哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称： 机器学习

课程类型： 选修

实验题目： 多项式拟合正弦函数

学号：1160300312
姓名：靳贺霖

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、实验要求及实验环境

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。

实验环境：

操作系统：windows 7
编译环境：python3.7,
编译器：PyCharm

三、设计思想（本程序中的用到的主要算法及数据结构）

1．算法原理

本次实验主要是使用多项式来拟合局部正弦函数。这个思想是由泰勒展开联想得到的。正弦函数在 x = 0 处的泰勒展开如下所示：

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots + (-1)^{m-1}\frac{x^{2m-1}}{(2m-1)!} + \mathrm{o}\left(x^{2m-1}\right)$$

这样，我们就可以通过设假设空间为多项式函数来拟合正弦函数。而且，对于不同的函数，都是可以使用正弦函数来拟合的。

然后我们要解决的问题就是参数求解问题。这里首先求得解析解，然后通过梯度下降和共轭梯度下降来求得优化解。

对于解析解，我们是通过建立误差函数：

$$\mathrm{E(w)} = \frac{1}{2m}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{w}) - t_n\}^2$$

其中 w 是我们要确认的参数矩阵，xi，ti 为每一个给定数据点的横坐标和纵坐标。然后我们只需要求在 w 取何值的时候误差函数最小即可。这里我们是通过求导，导函数取 0 得到 w 的解来得到解析解的。

梯度下降求优化解可以被类比为一个下山的过程。每次都选择自己当前位置最陡峭的位置下山，我们就认为是最快的方式。但是这样求解会陷入局部最优解，但是由于我们要

求解的函数是凸函数，故不存在局部最优解。每次通过求解函数在当前点的梯度，然后就找到了斜率最大的方向，朝着这个方向行进一个定义好的步长，直到斜率非常小的时候停止。

共轭梯度法是梯度下降法的一种优化，它克服了梯度下降有时存在的收敛慢的问题，通过计算出步长和最速下降方向来在 n~3n 次迭代内就可以获得比较好的解。

共轭梯度算法的原型是用来求形如 **Ax = b** 这样的方程的解的，其中 A 必须是对称的，正定的。事实上，这样的方程可以变换为求解二次泛函

$$\emptyset(\mathrm{x}) = \frac{1}{2}x^T Ax - b^T x$$

的最小值。可以验证$\emptyset'(x) = Ax - b$，导函数等 0，即为原始方程 Ax = b 的解。这里主要用到的是二次泛函以及共轭超平面等相关公式得到的结果，和梯度下降不同的是，共轭梯度法每次行的方向必然是朝着优化解的。算法步骤繁琐，不在此赘述。

2. 算法的实现

拟合函数为 sin(2πx)，取一个周期为拟合区间，即[0, 1]。首先要做的是数据的生成以及噪声的添加。为了保证数据的均匀分布，对于要生成的 n 个数据，将区间分为 n 份，每份中有一个点在其中。对于噪声的添加，我们对每个生成的 y 值添加服从分布$\mathrm{N}(0,\ 0.1)$的噪声。这样就完成了数据点的生成。这些数据点，按照 7：3 的比例分别划分为训练集和测试集。

因为要充分利用 python 矩阵运算的优势，所有的运算都在矩阵下运行。在矩阵表示中，代价函数可以被表示为 **E =** $\frac{1}{2m}(\boldsymbol{XW} - \boldsymbol{T})^{\boldsymbol{T}} \cdot (\boldsymbol{XW} - \boldsymbol{T})$，其中

$$\mathrm{X} = \begin{bmatrix} 1 & \cdots & {x_1}^m \\ \vdots & \ddots & \vdots \\ 1 & \cdots & {x_n}^m \end{bmatrix} \qquad \mathrm{W} = \begin{bmatrix} w_0 \\ \vdots \\ w_m \end{bmatrix} \qquad \mathrm{T} = \begin{bmatrix} t_1 \\ \vdots \\ t_n \end{bmatrix}$$

矩阵中的x_1，x_2 ...，x_n为样本横坐标，t_1，t_2 ...，t_n为样本横坐标对应的纵坐标，w_0，w_2 ...，w_m为 m 阶多项式中的系数，即我们要求的拟合方程的参数。这样，我们可以直接对代价函数 E 进行关于系数矩阵求导，得到

$$\frac{\partial E}{\partial W} = \frac{1}{2m}(X^T XW - X^T T)$$

令导函数为 0，得到解析解为

$$\mathrm{W} = (X^T X)^{-1} X^T T$$

因为$X^T X$为正定阵，所以一定可以求逆，故这个解析解一定是存在的。

对于添加正则项的表示，代价函数为 **E =** $\frac{1}{2\boldsymbol{m}}((\boldsymbol{XW} - \boldsymbol{T})^{\boldsymbol{T}} \cdot (\boldsymbol{XW} - \boldsymbol{T}) + \frac{\lambda}{2}\boldsymbol{W^T W})$，其中λ为超参数。求导，导函数得 0，得到添加了正则项的解析解为

$$\mathrm{W} = (X^T X + \lambda)^{-1} X^T T$$

梯度下降法其实就是每次往梯度方向下降一个步长。梯度的求解其实就是 E 对 W 求偏倒。当移动一个步长但是损失函数的差值变化不大时，停止移动，返回 W 即可。[3]

前面我们已经得到 E = $\frac{1}{2m}(XW - T)^T \cdot (XW - T)$，展开后得到公式

$$\frac{1}{2m}(W^T X^T XW - X^T T)$$

发现其与共轭梯度法中的二次泛函形式类似，即令A = X^TX，而且分母的 m 并不影响结果。这样，我们就可以使用共轭梯度法来求解。算法过程如下所示：[2]

$$
\begin{aligned}
&\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0 \\
&\mathbf{p}_0 := \mathbf{r}_0 \\
&k := 0 \\
&\text{repeat} \\
&\quad \alpha_k := \frac{\mathbf{r}_k^{\mathsf{T}} \mathbf{r}_k}{\mathbf{p}_k^{\mathsf{T}} \mathbf{A} \mathbf{p}_k} \\
&\quad \mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k \\
&\quad \mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k \mathbf{A} \mathbf{p}_k \\
&\quad \text{if } r_{k+1} \text{ is sufficiently small, then exit loop} \\
&\quad \beta_k := \frac{\mathbf{r}_{k+1}^{\mathsf{T}} \mathbf{r}_{k+1}}{\mathbf{r}_k^{\mathsf{T}} \mathbf{r}_k} \\
&\quad \mathbf{p}_{k+1} := \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k \\
&\quad k := k + 1 \\
&\text{end repeat} \\
&\text{The result is } \mathbf{x}_{k+1}
\end{aligned}
$$

四、实验结果与分析

这里的 n 代表数据点数量，m 代表多项式阶数大小。首先我们先来确定超参数λ的大小。理论上λ是无法准确确认的，只能通过多次实验不同参数找到最适合的超参数。取 n = 10，m = 9 这种会出现过拟合的情况，得到下表数据：

lambda	0.00001	0.0001	0.001	0.01	0.1	1
test 1	0.0736	0.0812	0.3917	0.1622	0.4102	0.4549
test 2	0.2221	0.0963	0.4724	0.2448	0.3525	0.5326
test 3	0.1084	0.0815	0.4108	0.2814	0.4145	0.5882
test 4	0.1707	0.0721	0.1281	0.2359	0.3385	0.6556
test 5	0.3235	0.0782	0.3773	0.2382	0.4629	0.4072
average	0.1797	0.0818	0.3561	0.2325	0.3957	0.5278
rank	2	1	4	3	5	6

由图表可以很直观的看出。当λ = 0.0001 的时候拟合度较高。

然后我们来测试在多项式阶数不同的时候拟合程度的好坏。这里取 n = 10，超参数λ = 0.0001，分别对不同的方法进行测试，得到如下的结果：

m	1	2	3	4	5	6	7	8	9
解析解	0.4859	0.4685	0.0829	0.1003	0.0764	0.0723	0.3976	0.4594	9.3841
解析+正则	0.3948	0.4104	0.0839	0.0975	0.1129	0.1636	0.0925	0.0605	0.0578
梯度下降	0.4512	0.3560	0.2524	0.1300	0.0800	0.0923	0.0562	0.0654	0.0687
共轭梯度	0.4125	0.3967	0.1297	01209	0.0511	0.0881	0.0763	0.0831	0.0416

可以看到，解析解在 m = 3~6 时候拟合较好，当 m 逐渐逼近 9 时候，因为模型的能力过于强大，已经开始出现了过拟合现象。这里给出 m=6 和 m=9 的拟合图片进行对比

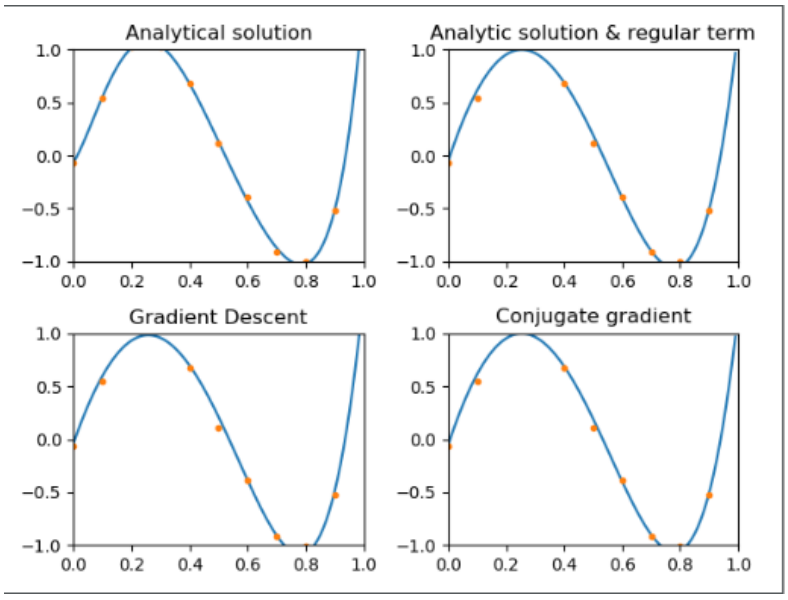

figure1：n=10,m=6 时不同拟合方案对比

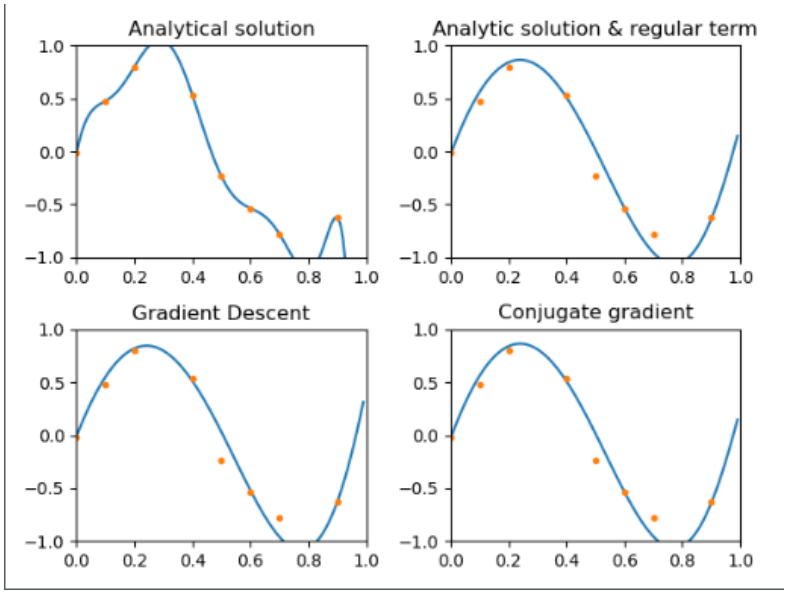

figure2：n=10,m=9 时不同拟合方案对比

明显看出解析解在 m=9 时候完全拟合了所有的点，但是并不是我们需要的结果，出现了过拟合现象。

然后我们来比较数据量不同对拟合程度的影响。这里取 m = 6, λ = 0.0001，对于不同的数据量，得到如下的结果

n	10	20	30	50	100
解析解	0.2055	0.0880	0.0717	0.0353	0.0350
解析+正则	0.1792	0.0698	0.0723	0.0317	0.0412
梯度下降	0.2400	0.0770	0.0719	0.0371	-----
共轭梯度	0.1792	0.0700	0.0723	0.0317	nan

首先这个表格中有一项没有给出，以及有一项为 nan。没有给出的原因是梯度下降在数据量很大的时候计算时间会明显提高，这里没有在能及的时间范围内测出来。共轭梯度在数据量很大的过程中出现了 nan，是因为迭代次数过多而导致迭代爆炸导致的。忽略这两个数据，

可以明显发现，无论是那种方法，数据量越大，得到的结果越好。但是当数据量大到一定程度的时候，这种提升变得不那么大。我认为是这种阶数情况下的拟合能力是有一个上限的，数据量大没改进，如果需要更加精确的结果就需要更换模型来实现。这里给出当 n=50 时不同方法都拟合较好的图片

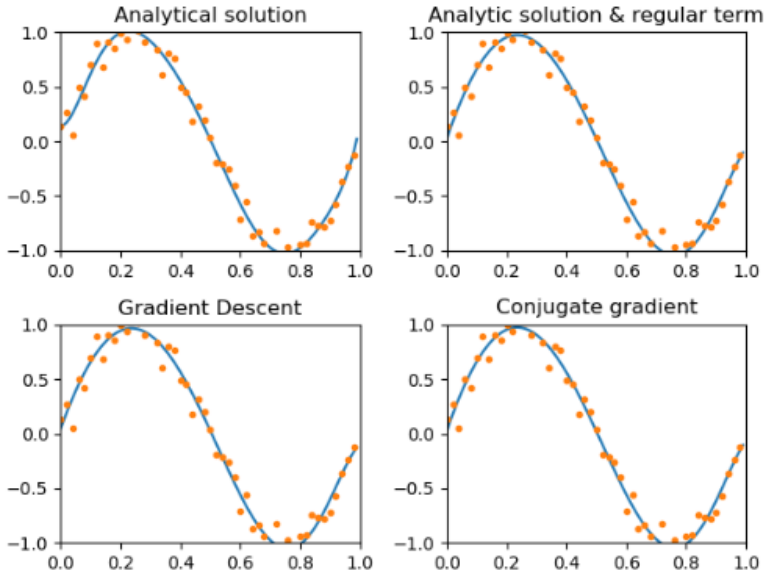

figure3：n=50,m=6 时不同拟合方案结果

五、结论

由以上的实验结果可以得到以下的结论：

1. 当 n = 10，m = 9 时，超参数λ = 0.0001 会有比较好的拟合结果
2. 当模型的拟合能力过于强大时，会因为过拟合而得到对于训练集拟合较好但是对测试集拟合不好的结果。加入惩罚项可以解决这个问题
3. 对于不同的数据量进行拟合，数据量越大拟合的效果越好，但是这种拟合的效果可能存在一个上界。
4. 对于解析解，解析解+正则，共轭梯度，梯度下降法拟合曲线，在没有过拟合的情况下，不同的方法的拟合能力大致相同。

六、参考文献

[1] Jonathan Richard Shewchuk, *An Introduction tothe Conjugate Gradient MethodWithout the Agonizing Pain*, August 4, 1994

[2] https://en.wikipedia.org/wiki/Conjugate_gradient_method

[3] https://en.wikipedia.org/wiki/Gradient_descent

七、附录：源代码（带注释）

已将源代码发送。