Efficient algorithm to compute a DFT for an arbitrary set of frequencies

Thomas.Rode@gmx.de
11/12/2009

When computing a DFT, some applications require the transformation for only a limited set of frequencies. FFT algorithms are designed for the calculation of whole spectra. Therefore, if the number of target frequencies is relatively small, using such algorithms would not be optimal. In such cases special algorithms, the most popular being “Goertzels Algorithm”, are employed. In this abstract I would like to suggest the use of another group of algorithms that can perform such partial DFT very efficiently. The computational cost per frequency depends on the target frequency itself. For real value input data (the most common case) the maximum is N/2 real value additions and N/2 – 2 real value multiplications. All computation, except for the final output, can be done in-situ. Additionally the algorithm is reversible, supporting the reverse-DFT of an arbitrary set of frequencies to time-domain.

Basic principle:

Similar to a radix-2-FFT, the algorithm requires input data as a set of $N = 2^X$ elements. In a first step, data is split into a maximum of ld(N) sub-sets the size N/2, N/4, … 2 by successive addition/subtraction. The mechanism is identical to the first steps of a decimation-in-frequency-FFT, before applying the twiddle factors. The sub-sets represent the frequencies 1, 2, 4, 8, … N/2 as well as their odd harmonics (e.g. 1, 3, 5, … / 2, 6, 10, … / 4, 12, 20, …). The algorithm should be programmed dynamically, so that only sub-sets containing target frequencies are calculated. Following an example:

If the input data consists of $2^4 = 16$ real elements, the frequency-domain contains 8 elements, distributed in the sub-sets:

Set 1: 1, 3, 5, 7
Set 2: 2, 6
Set 3: 4
Set 4: 8

If only the frequencies 1, 3, 6, 7 are of interest, calculation of set 3 and set 4 is not needed. For further illustration the below chart describes the splitting in detail.

Input	First split	Second split	Third split	
X0	X0-X8	Frequencies 1, 3, 5, 7		
X1	X1-X9			
X2	X2-X10			
X3	X3-X11			
X4	X4-X12			
X5	X5-X13			
X6	X6-X14			
X7	X7-X15			
X8	X0+X8	(X0+X8)-(X4+X12)	Frequencies 2, 6	
X9	X1+X9	(X1+X9)-(X5+X13)		
X10	X2+X10	(X2+X10)-(X6+X14)		
X11	X3+X11	(X3+X11)-(X7+X15)		
X12	X4+X12	(X0+X8)+(X4+X12)	((X0+X8)+(X4+X12))-((X2+X10)+(X6+X14))	Frequency 4
X13	X5+X13	(X1+X9)+(X5+X13)	((X1+X9)+(X5+X13))-((X3+X11)+(X7+X15))	
X14	X6+X14	(X2+X10)+(X6+X14)	((X0+X8)+(X4+X12))+((X2+X10)+(X6+X14))	Frequency 8 (0)
X15	X7+X15	(X3+X11)+(X7+X15)	((X1+X9)+(X5+X13))+((X3+X11)+(X7+X15))	

Chart 1: Splitting of time-domain data by frequency

As a second step, the splitting is followed by a DFT. In contrast to other methods, the algorithm is set up to only calculate DFT of the base frequencies of the sub-sets. To keep the above example, “base frequencies” would be 1 and 2, since sub-set 3 and 4 are not needed. In order to get the DFT of the harmonics 3, 6, 7, an additional frequency-shifting is required prior to the actual DFT. Therefore the exact sequence would have to be like that:

1.) Data input

2.) Calculation of sub-sets (e.g. Set 1 and Set 2)

3.) DFT of required "base frequencies" (e.g. 1)

4.) Frequency-shifting of first sub-set to do DFT of required harmonics
(e.g. shift 3 to 1 and DFT 3, shift 7 to 1 and DFT 7)

5.) Frequency-shifting of second sub-set to do DFT of required harmonics
(e.g. shift 6 to 2 and DFT 6)

Frequency-shifting in this case simply means expanding the step width between to consecutive elements of the time-domain data of a sub-set by a certain (odd) factor. Because of the limited resolution in time, all frequencies shifted out of the base band are mirrored back (aliasing effect). If started at the first data element of the sub-set, the shifting affects the frequency but not the phase of the signal. Here some examples:

Input	First split (frequencies **1**, 3, 5, 7)	Frequency-shifting by 5 (freq. **3->1**, 7->3, 1->5, 5->7)
X0	Y0 = X0-X8	Y0
X1	Y1 = X1-X9	Y5
X2	Y2 = X2-X10	-Y2
X3	Y3 = X3-X11	-Y7
X4	Y4 = X4-X12	Y4
X5	Y5 = X5-X13	-Y1
X6	Y6 = X6-X14	-Y6
X7	Y7 = X7-X15	Y3
X8		
X9		
X10		
X11		
X12		
X13		
X14		
X15		

Chart 2: Frequency-shifting by a factor of 5

In chart 2 the frequencies of the sub-set are shifted by 5, with frequency 3 taking the place of 1. Hence, the same DFT can be used for both frequencies (DFT to 1 -> shift with 5x -> DFT to 3). In this way all frequencies of interest can be sequentially “pushed” to 1 and transformed. The gain of this approach lies in the structure of the DFT, which becomes static (the same DFT-sequence with the same parameter is being used multiple times). That is very beneficial when designing hardware-based DFT but also gives significant savings in computation time at software solutions.

Instead of only shifting the target frequencies, it is also possible to progressively rotate the whole sub-set by continuously shifting with the same factor (e.g. 3 or 5). After N/2 – 1 such steps, all frequencies have “passed” 1 (N in this case being the length of the sub-set). Chart 3 illustrates the idea. This approach might be favourable for a hardware-based shifter, because the complexity of the circuit is being reduced even further.

Input	First split (frequencies **1**, 3, 5, 7)	Frequency-shifting by 3 (freq. **5->1**, 1->3, 7->5, 3->7)	Frequency-shifting by 3 (freq. **7->1**, 5->3, 3->5, 1->7)	Frequency-shifting by 3 (freq. **3->1**, 7->3, 1->5, 5->7)
X0	Y0 = X0-X8	Y0	Y0	Y0
X1	Y1 = X1-X9	Y3	-Y1	-Y3
X2	Y2 = X2-X10	Y6	Y2	Y6
X3	Y3 = X3-X11	-Y1	-Y3	Y1
X4	Y4 = X4-X12	-Y4	Y4	-Y4
X5	Y5 = X5-X13	-Y7	-Y5	Y7
X6	Y6 = X6-X14	Y2	Y6	Y2
X7	Y7 = X7-X15	Y5	-Y7	-Y5
X8				
X9				
X10				
X11				
X12				
X13				
X14				
X15				

Chart 3: Rotating frequencies by continuously shifting with 3x

Depending on the available resources and the preferred design (hardware / software), shifting can be done in many different ways and at different points within the algorithm. Following some possible scenarios:

1.) **Shifting of the actual content of the memory**

This can be done in-situ (see chart 4). Due to the odd frequency ratio, the structure of the algorithm is relatively complex. In addition many read / write operations are required. Therefore this approach seems most suitable for a hardware-based shifter, especially if the shifting factor is constant (as in chart 3). In this case only one array of memory the size of the sub-set is required. Output and input of the memory are connected in a way representing the desired shifting factor. With each load cycle the content is then shifted by that factor and ready for DFT.

2.) **Shifting of the memory pointer**

If memory is accessed indirectly via a pointer, shifting that pointer would be much faster and efficient that shifting the content of memory itself.

3.) **Shifting the DFT pointer**

Rather than manipulating the content of the memory, another approach is to change the pointer defining the access of the DFT algorithm to that data. To give an example, instead of computing X0, X1, X2, … with a 16-value-DFT for frequency 1, data can be loaded into the same DFT with a shifting factor of 5 (X0, X5, X10, …) to calculate frequency 3. Since the allocation of data between the memory and the DFT sequence has to be done anyway, this approach would require only minimal extra resources and be optimal for a software-based implementation. Later chart 5 provides an example.

Input	First split (frequencies 1, 3, 5, 7)	Shifting by 3 (in-situ)
X0	X0-X8	X0-X8
X1	X1-X9	X3-X11
X2	X2-X10	X6-X14
X3	X3-X11	-X1+X9
X4	X4-X12	-X4+X12
X5	X5-X13	-X7+X15
X6	X6-X14	X2-X10
X7	X7-X15	X5-X13
X8		
X9		
X10		
X11		
X12		
X13		
X14		
X15		

Chart 4: Principle of in-situ shifting

Additional optimisations:

Since the algorithm is basically just a disentangled radix-based FFT, even more computation time can and should be saved by exploiting the symmetries between the various frequencies of each sub-set. It is those symmetries that allow hierarchical computation of a DFT and give FFT its speed. Looking at the various options it becomes obvious, that only some symmetries can be successfully used when computing arbitrary frequencies. For most the computational cost of controlling the algorithm (decision points, …) would far exceed the benefit of saving some calculations. With respect to each sub-set, efficient implementation is possible for the symmetries at $(f_a+f_b)/2$, in particular $f_{max}/2$. The practical implication is that data points at N/4, N/2, 3N/4 (N being the length of the sub-set) have to be calculated only once (do not change when sub-set is frequency-shifted) and that frequencies symmetric to $f_{max}/2$ (e.g.1 and 7 or 3 and 5 at our example) give the same results in multiplication and hence need to be calculated only once.

Examples:

Step	1	2	3	4	5		6		7		8	
Comment	2^4 elements	Check list of target frequencies to decide which sub-set has to be calculated			Check list of target frequencies to decide which separation has to be calculated							
Element	Real value input	First split (freq. 1,3,5,7 and 2,4,6,8)	Second split (freq. 2,6 and 4,8)	Third split (freq. 4 and 8)	Separation Real / Imag. for freq. 1, 3, 5, 7		Separation Real / Imag. for freq. 2, 6		Separation Real / Imag. for freq. 4		Separation Real / Imag. for freq. 8 (0)	
					Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	X0	$Y1_0=X0-X8$			$Y7_0=Y1_0$							
1	X1	$Y1_1=X1-X9$			$Y7_1=Y1_1-Y1_7$							
2	X2	$Y1_2=X2-X10$			$Y7_2=Y1_2-Y1_6$							
3	X3	$Y1_3=X3-X11$			$Y7_3=Y1_3-Y1_5$							
4	X4	$Y1_4=X4-X12$				$Y7_4=Y1_4$						
5	X5	$Y1_5=X5-X13$				$Y7_5=Y1_3+Y1_5$						
6	X6	$Y1_6=X6-X14$				$Y7_6=Y1_2+Y1_6$						
7	X7	$Y1_7=X7-X15$				$Y7_7=Y1_1+Y1_7$						
8	X8	$Y2_0=X0+X8$	$Y3_0=Y2_0-Y2_4$				$Y8_0=Y3_0$					
9	X9	$Y2_1=X1+X9$	$Y3_1=Y2_1-Y2_5$				$Y8_1=Y3_1-Y3_3$					
10	X10	$Y2_2=X2+X10$	$Y3_2=Y2_2-Y2_6$					$Y8_2=Y3_2$				
11	X11	$Y2_3=X3+X11$	$Y3_3=Y2_3-Y2_7$					$Y8_3=Y3_1+Y3_3$				
12	X12	$Y2_4=X4+X12$	$Y4_0=Y2_0+Y2_4$	$Y5_0=Y4_0-Y4_2$					$Y9_0=Y5_0$			
13	X13	$Y2_5=X5+X13$	$Y4_1=Y2_1+Y2_5$	$Y5_1=Y4_1-Y4_3$						$Y9_1=Y5_1$		
14	X14	$Y2_6=X6+X14$	$Y4_2=Y2_2+Y2_6$	$Y6_0=Y4_0+Y4_2$							$Y10_0=Y6_0-Y6_1$	
15	X15	$Y2_7=X7+X15$	$Y4_3=Y2_3+Y2_7$	$Y6_1=Y4_1+Y4_3$								

In-situ calculation ends at step 8.

Step	9		10		11		12		13	
Comment	Check list of target frequencies to decide which frequency has to be calculated									
Element	Multiplication with f = 1 (freq. 1, 7)		Mult. with f = 1, input shifted by 5 (freq. 3, 5)		Multiplication with f = 2 (freq. 2, 6)		Multiplication with f = 4 (freq. 4)		Multiplication with f = 8 (freq. 8)	
	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	$P1_0=Y7_0$		$P3_0=Y7_0$							
1	$P1_1=Y7_1*Z1_4$		$P3_1=-Y7_3*Z1_4$							
2	$P1_2=Y7_2*Z1_3$		$P3_2=-P1_2$							
3	$P1_3=Y7_3*Z1_2$		$P3_3=Y7_1*Z1_2$							
4		$P2_0=Y7_4$		$P4_0=-Y7_4$						
5		$P2_1=Y7_5*Z1_4$		$P4_1=Y7_7*Z1_4$						
6		$P2_2=Y7_6*Z1_3$		$P4_2=P2_2$						
7		$P2_3=Y7_7*Z1_2$		$P4_3=-Y7_5*Z1_2$						
8					$P5_0=Y8_0$					
9					$P5_1=Y8_1*Z1_3$					
10						$P6_0=Y8_2$				
11						$P6_1=Y8_3*Z1_3$				
12							$P7_0=Y9_0$			
13								$P8_0=Y9_1$		
14									$P9_0=Y10_0$	
15										

Sin-table	
Phase angle	sin x
=0*pi/8	$Z1_1$
=1*pi/8	$Z1_2$
=2*pi/8	$Z1_3$
=3*pi/8	$Z1_4$

	N/2 - 2 = 6 real mult.			N/2 - 4 = 4 real mult.			N/4 - 2 = 2 real mult.			N/8 - 2 = 0 real			
f_1 →	$=P1_0+P1_1+P1_2+P1_3$	$=P2_0+P2_1+P2_2+P2_3$	f_3 →	$=P3_0+P3_1+P3_2+P3_3$	$=P4_0+P4_1+P4_2+P4_3$	f_2 →	$=P5_0+P5_1$	$=P6_0+P6_1$	f_4 →	$=P7_0$	$=P8_0$	f_8 →	$=P9_0$
f_7 →	$=P1_0-P1_1+P1_2-P1_3$	$=-P2_0+P2_1-P2_2+P2_3$	f_5 →	$=P3_0-P3_1+P3_2-P3_3$	$=-P2_0+P2_1-P2_2+P2_3$	f_6 →	$=P5_0-P5_1$	$=-P6_0+P6_1$					

Chart 5: Real value DFT for N=16, shifting the pointer of the DFT and exploiting some symmetries in data (Gray steps do not need to be calculated)

Step	1	2	3	4	5		6		7		8	
Comment	2^4 elements	Check list of target frequencies to decide which sub-set has to be calculated			Check list of target frequencies to decide which separation has to be calculated							
Element	Real value input	First split (freq. 1,3,5,7 and 2,4,6,8)	Second split (freq. 2,6 and 4,8)	Third split (freq. 4 and 8)	Separation Real / Imag. for freq. 1, 3, 5, 7		Separation Real / Imag. for freq. 2, 6		Separation Real / Imag. for freq. 4		Separation Real / Imag. for freq. 8	
					Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	-0,65	-1,19			-1,19							
1	0,95	1,66			2,30							
2	-0,32	-0,68			-1,07							
3	0,51	1,12			1,32							
4	0,64	0,20				0,20						
5	-0,27	-0,20				0,92						
6	0,31	0,39				-0,29						
7	0,27	-0,64				1,02						
8	0,54	-0,11	-1,19				-1,19					
9	-0,71	0,24	0,58				1,86					
10	0,36	0,04	-0,19					-0,19				
11	-0,61	-0,10	-1,28					-0,70				
12	0,44	1,08	0,97	0,70					0,70			
13	-0,07	-0,34	-0,10	-1,18						-1,18		
14	-0,08	0,23	0,27	1,24							0,26	
15	0,91	1,18	1,08	0,98								

In-situ calculation ends at step 8.

Step	9		10		11		12		13	
Comment	Check list of target frequencies to decide which frequency has to be calculated									
Element	Multiplication with f = 1 (freq. 1, 7)		Mult. with f = 1, input shifted by 5 (freq. 3, 5)		Multiplication with f = 2 (freq. 2, 6)		Multiplication with f = 4 (freq. 4)		Multiplication with f = 8 (freq. 8)	
	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.	Real	Imag.
0	-1,19		-1,19							
1	2,12		-1,22							
2	-0,76		0,76							
3	0,51		0,88							
4		0,20		-0,20						
5		0,85		0,94						
6		-0,21		-0,21						
7		0,39		-0,35						
8					-1,19					
9					1,32					
10						-0,19				
11						-0,49				
12							0,70			
13								-1,18		
14									0,26	
15										

	N/2 - 2 = 6 real mult.			N/2 - 4 = 4 real mult.			N/4 - 2 = 2 real mult.			N/8 - 2 = 0 real mult.			
f_1 →	0,68	1,24	f_3 →	-0,77	0,19	f_2 →	0,13	-0,68	f_4 →	0,70	-1,18	f_8 →	0,26
f_7 →	-4,58	1,25	f_5 →	-0,09	1,00	f_6 →	-2,51	-0,30					

Sin-table	
Phase angle	sin x
0,00	0,00
0,39	0,38
0,79	0,71
1,18	0,92

Chart 6: Algorithm from Chart 5 with noise signal as input