数据挖掘课程论文

BP神经网络在图像识别中的应用及改进

学号：1310100

姓名：姚智元

专业：应用数学

南开大学数学科学学院

2017 年 1 月 20 日

摘要

数字图像识别是近几年来的热门话题，其目的是最终让机器代替人去实现图像中物体的识别。较之于图片中的三维物体，文字等二维图像的识别更加容易，而由于数字是特殊的较为简单的一类文字，利用机器学习方法实现数字的识别是数字图像识别的基础。

人工神经网络是人类在现代生物神经学基础上，模仿大脑内部细胞结构而提出的一种算法。人工神经网络由人工神经元作为基本结构，多个神经元前后相连形成人工神经网络。人工神经网络具有高速、鲁棒性和快速学习性等优良特性，这些特性使其非常适合来解决简单的图像识别问题。

本文将应用BP人工神经网络来解决简单的手写数字图像识别，给出了BP神经网络的权值更新量的推导，并对比神经网络的结构和梯度下降法的各种改进方法，例如使用非精确搜索寻找步长、使用冲量避免收敛至局部最优等方法，对分类器学习速度，分类准确性的影响。并使用了五折交叉验证等验证方法对神经网络进行了检验。分析算法在实现的过程中所遇到的问题和可能的解决方法。

关键字：人工神经网络　梯度下降法　交叉验证

目录

1 前言

1.1 课题的研究背景和意义

21世纪是一个充满信息的时代，图像作为人类感知世界的视觉基础，是信息传输的主要载体。正是因为图像带给人们的相关信息，使得图像识别技术随着计算机技术、多媒体技术的飞速发展取得了长足的进步。

传统的图像识别技术，很多是基于大规模计算的基础之上的，在运算量和运算精度之间存在着不可调和的矛盾。因人工神经网络技术其分布式信息存储和大规模自适应并行处理满足了对大数据量目标图像的实时处理要求，其高容错性又允许大量目标图像出现背景模糊和局部残缺。相对于其他方法而言，利用神经网络来解决图像识别问题有3个优点：1）神经网络对问题的先验知识要求较少；2）可以实现对特征空间较为复杂的划分；3）适用于高速并行处理系统来实现。图像识别理论的发展与计算机科学的发展是分不开的，近年来计算机科学发展迅速，但要它直接感知声音、文字、图像等外界信息仍然十分困难。因此，需要以人工智能、神经网络为核心开辟新的研究领域来解决这一难题，开展神经网络图像识别理论的研究就显得尤为重要。

1.2 实验想法来源及我的改进

该项目最初来源于慕课网站*Coursera*中，斯坦福大学公开课：机器学习(Andrew Ng)，第三周课程：*Logistic Regression*课后作业。原项目中给出了BP神经网络的前向传播基于MatLab的实现。

我所做的工作：

1. 给出了基于R语言的算法实现。
2. 给出了梯度下降法中权重矩阵变化量的数学推导。
3. 改进了梯度下降法中步长的选择方法。
4. 编写了包含常数项的多隐藏层的神经网络的完整程序。

2 人工神经网络

2.1 人工神经网络介绍

人工神经网络（Artificial Neural Networks,ANN）是在现代神经生物学研究基础上提出的模拟生物过程以反映人脑某些特性的计算结构。神经元及其突触是神经网络的基本器件。因此，模拟生物神经网络应首先模拟生物神经元。

在人工神经网络中，神经元常被称为“处理单元”。有时从网络的观点出发常把它称为“节点”。人工神经元是对生物神经元的一种形式化描述，它对生物神经元的信息处理过程进行抽象，并用数学语言予以描述；对生物神经元的结构和功能进行模拟，并用模型予以表达。

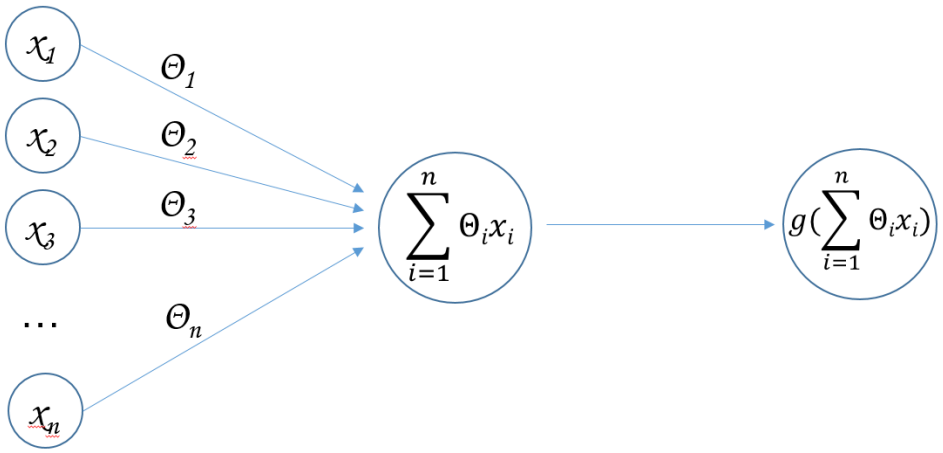

图 1: 人工神经元

一个人工神经元有4个主要组成部分：

1. 输入单元；
2. 各个输入单元的权重向量；
3. 求和单元。将输入单元中的数据根据权重计算加权和；
4. 挤压函数。将加权和限制在一定范围内。

在本次实验中，挤压函数采用sigmoid函数，即

$$g(z) = \frac{1}{1+e^{z}}$$

通常所说的人工神经网络结构，主要指它的连接方式。从拓扑结构上考虑，神经网络属于以神经元为节点，以节点间的连接为边的一种图。一个神经网络的拓扑结构确定后，为了使它具有某种智能特性，必须有相应的学习方法与之配合。权值如何设置是区分不同人工神经网络学习算法的重要特征。本实验中，主要采取确定结构的反馈型神经网络实现权重学习及预测。

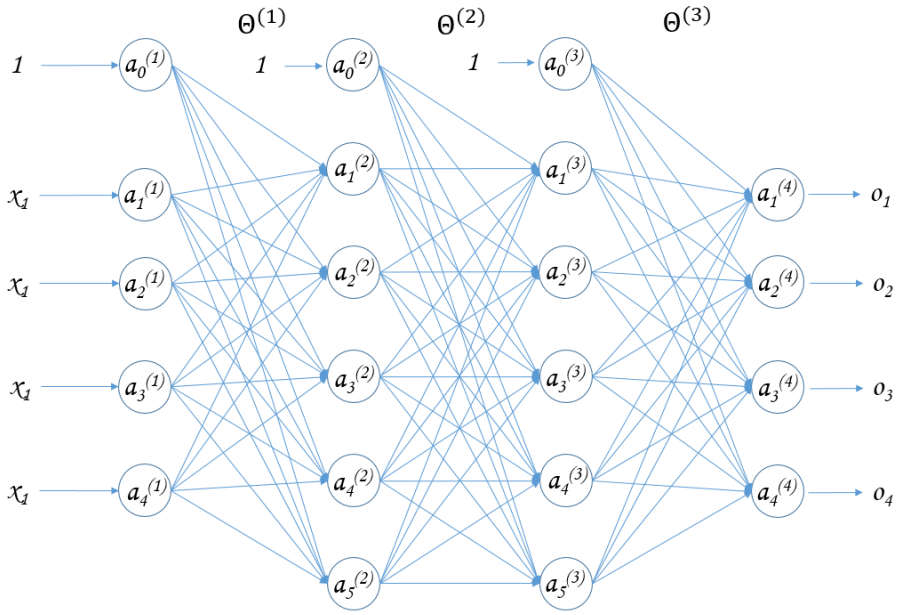

图 2: 人工神经网络

上图是一个4输入4输出，两个隐藏层的神经网络示意图，其中 $\Theta^{(1)}, \Theta^{(2)}, \Theta^{(3)}$ 为前后两层之间的权重矩阵。 $a_0^{(i)}$ $(i = 1, 2, 3)$ 称为偏差项，通常将常数1输入至偏差项中。

2.2 人工神经网络的应用及特点

人工神经网络在众多领域得到了广泛的应用。在民用领域的应用，如语言识别、图像识别与理解、计算机视觉、机器人的智能故障检测、实时语言翻译、企业管理、市场分析、决策优化、物资调运、自适应控制、专家系统、智能接口、神经生理学等，在军用领域的应用，如雷达、声纳的多目标识别与跟踪、战场管理和决策支持系统、军用机器人控制各种情况、信息的快速录取、分类与查询、导弹的智能引导、保密通信、航天器的姿态控制等。

神经网络的广泛应用主要得益于它的几个突出优点：

1. 可以充分逼近任意复杂的非线性关系；
2. 所有定量或定性的信息都等势分布贮存于网络内的各神经元，故有很强的鲁棒性和容错性；
3. 采用并行分布处理方法，使得快速进行大量运算成为可能；
4. 可学习和自适应不知道或不确定的系统；
5. 能够同时处理定量、定性知识。

2.3 算法推导

2.3.1 梯度下降法

求解无约束优化问题：

$$min \quad f(x)$$

设 $f(x)$ 是 R_n 上面连续可微函数。

对一迭代点 x_k，记 $g_k = \nabla f(x_k)$。在一个点 x_k 处的下降方向是指这样一个方向 d，它使得不等式 $f(x_k + t_d) < f(x_k)$ 对 $t > 0$ 充分小时成立。若取 $d = -g_k$，则由此得到的算法就是最速下降法。

故更新方程为：

$$X' = X + \alpha g_k = X - \alpha \nabla f(X)$$

2.3.2 前向传播及误差函数

将第 i 个样本传入网络中，对于网络中第 l 层中第 j 个节点的值有：

$$a_{ji}^{(l)} = g(net_{ji}^{(l)}) = g\Big(\sum_{t\in s(l-1)} \theta_{tj}^{(l)} \cdot a_{ti}^{(l-1)}\Big) \qquad for\ all\ l, j$$

最终误差函数：

$$J(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left(y_k^{(i)} log\Big(o_{ik}(\Theta)\Big) + (1-y_k^{(i)}) log\Big(1-o_{ik}(\Theta)\Big)\right)$$

其中

$$o_{ik}(\Theta) = a_{ik}^{(L)} = h_\Theta(a_{ik}^{(L-1)}) = \frac{1}{1+e^{(-a_i^{(L-1)}\Theta^{(L-1)})_k}}$$

表示第i个样例训练前向传播得到的第k个输出结果。

2.3.3 反向传播及梯度下降法

$$\frac{\partial J}{\partial \theta_{ij}^{(l)}} = \frac{\partial J}{\partial net_j^{(l+1)}} \cdot \frac{\partial net_j^{(l+1)}}{\partial \theta_{ij}^{(l)}}$$

其中

$$net_j^{(l+1)} = \sum_{t\in layer\ l} a_t^{(l)} \theta_{tj}^{(l)}$$

故

$$\frac{\partial net_j^{(l+1)}}{\partial \theta_{ij}^{(l)}} = a_j^{(l)}$$

定义：

$$\delta_k^{(l)} = \frac{\partial J}{\partial net_k^{(l)}}$$

对于输出层：

$$\delta_k^{(L)} = \frac{\partial J}{\partial net_j^L} = \frac{\partial J}{\partial g(net_j^{(L)})} \cdot \frac{\partial g(net_j^{(L)})}{\partial net_j^{(L)}} \qquad (1)$$

由 $g(z) = \frac{1}{1+e^{-z}}$ ，易知：

$$g'(z) = g(z)(1 - g(z)) \qquad (2)$$

并且

$$\frac{\partial J}{\partial g(net_j^{(L)})} = -\left(y_j \frac{1}{a_j^{(L)}} + (1 - y_j)\frac{-1}{1 - a_j^{(L)}}\right) = -\left(\frac{y_j - a_j^{((l))}}{a_j^{(l)}(1 - a_j^{(l)})}\right) \qquad (3)$$

由(1)(2)(3)式可得：

$$\delta_k^{(L)} = a_j^{(L)} - y_j$$

对于隐藏层：

$$\begin{aligned}
\delta_j^{(l-1)} &= \sum_{k \in s(l)} \frac{\partial J}{\partial net_k^{(l)}} \cdot \frac{\partial net_k^{(l)}}{\partial net_j^{(l-1)}} \\
&= \sum_{k \in s(l)} \delta_k^{(l)} \cdot \frac{\partial net_k^{(l)}}{\partial g(net_j^{(l-1)})} \cdot \frac{\partial g(net_j^{(l-1)})}{\partial net_j^{(l-1)}} \\
&= \sum_{k \in s(l)} \delta_k^{(l)} \cdot \theta_{jk}^{(l-1)} \cdot a_j^{(l-1)} \cdot (1 - a_j^{(l-1)})
\end{aligned}$$

故：

$$\delta_j^{(l)} = \begin{cases} a_j^{(l)} - y_j & , l = L \\ \sum\limits_{k \in s(l+1)} \delta_k^{(l+1)} \cdot \theta_{jk}^{(l)} \cdot a_j^{(l)} \cdot (1 - a_j^{(l)}) & , l < L \end{cases}$$

$$\frac{\partial J}{\partial \theta_{ij}^{(l-1)}} = a_i^{(l-1)} \delta_j^{(l)} \qquad for\ all\ i, j, l$$

$$\triangle \theta_{ij}^{(l-1)} = -\alpha \frac{\partial J}{\partial \theta_{ij}^{(l-1)}} \qquad for\ all\ i, j, l$$

2.4 人工神经网络中的基本改进方法

2.4.1 惩罚项

在使用上述算法学习权重时，很容易出现过拟合的现象。为了避免这种现象，在误差函数中加入一个随着权值向量的模增长的项（）不包括偏差项的权重），这将使得梯度下降法搜索使得这个向量模较小的权值向量，从而减小过拟合风险。

将误差函数修改为：

$$J(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left(y_k^{(i)} log\left(o_{ik}(\Theta)\right) + (1-y_k^{(i)}) log\left(1-o_{ik}(\Theta)\right)\right) + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s(l)}\sum_{j=1}^{s(l+1)}(\Theta_{ij}^{(l)})^2$$

从而，每次样例输入后，权重的更新值改为：

$$\delta_j^{(l)} = \begin{cases} a_j^{(l)} - y_j & ,l = L \\ \sum\limits_{k\in s(l+1)} \delta_k^{(l+1)} \cdot \theta_{jk}^{(l)} \cdot a_j^{(l)} \cdot (1-a_j^{(l)}) & ,l < L \end{cases}$$

$$\frac{\partial J}{\partial \theta_{ij}^{(l-1)}} = \begin{cases} a_i^{(l-1)}\delta_j^{(l)} & for\ all\ i,l;j=0 \\ a_i^{(l-1)}\delta_j^{(l)} + \lambda\theta_{ij}^{(l)} & for\ all\ i,l;j\neq 0 \end{cases}$$

$$\triangle\theta_{ij}^{(l-1)} = -\alpha\frac{\partial J}{\partial \theta_{ij}^{(l-1)}} \qquad for\ all\ i,j,l$$

2.4.2 Armijo准则

在使用梯度下降法时，如果选择固定的步长，若该步长较小，会使得收敛速度太慢；反之步长太大，会导致无法收敛在极值点，故引入Armijo准则实现动态选择合理的步长 α .

设 f_x 连续可微，d_k 是 f_x 在 x_k 处的下降方向，给定 $\rho \in (0, \frac{1}{2})$， $\beta \in (0,1)$，$\tau > 0$。设 m_k 是使得下式

$$f(x_k + \beta^m \tau d_k) \leqslant f(x_k) + \rho\beta^m \tau g_k^T d_k$$

成立的最小非负整数，令 $\alpha = \beta^{m_k}\tau$。因为 $g_k^T d_k < 0$，当 m 充分大时，上式是成立的。

Armijo准则是一种简单实用的非精确一维搜索方法，在算法的收敛性分析中比较方便，因而在各类算法中得到了广泛的实用。Backtracking线搜索时Armijo 搜索的一种特殊情形，它是指对给定 $\alpha \in (0, 0.5)$, $\beta \in (0, 1)$。寻找满足

$$f(x_k + \beta^m d_k) \leqslant f(x_k) + \alpha\beta^m g_k^T d_k$$

最小的非负整数m，然后令 $\alpha_k = \beta^m$。

应用在本问题中，对给定 $\rho \in (0, \frac{1}{2})$， $\beta \in (0, 1)$， $\tau > 0$，寻找满足

$$J(\Theta + \beta^m \tau D) \leqslant J(\Theta) - \rho\beta^m \tau D^T D$$

其中

$$D = grad(J) = \frac{\partial J}{\partial \Theta}$$

最小的非负整数m，最后权值更新方程变为

$$\triangle\Theta = -\beta^m \tau \frac{\partial J}{\partial \Theta}$$

2.4.3 带冲量项的梯度下降法

因为在多隐藏层的神经网络中，无法保证误差函数的凸性，所以误差函数可能存在多个局部最小值。而梯度下降法容易收敛在离初始点最近的局部最小值中，故在梯度下降法更新权值时加入冲量项，减小算法收敛在局部最小值的风险。

冲量还具有在梯度不变的区域逐渐增大搜索步长的优点，从而使得算法加速收敛。

在计算权值更新量时加入冲量项：

$$\triangle\theta_{ij}^{(l-1)}(n) = -\alpha a_i^{(l-1)} \delta_j^{(l)} + \eta \triangle\theta_{ij}^{(l-1)}(n-1)$$

2.5 统计检验方法

2.5.1 The Test Set Method

随机抽取50%的样本作为训练集，另50%的样本作为测试集。使用训练集学习一定次数或达到中止条件后，分别计算网络对训练集和测试集分类的准确率：

$$accuracy = \frac{r}{r+w}$$

其中r代表分类正确个数，w代表分类错误个数。

2.5.2 K-fold Cross Validation

该算法的过程为：

1. 将样本分为K等份
2. 取其中1份为测试集，剩下K-1份为训练集，对神经网络进行训练
3. 测试上述训练后的神经网络对训练集和测试集的正确率或MSE
4. 进行2，3步K次，计算所得训练集和测试集的正确率或MSE的平均值

K折交叉检验能有效的用较小的计算代价，达到避免浪费样本的目的。在本次实验中，均取K=5.

3 实验设计

3.1 实验工具

利用R Version 3.3.2, RStudio Version 0.99.896

3.2 数据说明

本实验采用手写数字图像数据[1]。该数字图像为 20×20 的图像共五千张[2] 。

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	4	8	10	7	8	1	1	10	10	1
[2,]	3	7	4	7	9	4	7	1	8	10
[3,]	2	6	8	8	6	5	10	10	7	4
[4,]	9	9	1	3	1	2	7	8	10	5
[5,]	6	5	6	5	10	8	7	4	6	4
[6,]	3	8	1	1	5	6	10	8	9	3
[7,]	6	1	1	3	4	4	8	8	1	8
[8,]	4	10	7	1	5	3	5	5	4	1
[9,]	2	8	8	5	8	9	6	9	7	9
[10,]	10	10	5	6	6	6	6	9	4	2

图 3: 左图为100个样本实例　　图 4: 右图为样本对应的数字

编程时，将样本处理为为 5000×400 的矩阵，每行为一个样本；每个样本的400个参数分别为对应像素点的灰度（0-1,全黑为0，全白为1）.真值处理为 5000×10 的矩阵，每行为一个样本的真值；每个样本真值是一个10维向量，每一位分别代表0-9中的一个数字。真值对应数字为1，其他位为0。例如真值7表示为：

$$(0, 0, 0, 0, 0, 0, 0, 1, 0, 0)$$

3.3 实验过程及结果

本次实验先随机抽取2500个样本作为训练集，剩下的2500个样本为测试集。

3.3.1 基本BP神经网络（对照组）

使用一层有200个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,200,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$ 的平均随机数；迭代最大次数设置2000次；终止条件为每个权值偏导数平均小于 10^{-5} 或达到最大迭代次数时停止；梯度下降法步长为1.

[1]This is a subset of the MNIST handwritten digit dataset (http://yann.lecun.com/ exdb/mnist/)
[2]为编程处理方便，数据中用10代表0

实验结果：

```
$test
[1] 0.9384
$train
[1] 0.996
$‘NumOfIter:‘
[1] 1999
$‘TheCostIS:‘
[1] 0.09219581
```

测试集准确率 93.84% ;训练集准确率99.60%;最终迭代了1999次，最终误差函数值为0.0922。

3.3.2 加入惩罚项的神经网络

使用一层有200个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,200,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$ 的平均随机数；迭代最大次数设置4000次；终止条件为每个权值偏导数平均小于 10^{-5} 或达到最大迭代次数时停止；梯度下降法步长为1；设置惩罚项，系数 λ 设置为 0.5.

实验结果：

```
#加入惩罚项
$test
[1] 0.942
$train
[1] 0.9992
$‘NumOfIter:‘
[1] 4000
$‘TheCostIS:‘
[1] 0.1911812
```

```
#对照组
$test
[1] 0.9404
$train
[1] 1
$‘NumOfIter:‘
[1] 4000
$‘TheCostIS:‘
[1] 0.02734707
```

测试集准确率 94.20% ;训练集准确率99.92%;最终迭代了4000次，最终误差函数值为0.1912。

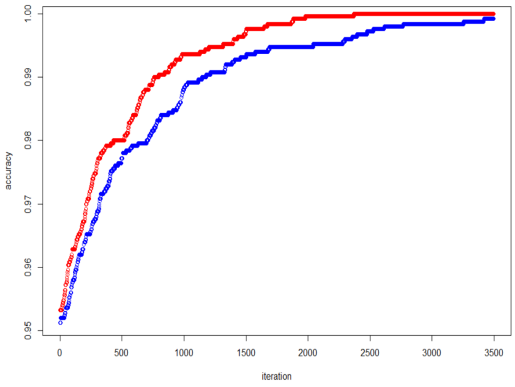

图 5: 训练集准确率随迭代次数变化

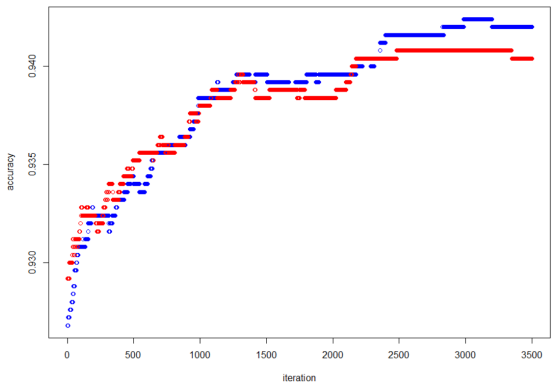

图 6: 测试集准确率随迭代次数变化

上图展示了迭代500-4000次的两组的训练集和测试集的正确率随迭代次数变化图，其中，蓝色点代表对照组，红色点代表实验组，即加入惩罚项的组。从图中可知，加入惩罚项后虽然训练集准确率始终低于不加惩罚项的训练集准确率，但测试集的准确率确实会稍高于不加惩罚项的测试集准确率，说明加入惩罚项确实可以避免过拟合的现象发生，这种效果可能在迭代次数更多时更加明显。

上述结果中还可以看出实验组误差显著高于对照组的误差；但准确率的差异并没有这么明显，原因应该是该误差中很大一部分来自于惩罚项。一个改进想法是寻找更合理的动态寻找惩罚项的系数来替代常数 λ。

3.3.3 使用Armijo准则选择参数的收敛速度

使用一层有200个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,200,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$ 的平均随机数；迭代最大次数设置2000次；终止条件为每个权值偏导数平均小于 10^{-5} 或达到最大迭代次数时停止；由Armijo准则确定梯度下降法步长，参数设置为：

$$\beta = 0.8, \tau = 3, \rho = 0.4$$

实验结果：

```
$test
[1] 0.9424
$train
[1] 0.9976
$‘NumOfIter:‘
[1] 1999
$‘TheCostIS:‘
[1] 0.08325961
```

测试集准确率 94.24% ;训练集准确率99.76%;最终迭代了1999次，最终误差函数值为0.0833。最终误差函数低于对照组，但准确率略高于对照组。

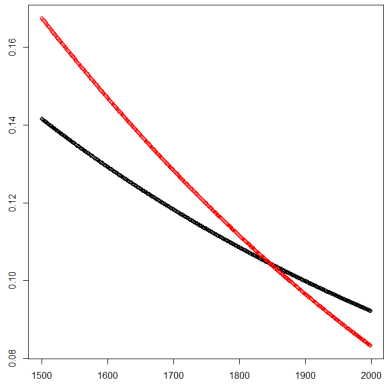

图 7: 误差随迭代次数下降图

上图展示了本组实验与对照组的误差随迭代次数（1500-1999）的下降对比，其中红色曲线是本组实验，黑色曲线是对照组，可以看出在后期对照组误差函数收敛速度衰减较为严重时，使用Armijo准则有加速函数收敛的作用。本组前期误差比对照组高可能是因为1)初始点的不同而导致的;2)前期Armijo准则选择的步长较小。

3.3.4 带有冲量项的梯度下降法

使用一层有200个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,200,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$

的平均随机数；迭代最大次数设置2000次；终止条件为每个权值偏导数平均小于10^{-5} 或达到最大迭代次数时停止；使用梯度下降法时步长固定为1，并且加入冲量项，冲量项系数设置为： $\eta = 0.1$

实验结果：

```
$test
[1] 0.9412
$train
[1] 0.9988
$‘NumOfIter:‘
[1] 2000
$‘TheCostIS:‘
[1] 0.0757796
```

测试集准确率 94, 12% ;训练集准确率99.88%;最终迭代了2000次，最终误差函数值为0.0833。最终误差函数低于对照组，但准确率略高于对照组。

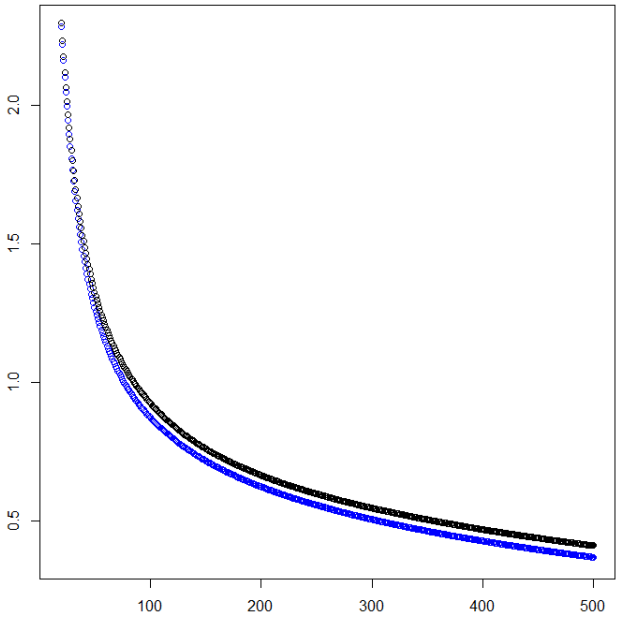

图 8: 误差随迭代次数下降图

上图展示了本组实验与对照组的误差随迭代次数（20-500）的下降对比，其中蓝色曲线是本组实验，黑色曲线是对照组。可以看出，比起对照组的误差函数收敛速度，带有冲量项的梯度下降法在前期有加速函数收敛的作用。

3.3.5 不同个数的隐藏层对人工神经网络的影响

使用两层，分别为有200个节点和有25个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,200,25,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$ 的平均随机数；迭代最大次数设置2000次；终止条件为每个权值偏导数平均小于 10^{-5} 或达到最大迭代次数时停止；使用梯度下降法时步长固定为1。

实验结果：

```
$test
[1] 0.9396
$train
[1] 0.9984
$‘NumOfIter:‘
[1] 1999
$‘TheCostIS:‘
[1] 0.05201518
```

测试集准确率 93.96% ;训练集准确率99.84%;最终迭代了1999 次，最终误差函数值为0.0520。最终误差函数低于对照组，准确率无显著差异。

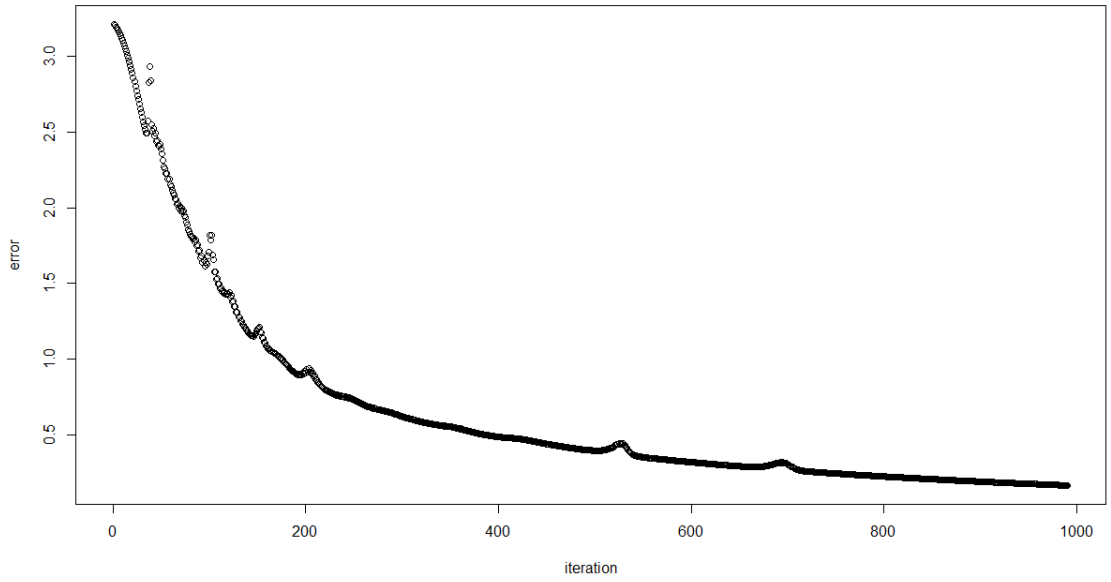

图 9: 误差随迭代次数下降图

上图绘制了误差随迭代次数（10-1000）的增加而下降的图，从图中可以看出，每次迭代并非都会减小误差。经分析，应该是由于多层隐藏层的神经网络导致最后的误差函数非凸，梯度下降法在迭代的过程中越过了很多局部最小点，故该方法应配合冲量项使用。

3.3.6 不同个数的节点对人工神经网络的影响

使用一层有100个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,100,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$ 的平均随机数；迭代最大次数设置2000次；终止条件为每个权值偏导数平均小于 10^{-5} 或达到最大迭代次数时停止；使用梯度下降法时步长固定为1.

实验结果：

```
$test
[1] 0.9356
$train
[1] 0.9952
```

```
$‘NumOfIter:‘
[1] 1999
$‘TheCostIS:‘
[1] 0.1024502
```

测试集准确率 93.56% ;训练集准确率99.52%;最终迭代了1999 次，最终误差函数值为0.1025。最终误差函数高于对照组，准确率无显著性差异。

3.3.7 五折交叉验证

将5000个样本等分为5份，每次用1份作为测试集，其他4份作为训练集，重复5 次，分别求出训练集和测试集的均方差的平均值。使用两层，分别有200个节点的和有25个节点的隐藏层，400维输入，10维输出的神经网络，即基本结构为(400,200,25,10)，并在除输出层外的层中加入偏差项；初始权值为来自 $(-0.5, 0.5)$ 的平均随机数；迭代最大次数设置2000次；终止条件为每个权值偏导数平均小于 10^{-5} 或达到最大迭代次数时停止；使用Armijo准则选取步长；加入惩罚项，设置系数 $\lambda = 0.2$;加入冲量项，设置系数 $\eta = 0.1$.

实验结果：

```
> mean(MNMSE1) #测试集
[1] 0.001316316
> mean(MNMSE2) #训练集
[1] 0.0009209209
```

测试集平均均方差为0.001316316;训练集平均均方差0.0009209209。最后一组训练所得的神经网络的成功率高达99.9%.

值得注意的是，由于有多层网络的非凸性和冲量项的存在，算法并非每一步都是下降的，下图展示了交叉验证时误差函数值随迭代次数的变化。

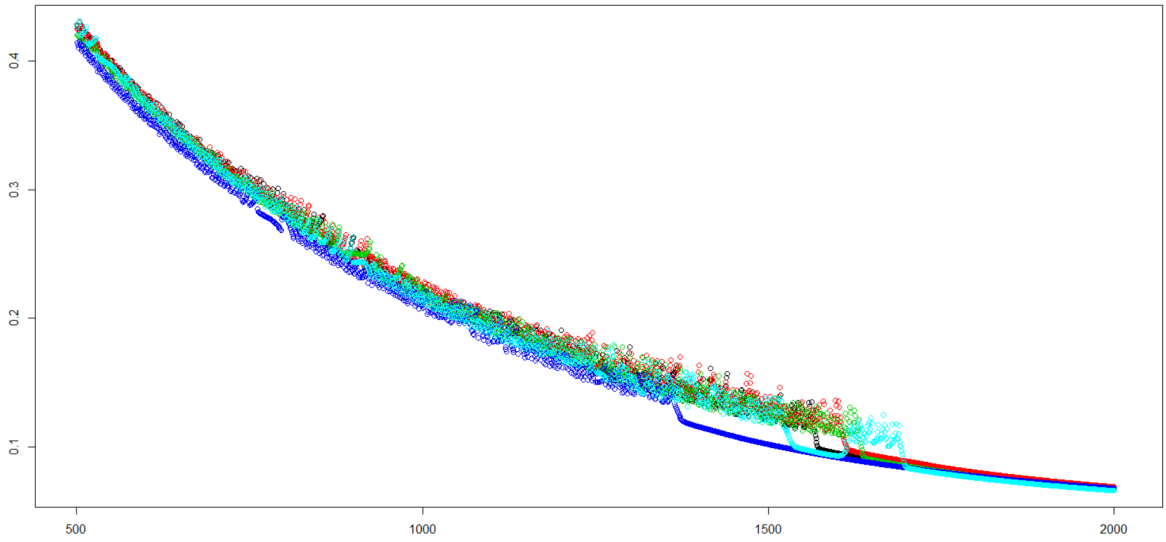

图 10: 误差随迭代次数下降图

上图中每种颜色的点代表一次学习，可以看到只有到后期算法才慢慢收敛至全局最优。

3.4 实验结果分析总结

1. 加入冲量项可使算法在前期较快速的收敛；
2. 使用Armijo准则选取搜索步长可以有效避免算法收敛速度随迭代次数增加而递减；
3. 使用较多节点数的隐藏层可以提高分类器的准确率；
4. 提高收敛速度和优化网络结构都可以在一定迭代次数后提高分类器的准确率；
5. 增加隐藏层会影响误差函数的凸性，虽然能给权重提供更多的逃逸路线，也同时增加了算法收敛在局部最优的风险，故应配合有冲量项的梯度下降法或其他能解决非凸多元函数的最优值得算法来解决这个问题。

抽取实验3.3.4中训练后的神经网络，抽取样本测试，准确率大约在97.5%.

```
[1] "predict value"
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    4    1    6    8    4    5    1    3    8     5
 [2,]    5    6    5    9    5    9    4    7    5     9
 [3,]    5    3    2    2    5    7    8   10    3    10
 [4,]    6    9    7    1    7    2    4    9    3     8
 [5,]    1    7    2    9    9    6    2    8   10     4
 [6,]    9    8    7    7    2   10    8    9    2     4
 [7,]    3    5    5    5    3    5    6    8    3    10
 [8,]    9    3    9    2   10    5    2    9    9     2
 [9,]    5    8    6    5    1    7    7    3    2     5
[10,]    9    6    6    9    8    4    3    9    8     3
[1] "true value"
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    7    9   10    5    2    6    5    2    1     6
 [2,]   10    4    7    8    6    4    9    9   10     5
 [3,]   10    7    5    3    3    2    3    7    7     7
 [4,]    4    7    9    7    3    8    6    4    7     6
 [5,]    7    2    8    3    4    4    4    5    7     9
 [6,]    4   10    9    3    5    9    3    5    8     2
 [7,]    2    8    3    3    9    6    4    2    9     8
 [8,]    4   10    4    3    9    9    5    4   10     3
 [9,]    9    3    4    3    4    2    2    8    7     8
[10,]    7    8    8    8   10    8    1    7    5     9
```

图 11: 左图识别错误样本的预测值和真值

图 12: 右图识别错误样本对应图像

```
[1] "predict value"
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]   10    7    8    9    4    4    6    8    9     6
 [2,]    2    3    9    7    6    4   10    3   10     2
 [3,]    8    8   10    8    5    5    2    9    7     9
 [4,]    7    7    6    3    5   10    4    4    2     8
 [5,]    7    3   10    3    2    2    1    7    9     1
 [6,]    2    3    8    1    5   10    9    4    2     2
 [7,]    6    5    1    3    9    9    4    6   10     3
 [8,]    8    9    7    9    5    7    4    3    5     3
 [9,]    6    1    8    5    8    5   10   10    4    10
[10,]    7    4    8    7    4    3    2    6    9     4
[1] "true value"
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]   10    7    8    9    4    4    6    8    9     6
 [2,]    2    3    9    7    6    4   10    3   10     2
 [3,]    8    8   10    8    5    5    2    9    7     9
 [4,]    7    7    6    3    5   10    4    4    2     8
 [5,]    7    3   10    3    2    2    1    7    9     1
 [6,]    2    3    8    1    5   10    9    4    2     2
 [7,]    6    5    1    3    9    9    4    6   10     3
 [8,]    8    9    7    9    5    7    4    3    5     3
 [9,]    6    1    8    5    8    5   10   10    4    10
[10,]    7    4    8    7    4    3    2    6    9     4
```

图 13: 左图识别正确样本的预测值和真值

图 14: 右图识别正确样本对应图像

各数字的识别率

数字	对应识别率(%)
0	98.2
1	98.8
2	96.8
3	96.4
4	96.2
5	96.2
6	97.6
7	95.8
8	96.6
9	96.4

从上图可以看出识别错误的图像大多是如下几种情况：1)粗细不均匀；2)字体比较扭曲；3)字体位置不在中心.

分类正确率最高的是1，分类正确率最低的是7.

3.5 需要改进之处

1. 编写程序时未考虑计算复杂度，导致程序运行很慢，考虑采用更简单的数据结构和更低级的编程语言从而提高程序运行速度；

2. 各组实验时没有确定统一的初始权值矩阵，而采用随机的方法取初始点，可能导致各组实验没有收敛到同一最优点，最终对比结果的说服力不足；

3. 由于计算时间成本较高，多组实验没有采用交叉验证，可能导致学习的过程中浪费了样本；

4. 寻找更有效的方法确定每次迭代时惩罚项的系数。

4 附录

4.1 参考文献

1. 卜富清.基于人工神经网络的图像识别和分类.[硕士论文]:成都理工大学,2010.05
2. 杨庆之.最优化方法.北京:科学出版社,2015. 71-72
3. https://www.coursera.org/learn/machine-learning/supplement/pjdBA/backpropagation-algorithm

4.2 程序及代码

4.2.1 函数库

```
nmatrix=function(arr,m=20,n=20)    #将向量数据变成20*20的矩阵
{
  x<-matrix(NA,nrow=m,ncol=n);
  for(i in 1:m)
  {
    for(j in 1:n)
    {
      x[j,i]<-as.numeric(arr)[(i-1)*n+j]
    }
  }
  x
}
#----------画图------------
nplot<-function(ma)    #将矩阵画成图片
{
  ro=length(ma[,1])
  co=length(ma[1,])
  plot(cornor(ma),col=gray(1),xlab ="", ylab = "",xaxt="n",yaxt="n")
```

```
  for(i in 1:ro)
  {
    for(j in 1:co)
    {
      points(j,ro+1-i,pch=15,col=gray(1-ma[i,j]),cex=2.6)
    }
  }
}

cornor<-function(x)  #nplot函数中所需函数，作用是返回图片四角定位点的坐
标
{
  ro<-length(x[,1])
  co<-length(x[1,])
  x<-array(NA,dim=c(4,2))
  x[1,]<-c(0,0)
  x[2,]<-c(co+1,0)
  x[3,]<-c(co+1,ro+1)
  x[4,]<-c(0,ro+1)
  x
}

displayData<-function(X,n,arr)   #随机展示n*n张图片
{
  x<-array(NA,dim=c(5000,20,20))
  for(i in 1:5000)
  {
    x[i,,]<-nmatrix(X[i,])
  }
  sa<-sample(arr,size=n*n)
  par(mfrow=c(n,n),mar=c(0.1,0.1,0.1,0.1))
```

```
  for(i in sa)
  {
    nplot(x[i,,])
  }
  par(mfrow=c(1,1),mar=c(5, 4, 4, 2) + 0.1)
  dm<-y[sa]
  dim(dm)<-c(n,n)
  dm<-t(dm)
  dm
}

#输入样本的标签数，在其中随机选n*n张图片展示，并打印真值和预测值
visualizeResult<-function(X,n,arr,th)
{
  x<-array(NA,dim=c(5000,20,20))
  for(i in 1:5000)
  {
    x[i,,]<-nmatrix(X[i,])
  }
  sa<-sample(arr,size=n*n)
  par(mfrow=c(n,n),mar=c(0.1,0.1,0.1,0.1))
  for(i in sa)
  {
    nplot(x[i,,])
  }
  par(mfrow=c(1,1),mar=c(5, 4, 4, 2) + 0.1)
  dm<-pred(X[sa,],th)
  dim(dm)<-c(n,n)
  dm<-t(dm)
  print("predict value")
  print(dm)
```

```
  dm2<-y[sa]
  dim(dm2)<-c(n,n)
  dm2<-t(dm2)
  print("true value")
  print(dm2)
}

#画出各结果中误差随迭代次数的下降函数
costAndIterPlot<-function(li,range)
{
  n<-length(li)
  costP<-cbind(array(1:2010,dim=c(2010,1)),li[[1]][[4]])
  plot(costP[range,],col=1)
  if(n>=2)
  {
    for(i in 2:n)
    {
      costP[,2]<-li[[i]][[4]]
      points(costP[range,],col=i)
    }

  }
}
#-----------------------------------------
sigmoid<-function(arr) #挤压函数
{
  1/(1+exp(-arr))
}

#实验核心函数
#输入样本，真值，权值及网络结构，lambda为惩罚项系数
```

```
#返回一个列表，包含：FP过程算出的误差；BP过程算出的权值更新量；每个结点的值。
costFunction<-function(sample,trueV,Theta,netStructure,lambda=0)
{
  nsample<-length(sample[,1])
  sample<-cbind(t(t(rep(1,nsample))),as.matrix(sample))
  L<-length(netStructure)
  netStructure[-length(netStructure)]<-netStructure[-length(netStructure)]+1
  A<-list()
  A[[1]]<-sample
  for(l in 2:L)
  {
    A[[l]]=sigmoid(as.matrix(A[[l-1]])%*%as.matrix(Theta[[l-1]]))
    A[[l]]=cbind(t(t(rep(1,length(A[[l]][,1])))),A[[l]])
  }
  A[[L]]<-A[[L]][,-1]
  o<-A[[L]]
  s<-0
  for(i in 1:length(Theta))
  {
    s<-sum((Theta[[i]]^2)[-1,])
  }
  cost<--sum(trueV*log(o)+(1-trueV)*log(1-o))/nsample+lambda*s/(2*nsample)
  delta<-list()
  delta[[L]]<-o-trueV
  for(i in (L-1):2)
  {
    if(i==(L-1))
      delta[[i]]<-delta[[i+1]]%*%t(Theta[[i]])*A[[i]]*(1-A[[i]])
    else
      delta[[i]]<-delta[[i+1]][,-1]%*%t(Theta[[i]])*A[[i]]*(1-A[[i]])
```

```
}
DTheta<-list()
for(k in 1:(L-1))
{
  DTheta[[k]]<-matrix(NA,nrow = dim(Theta[[k]])[1],ncol = dim(Theta[[k]])[2])
  if(k==(L-1))
  {
    for(i in 1:dim(A[[k]])[2])
    {
      for(j in 1:(dim(delta[[k+1]])[2]))
      {
        DTheta[[k]][i,j]<-sum(A[[k]][,i]*delta[[k+1]][,j])
      }
    }
  }
  else
  {
    for(i in 1:dim(A[[k]])[2])
    {
      for(j in 1:(dim(delta[[k+1]])[2]-1))
      {
        DTheta[[k]][i,j]<-sum(A[[k]][,i]*delta[[k+1]][,j+1])
      }
    }
  }
  DTheta[[k]]<-DTheta[[k]]/nsample
  DTheta[[k]][-1,]<-DTheta[[k]][-1,]+(lambda/nsample)*Theta[[k]][-1,]
}
a<-list("cost"=cost,"DTheta"=DTheta,"A"=A)
a
```

```
}

#上述函数简化版，只返回误差
cf<-function(sample,trueV,Theta,lambda=0)
{
  nsample<-length(sample[,1])
  sample<-cbind(t(t(rep(1,nsample))),as.matrix(sample))
  L<-length(Theta)+1
  A<-list()
  A[[1]]<-sample
  for(l in 2:L)
  {
    A[[l]]=sigmoid(as.matrix(A[[l-1]])%*%as.matrix(Theta[[l-1]]))
    A[[l]]=cbind(t(t(rep(1,length(A[[l]][,1])))),A[[l]])
  }
  A[[L]]<-A[[L]][,-1]
  o<-A[[L]]
  s<-0
  for(i in 1:length(Theta))
  {
    s<-sum((Theta[[i]]^2)[-1,])
  }
  cost<--sum(trueV*log(o)+(1-trueV)*log(1-o))/nsample+lambda*s/(2*nsample)
  as.numeric(cost)
}

#Armijo准则选择步长，返回最终步长
findAlpha<-function(th,Dth,beta=0.8,tao=3,rho=0.4)
{
  DTheta<-Dth
  sumofD<-0
```

```
  for(j in 1:length(DTheta))
  {
    sumofD<- -sum(DTheta[[j]]^2)
  }
  i<-0
  repeat
  {
    i<-i+1;
    if(cf(s,sY,updateTheta(th,DTheta,alpha = (beta^i)*tao))<=as.numeric(it[[1]])
        +rho*(beta^i)*tao*sumofD)
    {
      break;
    }
  }
  tao*beta^i
}

#输入网络结构，给出（-0.5~0.5）的初始权值，random选项控制是否随机选择
initialTheta<-function(netStructure,random=TRUE)
{
  th<-list()
  if(random)
  {
    for(i in 1:(length(netStructure)-1))
    {
      th[[i]]<-matrix(runif((netStructure[i]+1)*netStructure[i+1]
        ,min = -0.5,max = 0.5),nrow = netStructure[i]+1,ncol = netStructure[i+1])
    }
  }
  else
  {
```

```
    for(i in 1:(length(netStructure)-1))
    {
      th[[i]]<-matrix(0,nrow = netStructure[i]+1,ncol = netStructure[i+1])
    }
  }
  th
}

#将数字转化为向量的形式
#例: 2转化为(0,0,1,0,0,0,0,0,0,0)
transY<-function(arr,numberLabel)
{
  output<-matrix(0,nrow = length(arr),ncol = length(numberLabel)
    ,dimnames = list(dimnames(arr)[[1]],numberLabel))
  for(i in numberLabel)
  {
    output[which(arr==i),which(dimnames(output)[[2]]==i)]<-1
  }
  output
}

#更新权值函数, 默认步长为1
updateTheta<-function(theta,Dtheta,alpha=1)
{
  newtheta<-list()
  n<-length(theta)
  for(i in 1:n)
  {
    newtheta[[i]]<-theta[[i]]-alpha*Dtheta[[i]]
  }
  newtheta
```

```
}

#加入冲量项的更新权值函数，默认步长1，冲量项步长0.5
MupdateTheta<-function(theta,D1,D2,alpha1=1,alpha2=0.5)
{
  newtheta<-list()
  n<-length(theta)
  for(i in 1:n)
  {
    newtheta[[i]]<-theta[[i]]-alpha1*D1[[i]]-alpha2*D2[[i]]
  }
  newtheta
}

#求模函数
module<-function(arr)
{
  as.numeric((arr^2)%*%array(1,dim=length(arr)))
}

#返回 n*m矩阵的二阶范数/(m*n) 用作判断停止条件
stopCon<-function(th)
{
  n<-length(th)
  s<-0
  m<-0
  for(i in 1:n)
  {
    s<-s+sum((th[[i]])^2)
    m<-m+dim(th[[i]])[1]*dim(th[[i]])[2]
  }
```

```
  s/m
}

#预测函数，输入样本和权值，返回输出结果最大的节点对应的数字标签
pred<-function(sample,Theta)
{
  nsample<-length(sample[,1])
  sample<-cbind(t(t(rep(1,nsample))),as.matrix(sample))
  L<-length(Theta)+1
  A<-list()
  A[[1]]<-sample
  for(l in 2:L)
  {
    A[[l]]=sigmoid(as.matrix(A[[l-1]])%*%as.matrix(Theta[[l-1]]))
    A[[l]]=cbind(t(t(rep(1,length(A[[l]][,1])))),A[[l]])
  }
  A[[L]]<-A[[L]][,-1]
  o<-A[[L]]
  a<-matrix(NA,nrow = nsample,ncol = 1)
  dimnames(a)[[1]]<-dimnames(sample)[[1]]
  for(i in 1:nsample)
  {
    a[i,]<-numberLabel[which.max(as.matrix(o[i,]))]
  }
  a
}

#准确性计算函数，输入训练集和测试集的样本和真值及权值，返回训练集和测试集的准确性
accuracy<-function(tr,trY,te,teY,th)
{
```

```
  preY1<-pred(te,th)
  preY2<-pred(tr,th)
  a<-list("test"=length(which((preY1-teY)==0))/length(teY),
    "train"=length(which((preY2-trY)==0))/length(trY))
  a
}

#类似上述函数，求训练集和测试集的均方差
MSE<-function(tr,trY,te,teY,th)
{
  preY1<-pred(te,th)
  preY2<-pred(tr,th)
  a<-c(length(which((preY1-teY)!=0))/length(teY),
    length(which((preY2-trY)!=0))/length(trY))
  a
}
```

4.2.2 实验过程

```
setwd("C:/Users/yaozh/Desktop/dm")#设置路径
X<-read.table("data.txt")#读取数据
source("function.R")#加载函数库
X<-as.matrix(X)#将数据转换为矩阵类型
X<-(X-min(X))/(max(X)-min(X))   #将数据标准化至0—1
y<-read.table("dataY.txt")#读取真值
yName<-dimnames(y)#更改真值矩阵的行列名
y<-as.matrix(y)
dimnames(y)<-yName
dimnames(X)[[1]]<-yName[[1]]

#-----------exp1:基本BP神经网络------------
```

```
numberLabel<-c(10,1,2,3,4,5,6,7,8,9)#确定数字标签
Y<-transY(y,numberLabel)#将真值向量转化为矩阵
netStructure<-c(400,200,10)#确定神经网络结构

sampleIndex<-sample(1:5000,size = 2500)#随机抽取2500个样本的标签
#确定训练集和测试集
s<-as.matrix(X[sampleIndex,])
sy<-as.matrix(y[sampleIndex,])
sY<-transY(sy,numberLabel)
testGroup<-as.matrix(X[-sampleIndex,])
testGroupY<-as.matrix(y[-sampleIndex,])

theta<-initialTheta(netStructure)#初始化权值
k<-0#迭代次数
countIter<-array(NA,dim=c(2010,1))#用来保存每次迭代误差的向量
countAcc1<-array(NA,dim=c(2010,2))#用来保存每次迭代两集准确率的矩阵
repeat
{
  k=k+1
  it<-costFunction(s,sY,theta,netStructure,lambda =0)#返回权值更新矩
阵
  if((stopCon(theta)<10^(-5))||(k>=2000))
    break
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])#打印迭代次数和误差
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  theta<-updateTheta(theta,it[[2]],alpha =1)#更新权值
  countAcc1[k,]<-c(accuracy(s,sy,testGroup,testGroupY,theta)[[1]]
  ,accuracy(s,sy,testGroup,testGroupY,theta)[[2]])
}
#保存结果
```

```
result1<-list()
result1[[1]]<-theta
result1[[2]]<-accuracy(s,sy,testGroup,testGroupY,theta)
result1[[3]]<-pr
result1[[4]]<-countIter
#查看结果
result1[[2]]
result1[[3]]
#-------------exp2:加入惩罚项的神经网络---------------

theta<-initialTheta(netStructure)
k<-0
countIter<-array(NA,dim=c(2010,1))
countAcc2<-array(NA,dim=c(4010,2))
repeat
{
  k=k+1
  it<-costFunction(s,sY,theta,netStructure,lambda =0.5)#惩罚项系数设
置
  if((stopCon(theta)<10^(-5))||(k>=4000))
    break
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  theta<-updateTheta(theta,it[[2]],alpha =1)
  countAcc2[k,]<-c(accuracy(s,sy,testGroup,testGroupY,theta)[[1]]
  ,accuracy(s,sy,testGroup,testGroupY,theta)[[2]])

}
result2<-list()
result2[[1]]<-theta
```

```
result2[[2]]<-accuracy(s,sy,testGroup,testGroupY,theta)
result2[[3]]<-pr
result2[[4]]<-countIter
#查看结果
result2[[2]]
result2[[3]]
#-------------------exp3:使用Armijo准则选择参数的收敛速度----------------
theta<-initialTheta(netStructure)
countIter<-array(NA,dim=c(2010,1))
k<-0
repeat
{
  k=k+1
  it<-costFunction(s,sY,theta,netStructure,lambda =0)
  if((stopCon(theta)<10^(-5))||(k>=2000))
    break
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  theta<-updateTheta(theta,it[[2]],alpha =findAlpha(theta,it[[2]]))#用ARMIJO准
则确定alpha

  if(k%%100==0)
  {
    accuracy(s,sy,testGroup,testGroupY,theta)
  }
}
result3<-list()
result3[[1]]<-theta
result3[[2]]<-accuracy(s,sy,testGroup,testGroupY,theta)
result3[[3]]<-pr
```

```
result3[[4]]<-countIter

#查看结果
result3[[2]]
result3[[3]]
#-------------------exp4:带有冲量项的梯度下降法---------------
Mtheta<-initialTheta(netStructure)
MDtheta<-initialTheta(netStructure,random = FALSE)
k<-0
countIter<-array(NA,dim=c(2010,1))
repeat
{
  k=k+1
  it<-costFunction(s,sY,Mtheta,netStructure,lambda = 0)
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  Mtheta<-MupdateTheta(Mtheta,it[[2]],MDtheta,alpha1=1,alpha2 = 0.1)#设
置冲量项
  MDtheta<-it[[2]]
  if((stopCon(Mtheta)<10^(-5))||(k>=2000))
    break
  if(k%%100==0)
  {
    accuracy(s,sy,testGroup,testGroupY,Mtheta)
  }
}
result4<-list()
result4[[1]]<-Mtheta
result4[[2]]<-accuracy(s,sy,testGroup,testGroupY,Mtheta)
result4[[3]]<-pr
```

```
result4[[4]]<-countIter

#查看结果
result4[[2]]
result4[[3]]
#-----------------------exp5.1: 不同节点数对结果的影响------------------
numberLabel<-c(10,1,2,3,4,5,6,7,8,9)
Y<-transY(y,numberLabel)
netStructure<-c(400,100,10)#隐藏层改为100个节点

theta<-initialTheta(netStructure)
k<-0
countIter<-array(NA,dim=c(2010,1))
repeat
{
  k=k+1
  it<-costFunction(s,sY,theta,netStructure,lambda =0)
  if((stopCon(theta)<10^(-5))||(k>=2000))
    break
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  theta<-updateTheta(theta,it[[2]],alpha =1)

  if(k%%100==0)
  {
    accuracy(s,sy,testGroup,testGroupY,theta)
  }
}
result51<-list()
result51[[1]]<-theta
```

```
result51[[2]]<-accuracy(s,sy,testGroup,testGroupY,theta)
result51[[3]]<-pr
result51[[4]]<-countIter

#查看结果
result51[[2]]
result51[[3]]
#-----------------exp5.2:不同个数的隐藏层对人工神经网络的影响-----------------
numberLabel<-c(10,1,2,3,4,5,6,7,8,9)
Y<-transY(y,numberLabel)
netStructure<-c(400,200,25,10)#设置2层隐藏层


theta<-initialTheta(netStructure)
k<-0
countIter<-array(NA,dim=c(2010,1))
repeat
{
  k=k+1
  it<-costFunction(s,sY,theta,netStructure,lambda =0)
  if((stopCon(theta)<10^(-5))||(k>=2000))
    break
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  theta<-updateTheta(theta,it[[2]],alpha =1)

  if(k%%100==0)
  {
    accuracy(s,sy,testGroup,testGroupY,theta)
  }
```

```
}
result52<-list()
result52[[1]]<-theta
result52[[2]]<-accuracy(s,sy,testGroup,testGroupY,theta)
result52[[3]]<-pr
result52[[4]]<-countIter

#查看结果
result52[[2]]
result52[[3]]

save(result1,result2,result3,result4,result51,result52
    ,file = "result.RData")#保存所有结果

#----------五折交叉验证------------
AllsampleIndex<-sample(1:5000,size = 5000)#将样本标签打乱
sampleM<-matrix(AllsampleIndex,ncol = 5)#将样本标签等分为5组

MNMSE<-matrix(NA,nrow = 5,ncol = 2,dimnames=list(NULL
    , c("test","train")))#记录每次迭代的均方差
RE<-list()#用列表RE来记录每次实验数据
for(q in 1:5)
{
  sampleIndex<-as.vector(sampleM[q,])
  s<-as.matrix(X[-sampleIndex,])
  sy<-as.matrix(y[-sampleIndex,])
  sY<-transY(sy,numberLabel)
  testGroup<-as.matrix(X[sampleIndex,])
  testGroupY<-as.matrix(y[sampleIndex,])
```

```
netStructure<-c(400,200,25,10)
Mtheta<-initialTheta(netStructure)
MDtheta<-initialTheta(netStructure,random = FALSE)

k<-0
countIter<-array(NA,dim=c(2010,1))
repeat
{
  k=k+1
  it<-costFunction(s,sY,Mtheta,netStructure,lambda = 0.2)
  pr<-list("NumOfIter:"=k,"TheCostIS:"=it[[1]])
  print(pr);
  countIter[k,1]<-as.numeric(it[[1]])
  Mtheta<-MupdateTheta(Mtheta,it[[2]],MDtheta
      ,alpha1=findAlpha(Mtheta,it[[2]]),alpha2 = 0.1)
  MDtheta<-it[[2]]
  if((stopCon(Mtheta)<10^(-5))||(k>=2000))
    break

}
result7<-list()
result7[[1]]<-Mtheta
result7[[2]]<-accuracy(s,sy,testGroup,testGroupY,Mtheta)
result7[[3]]<-pr
result7[[4]]<-countIter

RE[[q]]<-result7
MNMSE[q,]<-MSE(s,sy,testGroup,testGroupY,Mtheta)
#查看结果
result7[[2]]
result7[[3]]
```

```
}
#-----------analysis-------------------
#惩罚项的准确率随着迭代次数变化的画图
cp.test<-append(countAcc1[1:1999,1],countAcc11[1:1999,1])
cp.train<-append(countAcc1[1:1999,2],countAcc11[1:1999,2])
exp.test<-append(countAcc2[1:1999,1],countAcc22[1:1999,1])
exp.train<-append(countAcc2[1:1999,2],countAcc22[1:1999,2])
plot(exp.test[-(1:500)],col="blue",xlab = "iteration",ylab = "accuracy")
points(cp.test[-(1:500)],col="red")
plot(exp.train[-(1:500)],col="blue",xlab = "iteration",ylab = "accuracy")
points(cp.train[-(1:500)],col="red")
#画误差随迭代次数下降图(3.3.3)
result<-list(result3,result1)
costAndIterPlot(result,1500:2000)
#画误差随迭代次数下降图(3.3.4)
result<-list(result4,result1)
costAndIterPlot(result,20:2000)
#画误差随迭代次数下降图(3.3.6)
plot(result52[[4]][10:1000],xlab = "iteration",ylab = "error")
#画误差随迭代次数下降图(3.3.7)
costAndIterPlot(RE,500:2000)
#画识别错误的样本的图像
visualizeResult(X,10,which(pred(X,theta)!=y)[sample(1:length(
which(pred(X,theta)!=y)),100)],theta)
#画识别正确的样本的图像
visualizeResult(X,10,which(pred(X,theta)==y)[sample(1:length(
which(pred(X,theta)==y)),100)],theta)
#各数字的识别率
asd<-0
for(asd in 0:9)
{
```

```
    sampleIndex<-(500*asd+1):(500*asd+500)#提取相同数字所有样本的标签
    s<-as.matrix(X[sampleIndex,])#选出相同数字的所有样本
    sy<-as.matrix(y[sampleIndex,])
    sY<-transY(sy,numberLabel)
    testGroup<-as.matrix(X[-sampleIndex,])
    testGroupY<-as.matrix(y[-sampleIndex,])
    print(accuracy(s,sy,testGroup,testGroupY,theta))
}
```

4.3 原始数据样例展示

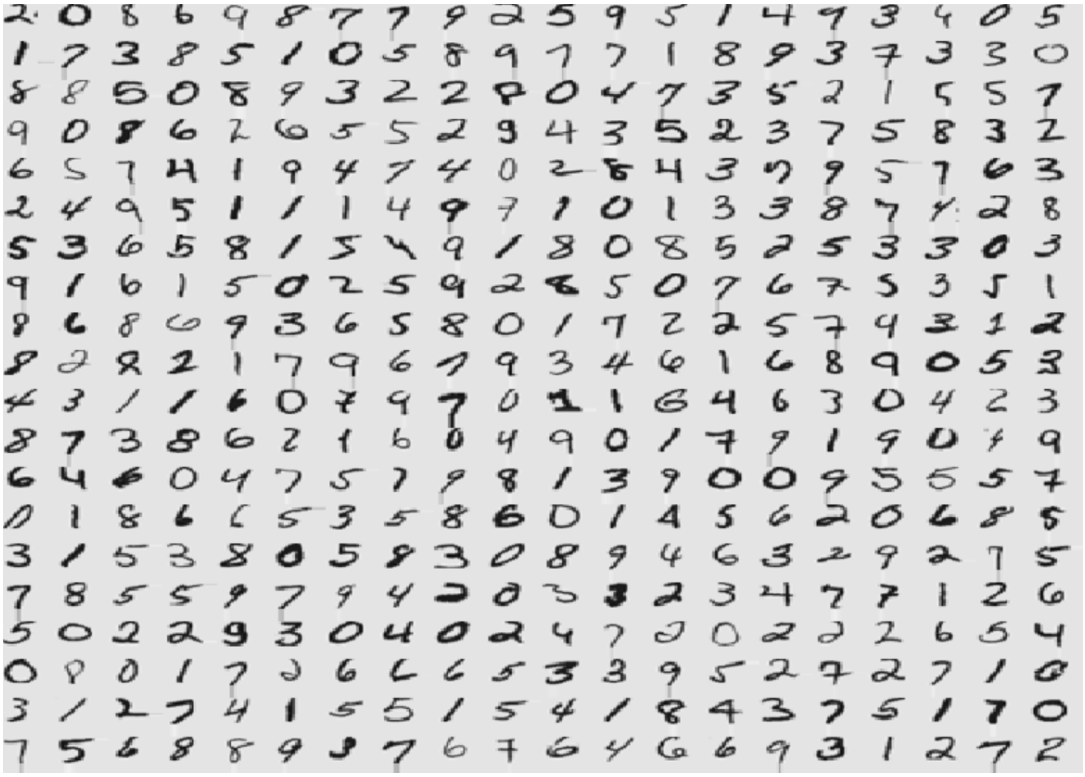

图 15: 随机400个数据展示

#1至10号原始数据展示

```
> X[1:10,]
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	V21	V22	V23	V24	V25	V26	V27	V28	V29	V30	V31	V32
1	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
2	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
3	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
4	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
5	0	0	0	0	0	0	0	0	0	0	0.000000e+00	7.829521e-05
6	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
7	0	0	0	0	0	0	0	0	0	0	3.266722e-18	1.242511e-04
8	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
9	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00
10	0	0	0	0	0	0	0	0	0	0	0.000000e+00	0.000000e+00

	V33	V34	V35	V36	V37	
1	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
2	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
3	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
4	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
5	-0.0007266316	-0.002231413	0.000195738	0.000000000	0.000000000	
6	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
7	-0.0045259060	-0.006088303	-0.005997753	-0.005976988	-0.002124013	
8	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
9	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	
10	0.0000000000	0.000000000	0.000000000	0.000000000	0.000000000	

	V38	V39	V40	V41	V42	V43	V44	V45	V46	V47	
1	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000	
2	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000	
3	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000	
4	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000	
5	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000	

6	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000
7	0.0002527414	0	0	0	0	0	0	0	0.000000e+00	0.0000000000
8	0.0000000000	0	0	0	0	0	0	0	7.078707e-18	0.0009212833
9	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000
10	0.0000000000	0	0	0	0	0	0	0	0.000000e+00	0.0000000000

	V48	V49	V50	V51	V52
1	0.00000000	0.00000000	0.0000000000	0.0000000000	0.000000000
2	0.00000000	0.00000000	0.0003228568	-0.0009155762	-0.025353227
3	0.00000000	0.00000000	0.0000000000	0.0000000000	0.000000000
4	0.00000000	0.00000000	0.0000000000	0.0000000000	0.000000000
5	0.00000000	0.00000000	0.0008731809	-0.0065105377	-0.024494109
6	0.00000000	0.00000000	0.0000000000	0.0000000000	0.000000000
7	0.00000000	0.00000000	0.0000000000	0.0008560597	-0.009889018
8	-0.01401385	-0.04756635	-0.0506098410	-0.0476384980	-0.029302923
9	0.00000000	0.00000000	0.0000000000	0.0000000000	0.000000000
10	0.00000000	0.00000000	0.0000000000	0.0000000000	0.000000000

	V53	V54	V55	V56	V57
1	0.000000000	0.0000000000	0.000000000	0.0000000000	0.000000e+00
2	-0.038593113	-0.0384044680	-0.019374669	0.0003909339	8.193714e-05
3	0.000000000	0.0000000000	0.000000000	0.0000000000	0.000000e+00
4	0.000000000	0.0000000000	0.000000000	0.0000000000	0.000000e+00
5	-0.020270334	-0.0144628230	-0.024110004	-0.0219583380	-6.544372e-03
6	0.000000000	0.0000000000	0.000000000	0.0000000000	0.000000e+00
7	-0.001149515	0.0067240430	0.006173108	0.0055870124	-9.521779e-03
8	-0.006426219	0.0004647181	0.000000000	0.0000000000	0.000000e+00
9	0.000000000	0.0000000000	0.000000000	0.0000000000	0.000000e+00
10	0.000000000	0.0000000000	0.000000000	0.0000000000	0.000000e+00

	V58	V59	V60	V61	V62	V63	V64	V65
1	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000
2	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000
3	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000

4	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000	
5	0.0008005823	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000	
6	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000	
7	-0.0031815825	0.0003383474	0	0	0	0	0.000000e+00	0.0000000000	
8	0.0000000000	0.0000000000	0	0	0	0	4.442402e-06	0.0003395118	
9	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000	
10	0.0000000000	0.0000000000	0	0	0	0	0.000000e+00	0.0000000000	

	V66	V67	V68	V69	V70
1	0.000000e+00	0.0000000000	8.560597e-06	1.940360e-06	-7.374387e-04
2	0.000000e+00	0.0000000000	1.174025e-04	-8.247549e-04	-7.052849e-03
3	0.000000e+00	0.0000000000	0.000000e+00	0.000000e+00	0.000000e+00
4	0.000000e+00	0.0000000000	0.000000e+00	1.685049e-06	1.078942e-04
5	3.888955e-19	0.0001051879	-7.160837e-04	-8.172028e-03	-3.530004e-02
6	1.787173e-05	-0.0001199959	-6.987587e-04	-7.759651e-03	-2.778748e-02
7	0.000000e+00	0.0000000000	0.000000e+00	1.003370e-04	-7.661356e-04
8	-1.910692e-03	-0.0362806540	8.982435e-02	4.129035e-01	4.503259e-01
9	0.000000e+00	0.0000000000	0.000000e+00	0.000000e+00	8.455882e-05
10	0.000000e+00	0.0000000000	0.000000e+00	0.000000e+00	2.573529e-05

	V71	V72	V73	V74	V75
1	-8.134038e-03	-0.018610447	-0.018741287	-0.018757251	-0.0190963540
2	-1.096621e-02	0.196883340	0.320908870	0.318912390	0.1417196500
3	0.000000e+00	0.000000000	0.000000000	0.000000000	0.0000000000
4	-1.077768e-03	-0.004259702	-0.003564469	-0.004818781	-0.0037480086
5	2.545474e-02	0.178062230	0.232222100	0.352079760	0.1612493500
6	-3.749923e-02	-0.037117545	-0.037243233	-0.035247089	-0.0128583540
7	-1.865645e-02	0.046798986	0.492272820	0.600944990	0.5971974100
8	4.158121e-01	0.237812960	0.018746008	-0.026718903	-0.0021041667
9	-7.086397e-04	-0.001421160	-0.001213668	-0.002855699	-0.0005267565
10	2.415237e-05	-0.002732996	-0.008730178	-0.022385672	-0.0214310150

	V76	V77	V78	V79	V80	V81	V82	V83
1	-0.0164039010	-3.781914e-03	3.303473e-04	1.276552e-05	0	0	0	0

2	-0.0156820190	-2.650633e-04	3.301945e-05	0.000000e+00	0	0	0	0
3	0.0000000000	0.000000e+00	0.000000e+00	0.000000e+00	0	0	0	0
4	-0.0034549122	-4.626226e-05	2.568179e-05	0.000000e+00	0	0	0	0
5	0.1643776200	2.582850e-02	-2.013992e-02	6.684028e-04	0	0	0	0
6	0.0004798815	5.024510e-05	0.000000e+00	0.000000e+00	0	0	0	0
7	0.5981599600	3.004975e-01	-1.111492e-02	-2.580508e-03	0	0	0	0
8	0.0003128574	4.442402e-06	0.000000e+00	0.000000e+00	0	0	0	0
9	-0.0019986213	1.130515e-04	0.000000e+00	0.000000e+00	0	0	0	0
10	-0.0072344261	-2.351818e-03	8.453589e-05	1.685049e-05	0	0	0	0

	V84	V85	V86	V87	V88
1	0.000000000	0.000000e+00	0.000000e+00	0.0001164216	0.0001200522
2	0.000000000	0.000000e+00	0.000000e+00	0.0002450980	-0.0019615996
3	0.000000000	0.000000e+00	0.000000e+00	0.0000000000	0.0000000000
4	0.000000000	0.000000e+00	0.000000e+00	0.0000000000	0.0003448698
5	0.000000000	8.823529e-05	-2.113971e-05	-0.0102743400	-0.0219014550
6	0.000000000	3.143382e-04	-1.174156e-03	-0.0224728690	-0.0320393070
7	0.000000000	0.000000e+00	0.000000e+00	0.0000000000	0.0001981167
8	0.000152931	-6.275854e-03	-1.205300e-02	0.4251394500	0.8380376200
9	0.000000000	0.000000e+00	0.000000e+00	0.0000000000	0.0000000000
10	0.000000000	0.000000e+00	0.000000e+00	0.0000000000	0.0000000000

	V89	V90	V91	V92	V93
1	-1.404446e-02	-0.0284542480	0.0803826590	0.266540340	0.27385375
2	-1.207159e-02	0.0056237234	0.2903474800	0.864413040	0.98839085
3	0.000000e+00	0.0000000000	0.0004825368	-0.005166973	-0.01552128
4	-2.536101e-03	-0.0148564300	-0.0130160510	0.014937057	0.00706470
5	7.871982e-02	0.4125784100	0.6236381200	0.893358130	0.90768626
6	7.472273e-02	0.4310125400	0.6077349400	0.600787240	0.60322160
7	-1.857435e-03	-0.0155327650	0.1175179700	0.567606520	1.00022560
8	9.700844e-01	0.8309023700	0.9292359600	0.893987290	0.77071309
9	7.536765e-05	-0.0006985294	-0.0091118260	-0.015672420	-0.01715934
10	1.112132e-04	-0.0005199142	-0.0208095210	-0.003532833	0.09006172

	V94	V95	V96	V97	V98
1	2.787295e-01	0.274293610	0.224676400	2.775630e-02	-7.063155e-03
2	9.855265e-01	0.799468440	0.156723180	-1.741983e-02	5.393176e-04
3	-1.616483e-02	-0.016217831	-0.016096201	-5.737132e-03	6.824019e-04
4	2.238138e-02	0.008874336	0.007045020	-1.360774e-02	1.886389e-04
5	9.577360e-01	0.880227700	0.852510980	6.210229e-01	1.618952e-01
6	5.662254e-01	0.168896510	-0.030083367	3.533497e-05	5.136358e-05
7	1.036948e+00	1.032617700	1.040287800	7.654275e-01	1.581981e-01
8	3.348544e-01	-0.007164777	-0.005947117	1.529310e-04	0.000000e+00
9	7.291667e-05	-0.024678581	-0.006660386	-8.246630e-03	8.878562e-04
10	3.330842e-01	0.316191620	0.063266919	-3.537905e-03	-4.275763e-03

	V99	V100	V101	V102	V103	V104	V105
1	2.347154e-04	0	0	0	0.000000e+00	0.000000e+00	0.000000e+00
2	0.000000e+00	0	0	0	0.000000e+00	0.000000e+00	0.000000e+00
3	0.000000e+00	0	0	0	0.000000e+00	0.000000e+00	0.000000e+00
4	9.497549e-05	0	0	0	0.000000e+00	0.000000e+00	0.000000e+00
5	-1.061394e-02	0	0	0	0.000000e+00	2.177815e-19	4.460784e-04
6	0.000000e+00	0	0	0	0.000000e+00	2.450980e-04	-4.780841e-03
7	-1.185185e-02	0	0	0	0.000000e+00	0.000000e+00	5.880100e-18
8	0.000000e+00	0	0	0	3.601447e-18	-9.136029e-04	-3.245634e-02
9	0.000000e+00	0	0	0	0.000000e+00	9.644609e-19	3.140319e-04
10	3.405842e-04	0	0	0	0.000000e+00	0.000000e+00	0.000000e+00

	V106	V107	V108	V109	V110
1	1.283355e-17	-3.262868e-04	-1.386516e-02	8.156516e-02	0.382800380
2	2.385226e-17	-1.088474e-03	-2.907228e-02	2.357536e-01	0.792536320
3	1.153158e-31	3.785250e-18	-1.396078e-16	-3.266619e-16	0.001393995
4	0.000000e+00	1.148897e-04	-1.962262e-03	7.148523e-04	0.056742426
5	-7.947032e-03	2.283946e-02	3.192713e-01	8.855720e-01	0.993029120
6	-1.890588e-02	1.728360e-01	4.066546e-01	8.246658e-01	1.007341100
7	-1.706992e-17	1.076559e-03	-2.736898e-02	-5.683672e-02	0.262759400
8	4.951186e-01	1.033364e+00	5.176248e-01	1.961433e-02	0.017962946

```
9  -7.191245e-05 -3.461414e-02 -3.051369e-02 -2.280159e-02 -0.041178649
10  0.000000e+00  0.000000e+00  5.707064e-04 -4.414607e-03 -0.036472529
V111        V112       V113       V114       V115        V116
1   0.85784978 1.001097600 0.96971064 0.93092860 1.00383760  0.96415736
2   1.00266260 1.058621300 0.87624713 0.92912069 1.06343230  0.73178139
3  -0.01422505 0.002406046 0.08631983 0.09141064 0.09163628  0.08998744
4   0.29801757 0.745183310 0.66320550 0.80106558 0.68706602  0.63895435
5   1.03533950 0.973184350 0.92769227 0.92559329 0.93809315  1.00063680
6   1.03292250 1.044407200 1.03464410 1.02917790 0.80114090  0.15150437
7   0.87607972 1.004013700 1.00489620 0.75625921 0.43580459  0.81773652
8   0.02572486 0.127707480 0.66018120 0.97793342 0.46654776 -0.02912209
9   0.18568164 0.396498830 0.36432460 0.57027277 0.28065957  0.44554604
10  0.12593663 0.594678940 0.88306333 1.03458120 1.02200210  0.85927914
V117          V118          V119 V120 V121 V122         V123
1   0.4492565500 -5.604083e-03 -3.783190e-03    0    0    0 0.0000000000
2   0.0359943150 -1.313364e-02  4.042416e-04    0    0    0 0.0000000000
3   0.0172561610 -7.639364e-03  4.382830e-04    0    0    0 0.0000000000
4   0.0841348210 -1.260369e-02  1.208469e-05    0    0    0 0.0000000000
5   1.0295737000  8.055771e-01 -2.096014e-02    0    0    0 0.0000000000
6  -0.0269049560  5.985284e-04  7.659314e-05    0    0    0 0.0002282826
7   1.0181841000  8.218292e-01 -2.042705e-02    0    0    0 0.0000000000
8  -0.0009136029  1.068016e-17  0.000000e+00    0    0    0 0.0008071420
9   0.0316111110 -6.889854e-03  8.510349e-05    0    0    0 0.0000000000
10  0.5017707700  1.482512e-03 -5.655467e-03    0    0    0 0.0000000000
V124          V125          V126          V127         V128
1   5.106209e-06  0.0004364107 -3.955099e-03 -0.0268537240  0.100755010
2   0.000000e+00  0.0002450980 -1.088474e-03 -0.0302730120  0.222924170
3   0.000000e+00  0.0000000000  1.311874e-17  0.0004272195 -0.017681369
4   0.000000e+00  0.0000000000  6.170003e-04 -0.0079418573 -0.030073224
5   2.154820e-04 -0.0176221750  7.585544e-02  0.6052757500  0.928175790
6  -4.105222e-03  0.0035990945  2.802954e-01  0.8701657300  1.042858600
```

```
7   2.961601e-05  0.0003424564 -8.411629e-03 -0.0327294390  0.268863920
8  -2.540998e-02  0.2551782700  9.241262e-01  0.4501178000 -0.005678937
9   2.976920e-04 -0.0075936138 -1.700785e-02  0.3738638700  0.327399600
10  0.000000e+00  0.0000000000  4.563041e-18  0.0003914760 -0.014242863
V129         V130         V131       V132        V133      V134
1   0.64203171  1.031368400  0.850968610  0.5431224  0.34259974 0.2689188
2   0.85691410  1.007231100  0.452620660  0.4226028  0.18243150 0.2298942
3  -0.02093375 -0.036391953  0.086954912  0.5371760  0.75332669 0.7681749
4   0.25466102  0.730819410  0.900046300  1.0212837  1.01367550 1.0168308
5   1.01963800  1.011913100  0.710493970  0.3676907  0.22504863 0.2368516
6   1.01214840  0.983427410  0.855948720  0.7168462  0.80704366 0.9677421
7   0.58714956  0.864134680  1.026320200  1.0209873  0.87443126 0.1965629
8  -0.01177082 -0.004540492 -0.005195261 -0.0170682 -0.01959021 0.5143337
9   0.23098701  0.305055220  0.763240980  0.9814693  0.97892836 1.0097555
10  0.01776283  0.386222990  0.848552460  0.9987317  0.83833437 0.5080836
V135      V136        V137         V138          V139 V140 V141
1   0.66837464 1.0125696  0.90379560  0.104481570 -0.0166424970    0    0
2   0.76364990 0.9839787  0.45445391 -0.001252955 -0.0041811343    0    0
3   0.77148795 0.7766082  0.43210914 -0.007761896 -0.0045692062    0    0
4   1.01376810 0.9954434  0.68290943  0.051745139 -0.0104670480    0    0
5   0.26316525 0.8963408  1.02256220  0.855357300 -0.0207092520    0    0
6   1.03209090 0.7847754  0.16399985 -0.013788981 -0.0006382761    0    0
7  -0.05210413 0.4039984  1.00869310  0.855210300 -0.0206510420    0    0
8   0.97183233 0.2504876 -0.02543908  0.000807142  0.0000000000    0    0
9   0.96789255 0.9173716  0.25503911 -0.034844415 -0.0005446623    0    0
10  0.53587427 0.9609202  0.94906311  0.227030120 -0.0180444920    0    0
V142         V143          V144          V145          V146
1  0.0000000000  0.000000000  2.598753e-05 -0.0031060699  0.0075245608
2  0.0000000000  0.000000000  1.956708e-05 -0.0011878847 -0.0145840810
3  0.0000000000  0.000000000  3.913416e-05 -0.0002111614 -0.0049720761
4  0.0000000000  0.000000000  0.000000e+00  0.0000000000 -0.0031478028
```

```
5  0.0000000000  0.000000000 -1.069259e-03 -0.0202441980  0.3483004000
6  0.0000000000 -0.001093478 -1.807947e-02  0.4516305000  0.9539127000
7  0.0000000000  0.000000000 -9.477804e-06 -0.0112161140  0.0728136340
8  0.0001182178 -0.010144778  4.059071e-02  0.7841849100  0.5908279800
9  0.0000000000  0.001981929 -2.062110e-02 -0.0182807640  0.4030450800
10 0.0000000000  0.000000000  0.000000e+00  0.0001540907  0.0002286902
V147         V148         V149      V150        V151          V152
1   0.17753983  0.792890120 0.9656265000 0.4631661  0.06917207 -0.0036410053
2   0.22777273  0.849794330 1.0378720000 0.5896197 -0.04630461 -0.0530524640
3  -0.02965438  0.272686530 0.5118432500 0.6257573  0.87373388  1.0144224000
4   0.02756860  0.081832339 0.6617469900 1.0481730  1.00191170  0.9386243100
5   1.01429510  1.020437700 0.9247806000 0.5553584  0.06503263 -0.0177842320
6   1.01197020  0.993849560 0.9859599600 0.7306070  0.10319363  0.0397518110
7   0.44360953  0.912250260 1.0457054000 1.0166133  0.90066334  0.9266078100
8  -0.03664776 -0.008254785 0.0006930007 0.0000000  0.00000000  0.0006930007
9   0.96354235  0.806824440 0.8235737100 0.9628617  1.00120320  0.9956324500
10 -0.02266051  0.001910028 0.3217363700 0.9435976  0.98999852  0.5550485900
V153         V154         V155      V156      V157         V158
1  -0.04121804 -0.050190066  0.156102910 0.9017626 1.0474835  0.151055250
2  -0.02389868 -0.065509203  0.497592820 1.0192615 0.9173342  0.106986870
3   1.02447320  1.049536900  1.056835300 1.0223237 0.9825927  0.480230580
4   0.93311478  0.817322040  0.991339040 0.9934598 1.0368490  0.148108440
5  -0.03220920 -0.031976173 -0.045740424 0.4561291 1.0464948  0.849830830
6   0.06253314  0.444309990  0.957748500 1.0445964 0.5595171 -0.007746743
7   0.93376390  0.260573320 -0.059897239 0.2095408 0.9842879  0.851434300
8  -0.01037340 -0.009057774  0.460882710 0.8197749 0.1169071 -0.018098124
9   0.82964867  0.634221630  0.958821100 1.0064539 0.7599297  0.166362610
10  0.08266130 -0.031354935  0.005016561 0.8118567 1.0300483  0.656595620
V159 V160 V161          V162          V163          V164
1  -0.0216044660    0    0  0.0000000000  5.870124e-05 -6.409314e-04
2  -0.0169564360    0    0  0.0000000000  0.000000e+00 -9.165645e-05
```

```
3  -0.0193083920    0    0  0.0000000000  6.481595e-05 -5.790952e-04
4  -0.0213461200    0    0  0.0000000000  0.000000e+00  3.314951e-04
5  -0.0206699350    0    0  0.0000000000  4.659411e-04 -1.989880e-02
6  -0.0049884160    0    0  0.0000000000 -1.425462e-02  9.204381e-02
7  -0.0205381290    0    0  0.0000000000  2.568179e-05 -2.589512e-03
8   0.0001182178    0    0 -0.0004644948 -4.000250e-02  3.972473e-01
9  -0.0125674320    0    0  0.0000000000 -1.653010e-02  1.312009e-01
10 -0.0207820380    0    0  0.0000000000  7.582243e-05  1.468035e-04
V165         V166        V167         V168      V169        V170
1  -0.032330525  0.278203470  0.93672016 1.0432096000 0.5980032 -0.00359409
2  -0.016770612  0.115367580  0.80661652 1.0273074000 0.7509306  0.10106269
3  -0.002743532  0.008840669  0.14660364 0.9160305500 1.0202932  1.01054900
4  -0.004845180 -0.055757574  0.42427621 0.8242826800 0.9950807  1.00085190
5   0.208611380  0.906177000  1.00904580 1.0119818000 0.4187363 -0.06084772
6   0.792186340  1.026351900  0.99187500 1.0340900000 0.7216969  0.07623303
7  -0.011476035  0.600159130  1.00883010 1.0039574000 0.9877753  0.72414825
8   0.909575950  0.108580920 -0.01415508 0.0004647181 0.0000000  0.00000000
9   0.489384210  0.925439460  0.99787585 0.9895907400 0.9964416  1.04178070
10 -0.014482588 -0.044024152  0.12518871 0.6108555700 1.0081544  0.89340057
V171          V172          V173          V174        V175
1  -0.02167518 -0.0048102192  6.165668e-05 -0.0123773320  0.15547748
2  -0.02012149  0.0001347018  0.000000e+00 -0.0190195060  0.27271647
3   1.00844260  0.9684874600  9.496713e-01  0.7262071600  0.69479555
4   0.45258042  0.2512175800  3.558486e-01  0.1840353300  0.85024738
5  -0.01013906 -0.0023544730  3.057356e-04 -0.0003835784 -0.03247416
6  -0.02202829 -0.0106267020 -1.759373e-02 -0.0134731580  0.54667659
7   0.14032794  0.1841497500  7.436757e-01  0.3401562200 -0.06294242
8   0.00000000  0.0000000000  6.930007e-04 -0.0112798030  0.01600335
9   0.73681281  0.4803092200  1.986144e-01  0.0045625340  0.46651792
10  0.24396318 -0.0123940460 -1.670047e-02 -0.0044905195 -0.02147266
V176      V177        V178          V179 V180 V181         V182
```

1	0.9148675	0.9204014	0.10917390	-0.0171058010	0	0	0.000156250
2	0.9735190	0.9892304	0.13474043	-0.0198124320	0	0	0.000000000
3	0.9771265	1.0140293	0.84782184	-0.0207584420	0	0	0.000000000
4	1.0186443	0.8573460	0.09278928	-0.0153004150	0	0	0.000223652
5	0.3187803	1.0539061	0.84980983	-0.0206694240	0	0	0.000000000
6	1.0356993	0.8376889	0.11803159	-0.0143758510	0	0	0.000000000
7	0.2273342	0.9846912	0.86631254	-0.0205021110	0	0	0.000000000
8	0.8932691	0.4045179	-0.03800742	-0.0006794662	0	0	-0.011071283
9	0.9869429	1.0170043	0.72454595	-0.0208101850	0	0	0.000000000
10	0.4443075	1.0476591	0.74622179	-0.0206918910	0	0	0.000000000

	V183	V184	V185	V186	V187
1	-0.0004277241	-0.0251466500	0.13053256	0.78166486	1.028365800
2	0.0000000000	-0.0008661663	-0.02723977	0.40702601	1.014674900
3	0.0010717052	-0.0148640390	0.02684755	0.65235909	0.859682730
4	-0.0015399902	-0.0148742850	0.04335454	0.16568961	0.689346000
5	-0.0058517794	0.0197588850	0.60924663	1.01615190	1.011176100
6	-0.0389299250	0.3237009800	0.97810800	0.99989638	1.006288800
7	0.0009746851	-0.0262867140	0.23213598	0.94946262	0.992322180
8	0.0520591290	0.7851679400	0.46706883	-0.02802272	-0.001023284
9	-0.0489678370	0.4188714800	1.03674800	0.99097215	0.990013500
10	-0.0002674167	-0.0100544490	0.17763972	0.68400824	0.838561920

	V188	V189	V190	V191	V192
1	0.7571376	0.28466719	0.0048686513	-3.186887e-03	0.000000e+00
2	0.8121848	0.09606105	0.0004992681	8.884804e-06	-4.687500e-05
3	1.0083264	1.02331170	0.9898203100	6.066541e-01	2.070830e-01
4	1.0515391	0.82441873	0.6130288800	1.244199e-02	-2.529512e-02
5	0.6194679	0.06172598	-0.0072282646	0.000000e+00	0.000000e+00
6	0.9628099	0.34828932	-0.0350931710	-5.584831e-04	3.574346e-05
7	0.9926775	0.99168566	0.5132921600	-1.153329e-02	-2.772091e-02
8	0.0000000	0.00000000	0.0000000000	0.000000e+00	0.000000e+00
9	0.9947825	0.95498945	0.4308214400	1.016996e-01	-7.049581e-03

```
10 1.0114888 0.96495290  0.3309545200 -3.674757e-02 -2.317640e-03
V193          V194         V195      V196      V197         V198
1   8.364926e-04 -0.0370751120  0.452644170 1.0318013 0.5390281 -0.002437426
2   8.805185e-05 -0.0297574550  0.451886270 1.0201287 0.7173337  0.045695498
3   2.145318e-01  0.0301579520  0.383145560 0.9847969 1.0321021  0.470512920
4  -4.333321e-02  0.1846876700  0.877360350 1.0398468 0.5792999  0.003019751
5   0.000000e+00 -0.0003978758 -0.032310083 0.3204300 1.0541682  0.859737540
6   3.668827e-04 -0.0132526550  0.111752110 0.7836358 1.0192914  0.386637880
7   1.591019e-01  0.1585568300 -0.058101818 0.2420504 1.0017735  0.592363100
8   0.000000e+00 -0.0024906556 -0.007218495 0.5873890 0.6802386  0.020794556
9  -9.397378e-03 -0.0116995170  0.027888634 0.9516509 1.0122086  0.868776170
10  2.343973e-04 -0.0007690462 -0.031604507 0.4309991 1.0591597  0.437163550
V199 V200 V201          V202          V203        V204       V205
1  -0.004802900    0    0 -7.036356e-04 -0.0127262440  0.16170665 0.77986538
2  -0.010245609    0    0  3.318575e-18  0.0009783539 -0.01795119 0.08237805
3  -0.021275531    0    0  0.000000e+00 -0.0210146350  0.16372675 0.69359812
4  -0.005607639    0    0 -8.990332e-04  0.0061700501  0.01461271 0.51813523
5  -0.020676232    0    0  0.000000e+00 -0.0285650810  0.22563142 0.91388429
6  -0.020371392    0    0  0.000000e+00 -0.0404895850  0.33914738 0.97511157
7  -0.020960308    0    0  0.000000e+00 -0.0116285520  0.07440547 0.64566910
8  -0.007679568    0    0 -2.147552e-02  0.1512907600  1.06380980 0.29216963
9  -0.020536152    0    0  0.000000e+00 -0.0496987490  0.42589301 1.03998060
10 -0.021070772    0    0  0.000000e+00 -0.0119921730  0.07251884 0.74208740
V206        V207         V208          V209          V210
1   1.03676710  0.80449040 0.1605867200 -0.0138173340  2.148795e-03
2   0.88509554  1.01910400 0.8775185300  0.5056462800  2.229610e-01
3   0.99701457  1.01653850 0.8780840200  0.6928199400  3.373264e-01
4   0.90635027  0.96851871 1.0203144000  0.4678821700 -3.682477e-02
5   1.01147400  1.01549550 0.4488454100 -0.0209244790  1.744622e-05
6   1.00363830  0.96327744 0.4058440200 -0.0184270150 -3.588984e-04
7   1.00296270  0.99091355 0.9952322600  0.9941142900  9.746939e-01
```

```
8  -0.07341976 -0.01561668 0.0002917227  0.0001234001  0.000000e+00
9   0.98035263  0.99210281 0.9779293900  0.4281902100 -2.724210e-02
10  1.04807990  1.01559980 1.0174120000  0.7998525300  1.088529e-01
V211          V212          V213          V214        V215
1  -2.126226e-04  2.042484e-04 -6.859076e-03  4.317130e-04  0.72068095
2  -8.772008e-03 -7.878370e-04  9.954751e-04 -4.422524e-02  0.55714369
3   3.252092e-02 -5.049700e-02 -5.055223e-02  3.228509e-01  0.91059144
4  -2.228282e-03 -1.243839e-03 -2.945173e-02  4.489136e-01  1.04976150
5   0.000000e+00  3.629692e-19  1.103719e-17 -1.716810e-03 -0.02104681
6   3.186274e-05  0.000000e+00  0.000000e+00 -3.875613e-04 -0.03476976
7   2.319037e-01 -1.662503e-02 -1.346660e-02 -2.103232e-02  0.17844417
8   0.000000e+00  0.000000e+00  0.000000e+00 -6.229575e-06 -0.03742317
9  -2.438542e-02 -3.713848e-03 -9.905833e-04 -1.028166e-02  0.11617094
10 -2.171043e-02  1.542586e-04  4.789858e-04 -1.432775e-02  0.11069174
V216      V217        V218          V219 V220 V221          V222
1  0.8481361 0.1513834 -0.02284044  0.0001989719    0    0 -0.0094041054
2  1.0212004 0.4573860 -0.01323575 -0.0032293369    0    0  0.0004306236
3  1.0085801 0.8335150  0.21589451 -0.0147817950    0    0  0.0000000000
4  0.9137138 0.2921208 -0.01993024 -0.0012661697    0    0 -0.0149565970
5  0.5549585 1.0438622  0.79544420 -0.0207255920    0    0  0.0000000000
6  0.3740498 1.0375331  0.72878309 -0.0207858460    0    0  0.0000000000
7  0.8024535 0.9341853  0.28163791 -0.0177888410    0    0  0.0000000000
8  0.2970085 0.8688898  0.07973819 -0.0139682050    0    0 -0.0157807390
9  0.9561913 1.0065673  0.75653508 -0.0204374320    0    0  0.0000000000
10 0.8003730 0.9476153  0.28284032 -0.0178123300    0    0  0.0000000000
V223         V224      V225      V226      V227         V228
1   0.037452050  0.694389110 1.0284484 1.0164807 0.8804884  0.392123950
2  -0.014126513  0.054790356 0.5133316 0.9861777 1.0037886  1.027840700
3  -0.051465900  0.441784960 1.0653416 0.9887873 1.0144790  0.269598300
4   0.103943940  0.713005960 0.8876797 1.0049279 1.0104083  0.771632170
5  -0.051048062  0.438275120 1.0544158 0.9908971 1.0185599  0.792388220
```

```
6  -0.003671273 -0.003891034 0.6508962 1.0324622 0.8546740  0.086345797
7  -0.044948433  0.380534990 1.0178887 0.9884544 0.9839173  0.989300680
8   0.098191931  0.874261970 0.8122937 0.4672409 0.1667071 -0.002860921
9  -0.050125963  0.430112850 1.0394404 0.9790247 1.0017402  0.742413530
10 -0.031745546  0.255857040 0.9312997 1.0062760 1.0046186  0.874157950
V229          V230          V231          V232          V233
1  -0.017412241 -0.0001200980  5.552151e-05 -0.0022390727 -0.0276068380
2   0.881669240  0.1756475400 -2.024762e-02  0.0001370166 -0.0260476200
3  -0.073488868 -0.0404160200 -3.582118e-02  0.0779742820  0.4121803500
4   0.140700610 -0.0338244830 -1.437422e-02 -0.0358306440  0.2903837000
5   0.049642480 -0.0132365710 -6.038603e-03  0.0005513004 -0.0008010273
6  -0.019732298  0.0003734681  3.676471e-06  0.0000000000  0.0000000000
7   1.001262800  0.7938325000  1.180732e-01 -0.0095859545 -0.0086206908
8  -0.001406761  0.0000000000  0.000000e+00  0.0000000000  0.0000000000
9   0.071224503 -0.0339428790 -2.008265e-02 -0.0204238150 -0.0205413560
10  0.203332470 -0.0398965990 -3.940785e-03 -0.0030205440 -0.0118861510
V234        V235      V236        V237         V238
1   3.686455e-01  0.93641117 0.4590067 -0.04247018  0.001173566
2   1.416109e-01  0.80720129 0.7406545  0.09939456 -0.017355999
3   9.305753e-01  1.04236680 0.9338868  0.22053217 -0.047775875
4   8.463051e-01  1.04471680 0.7084632  0.03170202 -0.008817975
5  -2.998827e-02  0.18793136 0.9308789  1.03769750  0.421274240
6   1.154003e-04 -0.03667894 0.2474320  1.03302760  0.868603520
7  -3.254660e-02  0.66894305 1.0216516  0.43512871  0.018607221
8   2.798203e-05 -0.03784511 0.2934062  0.86764951  0.079267458
9  -5.288307e-02  0.49493195 0.9995748  1.00532760  0.359177110
10 -3.271974e-03  0.57628302 1.0652954  0.53838329 -0.019960356
V239 V240 V241         V242          V243        V244      V245
1   1.889297e-05    0    0 -0.019351195  0.1299997900  0.97982171 0.9418624
2   2.287582e-04    0    0 -0.006685588  0.0227848150  0.51873667 0.9435012
3   8.971610e-04    0    0  0.000000000 -0.0507446350  0.43579422 1.0461276
```

```
4   9.752859e-05    0    0 -0.017991351  0.1153687200  0.95551734 1.0273760
5  -2.109750e-02    0    0  0.000000000 -0.0496906160  0.42548794 1.0464824
6  -2.070840e-02    0    0  0.000000000  0.0006150816 -0.04350959 0.6075361
7  -3.212657e-03    0    0  0.000000000 -0.0478461850  0.40734542 1.0467647
8  -1.391748e-02    0    0 -0.001187783 -0.0084171490  0.17346886 0.8614416
9  -2.052764e-02    0    0  0.000000000 -0.0386675570  0.32108084 0.9785390
10 -4.246153e-03    0    0  0.000000000 -0.0504879440  0.43301175 1.0503592
V246      V247       V248         V249         V250         V251
1  0.7751477 0.8736322 0.21277835 -0.017235335  0.000000000  0.001099374
2  0.9505838 0.9068309 0.82616136  0.259302620 -0.008147769 -0.002080123
3  1.0093249 0.6999901 0.05148959  0.018867606  0.364683530  0.746856260
4  0.9839509 1.0079568 0.46436896  0.116999460  0.140808200  0.131075950
5  0.9914125 0.8590546 0.41978695 -0.016210458  0.015668389  0.011214891
6  1.0194068 0.9962129 0.40872157 -0.008235413 -0.001038227  0.000000000
7  0.9884254 0.9836680 1.01226970  0.810863140  0.150141420 -0.034692991
8  0.9618792 0.3540858 0.00419248 -0.003778892  0.000000000  0.000000000
9  1.0027549 1.0094540 0.71670750  0.729940860  0.449204150  0.597090160
10 0.9914179 0.9997250 0.93565911  0.523072400  0.161518330 -0.020604694
V252         V253          V254        V255       V256
1  -0.02617938  0.122872880  8.308127e-01  0.72650177 0.05244419
2  -0.01695749 -0.019870399  5.628242e-01  0.96671925 0.26919408
3   0.81495827  0.990069590  1.031752e+00  0.69257852 0.19128463
4   0.23435692  0.614451720  1.039108e+00  0.84951177 0.20497955
5  -0.03503589 -0.037929876  2.152071e-01  0.81657403 0.99942924
6   0.00000000  0.000000000  5.185276e-04 -0.04470568 0.25937007
7  -0.02397101  0.066720907  5.743376e-01  0.99744605 0.61507976
8   0.00000000  0.000000000 -3.225511e-05 -0.03710219 0.29918752
9   0.59245215  0.439493500  8.488298e-01  0.94065533 1.02596520
10 -0.02455555  0.005119883  5.027216e-01  0.99950038 0.70461606
V257         V258          V259 V260 V261          V262
1  -0.006189719  0.000000000  0.0000000000    0    0 -0.0093656386
```

2	-0.029205240	0.001072731	0.0000000000	0	0	-0.0218887870
3	0.013963132	-0.002826039	0.0000000000	0	0	0.0000000000
4	-0.024783437	0.001005609	0.0000000000	0	0	-0.0039183347
5	0.642266720	0.057606401	-0.0092571644	0	0	0.0000000000
6	1.032832400	0.839363510	-0.0207188520	0	0	0.0000000000
7	0.025528090	-0.015929473	0.0004361828	0	0	0.0000000000
8	0.876232220	0.082180000	-0.0142313130	0	0	0.0002399918
9	1.015944200	0.261005270	-0.0204338390	0	0	0.0000000000
10	0.093721991	-0.016963050	0.0002517223	0	0	0.0000000000

	V263	V264	V265	V266	V267	V268
1	0.036834974	0.69907930	1.0029358	0.6057044	0.3272992	-0.03220993
2	0.152704280	1.05740830	1.0074732	0.6907010	0.3157535	0.28835083
3	-0.039785985	0.33158967	0.9875608	1.0084808	0.8515058	0.67727968
4	-0.007803443	0.49251304	0.8621862	0.9961722	0.9997925	0.80340364
5	-0.050163059	0.42990744	1.0509992	0.9919064	0.7672667	0.56782255
6	0.000000000	-0.02915395	0.4554755	1.0128669	1.0125210	0.90024603
7	-0.013374913	0.09156774	0.6446338	1.0036885	0.9912670	1.03341000
8	-0.001187783	-0.03427793	0.2527428	0.9172896	0.2660115	-0.04178697
9	-0.006740451	0.02522307	0.6751016	0.9638044	0.9152760	1.00765110
10	-0.033916269	0.27583110	0.9543883	1.0112208	1.0057265	1.01286420

	V269	V270	V271	V272	V273
1	-0.0483053000	-0.04340691	-0.05751511	0.095567419	0.726512630
2	0.0517292690	-0.03854868	-0.03645139	0.055543437	0.503607070
3	0.6834143500	0.93037551	1.02467410	1.037509200	0.795024030
4	0.8457585600	0.83779180	0.83065472	1.047692100	0.896665260
5	0.3140132800	0.59428462	0.55164023	0.219243240	0.290728570
6	0.1521501700	-0.05072220	-0.01565375	-0.002343699	-0.003708012
7	0.6562671100	0.07299557	0.08615744	0.072504902	0.488453490
8	0.0006115537	0.00000000	0.00000000	0.000000000	0.000000000
9	1.0044779000	1.00900140	1.00536960	1.004048500	1.017502400
10	1.0181242000	0.91385250	0.54313404	0.513203380	0.619168870

	V274	V275	V276	V277	V278	
1	0.695366970	0.14711448	-0.01200487	-0.0003027982	0.000000e+00	
2	0.999676780	0.49954054	-0.02168850	-0.0020837929	6.114712e-05	
3	0.589514310	0.08735515	-0.03431594	-0.0034100967	4.565652e-04	
4	0.533587880	0.14868558	-0.01430973	-0.0001008476	0.000000e+00	
5	0.834079570	1.05518180	0.77846916	0.0525509940	-1.871713e-02	
6	-0.027341708	0.05894536	0.65709305	1.0357764000	5.271762e-01	
7	1.059593000	0.69313118	0.15592753	-0.0193967010	7.704537e-04	
8	-0.001115298	-0.02415504	0.43118255	0.7862130800	5.451419e-02	
9	1.039606700	1.01412050	0.77276649	0.6098818900	1.034760e-01	
10	1.011297200	0.71978505	0.08237757	-0.0180526110	7.490522e-04	

	V279	V280	V281	V282	V283	V284
1	0.000000e+00	0	0	-0.0006765727	-0.0065141556	0.117339360
2	0.000000e+00	0	0	-0.0093309164	0.0437851600	0.625597750
3	0.000000e+00	0	0	0.0000000000	-0.0054588072	0.019832857
4	0.000000e+00	0	0	-0.0002576934	-0.0063284212	0.082571504
5	4.306236e-04	0	0	0.0000000000	-0.0464167790	0.393847830
6	-2.078006e-02	0	0	0.0000000000	0.0000000000	-0.008571589
7	0.000000e+00	0	0	0.0000000000	0.0012637071	-0.018665816
8	-1.126532e-02	0	0	0.0000000000	0.0000000000	-0.002332874
9	-1.028135e-02	0	0	0.0000000000	-0.0007765684	-0.003083793
10	2.382898e-05	0	0	0.0000000000	-0.0051742693	0.016976154

	V285	V286	V287	V288	V289	V290
1	0.42194841	0.9932109	0.8820140	0.74575873	0.72387427	0.723341720
2	0.99868268	0.9471967	0.8445183	0.86564084	0.76012960	0.574255500
3	0.53901971	0.9914155	0.9968144	1.01788610	1.01155400	0.998026690
4	0.13331585	0.5851260	0.9798063	1.02565160	1.03666350	1.029254000
5	1.03523260	1.0150899	1.0345167	1.03145470	0.95625974	1.027946500
6	0.09169584	0.7857797	1.0284687	1.00779900	0.66302274	0.505559840
7	0.05654422	0.6668875	1.0211240	1.02513390	0.93081143	0.803803900
8	-0.01622161	0.6896726	0.7157428	0.02282449	-0.02360498	0.001302083

```
9   0.17422828 0.3113756 0.3367851 0.92161326  0.96199100 0.990081020
10  0.54554856 0.9565716 0.8113936 0.79328379  0.81322343 0.958003590
V291        V292        V293         V294         V295          V296
1  0.7200203  0.84532496 0.831859740  0.068883187 -0.027776501  0.0003591367
2  0.5762348  0.75872428 1.031090600  0.683323730  0.081617528 -0.0106374250
3  0.9354579  0.60077575 0.115578830  0.009907578 -0.005232247  0.0003507966
4  1.0096093  0.59916234 0.366240540  0.002474469 -0.020296330  0.0001348890
5  1.0187666  0.93408984 0.951134190  1.024546800  0.952450320  0.3159756900
6  0.2155755 -0.01289417 0.007763357  0.259128250  0.719692540  1.0215714000
7  0.8130092  0.80273199 0.995459810  0.889682750  0.134556590 -0.0441140390
8  0.0000000  0.00000000 0.000212792 -0.006116626  0.008305675  0.8390241900
9  0.9692474  0.98022569 0.838384920  0.394424510  0.391673530  0.0948667620
10 1.0316086  1.03294450 0.914452610  0.644388400  0.110930980 -0.0209250410
V297         V298          V299 V300 V301         V302
1   7.148693e-05  0.000000000  0.0000000000    0    0 1.531863e-04
2  -1.621732e-04  0.000000000  0.0000000000    0    0 3.844975e-04
3   0.000000e+00  0.000000000  0.0000000000    0    0 0.000000e+00
4   7.786969e-05  0.000000000  0.0000000000    0    0 6.433824e-05
5  -3.204032e-02  0.001090457  0.0000000000    0    0 0.000000e+00
6   6.786163e-01  0.096901438 -0.0107616760    0    0 0.000000e+00
7   1.142514e-03  0.000000000  0.0000000000    0    0 0.000000e+00
8   4.878182e-01 -0.034948433 -0.0016040305    0    0 0.000000e+00
9   3.958173e-02 -0.005574919 -0.0002782884    0    0 0.000000e+00
10  2.601103e-04  0.000000000  0.0000000000    0    0 0.000000e+00
V303          V304         V305        V306        V307
1   0.0003173536 -2.291672e-02 -0.004144029  0.38703845  0.50458344
2  -0.0198596160  1.099430e-01  0.648968840  0.77550711  0.77691085
3   0.0009098692 -1.709058e-02  0.124395540  0.32652854  0.32638026
4   0.0006796502 -1.312720e-02 -0.030502025  0.05274015  0.29234686
5  -0.0071990543  3.727863e-02  0.540630580  0.87806801  0.86681944
6   0.0000000000 -5.412582e-06 -0.014614617  0.10652602  0.55934891
```

```
7   0.0000000000  5.054126e-04 -0.022382013  0.08771463  0.50538276
8   0.0000000000 -7.659314e-06 -0.015515829  0.08573356  0.79654585
9   0.0001467531 -5.919628e-04 -0.013984664 -0.02091342 -0.01761826
10  0.0004867311 -7.637408e-03  0.036953295  0.25076632  0.04246657
V308       V309       V310        V311        V312         V313
1  0.77488588 0.99003745  1.0076948  1.00851440  0.73790504  0.215455290
2  0.77393796 0.77075027  0.7685812  0.77199202  0.78349784  0.610717490
3  0.32755131 0.32542484  0.3369604  0.24118697 -0.01373021 -0.015464361
4  0.56676967 0.75895932  0.8832599  0.52686644  0.02834642 -0.009586679
5  0.86770593 0.87766503  0.8678176  0.87079672  0.88986261  0.831552590
6  0.75017240 0.93290164  1.0498261  0.89571943  0.62439663  0.704029560
7  0.72228982 0.95574774  1.0409370  0.98887970  1.06797850  0.850660760
8  0.63573224 0.05545879 -0.0425994 -0.01671177 -0.01693389 -0.031262128
9  0.22966417 0.14435112  0.4222663  0.27817584  0.28311703  0.416415740
10 0.01438624 0.04593883  0.2647892  0.33076093  0.32872413  0.121454440
V314          V315          V316          V317         V318
1  -0.026962486  0.0013250613  0.000000e+00  0.000000e+00  0.000000000
2   0.068429705 -0.0189090580  3.235294e-04  3.584559e-05  0.000000000
3  -0.007300177 -0.0004304534  1.102941e-04  0.000000e+00  0.000000000
4  -0.005751634  0.0005438113  0.000000e+00  0.000000e+00  0.000000000
5   0.527473750  0.2841057000 -1.254590e-02 -7.801777e-04  0.000000000
6   1.003229800  0.9914636800  5.595433e-01 -4.534058e-03 -0.012336737
7   0.234439850 -0.0098320908 -1.272978e-03  0.000000e+00  0.000000000
8  -0.055219431  0.3318790300  8.803706e-01  1.653041e-01 -0.025095390
9  -0.032335937 -0.0216657650 -1.545229e-02 -9.727992e-03 -0.000403571
10 -0.001223805 -0.0124882050 -5.238971e-05  5.422794e-05  0.000000000
V319 V320 V321         V322         V323          V324
1  0.0000000000    0    0 0.000000e+00 0.000000e+00  2.363664e-04
2  0.0000000000    0    0 3.216912e-05 3.981187e-04 -8.110941e-03
3  0.0000000000    0    0 0.000000e+00 0.000000e+00  6.122855e-04
4  0.0000000000    0    0 0.000000e+00 0.000000e+00  1.210172e-05
```

```
5  0.0000000000    0    0 0.000000e+00 5.931271e-04 -9.007098e-03
6  0.0001307189    0    0 0.000000e+00 0.000000e+00  5.024510e-05
7  0.0000000000    0    0 0.000000e+00 0.000000e+00  2.190564e-05
8  0.0002634804    0    0 0.000000e+00 0.000000e+00  1.746324e-05
9  0.0001225490    0    0 0.000000e+00 0.000000e+00  2.588848e-05
10 0.0000000000    0    0 0.000000e+00 1.467531e-05  1.760110e-04
V325          V326          V327          V328          V329
1  -0.0022603145 -0.0251994490 -0.037388991  0.066212123  0.291134500
2   0.0056730664  0.0152478550  0.013971099  0.014042453  0.014741728
3  -0.0095008170 -0.0220447300 -0.021994179 -0.022076098 -0.021936275
4   0.0004886131 -0.0058008578 -0.021525412  0.001746854  0.013567640
5   0.0453836640  0.1692778500  0.171453020  0.170776140  0.170776140
6   0.0001725899 -0.0105664830 -0.024993175 -0.016948113  0.298888410
7   0.0005836908 -0.0079629800 -0.037580082  0.036593545  0.358675690
8   0.0007965686 -0.0304429810  0.191937010  0.872527110  0.799422060
9  -0.0004173305 -0.0008222529 -0.001004289 -0.016536525 -0.011961908
10 -0.0046927594 -0.0178083130 -0.005953482 -0.004363611 -0.006147059
V330        V331          V332          V333          V334
1   0.32305573  0.30626031  0.087607094 -0.0250581920  2.374387e-04
2   0.01567285  0.01548034  0.015301794  0.0025144545 -8.210904e-03
3  -0.02259161 -0.01719174 -0.002261132 -0.0001423199  3.931781e-05
4   0.18405651  0.02137289 -0.008318321 -0.0014463332  2.604167e-05
5   0.17071334  0.17133093  0.175457010  0.1446767200 -2.745740e-02
6   0.77373116  0.92670614  0.936280980  0.9332400500  8.917209e-01
7   0.56743981  0.41048346  0.610979950  0.2457078100 -3.020597e-02
8   0.33816500  0.19051420  0.193046360  0.3322883800  5.752145e-01
9  -0.02725705 -0.01930617 -0.019608456 -0.0264846010 -6.064645e-04
10 -0.01852640 -0.02225888 -0.022143587 -0.0104806160 -3.053156e-03
V335          V336          V337          V338          V339 V340
1   0.0000000000  0.000000e+00  0.000000e+00  0.0000000000 0.000000e+00    0
2   0.0003382353  3.584559e-05  0.000000e+00  0.0000000000 0.000000e+00    0
```

```
3   0.0000000000  0.000000e+00  0.000000e+00  0.0000000000 0.000000e+00    0
4   0.0000000000  0.000000e+00  0.000000e+00  0.0000000000 0.000000e+00    0
5  -0.0205346200 -1.089461e-03  1.571691e-04  0.0000000000 0.000000e+00    0
6   0.4317076400 -1.994511e-02 -5.387851e-03 -0.0001691737 4.901961e-05    0
7  -0.0003108660  1.902574e-04  0.000000e+00  0.0000000000 0.000000e+00    0
8   0.8571242500  3.575653e-01 -9.905501e-03 -0.0013252619 0.000000e+00    0
9  -0.0011534416 -4.753881e-05  7.506128e-06  0.0000000000 0.000000e+00    0
10 -0.0002219669  5.330882e-05  0.000000e+00  0.0000000000 0.000000e+00    0
V341 V342        V343          V344         V345          V346
1     0    0 0.00000e+00  0.0000000000  0.000000000  6.209392e-18
2     0    0 3.90528e-05 -0.0001031348 -0.004631690 -6.263911e-03
3     0    0 0.00000e+00  0.0000000000  0.000000000  0.000000e+00
4     0    0 0.00000e+00  0.0000000000  0.000000000  2.483757e-18
5     0    0 0.00000e+00  0.0005380947 -0.008096694 -2.261384e-02
6     0    0 0.00000e+00  0.0000000000  0.000000000  1.222942e-04
7     0    0 0.00000e+00  0.0000000000  0.000000000  5.215889e-18
8     0    0 0.00000e+00  0.0000000000  0.000000000  2.303208e-03
9     0    0 0.00000e+00  0.0000000000  0.000000000  0.000000e+00
10    0    0 0.00000e+00  0.0000000000  0.000000000  0.000000e+00
V347         V348         V349         V350         V351
1   0.0006726183 -0.011315141 -0.035464107 -0.038821491 -0.037107741
2  -0.0060657943 -0.006065794 -0.006065794 -0.006065794 -0.006065794
3   0.0000000000  0.000000000  0.000000000  0.000000000  0.000000000
4   0.0001426766 -0.003773070 -0.006056826 -0.024150259 -0.005549305
5  -0.0227736330 -0.022697811 -0.022697811 -0.022690473 -0.022762627
6  -0.0010109657 -0.005661028 -0.017466777  0.058854460  0.110567040
7   0.0005788594 -0.011236222 -0.016523175  0.003056133 -0.010929742
8  -0.0329124370  0.091328423  0.476611620  0.671783560  0.771763020
9   0.0000000000  0.000000000  0.000000000  0.000000000  0.000000000
10  0.0000000000  0.000000000  0.000000000  0.000000000  0.000000000
V352          V353          V354         V355         V356
```

1	-0.0133524930	0.0009909647	4.891770e-05	0.000000e+00	0.000000000					
2	-0.0061921650	-0.0040793901	1.263707e-04	0.000000e+00	0.000000000					
3	0.0000000000	0.0000000000	0.000000e+00	0.000000e+00	0.000000000					
4	0.0003840039	0.0000000000	0.000000e+00	0.000000e+00	0.000000000					
5	-0.0232261220	-0.0197725120	-5.727447e-04	9.172068e-05	0.000000000					
6	0.1085445500	0.1098522600	9.802057e-02	4.939311e-03	-0.006323275					
7	0.0058694101	-0.0130501940	-9.834495e-05	0.000000e+00	0.000000000					
8	0.7802778600	0.8113112300	6.481225e-01	2.151157e-01	-0.026218407					
9	0.0000000000	0.0000000000	0.000000e+00	0.000000e+00	0.000000000					
10	0.0000000000	0.0000000000	0.000000e+00	0.000000e+00	0.000000000					

	V357	V358	V359	V360	V361	V362	V363	V364	V365
1	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
2	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
3	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
4	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
5	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
6	0.0001391606	9.763199e-06	0	0	0	0	0	0	0
7	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
8	-0.0026325873	2.831328e-04	0	0	0	0	0	0	0
9	0.0000000000	0.000000e+00	0	0	0	0	0	0	0
10	0.0000000000	0.000000e+00	0	0	0	0	0	0	0

	V366	V367	V368	V369	V370
1	0.000000e+00	0.0000000000	0.0000000000	0.000000000	0.000000000
2	0.000000e+00	0.0000000000	0.0000000000	0.000000000	0.000000000
3	0.000000e+00	0.0000000000	0.0000000000	0.000000000	0.000000000
4	0.000000e+00	0.0000000000	0.0000000000	0.000000000	0.000000000
5	0.000000e+00	0.0000000000	0.0000000000	0.000000000	0.000000000
6	0.000000e+00	0.0000000000	0.0002282826	-0.001517906	-0.011813896
7	0.000000e+00	0.0000000000	0.0002486650	-0.002118736	-0.005507898
8	1.026684e-17	0.0004340278	-0.0145239700	-0.021785641	-0.012439066
9	0.000000e+00	0.0000000000	0.0000000000	0.000000000	0.000000000

```
10 0.000000e+00 0.0000000000  0.0000000000  0.000000000  0.000000000
V371         V372         V373         V374         V375
1   0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
2   0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
3   0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
4   0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
5   0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
6  -0.017265795 -0.017049803 -0.01716875 -0.015992307 -0.004499421
7  -0.003019812 -0.006116217 -0.00141063  0.000120847  0.000000000
8  -0.006684709 -0.006773557 -0.01175299 -0.019918471 -0.018700640
9   0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
10  0.000000000  0.000000000  0.00000000  0.000000000  0.000000000
V376         V377 V378 V379 V380 V381 V382 V383 V384 V385 V386
1   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
2   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
3   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
4   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
5   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
6   0.0003953908 8.680556e-06    0    0    0    0    0    0    0    0    0
7   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
8  -0.0022560934 2.042484e-04    0    0    0    0    0    0    0    0    0
9   0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
10  0.0000000000 0.000000e+00    0    0    0    0    0    0    0    0    0
V387 V388 V389 V390 V391 V392 V393 V394 V395 V396 V397 V398 V399 V400
1     0    0    0    0    0    0    0    0    0    0    0    0    0    0
2     0    0    0    0    0    0    0    0    0    0    0    0    0    0
3     0    0    0    0    0    0    0    0    0    0    0    0    0    0
4     0    0    0    0    0    0    0    0    0    0    0    0    0    0
5     0    0    0    0    0    0    0    0    0    0    0    0    0    0
6     0    0    0    0    0    0    0    0    0    0    0    0    0    0
7     0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0