
```
1. procedure R_FFT( $X, Y, n, \omega$ )
2. if ( $n = 1$ ) then  $Y[0] := X[0]$  else
3. begin
4.   R_FFT( $\langle X[0], X[2], \dots, X[n-2] \rangle, \langle Q[0], Q[1], \dots, Q[n/2] \rangle, n/2, \omega^2$ );
5.   R_FFT( $\langle X[1], X[3], \dots, X[n-1] \rangle, \langle T[0], T[1], \dots, T[n/2] \rangle, n/2, \omega^2$ );
6.   for  $i := 0$  to  $n - 1$  do
7.      $Y[i] := Q[i \bmod (n/2)] + \omega^i T[i \bmod (n/2)]$ ;
8. end R_FFT
```

Algorithm 13.1 The recursive, one-dimensional, unordered, radix-2 FFT algorithm. Here $\omega = e^{2\pi\sqrt{-1}/n}$.

```

1.  procedure ITERATIVE_FFT( $X, Y, n$ )
2.  begin
3.       $r := \log n$ ;
4.      for  $i := 0$  to  $n - 1$  do  $R[i] := X[i]$ ;
5.      for  $m := 0$  to  $r - 1$  do      /* Outer loop */
6.          begin
7.              for  $i := 0$  to  $n - 1$  do  $S[i] := R[i]$ ;
8.              for  $i := 0$  to  $n - 1$  do /* Inner loop */
9.                  begin

/* Let  $(b_0b_1 \dots b_{r-1})$  be the binary representation of  $i$  */

10.                      $j := (b_0 \dots b_{m-1}0b_{m+1} \dots b_{r-1})$ ;
11.                      $k := (b_0 \dots b_{m-1}1b_{m+1} \dots b_{r-1})$ ;
12.                      $R[i] := S[j] + S[k] \times \omega^{(b_m b_{m-1} \dots b_0 0 \dots 0)}$ ;
13.                 endfor; /* Inner loop */
14.             endfor; /* Outer loop */
15.         for  $i := 0$  to  $n - 1$  do  $Y[i] := R[i]$ ;
16.     end ITERATIVE_FFT

```

Algorithm 13.2 The Cooley-Tukey algorithm for one-dimensional, unordered, radix-2 FFT. Here $\omega = e^{2\pi\sqrt{-1}/n}$.