|  |  |
| --- | --- |
|  | 2029320381452802095400 |

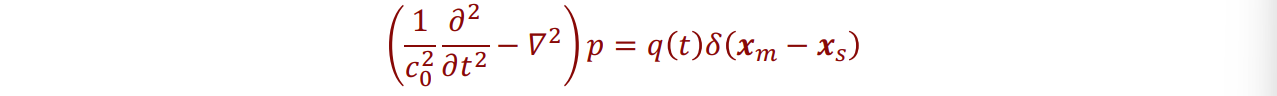
**智能信号处理与应用**

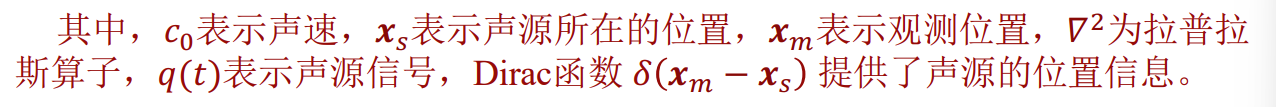
**作业四**

|  |  |
| --- | --- |
| **姓 名：** |  |
| **学 号：** |  |
| **专业班级：** |  |
| **任课教师：** | **涂晓彤** |
| **日 期：** |  |

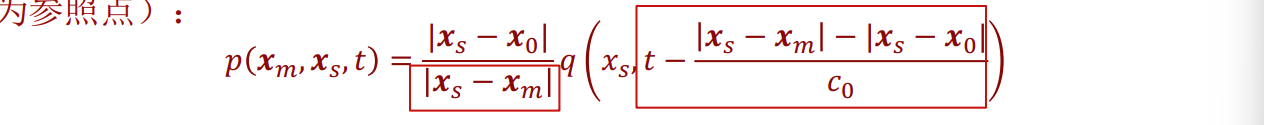
1. **实验目的**
2. 熟悉掌握MATLAB编程工具
3. 认识并理解DAS算法实现原理
4. 熟悉Python语言对DAS算法的编程实现
5. 为完成最后大作业打下算法基础
6. **实验原理**
7. **声学相机**：即麦克风阵列(Microphone Array),也叫声相仪。它由多个麦克风按照一定规律排列，通过阵列信号处理算法生成声音在一个平面上声压级分布，以彩色等高线图的方式实现声音可视化，通过照片或视频的方式显示被测物的声音分布。
8. **延迟求和算法(Delay-and-Sum,DAS)**：一个给定声源产生的声波随传播距离的不同将会呈现出不同的相位。因此阵列中每个麦克风将“感知”不同的相位，从而确定声源位置。

声波在连续介质中的传播可以用Navier-Stokes方程精确表示。然而在日常和工业应用中，声波传播可以被认为是线性等熵现象。因此复杂的Navier-Stokes方程可以简化为Helmholtz波动方程。





以上非齐次波动方程具有自由场解（无反射或固体边界）。假设在xs处有一个声源，则在第m个麦克风xm处接收到的声音信号为



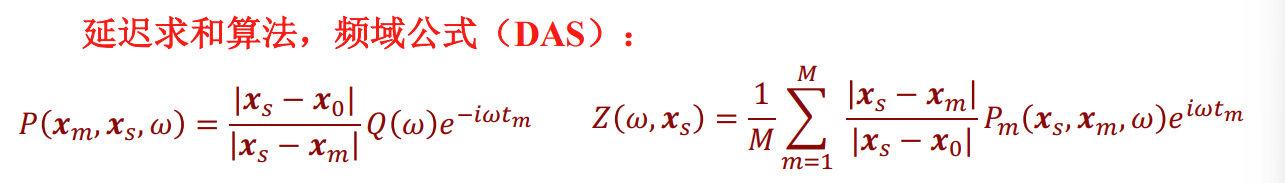
以上可以看出声波传播的一些性质：

1. 信号的振幅和观测点到声源的距离成反比
2. T时刻的信号观测存在一个时间延迟

**DAS算法定位流程：**

1. 确定声源可能处在的平面，并将该平面划分为若干个矩形网格。
2. 扫描每个网格点，在每个网格扫描点，对每个麦克风测量到的信号按各自的延迟时间进行信号补偿。
3. 将每个麦克风的信号相加并除以麦克风数，得到波束形成器输出图。

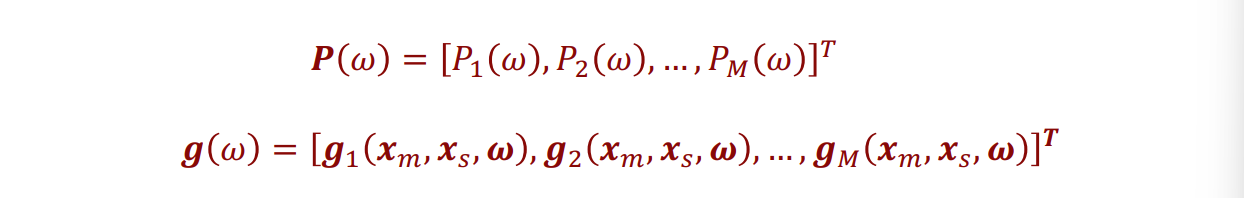
延迟求和算法的实现描述了声学成像的核心思想，利用波束形成和时间差反演出声源分布。随着快速傅里叶变换的出现和计算机性能的不断提升，在时域计算DAS波束成像已被基于互谱矩阵（CSM）和相移的频域波束成像所替代。



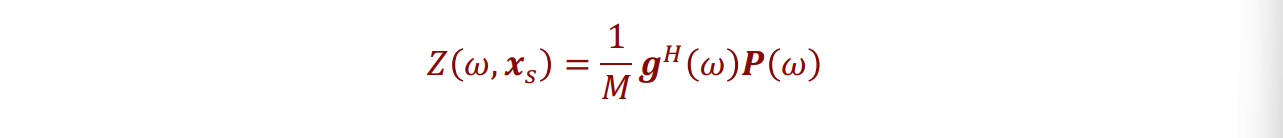
为了简化公式我们定义转向矢量gm(x,x0,w)



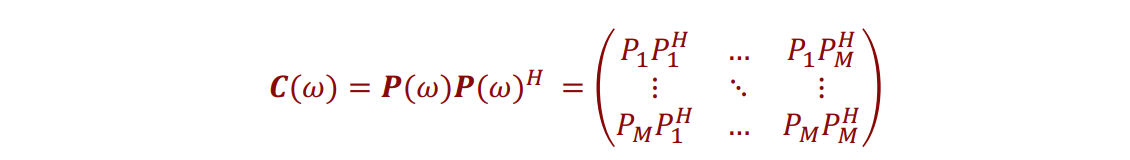
写成矩阵形式得到：



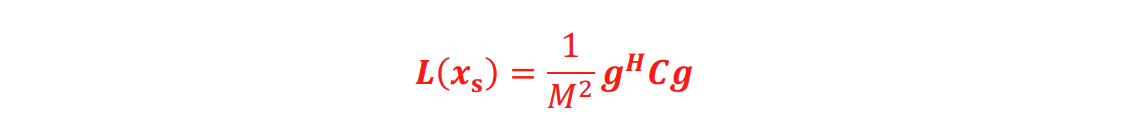
则波束输出Z(w,xs)可以表示为矩阵形式



又为了简化，我们定义了互谱矩阵(Cross-Spectral Matrix,CSM),为

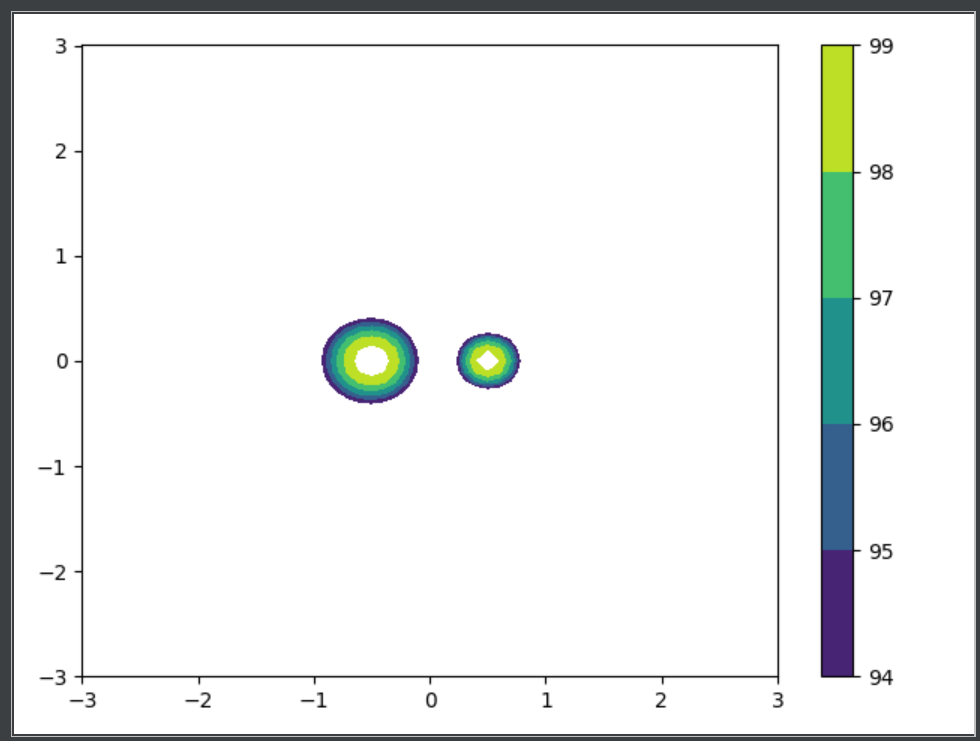


则延迟求和算法（频域形式）的波束形成位于网格点xs的表达式为



**三、实验内容**

本次实验主要实现的是将MATLAB语言编写的DAS算法转换成Python语言编写的格式。以下图片便是通过模拟声源后经DAS算法绘制的等高线图。



在实验过程中遇到了很多的小问题，在这里记录几个比较隐晦的问题。

1. matlab中的fft和np.fft.fft函数有一些细微的区别，就是matlab中的fft函数默认沿列向量方向进行fft运算，而np.fft.fft函数默认沿行向量方向做fft运算。
2. matlab中对一个矩阵做’运算，它是做的共轭转置运算，因此在python中要等价运算需要使用np.conj(np.T)这样才能实现共轭转置。
3. 在python中使用numpy工具对于一维向量是无法进行转置的，也就是一个行向量的转置还是它本身，因此这是就需要我们将其进行reshape编程一行n列的向量，这样才能转置成n行1列的向量。
4. 在使用numpy创建零矩阵时，需要声明其dtype为’complex’类型，否则在进行复数运算时，它会自动舍弃虚部。
5. 在python中使用\*对两个矩阵进行运算时，它默认的是求内积，而matlab中的矩阵之间的运算采用\*默认为矩阵乘法，因此这时需要用到numpy中的matmul函数进行运算。

**四、实验总结**

本次实验完成了对MATLAB语言实现的DAS算法到Python语言的DAS算法实现的转换，在这个过程中，需要我们先对DAS算法中几个重要的矩阵计算有比较深入的理解，这样有利于我们看懂MATLAB的代码。其次在实现Python语言的DAS算法编写过程中，需要我们非常细致地去找到和MATLAB中函数相对应的Python语言函数实现，同时还需要特别注意它们之间存在的区别，这样可以有效地避免在整体代码框架完成后花费大量的时间进行debug。最后，在发现程序输出的结果与实际理论结果不一致的时候，我们需要进行调试分析，通过在一些关键节点对关键的数据进行打印输出，然后观察它和MATLAB输出结果的一致性来判断这一部分代码的正确性。

通过本次实验增强了我对MATLAB代码的阅读能力、对Python语言代码的编写能力以及对报错信息的查看、理解、解决能力。