专业：___信息工程_____
姓名：___袁东琦_____
学号：___3200103602___
日期：___2022/11/7_____

浙江大学实验报告

课程名称：__矩阵论__指导老师：__程磊__实验名称：__MUSIC 算法的实现__

一、MUSIC 算法原理

MUSIC 算法是最先提出来的一种超分辨率的用于目标估计的算法。其基本思想是通过将阵列输出信号的协方差矩阵进行特征值分解，从而得到信号子空间和噪声子空间，同时利用两个子空间的正交性来构造空间谱函数，通过谱峰搜索来估计信号的波达方向。

MUSIC 算法原理介绍如下：

阵列接收信号为

$$\begin{cases} x_1(t) = s(t)e^{j\omega t} + w_1(t) \\ x_2(t) = s(t)e^{j\omega t}e^{j\frac{2\pi}{\lambda}d\sin\theta} + w_2(t) \\ \cdots \\ x_8(t) = s(t)e^{j\omega t}e^{j\frac{2\pi}{\lambda}(8-1)d\sin\theta} + w_8(t) \end{cases} \tag{1}$$

$$\begin{aligned} \mathbf{X}(t) &= [x_1(t), x_2(t), \ldots, x_8(t)]^T \\ &= s(t)e^{j\omega t}\begin{bmatrix} 1 \\ e^{j\frac{2\pi}{\lambda}d\sin\theta} \\ \cdots \\ e^{j\frac{2\pi}{\lambda}(8-1)d\sin\theta} \end{bmatrix} + \begin{bmatrix} w_1(t) \\ w_2(t) \\ \cdots \\ w_8(t) \end{bmatrix} \\ &= s(t)\mathbf{a}(\theta) + \mathbf{W}(t) \end{aligned} \tag{2}$$

由阵列的协方差矩阵为

$$\mathbf{R} = E\left[\mathbf{X}(t)\mathbf{X}^{\mathrm{H}}(t)\right] \tag{3}$$

阵列的协方差矩阵 R 经过特征值分解后如下：

$$\mathbf{R} = \mathbf{U}_s\mathbf{\Sigma}_s\mathbf{U}_s^H + \mathbf{U}_{\mathrm{W}}\mathbf{\Sigma}_{\mathrm{W}}\mathbf{U}_{\mathrm{W}}^H \tag{4}$$

其中U_S是较大特征值对应的特征向量张成的子空间，为信号子空间。U_N是由较小特征值对应的特征向量张成的子空间，为噪声子空间。根据两子空间相互正交，则可得：

$$\mathbf{a}^H(\theta)\mathbf{U}_{\mathrm{W}} = \mathbf{0} \tag{5}$$

则可构造谱函数如下：

$$P(\theta) = \frac{1}{\mathbf{a}^H(\theta)\mathbf{U}_{\mathrm{W}}\mathbf{U}_{\mathrm{W}}^H\mathbf{a}(\theta)} \tag{6}$$

通过对θ进行扫描，就可以通过上式找到谱函数中的峰值，即可找到信号的到达角，即进行最小化搜索：

$$\hat{\theta} = \arg\min_{\theta} \mathbf{a}^{H}(\theta)\mathbf{U}_{W}\mathbf{U}_{W}^{H}\mathbf{a}(\theta) \quad (7)$$

二、推导过程

$R = E[X(t)X^H(t)] = E[(a(\theta)S(t)+W(t))(a(\theta)S(t)+W(t))^H]$

$= a(\theta)E[S(t)S^H(t)]a^H(\theta) + E[W(t)W^H(t)] = a(\theta)E[S(t)S^H(t)]a^H(\theta) + \Sigma_W$

$\because R = U\Sigma U^H = U_S\Sigma_S U_S^H + U_W\Sigma_W U_W^H \qquad U = [U_S\ U_W]$

$\therefore RU_W = [U_S\ U_W]\begin{bmatrix}\Sigma_S & \\ & \Sigma_W\end{bmatrix}\begin{bmatrix}U_S^H \\ U_W^H\end{bmatrix}U_W = [U_S\ U_W]\begin{bmatrix}\Sigma_S & \\ & \Sigma_W\end{bmatrix}\begin{bmatrix}0 \\ 1\end{bmatrix} = \Sigma_W U_W$

（$\because U_S$与U_W正交，$\therefore U_S U_W = 0$；$U_W U_W^H = 1$　$U_S U_S^H = 1$）

$\therefore RU_W = a(\theta)E[S(t)S^H(t)]a^H(\theta)U_W + \Sigma_W U_W = \Sigma_W U_W$

$\Rightarrow a(\theta)E[S(t)S^H(t)]a^H(\theta)U_W = 0 \Rightarrow U_W^H a(\theta)E[S(t)S^H(t)]a^H(\theta)U_W = 0$

即$(a^H(\theta)U_W)^T E[S(t)S^H(t)](a^H(\theta)U_W) = 0 \Rightarrow a^H(\theta)U_W = 0$

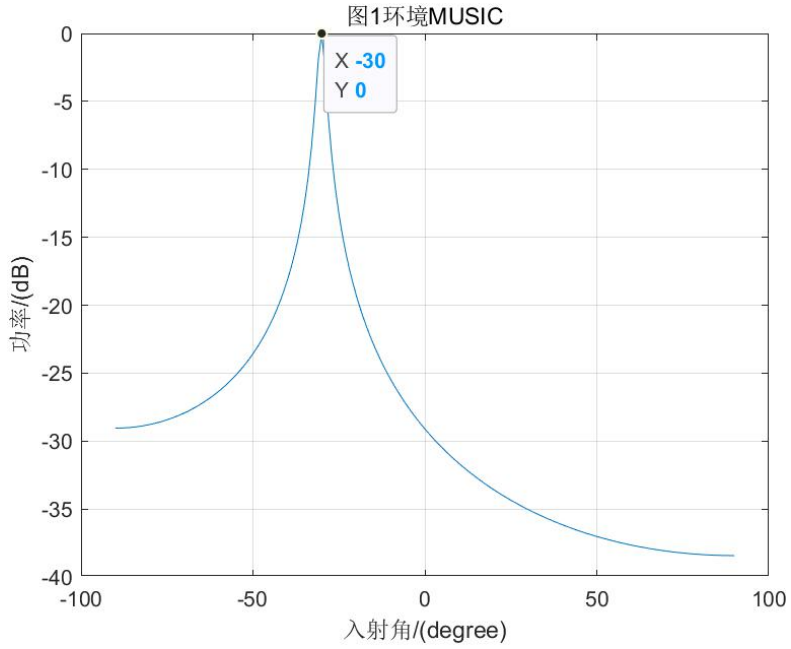
$rank(U_S) = M$　$\therefore U_S$即信号子空间维度为 $N \times M$ 即 8×1

U_W即噪声子空间维度为 ~~8×7~~ ~~$M\times M$~~ ~~$N\times M$~~ $N\times(N-M)$即8×7

（N为阵元个数，M为信源数目）

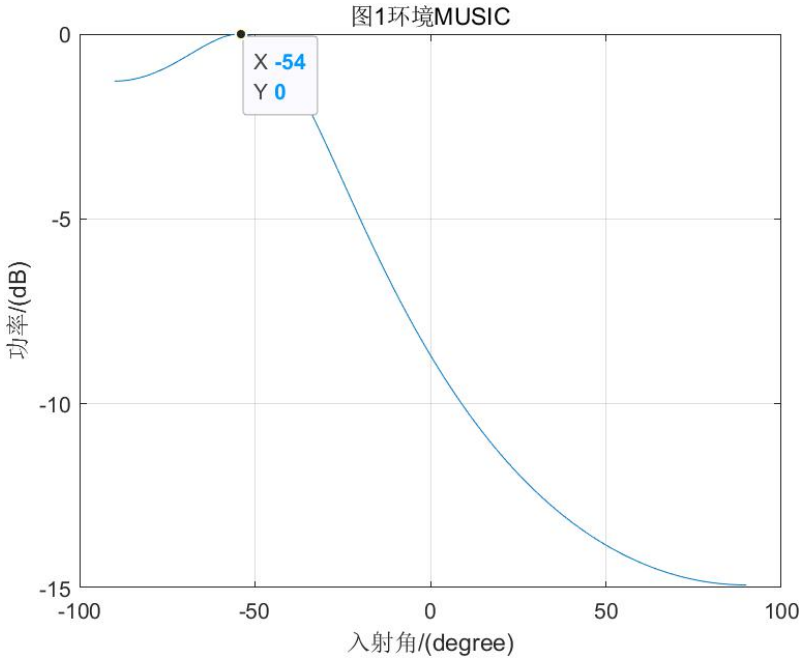
三、输出结果及分析

根据作业说明里的环境（ｓｎｒ＝２０ｄＢ，Ｋ＝１０２４）编写 MUSIC 算法，输出结果如下：



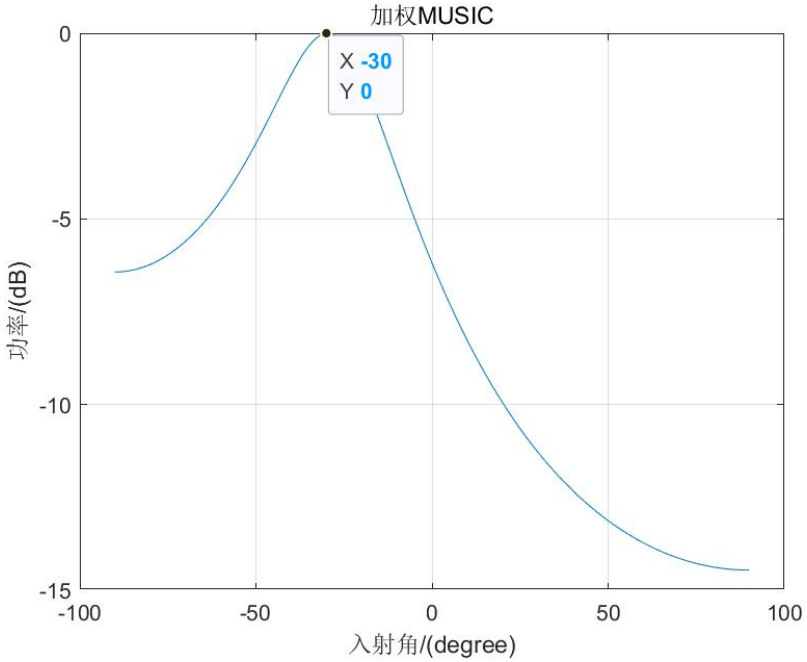
结果分析：由上图可得，ＭＵＳＩＣ算法在信噪比较大、快拍数较大的情况下对单个信号到达角的估计结果较为准确。

在信噪比较低的情况下（ｓｎｒ＝－５），传统ＭＵＳＩＣ算法输出结果如下：



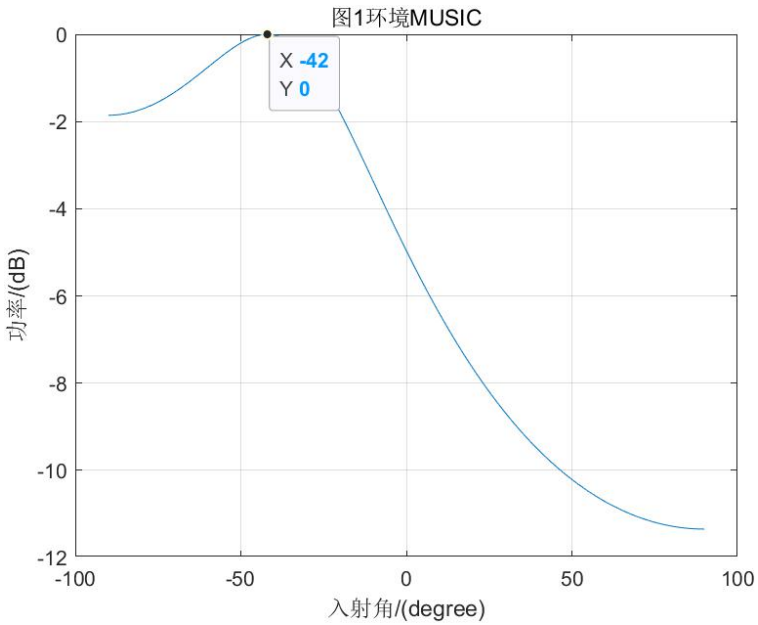
结果分析：可见在信噪比较低的情况下传统ＭＵＳＩＣ算法的性能大大下降，这是因为在信噪比较低的情况下噪声分量对信号分量的影响大大增大，导致式（６）的计算结果与预期差距较大。

因此我们对噪声分量进行加权，即对每一个噪声分量的特征值对应的特征向量乘上特征值的ｎ次，重新构造噪声子空间。经过调试，我发现ｎ在取０.８－１.２的情况下估计角的结果与预期较为接近，因此此处ｎ取１。输出结果如下：



结果分析：可见在信噪比较低的情况下加权 MUSIC 算法可以较为准确的给出信号的估计角，性能优于传统的 MUSIC 算法。

在快拍数较少（K<N，此处快拍数取 4）的情况下，传统的 MUSIC 算法输出结果如下：



结果分析：可见在快拍数较少的情况下，传统 MUSIC 算法的性能大大下降，这是因为协方差矩阵的计算与快拍数相关，在快拍数小的情况下，协方差矩阵的计算存在较大误差，导致之后噪声子空间存在误差，从而导致最终结果误差较大。

因此我们采用求根 MUSIC 算法来改善性能。求根 MUSIC 原理如下所示：

基本 MUSIC 空间谱的谱峰等价为 $\boldsymbol{a}^{\mathrm{H}}(\omega)\boldsymbol{U}_n = \boldsymbol{0}^{\mathrm{T}}$ 或 $\boldsymbol{a}^{\mathrm{H}}(\omega)\boldsymbol{u}_j = 0, j = p+1, \cdots, M$。其中，$\boldsymbol{u}_{p+1}, \cdots, \boldsymbol{u}_M$ 是阵列样本协方差矩阵 $\hat{\boldsymbol{R}}_x$ 的次特征向量。

若令 $\boldsymbol{p}(z) = \boldsymbol{a}(\omega)|_{z=\mathrm{e}^{\mathrm{j}\omega}}$，则有

$$\boldsymbol{p}(z) = [1, z, \cdots, z^{M-1}]^{\mathrm{T}} \tag{10.5.16}$$

向量内积 $\boldsymbol{u}_i^{\mathrm{H}}\boldsymbol{p}(z)$ 给出多项式表示

$$p(z) = \boldsymbol{u}_i^{\mathrm{H}}\boldsymbol{p}(z) = u_{1,j}^* + u_{2,j}^* z + \cdots + u_{M,j}^* z^{M-1} \tag{10.5.17}$$

式中，$u_{i,j}$ 是 $M \times (M-p)$ 次特征向量矩阵 $\boldsymbol{U}_n$ 的第 (i,j) 元素。于是，基本 MUSIC 空间谱表示 $\boldsymbol{a}^{\mathrm{H}}(\omega)\boldsymbol{u}_j = 0$ 或 $\boldsymbol{u}_j^{\mathrm{H}}\boldsymbol{a}(\omega) = 0, j = p+1, \cdots, M$ 可以等价表示为

$$p_i(z) = \boldsymbol{u}_j^{\mathrm{H}}\boldsymbol{p}(z) = 0, \quad j = p+1, \cdots, M \tag{10.5.18}$$

上式又可综合为 $\boldsymbol{U}_n^{\mathrm{H}}\boldsymbol{p}(z) = \boldsymbol{0}$ 或者 $\|\boldsymbol{U}_n^{\mathrm{H}}\boldsymbol{p}(z)\|_2^2 = 0$，即有

$$\boldsymbol{p}^{\mathrm{H}}(z)\boldsymbol{U}_n\boldsymbol{U}_n^{\mathrm{H}}\boldsymbol{p}(z) = 0, \quad z = \mathrm{e}^{\mathrm{j}\omega_1}, \cdots, \mathrm{e}^{\mathrm{j}\omega_p} \tag{10.5.19}$$

换言之，只要对多项式 $\boldsymbol{p}^{\mathrm{H}}(z)\boldsymbol{U}_n\boldsymbol{U}_n^{\mathrm{H}}\boldsymbol{p}(z)$ 求出单位圆上的根 z_i，即可得到空间参数 $\omega_1, \cdots, \omega_p$。这就是求根 MUSIC 的基本思想。

然而，式 (10.5.19) 并不是 z 的多项式，因为它还包含了 z^* 的幂次项。由于我们只对单位圆上的 z 值感兴趣，所以可以用 $\boldsymbol{p}^{\mathrm{T}}(z^{-1})$ 代替 $\boldsymbol{p}^{\mathrm{H}}(z)$，这就给出了求根 MUSIC 多项式

$$p(z) = z^{M-1}\boldsymbol{p}^{\mathrm{T}}(z^{-1})\hat{\boldsymbol{U}}_n\hat{\boldsymbol{U}}_n^{\mathrm{H}}\boldsymbol{p}(z) \tag{10.5.20}$$

现在，$p(z)$ 是 $2(M-1)$ 次多项式，它的根相对于单位圆为镜像对。其中，具有最大幅值的 p 个根 $\hat{z}_1, \hat{z}_2, \cdots, \hat{z}_p$ 的相位给出波达方向估计，即有

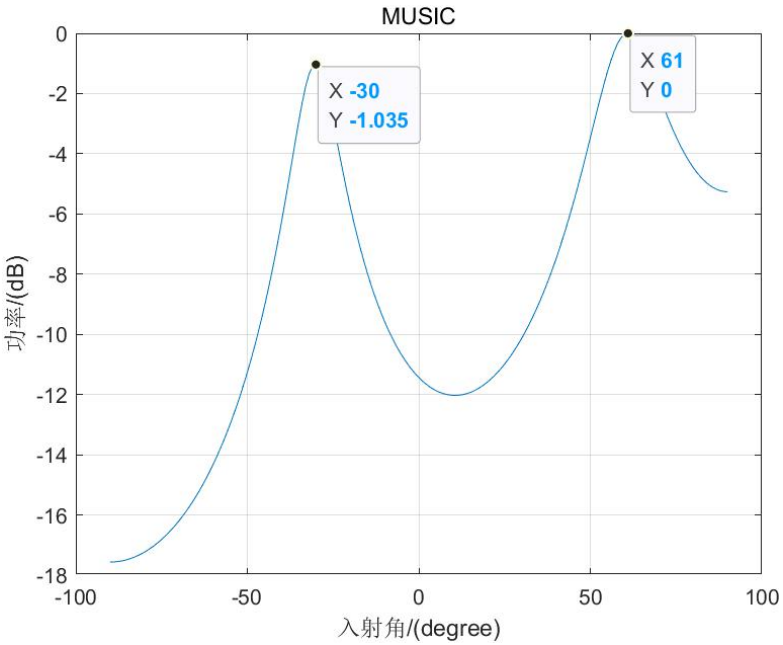
$$\hat{\theta}_i = \arccos\left[\frac{1}{kd}\arg(\hat{z}_m)\right], \quad i = 1, \cdots, p \tag{10.5.21}$$

根据上述原理进行算法编写，最后运行得到如下结果：

```
>> root_music
  -29.7369
```

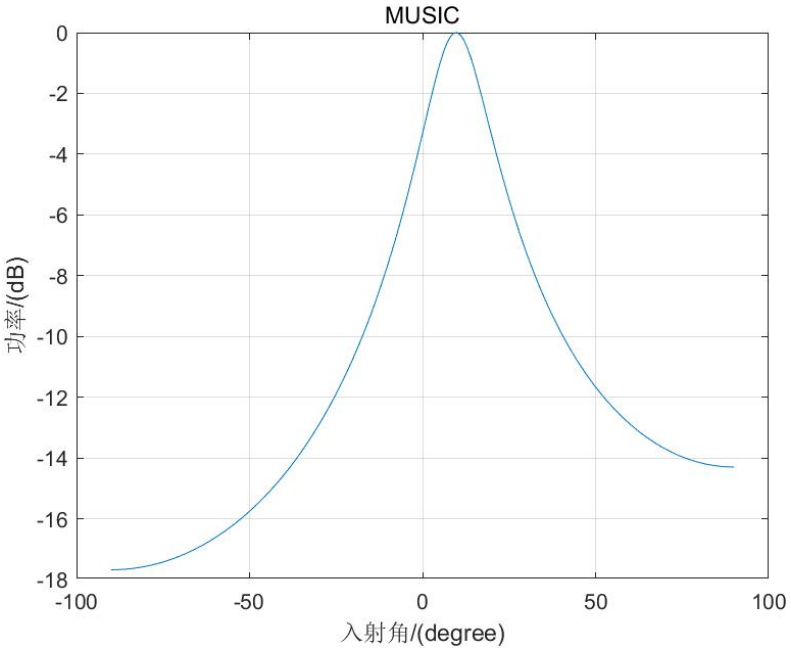
可见求根 MUSIC 算法的估计精确度大大提升。

对于不相干信号，传统 MUSIC 算法的输出结果如下：



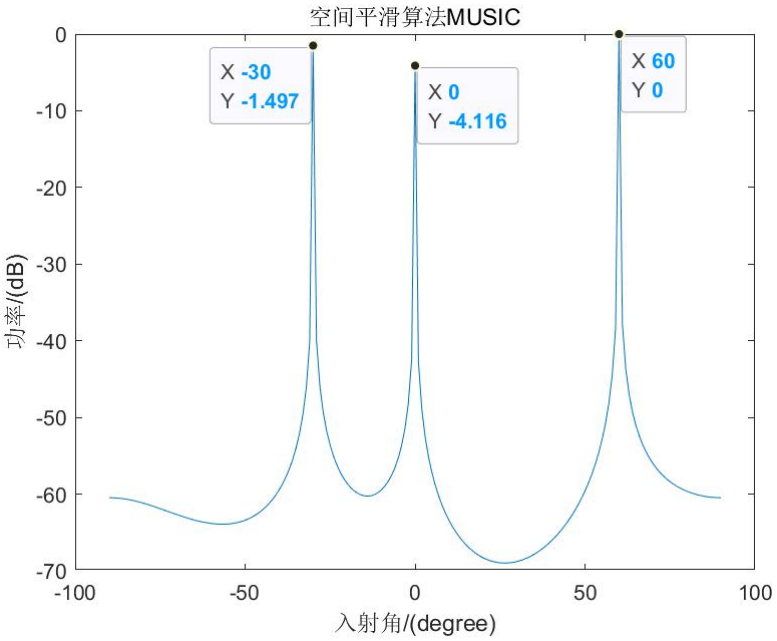
结果分析：依旧可以较为准确的给出出信号的估计角度，这是由于不相干信号并不会对协方差矩阵的秩产生影响，从而不会对噪声子空间的特征向量产生影响。

对于相干信号，则传统算法的性能大大下降，输出结果如下：



结果分析：对于两个相干信号，传统 MUSIC 算法并不能将其很好的分辨，原因是相干信号会导致协方差矩阵的秩亏缺，使得信号特征向量发散到噪声子空间去，即组成噪声子空间的向量中混入了信号源对应的特征向量，从而使得传统 MUSIC 算法性能下降。

为解决上述问题，我采用空间平滑算法来提高 MUSIC 算法的性能，空间平滑算法通过牺牲有效阵列孔径来实现信号源的去相干，以达到信号入射角估计的目的。输出结果如下：



结果分析：可见对于相干信号，空间平滑算法依旧可以准确的分辨出各个信号的入射角度。

四、个人心得

通过此次作业，我对信号到达角估计即 DOA 有了更深的了解，也系统的学习了 MUSIC 算法以及其改进算法，MUSIC 算法通过计算多个阵列阵元接收到的信号的协方差矩阵，得到相对应的信号子空间和噪声子空间，再通过方向矢量和噪声子空间得到最小二乘估计，从而得到信号到达角的估计值。通过加权、求根、空间平滑等方法改进 MUSIC 算法的性能，可以使得 MUSIC 算法更好的适应不同的情况。经过对 MUSIC 算法的学习和实践，我认识到矩阵论所学习的内容在信号处理领域有很大的用处，在今后的学习中，我将更加努力学习专业知识，并对矩阵论相关知识进行更加深入的钻研。

五、附录

传统 MUSIC 算法

```
clear;
close all;
N = 8;                    % 阵元个数
M = 1;                    % 信源数目
theta = -pi/6;        % 单个信号待估计角度
%M = 2;                   % 信源数目
%theta = [-pi/6,pi/3];   % 两个信号待估计角度
snr = 20;                 % 信噪比
K = 1024;                 % 快拍数
fs = 1000;
Ts = 0.001;
T1 = Ts*(K-1);
T = 0:Ts:T1;
dd = 0.5;                 % 阵元间距
```

```
d = 0:dd:(N-1)*dd;
S = sin(100*pi*T);          %信源信号
%S2 = cos(100*pi*T);       %不相干信号
A = exp(-1j*2*pi*d'*sin(theta)*50/1500);     %方向向量
X = zeros(N,K);
for i = 1:N
      X(i,:)=S*A(i,1);                          %单个信号
      %X(i,:)=S*A(i,1)+S*A(i,2);               %两个相干信号
      %X(i,:)=S*A(i,1)+S2*A(i,2);             %两个不相干信号
end
X1 = awgn(X,snr);     %添加噪声，使得信噪比为 20dB
R = X1*X1'/K;            %计算协方差矩阵
[V,D] = eig(R);         %特征值分解
Uw=V(:,1:N-M);          %噪声子空间
P = zeros(1,181);
w = -pi/2:pi/180:pi/2;
theta1 = -90:1:90;
for i = 1:length(w)           % 遍历每个角度，计算空间谱
      a = exp(-1j*2*pi*d'*sin(w(i))*50/1500);
      P(i) = 1/(a'*(Uw*Uw')*a);
end
P = abs(P);
[Pmax,index]=max(P);
P = 10*log10(P/Pmax);     %归一化功率转化为 dB
plot(theta1,P);
title('MUSIC');
xlabel('入射角/(degree)');
ylabel('功率/(dB)');
grid on;
```

加权 MUSIC 算法

```
clear; close all;
N = 8;                        % 阵元个数
M = 1;                        % 信源数目
theta = -pi/6;   % 待估计角度
snr = -5;                    % 信噪比
K = 1024;                     % 快拍数
fs = 1000;
Ts = 0.001;
T1 = Ts*(K-1);
T = 0:Ts:T1;
dd = 0.5;                    % 阵元间距
d = 0:dd:(N-1)*dd;
S = sin(100*pi*T);
A = exp(-1j*2*pi*d'*sin(theta));
```

```
X = zeros(N,K);
for i = 1:N
    X(i,:)=X(i,:)+S*A(i);
end
X1 = awgn(X,snr);
I=eye(N);
I1=rot90(I);
X2=conj(X1);
Y=I1*X2;
% 计算协方差矩阵
R2 = X1*X1'/K;
R3 = Y*Y'/K;
R = R2+R3;
[V,D] = eig(R);        %特征值分解
Uw=V(:,1:N-M);
D1 = diag(D);
P = zeros(1,181);
w = -pi/2:pi/180:pi/2;
theta1 = -90:1:90;
for i = 1:N-M
    Uw(:,i)=Uw(:,i)*(D1(i)^(1));
end
for i = 1:length(w)          % 遍历每个角度，计算空间谱
    a = exp(-1j*2*pi*d'*sin(w(i)));
    P(i) = 1/(a'*Uw*Uw'*a);
end
P = abs(P);
Pmax=max(P);
P = 10*log10(P/Pmax);    %功率转化为 dB
plot(theta1,P);
xlabel('入射角/(degree)');
ylabel('功率/(dB)');
grid on;
```

求根 MUSIC 算法

```
clear; close all;
N = 8;                      % 阵元个数
M = 1;                      % 信源数目
theta = -pi/6;   % 待估计角度
snr = 20;                   % 信噪比
K = 3;                      % 快拍数
fs = 1000;
Ts = 0.001;
T1 = Ts*(K-1);
T = 0:Ts:T1;
```

```
dd = 0.5;                    % 阵元间距
d = 0:dd:(N-1)*dd;
S = sin(100*pi*T);
A = exp(-1j*2*pi*d'*sin(theta));     %方向向量
X = zeros(N,K);
for i = 1:N
    X(i,:)=S*A(i,1);
end
X1 = awgn(X,snr);     %添加噪声，使得信噪比为 20dB
R = X1*X1'/K;
[V,D] = eig(R);         %特征值分解
Uw=V(:,1:N-M);          %噪声子空间
syms z;
pz = z.^([0:N-1]');
pz1 = (z^(-1)).^([0:N-1]);
fz = z.^(N-1)*pz1*Uw*Uw'*pz;
a=sym2poly(fz);
zx=roots(a);
zx1=zx';
[as,ad]=(sort(abs((abs(zx1)-1))));
angleest=asin((angle(zx1(ad(1)))/pi))*180/pi;
disp(angleest);
```

空间平滑 MUSIC 算法

```
clear; close all;
N = 8;                       % 阵元个数
M = 3;                       % 信源数目
J = 5;
J_N=N-J+1;
theta = [-pi/6,0,pi/3];   % 待估计角度
snr = 20;                    % 信噪比
K = 1024;                    % 快拍数
fs = 1000;
Ts = 0.001;
T1 = Ts*(K-1);
T = 0:Ts:T1;
dd = 0.5;                    % 阵元间距
d = 0:dd:(N-1)*dd;
S = sin(100*pi*T);
A = exp(-1j*2*pi*d'*sin(theta));
X = zeros(N,K);
for i = 1:N
    for j = 1:length(theta)
        X(i,:)=X(i,:)+S*A(i,j);
    end
```

```
end
X1 = awgn(X,snr);
% 计算协方差矩阵
R = X1*X1'/N;                                          % 原始协方差矩阵
Rf = zeros(J_N, J_N);                                  %运用空间平滑算法重新构造协方差矩阵
for i = 1:J
    Rf = Rf+R(i:i+J_N-1,i:i+J_N-1);
end
Rf = Rf/J;
[V,D] = eig(Rf);        %特征值分解
Uw=V(:,1:J_N-M);        %噪声子空间
P = zeros(1,181);
w = -pi/2:pi/180:pi/2;
theta1 = -90:1:90;
for i = 1:length(w)          % 遍历每个角度，计算空间谱
    a = exp(-1j*2*pi*d(1:J_N)'*sin(w(i)));
    P(i) = 1/(a'*(Uw*Uw')*a);
end
P = abs(P);
Pmax=max(P);
P = 10*log10(P/Pmax);    %功率转化为 dB
plot(theta1,P);
title('空间平滑算法 MUSIC');
xlabel('入射角/(degree)');
ylabel('功率/(dB)');
```