# DSSS代码学习记录文档

本文档对应代码DSSSBPSK.m，这份代码不是我写的。由于之前看过DSSS-BPSK代码，当时看仅是根据这份代码而看，尽管涉及到的代码框架和代码方法，当时看来就认为“就是这样用的”，但对里面中涉及的一些原理等于没懂。

后来，导师给了一份扩频信号文件（中频70MHz，采样率38.4MHz），让去使用这份代码将码速、扩频码长、扩频码估计出来，再进行解扩。于是，就遇到了很多问题，每个模块输出的示意图都不清楚对应着什么，实在是进行不下去。

于是开始重新看这份代码，并借鉴相关论文，力求从原理上搞清楚代码中的原理。毕竟代码肯定不是乱写的，肯定是依据数学公式过来了，而数学公式就在书中、论文中。要根据这些理论去看代码，效果会更好。这次将新的收获以文档形式记录下来。2024/01/12

同一个符号在不同的频率上发送，叫“频率分级”，作用是抗衰落。

## 1 DSSS-BPSK参数估计

本文档是一边看论文及代码一边写得，所以文档里面涉及的论文可能比较局限，也没有涉及到相关书籍。后续要是看了再补充。知网、万方里有很多关于直扩信号参数估计检测的论文，都可以查阅来看看。

### 1.1 信号生成

DSSSBPSK.m文件生成代码中含有“成形滤波”，细看了一下，成型后的信号速率是成型前的扩频码速的四倍。这里成型操作是：信息码经过扩频码扩频后，再根据扩频信号每个码片后面插值3个零，在经过一个线性滤波器，使得成型后的信号速率是成型前的扩频码速的四倍。

DSSS.m文件生成代码中添加了flag标志位，作为“成型开关”。细看了以下，成型后的信号速率是成型前的扩频码速的四倍。这份代码里“成型”的操作是：信息码经过扩频码扩频后，再根据扩频信号每个码片后面插值3个零，在经过一个线性滤波器，使得成型后的信号速率是成型前的扩频码速的四倍；“不成型”的操作是：信息码经过扩频码扩频后，再根据扩频信号对每个码片后面重复三次，使得成型后的信号速率是成型前的扩频码速的四倍。

|  |  |
| --- | --- |
| (a) DSSSBPSK.m文件中的生成 | (b) DSSS.m文件中的生成 |

DSSS.m代码中生成实部示意图(升采样前)

|  |  |
| --- | --- |
|  |  |

### 1.2 参数估计

在不修改上述生成代码的情况下，继续该生成信号进行参数估计。以下出现的FFT指得是matlab执行FFT函数并取abs(模值)后的结果。

#### 1.2.1 载频估计

#### 1.2.2 扩频信号检测

##### 1.2.2.1 基本编程语法

[具有常量 y 值的水平线 - MATLAB yline - MathWorks 中国](https://ww2.mathworks.cn/help/matlab/ref/yline.html) 这个链接是关于如何在figure里面绘制水平线，里面涉及到yline函数，因为不知道如何在figure图片中绘制水平线，所去搜索了一下，之后感觉这个yline函数很好用。不过问题又来了，那么绘制水平线之后又不想要了，想要去掉怎么办呢？直接问chatgpt，它给的答案是 将yline函数的返回值赋给变量hline，再调用delete函数删掉即可（运行代码delete(hline)，即可看到figure中没有水平线了。）

##### 1.2.2.2 检测方法

再次回到扩频信号检测，DSSSBPSK.m代码和DSSS.m代码中在该模块使用的是基本思想等同于基于延迟相乘的二次功率谱估计方法，可以参考1.2.4.2.1节。但是我目前无法将该模块代码与论文中基于延迟相乘的二次功率谱估计方法对应上。只能大致认为基本思想是这个方法。还需要再看看相关论文补充知识！！！

这里解释以下代码中相关参数的表示意义。

|  |
| --- |
| 图(a) DSSSBPSK.m文件 |
| 图(b) DSSS.m文件 |

两份代码虽然写法不同，其实是一个意思。就是检测出二次谱中的相距一定长度的峰值个数，如果个数不小于2，则认为是扩频信号。这应该是DSSS信号二次方谱特征。

那么这个阈值（(a)图中的0.7，(b)图中的0.967）应该是根据matlab中plot(xgjf1)，plot(XX\_tDec)绘制出来之后人为观测后得出的。根据1.2.4.2.1节内容以及该代码使用的是一个周期的PN码调制一位信息码，因此这个“相距一定长度”中的长度大致是接近且小于扩频码长乘以一个码片采样点数。图(b)中500，即使根据扩频码长127，对一个码片采样4个点即508，得出的。当然也可以根据plot函数绘制完图像之后人为观测获得。下图为DSSS.m该模块运行效果图。显然8588-8082 = 506，接近508。

|  |  |
| --- | --- |
| 图(1) DSSS.m该模块运行效果图 | 图(2) DSSS.m该模块运行效果图 |

上面代码图中还标记了一处：“xgjf1 = xgjf1(jiancedian1+yanchi1:end)”和“XX\_t\_hDec=XX\_tDec(IvDec:ncDec/2)”这是因为直接对二次方谱plot的话，从图上看除左边第一峰值很高之外貌似是对称的，所以上述代码选择了前半段绘图，见上图(2)所示。至于为什么近似对称呢，是因为fft后正频率和负频率对称导致的吗？我还不是很确定，有机会再研究研究！！

补充：

由于直接plot(XX\_tDec)有一个峰值特别大，导致整个图看上去对比度太低，因此加了一个ylim函数。给纵坐标限制一个范围，有了图(2)效果。代码如下：figure; plot(fftshift(XX\_tDec),'b'); title("扩频信号检测"); ylim([0.21 0.245])

现在时间：2024/04/02 16:35

这份文档是几个月前写得了，但是没有发现扩频信号的问题。经过不断试错之后发现，上面所示的扩频信号检测示意图是不正确的，扩频序列的自相关性已经被削弱很多。正确的扩频信号检测示意图如下所示。

|  |  |
| --- | --- |
| (a)缩小图 | (b)放大图 |
| (c)截取一半绘图 | (d)放大图 |

只有了正确的扩频信号，才能对该扩频信号进行正确的参数估计。

#### 1.2.3 码速估计

从论文中获悉这里的码速应该对应的扩频之后的码速。

##### 1.2.3.1 延迟相乘法

参考论文《直扩信号检测和PN码参数估计的研究》——孙铭芳。具体对应论文中3.2节延时相乘法的理论。这里的理论方法与代码几乎对应。数学公式推导见论文，这里省略不提。下面是论文中该方法对应的结果分析。

|  |  |
| --- | --- |
|  |  |

从式（3-7）和（3-9）可以看出，功率谱在频率处和二倍载频及、处有离散谱线，且离散谱线的位置与延时时宽的大小并无关系。频率与伪码码速率或码元宽度的对应关系为，因此离散谱线的位置代表了码速率的信息。连续谱的包络为型，而代表码速率的频率位于连续谱主瓣的衰减处，这一特性说明位于处的离散谱线，不易被连续谱覆盖，有助于顺利检测到该离散谱线。因此我们可以在检测到该离散谱线后根据其频域位置来提取码速率。

考察当取不同值时，式（3-7）中表示离散谱线的表达式的值变化。当取时，离散谱线幅度为0；当时，离散谱线幅度为；当时，即K=1/2，离散谱线幅度最大，为。

图中横坐标Tdn表示延时Td的大小与PN码chip时宽Tc的大小的比值，即Tdn=Td/Tc。由图3-4中可以看出，Tdn=0.5时，谱线的幅度最大，即对检测到代表码速率的谱线最有利。

上面文字全来自论文。这里补充一下，按照论文描述，这个延迟时间，谱线峰值最大，效果也更好。在仿真中，比如码元宽度，码片宽度，N为扩频码长，则，那么这里估计的码速即为，也就是码元扩频后的速率。若扩频后，对每个码片又采样了4个点，即，那么这里时，仿真时延迟就设置为2个点。

下面是代码中的仿真，一个码片对应60个采样点，代码中设置了不同的延迟，效果如下。结果显示，T延迟Tc时谱线峰值最高，与论文中有些冲突。暂时不清楚具体原因。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

下图是故意添加了频偏，频谱图如下。

|  |  |  |
| --- | --- | --- |
| (a) | (b) | (c) |

从上面示意图看，这份代码中的码速估计部分，频谱图示意效果，码速对应的峰值延迟时间以及载频有关，而且与论文中《直扩信号检测和PN码参数估计的研究》不太能对应上，似乎没有看到“二倍载频及、处有离散谱线”。

我想表达的意思是：在使用这份代码过程中，最终运行出来的频谱示意图与下图中代码的几个参数有关，同时为了达到这份代码的效果要根据运行出的频谱示意图修改相关参数（修改的依据应该是频谱示意中两个相对较高的谱线差，即为估计的码速（如图(a)），这与论文中提出的结论貌似可以对的上，即“功率谱在频率处有离散谱线”）。

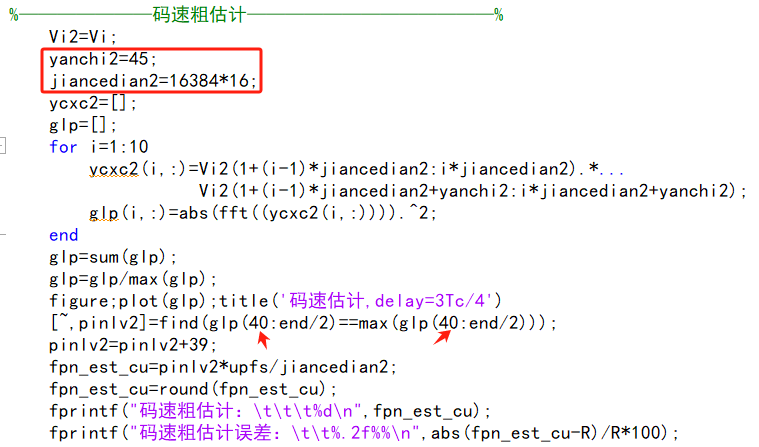


图 代码修改示意图

补充：

又继续看了一下论文，论文中“3.2.2延时相乘法改进结构的算法推导”提到了。然后下面又展现了仿真图。

|  |  |
| --- | --- |
| 图1 信噪比是0dB的情况下仿真 | 图2 信噪比是-14dB的情况下仿真 |

图 延时相乘法的两种结构的比较

发现这份代码提供的频谱示意图与论文中改进结构提供的频谱示意图很相似，因为上图b)结构示意图中“功率谱在频率处有离散谱线”且相对较高；“二倍载频及、处有离散谱线”幅值很低。在这点上，与代码运行的频谱示意图很貌似对得上。

##### 1.2.3.2 延迟相乘法算法仿真

参考论文《直扩信号检测和PN码参数估计的研究》3.3节理论仿真结果。

延时相乘法处理的具体算法如下，这部分完全来自论文内容。

1. 根据先验信息（扩频码元宽度可能的取值范围）确定延时的点数，尽量

选取；

2. 信号作延时相乘；

3. 采取3dB门限判决，高于门限即判定提取到正确的码速率；

4 如果未搜索到代表码速率的谱线，调整延时点数，重复 2、3 步骤。

DSSSBPSK.m代码没有完全按照论文中提到的算法实现，因此还可以按照上述算法加以改进！！！

#### 1.2.4 码长估计

论文《直扩信号检测和PN码参数估计的研究》中“2.2.2 伪随机码”节内容上面，提到了短PN码调制和长PN码调制概念，我也是第一次看到这个概念，可以花时间再研究研究。下面我根据论文内容试着给出短PN码调制和长PN码调制的区别示意图。其中指的是PN码周期，指的是信息码元长度

|  |
| --- |
| 图1 短PN码调制 |
| 图2 长PN码调制 |

由上图可知，短PN码调制相当于一个信息码元长度由多个周期的PN码去调制；长PN码调制相当于一个周期PN码内调制了多个信息码元。个人理解，可能存在误解的情况。那么写到这，尴尬的点出来了：这份DSSSBPSK.m代码中是一个信息码元内由一个周期的PN码调制，那么这应该较短PN码调制呢？还是长PN码调制呢？暂且认为这份DSSSBPSK.m代码中对应的调制就是短PN码调制，这里一个信息码元长度用一个PN码周期调制。

上面疑问解答补充：

论文《直扩信号检测和PN码参数估计的研究》中4.4.2节对长码调制信号的码周期的估计性能比较，提到短PN码调制和长PN码调制的区别。

论文《直扩信号的检测与恢复》--简秦勤，在第五章直接序列扩频信号的码型恢复提到了两者区别。

##### 1.2.4.1 短PN码调制下PN码周期估计

###### 1.2.4.1.1 二次功率谱估计方法

“所谓二次功率谱，即是信号的功率谱作二次处理，将信号的功率谱作为输入信号再求其功率谱从而得到信号的二次功率谱”选择论文4.1 二次功率谱估计方法章节内容。

这部分原理公式都提到了，但是我有个疑点就是：matlab仿真中如何编写二次功率谱代码呢？一次功率谱代码如何编写？直接对信号进行FFT？看了网上博客<http://t.csdnimg.cn/S85eL> 还是不太清楚这个功率谱代码怎么正确编写，二次功率谱代码如何正确编写？之后再花时间研究研究！！

论文中二次功率谱算法流程图如图所示：

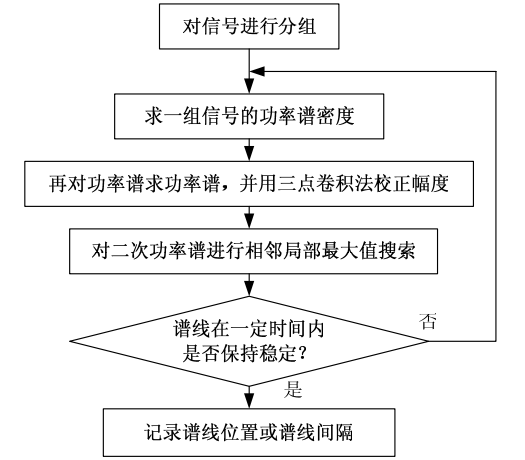


图 二次功率谱算法流程图

论文中该算法的示意图如下：

|  |  |  |
| --- | --- | --- |
| (a)码长为127 | (b)码长为127 | (c)累加次数与信噪比容限关系 |

###### 1.2.4.1.2 倒谱估计方法

所谓倒谱，就是信号功率谱的功率谱。但是在求第二次功率谱之前要对第一次的功率谱先求其对数，这是与前面的功率谱二次处理方法的不同之处。倒谱的数学表达式为：

|  |  |
| --- | --- |
|  |  |

倒谱的工作过程如下，信号经过功率谱运算并滤波后，成为加权的正弦信号的形式，再经过求对数和功率谱变换，就得到了表示直扩信号存在的脉冲输出，此脉冲的位置表示了该正弦信号的频率，由此可得到 PN 码周期。如果输入信号不是扩频信号，“假设为一正弦信号，则其第一个功率谱变换为一脉冲，经滤波后进入第二次功率谱变换，其输出为幅度很低的三角波输出，因而检测不到其存在。”

上述文字全来自论文《直扩信号检测和PN码参数估计的研究》，上面提到“假设为一正弦信号，则其第一个功率谱变换为一脉冲，经滤波后进入第二次功率谱变换，其输出为幅度很低的三角波输出，因而检测不到其存在”，这表明脉冲一次功率谱变换是一个三角波吗？忘了，还要再看看相关书籍！！！！

下面是该算法示意图：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

图 倒谱算法示意图

突然发现，由倒谱的数学公式是不是可以认为，在matlab编程时，二次功率谱就是对信号做一次FFT，再做一次FFT，即调用两次fft函数，估计这不太规范甚至有误。

另外，论文中“4.2.2 理论仿真结果”表明，使用一周期PN码调制一位信息位，采样率是PN码速率的40倍，即一个码片采样40个点，分段长度40000，累加次数127。

|  |  |  |  |
| --- | --- | --- | --- |
| (a) -10dB | (b) -12dB | (c) -13.7dB | (d) -14dB |

图 不同信噪比下的倒谱计算结果

从图 (a~c)中可以看出有周期谱线，通过检测这些谱线可以检测直扩信号的存在，而且可以通过周期谱线之间的间隔估计出直扩信号的 PN 码周期。在图中，周期谱线的间隔是5080。因为PN码速率是1MHz，采样率是40MHz，所以根据仿真结果计算得到的码周期是5080/（40/1）=127，正好是DS信号的PN码周期。

上面来自论文内容。也就是说，在一个周期PN码调制一位信息码元时，当我们检测到周期谱线差时，用差除以一个码片采样点数即为PN码长。那么当n个周期PN码调制一位信息码元时，其它参数不变，那么码长计算公式是：PN= 5080/(40/n)吗？

试着给出答案：由上述短PN码调制示意图知：假设一位信息码由四个周期PN码调制，那么信息码速Rb，PN码速为Rc = 4\*127\*Rb，….写不出，之后再看看。

##### 1.2.4.2 长PN码调制下PN码周期估计

###### 1.2.4.2.1 基于延时相乘的二次功率谱估计方法

参考论文《直扩信号检测和PN码参数估计的研究》4.3节。

二次功率谱和倒谱法从本质上说都是对功率谱进行二次处理，不同之处是倒谱法在对功率谱再求功率谱之前先求对数，这样有利于离散谱线的提升，因而倒谱法能做到更低的信噪比。但是，这两种方法对长PN码扩频调制均不适用。第三章提到的延时相乘法可以对直扩信号进行检测和PN码速率估计，所以考虑在延时相乘法和二次功率谱法的基础上提出一个新的方法来估计PN码参数，原理框图如图。先对直扩信号延时相乘，求自相关后再求两次功率谱后，可以在PN码周期的整数倍处得到离散谱线，由此可以估计出PN码周期。

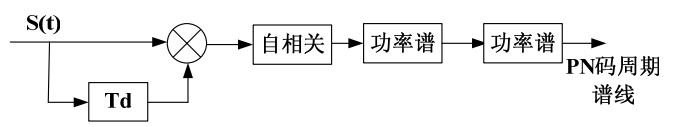


图 基于延时相乘的二次功率谱算法原理框图

仿真实现。仿真数据是码长为127，码速率为1MHz，载波频率为5MHz，采样率是 40MHz，一周期PN码调制一位信息码，在信噪比是-5dB的情况下，对其先作延时相乘，再求二次功率谱，得到离散谱线。由图中可以看出，取其中相邻两条谱线求期间隔，10061-4981=5080。而根据仿真数据的参数可以算出 PN 码周期为127\*(40/5)\*(5/1) = 5080，正好是仿真所得谱线间隔。



图 采用基于延时相乘法的二次功率谱法估计码周期的样本

这篇论文中由该算法得出的结论是“倒谱法和二次功率谱法对于长 PN 码扩频调制的信号进行PN码周期的估计不适用，而基于延时相乘的二次功率谱法则能很好的对长PN码码周期估计出来”

这份代码DSSSBPSK.m采用的就是该算法。我现在还无法将代码与该算法完全对应上。

#### 1.2.5 码序列估计

##### 1.2.5.1 码序列估计核心思想

这份代码码序列估计的核心思想就是：对已经下变频后的扩频序列估计码速，通过码速进而估计码周期。

然后利用码周期进行自相关操作，找到伪码的起始点，再接着利用这个起始点，开始对扩频信号进行积分操作，每一次积分长度为一个码片周期，再进行判决，即可还原出一个码片（这里与真实值只有一个符号的差别，后续再进行操作可消除这个符号）。

当最后所有码片都还原出来后，即一个伪码也还原出来了。

但是，仿真测试发现，这种方法容易受频偏的影响。显然，含有频偏的扩频信号，即使正确找到伪码起始点，那么之后码片周期进行积分判决很容易出错。

##### 1.2.5.2 码序列估计实验仿真

这部分目前参考几篇论文《直扩信号的检测与恢复》--简秦勤；《长码直扩信号中的扩频序列估计》--邱轶修；《非合作直扩信号扩频序列盲估计》--贾鹤鸣；《直接序列扩频信号参数及扩频序列估计算法的研究》--王猛

对DSSS.m代码中输入该模块得信号实部进行查看，如下图所示：

|  |  |
| --- | --- |
| (a)实部示意图 | (b)实部放大示意图 |
| (c)自己写生成代码进入该环路实部图 | (d)自己写生成代码进入该环路实部放大图 |

显然(d)可知，实部要是大于0和小于0区分如此明显，进入下面码元序列估计显然会出问题，根本估计不出来。因为下面估计代码中使用了sign，去对码片进行判决，而(d)图中一个符号要么都在上方，要么都在下方，判决不是1就是-1，一个周期PN码判决结果会存在大串1或者-1，如下图所示，错误明显。

|  |  |
| --- | --- |
|  |  |

现在时间：2024/04/02 16:47

上述扩频序列估不出来的原因就是扩频信号生成不对，生成的信号已经都没有了扩频序列的自相关性了。见扩频序列检测那一节。

## 2 DSSS仿真实现

现在时间：2024/04/02 17:09

第一节内容大概是几个月前写的了，现在我有些时间，而且最近也在学习《扩频通信》这门课程，对扩频技术有了一定的了解。本节的目标是自己编写直接序列扩频（DSSS）信号的生成和参数估计，以及解调代码。在编写代码的过程中，我会尽量使用有理有据的方法，并参考相关论文中的原理进行复现。希望通过记录这一过程，能够更好地理解和掌握扩频通信技术。

### 2.1 信号生成

### 2.2 参数估计

### 2.3 信号解扩

### 2.3 信号解调

## 3 扩频信号文件测试记录

现在时间：2024/04/24 19:57

第二部分“DSSS仿真实现”章节，目前还一字未留。其实DSSS仿真代码已经在写且调试了，但还没有完全写完，还差跟踪模块的代码。

今天准备记录第一部分“DSSS-BPSK参数估计”中提到了中频文件测试过程。

### 3.1 中频文件测试记录

#### 3.1.1 信号读取

|  |
| --- |
|  |

图 测试信号文件

表 读取代码示意

|  |  |
| --- | --- |
| 代码一 | 代码二 |
| clear all;  close all;  %% select a file  [filename, pathname] = uigetfile('\*.\*','Selecte A File','D:\MODIFI\_CODE\CDMA\1\_2\中频70.000M\_采样率38400.000K\_IQ8bit\'); %%实际信号  if isequal(filename, 0)  disp('User selected Cancel');  return;  else  disp('Filename：');  disp(filename);  end  %% Get file length  filepathname = fullfile(pathname, filename);  inf = dir(filepathname);  file\_len = inf.bytes - 50;  fprintf('File length：\n%ld Byte\n', fix(file\_len));  fid = fopen(filepathname, 'r');  if fid<0  disp('Cannot open the file!');  return;  end  fseek(fid, 0, 'bof');  datalen = 1024\*1024\*100;  % data = fread(fid,datalen,'int16');  %读测试环境采集下来的信号文件  data = fread(fid,datalen,'int16','ieee-be');  %读matlab里面生成的PCM文件，放到信号发生  %器里过接收机下的PCM文件  fclose(fid);  signal=data(1:2:end-1)+1i\*data(2:2:end);  signal=conj(signal'); | clear all;  close all;  %% select a file  [filename, pathname] = uigetfile('\*.\*','Selecte A File','D:\MODIFI\_CODE\CDMA\1\_2\中频70.000M\_采样率38400.000K\_IQ8bit\'); %%实际信号  if isequal(filename, 0)  disp('User selected Cancel');  return;  else  disp('Filename：');  disp(filename);  end  %% Get file length  filepathname = fullfile(pathname, filename);  inf = dir(filepathname);  file\_len = inf.bytes - 50;  fprintf('File length：\n%ld Byte\n', fix(file\_len));  fid = fopen(filepathname, 'r');  if fid<0  disp('Cannot open the file!');  return;  end  fseek(fid, 0, 'bof');  datalen = 1024\*1024\*100;  data = fread(fid,datalen,'int16');  %读测试环境采集下来的信号文件  %data = fread(fid,datalen,'int16','ieee-be');  %读matlab里面生成的PCM文件，放到信号发生  %器里过接收机下的PCM文件  fclose(fid);  signal=data(1:2:end-1)+1i\*data(2:2:end);  signal=conj(signal'); |

#### 3.1.2 信号测试

这里记录一下上面两种代码读取信号文件的测试效果。

将信号进入扩频信号检测模块，也间接的获得扩频序列长度（采样后）。这里延迟点设置为6。

|  |  |
| --- | --- |
| (a) 代码一 | (b) 代码二 |

从上面示意图来看，两种读文件代码貌似对扩频信号检测没影响，并且有明显的谱峰在的，且谱峰之间相差4000。按道理来说，这谱峰差应该是扩频码长乘以采样倍数。又被告知这个扩频文件是一个周期的扩频码调制一个符号，而扩频码周期应该是2^n-1，显然4000这个值感觉怪怪的。难道是非整数倍采样？

接下来进行码速和载频估计效果也很差。感觉不像是代码的问题，代码是针对短码调制的扩频信号，是信号的问题？感觉像是，工作停滞。

### 3.2 采集信号文件测试记录

#### 3.2.1 信号读取

这里的读文件的代码采用的是3.1.1节中的代码二，读取测试环境采集下来的信号文件。

|  |
| --- |
|  |

图 信号文件

上述文件时m生成扩频码127，信号速率1k，采样率是508k。目前测试第一个文件，发现扩频码长以及码速载频都能估计出来。从扩频信号检测模块结果示意图来看，中间三个文件扩频序列自相关性被破坏了。显然码长是估计不出来的。

#### 3.2.2 信号测试

##### 3.2.2.1 dis3303711文件

|  |  |
| --- | --- |
| (a) 码速载频估计模块输出 | (b) 左图计算结果 |

##### 3.2.2.2 dis3066894文件

|  |  |
| --- | --- |
| (a) 码速载频估计模块输出 | (b) 左图计算结果 |

##### 3.2.2.3 dis2880859文件

|  |  |
| --- | --- |
| (a) 码速载频估计模块输出 | (b) 左图计算结果 |

##### 3.2.2.4 扩频信号检测效果

上述三个文件的扩频信号检测输出示意图，显然自相关性被破坏了。

|  |
| --- |
|  |

## 4 直接扩频信号测试

代码位于下面地址，文件名为CDMA\_test。

D:\MODIFI\_CODE\CODE\_FROM\_LAB\signal\_24\code\_from\_cuiyaming\signal\_send\CDMA

### 4.1 跟踪环路测试记录

PN码采样后长度为508点。

|  |  |
| --- | --- |
| (a) 无延迟 | (b) 延迟一个点 |
| (c) 延迟两个点 | (d) 延迟三个点 |
| (d) 延迟四个点 | (e) 延迟508个点 |

#### 4.1.1 跟踪环路改进

时间：20240521

显然上述代码测试运行效果图来看，阈值很难区分。所以需要改进跟踪环路代码。

#### 4.1.2 跟踪环路改进调试记录

以下仿真是对DSSS-BPSK的m代码进行仿真的，PN=127；一个码片4个采样点，然后信噪比为0dB情况下，符号速率是1000Band。

##### 4.1.2.1 信号超前PN码测试记录

|  |  |
| --- | --- |
| (a) 无时钟误差E、P、L输出 | (b) 无时钟误差时环路输出 |
| (c) 信号超前PN码一个点E、P、L输出 | (d) 信号超前一个点的环路输出 |
| (d) 信号超前PN码两个点E、P、L输出 | (e) 信号超前两个点的环路输出 |

##### 4.1.2.2 信号滞后PN码测试记录

|  |  |
| --- | --- |
| (a) 无时钟误差E、P、L输出 | (b) 无时钟误差时环路输出 |
| (c) 信号滞后PN码一个点E、P、L输出 | (d) 信号滞后一个点的环路输出 |
| (d) 信号滞后PN码两个点E、P、L输出 | (e) 信号滞后两个点的环路输出 |

##### 4.1.2.3 信号超前或滞后PN码EPL输出测试记录

|  |  |
| --- | --- |
| (a) 信号滞后PN码一个点时E、P、L输出 | (b) 信号超前PN码一个点时E、P、L输出 |
| (c) 信号滞后PN码两个点E、P、L输出 | (d) 信号超前PN码两个点时E、P、L输出 |

由上图(a)(b)(c)(d)仿真结果来看，比较E、P、L三值大小，可得出如下结论：

(1)当出现时钟误差时，即信号超前或者滞后PN码，P输出始终处于中间位置。

(2)同时，无论是一个点(T\_chip/4)还是两个点(T\_chip/2)，信号滞后PN码时，L输出时最大的；信号超前PN码时，E输出时最大的。

##### 4.1.2.4 信号超前或滞后PN码Loop输出测试记录

|  |  |
| --- | --- |
| (a) 信号滞后PN码一个点时环路输出 | (b) 信号滞后PN码一个点时环路差分输出 |
| (c) 信号滞后PN码两个点环路输出 | (d) 信号滞后PN码两个点时环路差分输出 |
| (e) 信号超前PN码一个点时环路输出 | (f) 信号超前PN码一个点时环路差分输出 |
| (g) 信号超前PN码两个点时环路输出 | (h) 信号超前PN码两个点时环路差分输出 |

从上面图所示，可得出如下结论：

(1)无论信号超前还是滞后PN码，环路输出都会在时钟误差时刻出现拐点，对环路输出进行差分即可看出明显峰值。(备注：环路差分输出在第一个值往往很大，原因时进行仿真时先对环路输出变量进行初始化设为0，环路输出从索引2开始的)

### 4.2 跟踪环路修改记录

#### 4.2.1 m代码修改

代码位于下面地址，文件名为CDMA\_test\_two\_version。

D:\MODIFI\_CODE\CODE\_FROM\_LAB\signal\_24\code\_from\_cuiyaming\signal\_send\CDMA\_SYF

修改原因：由于之前代码都是生成一段数据然后直接解扩，但是上机测试发现这种有点脱离实际。测试过程是先从连续信号里面取出一段进行解扩，然后在继续取下一段进行解扩。为了满足测试需求，于是进行了修改。

|  |  |
| --- | --- |
| % 跟踪环路参数  sigma = 0.707;  Ko\_1=0.01; %压控振荡器增益0.1  Kd\_1=1; %鉴相器增益  K\_1=Ko\_1\*Kd\_1;  % fs = round(est\_rb) \* samplePoint;  % T\_nco=1/fs;  T\_nco=1/(est\_rb \* samplePoint);  Wn\_1=8\*sigma\*BL\_1/(1+4\*sigma^2);  C1=(8\*sigma\*Wn\_1\*T\_nco)/(K\_1\*(4+4\*sigma\*Wn\_1\*T\_nco+(Wn\_1\*T\_nco)^2));%环路滤波器系数  C2=(4\*(T\_nco\*Wn\_1)^2)/(K\_1\*(4+4\*sigma\*Wn\_1\*T\_nco+(Wn\_1\*T\_nco)^2)); %环路滤波器系数K3  PN\_sps\_length = PN\_length1\*samplePoint;  slice\_temp = 3; % 分三段  slice\_length = length(RR) / slice\_temp;  for k = 1 : 1 :slice\_temp  % 分段数据  RR\_slice(k,:) = RR(1 + (k-1)\*slice\_length : k\* slice\_length);  end  flag\_tracking = 1;  sum\_len\_temp = 0; % 解扩数据长度累加器  for kk = 1 : 1 : slice\_temp  RR\_remainder = zeros(1,PN\_sps\_length); % 数据缓存器  RR\_quotient = 0; % 初始化  symbol\_num =0; % 初始化  resv\_i = 0;resv\_q = 0;len = 0;length\_temp = 0;  if flag\_tracking == 0  % 第二段及之后数据开始执行  if remainder(kk-1) % 有余数  % 缓存器添值  length\_increas = PN\_sps\_length - remainder(kk-1);  RR\_remainder(1:remainder(kk-1)) = RR\_slice(kk-1, end - remainder(kk-1) + 1 : end);  RR\_remainder(remainder(kk-1)+1:end) = RR\_slice(kk,1 : length\_increas); | length\_temp = length(RR\_slice(kk,:)) - length\_increas;  % 商和余数  quotient(kk) = floor( length\_temp/ PN\_sps\_length );  remainder(kk) = mod( length\_temp, PN\_sps\_length );  % 跟踪模块的输入数据及码元数  RR\_quotient = [RR\_remainder RR\_slice( kk , length\_increas+1 : end - remainder(kk))];  symbol\_num = quotient(kk) +1;  else  % 商和余数  quotient(kk) = floor( length(RR\_slice(kk,:))/ PN\_sps\_length );  remainder(kk) = mod( length(RR\_slice(kk,:)), PN\_sps\_length );  % 跟踪模块的输入数据及码元数  symbol\_num = quotient(kk);  RR\_quotient = RR\_slice(kk,1 : end - remainder(kk));  end  end % if flag\_tracking == 0  if flag\_tracking  % 第一段数据开始执行  flag\_tracking = 0;  % 商和余数  quotient(kk) = floor( length(RR\_slice(kk,:))/ PN\_sps\_length );  remainder(kk) = mod( length(RR\_slice(kk,:)), PN\_sps\_length );    % 跟踪模块的输入数据及码元数  symbol\_num = quotient(kk);  RR\_quotient = RR\_slice(kk,1 : end - remainder(kk));  end  resv\_i=real(RR\_quotient);  resv\_q=imag(RR\_quotient); |

上述m代码在matlab里面运行得到了想要的结果。现在需要把它修改成c代码。

#### 4.2.2 c代码修改

现在时间：20240630 17:10 周日

C代码早已经修改好。

## 5 扩频课代码编写记录

编写代码路径位于：

D:\MODIFI\_CODE\CDMA\扩频\DSSS\_self\Spread\_spectrum\_course文件夹里面。

编写代码参考资料位于：

D:\MODIFI\_CODE\CDMA\扩频\DSSS\_self\Spread\_spectrum\_course\reference 文件夹中。文件名为：C.S0002-0\_v1.0

### 5.1 信号生成

信号生成参考文件C.S0002-0\_v1.0中第235页流程图，如下图所示。目前编写只针对一个用户，也就是下图中上半部分。

|  |
| --- |
|  |

#### 5.1.1 参数设置

符号速率Rb = 50K Baud，仿真时间一秒钟。

扩频码使用长度为walsh\_len = 64的walsh码，然后扰码使用长度为scamble\_len = 1023的m序列。

一个码片采样sps = 5个点，采样率fs = sps \* Rc。其中，Rc = Rb \* walsh\_len。

#### 5.1.2 调制

调制选择QPSK调制，进行功率归一化操作。

如何进行功率归一化操作，参考CSDN博客：http://t.csdnimg.cn/8CxTR

#### 5.1.3 加扩

#### 5.1.4 加扰

#### 5.1.5 成型滤波

### 5.2 信道

### 5.3 信号解扩

### 5.4 信号解调