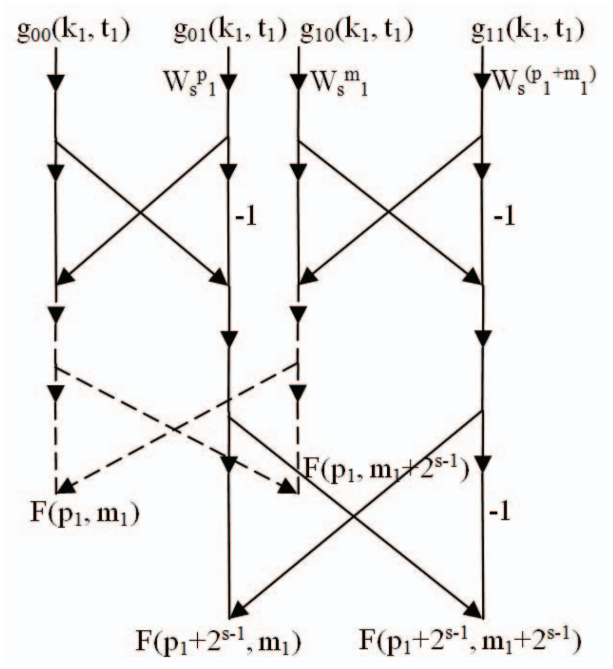
Fig. 3. 2D butterfly of analog Cooley-Tukey algorithm part 1

Let us to calculate the number of operations. Butterfly (8) takes three complex multiplication and eight complex operations of addition. This expansion procedure can be repeated as long as there will be only 2×2 elements. In general, if period signal f is $N\times N, N=2^S$, then require $\frac{3}{4}N^2\log_2 N$ complex multiplications and $2N^2\log_2 N$ additions [5].



$g_{00}(k_1, t_1)$ $g_{01}(k_1, t_1)$ $g_{10}(k_1, t_1)$ $g_{11}(k_1, t_1)$
$W_s^{m_1}$
$W_s^{p_1}$
$W_s^{(p_1+m_1)}$
-1
-1
-1
$F(p_1, m_1+2^{s-1})$
$F(p_1+2^{s-1}, m_1+2^{s-1})$

Fig. 4. 2D butterfly of analog Cooley-Tukey algorithm part 2



$g_{00}(k_1, t_1)$ $g_{01}(k_1, t_1)$ $g_{10}(k_1, t_1)$ $g_{11}(k_1, t_1)$
$W_s^{p_1}$
$W_s^{m_1}$
$W_s^{(p_1+m_1)}$
-1
-1
$F(p_1, m_1+2^{s-1})$
$F(p_1, m_1)$
-1
$F(p_1+2^{s-1}, m_1)$
$F(p_1+2^{s-1}, m_1+2^{s-1})$

Fig. 5. 2D butterfly of analog Cooley-Tukey algorithm

IV. OBTAINED RESULT

Two FFT algorithm in two dimensions is shown in Table 1, wherein the signal f comprises $N\times N, N=2^S$ samples. In this table, we can see that the analogue of Cooley-Tukey algorithm requires fewer complex multiplications than the standard method calculating by the row and column, and the number of complex additions identical in both cases. But standard method of calculation would require additional time to transpose the matrix in the transition from one-dimensional FFT computation by lines to the calculation of the columns.

TABLE I. COMPARING THE NUMBER OF OPERATIONS IN 2D

Complex operation	Standard 2D FFT by rows and columns	2D FFT by analog of Cooley-Tukey algorithm
Multiplication	$N^2 \log_2 N$	$\frac{3}{4} N^2 \log_2 N$
Addition	$2N^2 \log_2 N$	$2N^2 \log_2 N$

For the algorithm testing program in the programming language C++ has been written for two-dimensional signal. The testing was conducted on PC with following characteristics:

- Processor: AMD FX-4170 4.2 GHz;
- RAM: 8 GB;
- Operating system: Windows 7.

It was compared to a standard method of calculating two-dimensional discrete Fourier transform in environment of Matlab 7.5.0 (R2007b). Table 2 shows a comparison runtime in seconds: in the first column indicates the size of signal f in pixels, the second column shows the time of calculating the 2D FFT in Matlab, the third – standard method for computing 2d FFT in C++, the last – analog of Cooley-Tukey algorithm in two-dimensional case.

TABLE II. CALCULATING 2D FFT

Size signal	2D FFT by Matlab	2D FFT by rows and columns	2D FFT by analog of Cooley-Tukey algorithm
128*128	0.001	0.001	0.001
256*256	0.005	0.006	0.004
512*512	0.027	0.027	0.017
1024*1024	0.125	0.136	0.087
2048*2048	0.620	0.622	0.389
4096*4096	2.634	2.691	1.637
8192*8192	13.609	11.736	6.904
16384*16384	-	34.561	20.383

In [7] describes a method for calculating the two-dimensional FFT for a rectangular signal with $2^s \times 2^v, s, v \in \mathrm{N}$ samples, based on 2D analog of Cooley-Tukey algorithm. In [8] described extension of this algorithm to the three-dimensional case, in [3] extension to the multidimensional case, and in [9] - parallelization in the general case.

V. CONCLUSIONS

Two-dimensional fast Fourier transform by analog of Cooley-Tukey algorithm requires less complex operations of multiplication than the standard method calculating by rows and columns, and runs 1.5 times faster than analogue in Matlab.

ACKNOWLEDGMENT

Work performed under the state order of the Ministry of Education and Science if the Russian Federation in the Siberian Federal University to perform R&D in 2014 (Task No 1.1462.2014/K). Project title: "Algebraic and analytic methods for creating algorithms for solving differential and polynomial systems: factorization, resolution of singularities and the optimal lattice".

REFERENCES

[1] D.E. Dudgeon, R.M. Mersereau, "Multidimensional Digital Signal Processing", Prentice Hall, 1983.

[2] R.E. Blahut, "Fast Algorithms for Digital Signal Processing", Addison-Wesley Press, 1985.

[3] V.S. Tutatchikov, O.I. Kiselev, M.V. Noskov, "Calculating the n-Dimensional Fast Fourier Transform", Pattern Recognition and Image Analysis, 2013, vol. 23, no. 3, pp. 429-433.

[4] R.C. Gonzalez, R.E. Woods, S.L. Eddins, "Digital Image Processing Using MATLAB", Gatesmark Publishing. Knoxville, 2009.

[5] A.V. Oppenheim, R.W. Schafer, "Digital signal processing", Prentice Hall, Englewood Cliffs, New Jersey, 1975, pp. 206-220.

[6] A.V. Starovoitov, "On multidimensional analog of Cooley-Tukey algorithm", Reporter Siberian State Aerospace University named after academician M.F.Reshetnev, No. 1 (27), 2012, pp. 69-73.

[7] M.V. Noskov, V.S. Tutatchikov,"Modification of a Two-Dimensional Fast Fourier Transform by the Analog of thr Cooley-Tukey Algorithm for a Rectangular Signal", Pattern Recognition and Image Analysis, 2015, Vol. 25, No. 1, pp. 81-83.

[8] M.V. Noskov, V.S. Tutatchikov, I.V. Kolcova, "Three –dimensional fast Fourier transform algorithm modification by analogue of Cooley-Tukey", Pattern Recognition and Information Processing (PRIP'2014): Processing of the 12th International Conference (28-30 May 2014, Minsk, Belarus). – Minsk: UIIP NASB, 2014, pp. 223-226,

[9] M.V. Noskov, V.S. Tutatchikov, "Parallel Version n-Dimensional Fast Fourier Transform Algorithm Analog of the Cooley-Tukey Algorithm", Proceeding of the 5TH International Workshop on Image Mining. Theory and Applications, Berlin, Germany, 2015, pp. 114-117.