通信原理 实验报告

**实验报告撰写说明：**

1. **将以上学号班级姓名补全**
2. **实验报告文件命名中需将学号和姓名改为自己的学号和姓名**
3. **word转成pdf，**[**发到QQ群指定邮箱**](mailto:发到496179081@qq.com)
4. **严禁粘贴复制，若发现雷同，报告零分处理**

**3 接收端的同步处理**

## 一 实验内容（10分）

* 1. 接收端的同步处理

① 在前两次实验课的基础上，进一步理解通信系统的组成

② 深入理解“帧组同步、频偏估计和补偿、相偏估计和补偿” 的原理

③ 参考老师给的帧组同步的代码，学习其具体实现

④ 对帧组同步前后的星座图进行分析和理解

⑤ 参考老师给的频偏估计的代码，学习其具体实现

⑥ 补充频偏补偿的代码，并验证

⑦ 对频偏补偿前后的星座图进行分析和理解

⑧ 编写相偏估计和补偿的代码，并验证

⑨ 对相偏补偿前后的星座图进行分析和理解

## 二 实验原理（40分）

* 1. 接收端为什么需要同步？

在通信系统中，接收端同步是确保正确解调和恢复发送信号的关键步骤。同步主要包括载波同步、符号同步和帧同步。

1. **载波同步**：接收端需要与发送端的载波频率和相位保持一致。如果接收端的本地振荡器与发送端的载波频率或相位不一致，会导致解调信号失真，影响误码率。载波同步通过频率和相位估计与补偿来实现。
2. **符号同步**：接收端需要准确确定每个符号的起始和结束时间。符号同步错误会导致符号间干扰（ISI），增加误码率。符号同步通常通过定时恢复算法实现，如早迟门同步器。
3. **帧同步**：在分组或帧传输中，接收端需要识别帧的起始和结束位置。帧同步错误会导致数据解析错误，影响通信的可靠性。帧同步通常通过特定的同步字或前导码实现。

同步的缺失会导致信号失真、误码率上升，甚至通信中断。因此，同步是通信系统正常工作的基础。

* 1. 频偏估计和补偿的原理

频偏是指接收端与发送端载波频率的差异，主要由硬件不理想和 Doppler 效应引起。频偏会导致接收信号旋转，影响解调性能。

1. **频偏估计**：
   * **时域方法**：通过分析接收信号的相位变化估计频偏。例如，利用相邻符号的相位差计算频偏。
   * **频域方法**：通过FFT将信号转换到频域，寻找频谱峰值估计频偏。
2. **频偏补偿**：
   * **数字下变频**：通过数字信号处理调整本地振荡器频率，补偿频偏。
   * **相位旋转**：在基带信号上施加相反的相位旋转，抵消频偏影响。

频偏估计和补偿能有效提高系统性能，降低误码率。

* 1. 相偏估计和补偿的原理

相偏是指接收端与发送端载波相位的差异，主要由传播延迟和硬件不理想引起。相偏会导致接收信号相位旋转，影响解调性能。

1. **相偏估计**：
   * **导频辅助**：通过已知导频信号估计相偏。
   * **盲估计**：通过接收信号的统计特性估计相偏，如Viterbi-Viterbi算法。
2. **相偏补偿**：
   * **相位旋转**：在基带信号上施加相反的相位旋转，补偿相偏。
   * **锁相环（PLL）**：通过反馈控制调整本地振荡器相位，补偿相偏。

相偏估计和补偿能有效提高系统性能，降低误码率。

* 1. 帧组同步的原理

帧组同步是确保接收端正确识别帧起始和结束位置的过程，通常在分组或帧传输中使用。

1. **同步字或前导码**：发送端在帧起始处插入特定的同步字或前导码，接收端通过检测这些特定序列确定帧起始位置。
2. **相关检测**：接收端计算接收信号与已知同步字的相关性，当相关性达到峰值时，判定为帧起始位置。
3. **保护间隔**：在帧之间插入保护间隔，防止帧间干扰，确保帧同步的可靠性。
4. **错误检测和纠正**：通过CRC等机制检测和纠正帧同步错误，提高系统可靠性。

帧组同步能确保接收端正确解析数据，提高通信系统的可靠性。

## 三 具体实现（15分）

3.1 帧组同步的具体实现

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47 | function [out\_signal,cor\_abs,col,index\_s] = rx\_package\_search(rxdata,local\_sync,len\_frame,ratio)        down\_sig=reshape(rxdata,ratio,[]);% 假设为4倍过采样      [m,n]=size(down\_sig);      cor\_abs=zeros(m,n);      len\_window=8;      threshold=0.3;      l=0;      flag=false;      for j=1:ratio          signalo=down\_sig(j,:);          N=length(local\_sync);          for coarse\_i=N:n              nor\_sig=signalo(coarse\_i-N+1:coarse\_i)./ ...                  max(abs(signalo(coarse\_i-N+1:coarse\_i)));% 按最大的模长放缩              cor\_abs(j,coarse\_i)=abs(nor\_sig\*local\_sync')/N;              if (cor\_abs(j,coarse\_i)>threshold)||(flag==true)                  flag=true;                  l=l+1;                  if l>=len\_window                      break;                  end              end          end          if flag==true              break;          end      end      if flag==true          [row,col]=find(cor\_abs==max(max(cor\_abs(:,1:end/2))));% 保证索引不超出数组边界            figure(10);clf;          x=1:length(cor\_abs(row,:));          axis equal;          plot(x,cor\_abs(row,:));          grid on;          hold on;            index\_s=col-N+1;          index\_e=index\_s+len\_frame-1;          out\_signal=down\_sig(row,(index\_s:index\_e));      else          out\_signal=0;          cor\_abs=0;          col=0;          index\_s=0;  end |

函数文件 rx\_package\_search.m

[out\_signal, cor\_abs, col, index\_s] = rx\_package\_search(rxdata, local\_sync, len\_frame, ratio)

输入参数说明：

- rxdata：输入信号序列

- local\_sync：127位m序列作为训练序列

- len\_frame：输出序列长度（639 = 127 + 516 = 127 + 128\*4，即训练序列长度加传输数据长度）

- ratio：过采样倍数，设置为4

算法流程：

1. 数据预处理：将输入数据重组为4行矩阵，每行包含过采样点

2. 粗同步检测：

- 对每行数据应用127点窗口

- 计算与训练序列的相关值

- 当相关值超过阈值（0.3）时，标记flag为true，进入细定位阶段

3. 细定位：

- 使用8点窗口进行精确同步

- 确定最大相关值位置，完成同步

4. 输出处理：

- 若成功同步，输出同步头后639个数据点，并绘制相关值曲线

- 若未同步成功，返回空序列

3.2 频偏估计和补偿的具体实现

|  |  |
| --- | --- |
| 1  2 | fine\_sync\_seq=rx\_frame(1:120);  [deltaf,out\_signal3]=rx\_freq\_sync(rx\_frame,fine\_sync\_seq); |

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16 | function [freq\_offset,out\_signal] = rx\_freq\_sync(rxdata,training\_seq)    T\_sym=1/10e6;  len=length(rxdata);  N=length(training\_seq);    z\_k=training\_seq.^2;  r=zeros(1,N-1);  for k=1:N-1      r(k)=mean(z\_k(1+k:N).\*conj(z\_k(1:N-k)));  end    freq\_offset=angle(sum(r))/(2\*pi\*N\*T\_sym);  out\_signal=rxdata.\*exp(-1i\*2\*pi\*freq\_offset\*(1:len)\*T\_sym);    end |

在本方法中，我们选取了m序列的前120个比特作为训练序列。由于采用BPSK调制方式，调制阶数M=2。将训练序列training\_seq（即理论中的rk）进行M次方运算得到zk，随后应用L&R算法计算得到R值，进而估计出频率偏移量△w。最后，利用该估计值对输入序列进行频率偏移补偿。

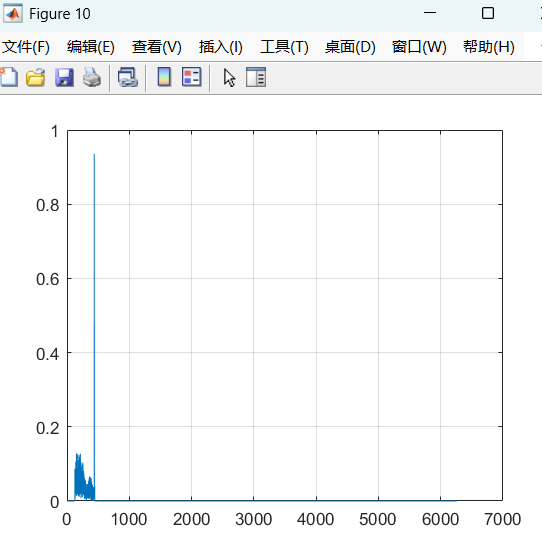
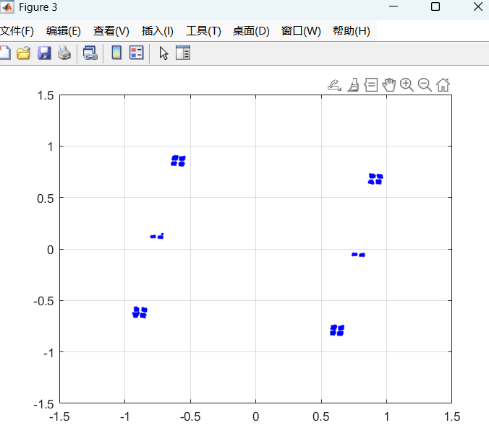
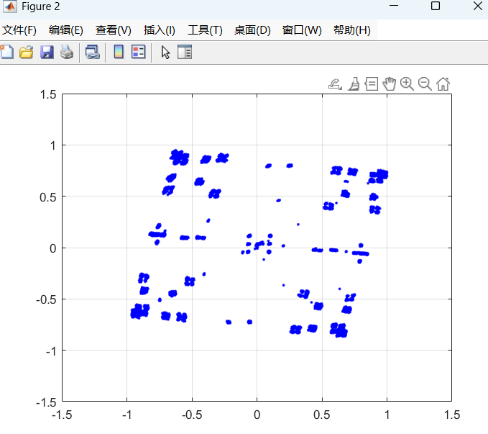
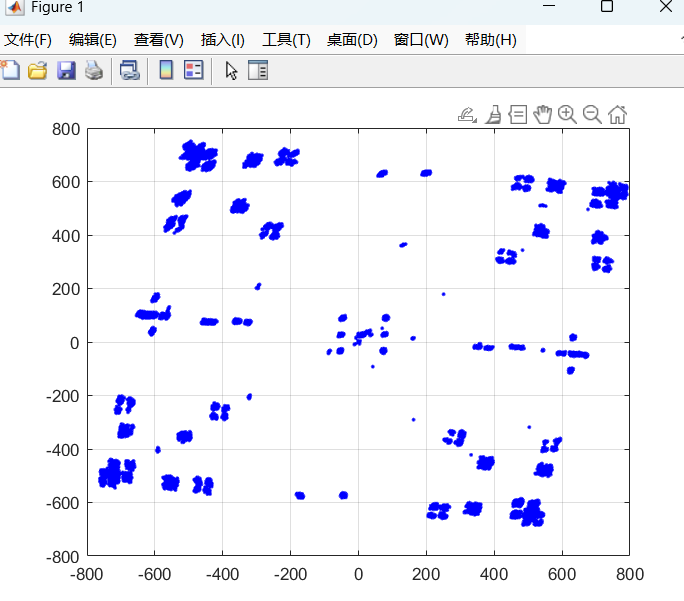
3.3 相偏估计和补偿的具体实现

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | function [out\_signal,ang] = rx\_phase\_sync(signal\_freq\_sync,local\_seq)      len=length(local\_seq);      L=len;      for i=1:L-1          cor(i)=signal\_freq\_sync(i).\*conj(local\_seq(L-i));      end      ang=angle(mean(cor))-pi;      out\_signal=signal\_freq\_sync.\*exp(-1i\*ang); |

通过对频率偏移进行校正后的序列，将其与本地序列的共轭进行乘积运算，进而计算出相位偏移的平均值，并据此对相位偏移进行补偿。

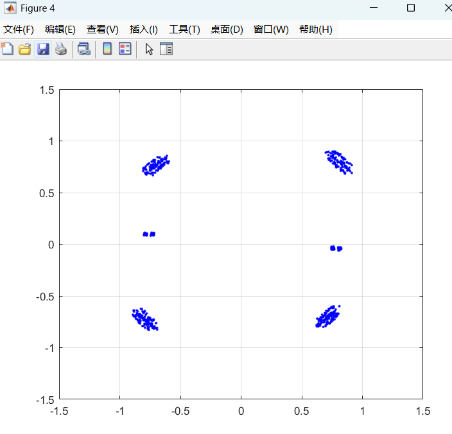
## 四 实验结果图示及分析（15分）

* 1. 帧组同步前后星座图分析



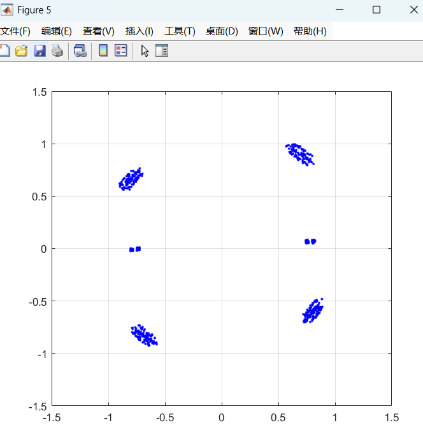
一旦实现同步，数据点将呈现出集中的分布状态。在未同步的情况下，码元的对应关系无法确定，而同步过程则有效地解决了这一难题。尽管如此，数据分布仍存在不对称性和偏差，这主要归因于频率偏移和相位偏移的误差。通过观察相关值的变化，可以发现在某一特定位置相关值达到峰值，此时即表明同步已完成。

* 1. 频偏估计前后星座图分析



相较于前面，数据偏差已经减小了很多，这是用了频偏补偿的效果，但是仍然还有偏移，这是相偏的原因。

* 1. 相偏估计前后星座图分析



星座图已经对称，消除了相偏对它相位的影响。

## 五 总结和思考（20分）

5.1 实验过程中遇到的问题及解决方法（4分）

问题：通信原理基础较差，从而导致对于代码的理解有问题

解决方法：现场翻通信原理的PPT。

5.2 帧组同步一定要在频偏估计之前吗？为什么？（8分）

在通信系统中，帧组同步和频偏估计是两个关键步骤，它们的顺序对系统性能有重要影响。通常，帧组同步需要在频偏估计之前进行，主要原因如下：

1. **同步是频偏估计的前提**：帧组同步的目的是确定数据帧的起始位置，确保接收端能够正确解析数据。如果未完成同步，接收端无法准确识别数据帧，频偏估计也就失去了意义，因为频偏估计依赖于已知的帧结构。
2. **频偏估计依赖同步信号**：频偏估计通常利用已知的同步序列（如导频或训练序列）进行计算。如果帧组同步未完成，接收端无法定位这些序列，频偏估计将无法进行。
3. **同步误差影响频偏估计精度**：如果先进行频偏估计而未完成同步，频偏估计可能会受到同步误差的干扰，导致估计不准确。同步误差会引入额外的相位噪声，进一步影响频偏估计的精度。
4. **系统设计的逻辑顺序**：从系统设计角度看，帧组同步是更基础的步骤，只有在完成同步后，才能进行后续的信号处理，如频偏估计、信道均衡等。这种顺序符合信号处理的逻辑流程。

因此，帧组同步通常需要在频偏估计之前完成，以确保频偏估计的准确性和系统的整体性能。

5.3 含频偏和相偏的M序列，自相关性会受影响吗？（8分）

M序列（最大长度序列）是一种伪随机序列，具有良好的自相关特性，常用于同步和信道估计。然而，当M序列受到频偏和相偏的影响时，其自相关性会发生变化，具体表现如下：

1. **频偏的影响**：频偏会导致接收信号的频率与本地M序列的频率不一致，从而在自相关计算中引入相位旋转。这种相位旋转会降低自相关峰值的幅度，导致自相关函数的峰值减弱，旁瓣电平升高，进而影响同步和信道估计的精度。
2. **相偏的影响**：相偏会导致接收信号与本地M序列之间存在固定的相位差。虽然相偏不会改变自相关峰值的幅度，但会引入一个固定的相位偏移，这可能影响某些依赖于相位信息的应用场景。
3. **频偏和相偏的联合影响**：当频偏和相偏同时存在时，自相关函数的峰值会进一步减弱，旁瓣电平升高，且可能出现多个伪峰值。这会显著降低M序列的自相关性能，增加同步和信道估计的难度。
4. **解决方法**：为了减轻频偏和相偏对M序列自相关性的影响，通常需要在接收端进行频偏和相偏的估计与补偿。例如，可以通过频偏估计和相位旋转校正来消除频偏和相偏的影响，从而恢复M序列的良好自相关特性。

综上所述，频偏和相偏会显著影响M序列的自相关性，降低其在同步和信道估计中的性能。因此，在实际系统中，通常需要采取频偏和相偏补偿措施，以恢复M序列的理想特性。