北 京 科 技 大 学

课程设计（通信系统）报告

专业班级：_____通信 1902_____

学　　号：_____41924275_____

姓　　名：_____成江波_____

成　　绩：_____

2022 年 6 月 6 日

目　　录

1 窄带 MUSIC 实现方向估计 DOA 估计算法

1.1 基本原理

1.窄带信号：频带范围△f 远小于中心频率 fc，且 fc 远离零频率的信号为窄带信号。

2.DOA 估计：波达方向（Direction-of- Arrival, DOA）估计指的是要确定同时处在空间某一区域内多个感兴趣信号的空间位置，即各个信号到达阵列参考阵元的方向角。DOA 估计也称空间谱估计。一个信源有很多可能的传播路径和到达角。如果几个发射源同时工作，每个信源在接收机处形成潜在的多径分量。因此，接收天线能估计出这些到达角就显得很重要，目的是估计出哪个发射源在工作以及发射源所处的方向。

3.MUSIC 算法原理：MUSIC 算法是一种基于矩阵特征空间分解的方法，从几何角度讲，信号处理的观测空间可以分解为正交的信号子空间和噪声子空间。信号子空间由阵列接收到的数据协方差矩阵中与信号对应的特征向量组成，噪声子空间则由协方差矩阵中所有最小特征值（噪声方差）对应的特征向量组成。MUSIC 算法是基于特征结构分析的空间谱估计方法，是空间谱估计技术的典型代表。其测向原理是根据矩阵特征分解的理论，对阵列输出协方差矩阵进行特征分解，将信号空间分解为噪声子空间 G 和信号子空间 S，利用噪声子空间 G 与阵列的方向矩阵 A 的列矢量正交的性质，构造空间谱函数 P(w)并进行谱峰搜索，从而估计出波达方向信息。

1.2 设计步骤

本实验利用 MATLAB 软件进行 MUSIC 算法估计 DOA 。具体步骤如下：

1.设置初始条件，包括阵元个数，信源个数，DOA，快拍数，阵元间距等参数。

2.生成信号，模拟生成带有白色高斯噪声的窄带信号。

3.估计计算接收信号的协方差矩阵。

4.将协方差矩阵进行特征值分解，获得噪声子空间和信号子空间

5.估计 DOA，遍历空间中的每个角度，构造空间谱函数，进行谱峰搜索，估计出波达信息。

1.3 仿真实现

1. 设置初始条件

```
clear all; close all; %delete(findall(0, 'Type', 'figure'));
% STEP a: Simulating the Narrowband Sources %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
p = [100,200,300];    % Number of time snapshots
fs = 10^7;  % Sampling frequency
fc = 10^6;  % Center frequency of narrowband sources
M = [10,20,30];     % Number of array elements, i.e., sensors or antennas
N = 5;      % Number of sources
sVar = 1;   % Variance of the amplitude of the sources
% p snapshots of N narrowband sources with random amplitude
% of mean zero and covariance 1
s = sqrt(sVar)*randn(N, p(1)).*exp(1i*(2*pi*fc*repmat([1:p(1)]/fs, N, 1)));
% STEP a: Simulating the Narrowband Sources %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% STEP b: Mixing the sources and getting the sensor signals %%%%%%%%%%%%%%
doa = [20; 50; 85; 110; 145];  %DOAs
cSpeed = 3*10^8 ;              % Speed of light
dist = [150,200,250];                    % Sensors (i.e., antennas)
% spacing in meters
```

图 1-1 设置初始条件代码

2. 模拟生成信号

```
% Constructing the Steering matrix
for index=1:length(M)
    A = zeros(M(index), N);
for k = 1:N
A(:, k) = exp(-1i*2*pi*fc*dist(1)*cosd(doa(k))*(1/cSpeed)*[0:M(index)-1]');
end
noiseCoeff = 1;                          % Variance of added noise
x = A*s + sqrt(noiseCoeff)*randn(M(index), p(1)); % Sensor signals
```

图 1-2 模拟生成信号

3. 计算协方差矩阵，并进行特征值分解

```
R = (x*x')/p(i); % Empirical covariance of the antenna data
[V, D] = eig(R);
noiseSub = V(:, 1:M(1)-N); % Noise subspace of R
```

图 1-3 特征值分解得到噪声子空间

4. 通过谱峰搜索，估计出波达信息

```
theta = 0:1:180;        %Peak search
a = zeros(M(index), length(theta));
res = zeros(length(theta), 1);
for i = 1:length(theta)
    a(:, i) = exp(-1i*2*pi*fc*dist(1)*cosd(i)*(1/cSpeed)*[0:M(index)-1]');
    res(i, 1) = 1/(norm(a(:, i)'*noiseSub).^2);
end
[resSorted, orgInd] = sort(res, 'descend');
DOAs = orgInd(1:N, 1);
```

图 1-4 DOA 估计

5. 利用 plot 函数，画出仿真结果图。

1.4 结果及分析

1.MUSIC 算法估计 DOA 的仿真结果。

阵元数为 10，快拍数为 100，阵元间距为 150，噪声系数为 1 的 DOA 估计。初始条件为 DOA=[20,50,85,110,145].

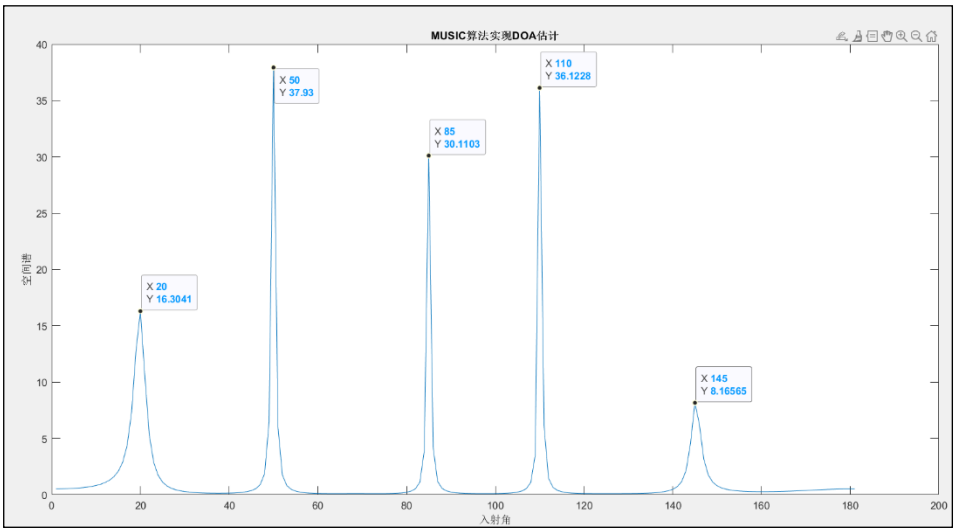

图 1-5 MUSIC 算法实现 DOA 估计仿真结果图

```
DOAs =

   145
   110
    85
    50
    20
```

图 1-6 MUSIC 估计出的 DOA 角度

结果分析：从仿真结果可以看出，本实验能够实现首先给定 DOA，之后用 MUSIC 算法进行估计，最后得出方向角度，并且估计得出的角度和事先给定的 DOA 能够很好的吻合，但在实验中并不是每次都得到这样非常好的结果，有时会有一些角度的误差，对 MUSIC 算法效果的影响的因素有阵元数、快拍数、阵元间距等等。

2.阵元数对 MUSIC 算法的影响

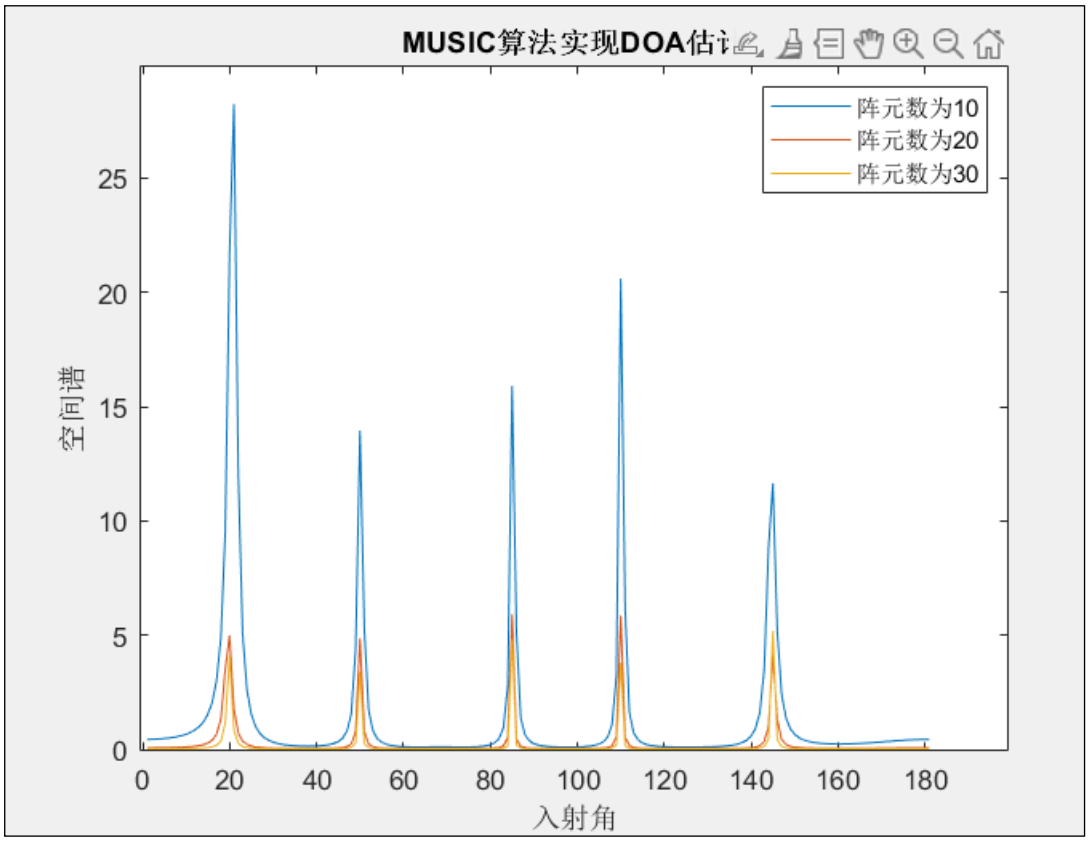

图 1-7 阵元数对 MUSIC 算法的影响

结果分析：随着阵元数的增加，DOA 估计谱波形的波束宽度变窄，阵列的指向性变好，阵列分辨空间信号的能力增强。由此可得，如果要得到更加精确的 DOA 估计谱，可以增加阵元数量。但阵元数量越多就需要处理的数据越多，运算量越大，运行速度会越慢，并且由上图可以看出阵元数为 20 和 30 时，波形变化不会很明显。因此，在实际应用中可根据具体条件适当选取阵元数量，在确保估计谱准确的前提下，尽量减少资源浪费，提高效率。

3. 快拍数对 MUSIC 算法的影响

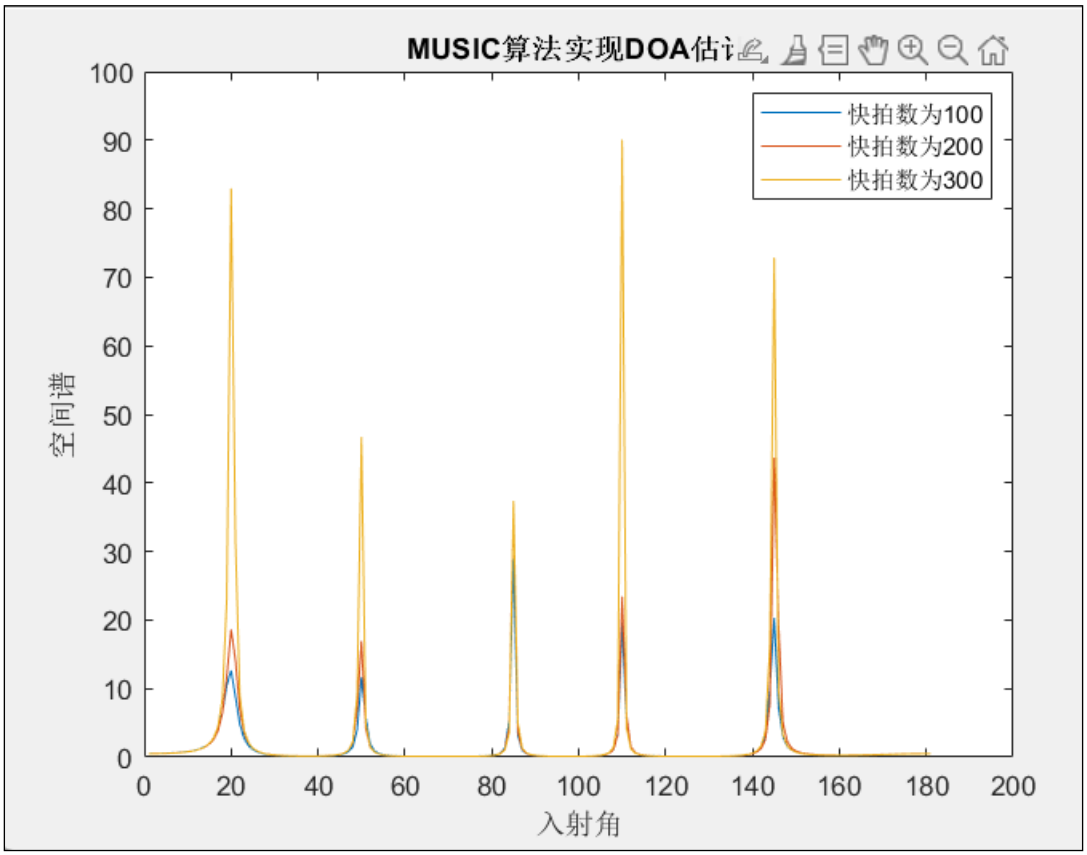

图 1-8 快拍数对 MUSIC 算法的影响

结果分析：随着快拍数的增加，DOA 估计谱的波束宽度也会变窄，阵列的指向性变好，阵列分辨空间信号的能力增强，MUSIC 算法的估计精度增加。但与阵元数的影响类似，过多的快拍数也会加大运算量，造成运算时间加长，降低效率。

4.阵元间距对 MUSIC 算法的影响

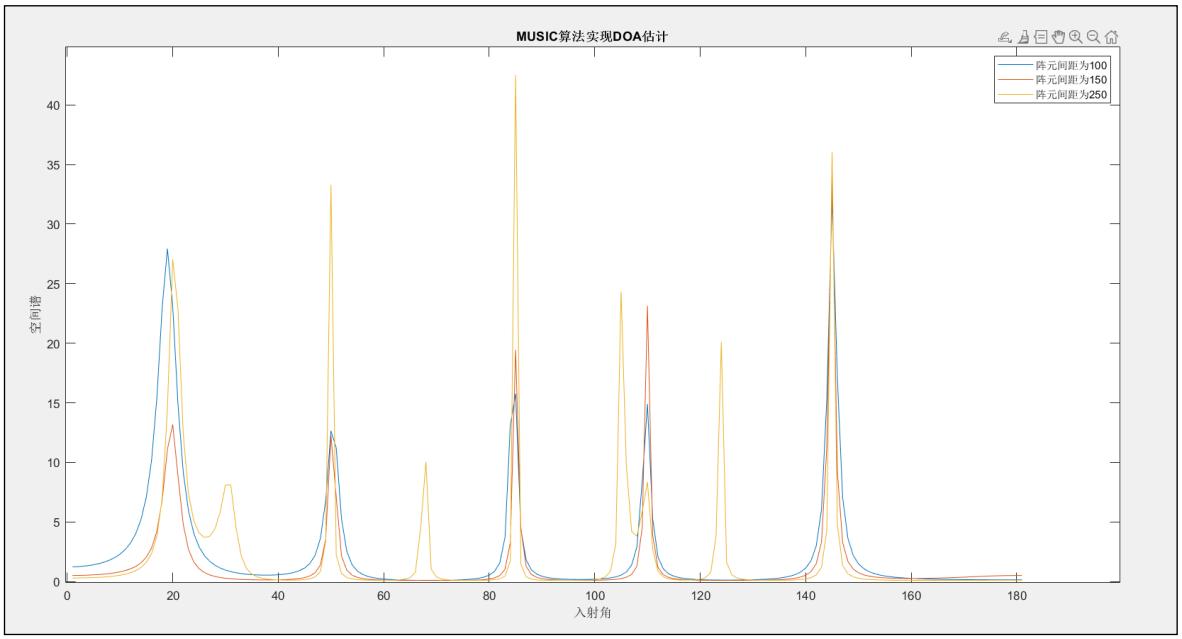

图 1-9 阵元间距对 MUSIC 算法的影响

结果分析：由仿真结果可以看出，当阵元间距不大于半波长时，DOA 的波束宽度随着间距的增加而变窄，阵列的指向性变优，但阵元间距大于半波长时，估计谱除了信号源方向外在其他方向出现了虚假谱峰，这意味着估计的准确性大大降低，因此在应用中阵元间距一定不能超过半波长。

ISI 信道下 Viterbi 译码

2.1 基本原理

1.卷积编码：卷积码是一种性能优越的信道编码，它的编码器和解码器都比较易于实现，同时还具有较强的纠错能力，这使得它的使用越来越广泛。卷积码一般表示为(n, k, K)的形式，即将 k 各信息比特编码为 n 个比特的码组，K 为编码约束长度，说明编码过程中相互约束的码段个数。卷积码编码后的 n 各码元不经与当前组的 k 个信息比特有关，还与前 K-1 个输入组的信息比特有关。编码过程中相互关联的码元有 K*n 个。R=k/n 是编码效率。编码效率和约束长度是衡量卷积码的两个重要参数.典型的卷积码一般选 nk 较小,但 K 值可取较大(>10),以获得简单而高性能的卷积码。

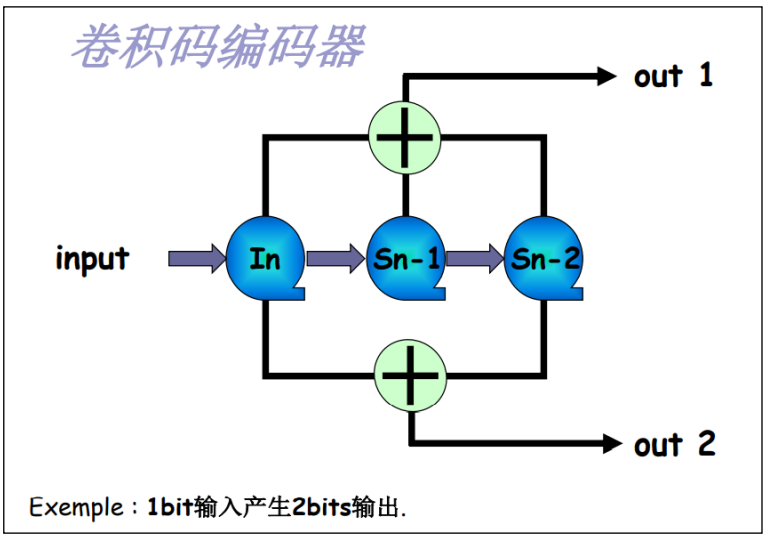

图 2-1 卷积编码器

上图中卷积编码器 1 位输入产生两位输出，并且上边输出与当前输入、当前输入前一、二位的输入有关，下边输出与当前输入、当前输入的前二位输入有关，以此类推，可得卷积编码器的输出与所有的输入有关。

2．ISI 信道：ISI 信道是线性时变滤波信道，通常是用于信道特性不理想或者是发生时变多径传播造成被传输信号的幅度和相位都有失真，这类物理信道在数学上可以表征为时变线性滤波器。

3．码间串扰（ISI）：信道总是带限的，带限信道对通过的的脉冲波形进行拓展。当信道带宽远大于脉冲带宽时，脉冲的拓展很小，当信道带宽接近于信号的带宽时，拓展将会超过一个码元周期，造成信号脉冲的重叠，称为码间串扰。

4．Viterbi 译码：卷积码概率译码的基本思路是:以接收码流为基础，逐个计算它与其他所有可能出现的、连续的网格图路径的距离，选出其中可能性最大的一条作为译码估值输出。概率最大在大多数场合可解释为距离最小，这种最小距离译码体现的正是最大似然的准则。卷积码的最大似然译码与分组码的最大似然译码在原理上是一样的,但实现方法上略有不同。主要区别在于:分组码是孤立地求解单个码组的相似度，而卷积码是求码字序列之间的相似度。基于网格图搜索的译码是实现最大似然判决的重要方法和途径。用格图描述时，由于路径的汇聚消除了树状图中的多余度，译码过程中只需考虑整个路径集合中那些使似然函数最大的路径。如果在某一点上发现某条路径已不可能获得最大对数似然函数，就放弃这条路径，然后在剩下的“幸存"路径中重新选择路径。这样一直进行到最后第 L 级(L 为发送序列的长度)。由于这种方法较早地丢弃了那些不可能的路径，从而减轻了译码的工作量，Viterbi 译码正是基于这种想法。对于(n, k,K)卷积码，其网格图中共 2kL 种状态。由网格图的前 K-1 条连续支路构成的路径互不相交，即最初 2k-1 条路径各不相同，当接收到第 K 条支路时，每条路径都有 2 条支路延仲到第 K 级上，而第 K 级上的每两条支路又都汇聚在一个节点上。在 Viterbi 译码算法中，把汇聚在每个节点上的两条路径的对数似然函数累加值进行比较，然后把具有较大对数似然函数累加值的路径保存下来，而丢弃另一条路径，经挑选后第 K 级只留下 2K 条幸存路径。选出的路径同它们的对数似然函数的累加值将一起被存储起来。由于每个节点引出两条支路，因此以后各级中路径的延伸都增大一倍，但比较它们的似然函数累加值后，丢弃一半，结果留存下来的路径总数保持常数。由此可见，上述译码过程中的基本操作是，“加-比-选”，即每级求出对数似然函数的累加值，然后两两比较后作出选择。有时会出现两条路径的对数似然函数累加值相等的情形，在这种情况下可以任选择其中一条作为"幸存"路径。卷积码的编码器从全零状态出发，最后又回到全零状态时所输出的码序列,称为结尾卷积码。因此，当序列发送完毕后，要在网格图的终结处加上(K-1)个已知的信息作为结束信息。在结束信息到来时，由于每一状态中只有与已知发送信息相符的那条支路被延伸，因而在每级比较后，幸存路径减少一半。因此，在接收到（K-1）个已知信息后，在整个网格图中就只有唯一的一条幸存路径保留下来，这就是译码所得的路径。也就是说，在已知接收到的序列的情况下，这条译码路径和发送序列是最相似的。

2.2 设计步骤

1. 进行卷积编码，运用课上所学的卷积编码器的原理进行比特流的编码。

2.模拟生成 ISI 信道，接受卷积编码后的比特流。

3.运用课上学习的 Viterbi 译码的原理，进行比特流的 Viterbi 译码。

4.译码输出后，计算误比特率并进行结果分析。

2.3 仿真实现

1.卷积编码

```
% 比特流生成
number=10^4;
bit_stream=[randi([0 1],number,1);[0;0]];
codeword=zeros(2*(number+2),1); %卷积码序列
D1=0; % 寄存器初始状态
D2=0;
for i_b=1:1:number+2
y1=mod(bit_stream(i_b)+D1+D2,2);
y2=mod(bit_stream(i_b)+D2,2); %计算每个输入对应的两个输出
D2=D1;
D1=bit_stream(i_b); %开始移位
codeword(2*i_b-1)=y1;
codeword(2*i_b)=y2; %以y1y2的顺序存储进Codeword
end
```

图 2-2 卷积编码

2. 模拟生成 ISI 信道并接受比特流

```
%ISI信道
data_len = 2*(number+2); % 数据流长度
SNR_dB = 2; % 单位比特信噪比
chan_len = 3; % 通道抽头数
fade_var = 1; % 信道的衰减方差
decoding_delay = 10; % 维特比算法的译码延迟
SNR = 10^(0.1*SNR_dB);
noise_var = 1*(fade_var*chan_len)/(2*SNR); % 信噪比参数
chan_input = codeword'; % 信道输入,将卷积码作为信道输入
bpsk_seq = 1-2*chan_input; % bpsk
fade_chan = normrnd(0,sqrt(fade_var),1,chan_len);% ISI信道的频率响应脉冲
noise = normrnd(0,sqrt(noise_var),1,data_len+chan_len-1);% awgn
chan_op = conv(fade_chan,bpsk_seq)+noise; % 信道输出
```

图 2-3 ISI 信道生成

3. Viterbi 译码，并计算误比特率

```
%viterbi译码
steady_state = chan_op(chan_len:data_len);
% branch metrics for the Viterbi algorithm
branch_metric = zeros(2^chan_len,data_len-chan_len+1);
branch_metric(1,:) = (steady_state-(fade_chan(1)+fade_chan(2)+fade_chan(3))).^2;
branch_metric(2,:) = (steady_state-(fade_chan(1)+fade_chan(2)-fade_chan(3))).^2;
branch_metric(3,:) = (steady_state-(fade_chan(1)-fade_chan(2)+fade_chan(3))).^2;
branch_metric(4,:) = (steady_state-(fade_chan(1)-fade_chan(2)-fade_chan(3))).^2;
branch_metric(5,:) = (steady_state-(-fade_chan(1)+fade_chan(2)+fade_chan(3))).^2;
branch_metric(6,:) = (steady_state-(-fade_chan(1)+fade_chan(2)-fade_chan(3))).^2;
branch_metric(7,:) = (steady_state-(-fade_chan(1)-fade_chan(2)+fade_chan(3))).^2;
branch_metric(8,:) = (steady_state-(-fade_chan(1)-fade_chan(2)-fade_chan(3))).^2;
%-------------------------------------------------------------------------------
dec_a=Viterbi_algorithm(data_len-chan_len+1,decoding_delay,branch_metric);
% 误比特率
ber = nnz(chan_input(chan_len:data_len-decoding_delay)-dec_a )/(data_len-chan_len+1-decoding_delay)
```

图 2-4 Viterbi 译码

```
% Viterbi Algorithm - Soft Input, Hard Output
function[dec_ip]=Viterbi_algorithm(num_bit,decoding_delay,branch_metric)
[Prev_State,Prev_State_trans,~,Prev_Ip_trans,Outputs_prev]= Get_Trellis();
num_states = 4;  % number of states in the trellis
% General Initialization
ip=0; % input initialization
dec_ip = zeros(1,num_bit-decoding_delay); % decoded symbols (-decoding delay is because,ignoring last transient part)
survivor_node = zeros(num_states,num_bit); % survivor nodes
survivor_ip = zeros(num_states,num_bit); % survivor inputs
path_metric = zeros(num_states,num_bit+1); % path metrics
index_temp = [0;1*2;2*2;3*2]; %for linear indexing.
for sym_cnt=  1:num_bit
   [path_metric(:,sym_cnt+1),index] = min([path_metric(Prev_State(:,1),sym_cnt)+ branch_metric(Outputs_prev(:,1),sym_cnt)
       path_metric(Prev_State(:,2),sym_cnt)+ branch_metric(Outputs_prev(:,2),sym_cnt)],[],2);
   survivor_node(:,sym_cnt) = Prev_State_trans(index+index_temp);
   survivor_ip(:,sym_cnt) = Prev_Ip_trans(index+index_temp);
   %      Back tracing
if (sym_cnt>decoding_delay)
[~,trace_bf] = min(path_metric(:,sym_cnt+1));
for bk_cnt= 1 : decoding_delay+1
ip = survivor_ip(trace_bf,sym_cnt+1-bk_cnt);
trace_bf = survivor_node(trace_bf,sym_cnt+1-bk_cnt);
end
dec_ip(sym_cnt-decoding_delay)=ip;
end % for if statement
end % for forward recursion
dec_ip = dec_ip-1; % decoded bits ( now in 0 and 1)
end % for function
```

图 2-5 Viterbi 译码原理代码

2.4 结果与分析

1. 误比特率结果

```
ber =

    0.1156

>> main

ber =

    0.1099

>> main

ber =

    0.1535

>> main

ber =

    0.0417

>> main

ber =

    0.1619
```

图 2-6 误比特率输出

2.结果分析

通过对 10000 个比特进行卷积编码并通过 ISI 信道后进行 Viterbi 译码，得到误比特率。从结果来看，当信噪比、抽头数量等环境因素不变时，不同的比特码经过相同的过程得到的误比特率有一定的差异，不过误比特率都在 20%以下，这可能是每次噪声与码间干扰不同所致。并且经过理论分析，实验中用到的 Viterbi 译码为软判决方式，与硬判决性比，硬判决判断的是汉明距离，而软判决用的大多数为欧氏距离，并以欧氏距离作为选择路径的标准，理论上比硬判决有着 3dB 的优势，尤其是在现在通信环境下，软判决译码比硬判决译码有着相当大的优势。

3 思考与总结

3.1 思考题

（1）**解释 Music 算法和 Root Music 算法的不同。**

MUSIC 算法中，构建空间谱函数后，需要进行谱峰搜索得到空间估计，结果的精度和扫描的精度直接相关，但增大扫描精度会增加运算量，即用运算量换取精度，而 ROOT MUSIC 算法是在 MUSIC 算法的基础上，不进行谱峰搜索，而是将谱峰的搜索问题转化为多项式求根的问题，从而去除算法运算量和精度之间的制约关系，具体来说是将e^{jw}看作复数 z，得到$a^H(w)GG^Ha(w) = a^H(z)GG^Ha(z) = 0, z^k = e^{jw_k}$.k=1,⋯,K,就是方程的根，信号频率的估计由搜索/遍历问题转化成了一元高次方程的求根问题。

（2）**解释 Viterbi 译码的核心思想。**

最大似然译码的基本想法是把已接收序列与所有可能的发送序列做比较，选择其中码距最小的一个序列做为发送序列。如果发送 L 组信息比特对于(n，k) 卷积码来说，网格图上可能发送的序列有2^{kL} 个，译码器需存储这些序列并进行比较，以找到码距最小的那个序列。当传信率和信息组数 L 较大时，使得译码器难以实现。Viterbi 算法则对上述最大似然解码做了简化，成为了一种实用化的最大似然译码算法。它不是在网格图上一次性比较所有可能的2^{kL} 条路径（序列），而是接收一段，计算和比较一段，选择一段有最大似然可能的码段，从而达到整个码序列是一个有最大似然值的序列。译码过程类似于动态规划的过程。

Viterbi 译码过程：在当前时间 t，对每个状态进行：

1.计算分支度量 BM：即计算进入该状态的两个路径的分支度量(BM)，即接收到的码元与两个路径对应的输出码元之间的汉明距离(硬判决)或欧几里得距离(软判决)。

2.计算路径度量 PM：把两个路径的 BM 同各自对应的 t-1 时刻状态所存储的状态度量进行求和，得到 t 时刻的两个路径度量 PM。

3.选取幸存路径： 比较两个路径度量 PM，选取并保留最小的作为 t 时刻的状态度量，同时保留与之对应的路径作为形成路径。如两个相等，则任选其一作为幸存路径。

4.回溯(Traceback)： 从各个状态的形成路径中选取出状态度量最小的一条，顺次向前进行回溯，并输出译码结果。其中第 2 步第 3 步是译码的核心步骤。

（3）**说明多径和频率选择性的相关性。**

多路信号到达接收机的时间有先有后，即有相对时间延迟。如果这些相对时延远小于一个符号的时间，则可以认为多路信号几乎是同时到达接收机的，这种情况下多径不会造成符号间的干扰。这种衰落称为平坦衰落，因为这种信道的频率响应在所用的频段内是平坦的。信号的频率选择会引起符号间的干扰。

（4）**解释宽带信号和窄带信号的物理意义**。

“宽带”信号和“窄带”信号实质上是一个相对的概念。比如说，10MHz 带宽的接收机在雷达系统中被认为是窄带接收机，然而，达到这种带宽的接收机在通信中却被常常被认为是宽带接收机。这是因为应用的场合不同，造成从信号的带宽的具体数值上很难区分出“宽带”和“窄带”的概念。目前主流的观点认为：宽带信号与窄带信号是根据信号的带宽与载频的比值大小进行区分的。如果信号的带宽和载频的比值不大于 10%的时候，该信号会被认定是窄带信号，若信号的带宽和载频的比值超过百分之十的时候，就可以称之为“宽带”信号了。

3.2 总结

经过对课程设计的学习，我收获很多。通过对实验一的实践，我对 MUSIC 算法有了一定的了解，MUSIC 算法解决的是设法估计出入射到阵列的空间信号的个数 D 以及空间信号源的强度及其来波方向。MUSIC 算法的原理是根据信号的矩阵模型，进行协方差分解得到两组相互正交的信号子空间和噪声子空间，之后构建空间谱进行谱峰搜索得到估计的角度值。并且如果想要提高估计的准确性和效率，可以通过增加阵元数、快拍数、阵元间距来实现，但也不能增加得过多，否则会增加计算量，反而加大了运算时间。同时，当存在信号源之间是相干或相关时，MUSIC 算法的估计性能下降，可见 MUSIC 算法还有很大的发展空间，值得以后我们进一步探索。

通过对实验二 Viterbi 译码的学习，我首先学习了卷积编码，它是一位输入两位输出，并且每位的输出与其他所有输入关系，接着又学习了 ISI 信道，即为有码间干扰的

信道，最后是 Viterbi 译码的原理，核心是逐段比较，舍弃不可能的路径，最终找到一条路径最短、概率最大的路径。

经过课程设计的学习，我感觉不仅在知识上有了新的认知，更增加了我对通信领域的兴趣，并且提高了我查阅资料，对新知识自学的能力。虽然经过课程学习，我对其中的原理有了一定的了解，但在原理转化成代码过程中有一些的困难，并不能熟练的编写代码，在未来对代码的编写能力还有待提高，并且对通信要需要进一步的研究。

本人签名：成江波

2022 年　　6 月　　6 日

指导教师意见

指导教师签字：

年　　月　　日