

# CIRCULANTS (Extract)

Alun Wyn-jones

Last revised in January 2008.

Please copy this book for your own reading only. Refers others to this website. Thank You.

APPENDIX B  
The Cooley-Tukey Fast Fourier Transform

**B.1 Fast Fourier Transforms.** This appendix is brief introduction to methods which have been developed in the last 50 years for the efficient computations of Fourier transforms. Since a Fourier transform is just multiplication by the Fourier matrix, all these techniques apply to the computation of circulant eigenvalues.

The entire class of techniques are called Fast Fourier Transforms or FFT for short. Their importance and main application is in calculating frequency distributions from time series. Their commercial applications range from pattern recognition, to analysis of earthquakes, electronic circuit design, electrical power transmission, acoustics, automotive engineering, aeronautics, and just about any engineering or scientific analysis involving periodic phenomena.

**B.2 Analysis of the Straightforward Method.** Let us estimate the number of arithmetic operations needed to compute a full set of eigenvalues, that is, the eigenvector, of a general circulant  $c \in \mathbf{circ}_n(\mathbb{C})$  using the formula  $\lambda(c) = Fc$  of §1.7 where  $F$  is the Fourier matrix.

We can suppose that the  $n^{\text{th}}$  roots of unity,  $1, \zeta, \zeta^2, \dots, \zeta^n$  are given to us; either they have been computed in advance or they are available from tables. We have  $\lambda(c) = F_n c$  where  $F_n$  is the  $n \times n$  Fourier matrix, which when written out is

$$\begin{array}{rcccccc} \lambda_0 & = & a_0 & + & a_1 & + & \cdots & + & a_{n-1} \\ \lambda_1 & = & a_0 & + & a_1\zeta & + & \cdots & + & a_{n-1}\zeta^{n-1} \\ \lambda_2 & = & a_0 & + & a_1\zeta^2 & + & \cdots & + & a_{n-1}\zeta^{n-2} \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ \lambda_{n-1} & = & a_0 & + & a_1\zeta^{n-1} & + & \cdots & + & a_{n-1}\zeta \end{array}$$

This method requires  $n(n-1)$  additions, and  $(n-1)^2$  multiplications. Denote the time required to do a single complex addition by  $A$ , and the time to a single complex multiplication by  $M$ , then the time required to compute the eigenvector using the Fourier matrix is  $T_F$  where

$$T_F = n(n-1)A + (n-1)^2M$$

The expression 3.5.2.3(ii) for the eigenvalues of the circulant shows a possibility of computing the eigenvector of order  $n$  without having to perform this many arithmetic operations. However, when this technique is analyzed we find that although there is improvement, the process is dominated by approximately  $(\phi(n)/n)n^2$  operations required to calculate the set  $L_{1|n}$  of eigenvalues, and this term still scales as  $n^2$ .

**B.3 The Cooley-Tukey Algorithm.** There is a much better FFT developed by Cooley following an idea of Tukey, and is now called the Cooley-Tukey FFT. This algorithm will always beat the above circulant decomposition algorithm.

Here is a brief description of the Cooley-Tukey FFT. We suppose that  $n = pq$  for some  $p, q > 1$ . To avoid double subscripting, we shall temporarily write  $c_i$  as  $c(i)$ ,  $\zeta_n^x$  as  $e_n(x)$ , and  $\lambda_i(c)$  as  $\lambda[i](c)$ .

$$\begin{aligned}
\lambda[i](c) &= \sum_{j=0}^{n-1} e_n(ij) c(j) \\
&= \sum_{j_1=0}^{q-1} \sum_{j_2=0}^{p-1} e_n((pj_1 + j_2)(qi_1 + i_2)) c(pj_1 + j_2) \\
&\quad \text{where } i_1 = \lfloor i/q \rfloor, i_2 = i \bmod q, j_1 = \lfloor j/p \rfloor, j_2 = j \bmod p \\
&= \sum_{j_1=0}^{q-1} \sum_{j_2=0}^{p-1} e_q(j_1 i_2) e_p(i_1 j_2) e_n(j_2 i_2) c(pj_1 + j_2) \\
&= \sum_{j_2=0}^{p-1} e_n(j_2(qi_1 + i_2)) \sum_{j_1=0}^{q-1} e_q(j_1 i_2) c(pj_1 + j_2) \\
&= \sum_{j_2=0}^{p-1} e_n(j_2(qi_1 + i_2)) \lambda^{(q)}[i_2](c[j_2])
\end{aligned}$$

where  $c[j_2]$  is the circulant vector  $(c(p + j_2), c(2p + j_2), \dots, c(pq - p + j_2))$

$$\therefore \lambda[qi_1 + i_2](c) = \sum_{j_2=0}^{p-1} e_n(j_2(qi_1 + i_2)) \lambda^{(q)}[i_2](c[j_2]) \quad (1)$$

This final formula requires the calculation of  $\lambda^{(q)}[i_2](c[j_2])$  for  $i_2 = 0, 1, \dots, q-1$ , and  $j_2 = 0, 1, \dots, p-1$ . The time required for these calculations will be  $pq(q-1)A + p(q-1)^2M$ . The results of these calculations are plugged into the above formula which will require a further time of  $(p-1)(A+M)$  for every  $i$ , that is,  $n(p-1)(A+M)$ . When all the operations are tallied we get a total of

$$\begin{aligned}
T_c &= (pq(q-1) + pq(p-1))A + (p(q-1)^2 + pq(p-1))M \\
&= n(q+p)(A+M) + O(n)
\end{aligned} \quad (2)$$

Already, this beats the decomposition method at all values of  $p, q > 2$ . But the interesting feature of the Cooley-Tukey FFT is that the process of calculating eigenvalues of lower dimensional circulants can be applied again to the calculation of  $\lambda^{(q)}[i_2](c[j_2])$  when  $q$  is compound. This opens up the possibility of a recursive calculation which applies the above method to reduce the calculation to circulants of prime dimension. Thus, by taking  $p$  to be the smallest prime dividing  $n$  at each step, the cost of executing this FFT method will be dominated by the  $q^2$  operations required for the lower-order eigenvalue calculations until  $q$  approaches  $p^2$ . Thereafter, the process is dominated by  $qp^2$ .

Even the last step in the Cooley-Tukey FFT can be viewed as a collection of  $q$  calculations of eigenvalues for circulants of order  $p$ , and it is therefore also amenable to a recursive reduction when  $p$  is compound. To see this, in (1) fix  $i_2$ ; and define a vector  $v$ , and eigenvalues  $\mu$  (both depending on  $i_2$ ) by

$$\begin{aligned}
v_j &:= e_n(ji_2) \lambda^{(q)}[i_2](c[j]) \\
\mu_i &:= \lambda[qi + i_2](c)
\end{aligned}$$

Equation (1) becomes

$$\mu_i = \sum_{j=0}^{p-1} \zeta_n^{qij} v_j = \sum_{j=0}^{p-1} \zeta_p^{ij} v_j, \quad i = 0, 1, \dots, p-1$$

That is,  $\mu = F_p v$ , which can also be reduced using the Cooley-Tukey method when  $p$  is compound.

Readers interested in frequency analysis of sample data, and who have some measure of control over the value of  $n$ , should be aware that there is an especially efficient FFT available when  $n$  is a power of 2 called the Bit-Reversal Method. It is based on a finding of Danielson and Lanczos. See [PFTV].