图像去噪中几类稀疏变换的矩阵表示

摘 要

基于稀疏表示的图像去噪是信号处理领域里的前沿问题之一，对于该问题的研究也对图像处理的发展有重要意义。本文基于含有加性噪声的灰度图像，将图像分解为各重叠的小块，并通过离散小波变换、离散余弦变换、主成分分析、奇异值分解四种方法对各小块计算稀疏系数矩阵。最后依据峰值信噪比（PSNR）和结构相似性指标（SSIM）评价分析各方法去噪功效，并提出新的稀疏去噪方法。

针对任务一，任务一包含了三个部分，首先是重叠小块的划分，将图像划分成两两重叠度为 25%的 8×8 像素块。接下来是按照四种方法对每个重叠小块进行变换，得到变换后的矩阵表达形式；随后，将四种表达形式进行分类，根据每种变换的表示形式，将离散小波变换和离散余弦变换归为形式 b，将主成分分析及奇异值分解归为形式 a；接下来，针对离散小波变换和离散余弦变换所得结果，通过构建过完备稀疏字典，然后进行稀疏分解，得到信号的稀疏系数矩阵，最后利用提取出的稀疏系数矩阵和字典做积运算，重构图像；随后，对奇异值分解所得矩阵进行稀疏化构建，即将包含图像信息的矩阵分解到一系列奇异值和奇异值矢量对应的子空间中，根据奇异值大小，通过筛选有效奇异值重构矩阵。最后，对于基于主成分分析的稀疏去噪，可以利用奇异值分解得到三个分量，基于其中一个分量包含了该影像的特征向量的特点，从中挑选出主成分中最能代表整体影像的特征向量进行回归处理，将其稀疏化，得到稀疏矩阵并重构。

针对任务二，选取一个 8×8 像素块基于四种方法进行稀疏化表示，并重构出结果图像，利用峰值信噪比对四种方法的重构性能进行判断，得到由于存在数据字典的训练过程，使得离散小波稀疏去噪的性能最佳的结论。

针对任务三，首先，根据任务一中的方法，分别对每个小块构建稀疏系数矩阵。其中离散小波变换及离散余弦变换涉及到过完备稀疏字典的建立；随后，重构出每小块去噪结果矩阵，按照重叠处取平均的原则恢复出去噪结果影像；接着，提出滑动窗口算法，对图像进行重新划分，加密处理使每幅图图像的非边缘点均有 8 个稀疏变换重构值，对 8 个值进行平均处理赋予原像素点，从而得到去噪性能更好的基于滑动窗口的稀疏去噪。最后，提出了基于独立成分分析的字典建立法，用于获得稀疏矩阵进行图像去噪的方法。

关键词：稀疏矩阵，图像去噪，稀疏字典，PSNR，SSIM

目录

一、问题重述

图像去噪是信号处理领域中的一个重要研究课题，稀疏表示理论的研究随着近年来兴起的压缩传感理论，越来越引起研究学者的重视。因此基于稀疏表示的图像去噪成为近年来该领域的一个前沿研究课题。

本题目假设了一幅二维灰度图像受到加性噪声的干扰。而通过对该图像进行稀疏表示可以达到去除噪声的目的。基于稀疏表示提出了如下任务：

1. 将图像分割为相互重叠的小块，对于讨论离散余弦变换（DCT），离散小波变换（DWT，用 DB4 小波），主成分分析（PCA）和奇异值分解（SVD）。分为以下两种形式：

(a) $(Y_{ij})_{\sqrt{m}\times\sqrt{m}} = U_{\sqrt{m}\times\sqrt{m}} D_{\sqrt{m}\times\sqrt{m}} V_{\sqrt{m}\times\sqrt{m}}$

(b) $(Y_{ij})_{m\times 1} = D_{m\times 1}\alpha_{k\times 1}$（将$Y_{ij}$堆垒为列向量的形式）

其中，下标为矩阵或者列向量的行列数。

2. 利用 Cameraman 图像中的一个小图像块进行验证。

3. 分析稀疏系数矩阵，比较四种方法的硬阈值稀疏去噪性能，并提出可能的新的稀疏去噪方法。

二、问题分析

2.1 概论

该问题为信息交叉学科题目，主要是关于数字图像处理领域的图像去噪。涉及到对稀疏图像去噪的探索。稀疏信号是指某信号可以使用有限个信号特征值对其进行表示，信号一般不具备稀疏的特性，但对信号进行某种变换，将其投影到变换域上时，信号的稀疏性就大大增强，这里的变换域也可统称为稀疏字典。基于此，对于题目中所给的含噪图像，可以先进行题目中的四种变换，在对得到矩阵表示构建稀疏模型，其中，在离散余弦稀疏去噪和小波去噪中对稀疏字典进行构建，图像在字典上进行分解，以达到去噪效果。奇异值分解和主成分分析采用模式 a 的构建方法，构建稀疏表示后，重建得到去噪图像，最后对各方法进行去噪性能评判。

2.2 任务一

任务一要求我们先将图像进行重叠分割，接下来基于四种稀疏变换矩阵表示方法，即离散小波变换、离散余弦变换、主成分分析以及奇异值分解，进行分块图像的稀疏字典的建立及稀疏化表示。其中基于离散小波变换、离散余弦变换的图像稀疏表示属于模式 b，即

$$(Y_{ij})_{m\times 1} = D_{m\times k}\alpha_{k\times 1} \tag{1}$$

而基于主成分分析及奇异值分解的图像稀疏表示属于模式 a，即

$$(Y_{ij})_{\sqrt{m}\times\sqrt{m}} = U_{\sqrt{m}\times\sqrt{m}}D_{\sqrt{m}\times\sqrt{m}}V_{\sqrt{m}\times\sqrt{m}} \tag{2}$$

离散小波变换与离散余弦变换相似，对变换后的矩阵进行首先构建稀疏字典，进行离散小波变换或离散余弦变换来构建过完备稀疏字典，然后进行稀疏分解，得到信号的稀疏系数矩阵；最后利用提取出的稀疏系数矩阵和字典做积运算，重构图像。奇异值分解并构建稀疏化的矩阵去噪，可以将包含图像信息的矩阵分解到一系列奇异值和奇异值矢量对应的子空间中，根据奇异值大小，通过筛选有效奇异值重构矩阵达到去噪的目的。主成分分析可以利用奇异值分解得到三个分量，其中一个分量包含了该影像的特征向量。挑选出主成分中前 n 个最能代表整体影像的特征向量进行回归处理，将其稀疏化，使得主成分分量能更好地反映影像的信息，从而除去原影像中的噪声。

2.3 任务二

任务二要求我们利用 Cameraman 图像中的一个小图像块，进而验证任务一建立的稀疏矩阵去噪数学模型。我们可以读入图像像素矩阵，按照任务一建立的模型进行去噪处理，得到各图像结果矩阵，并采用 PSNR 指标，测度几种去噪模型的性能。

2.4 任务三

任务三要求我们比较四种方法的去噪性能，并提出可能的新的去噪方法。衡量图像去噪的主流方法有信噪比、峰值信噪比以及结构相似性指标 SSIM。通过编程实现指标的对比与计算，从而评价四种方法的性能。接下来，根据建立的四种模型的思想，探索新的稀疏变换的去噪算法。

三、模型假设

1. 加噪后图像中噪声仅包括高斯白噪声；
2. 利用 SSIM 及 PSNR 两个指数即能够客观评价去噪性能；
3. 处理影像均为与题目中类似的方阵，即仅考虑方阵数字影像的处理；
4. 在稀疏矩阵中接近于 0 的量视为 0；
5. 原始图像所收到的噪声干扰适度，在可以去噪的范围之内。

四、符号说明

符号	符号说明
S	大小为 8x8 的图像灰度矩阵
$p_{i,j}$	代表在矩阵 S 位置(i,j)处像素的灰度值
x	S 的列向量表示
X	整幅原始图像的列向量表示
D	数据字典
α	稀疏系数矩阵
α_j	主成分对应的第 j 个向量
β	p 维参数
$\hat{V}_j$	标准化的第 j 个 p 维参数
I	稀疏去噪重构后 8×8 矩阵
U	奇异值分解后左酉矩阵
V	奇异值分解后右酉矩阵
MSE	均方根误差
PSNR	峰值信噪比

五、模型的建立与求解

5.1 问题一

5.1.1 问题一的分析

任务一的首要步骤是得到和题目要求值相同高斯噪声图像，接着对图像进行重叠小块分块处理，第二个步骤便是将每个小块矩阵进行四种方法的初步处理得到处理后矩阵，随后，根据题目所给的两种模式，根据四种方法的特点，分别构建相应的数据字典并进行稀疏化表示，得到四种方法的稀疏化表示结果。

5.1.2 高斯噪声图像合成

使用均值为 0，标准偏差为 10 的高斯噪声，采用 MATLAB 函数 Out = In + 10*randn(size(In))实现，其中 Out 为加噪声后图像，In 为原始图像。加噪声前后图像对比如下：

图 1 原始图像

图 2 加噪声图像

5.1.3 图像分块与合并

将图像分为相互重叠的小块，一一进行处理，其主要目的为：在字典学习过程中，仅有较小的字典可以被分解，并且较小的字典更能够突出图像的局部特征。采用重叠的小块可以有助于每个小块在分别处理之后的拼接中能够做到无缝镶嵌。即在组合成原来大小的图像时不在合并的边界上留下边界杂线，使得处理效果下降。

根据题目图片以及说明，本小组首先确定使用两两重叠度为 25%的 8×8 像素

块划分图像。分块示意图如图 3：

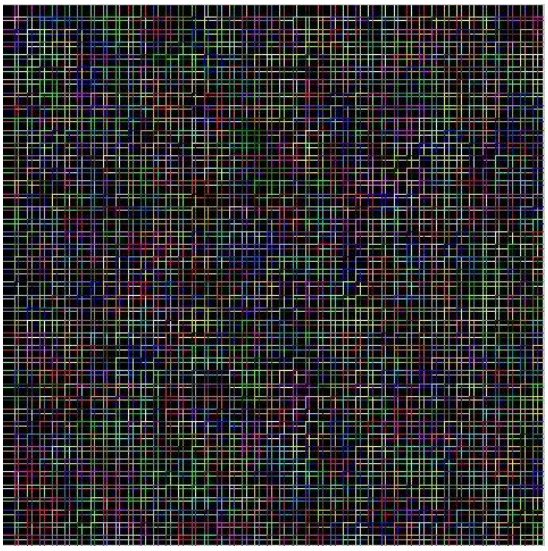

图 3 图像分块示意图

其中每个 8×8 小图像块可以用矩阵 S 表示如下：

$$S = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,8} \\ p_{2,1} & p_{2,2} & & p_{2,8} \\ \vdots & & \ddots & \vdots \\ p_{8,1} & p_{8,2} & \cdots & p_{8,8} \end{bmatrix} \quad (1)$$

其中$p_{i,j}$代表在矩阵 S 位置(i,j)处像素的灰度值。

对于每个矩阵 S，利用四种类型的稀疏分解去噪方法得到每个窗口中各像素经过去噪变换后的灰度值。由于相邻图像块之间重合度为 25%，这样重叠部分的每一个像素将会被 m（m=2 或 4）个滑动窗口所包含。对这些像素在各个包含图像块S_k中的灰度值$p_{k_{i,j}}$求算术平均，即可得到小图像块边缘重叠区域内一个像素的平均灰度值$\bar{p}$，公式如下：

$$\bar{p} = \frac{1}{m}\sum_{k=1}^{m} p_{k_{i,j}} \quad m = 2 \ or \ 4 \quad (2)$$

非重叠部分像素灰度值不变，重叠部分灰度值取平均，即可得到合并后图像各像素的灰度值。

5.1.4 离散小波变换与离散余弦稀疏模型的建立与求解

传统的小波变换和余弦变换去噪是变换域去噪方法。余弦变换作为傅里叶变换的特例，是将图像由空间域变换到频域。小波变换则是一个时间和频率的局域变换。信号经变换后，不同部分表现出不同的特性，其中信号的高频部分，其频

率分辨率较低、时间分辨率较高，而其低频部分则正好相反。相似的二者均可采用模型 b 来进行稀疏的实现。另外，由于图像的有效信息与噪声信息在经小波变换后，其在不同的小波尺度上具有不同的特点，利用小波变换能对信号进行多尺度细化分析，使得图像与噪声对应的小波系数具有较好的区分度。小波变换去噪基本方法流程图如图 4 所示。

传统的小波变换和余弦变换去噪是变换域去噪方法。余弦变换作为傅里叶变换的特例，是将图像由空间域变换到频域。小波变换则是一个时间和频率的局域变换。信号经变换后，不同部分表现出不同的特性，其中信号的高频部分，其频率分辨率较低、时间分辨率较高，而其低频部分则正好相反。相似的二者均可采用模型 b 来进行稀疏的实现。另外，由于图像的有效信息与噪声信息在经小波变换后，其在不同的小波尺度上具有不同的特点，利用小波变换能对信号进行多尺度细化分析，使得图像与噪声对应的小波系数具有较好的区分度。小波变换去噪基本方法流程图如图 4 所示。

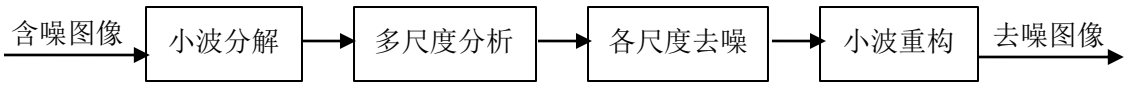

图 4 传统的小波变换去噪流程图

变换域去噪的核心思想就是通过变换将信号与噪声在变换域中根据一定的规律区分开。同样，传统的小波变换去噪方法假定在含噪图像中，有用信息频率较低，而噪声频率较高，且两者在频域有较大的区分度，实际上此假设条件并不总是成立，图像信息往往和噪声互相重叠：一方面，图像细节和边缘等有用信息含有高频分量；另一方面，噪声虽然以高频成分为主，但也含有低频成分[1]。这是造成传统变换域去噪方法具有丢失图像的边缘信息、不利于后期图像处理、影响人视觉感观等缺陷的根本原因。

近些年来，随着稀疏表示理论的不断发展，研究者们开始将稀疏表示技术应用于图像去噪。本文采用的基于过完备字典的小波（或余弦）稀疏去噪方法过程如下：首先构建稀疏字典，进行小波变换（或余弦变换）来构建过完备稀疏字典；然后进行稀疏分解，得到信号的稀疏系数矩阵；最后利用提取出的稀疏系数矩阵和字典做积运算，得到去噪后的图像矩阵，再用小波逆变换（或余弦逆变换）重建图像。其基本方法流程图如图 5 所示。

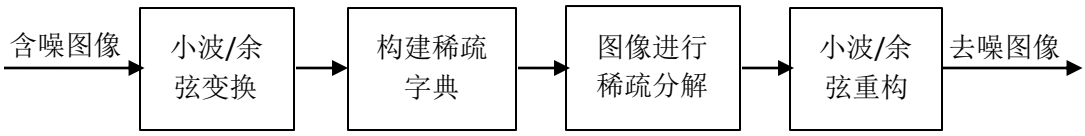

图 5 稀疏去噪流程图

为防止在分块处理后进行图像合并时边界杂线的问题，本文将原图分割成重叠度为 25%，像素大小为$\sqrt{m}\times\sqrt{m}$（m=64）的小块图像$\{Y_{ij}\}$，并在合并时将重叠部分进行平均处理。

对于每一小块8×8图像$\{Y_{ij}\}$，将其图像矩阵 S 转换为64×1的列向量$x\in R^{m\times k}$，小波变换后构建字典$D\in R^{m\times k}$，其中$m>k$，即该字典为过完备字典。

由$x=D\alpha$，得 x 基于 D 的稀疏表示为：

$$\hat{\alpha}=\arg\min\|\alpha\|_0 \tag{3}$$

为使模型容易计算，采用稀疏误差项：

$$\|x-D\alpha\|_2^2\le\varepsilon \tag{4}$$

代替$x=D\alpha$，同时定义稀疏度 s，且满足$\|\hat{\alpha}\|\le s$，加入噪声后可得如下模型：

$$\hat{\alpha}=\arg min\|x-D\alpha\|_2^2+\mu\|\alpha\|_0 \tag{5}$$

其中，μ 为惩罚因子。

对于完整的图像 X，其中每个小块图像均符合上式，则有：

$$\{\hat{\alpha}_{ij},\hat{X}\}=\arg\min\lambda\|X-Y\|_2^2+\sum\mu_{ij}\|\alpha_{ij}\|_0+\sum\|D\alpha_{ij}-R_{ij}X\|_2^2 \tag{6}$$

上式中，第一个式子用来判断原始图像 X 与含噪图像 Y 之间的趋近程度，第二个式子是对图像稀疏特性的约束，第三个式子中R_{ij}用于提取子图的矩阵，其中 ij 为图形提取的下标，$R_{ij}X$为第 ij 张子图像，$D\alpha_{ij}$为重构后的近似子图像，希望两者之差越小越好。

假设字典 D 已知，则未知量为局部稀疏系数$\hat{\alpha}_{ij}$和输出图像 X。对于这两个未知量的计算，使用块协调最小化算法分别进行：

首先，初始化 X = Y，然后寻找最优的$\hat{\alpha}_{ij}$。在这个过程中，对提取出来的小图像块，按下式进行求解：

$$\hat{\alpha}_{ij}=\arg\min\mu_{ij}\|\alpha_{ij}\|_0+\sum\|D\alpha_{ij}-R_{ij}x_{ij}\|_2^2 \tag{7}$$

这里本文采用的稀疏分解算法为正交匹配追踪算法(Orthogonal Matching Pursuit, OMP)[2]，它在每一步分解时将所选的原子进行正交化处理，再将信号在这些正交原子构成的空间上投影，得到信号在各个已选原子上的分量和残余分量，每次迭代的过程中，用最小二乘法找出与当前样本残差最接近的字典原子，然后更新残差，直至残差满足结束条件$\left\|\mathrm{D}\alpha_{\mathrm{ij}}-\mathrm{R}_{\mathrm{ij}}\mathrm{x}_{\mathrm{ij}}\right\|_2^2\leq\mathrm{T}$时，停止迭代，得出该图像块系数原子$\hat{\alpha}_{\mathrm{ij}}$。当所有$\hat{\alpha}_{\mathrm{ij}}$求出来之后，对 X 进行更新。用下式进行求解：

$$\hat{X}=\arg\min\lambda\left\|X-Y\right\|_2^2+\sum\left\|D\alpha_{ij}-R_{ij}X\right\|_2^2 \quad (8)$$

其近似解为：

$$\hat{X}=\left(\lambda I+\sum_{ij}{R_{ij}}^{T}R_{ij}\right)^{-1}\left(\lambda Y+\sum_{ij}{R_{ij}}^{T}D\hat{\alpha}_{ij}\right) \quad (9)$$

其中，I 为单位矩阵。

最后将得到的列向量还原成图像块，再进行小波（或余弦）逆变换得到去噪后图像。

5.1.5 主成分分析稀疏模型的建立与求解

主成分分析（Principal components analysis，PCA）的原理就是将原始数据投影到一个新的空间中，相当于矩阵分析里面将一组矩阵映射到另外的坐标系下。但在新的坐标系中，表示原来的样本不需要那么多的变量，只需要原来样本的最大的一个线性无关组的特征值对应的空间的坐标即可，即完成了降维。

在具体的实际状况下，通过主成分分析算法会将对象携带的 80%以上的信息集中在对应的特征值大的几个特征向量上，而剩下的特征值较小的一部分则被认为是噪声。根据经验，可以认为 5%最小特征值对应噪声，在去除这 5%最小特征值所对应的噪声信息后，剩下的 95%的特征值对应的特征向量组成的矩阵作为映射，即可充分识别对象携带的有用信息[3]。


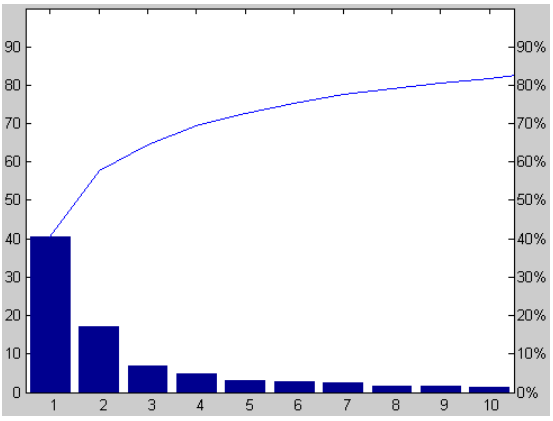

图 6 噪声影像及其 PCA 变换后各特征向量贡献率

图 6 所示，图 6 左为加入标准偏差为 10 的高斯噪声的影像，图 6 右为影像进行主成分分析变换后各特征值的贡献率，只显示了前 10 维。

主成分分析之所以能得到广泛的应用首先是因为主成分之间互不相关，其次是因为主成分能够按照顺序达到方差最大值，从而我们可以一一考虑每一个主成份而不用顾及其他。然而这也对此方法造成了一定的缺陷，比如每个主成份都是所有变量的线性组合，并且对应的单位特征向量中大部分并不为零，当我们想对主成份进行分析和解释的时候，就无法确定特定的主成份对应的特征是什么。

稀疏主成分分析对主成分分析是一个很好的改进，稀疏主成分的出现在很大程度上简化了主成分的解释，使主成分的提取结果赋予了更为实际的意义。

与小波变换的处理方法相似，稀疏主成份的算法如下[4]，设原始数据资料阵为 X：

（1）计算一般主成分的前 k 个主成分对应的向量α_j，$j = 1,2,\cdots,k$。

（2）在给定$A = (\alpha_1,\alpha_2,\cdots,\alpha_k)$ 的情况下解如下的回归问题：

$$\beta_j = argmin(\alpha_j - \beta)^T X^T X(\alpha_j - \beta) + \lambda\|\beta\|^2 + \lambda_{1,j}\|\beta\|_I \qquad (10)$$

其中，$\|\beta\|^2 = \beta^T\beta$，$\|\beta\|_I$表示β各分量的绝对值之和。λ、r、$\lambda_{1,j}$是参数，对它们给定一系列值后，相应地计算出$\beta_j$，再回头确定它们的最优值。

（3）对于给定的$B = (\beta_1,\beta_2\cdots,\beta_k)$，对矩阵进行奇异值分解：$X^TXB = UDV^T$，其中$UU^T = I$，$VV^T = I$，并且令$A = UV^T$。

这说明，主成分分析可以通过借助奇异值分解转化为题目所描述的 a 模式。

除此之外，A 即为我们所需要稀疏矩阵，通过不断循环使得稀疏矩阵最优，从而进行主成分分析的逆变换即可得到去噪的影像。

（4）重复步骤（2）、（3）至收敛。

（5）标准化$\widehat{V_j}=\frac{\beta_j}{|\beta_j|}$，j = 1,2,⋯ ,k。

5.1.6 奇异值分解稀疏模型的建立与求解

矩阵的奇异值分解定义为给定一个秩为 r 的m × n矩阵 X, 其奇异值分解形式为：

$$X = U\begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix}V^H \tag{11}$$

式中 U, V 分别为 m × m，n × n正交矩阵，E 是 r × r维对角阵，其对角线元素为矩阵 X 的非零奇异值的非增顺序排列。

对于每个8 × 8小块 S，对其进行奇异值分解变换：

$$S_{8\times8} = U_{8\times8}D_{8\times8}V_{8\times8} \tag{12}$$

这便是题目所述的 a 模式，即：

$$(Y_{ij})_{\sqrt{m}\times\sqrt{m}} = U_{\sqrt{m}\times\sqrt{m}}D_{\sqrt{m}\times\sqrt{m}}V_{\sqrt{m}\times\sqrt{m}} \tag{13}$$

然而，那些等于 0 或者接近于 0 的奇异值并没有携带重要信息，其携带的值大多为噪声值，所以，对经过奇异值分解而得到的矩阵进行重构时，可以将其忽略，那么矩阵将会稀疏化，并达到原有信息量不变的同时去除噪声的目的。

通过奇异值的分解，将图像矩阵在其奇异值分解左奇异值矩阵 U 上作正交投影，就可以将包含图像信息的矩阵分解到一系列奇异值和奇异值矢量对应的子空间中，因为噪声的能量比较小所以它对应的奇异值也比较小，可以通过去除小奇异值滤掉噪声子空间然后在有效的信号子空间上重构图像矩阵，就可以实现去除噪声的目的。同时，用较小的信息量存储了较大的图像，即对矩阵进行了稀疏化处理。

按照从上到下从左到右的顺序遍历每一块子图像，在每一块子图像中，首先可以除去不含有重构重要信息的 0 奇异值，得到含有 r 个奇异值的 S 的第一个稀疏形式：

$$S = \sum_{i=1}^{r} U_i D_i V_i \tag{14}$$

同时，对于接近零或者较小的奇异值及其对应的左右特征向量，可以想象其也同样只含有少量矩阵重构信息，在删去这些值时，便需要确定一个阈值以使之稀疏化。

但稀疏化的同时，也同样要保证其原始信息尽可能少的丢失以及噪声的去除

效果，因此，在设定阈值时，利用每个阈值来计算重构后图像的峰值信噪比 PSNR：

$$\mathrm{MSE} = \frac{\sum_{n=1}^{F}(I^n - S^n)^2}{F} \tag{15}$$

$$\mathrm{PSNR} = 10 \times \log_{10}\frac{255^2}{MSE} \tag{16}$$

其中 S 表示原始图像在 n 像素点灰度值，I 表示重构后图像在 n 像素点灰度值，MSE 代表均方误差。

由于奇异值阵 D 具有沿对角线依次降落的特性，因此在选阈值 T 的循环中可以每次滑动一个单元，对于选取的一个小块，可以得到不同稀疏化的 PSNR 值，下图 7 为一个具体小块 PSNR 曲线：

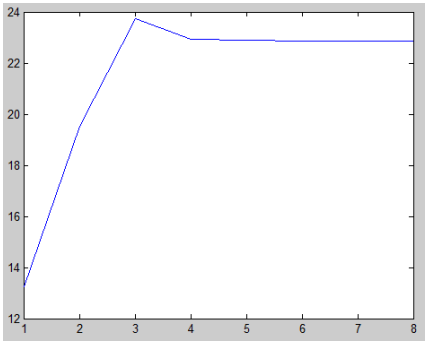

图 7 图像小块 PSNR 曲线

由曲线可以容易看出，在将原奇异值分解后的矩阵保留其中的 3 个主奇异值，得到缩减了数据量的稀疏矩阵，即可较好的恢复原图，并同时去噪。

综上所述，每个小块的稀疏表示并重构的公式为：

$$S_{8\times 8} = U_{8\times T} D_{T\times T} V_{T\times 8} \tag{17}$$

上式相当于，将 D 矩阵的除去前 T 个对角线元素均置零，使其变为稀疏系数阵。

5.2 任务二

5.2.1 任务二的分析

任务二要求我们利用 Cameraman 图像中的一个小图像块进行验证，因此，我们选取了 144-151 行、167-174 列的8×8图像块，对于验证性能，本小组采用最经典的峰值信噪比来检验。

PSNR 整个意思就是到达噪音比率的顶点信号。在影像去除噪声后，需要判断输出的影像同原始影像相近的程度。为了衡量经过去噪后的影像品质，我们通常会参考 PSNR 值来认定某个处理去噪算法够不够令人满意。其计算公式为：

$$\mathrm{MSE} = \frac{\sum_{n=1}^{F}(I^n - S^n)^2}{F} \tag{18}$$

$$\mathrm{PSNR} = 10 \times \log_{10}\frac{255^2}{MSE} \tag{19}$$

基于以上公式利用峰值信噪比及稀疏系数矩阵对该小块进行了验证。

5.2.2 任务二结果

原噪声图像，其 PSNR = 22.8551dB：

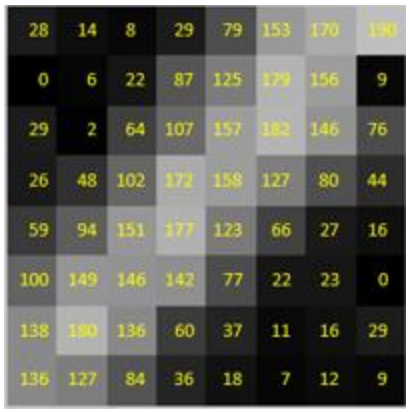

图 8 原噪声图

离散余弦变换稀疏去噪结果图，其 PSNR=23.2358dB:

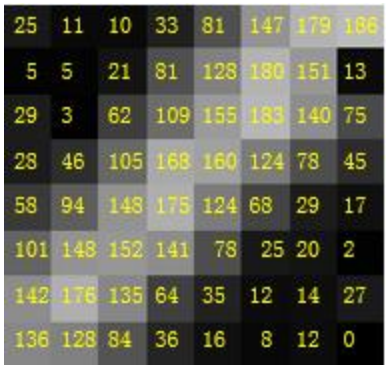

图 9 离散余弦变换稀疏去噪

小波变换稀疏去噪结果图，其 PSNR=23.7047dB:

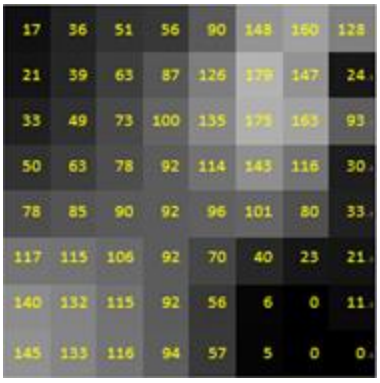

图 10 小波变换稀疏去噪

主成分分析稀疏去噪结果图及，PSNR=22.8632dB:

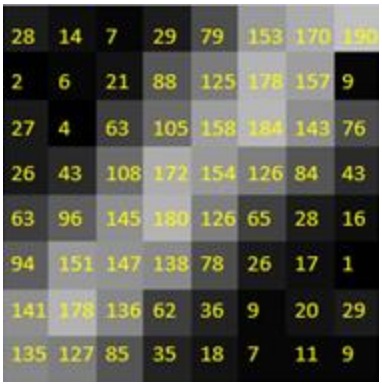

图 11 主成分分析稀疏去噪

奇异值分解稀疏去噪结果图，其 PSNR=23.7341dB:

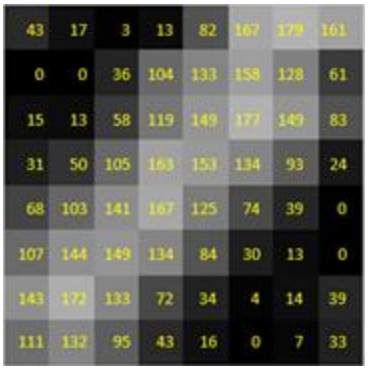

图 12 奇异值分解稀疏变换

根据上述结果总结于下表：

表 1 稀疏去噪方法峰值信噪比对比

稀疏去噪方法	PSNR
原噪声图像	22.8551dB
离散余弦变换稀疏去噪	23.2358dB
小波变换稀疏去噪	23.7407dB
主成分分析稀疏去噪	22.8632dB
奇异值分解稀疏去噪	23.7341dB

结果分析：

根据以上结果，我们可以发现，四种方法的去噪后图像的 PSNR 均与原噪声图像无明显差别，但并不能说这种微小的差别表明稀疏去噪的性能存在问题，这是因为，本次稀疏去噪测试是基于8×8图像块的，像素个数及范围都较小，很难测定出较为客观的去噪性能，不过，从小窗口中便可以看出：利用小波变换的稀疏去噪要优于其他三种方案，因此可以看出基于数据字典的稀疏化处理可以使小波变换稀疏化处理结果较为完善，即在一定的稀疏度情况下，达到去除噪声的最优性能，其他没有经过数据字典训练过程，其结果便差强人意。

5.3 任务三

5.3.1 任务三的分析

任务三要求我们分析四种方法的稀疏系数矩阵，进而比较四种方法的硬阈值去噪性能。那么便需要我们首先按照任务一的分块方法将整张图像进行分块，随后分别进行四种方法的稀疏去噪，再按照前述合并方法，将各小块进行合并，最后得到去噪后的结果图，并评价去噪性能。

5.3.2 任务三的模型建立与求解

首先，分块计算离散余弦变换的稀疏表达矩阵及数据字典，并重构出去噪结果矩阵，随后计算 PSNR 峰值信噪比。

除峰值信噪比外，在图像去噪处理的相似度评价上，结构相似性指标 SSIM 指标能够全面超越峰值信噪比。下面对结构相似性指标 SSIM 进行介绍。

结构相似性指标 SSIM 是一种用以衡量两张数位影像相似程度的指标。当两

张影像其中一张为无失真影像，另一张为失真后的影像，二者的结构相似性可以看成是失真影像的影像品质衡量指标。相较于传统所使用的影像品质衡量指标，结构相似性在影像品质的衡量上更能符合人眼对影像品质的判断。

在计算两张影像的结构相似性指标 SSIM 时，会开一个局部性的视窗，一般为$N \times N$的小区块，计算出视窗内信号的结构相似性指标，每次以像素为单位移动视窗，直到整张影像每个位置的局部结构相似性指标都计算完毕。将全部的局部结构相似性指标平均起来即为两张影像的结构相似性指标。这种分小块的方法恰好与任务一中分小块的思想相一致，编程运算也可更为方便。计算 SSIM 的公式如下：

$$\mathrm{SSIM}(x,y) = \frac{(2u_x u_y + C_1)(2\sigma_{xy} + C_2)}{(u_x^2 + u_x^2 + C_1)(\sigma_x^2 + u\sigma_y^2 + C_2)} \qquad (20)$$

分别编程实现离散余弦变换重构去噪图的以上两个指标 PSNR、SSIM 的计算，并将重构所得去噪图显示如下：

图 13 DCT 去噪结果

离散余弦变换 PSNR = 32.2774 SSIM = 0.8214

同样计算离散小波变换的稀疏表达矩阵及数据字典，并重构出去噪结果矩阵，编程计算 PSNR 峰值信噪比，结构相似性指标 SSIM。得到效果图如下：

图 14 DWT 去噪结果

离散小波变换 PSNR = 32.3393 SSIM = 0.8333

计算基于奇异值分解的稀疏矩阵，重构出去噪结果矩阵，编程计算 PSNR 峰值信噪比，结构相似性指标 SSIM 如下：

图 15 SVD 去噪结果

奇异值分解 PSNR = 31.6145 SSIM = 0.8202

计算基于主成分分析的稀疏矩阵，重构出去噪结果阵，编程计算 PSNR 峰值信噪比，结构相似性指标 SSIM：

图 16 PCA 去噪结果

主成分分析分解 PSNR = 31.3393 SSIM = 0.8207

以上四类结果均为将图像分成8×8图像块，并设定重叠度 25%去噪得到，考虑到为进一步提高性能，可以缩小滑动窗口，但缩小滑动窗口后会使稀疏化表示出现问题，因此，小组想到可以利用滑动窗口思想，进而使去噪能力更强，去噪效能更好。

滑动窗口的主要流程如下：

对于每个8×8窗口，利用四种类型的稀疏分解去噪方法重构出每个窗口中各元素去噪后的灰度值。这样在整幅图像中每一个像素（除去边缘像素外）将会被 8 个滑动窗口所包含。对于每个像素灰度值$S_{i,j}$，随着窗口的滑动，每个窗口会计算出不同的去噪后的值，即每个像素有 8 组不同的去噪后值 Sn：

$$Sn_{i,j}^{k}=function(S_{i,j}^{k}),k=1\ldots8 \tag{21}$$

其中 function 表示对原始每个像素进行稀疏化去噪处理的模型。

而最终获取的去噪后图像仅存在一组灰度值，因此，将每个像素 8 组灰度值进行如下平均计算，得到每个像素最终处理后灰度值 Sf。

$$Sf_{i,j}=\frac{1}{8}\sum_{k=1}^{8}Sn_{i,j}^{k} \tag{22}$$

而上述方法略去了边缘的处理，因此，本小组采用如下方法进行边缘的处理：由于位于每幅图像四个边缘向内七个像素将会无法被足够多的滑动窗口覆盖，小组根据位置不同的行和列分别减小参与计算像素值个数，并适当缩小窗口，最终得到了较好的效果。四种方法的去噪效果图及其峰值信噪比 PSNR 及结构相似性指标 SSIM 如下：

图 17 滑动窗口 DCT 去噪结果

滑动窗口离散余弦变换 PSNR = 34.3143 SSIM = 0.8773

图 18 滑动窗口 DWT 去噪结果

滑动窗口离散小波变换 PSNR = 34.5712 SSIM = 0.8808

图 19 滑动窗口 SVD 去噪结果

滑动窗口奇异值分解 PSNR = 33.7799　SSIM = 0.8609

图 20 滑动窗口 PCA 去噪结果

滑动窗口主成分分析 PSNR = 33.7246　SSIM = 0.8567

将以上通过不同方法得到的 PSNR 显示如下图(其中横轴从左到右依次为：离散余弦变换稀疏去噪 PSNR、离散小波变换稀疏去噪 PSNR、奇异值分解稀疏去噪 PSNR、主成分分析稀疏去噪 PSNR)：

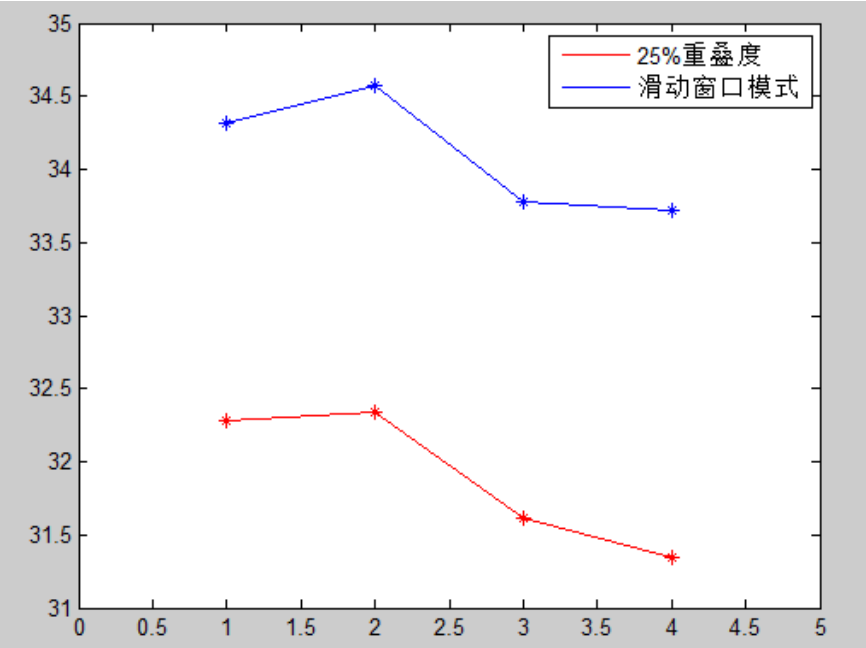

图 21 两种窗口方式的 PSNR 对比

将以上通过不同方法得到的 SSIM 显示如下图(其中横轴从左到右依次为：离散余弦变换稀疏去噪 SSIM、离散小波变换稀疏去噪 SSIM、奇异值分解稀疏去噪 SSIM、主成分分析稀疏去噪 SSIM)：

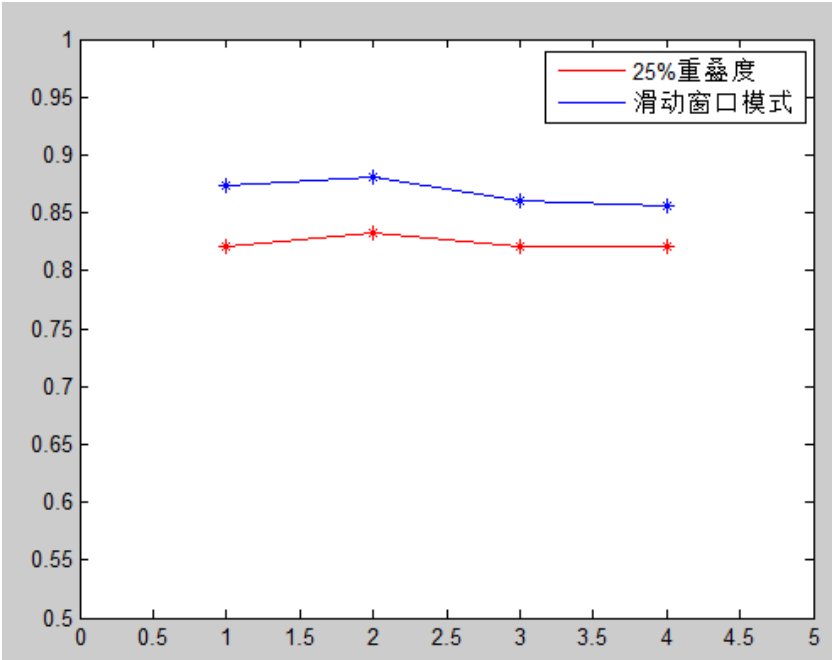

图 22 两种窗口的 SSIM 对比

通过以上两个图形可以看出，两个指标基于四种不同方法所得值变化趋势大体相同，且滑动窗口模式的去噪性能要优于 25%重叠度的去噪性能，这是因为滑动窗口模式对每个像素都进行了平均处理，而 25%重叠度模式仅对镶嵌边进行平

滑处理，当然，从时间复杂度上来讲，滑动窗口模式以时间复杂度为代价换取到去噪性能的增强。另外，从两个图形中也可以发现：无论是哪个指标，抑或是哪种平均方法，小波变换的去噪性能都是最好的。其原因如下两点：小波变换自身便具有变尺度特性，并对确定信号具有一种集中的能力，该能力要优于其他去噪方法涉及的算法；第二，小波稀疏去噪中采用了过完备稀疏字典，该字典对于稀疏矩阵的优化生成具有一定的辅助作用。

5.3.3 稀疏去噪新方法的提出

正如本文分析的四种方法那样，通过将空间域的影像转换到其他域并进行稀疏表示，还原图像去噪的方法近些年来得到迅速的发展。而最先对稀疏表示进行实验的便是著名的单一小波字典收缩算法。

从一些文献中我们发现，当将独立成分分析用于自然影像时，独立成分分析和稀疏编码等效。然而，独立成分分析更注重于输出系数中稀疏的独立性，而稀疏编码则要求输出系数必须是稀疏的，只需尽可能的独立。而由于自然影像的稀疏结构，稀疏编码相对与独立成分分析更加适用于自然影像。因此，稀疏编码的方法被广泛地应用与图像的处理中。

现在最流行的稀疏信号模型，在另一个角度上，认为一些字典里少量的原子可以准确地表示信号。很多有效的图像复原方式就是利用了从原始影像上学习得到的字典，来计算出被分为小块后的影像中每一块的稀疏表示。

然而，字典的学习具有高度的计算复杂度，并且其数学上的解释并不是特别完备。而在前述的字典中，实际的稀疏影像表示被计算为非线性的估计。这种非线性的估计会因为字典本身的相干性使得去噪结果不稳定或不精确。

我们认为，为了获得过完备字典，可以使用独立成分分析技术去捕获用于字典学习的表征影像主要成分的数据的结构。

在字典的学习中使用独立成分分析出于以下两点原因：

1.独立成分分析拓展了概率框架学习过完备字典的基础，这一点依赖于使用工作于顺序模式的快速独立成分分析算法。

2.由于选择了合适的非线性函数，使得与预先设定好稀疏度的稀疏表示联系在一起的基础学习方程能够在不影响整体结构的情况下得以进行。

然而，基础学习方程的结构取决于我们事先选择的置于稀疏表示系数的概率密度函数。我们认为线性模型$y = Dx$是有效的；在该模型中，y 和 x 是随机的矢量（我们设定数据矩阵 Y 的列向量为y_i，简记为 y），D 是我们想要估计的基础矩

阵。现在我们仅考虑完整的情型（D 是n × n的方阵，y 和 x 为 n 维的列向量）。编码矩阵 X 的提取（也可以被看作从盲源中分离目的矩阵）可以通过使用独立成分分析算法来完成。

对于盲源分离问题，$y = Dx$，最小化互信息 I(x)：

$$I(x) = \sum_{i=1}^{n} H(x_i) - H(y) - log|detD^{-1}| \quad (23)$$

$H(x_i)$代表不同的熵的表示，H(y)代表数据的联合熵。

独立成分分析算法中，流动于非线性网络的最大化信息，独立成分分析模型$y = Dx$的最大似然，或者最小化$x = D^{-1}y$组成成分之间的互信息，这些方法在某种意义上等同于最小化 I(x)并且产生新的D^{-1}。

$$D^{-1}(i+1) \leftarrow D^{-1}(i) + \eta\,[I - \phi x(k)x(i)^T)]D^{-1}(i) \quad (24)$$

在顺序模式的快速独立成分分析中，基向量会被逐一估算。在每一次迭代之后，基向量与之前格莱姆-施密特正交化估计的基向量是正交的。这种想法能被拓展后用在完整的情况，如下所示：

$$d_i \leftarrow d_i - \alpha \sum_{j}^{i-1} (d_i^T d_j) d_j \quad (25)$$

并且字典利用公式（23）进行更新。

重建：重建除噪后的影像$x = D^{-1}y$。

六、模型评价

6.1 模型优点

1. 小波变换稀疏去噪及离散余弦变换稀疏去噪充分采用数据字典进行稀疏化表示，并借用 OMP 算法的思想，能够较好地利用图像像素间灰度的关系，进而使稀疏化表示更加合理。

2. 评价去噪性能时，同时比较两种去噪性能指数 PSNR 及 SSIM，使得评价结果更加客观准确。

3. 不仅采用了重叠矩阵的方法进行全图像处理，亦利用滑动窗口法对全图像进行了处理，比较发现，滑动窗口法的性能更加优异。

6.2 模型缺点

1. 稀疏矩阵的构建利用硬阈值进行构建，然而，硬阈值是将绝对值小于阈值的小波系数置为零，而将绝对值大于阈值的小波系数保留。硬阈值能取得较好的去噪效果，但由于其不连续，所以在处理含有丰富边缘信息的图像时会产生许多额外噪声。

2. 小波变换和余弦变换稀疏去噪虽然采用了字典的方法，效果较好，但该字典是固定字典方法，虽然去噪速度较快，但没有结合去噪图像本身的特性，去噪效果有限，且不具备很好的适应性。

6.3 模型改进

由于本模型中使用的固定字典存在上述缺点，因此一种改进方法为使用基于学习型字典：通过对图像信号进行机器学习得到学习型字典，这样结合了图像的自有特征，效果应当更加理想。

具体思路流程为在构建字典时添加训练字典的环节，流程如下图：

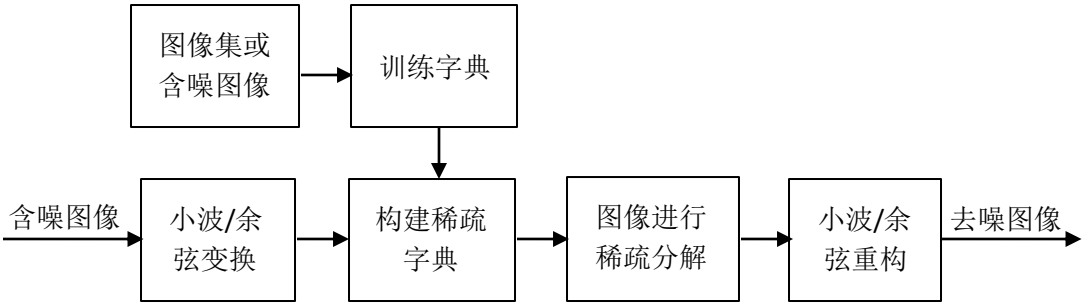

图 1 基于学习型字典的去噪流程图

在训练字典的环节中，可以先选取小波/余弦变换得到的固定字典作为初始字典，然后输入需要进行训练的图像集合或含噪图像，基于当前字典进行系数编码，若此稀疏表示的误差大于一定阈值则更新字典并进行下一次循环。

由于进行训练的过程计算量庞大，考虑到时间有限，本小组并没有进行字典学习，不过进行字典训练理论上能够大大提高图像去噪的效果。

七、 参考文献

[1] 尹忠科,解梅,王建英．基于稀疏分解的图像去噪[J].电子科技大学学报，2006, 35(6) : 876-878．

[2] Pati Y C, Rezaiifar R, Krishnaprasad P S. Orthogonal matching pursuits: recursive function approximation with applications to wavelet decomposition[C]. Conference Record of the 27th Asilomar Conference on Signals, Systems & Computers, IEEE, Los Alamitos, CA, USA, 1993, 1:40-44.

[3]宣士斌．带权稀疏PCA算法及其应用(J) ．重庆大学学报, 2014, 37(4):47-51

[4]喻胜华,张新波．稀疏主成分在综合评价中的应用(J)．财经理论与实践(双月刊),2009,30(161): 106-109

八． 附录

使用软件：MATLAB

8.1 图像分块及滑动窗口代码

```
clc; clear all; close all;
A = imread('D:\matlab\R2012a\bin\Cut1.tif');
B = imread('D:\matlab\R2012a\bin\NoiseCut1.tif');
L = size(A);
window = 8;
sum = 0;
seg_origin = zeros(window);
seg_noise = zeros(window);
Value = zeros(window);


NewImg = zeros(L(1),L(2));

First_SVD=SVD(double(A),double(B),window);

right_limit = L(1)-window;
bottom_limit = L(2)-window;
for i=window:bottom_limit
    for j=window:right_limit

        left = i-window+1;
        right = left+window-1;
        top = j-window+1;
        bottom = top+window-1;
        count1=window;
        count2=window;
        for i1=top:bottom
            for j1 = left:right
                seg_origin = double(A(i1:i1+window-1,j1:j1+window-1));
                seg_noise = double(B(i1:i1+window-1,j1:j1+window-1));

                Value = WaveletDenoising(seg_noise,2);
```

```
                sum = sum+Value(count1,count2);
                count2 = count2-1;
            end
            count2 =window;
            count1 = count1-1;
        end
        sum = sum/(window*window);
        NewImg(j,i) = sum;
    end
end

for i=1:L(1)
    for j=1:L(2)
        if(i<window||j<window||(j>L(2)-window+1)||(i>L(1)-window+1))
            NewImg(i,j) = FirstSVD(i,j);
        end
    end
end


NewImg = uint8(NewImg);
imshow(NewImg);
```

8.2 小波稀疏变换代码

```
function outputImg = sparseWaveletDenoise(X)
[a,b]=size(X);
%  小波变换矩阵生成
ww=DWI(a);
%  小波变换让图像稀疏化（注意该步骤会耗费时间，但是会增大稀疏度）
X1=ww*sparse(X)*ww';
threshold=3*10^(-4);%阈值
X1=X1.*sparse(abs(X1)>(threshold*norm(abs(X1),1)));  %  大于阈值的元素保留,i
X1=full(X1);
%  随机矩阵生成
M=190;
R=randn(M,a);
%  测量
Y=R*X1;
```

```
%  OMP算法
X2=zeros(a,b);  %  恢复矩阵
for i=1:b  %  列循环
    rec=omp(Y(:,i),R,a);
    X2(:,i)=rec;
end
X3=ww'*sparse(X2)*ww;  %  小波反变换
X3=full(X3);
outputImg = X3;
errorx=sum(sum(abs(X3-X).^2));        %  MSE误差
psnr=10*log10(255*255/(errorx/a/b))   %  PSNR
end

%  OMP的函数
%  s-测量；T-观测矩阵；N-向量大小
function hat_y=omp(s,T,N)

Size=size(T);                                  %  观测矩阵大小
M=Size(1);                                     %  测量
hat_y=zeros(1,N);                              %  待重构的谱域(变换域)向量
Aug_t=[];                                      %  增量矩阵(初始值为空矩阵)
r_n=s;                                         %  残差值

for times=1:M/4;                               %  迭代次数(稀疏度是测量的
    for col=1:N;                               %  恢复矩阵的所有列向量
        product(col)=abs(T(:,col)'*r_n);       %  恢复矩阵的列向量和残差的
    end
    [val,pos]=max(product);                    %  最大投影系数对应的位置
    Aug_t=[Aug_t,T(:,pos)];                    %  矩阵扩充
    T(:,pos)=zeros(M,1);                       %  选中的列置零（实质上应该
    aug_y=(Aug_t'*Aug_t)^(-1)*Aug_t'*s;        %  最小二乘,使残差最小
    r_n=s-Aug_t*aug_y;                         %  残差
    pos_array(times)=pos;                      %  纪录最大投影系数的位置

    if (norm(r_n)<9)                           %  残差足够小
        break;
    end
end
hat_y(pos_array)=aug_y;                        %  重构的向量
end
```

```
function ww=DWI(N)

[h,g]= wfilters('db4','d');        %  分解低通和高通滤波器

% N=256;                           %  矩阵维数(大小为2的整数幂次)
L=length(h);                       %  滤波器长度
rank_max=log2(N);                  %  最大层数
rank_min=double(int8(log2(L)))+1;  %  最小层数
ww=1;    %  预处理矩阵

%  矩阵构造
for jj=rank_min:rank_max

    nn=2^jj;

    %  构造向量
    p1_0=sparse([h,zeros(1,nn-L)]);
    p2_0=sparse([g,zeros(1,nn-L)]);

    %  向量圆周移位
    for ii=1:nn/2
        p1(ii,:)=circshift(p1_0',2*(ii-1))';
        p2(ii,:)=circshift(p2_0',2*(ii-1))';
    end

    %  构造正交矩阵
    w1=[p1;p2];
    mm=2^rank_max-length(w1);
    w=sparse([w1,zeros(length(w1),mm);zeros(mm,length(w1)),eye(mm,mm)]);
     ww=ww*w;

     clear p1;clear p2;
end
end
```

8.3 离散余弦变换及 DCT 数据字典生成代码

```
bb=8; % block size
RR=4; % redundancy factor
K=RR*bb^2; % number of atoms in the dictionary

sigma = 10;
pathForImages ='';
imageName = 'cameraman.tif';
```

```
[IMin0,pp]=imread(strcat([pathForImages,imageName]));
IMin0=im2double(IMin0);


%sigma = 25;
%IMin0 = imread('NNOI.tif');
IMin0=im2double(IMin0);
if (length(size(IMin0))>2)
    IMin0 = rgb2gray(IMin0);
end
if (max(IMin0(:))<2)
    IMin0 = IMin0*255;
end
IMin=imread('outout.tif');
IMin=double(IMin);
%IMin=IMin0+sigma*randn(size(IMin0));
PSNRIn = 20*log10(255/sqrt(mean((IMin(:)-IMin0(:)).^2)));
%=========================================================================
%  PERFORM  DENOISING  USING  OVERCOMPLETE
%                    DCT  DICTIONARY
%=========================================================================

[IoutDCT,output] = denoiseImageDCT(IMin, sigma, K);
%PSNRIn=psnr(IMin0,IMin);
%PSNROut=psnr(IMin0,IoutDCT);
PSNROut = 20*log10(255/sqrt(mean((IoutDCT(:)-IMin0(:)).^2)));
figure;
subplot(1,3,1); imshow(IMin0,[]); title('原图');
subplot(1,3,2); imshow(IMin,[]); title(strcat(['噪声图,PSNR= '
```

```
[IoutDCT,output] = denoiseImageDCT(IMin, sigma, K);
%PSNRIn=psnr(IMin0,IMin);
%PSNROut=psnr(IMin0,IoutDCT);
PSNROut = 20*log10(255/sqrt(mean((IoutDCT(:)-IMin0(:)).^2)));
figure;
subplot(1,3,1); imshow(IMin0,[]); title('原图');
subplot(1,3,2); imshow(IMin,[]); title(strcat(['噪声图,PSNR= ',
    num2str(PSNRIn),'dB']));
subplot(1,3,3); imshow(IoutDCT,[]); title(strcat(['DCT字典,PSNR= ',
    num2str(PSNROut),'dB']));
figure;
I = displayDictionaryElementsAsImage(output.D, floor(sqrt(K)),
floor(size(output.D,2)/floor(sqrt(K))),bb,bb,0);
title('The DCT dictionary');
```

8.4 奇异值分解稀疏去噪代码

```
X1 = imread('cameraman.tif');
Out = X1 + uint8(10*randn(size(X1)));
figure(1)
imshow(Out);
A1=double(Out);
figure(2);
imshow(X1);
X1=double(X1);
[PSNR2, MSE] = psnr(X1,A1);
[s,v,d]=svd(double(A1));
MSE = 0;
PSNR1=ones(256,1);
for value=1:1:256
recv3=s(:,1:value)*v(1:value,1:value)*d(:,1:value)';
C = recv3;% svd取最高的value个特征值进行恢复
[PSNR, MSE] = psnr(X1, C);
PSNR1(value)=double(PSNR);
end
[mirm,id]=max((PSNR1))
figure(4);
xx=1:1:256;
plot(xx,PSNR1);
figure(3);
CK=s(:,1:mirm)*v(1:mirm,1:mirm)*d(:,1:mirm)';
imshow(uint8(CK));
```

8.5 主成分分析稀疏去噪代码

```
clc; clear all; close all;
noise_cut = imread('F:\Study\数模相关\NOISESE.tif');
origin_cut = imread('F:\Study\数模相关\NNOT.tif');
[m,n] = size(noise_cut);
noise_cut = double(noise_cut);
origin_cut = double(origin_cut);
[pc,score,latent,tsquare] = princomp(noise_cut);
latent=100*latent/sum(latent)
result = cumsum(latent)./sum(latent);
X_mean=mean(noise_cut);
X_adjust = noise_cut;
for i = 1:m
    X_adjust(:,i) = noise_cut(:,i)-X_mean(i);
end
tranMatrix = pc(:,1:5);
X1 = X_adjust*tranMatrix*tranMatrix';
for i=1:m
    X1(:,i) = [X1(:,i)+X_mean(i)];
end
X_re = X1*pc*pc';
psnr_pca = psnr(origin_cut,X_re);
subplot(1,2,1);
imshow(uint8(X_re));
```

```
clc; clear all; close all;
noise_cut = imread('F:\Study\数模相关\NOISESE.tif');
origin_cut = imread('F:\Study\数模相关\NNOT.tif');
[m,n] = size(noise_cut);
noise_cut = double(noise_cut);
origin_cut = double(origin_cut);
[pc,score,latent,tsquare] = princomp(noise_cut);
latent=100*latent/sum(latent)
result = cumsum(latent)./sum(latent);
X_mean=mean(noise_cut);
X_adjust = noise_cut;
for i = 1:m
    X_adjust(:,i) = noise_cut(:,i)-X_mean(i);
end
tranMatrix = pc(:,1:5);
X1 = X_adjust*tranMatrix*tranMatrix';
for i=1:m
    X1(:,i) = [X1(:,i)+X_mean(i)];
end
X_re = X1*pc*pc';
psnr_pca = psnr(origin_cut,X_re);
subplot(1,2,1);
imshow(uint8(X_re));
```

8.6 计算峰值信噪比 PSNR 代码

```
function [PSNR, MSE] = psnr(X, Y)
% 计算峰值信噪比PSNR、均方根误差MSE
% 如果输入Y为空，则视为X与其本身来计算PSNR、MSE

if nargin<2
    D = X;
else
    if any(size(X)~=size(Y))
        error('The input size is not equal to each other!');
    end
    D = X-Y;
end
MSE = sum(D(:).*D(:))/prod(size(X));
PSNR = 10*log10(255^2/MSE);
```

8.7 计算结构相似性指标 SSIM 算法

```
function [mssim, ssim_map] = ssim(img1, img2, K, window, L)
if (nargin < 2 | nargin > 5)
   ssim_index = -Inf;
   ssim_map = -Inf;
   return;
end
if (size(img1) ~= size(img2))
   ssim_index = -Inf;
   ssim_map = -Inf;
   return;
end
[M N] = size(img1);
if (nargin == 2)
   if ((M < 11) | (N < 11))
    ssim_index = -Inf;
    ssim_map = -Inf;
      return
   end
   window = fspecial('gaussian', 11, 1.5); %
   K(1) = 0.01;      % default settings
   K(2) = 0.03;      %
   L = 255;                                  %
end
if (nargin == 3)
   if ((M < 11) | (N < 11))
    ssim_index = -Inf;
    ssim_map = -Inf;
      return
   end
   window = fspecial('gaussian', 11, 1.5);
   L = 255;
   if (length(K) == 2)
```

```
   L = 255;
   if (length(K) == 2)
      if (K(1) < 0 | K(2) < 0)
     ssim_index = -Inf;
     ssim_map = -Inf;
     return;
      end
   else
    ssim_index = -Inf;
    ssim_map = -Inf;
    return;
   end
end
```

```
    img2 = img2(1:f:end,1:f:end);
end
C1 = (K(1)*L)^2;
C2 = (K(2)*L)^2;
window = window/sum(sum(window));
mu1   = filter2(window, img1, 'valid');
mu2   = filter2(window, img2, 'valid');
mu1_sq = mu1.*mu1;
mu2_sq = mu2.*mu2;
mu1_mu2 = mu1.*mu2;
sigma1_sq = filter2(window, img1.*img1, 'valid') - mu1_sq;
sigma2_sq = filter2(window, img2.*img2, 'valid') - mu2_sq;
sigma12 = filter2(window, img1.*img2, 'valid') - mu1_mu2;
if (C1 > 0 & C2 > 0)
   ssim_map = ((2*mu1_mu2 + C1).*(2*sigma12 + C2))./((mu1_sq + mu2_sq + C1)
else
   numerator1 = 2*mu1_mu2 + C1;
   numerator2 = 2*sigma12 + C2;
denominator1 = mu1_sq + mu2_sq + C1;
   denominator2 = sigma1_sq + sigma2_sq + C2;
   ssim_map = ones(size(mu1));
   index = (denominator1.*denominator2 > 0);
   ssim_map(index) = (numerator1(index).*numerator2(index))./(denominator1(
   index = (denominator1 ~= 0) & (denominator2 == 0);
   ssim_map(index) = numerator1(index)./denominator1(index);
end
mssim = mean2(ssim_map);
```