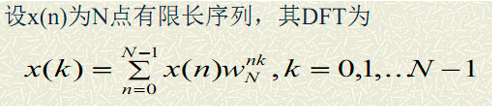
FFT与DFT的计算复杂度相比较

<https://blog.csdn.net/sdwuyulunbi/article/details/7476740>

# DFT的定义及复杂度分析

## DFT的定义



## DFT的计算复杂度分析

一般情况下，x(n)和wNnk都是复数，X(k)也是复数，因此每计算一个X(k)值，需要N次复数乘法运算以及(N-1)次复数加法运算。而X(k)共有N个点，所以**完成整个DFT运算总共需要N\*N次复数乘法运算和N(N-1)次复数加法运算**。

由于一次复数乘法运算需用四次实数乘法运算，一次复数加法运算需要二次实数加法运算，因此每运算一个X(k)需要4N次实数乘法和2(N-1)+2N=2(2N-1)次实数加法运算。**故整个DFT运算总共需要4N\*N次实数乘法运算和N\*2(2N-1)=2N(2N-1)次实数加法运算**。

当N=1024时，DFT需要1,048,576次复数乘法运算。所以，直接计算DFT对实时性很强的信号处理来说，对计算速度要求实在是太高了。

N点DFT共需要N\*N次复数乘法和N(N-1)次复数加法，共4N\*N次实数乘法和(2N\*(N-1)+2N\*N)= 2N(2N-1)次实数加法。当N很大时，这是一个非常大的计算量。

## 补充：复数乘法运算及其优化

### 一般的复数乘法运算：

复数乘法运算：设z1=a+bi，z2=c+di(a、b、c、d∈R)是任意两个复数，那么它们的积(a+bi)(c+di)=(ac-bd)+(bc+ad)i. 需要计算出ac、ad、bc、bd四个乘积，即四次实数乘法，又计算(ac-bd)和(bc-ad)两次加法运算。

因此，**一般情况下，一次复数乘法运算需用四次实数乘法运算和两次实数加法运算。(一次复数加法运算需要二次实数加法运算。)**

### 复数乘法运算优化：仅用**三次实数乘法**完成复数相乘的运算

有两个复数m=a+b\*i,n=c+d\*i,现在计算它们的乘法，设计一个只使用三次实数乘法的算法来完成m和n的计算。

根据复数乘法的计算公式：**m\*n=（a\*c-b\*d）+(a\*d+b\*c)i**，可以使用加法或者减法来削减乘法运算的数量。

计算C1=a\*d,C2=b\*c,C3=(a+b)(c-d)，虚部为C1+C2，实部为C3+C1-C2，计算完毕。

这样的话，**一次复数乘法运算需要三次实数乘法运算和五次实数加法运算。**

**优化：利用三次实数加法运算换取了一次实数乘法运算。**

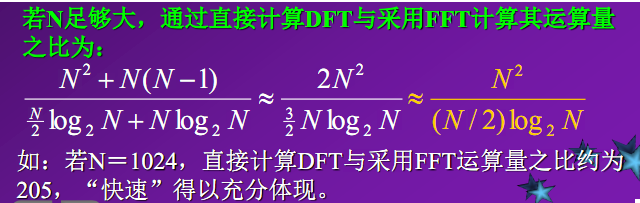
# FFT的复杂度分析

利用FFT算法之后，任何一个N为2的整数幂（即N= 2M）的DFT，都可以通过M次分解，最后成为**2点的DFT**(蝶形运算)来计算。**M次分解构成了从x(n)到X(k)的M级迭代计算，每级由N/2个蝶形运算组成**。**完成一个蝶形计算需一次复数乘法和两次复数加法**。因此，**完成N点的时间抽选FFT计算的总运算量为：**

**复数乘法次数：M\*N/2= N/2\*log2N**

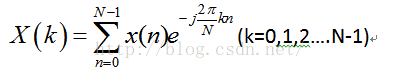
**复数加法次数：M\*2\*N/2= N\*log2N**

大多数情况下复数乘法所花的时间最多，所以以复数乘法的计算次数来比较DFT与FFT的效率为：**DFT/FFT=2N/log2N**。

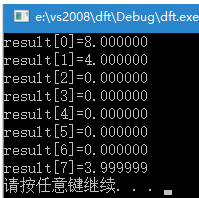


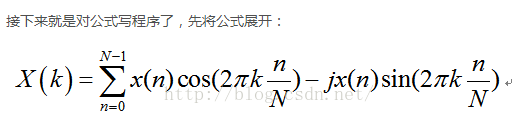
# DFT的进一步分析

DFT的公式：



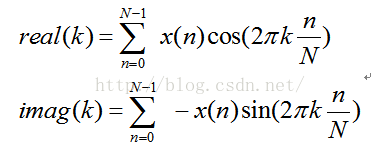
从这个公式可以看出，变换后的数据就是原信号对cos和sin的相关操作，即进行相乘求和（连续信号即为积分）,为什么我要将n\N写在2k\*pi后面呢？因为我觉得在对cos和sin进行相关操作时，k代表和频率为多少的正弦相关，而n和N则是在一个正弦周期内采样N个点，采样间隔为2\*pi\N,,n用来步进，一次步进2\*pi\N,最后进行累加求和，就得出了X(k),《实用数字信号处理》这本书的DFT章节详细的解释了此公式，并且还进行了举例,看了以后明白了不少，另外，DFT之后的数据是对称的，具体原因还是在那本书上面有，在FFT的章节。比如做8点DFT，采样信号为x(n),DFT之后的数据为X(k),那么X(0)为直流信号,X(1), X(2), X(3), X(5), X(6), X(7),关于X(4)对称,即X(1)=X(7), X(2)=X(6),X(3)=X(5),如下图，是对1+sin(2\*PI)进行DFT变换，具体的幅值先不关心，只要知道它是对称的就行了。





**(I与Q的正交分解。)**

在计算机中可以这样展开：

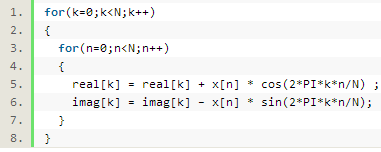


里面有个j，不用管它，我们用两个数组，一个保存sin相关，一个保存cos相关，由于cos为实部，sin为虚部，可以定义以下两个数组：

float real[N];//用来保存cos相关。

float imag[N];//用来保存sin相关。

程序实现：



Real就是cos相关的幅值，imag就是sin相关的幅值。

最后将sin与cos合成一个sin：





<https://blog.csdn.net/axiqia/article/details/50904167>