DOA传统算法仿真

1. DOA估计概述

波达方向（Direction Of Arrival）估计，又称为角谱估计（Angle spectral estimation）、波达角（Angle Of Arrival）估计。目的是估计出哪个发射机在工作以及发射机所处的方向，简单的说就是利用己方雷达接收来自目标发射机的来波方向进行估计。

1. 阵列信号模型

本仿真仅考虑均匀线阵情况。设有N个接收机阵元，并且每个阵元间隔离为d。考虑发射机发射的信号为一窄带信号，载频为。因此发射信号可以表示为

   

现假设目标回波从无穷远处到达阵列。在无穷远的假设下，雷达回波可视为平面波，因此到达每个阵元的回波信号为



其中，为阵元接收回波信号幅度，由于发射机发射的信号为窄带信号，相对于载频而言，变化的时间，可以被视为一常量。因此，在窄带、远场假设下每个接收机接收到的回波信号为



由此可见，对于接收机接收的每一个信号来说，接收信号的幅度不变，区别在于相位上的由阵元位置排列引起的时延而导致的相位变化。

现在将上式写成矢量形式



其中,,为空域导向矢量。

下面是仿真的Matlab代码

|  |
| --- |
| function [lambda, d]=parameter()  c = 3e8; %光速  f0 = 2e9; %载频  lambda = c/f0; %波长  d = lambda/2; %阵元间距  end  function s = GenerateSignal(N, SNR,target\_theta)  f0 = 2e9; %载频  fs = 10e6; %快时间采样速率（系统带宽）  Ts = 1 / fs; %采样间隔  t = 0:Ts:(N-1)\*Ts;  %% 回波信号为随机信号  s = 10^(SNR/20).\*randn(1,N).\*ones(1,length(target\_theta))'.\*exp(1i\*2\*pi\*f0.\*t);  end  function X = GenerateX(s, ArrayNum, N, target\_theta)  [lambda, d]=parameter();  Noise=randn(ArrayNum, N).\*exp(1i\*2\*pi\*rand(ArrayNum,N));  A = exp(-1i\*2\*pi\*d\*(0:ArrayNum-1)'\*sind(target\_theta)/lambda);  X = A\*s + Noise;  end |

由于接收机接收的信号幅度与信号波形无关，因此在仿真中采用随机信号幅度作为接收信号幅度。

1. DOA传统算法仿真

3.1.1 CBF算法介绍3.1 常规波束形成算法(CBF)仿真

设各阵元权矢量



阵列输出



阵列的功率输出为



其中为阵列接收信号自相关矩阵

在实际中，可以用过最大似然估计近似



其中，为空域快拍数。

当时为常规波束形成算法。有



3.1.2 CBF算法仿真结果

本实验仿真了阵元数、快拍数、信噪比变化对CBF算法性能的影响。以下是实验结果及仿真代码：

|  |
| --- |
| 从上图可以看出随着阵元数量的提高，CBF算法的空间谱主瓣明显变窄并且第一副瓣降低。检测性能显著提高。 |
| 从上图可以看出在快拍数增多的情况下，空间谱的主瓣宽度变化不大，第一副瓣降低2dB左右。快拍数增多在此情形下对性能改善不明显。 |
| 从上图可以看出随着信噪比的提升，CBF算法的空间谱主瓣变化不大，第一副瓣降低。检测性能显著提高。 |

综上，CBF算法能够完成单目标DOA估计任务，在实际应用中增多阵元数目以及改善信噪比对性能提升明显。

下面是CBF算法的仿真代码：

|  |
| --- |
| function CBF(ArrayNum,N,SNR,target\_theta)  [lambda, d]=parameter();  s = GenerateSignal(N, SNR, target\_theta);  X = GenerateX(s, ArrayNum, N, target\_theta);  Theta = -90:0.1:90;  a = exp(-1i\*2\*pi\*d\*(0:ArrayNum-1)'\*sind(Theta)/lambda);  Rxx = (1/N)\*X\*X';  P = abs(diag(a' \* Rxx \* a));  plot(Theta, 10\*log10(P/max(P)));  xlabel('theta(°)');  ylabel('方位谱图(dB)');  end |

3.2.1 Capon算法介绍

Capon算法属于一种在线性约束最小方差准则(LCMV)下的波束形成算法，所谓波束形成即在某些准则约束下，求解阵列输出的最优权。实质上波束形成属于一种空域滤波器。波束形成的“导向”作用是通过调整加权系数完成的,阵列的输出是对各阵元的接收信号量x在 各阵元上的加权和



假设空间远场有一个感兴趣的信号d(t) (其波达方向为)和J个干扰信号,1,..J((其波达方向为),令每个阵元上的加性白噪声为nk(t),它们具有相同的方差。在这些假定的条件下，第k个阵元上的接收信号可以表示为



将上式写成矩阵形式可得



其中,。

假设接收端得到了个快拍数据，并且当时有



为了保证来自方向的信号正常接收，同时完全抑制掉其它J个干扰。得到如下约束条件



综上，求解Capon算法的最优化问题可以表述为



使用拉格朗日乘子法可以得到Capon的空间谱为



3.2.2 Capon算法仿真结果

本实验仿真了阵元数、快拍数、信噪比变化对Capon算法性能的影响。以下是实验结果及仿真代码：

|  |
| --- |
| 从上图可以看出随着阵元数量的提高，Capon算法的空间谱主瓣明显变窄并且第一副瓣降低。检测性能显著提高。 |
| 从上图可以看出Capon需要一定的快拍数才能起作用。在快拍数为10的情况下，Capon算法识别出多个假目标并且识别精度差。在快拍数为20和100的时候主瓣宽度不变，说明其主瓣宽度以接近极限值。在这种情况下，增大快拍数对提高检测精度无明显作用。 |
| 从上图可以看出随着信噪比的提升，Capon算法的空间谱主瓣变窄，第一副瓣降低。检测性能显著提高。 |

综上，Capon算法能够完成单目标DOA估计任务，在实际应用中应选择合适的快拍数，并在此基础上增多阵元数目以及改善信噪比实现性能的明显提升。

下面是Capon算法的仿真代码：

|  |
| --- |
| function Capon(ArrayNum, N, SNR, target\_theta)  [lambda, d]=parameter();  s = GenerateSignal(N, SNR, target\_theta);  X = GenerateX(s, ArrayNum, N, target\_theta);  %% 求x的相关矩阵逆  Rxx = (1/N).\*X \* X';  Rxx\_inv = inv(Rxx);  Theta = -90:0.1:90;  a = exp(-1i\*2\*pi\*(0:ArrayNum-1)'\*d\*sind(Theta)/lambda);  P\_Capon = abs(diag(1./(a'\*Rxx\_inv\*a)));  %music = abs(diag(1./(a\_omega'\*(Un\*Un')\*a\_omega)));  P\_Capon = 10\*log(P\_Capon/max(P\_Capon));  plot(Theta, P\_Capon);  title('Capon算法');  xlabel('Theta(°)');  end |

3.3.1 Music算法介绍

Music算法时将阵列接受数据协方差矩阵R进行特征分解，得到相互正交的信号子空间和噪声子空间，然后利用该正交特性来估计信号的参数。

省去复杂的推导，可以得到基于信号子空间法的空间谱



Music算法与Capon算法的空间谱表达式类似，区别在于使用了噪声子空间的相关矩阵代替了。

3.3.2 Music算法仿真结果

本实验仿真了阵元数、快拍数、信噪比变化对Music算法性能的影响。以下是实验结果及仿真代码：

|  |
| --- |
| 从上图可以看出随着阵元数量的提高，Music算法的空间谱主瓣变窄并且第一副瓣降低。增多阵元数对空间谱改善效果不明显。 |
| 从上图可以看出随着快拍数的提高，Music算法的空间谱主瓣明显变窄并且第一副瓣降低。检测性能显著提高。 |
| 从上图可以看出随着信噪比的提升，Music算法的空间谱主瓣变窄，第一副瓣降低。检测性能提高。 |

综上，Music算法能够完成单目标DOA估计任务，在实际应用中应选择合适的快拍数，并在此基础上增多阵元数目以及改善信噪比实现性能的明显提升。

下面是Music算法的仿真代码：

|  |
| --- |
| function Music(ArrayNum, N, SNR, target\_theta)  [lambda, d]=parameter();  s = GenerateSignal(N, SNR, target\_theta);  X = GenerateX(s, ArrayNum, N, target\_theta);  %% 计算相关系数  Rxx = (1/N)\*X\*X';  [U,V] = eig(Rxx);  V = diag(V);  [V,idx] = sort(V,'descend');  U = U(:,idx);  P = sum(V);  P\_cum = cumsum(V);  J = find(P\_cum/P>=0.9);  J = J(1);  Un = U(:,J+1:end);  theta = -90:0.1:90;  a = exp(-1i\*2\*pi\*(0:ArrayNum-1)'\*d\*sind(theta)/lambda);  music = abs(diag(1./(a'\*(Un\*Un')\*a)));  music = 10\*log(music/max(music));  plot(theta, music);  title('Music算法');  xlabel('Theta(°)');  end |

3.4 CBF、Capon、Music算法对比



在单目标的情况下看以看出Music算法在这三种算法里旁瓣电平最低，主瓣宽度最窄。在单目标的情况下效果最好。

3.5.1 Espirt算法介绍

Espirt算法估计信号参数时要求阵列的几何结构存在所谓不变性。该不变性可以通过取前N-1个阵元组成子阵列和后N-1个阵元组成子阵列实现。利用前一个子阵列和后一个子阵列之间的相位关系求解。

假设两个子阵列接收到的数据分别为,





其中

只要得到两个子阵间的旋转不变关系，就可以得到信号到达角。

3.5.2 Espirt算法仿真结果

本实验设定在阵元数为10，快拍数为100，信噪比为-5dB，目标在0°情况下，以下是实验结果及仿真代码：

Esprit估计结果：

0.1353

下面是Esprit算法的仿真代码：

|  |
| --- |
| function Esprit(ArrayNum, N, SNR, target\_theta)  K = length(target\_theta);  s = GenerateSignal(N, SNR, target\_theta);  X = GenerateX(s, ArrayNum, N, target\_theta);  Rxx = (1/N)\*X\*X';  [U, D] = eig(Rxx);  [D, idx] = sort(diag(D));  U = fliplr(U(:, idx));  Us = U(:, 1:K);  Ux = Us(1:ArrayNum-1, :);  Uy = Us(2:ArrayNum, :);  Uxy = [Ux, Uy];  Uxy = Uxy'\*Uxy;  [U\_xy, D\_xy] = eig(Uxy);  [D\_xy, idx\_xy] = sort(diag(D\_xy));  F = fliplr(U\_xy(:, idx\_xy));  F0 = F(1:K, K+1:K\*2);  F1 = F(K+1:K\*2, K+1:K\*2);  Psi = -F0/F1;  [T,Phi] = eig(Psi);  Theta = asind(-angle(diag(Phi))/pi); % 估计角度  Theta = sort(Theta).';  disp('Esprit估计结果：');  disp(Theta);  end |