压缩感知数据重构算法与仿真

完成日期：

指导教师签字：

答辩小组成员签字：

摘要

作为一种新型的采样理论，压缩感知突破了传统奈奎斯特采样定律的限制，在信号稀疏的前提下通过采样矩阵实现对信号的采样和压缩。压缩感知不再以信号的频率作为采样条件，并且在采样的同时完成对信号的压缩，从而减少了采样得到的无效数据，降低了数据存储、传输的代价。

本文首先介绍压缩感知相关基本理论，包括信号的稀疏表示法，采样矩阵的设计，以及压缩感知数据重构三个部分。重点和关键在于重构算法的研究和仿真，对常用的基追踪算法、匹配追踪算法、正交匹配追踪算法、子空间追踪算法，以及压缩采样匹配追踪算法进行理论研究。分析比较各算法的原理和执行步骤，并利用一维信号和二维信号作为原始信号进行重构仿真实验，比较不同重构算法在运行的时间复杂度上和重构精度上的差别，总结不同算法的优点和缺点。

关键词：稀疏表示，采样矩阵，数据重构

Abstract

As a new sampling theory, Compressed Sensing breaks the limitations of traditional sampling theorem, it can sample and compress the signals which are sparse through the sampling matrix. Compressed Sensing does not take the frequency of the signal as the sampling condition, and completes the compression and sampling of signals at the same time, thus reducing the redundant data from sampling, and reducing the cost of the data storage and transmission.

This article explains the basic theory of Compressed Sensing, including the sparse representation of signal, the design of sampling matrix, and the data reconstruction of Compressed Sensing. The key is to study the reconstruction algorithms and get their simulation data. Do some theoretical research on Basis Pursuit algorithm, Matching Pursuit algorithm, Orthogonal Matching Pursuit algorithm, Subspace Pursuit algorithm and Compressive sampling matching pursuit algorithm, analyze their principle and operation steps, and get the simulation data by using one dimensional and two dimensional signal to do experiment on reconstruction algorithms. After getting the simulation data of different reconstruction algorithms, Compare the time complexity by using their running time data, and compare the reconstruction accuracy by using their reconstruction error data. At last, summarize the advantages and disadvantages of different reconstruction algorithms.

**Keywords**：sparse representation，sampling matrix，data reconstruction

目录

[第一章 绪论 1](#_Toc450332670)

[1.1 课题研究背景及意义 1](#_Toc450332671)

[1.2 国内外研究现状 2](#_Toc450332672)

[1.2.1 信号稀疏表示理论 2](#_Toc450332673)

[1.2.2 采样矩阵 3](#_Toc450332674)

[1.2.3 重构算法 3](#_Toc450332675)

[1.3 本文的主要研究内容和结构安排 4](#_Toc450332676)

[第二章 压缩感知基本理论 5](#_Toc450332677)

[2.1 信号的稀疏表示 5](#_Toc450332678)

[2.2 采样矩阵的研究 8](#_Toc450332679)

[2.3 压缩感知数据重构 11](#_Toc450332680)

[第三章 压缩感知重构算法研究及仿真 13](#_Toc450332681)

[3.1 基追踪算法(BP) 13](#_Toc450332682)

[3.2 匹配追踪算法(MP) 16](#_Toc450332683)

[3.3 正交匹配追踪算法(OMP) 18](#_Toc450332684)

[3.4 子空间追踪算法(SP) 22](#_Toc450332685)

[3.5 压缩采样匹配追踪算法(CoSaMP) 26](#_Toc450332686)

[3.6 算法性能比较 29](#_Toc450332687)

[第四章 总结与展望 33](#_Toc450332688)

[参考文献 34](#_Toc450332689)

[致谢 35](#_Toc450332690)

[附录 36](#_Toc450332691)

# 第一章 绪论

## 1.1 课题研究背景及意义

随着信息技术的发展，数字信号已进入人类社会生活的各个领域。例如，手机和计算机里的数码图片，MP3音乐，MP4视频文件等。但是在现实生活中人们接触到的往往是模拟信号，比如环境温度，空气湿度，广播电塔发射出的电磁波信号，有线电话通讯中传输的电压信号等。对这一类信号进行处理时通常依赖于现代计算机技术，由于计算机的离散化存储机制，在信息保存的时候，需要对这些信号进行时域采样，将其由模拟信息转化为数字信息。

过去的几十年中，奈奎斯特采样定律一直在信号处理领域占据主导地位，通过对模拟信号进行时域间隔采样便可得到离散的样本数据，从而借助计算机进行存储，处理和传输。但该定律指出：在采样的时候，采样的频率必须高于目的信号最高频率的两倍才能够保证重构出来的信号不失真。随着科学技术的发展，需要观测的数据不断增大，比如在天文观测，医疗成像，雷达扫描等方面，信号的带宽很大，采样得到的数据量急剧增长，在传输，处理这些数据的时候，往往还要对其进行压缩，丢弃其中冗余的信息，这就带来了很大的时间代价。另一方面，当目的信号的频率过高，超出了现有技术所能采样的频率限制的时候，传统的奈奎斯特采样定律根本无法实现。

为了打破奈奎斯特定律的限制，降低采样、压缩过程带来的成本代价，Donoho和Candes等人于2006年提出了一种新型的采样理论——压缩感知（Compressed Sensing，CS）。与传统采样定律先采样后压缩的过程不同，压缩感知采样理论基于信号的稀疏性，在采样的过程中完成对信号的压缩，能够以远低于奈奎斯特采样定律的频率进行采样，获得的采样数据也更少。在信号的重构方面，压缩感知也能借助于极少的采样数据，选取适当的重构算法，实现信号重建。目前常用的算法包括凸优化类算法，匹配追踪类算法等。

对于奈奎斯特采样定律而言，采样频率大于目的信号频率的两倍是其先决条件，同样，信号的稀疏性也是压缩感知理论的前提。但现实世界中的自然信号往往是非稀疏的，这便是压缩感知理论中的一个研究热点——信号的稀疏表示。虽然原始信号并不是稀疏的，但可以借助于一些正交变换基，将原始信号投影到正交基上，得到在变换域中表现为稀疏的数据。例如，对于一幅自然图片，其中的像素值几乎都是非零的，但是将这幅图片进行小波变换，得到的数据中大量的系数的绝对值都趋于零，这时便可利用压缩感知做进一步处理。

压缩感知理论自提出以来就受到广泛关注，目前对该理论研究的热点除了上面所述的信号稀疏表示外，还有采样矩阵的设计以及重构算法的研究。信号通过正交基变换可以转化为稀疏信息，再通过设计好的采样矩阵便能得到采样压缩后的数据，然后通过重构算法实现重构。信号的稀疏表示，采样矩阵的设计都很容易实现，但如何通过采样数据进行信号重建，并且保证较高的恢复效果，一直是研究的重点。所以压缩感知重构算法的研究在整个压缩感知理论中有着重要的意义。

## 1.2 国内外研究现状

压缩感知于2006年被提出以后，受到了国内外的广泛关注，很多领域的学者都对其展开了研究。目前对压缩感知的研究主要集中在信号的稀疏表示，采样矩阵的选取，重构算法这三个方面。

### 1.2.1 信号稀疏表示理论

压缩感知理论要求所采样的信号必须是稀疏的，而现实中的信号通常不具备稀疏性，这就需要对信号进行某种表示，将其变成稀疏性信号。目前常用的表示方法有傅里叶变换、离散余弦变换、小波变换、以及多尺度几何分析。傅里叶变换是把时间函数变换到频率域，通过一系列不同频率的正弦波实现对原信号的线性组合，逼近。傅里叶变换虽然应用很广泛，但是缺乏对原始信号任意局部特征的表示。相对而言，小波变换就具有良好的局部表示特性。但小波信号在一维空间所具有的优势并不能应用于多维空间中，于是就引出了多尺度集合分析方法。

国外的Meyer，Coifman，Donoho，Candes等人先后提出了梳妆波变换，楔波变换，小线变换，脊波变换，曲线波变换，为高维信号的多尺度分析奠定了理论基础。目前，对信号的稀疏表示研究仍在继续，研究的热点问题主要是如何找到更好的变换域，使得目的信号在该变换域下的表示尽量稀疏，减少采样数据，并且保证较高的重构精度。

### 1.2.2 采样矩阵

采样矩阵是压缩感知数据获取和重构过程中非常重要的一步。Candes已经证明：为了重构稀疏信号，采样矩阵必须满足约束等距条件。目前国内外研究的采样矩阵中满足这一要求的有局部傅里叶矩阵、高斯随机测量矩阵、伯努利随机测量矩阵、二进制随机矩阵、循环矩阵等。

在压缩感知中，对于目的信号X∈RN，构造一个采样矩阵Φ(M×N，MN)，若信号X是稀疏的，则采样数据为Y=ΦX。于是将N×1的目的信号采样后得到M×1的数据，当M远小于N时采样得到的冗余数据很少。因此压缩感知突破了传统采样定律在高维高频率信号采样方面的局限。目前，国内外对采样矩阵的研究仍在继续。如何找到一种新型的采样矩阵使得采样得到的数据更少，并且在保证重构精度的前提下采样矩阵的性质更好，以及冗余字典和采样矩阵的匹配关系，一直是当前研究的热点问题。

### 1.2.3 重构算法

根据信号的稀疏表示方法，目前通常采用的重构算法包括凸优化类算法、贪婪算法、组合算法、贝叶斯方法等。在凸优化类算法中，使用最早的是国外的Mohimani提出的SLO算法，该算法利用多次迭代，逼近求得最优解。国内的研究学者林婉娟等人在2011年提出了改进的SLO算法—NSLO算法，提高了SLO算法中收敛速度较慢的问题。目前NSLO已被用于高分辨率SAR图像的成像系统中。而在贪婪算法领域，国外的学者提出了匹配追踪算法，正交匹配追踪算法，逐步正交匹配算法等。使得压缩感知重构算法在时间复杂度方面获得了很大改善。

时间复杂度和重构精度是判断该算法性能好坏的条件。现有的相关算法中，在满足时间复杂度较低的时候重构精度会下降，但在保证重构精度的前提下往往又需要较高的复杂度，给后期的重构运算带来很大代价。所以怎样找到一种在保证较高重构精度的条件下让运算时间更少的重构算法，一直是国内外研究者的主要任务。

## 1.3 本文的主要研究内容和结构安排

本文主要工作在于对压缩感知重构算法进行研究，并利用MATLAB仿真软件对常用算法进行仿真，分析比较不同的算法之间的复杂度和精度。

本文的内容安排如下：

第一章是绪论部分，介绍了本课题研究的相关背景情况及意义，阐述了国内外学者对该领域的研究现状。

第二章主要讲述压缩感知相关基本理论，包括信号的稀疏表示方法、采样矩阵的研究以及压缩感知数据重构三个部分。

第三章是本文的主要部分，对常用的重构算法进行详细的理论分析，并利用MATLAB进行仿真实验，对比分析各个算法之间的复杂度和精度。

第四章是总结与展望部分。回顾本文的研究内容，对相关研究工作进行总结，分析其中的不足。

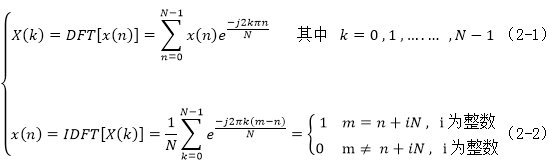
# 第二章 压缩感知基本理论

## 2.1 信号的稀疏表示

信号具有稀疏的特性是压缩感知应用的前提。然而常见的信号通常都含有大量的冗余信息，对这类信号做处理时，会因为采样数据的增加而带来较大的成本代价。因此就需要利用稀疏表示方法对信号进行稀疏处理，目前常用的稀疏表示方法如下：

(1)傅里叶变换

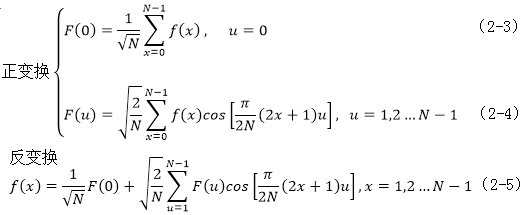
傅里叶变换实现了信号由时域到频域的转换，并借助于一系列正弦函数对原始信号进行线性组合和逼近，使信号的处理变得更加的灵活和容易。傅里叶变换虽然改变了信号的时域表示，但在频域中表示的信息与原信号所携带的信息并没有差别，不同的仅仅是表示的方式而已。有了信号的频域表示后，如果需要对频域信号进行离散化处理，这时就需要借助离散傅里叶变换（DFT）。离散傅里叶变换可以看做是傅里叶变换的离散采样，对于一个长度为M的有限长信号x(n),其N点离散傅里叶变换（DFT[x(n)]）及离散傅里叶逆变换（IDFT[x(x)]）如下：

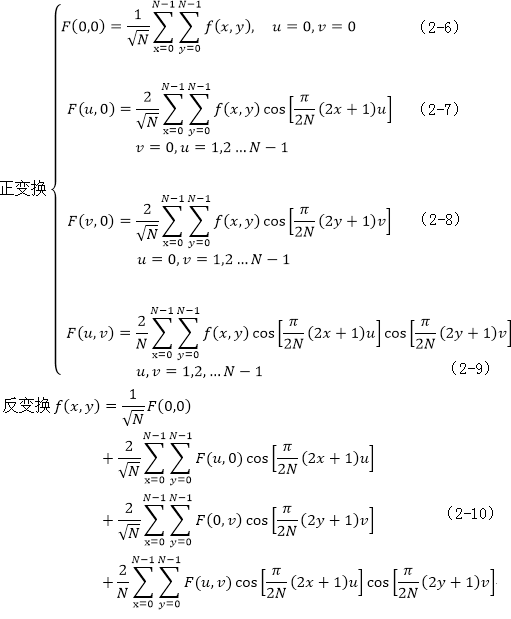


(2)离散余弦变换

同傅里叶变换一样，离散余弦变换(DCT)也是正交变换，但与傅里叶变换相比，离散余弦变换不再限制于复数的运算，而是可以在实数域进行计算，在图像处理的相关方面，也比傅里叶变换具有更好的性能。对于一维和二维信号，离散余弦变换及其反变换公式如下：

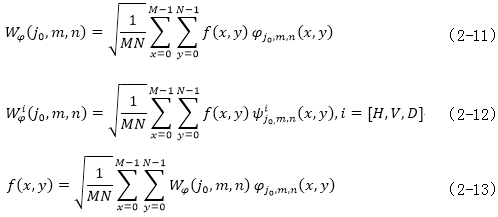
其中一维信号为f(x),x=0,1,…N-1,二维信号为f(x,y),x,y=0,1,…N-1。



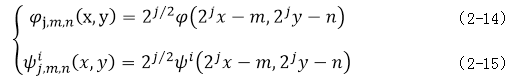


(3)小波变换

小波变换是一种在信号局部分析中被广泛运用的变换方法，与傅里叶对信号的全局表示不同，小波变换可以很好的反应信号在某一局部的特性，弥补了傅里叶变换在这方面的不足。二维离散小波相关变换公式如下：



在式2-12中，表示3个不同方向，和如下：



下面，以离散余弦变换为例，说明其在信号稀疏表示中的应用，如图1所示。

(a)一维信号y (b)一维信号DCT变换

(c) 原始图像 (d)DCT变换重构图像

图1 信号的DCT变换

图1-(a)是对一维信号的连续采样，在两个周期的时间里获得了100个采样数据，由图(a)可知，该信号在时域中的非零数值较多，不满足压缩感知理论对信号稀疏性的要求。图1-(b)是利用DCT变换对采样的数据进行处理的结果，采样数据做离散余弦变换后，绝大多数的数值都变为零，非零元素的个数很少,于是便实现了非稀疏信号的稀疏表示。

图1-(c) 是一幅256×256大小的8位灰度bmp格式图像，如果要无失真的保存该图像需要记录65536个像素点的像素值。如果对该图像做离散余弦变换，并将得到的数据中数值绝对值小于10的数设置为零，再利用更改后的数据对图像进行重构，得到的重构图像如图1-(d)所示。在进行离散余弦变换后非零点的个数为19433，在保存的时候只需要保存这些非零点的值，极大的减少了存储的信息量，但重构的图像与原始图像差别却很小。

这样利用正交变换基对非稀疏信号进行稀疏表示，不仅减少了数据存储带来的成本代价，而且还满足了压缩感知理论对信号稀疏性的要求，在包括压缩感知在内的很多领域都有重大意义。

## 2.2 采样矩阵的研究

与传统采样理论不同，压缩感知打破了先采样后压缩的限制，通过采样矩阵实现了在采样的同时进行压缩。对于一个已知信号,稀疏度为K，即x的N个数值中只有K个非零值，其中。如果利用传统采样定律进行采样，为了不丢失信号x中的元素，需要获得N个测量值，这时可以理解为采样矩阵是N阶单位矩阵。如图2所示.



图2 传统采样数学模型

在图2中，目的信号x共有N = 12个元素，其中非零值的个数为K = 3,采样矩阵Φ为12阶单位矩阵，这样通过Φ和x得到的测量值y就包含了x中的所有信息，但除了非零数据，其他的都是多余的，如果对这些数字为零的数据也进行采样保存必然造成巨大的资源浪费。

解决这一问题的关键在于采样矩阵的选取，能否构造这样一个矩阵，使得采样后得到的数据y中只包含x的非零数据的信息，或者很少包含其他冗余数据信息。y不一定是x中的所有非零元素的集合，而是x非零信息的一种表示，包含了x的全局信息。采样过程如图3所示。



图3 压缩感知采样数学模型

在图3中，对于稀疏信号x选取一个大小为的矩阵Φ作为采样矩阵。得到的数据y中共含有M个测量值，通常。这样通过新的采样矩阵Φ将x的信息保存在了y中，并且y里面的测量值远小于原始信号x的元素个数，从而实现了对信号的压缩。

上面的讨论中，默认信号x是稀疏的，包含很少的非零值，并且给出了稀疏度K，然而现实世界中的自然信号往往是非稀疏的，这时就需要利用2.1节中的信号稀疏表示方法对原始信号做处理，让目的信号在某个正交变换基Ψ下具有稀疏性，即。此时的采样模型如下：



图4 非稀疏信号采样模型

采样矩阵的好坏在于是否能够保存信号x中的有效信息，并且保证能够基于测量数据y对原始信号进行重构。为了实现这一点，采样矩阵通常需要满足一定的条件。下面以随机高斯矩阵为例，探讨其在采样和重构过程中表现出来的特性。

高斯矩阵元素满足高斯分布：



对于一幅256×256的bmp格式图片，稀疏度确定，利用高斯矩阵采样，将采样的数据用OMP算法进行重构，以测量数M为变量，观测高斯矩阵在不同M值下重构图像的信噪比。因为高斯矩阵是随机矩阵，每次测量时生成的数据都不一样，为了避免偶然误差，对每个M值重复500次重构操作，对所有的数据求平均值，得到的结果如下：

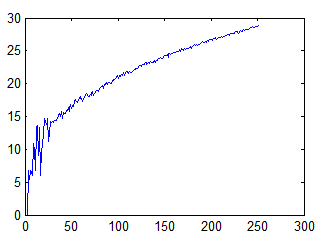


图5 高斯采样矩阵恢复信噪比

由图5可知，在测量数M不断增大的过程中，重构的信噪比不断增大，重构精度也更高。

## 2.3 压缩感知数据重构

信号，经过矩阵采样后得到，即。因此，由采样数据y和采样矩阵重构数据x的过程就是求方程的解的过程。但在式中，y的维数m远小于x的维数n，此方程属于欠定方程，在有解的情况下需要穷举信号x中的种非零值的排列可能。因此，当信号x的长度N和稀疏度K都较大时直接求解信号x是非常困难的。

压缩感知要求信号必须是稀疏的，这样，在求解的时候，加上适当的限定条件，让重构的信号尽可能的稀疏，便能得到原始信号x的近似解。理论证明，可以通过求解最优范数实现对信号x的重构：



上式中，s.t代表subject to,信号x为稀疏度为K的稀疏信号，并且为了实现精确重构信号，测量结果y中的测量数M必须满足以下条件：



而采样矩阵必须满足约束等距条件。

在求解式2-17的方法中，通常采用的是贪婪类算法，这些算法通过迭代的方式从感知矩阵中寻找与原始信号x最匹配的原子，并利用这些原子对信号进行稀疏表示，不停的向x逼近，并求出每次迭代的余量，然后针对余量继续进行最为匹配原子的寻找。由于式2-17在求解的过程中带来的复杂性和不稳定性，贪婪类算法虽然能通过迭代的方式求出x的近似解，但重构的精度较低，在精度要求较高的情况下可以使用范数代替，并且放宽约束条件，即：



这样，在知道y和Φ的情况下式2-17就转化成了一个凸最优化问题。通过一定的约束条件，进而转变为线性规划相关问题，完成从完备的基字典里找出原始信号x的稀疏表示，实现对压缩感知数据的重构。

# 第三章 压缩感知重构算法研究及仿真

## 3.1 基追踪算法(BP)

由2.3节可知，信号重构的过程转化为了一个求最小范数的问题。但由于范数问题是N-P问题，在实际应用的过程中几乎不可行，因此，通常利用范数替代，通过求解最小范数来实现对原始信号的重构。

最小范数法也被称为基追踪算法(Basis Pursuit,BP)。BP算法通过迭代的方式从采样矩阵里寻找信号的最稀疏表示，用最少的基表示稀疏信号x。考虑重构误差的因素，BP算法可表示为：



下面，利用matlab仿真软件，研究BP算法在一维离散和二维图像信号重构方面的特性。

(1)一维信号BP算法仿真实验

给定待测信号x，其长度为N=1000，并且x为稀疏度为K=50的稀疏信号，按压缩比M/N = 0.5对信号x进行处理，得到测量数为M=50的采样数据y，其中采样矩阵选择高斯随机矩阵Φ(M×N)，然后根据得到的采样数据y，和高斯矩阵Φ，利用基追踪算法进行重构，得到实验结果如下：



(a)原始信号(x)



(b)重构信号(rec\_x)



(c)重构误差(rec\_x-x)

图6 一维信号BP算法仿真结果

图6-(a)为原始信号x，其离散值随机分布在[-0.5,0.5]之间，图6-(b)是经过基追踪算法重构后得到的信号rec\_x。图6-(c)是原信号与重构信号的误差recv\_x-x,其中重构的绝对误差为0.0258，相对误差为0.0135。从图6可以看出，利用BP算法进行重构，得到的重构信号与原信号几乎没有差别。

(2)二维信号BP算法仿真实验

以二维灰度图像作为仿真原始信号x，高斯随机矩阵Φ作为采样矩阵，此处共选取3幅大小为256×256的bmp格式图像，仿真执行环境为：WINDOWS 7 64位，CORE i5 CPU，MATLAB 8.3。仿真结果如下：

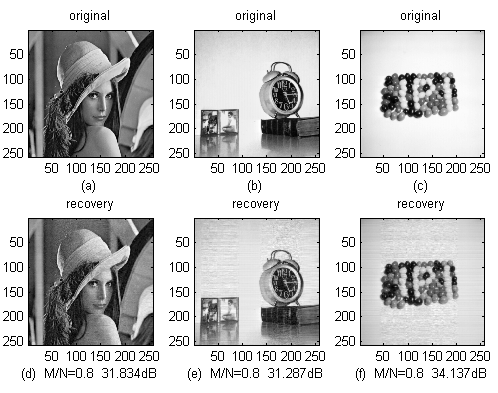


图7 二维信号BP算法仿真结果

在上图中，(a)-(c)是原始图像"lena"，"clock"，"jellbeans"，(d)-(f)是相对应的重构图像，对原图像进行采样时选择的采样矩阵大小为204×256，压缩比M/N=0.8。其中三幅图像重构的峰值信噪比分别为：31.780dB，30.641dB，34.228dB。

由图7可以看出，当压缩比M/N为0.8的时候，利用高斯采样矩阵和BP算法能够很好的实现对原图像信号的重构，重构的精度较高。

上面所讨论的是BP算法在不同的原始图像信号下，以确定的测量数和压缩比对图像进行重构，下面探讨BP算法以不同测量数进行重构的时候所表现出来的特性，结果如图8所示。

在图8中，选取"jellbeans"作为原始图像信号，压缩比M/N取值分别为0.4，0.5，0.6，0.7，0.8，分别以这些压缩比进行图像的重构，可以看出，重构的峰值信噪比跟压缩比是正相关的，并且图像重构的效果也随着压缩比的增大而更加精确，但是由于测量数M的不断增多，BP算法运行的时间不断增加。实际运用时需要根据对重构精度和运行时间的要求选择合适的压缩比。

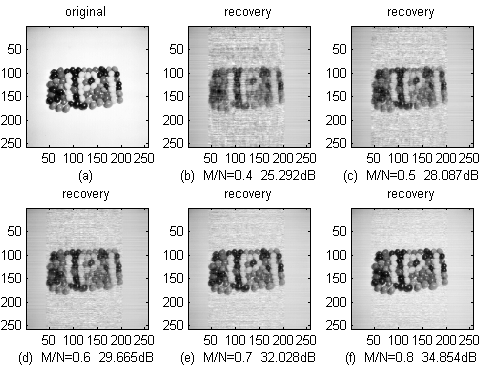


图8 不同压缩比下的BP算法仿真结果

## 3.2 匹配追踪算法(MP)

在压缩感知中，原始信号x如果是稀疏的，则采样数据可直接由采样矩阵得到：y = x；但是当x是非稀疏信号时，需要对信号x进行稀疏表示，这时采样数据为：y = 。通常将称为感知矩阵。匹配追踪算法在对信号进行重构的时候，将感知矩阵视为字典，感知矩阵中的每一列元素视为原子信号，通过对原子信号的稀疏线性组合来描述信号y。

MP算法在重构的时候通过迭代的方式每次从感知矩阵中找出与信号最为匹配的原子信号组合，对信号进行逼近，并得到信号与原子基之间的残差，找到残差之后继续寻找跟其最匹配的原子信号，重复这一步骤直到信号由一些原子信号线性表示。

MP算法详细步骤如下:( y为测量数据，为感知矩阵，r0为初始余量，k迭代计数器，为索引集合，为重建信号，K为信号稀疏度)

1. 令 = y，，，k= 1;
2. 求系数;
3. 索引集;
4. ;
5. 余量r = r – x;
6. 如果k<K，跳转到步骤(2)，否则输出重构信号。

算法流程图如下：

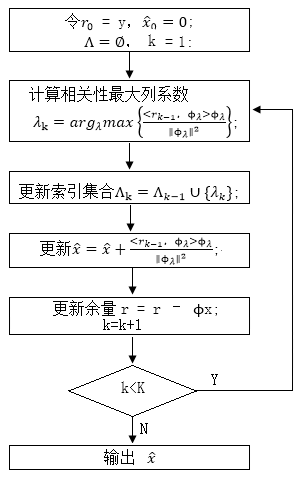


图9 MP算法流程图

下面给出MP算法在一维目的信号下的仿真结果。

给定原始稀疏信号x，长度为N=256，信号稀疏度为K=23，采样测量数据为M=80，压缩比为M/N=0.3，采样矩阵选择高斯矩阵。图10-(a)是原始信号，(b)是MP算法重构信号，(c)是重构误差。从图10-(c)可以看出MP算法不能保证重构的误差足够小，重构精度不是特别高。另外在二维图像信号的重构中，MP算法需要很多的循环计算步骤，给整个重构过程带来了很大的时间代价。因此，通常采用的是正交匹配追踪算法而不是MP算法。

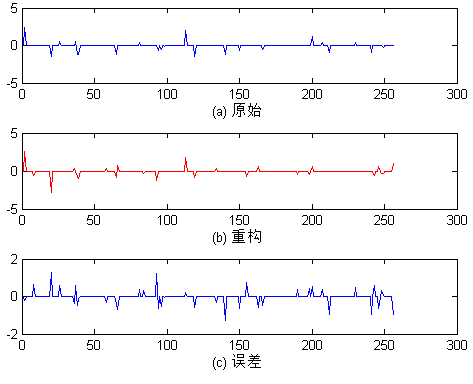


图10 MP算法一维仿真结果

## 3.3 正交匹配追踪算法(OMP)

与MP算法一样，正交匹配追踪算法(OMP)也是通过迭代的方式在感知矩阵中寻找最匹配的原子基来对信号进行稀疏逼近获得余量，然后不断的选出跟余量最匹配的原子，直到该信号可以由一些原子信号线性表示。但MP算法随着迭代次数的增加运算量逐渐增大，需要很大的循环次数才能逼近原始信号。为了解决这一问题，OMP算法通过格拉姆-施密特正交化的方式生成一个正交的列集合，这样每次更新残差的时候都是和正交的列进行比较，大大的减少的循环次数。

OMP算法详细步骤如下:(y为测量数据，A为感知矩阵，r0为初始余量，n为迭代次数，t为迭代计数器，和J为索引集合，为重建信号，K为信号稀疏度)

1. 令 = y，，，n = 2\*K，t = 1;
2. r0与A的內积：;
3. 求max{},及其位置P;
4. ，;
5. 求解方程;
6. = y - A，t = t + 1；
7. 如果迭代次数小于n，跳转到步骤(2)；否则算法停止，输出 = 。

算法流程图如下：

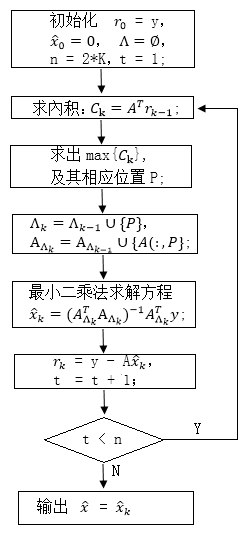


图11 OMP算法流程图

下面给出OMP算法在一维和二维信号下的仿真结果

1. 一维信号OMP算法仿真实验

一维信号仿真结果如下：

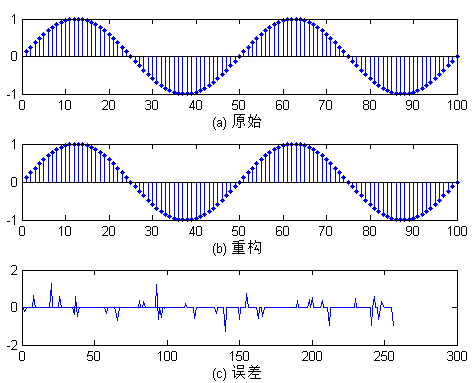


图12 一维信号OMP算法仿真

图12-(a)是长度为100的离散信号x( sin(4\*k\*pi/N)， 其中k = 1:100; N = 100 )，图12-(b)是OMP算法重建后的信号。采样过程中的的矩阵为高斯矩阵，正交变换基为傅里叶变换矩阵，正交变换后的稀疏度为K=20，采样测量值为M = 50。从图12-(c)中可以看出，当用OMP算法进行信号重构时，重构带来的误差几乎可以忽略，重构的精度很高。

1. 二维信号OMP算法仿真实验

对二维信号进行仿真时所选取的原始信号和仿真环境同3.2节。仿真结果如图11。

图13中(a)～(f)分别给出了原始二维图像信号及经过OMP算法重构后的二维图像信号。三种信号仿真中的压缩比都为0.8，对应的峰值信噪比为26.765dB，26.322dB和32.715dB。可以看出OMP算法在重构的精度上比BP算法稍差，但OMP算法的执行时间更短。

图14给出了OMP算法在不同压缩比下的重构结果。重构精度随压缩比的增大变得更高。

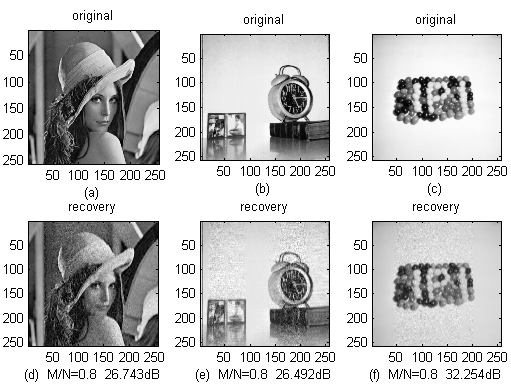


图13 二维信号OMP算法仿真

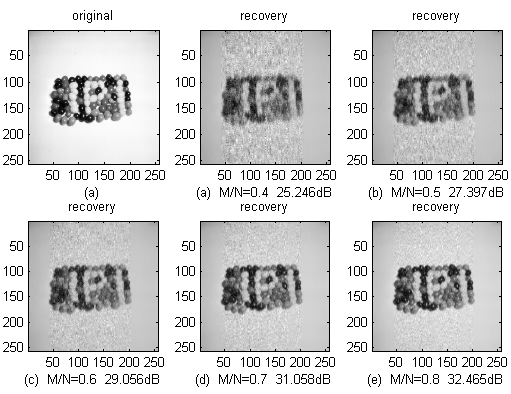


图14 不同压缩比下的OMP算法仿真结果

## 3.4 子空间追踪算法(SP)

子空间追踪算法(SP)在MP算法和OMP算法的基础上做了一些改进，不仅重构的效果更好，收敛的速度也更快。SP算法在执行的时候，需要根据感知矩阵的K个列向量生成子空间，确定测量信号y存在于哪个子空间中，然后根据确定的子空间位置就可以求出信号的非零系数。SP算法维护了一个含有K个列向量的列表，初始值为与信号y的K个最大相关列，然后在每次的迭代过程中检测这K个列向量的子集，如果重构信号与测量信号之间的余量较大，则需要更新这个列表。这样，SP算法会保留可靠地候选值，丢弃不可靠的候选值，在丢弃的时候同时增加相同数量的新值来更新列表，更新的前提是新的子空间重构后带来的余量要比原来的子空间产生的余量要小，于是，SP算法就能在每次的迭代中找到比上一次更好的子空间，完成对原始信号的重建。

SP算法详细步骤如下：(y为测量数据，为感知矩阵，y0为初始余量，m为迭代次数，l为迭代计数器，为索引集合，为重建信号，K为信号稀疏度)

1. = ;

;

1. ,;
2. ;
3. ; 
4. 如果，则，算法结束，输出；否则跳转到步骤(2)。

SP算法流程图如下：

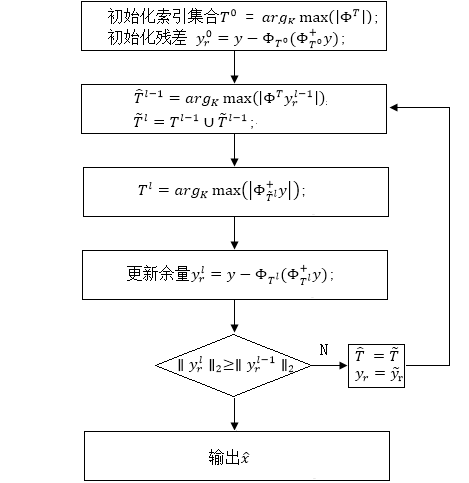


图15 SP算法流程图

下面给出SP算法在一维信号和二维信号下的仿真实验。

1. 一维信号SP算法仿真实验

给定稀疏度为K=50的稀疏信号x，长度为N=256，采样矩阵为高斯随机矩阵，测量数据量为M=204，压缩比为N/M=0.8。仿真执行环境同3.3节。仿真实验结果如图16所示。

图16-(a)～(c)分别为原始信号x，重构信号以及两者之间的误差。在图(c)中，重构误差维持在数量级，重构的精度比MP算法和OMP算法要高。

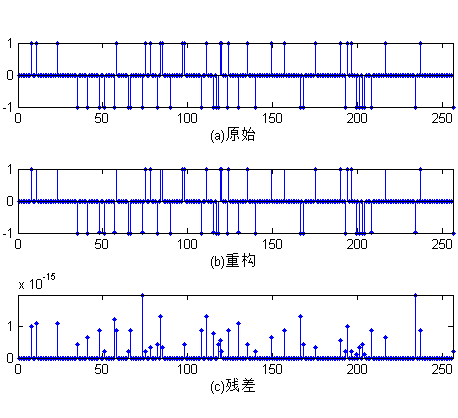


图16 一维信号SP算法仿真

1. 二维信号SP算法仿真实验

以bmp格式灰度图像作为带采样信号x，高斯随机矩阵作为采样矩阵，正交变换基选择DCT 变换，按压缩比N/M=0.8进行采样测量，然后利用SP算法进行重构，记录每次重构的峰值信噪比。仿真实验结果如图17所示。

图17-(d)～(f)为SP算法重构后的图像，对应的峰值信噪比为27.739dB，27.611dB和33.728dB。从图17可以看出，在同样的采样情况下，对于相同的压缩比，SP算法在对二维图像信号重构的时候其精度要比MP算法和OMP高。

图18给出了SP算法在同一信号，峰值信噪比随压缩比变化的仿真结果。从图可知，当压缩比不断增大的时候，峰值信噪比逐渐变大，信号重构的精度更高。

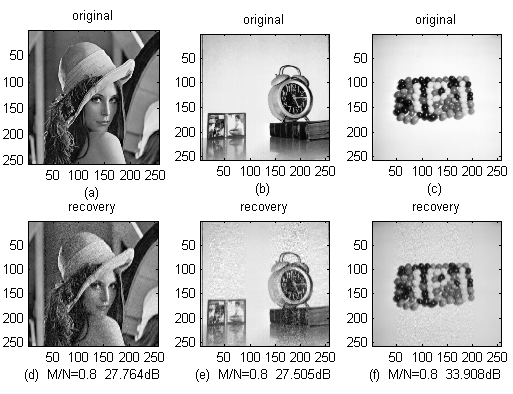


图17 二维信号SP算法仿真

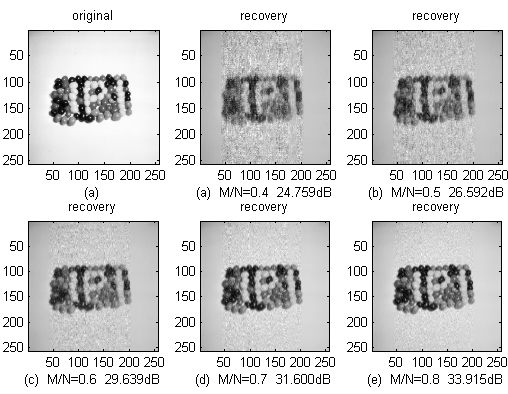


图18 不同压缩比下SP算法仿真

## 3.5 压缩采样匹配追踪算法(CoSaMP)

在MP和OMP算法中，每次只选用一个原子信号进行信号的逼近，获得残差，算法的效率较低，在SP算法中，每次选取K个列向量，然后从这K个列向量的子空间中寻找最匹配原子基，这样就提高了收敛速度，降低了算法的复杂度。压缩采样匹配追踪算法的主要思想也是不断的筛选相关的原子基，并剔除部分原子，但是保证了索引集合里的原子个数小于等于3K个，被移除的原子个数小于等于K个，并且每次迭代索引集包含了2K个原子。

CoSaMP算法详细步骤如下：(y为测量数据，为感知矩阵，为初始余量， k为迭代计数器,和J为索引集合，为重建信号，K为信号稀疏度)

1. = y, k = 1, ，J = ;
2. 计算系数 = ;
3. 求出中最大2K个值的索引存入J;
4. 更新;
5. 求解方程;

更新余量 = y - A;

1. 如果，算法结束；否则k=k+1，跳转到步骤(2)。

CoSaMP算法流程图如下：

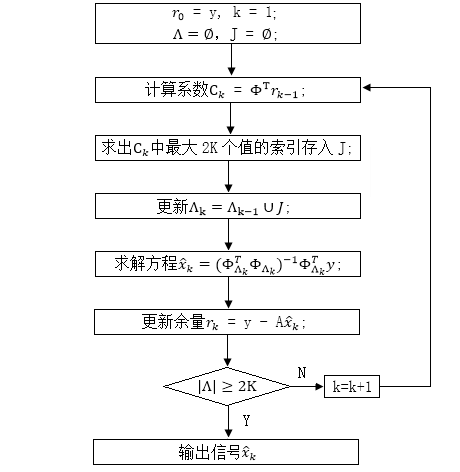


图19 CoSaMP算法流程图

1. 一维信号CoSaMP算法仿真实验

一维信号仿真时信号长度，测量压缩比，采样矩阵等条件都与3.4节相同，得到的结果如图20所示。从图20-(c)可以看出，CoSaMP算法的重构误差很小，数量级在～，近似为0。

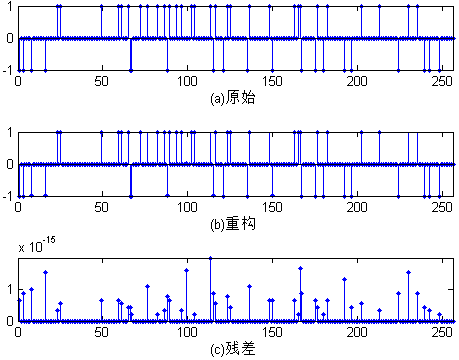


图20 一维信号CoSaMP算法仿真

1. 二维信号CoSaMP算法仿真实验

选取三幅256×256大小的bmp格式图片，正交变换选择DCT变换，压缩比M/N为0.8，并利用CoSaMP算法进行信号重构，结果如下图所示。

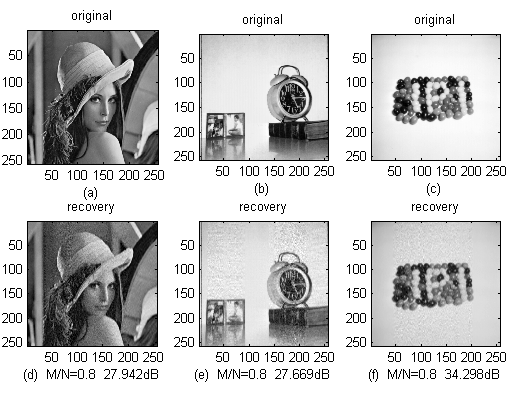


图21 二维信号CoSaMP算法仿真

在同样的采样条件下，CoSaMP算法在重构时的峰值信噪比与SP算法相差较小，两者的重构精度都比较高。图22给出了CoSaMP算法在不同压缩比下的重构结果。

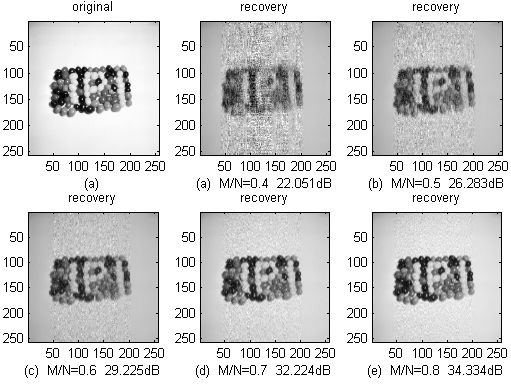


图22 不同压缩比下的CoSaMP算法仿真结果

## 3.6 算法性能比较

一种算法重构性能的好坏在于其是否有较低的时间复杂度和较高的重构精度，时间复杂度越低带来的运行时间代价越小，重构精度越高重构的效果就越好。因此在对算法性能进行分析的时候，通过研究算法的运行时间的长短来评判算法时间复杂度的好坏，通过研究算法在恢复时的峰值信噪比和重构误差的大小来评判算法重构精度的高低。

本节将对以上的重构算法进行比较，基于算法运行时间、峰值信噪比和重构误差三个方面，分析各个算法在信号重构过程中的精度和时间复杂度，总结各个算法的优缺点。

1. 不同采样率下的运行时间比较

在表1中，给出了BP算法，OMP算法，SP算法和CoSaMP算法在相同的目的信号不同的采样率下的运行时间，单位为秒(s)，采样率为0.1～0.8。

在图23中，以采样率为横坐标，运行时间为纵坐标，绘制了不同算法运行时间随采样率变化的二维曲线。从图中可以看出，BP算法在运行时间上比OMP算法要长，CoSaMP算法运行时间比SP算法长。

表1 算法运行时间(s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 算法 采样率 | BP | OMP | SP | CoSaMP |
| 0.1 | 0.749 | 0.078 | 0.437 | 0.406 |
| 0.2 | 0.85 | 0.1795 | 0.858 | 1.014 |
| 0.3 | 1.014 | 0.3355 | 1.716 | 2.1915 |
| 0.4 | 1.24 | 0.5145 | 2.73 | 3.947 |
| 0.5 | 1.451 | 0.8035 | 4.438 | 6.6145 |
| 0.6 | 2.27 | 1.139 | 7.0825 | 10.249 |
| 0.7 | 2.7065 | 1.5675 | 9.5865 | 15.1555 |
| 0.8 | 3.229 | 2.0905 | 13.361 | 22.893 |

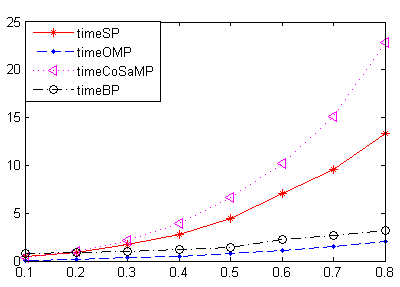


图23 不同采样率下的运行时间

1. 不同采样率下的峰值信噪比比较

表2给出了各个算法分别在相同的目的信号不同的采样率下的重构峰值信噪比，单位为分贝(dB)，采样率为0.1～0.8。

在图24中，以采样率为横坐标，峰值信噪比为纵坐标，绘制出各个算法峰值信噪比随采样率变化的二维曲线。从图中可以看出，BP算法在重构的精度方面要比其他算法高，而SP算法和CoSaMP算法在重构精度方面差别不大。

表2 峰值信噪比PSNR(dB)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 算法 采样率 | BP | OMP | SP | CoSaMP |
| 0.1 | 17.101576 | 16.8750618 | 15.4537245 | 13.9159704 |
| 0.2 | 19.563315 | 18.9563759 | 18.4944196 | 16.7670693 |
| 0.3 | 22.2229559 | 21.2317177 | 20.2307241 | 19.0417974 |
| 0.4 | 24.4868604 | 23.2738142 | 22.5569135 | 21.6834668 |
| 0.5 | 26.5345038 | 25.0536905 | 24.5991328 | 24.1985496 |
| 0.6 | 28.3480559 | 26.5566198 | 26.951376 | 26.4117078 |
| 0.7 | 30.5339896 | 28.0013884 | 28.5878874 | 28.5262477 |
| 0.8 | 33.5151931 | 29.2402972 | 30.0386322 | 30.6389808 |

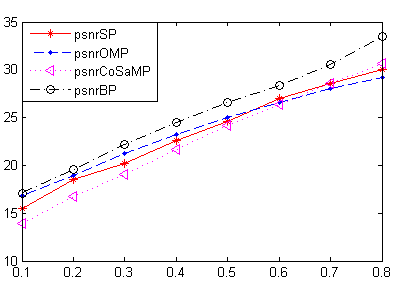


图24 不同采样率下的峰值信噪比

1. 重构误差比较

在比较重构误差时，主要比较均方误差、相对误差和绝对误差。

1. 均方误差：;
2. 相对误差：;
3. 绝对误差：;

结果如表3所示。从表3可知，BP算法在重构精度方面比其他算法要高，OMP算法的重构误差较大，而SP算法和CoSaMP算法在重构误差方面差别较小。

表3 算法重构误差

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 算法 误差 | BP | OMP | SP | CoSaMP |
| 均方误差 | 30.5948 | 75.0833 | 60.206 | 54.841 |
| 相对误差 | 0.0392 | 0.0616 | 0.0553 | 0.0527 |
| 绝对误差 | 1.42E+03 | 2.22E+03 | 1.99E+03 | 1.90E+03 |

下面根据以上仿真结果，总结各个算法的优缺点：

(1)BP算法基于求最小范数问题，优点在于重构的时候所需的测量个数少，重构的精度最高，缺点是算法运算量较大，执行效率较低，时间复杂度较高。

(2)MP算法作为贪婪类算法，通过在每次迭代的过程中寻找最匹配的原子信号实现对目标信号的重构，MP算法相对BP算法而言，时间复杂度低，但重构的精度不高。

(3)OMP算法在MP算法的基础上对选择的原子信号进行正交化处理，保证了迭代的最优性，提高了收敛速度，减少了迭代的次数。但OMP算法所需要的测量数更多，并且每次只用一个原子信号对原子集合进行更新，带来了重建时间代价。

(4)SP算法在MP算法和OMP算法的基础上做了一些改进，每次通过在K个向量的集合中寻找最匹配原子基，进一步较少迭代次数，提高收敛速度。但因为每次迭代都必须对K个向量的子集进行处理，因此SP算法在时间复杂度上比OMP算法要高。重构的误差相对较低。

(5)CoSaMP算法不同于MP和OMP算法每次只选择一个原子信号，也不同于SP算法维护K个向量的集合，而是保证了每次选择集合原子个数不超过3K个，迭代索引集合原子个数为2K个，每次剔除原子个数不超过K个，进一步提高收敛速度，CoSaMP算法的重构精度跟SP算法相差较小，但低于BP算法和OMP算法。

# 第四章 总结与展望

本文主要介绍了压缩感知技术相关知识，探讨了目前为止国内外学者对压缩感知中信号的稀疏表示、采样矩阵、重构算法等主要方向所做的研究，并阐述了这些方面的基本原理。

本文的核心研究内容和关键点在于对现有的一些压缩感知重构算法进行原理分析和仿真，列举了常用的BP算法、MP算法发、OMP算法、SP算法和CoSaMP算法，对这些算法进行了详细的理论分析，给出了每个算法的运行原理和执行步骤，制作了算法流程图，并利用一维和二维原始信号对重构算法进行仿真实验，得到各个算法的运行时间、峰值信噪比、重构误差等实验数据。

通过这些实验所得数据，分析比较了各个算法的性能，包括重构时的运行时间、信号的峰值信噪比以及重构的误差，根据各算法表现出来的特性总结了在实际运用中的优点和缺点。

当然，由于作者能力水平有限，本文还有很多相关知识没能涉及到，比如在稀疏表示中有关字典学习的知识、采样矩阵改进方法的研究以及其他复杂的重构算法等，本人将会在以后的学习中不断的完善。

# 参考文献

[1]Candes EJ，Tao T. Near-optimal signal recovery from random projections: Universal encoding strategies[J].IEEE Transactions on Information Theory，2006，52(12)：5406-6425.

[2]Prony R. Essai experimental[J]. de l’Ecole Polytechnique(Paris)，1795，1(2)：24-76.

[3]Donoho DL. Compressed sensing[J]. IEEE Transactions on Information Theory. 2006，52(4)：1289-1306

[4]Baraniuk R. Compressed sensing[J]. IEEE signal processing magazine，2007,24(4).

[5]Candes EJ. Compressed sampling[C]. Proceedings oh the International Congress of Mathematicians：Madrid，August 22-30，invited lectures，2006，pp. 1433-1452.

[6]李峰，郭毅. 压缩感知浅析[M]. 北京：科学出版社，2015.

[7]闫敬文，刘蕾，屈小波. 压缩感知及应用[M]. 北京：国防工业出版社，2015.

[8]高西全，丁玉美. 数字信号处理[M].第三版. 西安：西安电子科技大学出版社，2008.

[9]焦李成，杨淑媛，刘芳，侯彪. 压缩感知回顾与展望[N]. 电子学报，2011，39(7)：1651-1662.

[10]石光明，刘丹华，高大华，刘哲，林杰王，良君. 压缩感知理论及其研究进展[N]. 电子学报，2009，37(5)：1070-1081.

[11]戴琼海，付长军，季向阳. 压缩感知研究[N]. 计算机学报，2011,34(3)：425-434.

[12]谢晓春，张云华. 基于压缩感知的二维雷达成像算法[N]. 电子与信息学报，2010,32(5)：1234-1238.

[13]江海，林月冠，张冰尘，洪文. 基于压缩感知的随机噪声成像雷达[N]. 电子与信息学报，2011,33(3)：672-676.

[14]方红，杨海蓉. 贪婪算法与压缩感知理论[N]. 自动化学报，2011,37(12)：1413-1421.

[15]朱明，高文，郭立强. 压缩感知理论在图像处理领域的应用[N]. 中国光学，2011,4(05)：441-447.

[16]周灿梅. 基于压缩感知的信号重建算法研究[D]. 北京交通大学，2010.

[17]李维明. 基于矩阵分解的压缩感知重构算法的研究[D]. 安徽大学，2013.

[18]王智慧. 基于压缩感知的图像重构算法研究[D]. 西安电子科技大学，2012.

[19]曹离然. 面向压缩感知的稀疏信号重构算法研究[D]. 哈尔滨工业大学，2011.

[20]庞茜. 压缩感知恢复算法研究[D]. 天津大学，2011.

[21]屈冉. 压缩感知算法及其应用研究[D]. 南京邮电大学，2013.

[22]马春晖. 压缩感知重构算法研究[D]. 杭州电子科技大学，2011.

# 致谢

值此论文完成之时，向本人的指导老师任新敏老师表示深深的感谢和敬意。任老师从本课题的制定到实施以及论文的编写都提供了很大的帮助，在任老师的指导下完成了对课题相关内容的研究，掌握了课题内容的基本原理，顺利完成了论文的撰写。同时也对在仿真实验过程中提供帮助的同学表示感谢，感谢他们与我探讨软件程序相关问题。

# 附录

程序2-1

function ONE\_DIM\_SAMP()

%一维离散信号DCT变换

k = 1:100;

N = 100;

y = sin(4\*k\*pi/N);

y1 = dct2(y);

figure;

stem(y,'.');

title('y = sin(4\*k\*pi/N),k=1:100,N=100');

figure;

stem(y1,'.');

title('DCT( y )');

end

程序2-2

function DCT()

%二维离散信号DCT变换

img=imread('lena.bmp');

img\_data=dct2(img);

figure;

imshow(img);

title('original');

xlabel('(a)');

img\_data(abs(img\_data)<10)=0;

img\_rec=idct2(img\_data)./255;

figure;

imshow(img\_rec) ;

title('recovery');

xlabel('(b)');

end

程序3-1

function Gauss\_Mat = Get\_Gauss\_Mat( M, N )

%GET\_ GAUSS\_MAT 生成大小为M×N的随机高斯矩阵

Gauss\_Mat=randn(M,N);

Gauss\_Mat = Gauss\_Mat./repmat(sqrt(sum(Gauss\_Mat.^2,1)),[M,1]);

end

程序3-2

function Dct\_Mat = Get\_Dct\_Basis()

%获取DCT变换基矩阵

Dct\_Mat = zeros(256,256);

for k=0:255

tmp=cos([0:255]'\*k\*pi/256);

if k>0

tmp=tmp-mean(tmp);

end;

Dct\_Mat(:,k+1)=tmp/norm(tmp);

end

end

程序3-3

Function [rec\_x0] = bp(x,y,Phi,N)

%bp算法

Phi2 = [Phi,-Phi];

b = y;

for i =1 : N

u(i,1) = max(0,x(i));

v(i,1) = max(0,-x(i));

end

x = [ u ; v];

c = [ ones(1,length(u)) ones(1,length(v)) ];

x = linprog(c',[],[],Phi2,b,zeros(size(x)),[]);

rec\_x0 = x(1:length(u)) - x(length(u)+1:length(u)+length(v));

end

程序3-4

function [x\_rec] = mp()

%mp算法

x\_rec = zeros(N,1);

times = K;

g = zeros(N,1);

r = y;

for n=0:times

g = Phi' \* r;

[val,K] = max( abs(g) ) ;

x\_rec(K,1) = x\_rec(K,1) + g(K,1);

r = r - g(K,1) \* Phi(:,K);

end

end

程序3-5

function X=omp(Y,Compressed\_Mat,m)

%omp算法

X=zeros(1,m);

A=[];

r\_n=Y;

N=length(Y);

s=floor(n/4);

for t=1:s;

product=abs(Compressed\_Mat'\*r\_n);

[v,p]=max(product);

A=[A,Compressed\_Mat(:,p)];

Compressed\_Mat(:,p)=zeros(n,1);

a=(A'\*A)^(-1)\*A'\*Y;

r\_n=Y-A\*a;

p\_a(t)=p;

end

X(p\_a)=a;

end

程序3-6

function X=sp(Y,Compressed\_Mat,m)

%sp算法

r\_n=Y;

s\_p\_l=[];

N=length(Y);

s=floor(N/4);

for t=1:s

pd=abs(Compressed\_Mat'\*r\_n);

[v,p]=sort(pd,'descend');

s\_p\_c=p(1:s);

s\_p=union(s\_p\_c,s\_p\_l);

A\_t=Compressed\_Mat(:,s\_p);

a\_x\_c=zeros(m,1);

a\_x\_c(s\_p)=(A\_t'\*A\_t)^(-1)\*A\_t'\*Y;

[v,p]=sort(abs(a\_x\_c),'descend');

X=zeros(1,m);

X(p(1:s))=a\_x\_c(p(1:s));

s\_p\_l=p(1:s);

r\_n=Y-Compressed\_Mat\*X';

end