机器学习课程实验报告

多项式拟合正弦曲线

学　　号　1180301007

姓　　名　赵锦涛

实验时间　2020 年 9 月

一、 实验目的:

掌握最小二乘法求解（无惩罚项的损失函数）、掌握加惩罚项（2 范数）的损失函数优化、梯度下降法、共轭梯度法、理解过拟合、克服过拟合的方法 (如加惩罚项、增加样本)。

二、 实验要求:

1) 生成数据，加入噪声；

2) 用高阶多项式函数拟合曲线；

3) 用解析解求解两种 loss 的最优解（无正则项和有正则项）；

4) 优化方法求解最优解（梯度下降，共轭梯度）；

5) 用你得到的实验数据，解释过拟合；

6) 用不同数据量，不同超参数，不同的多项式阶数，比较实验效果；

7) 语言不限，可以用 matlab，python。求解解析解时可以利用现成的矩阵求逆。梯度下降，共轭梯度要求自己求梯度，迭代优化自己写。不许用现成的平台，例如 pytorch，tensorflow 的自动微分工具。

三、 实验环境:

Python 3.8, Windows 10

四、 实验原理:

（一）数据生成

训练数据通过 Python Numpy 库在指定区间内均匀选定 n 个 x 坐标点（数据点的数目可指定），然后调用 Numpy 库中的 sin 函数计算对应 $sin(2\pi x)$ 的 y 坐标值，接着调用 Python Random 库的 gauss 函数为数据加入噪声，其中高斯分布的均值设为 0，标准差设为 0.016。图 1 为在 [0,1] 区间上生成的 40 个数据点：

（二）最小二乘法求解

给定 n 个正弦数据点 $[(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)]$，使用形如 $y(x, w) = w_0 + w_1 x + \cdots + w_m x^m = \sum_{i=0}^{m} w_i x^i$ 的多项式拟合曲线，记 $\boldsymbol{x_i} = [1, x, x^2, ..., x^m]^T$，

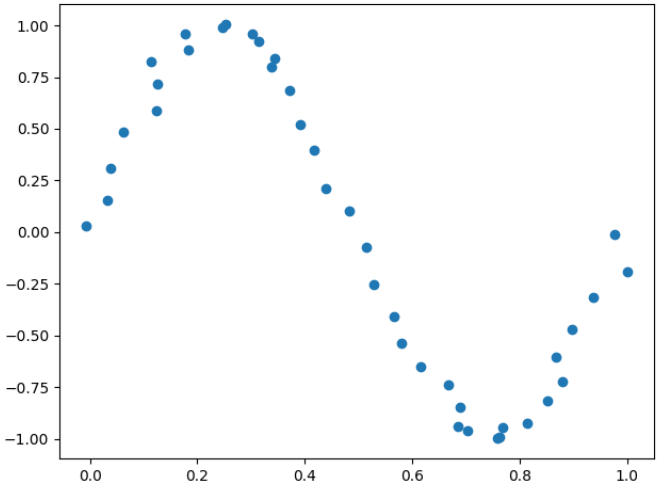

图 1: 生成数据点示例

$\boldsymbol{y} = [y_1, y_2, ..., y_n]^T$， $\boldsymbol{w} = [w_0, w_2, ..., w_m]^T$， $\boldsymbol{A} = [\boldsymbol{x_1}^T, \boldsymbol{x_2}^T, ..., \boldsymbol{x_n}^T]^T$。建立误差函数

$$E(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{y})^T(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{y})$$

1. 无正则项

求导得，

$$\frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} = \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w} - \boldsymbol{A}^T\boldsymbol{y}$$

令导数为 0，有

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w} = \boldsymbol{A}^T\boldsymbol{y}$$

若 $\boldsymbol{A}$ 列满秩，有

$$\boldsymbol{w} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}$$

2. 有正则项

加入正则项，误差函数变为

$$E(\boldsymbol{w}) = \frac{1}{2}[(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{y})^T(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{y}) + \lambda \boldsymbol{w}^T\boldsymbol{w}]$$

求导得，

$$\frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} = \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w} - \boldsymbol{A}^T\boldsymbol{y} + \lambda \boldsymbol{w}$$

令导数为 0，同理可解得

$$\boldsymbol{w} = (\boldsymbol{A}^T\boldsymbol{A} + \ \lambda \boldsymbol{E})^{-1}\boldsymbol{A}^T\boldsymbol{y}$$

（三）梯度下降法

梯度的方向是函数增长最快的方向，已知误差函数 $E(\boldsymbol{w})=\frac{1}{2}(\boldsymbol{A}\boldsymbol{w}-\boldsymbol{y})^T(\boldsymbol{A}\boldsymbol{w}-\boldsymbol{y})$，可以通过梯度下降的方法来迭代求得最小误差以及对应的参数值。

对于 $\boldsymbol{w}$ 向量，其梯度表达式为

$$\frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}}=\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w}-\boldsymbol{A}^T\boldsymbol{y}$$

设学习率为 α，则可以用当前 $\boldsymbol{w}$ 减去当前的梯度来更新 $\boldsymbol{w}$ 的值，迭代该过程，从而不断逼近使误差函数取最小时的 $\boldsymbol{w}$。迭代计算公式为

$$\boldsymbol{w}_{i+1}=\boldsymbol{w}_i-\alpha\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{w}_i-\boldsymbol{y})$$

加入正则项后（误差函数与解析法有正则项时相同），同理可得迭代计算公式

$$\boldsymbol{w}_{i+1}=(1-\lambda\alpha)\boldsymbol{w}_i-\alpha\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{w}_i-\boldsymbol{y})$$

（四）共轭梯度

二次型，是关于向量的二次数值型函数，形如：

$$f(\boldsymbol{x})=\frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}^T\boldsymbol{x}+c$$

由线性代数知识可得，当 $\boldsymbol{A}$ 为对称正定阵时，$f(\boldsymbol{x})$ 的最小值由 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 的解给出。由上文讨论可得到，误差函数为

$$\begin{aligned}E(\boldsymbol{w})&=\frac{1}{2}(\boldsymbol{A}\boldsymbol{w}-\boldsymbol{y})^T(\boldsymbol{A}\boldsymbol{w}-\boldsymbol{y})\\&=\frac{1}{2}(\boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w}-2\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w}+\boldsymbol{y}^T\boldsymbol{y})\end{aligned}$$

令 $\boldsymbol{A'}=\boldsymbol{A}^T\boldsymbol{A},\boldsymbol{b}=\boldsymbol{A}^T\boldsymbol{y},c=\boldsymbol{y}^T\boldsymbol{y}$, 误差函数可化为二次型

$$E(\boldsymbol{w})=\frac{1}{2}\boldsymbol{w}^T\boldsymbol{A'}\boldsymbol{w}-\boldsymbol{b}^T\boldsymbol{w}+c$$

所以求误差函数最小可以通过求 $\boldsymbol{A'}\boldsymbol{w}=\boldsymbol{b}$ 的解得出。

最速下降法在收敛中其早期的迭代步骤往往在沿着同一个方向，而共轭梯度法的思想是选择一系列正交的搜索方向 $\boldsymbol{d_0},...\boldsymbol{d_{n-1}}$，在每个方向上移动适当距离，使得刚好消除误差向量在该方向上的分量，当迭代结束后，会移动到最小值点。在这里引入残差向量的概念

$$\boldsymbol{r_i}=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_i$$

由线性代数知识，可给出共轭梯度法计算过程：

$$\boldsymbol{d_0} = \boldsymbol{r_0} = \boldsymbol{b_0} - \mathbf{A}\boldsymbol{x_0}$$

$$\alpha_i = \frac{\boldsymbol{r_i}^T \boldsymbol{r_i}}{\boldsymbol{d_i}^T \mathbf{A} \boldsymbol{d_i}}$$

$$\boldsymbol{x_{i+1}} = \boldsymbol{x_i} + \alpha_i \boldsymbol{d_i}$$

$$\boldsymbol{r_{i+1}} = \boldsymbol{r_i} - \alpha_i \mathbf{A} \boldsymbol{d_i}$$

$$\beta_{i+1} = \frac{\boldsymbol{r_{i+1}}^T \boldsymbol{r_{i+1}}}{\boldsymbol{r_i}^T \boldsymbol{r_i}}$$

$$\boldsymbol{d_{i+1}} = \boldsymbol{r_{i+1}} + \beta_{i+1} \boldsymbol{d_i}$$

五、 实验结果与分析

（一）最小二乘法求解

如图 2，当阶数为 10，数据点个数为 11 时，出现过拟合现象。为解决过拟合

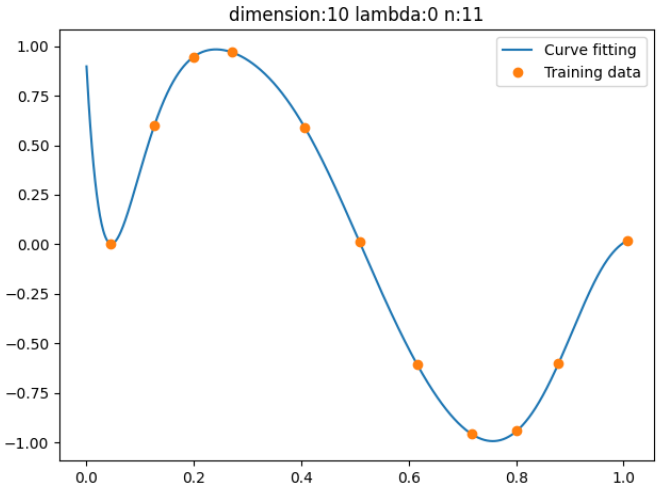

图 2: 最小二乘法求解一

现象，可加入正则项，令 $lambda = 0.001$，如图 3。

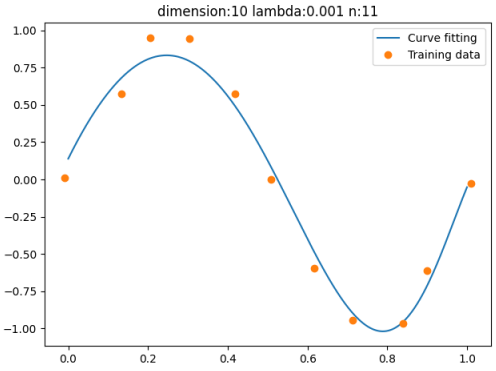

图 3: 最小二乘法求解二

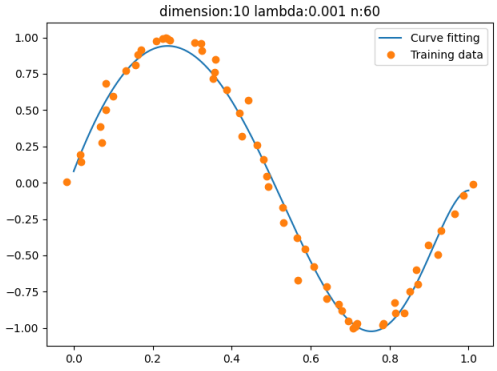

图 4: 最小二乘法求解三

也可增加数据点数量以解决过拟合现象，令 $n = 60$，如图 4。

（二）梯度下降法

当阶数为 10，数据点个数为 11 时，令正则项系数 $lambda = 0$，观察图像右端出现过拟合现象，令 $lambda$ 分别为 0.001 和 0.005，对比可发现正则项系数的增大一定程度上抑制了过拟合现象，但也对模型产生了负面影响。当增加样本点

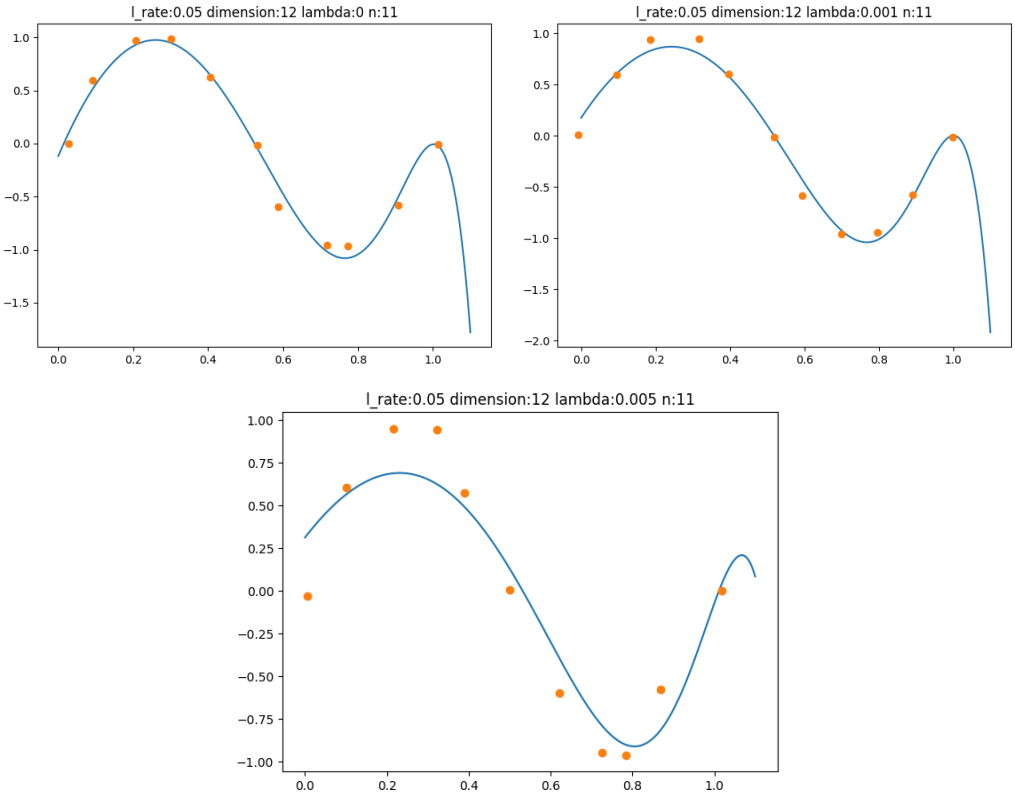

图 5: 梯度下降

使 $n = 60$ 后，可发现训练效果较好，也未出现显著的过拟合现象。在实验中还发

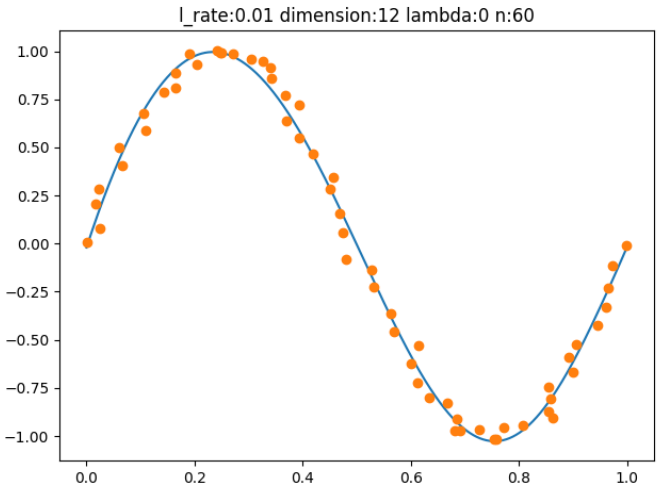

图 6: 梯度下降法，$n = 60$

现，相较于最小二乘求解方法，梯度下降法训练时间一般较长，收敛速度较慢。对于阶数为 12，数据点数为 60 的数据，最小二乘方法所需时间为 0.1698 秒，而学习率为 0.05 的梯度下降法训练同样的数据所需时间为 103.4624 秒。

（三）共轭梯度法

如图 7，为使用共轭梯度法拟合曲线效果，阶数为 12，数据点个数为 60，迭代次数也为 60。从图中可以看出拟合效果较好，未出现过拟合现象。除此之外，运行时间较短，仅为 0.1317 秒

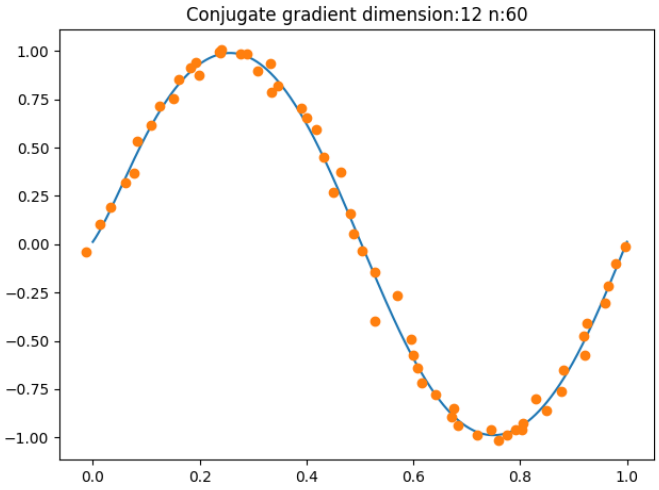

图 7: 共轭梯度法

（四）总结

从实验结果中可以得到，数据点越多，训练得到的模型拟合效果越好。其次，模型的复杂度与参数个数有关，模型参数越多，也即设置的多项式阶数越高，其拟合能力越强。当参数过多时，会出现过拟合现象，每个数据点都落在拟合曲线上，但此时拟合出的曲线离正弦曲线相差较远，此时可以通过加入正则项或者叫做惩罚项来调整模型的复杂度以降低过拟合现象，或者增多样本数据，避免数据过少导致的特殊曲线情况。

加入正则项后，正则项的系数会影响模型训练的好坏。当正则项系数设置过小时，难抑制过拟合现象。当正则项系数设置过大时，模型复杂度主要与参数有关，数据对模型的影响比较小，导致学习能力较差，会出现欠拟合现象。需要正确设置正则项系数，才能获得较好的模型训练效果。

在梯度下降中，需要设置学习率，当学习率设置的过小时，收敛过程将变得十分缓慢。而当学习率设置的过大时，梯度可能会在最小值附近来回振荡，甚至可能无法收敛。所以需要合理设置学习率，根据相关文献资料，可以使学习率服从指数衰减，这样可以加快收敛速度，同时避免在极值点附近出现过分振荡。共轭梯度法对比梯度下降法，其收敛速度较快。原因是使用了相互正交的搜索向量，迭代次数更少，收敛速度更快。

六、 代码实现:

本次实验共有 7 个文件，其名称和作用分别为：

- *datagen.py*　生成训练数据
- *curveplot.py*　绘制拟合曲线
- *loss.py*　计算损失函数
- *linear_regression.py*　最小二乘法拟合曲线
- *gradient.py*　梯度下降法拟合曲线
- *gradient_gpu.py*　使用 Cupy 库的梯度下降法拟合曲线，可使用 CUDA 进行 GPU 矩阵运算
- *conjugate_gradient.py*　共轭梯度法拟合曲线

实现细节请见具体代码文件。

参考文献

[1] 断鸿声里，立尽斜阳. 共轭梯度法通俗讲义 | 断鸿声里，立尽斜阳 [EB/OL].https://flat2010.github.io/2018/10/26/共轭梯度法通俗讲义/#8-共轭梯度法,2018-10-25.

[2] 刘建平 Pinard. 梯度下降（Gradient Descent）小结 [EB/OL].https://www.cnblogs.com/pinard/p/5970503.html,2016-10-17.

[3] LLLiuye. 学习率 (Learning rate) 的理解以及如何调整学习率 [EB/OL].https://www.cnblogs.com/lliuye/p/9471231.html,2018-08-13.