Automatic Model Selection Techniques for RBF Network

高枫 2015011208
gaof15@mails.tsinghua.edu.cn

Abstract

本文基于 Reversible Jump MCMC 模型选择技术，横向研究了 Simulated Annealing 和 AIC 等方法在 Radial Basis Function(RBF) 网络选择问题上的应用性能，此外，还将 RJ-MCMC 方法与针对 RBF 网络模型进行优化后的 Orthogonal Least Squares Learning(OLS) 方法进行了比较，从而给出了 RBF 网络模型自动选择技术的综合性说明。

***Keyword**: RBF Network, Metropolis-Hastings, RJMCMC, Simulated Annealing, AIC, OLS*

I. Introduction

Radial Basis Function(RBF) 网络是一种经典的神经网络结构，由于 RBF 网络能够逼近任意的非线性函数，RBF 网络通常被应用于处理系统内难以解析的规律性，具有良好的泛化能力，并有很快的学习收敛速度，现已成功应用于多种场景。而如何选择 RBF 网络模型的参数便成为了实际应用 RBF 模型的首要困难。

经典的 RBF 模型选择标准有很多种，如 Akaike 的 AIC 准则 [1]、Schwarz 的 MDL 准则 [2]，Andrieu 的 Reversible Jump MCMC Simulated Annealing(RJMCMCSA) 方法 [3] 等，其中 AIC 等准则可以用来评判不同模型间的优劣，而 RJMCMCSA 方法中使用 RJMCMC 过程进行模型更新，然后用模拟退火算法进行模型选择。使用 RJMCMC 方法的好处在于我们可以使用任意维度的初始模型开始训练，算法会自动在不同模型间跳转，直到找到“最优”模型。鉴于这种方法的优越性，本文中将 AIC 准则与 RJMCMC 方法相结合，比较了其与 RJMCMCSA 方法在 RBF 模型选择问题上的性能。此外，为了更全面地研究 RBF 模型选择方法，本文中还将展示 RJMCMC 方法替换成了 Orthogonal Least Squares(OLS) 方法 [4] 之后的选择性能。总而言之，本文的目标在于全面细致地展示当下几种 RBF 模型选择方法，并将不同方法间的性能优劣进行比较。

本文的结构安排如下，*II* 中将介绍一种传统的 MCMC 方法，Metropolis-Hastings 算法，以及改进之后的 RJMCMC 方法，在 *III* 中对 RBF 网络模型进行简单的介绍，然后在 *IV* 中分别介绍 SA、AIC 方法以及 OLS 方法，并阐述其实现过程，*V* 中会介绍将上述几种方法实际测试之后的结果，分析比较不同方法间的性能差异，最后在 *VI* 将对上述全部 RBF 模型选择方法进行总结。

II. Markov Chain Monte Carlo

A. Standard MCMC

马尔科夫蒙特卡洛 (Markov Chain Monte Carlo, MCMC) 方法，是用一组 Markov Chain 从随机分布中取样的算法，得到的序列可用于估计该概率分布或计算积分等。但不同于以往的蒙特卡洛 integration 是统计独立的，MCMC 中的是统计相关的。

在采用 MCMC 方法时马尔科夫链转移核的构造至关重要，不同的转移核构造方法将产生不同的 MCMC 方法，目前常用的 MCMC 方法主要有两种 Gibbs 抽样和 Metropo-Lis-Hastings 算法 (Algorithm 1)[5][6]。

本文将以 Metropolis-Hastings 算法为主进行介绍，后面提到的 RJMCMC 方法就是 MH 方法的一种扩展算法，其他很多的 MCMC 方法也都基于这种基本算法。Metropolis-Hastings 算法一般用于从多变量分布中采样，对于单变量分布而言，常会使用自适应判别采样 (adaptive rejection sampling) 等其他能抽取独立样本的方法，而不会出现 MCMC 中样本自相关的问题。

Algorithm 1 Metropolis-Hastings algorithms

Initialization:

a. Pick an initial state x_0

b. Set $t = 0$

Iteration t:

a. Generate: randomly generate a candidate state x' according to $g(x'|x_t)$

b. Calculate: calculate the acceptance probability $A(x'|x_t) = \min(1, \frac{P(x')g(x_t|x')}{P(x)g(x'|x_t)})$

c. Accept or Reject:

a. generate a uniform random number $u \in [0, 1]$

b. if $u \leq A(x'|x_t)$, accept the new state and set $x_{t+1} = x'$

c. if $u > A(x'|x_t)$, reject the new state and set $x_{t+1} = x_t$

d. Increment: set $t = t + 1$

B. Reversible Jump MCMC

可逆跳跃马尔科夫蒙特卡洛 (Reversible Jump Markov Chain Monto Carlo,RJMCMC) 方法 (Algorithm 2)[7][8] 是上一小节中介绍的标准 MCMC 方法的扩展，其好处在于可以模拟变维空间中的后验分布，即使模型参数是未知的，我们仍然可以进行模拟采样的过程。因此这种方法可以应用于各种复杂的模型选择问题，RBF 网络模型的选择就是其重要的应用场景之一。

在 RJMCMC 过程中，每次先在同一维度的模型中更新模型参数，然后再在不同维度的模型间跳转，然后用接受概率决定是否更新模型。这种方法与传统 MCMC 方法的区别就在于它会在不同模型间跳转，这就使得变维空间中的抽样变得可实现，从而简化了模型选择算法。

III. RBF Network

径向基函数网络（Radial basis function network，RBF network）是一种使用径向基函数作为激活函数的人工神经网络。径向基函数网络的输出是输入的径向基函数和神经元参数的线性组合。

径向基函数网络通常有三层：输入层、隐藏层和一个非线性激活函数和线性径向基神经网络输出层。输入可以被建模为实数向量。输出是输入向量的一个标量函数。

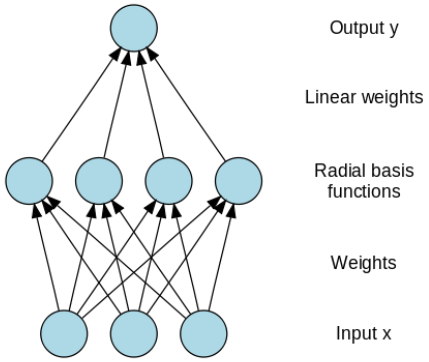


Figure 1: Architecture of a radial basis function network.An input vector x is used as input to all radial basis functions, each with different parameters. The output of the network is a linear combination of the outputs from radial basis functions.

RBF 模型可以简单表示为以下形式 [3]：

$$y_{N\times c} = D(\mu_{k\times d}, x_{N\times d})\alpha_{(1+d+k)\times c} + n_{N\times c}$$

y 为观测样本，N 为样本点数，c 为每个样本点的维度，x 为 N 个观测样本点对应的变量，d 为每个变量的维度，矩阵 D 有如下形式

$$D = \begin{bmatrix} 1 & x_{1,1} & \dots & x_{1,d} & \phi(x_1,\mu_1) & \dots & \phi(x_1,\mu_k) \\ 1 & x_{2,1} & \dots & x_{2,d} & \phi(x_2,\mu_1) & \dots & \phi(x_2,\mu_k) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N,1} & \dots & x_{N,d} & \phi(x_N,\mu_1) & \dots & \phi(x_N,\mu_k) \end{bmatrix}$$

$\phi(x, \mu)$ 表示 RBF 函数，k 为 RBF 函数的个数，μ 为函数的中心，α 为加权系数，n 是服从高斯分布的白噪声，$n_i \sim \mathcal{N}(0, \sigma_i^2), i = 1, 2, ..., c$。RBF 模型的参数集合可用 $\mathcal{M}_s = (\mu, \alpha, \sigma^2, k) = (\theta, k)$ 表示。

* 更多计算细节在 [3] 中。

IV. Model Selection for RBFN

A. Reversible Jump Simulated Annealing

Reversible Jump MCMC Simulated Annealing 算法 [3] 是 Andrieu 提出的一种结合了 RJMCMC 和模拟退火算法的模型选择算法。其过程见 Algorithm 2。模拟退火

(Simulated Annealing)[9] 是一种通用概率算法，用来在固定时间内寻求在一个大的搜寻空间内找到的最优解。

使用了模拟退火算法的 MH 步骤中的接受概率变为

$$A_{SA} = \min\{1, \frac{\pi^{(1/T_i-1)}(z^*)}{\pi^{(1/T_i-1)}(z)}\}$$

其中 T_i 是递减的 cooling schedule，满足 $lim_{i\to\infty}T_i = 0$，$\pi(z)$ 是样本的随机分布。

在 RBF 模型选择问题中，共有五种 Move 方式，分别是 Birth Move, Death Move, Split Move, Merge Move 和 Update，每次迭代中发生这几种 move 的概率分别为 b_k, d_k, s_k, m_k 和 u_k，满足 $b_k + d_k + s_k + m_k + u_k = 1$。其中 Birth Move 和 Split Move 允许 RBF 网络的隐层维度从 k 增长到 k+1，Death Move 和 Merge Move 允许维度从 k 减少到 k-1。

Algorithm 2 Reversible Jump Simulated Annealing

Initialization:

Set $(k^{(0)}, \theta^{(0)}) \in \Theta$

Iteration i:

a. Sampling u U[0,1] and set the temperature with a cooling schedule

b. if $u \le b_{k^{(i)}}$

do "birth" move

else if $u \le b_{k^{(i)}} + d_{k^{(i)}}$

do "death" move

else if $u \le b_{k^{(i)}} + d_{k^{(i)}} + s_{k^{(i)}}$

do "split" move

else if $u \le b_{k^{(i)}} + d_{k^{(i)}} + s_{k^{(i)}} + m_{k^{(i)}}$

do "merge" move

else

update the RBF centres

c. Perform an MH step with the annealed acceptance ratio.

d. i = i + 1

* 几种 Move 的具体实现可参考代码文件或 [3]

B. Akaike Information Criterion

赤池信息量准则 (Akaike information criterion, AIC)[1] 是评估统计模型的复杂度和衡量统计模型“拟合”资料之优良性的一种标准。AIC 准则建立在信息熵的概念基础上，目标在于选择可以最好地解释数据但包含最少自由参数的模型。

在一般情况下，AIC 可以表示为

$$AIC = 2k - 2\ln(L)$$

其中，k 是参数的数量，L 是似然函数。

可以看出，增加自有参数的数量可以提高拟合的优良性，而 AIC 鼓励拟合的优良性但尽量避免出现过拟合的情况。所以优先考虑的模型应该是 AIC 值小的那个。

C. Orthogonal Least Squares Learning

正交最小二乘 (Orthogonal Least Squares, OLS) 算法 [10] 最早在 20 世纪 80 年代后期提出，主要用于非线性系统建模。后来 Chen 又在 [4] 中提出将 OLS 方法应用于 RBF 网络模型的选择问题。为了最小化均方误差，OLS 算法逐项寻求回归模型，力求每一项都能够最小化回归残差。因为每一个回归项都会影响以后回归项的选择，所以每一项都会影响整个模型的性能。而 OLS 方法是一种贪心算法，只寻求当前步骤最优解，而忽略它对下一步的影响，即每步并非全局最优。

下面我将对一维输出的 OLS 算法进行描述，对于多维输出可以将下述算法进行平凡地推广。对于给定输入 x 时，网络的输出为

$$y = P\Theta + E$$

其中，

$y = [y(1), y(2), ..., y(N)]$

$P = [p_1, p_2, ..., p_M], p_1 = [p_i(1), ..., p_i(N)]^T$,

$p_i(t) = p_i(x(t)), 1 \le i \le M$

$\Theta = [\theta_1, ..., \theta_M]^T$

$E = [\epsilon(1), ..., \epsilon(N)]^T$

M 是 RBF 网络中隐层的维度，即 RBF 核函数的个数，N 是输入的样本数。

上述的回归矩阵 P 可被分解为

$$P = WA$$

其中 A 为 $M \times M$ 的上三角阵，且对角元都为 1

$$A = \begin{bmatrix} 1 & \alpha_{12} & \alpha_{13} & \dots & \alpha_{1M} \\ 0 & 1 & \alpha_{23} & \dots & \alpha_{2M} \\ 0 & 0 & 1 & \dots & \alpha_{3M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & 1 \end{bmatrix}$$

W 为 $N \times M$ 的矩阵，且其列向量是彼此正交的。则进行矩阵分解后，y 的表达式变为

$$y = Wg + E$$

定义误差减小率为

$$[err]_i = g_i^2 w_i^T w_i / (y^T y)$$

则训练过程如 Algorithm 3 所示。

Algorithm 3 Orthogonal Least Squares

At the first step, for $1 \le i \le M$

$w_1^{(}i) = p_i$

$g_1^{(}i) = (w_1^{(i)})^T y / ((w_1^{(i)})^T w_1^{(i)})$

$[err]_1^{(i)} = (g_1^{(i)})^2 (w_1^{(i)})^T w_1^{(i)} / (y^T y)$

Find

$[err]_1^{(i_1)} = \max\{[err]_1^{(i)}, 1 \le i \le M\}$

and select

$w_1 = w_1^{(i_1)} = p_{i_1}$

At the kth step where $k \ge 2$**, for** $1 \le i \le M, i \ne i_1, ..., i \ne i_{k-1}$**,**

$\alpha_{jk}^{(i)} = w_j^T p_i / (w_j^T w_j), 1 \le j < k$

$w_k^{(i)} = p_i - \sum_{j=1}^{k-1} \alpha_{jk}^{(i)} w_j$

$g_k^{(}i) = (w_k^{(i)})^T y / ((w_k^{(i)})^T w_k^{(i)})$

$[err]_k^{(i)} = (g_k^{(i)})^2 (w_k^{(i)})^T w_k^{(i)} / (y^T y)$

Find

$[err]_k^{(i_k)} = \max\{[err]_k^{(i)}, 1 \le i \le M, i \ne i_1, ..., i \ne i_{k-1}\}$

and select

$w_k = w_k^{(i_k)} = p_{i_k} - \sum_{j=1}^{k-1} \alpha_{jk}^{(i)} w_j$

The procedure is terminated at the $M_s th$ **step**

$1 - \sum_{j=1}^{M_s} [err]_j < \rho$

V. Experiment

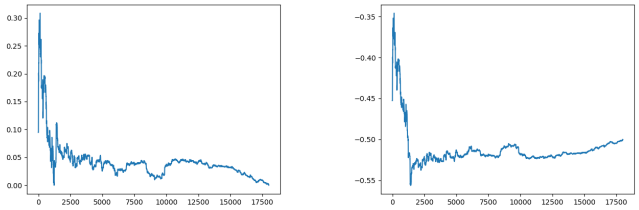
A. Metropolis-Hastings Simulation

使用 MH 算法对下述二维高斯分布进行随机采样，并利用采样结果估计二维高斯分布的相关系数 ρ

$$\mathcal{N}\left\{ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} | \begin{pmatrix} 5 \\ 10 \end{pmatrix}, \begin{pmatrix} 1 & -1 \\ -1 & 4 \end{pmatrix} \right\}$$

由于算法开始运行时采样并不准确，因此舍弃前面的 200 个采样点，使用后面的采样点进行估计。设置最大迭代次数为 50000 次。对不同的建议分布选取绘制其相对误差和相关系数的变化曲线。

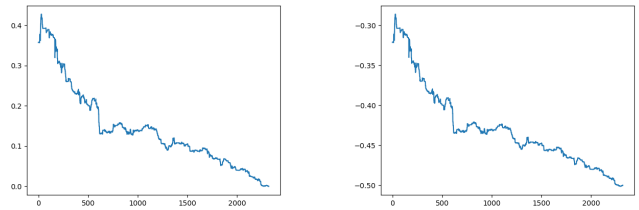
1. 取建议分布的协方差矩阵与已知的二维高斯分布相同，均值为满足二维高斯分布的随机向量，得到 loss 和 ρ 的变化曲线如 Figure 2 所示。



(a) *loss* (b) ρ

Figure 2: Metropolis-Hastings Simulation(Σ)

2. 取建议分布的均值与 1 中选取方式相同，协方差矩阵为原矩阵的 2 倍，得到 loss 和 ρ 的变化曲线如 Figure 3 所示。

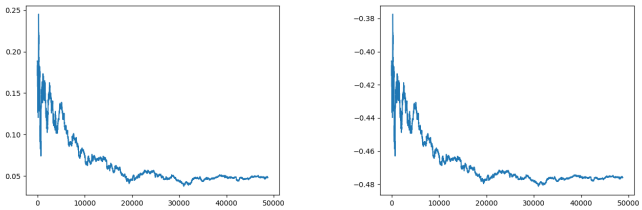


(a) *loss* (b) ρ

Figure 3: Metropolis-Hastings Simulation(2Σ)

3. 取建议分布的均值与 1 中选取方式相同，协方差矩

阵为原矩阵的 4 倍，得到 loss 和 ρ 的变化曲线如 Figure 4 所示。



(a) *loss* (b) ρ

Figure 4: Metropolis-Hastings Simulation(4Σ)

将上述三个实验的数据整理成 Tabel 1，可以看出，当建议分布比较接近目标分布时，收敛速度较快，且最后拟合的效果比较好，而如果建议分布与目标分布区别较大时，则不能以很快的速度收敛。

Σ_Q	#iter	ρ	loss
Σ	18846	-0.5000	0.0001
2Σ	3336	-0.5000	0.0000
4Σ	50000	-0.4759	0.0482

Table 1: Metropolis-Hastings Simulation

B. Reversible Jump MCMC Simulation

[7] 中在介绍 RJMCMC 过程时，选择了以 British Coal Mine Disaster 数据进行训练，期望拟合出一个分段常值函数，为验证其可靠性，本次选择使用相同的数据集进行训练。Figure 5 中为数据集的可视化图形。

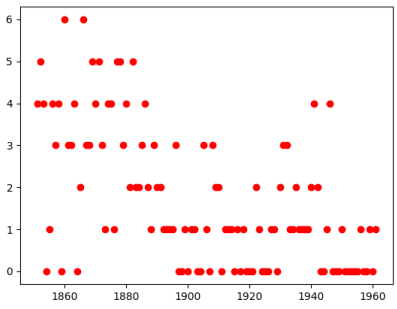


Figure 5: British Coal Mine Disaster Data

我们期望的分布为一个分段常值的函数，在给定区间中，k 为函数的 step 数，对于每个给定的 k，各个 step 的起止位置和高度都是未知的，从而可以转化为变维空间中的模型选择问题，选择使用 RJMCMC 方法进行训练。

本实验中选择使用均方误差 (Mean Square Error, MSE)，得到的 loss 变化曲线如 Figure 6 所示。可以看出，使用 RJMCMC 可以快速地将 loss 降低，最后能够得到一个稳定，且与期待分布十分接近的模型。

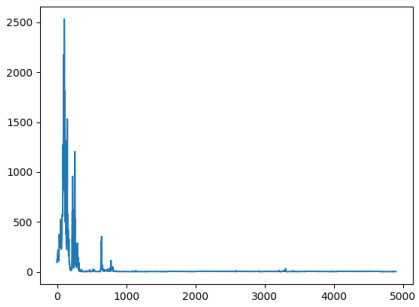


Figure 6: Loss Curve in RJMCMC Simulation

C. Model Selection Techniques for RBFN

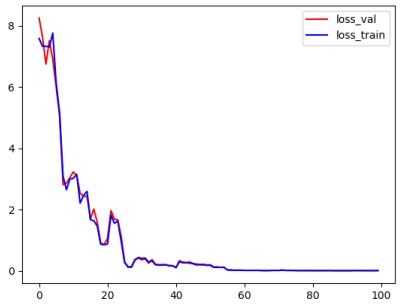
本小节中将分别给出使用 RJMCMC+SA, RJMCMC+AIC, OLS 方法，在给定的两组数据集上的训练结果。为检验训练出的模型在未知数据上的拟合效果，每次训练中随机选取 200 个数据点组成验证集 (Validation Set)，并分别输出模型在训练集和验证集中的 loss 值 (此处以及后文所述实验均选择均方误差作为 loss)。此外，为了减少数据中的奇异点对训练造成的影响，删除其中 y 值最大和最小的各两点，即原数据集中共 1000 个样本，参与训练和验证的样本有 996 个，其中训练样本 796 个，验证样本 200 个。

1. RJMCMC+SA

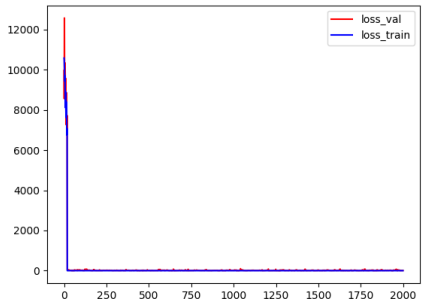
在 RJMCMCSA 的训练过程中，选择 Gauss Kernel 作为 RBF 函数，则根据 IV.A 中所述，我们只需选择出使模型最优的 RBF 函数中心矩阵 Φ 即可。

使用 RJMCMCSA 方法在 data1 和 data2 两个数据集中分别进行训练，设置最大迭代次数为 2000 次。得到 loss 曲线如 Figure 6 中所示。其中 (b) 为 data2 训练过

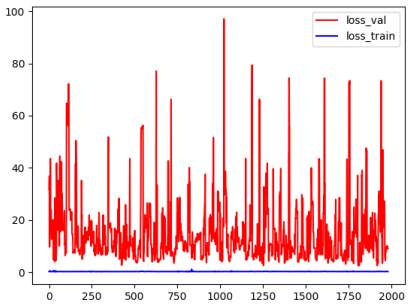
程中的全部 loss 曲线，(c) 为 data2 去掉前 20 个点之后的 loss 曲线。将 RJMCMCSA 方法分别在两个数据集上的时间开销及训练结果整理成 Table 2，其中 loss 为最终模型在全部 996 个样本上的均方误差。



(a) loss for data 1



(b) loss for data 2



(c) loss for data 2(without the first 20 points)

Figure 7: Loss Curve for RJMCMC-SA method

从 loss 曲线中可以看出，使用 RJMCMCSA 方法能够使 loss 迅速地下降到较低的值。不过也可以看出，RJMCMCSA 方法虽然能够得到一个比较优秀的模型选择结果，但是对于较为复杂的数据模型，RJMCMCSA 方法的结果波动性较大。在此次实验中，data1 的训练结果要远好于 data2 的训练结果。

DataSet	k	#iter	time	totalLoss
data1	31	100	39.3025s	0.00469
data2	261	2000	4.1444h	0.26107

Table 2: RJMCMC-SA Experiments

DataSet	k	#iter	time	totalLoss
data1	38	146	141.1047s	0.00366
data2	280	2000	4.6632h	0.23303

Table 3: RJMCMC-AIC Experiments

DataSet	k	#iter	time	totalLoss
data1	42	43	13.6006s	0.00249
data2	159	160	83.9666s	21.26043

Table 4: OLS Experiments

Method	k	totalLoss	trainLoss	valLoss
SA	31	0.00469	0.00476	0.00471
AIC	38	0.00366	0.00368	0.00376
OLS	42	0.00249	0.00244	0.00173

Table 5: Data1 Experiments for All Methods

Method	k	totalLoss	trainLoss	valLoss
SA	261	0.26107	0.23415	177.34351
AIC	280	0.23303	0.22801	9.07621
OLS	159	21.26043	0.27269	0.21333

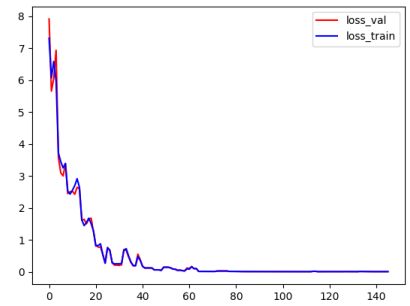
Table 6: Data2 Experiments for All Methods

2. RJMCMC+AIC

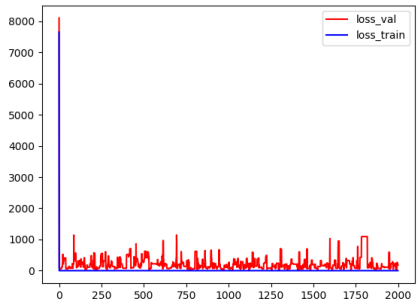
RJMCMC+AIC 的实验过程与 RJMCMCSA 的实验过程基本相同，只不过将其中模拟退火的过程换成了使用 AIC 准则进行模型更新的选择判定。

使用 RJMCMCAIC 方法在 data1 和 data2 两个数据集中分别进行训练，设置最大迭代次数为 2000 次。得

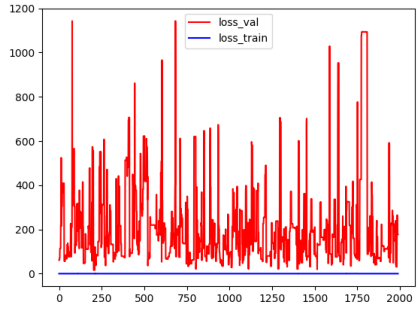
到 loss 曲线如 Figure 7 中所示。其中 (b) 为 data2 训练过程中的全部 loss 曲线，(c) 为 data2 去掉前 10 个点之后的 loss 曲线。将 RJMCMCSA 方法分别在两个数据集上的时间开销及训练结果整理成 Table 3，其中 loss 为最终模型在全部 996 个样本上的均方误差。



(a) loss for data 1



(b) loss for data 2



(c) loss for data 2(without the first 20 points)

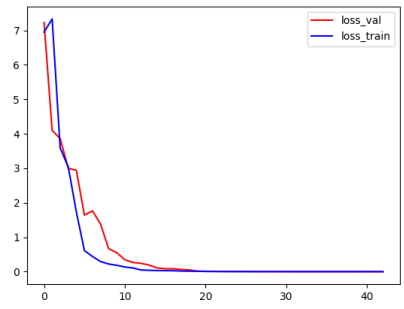
Figure 8: Loss Curve for RJMCMC-AIC method

对比 loss 曲线和数据表格，不难发现，使用 AIC 方法与使用 RJMCMCSA 方法得到的结果比较接近，其 data1 的训练结果和时间开销均要远好于 data2 的训练结果。不过对比 AIC 和 SA 方法的 loss 图象之后，我们可以清楚地看出本次实验中 AIC 的模型的稳定性要稍弱于使用 SA 算法取得的模型。

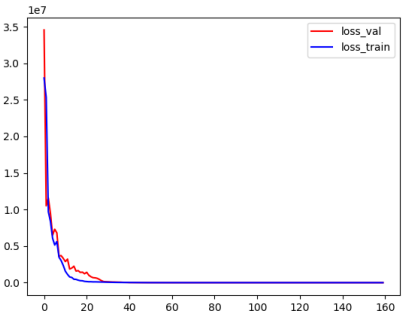
3. OLS

OLS 方法如 IV.C 所述，在本次实验中，分别在 data1 和 data2 的训练过程中取其截止条件中的 ρ 为 $1e-4$ 和 $5e-5$。

使用 OLS 方法在 data1 和 data2 两个数据集中分别进行训练，设置最大迭代次数为 2000 次。得到 loss 曲线如 Figure 7 中所示。将 OLS 方法分别在两个数据集上的时间开销及训练结果整理成 Table 4，其中 loss 为最终模型在全部 996 个样本上的均方误差。



(a) loss for data 1



(b) loss for data 2

Figure 9: Loss Curve for OLS method

从 loss 曲线中可以看出，OLS 方法的稳定性较好，loss 下降之后几乎没有太大的波动。且其训练所需时间要远小于使用 RJMCMC 方法的训练时间。

4. Discussion

将 3 种方法在 data1 和 data2 数据集上的训练结果分别整理成 Table 5 和 Table 6，从表中数据可以看出，在 data1 数据集上三种方法最终都能取得比较好的结果，而在 data2 数据集上三种方法则产生了较大的差异。

- OLS 方法的收敛速度最快，在训练集和验证集上的 loss 比较接近，没有出现明显的过拟合情况，不过最后在全体样本上的测试结果不如 SA 和 AIC 与 RJMCMC 结合的方法。

- 使用 RJMCMC 方法最后能取得一个优于 OLS 方法的模型，但容易出现比较明显的过拟合情况。训练集和验证集上的结果差异较大。在尝试了几次之后发现并不能将过拟合的情况消除。

- 当数据模型比较复杂时，RJMCMC 方法的稳定性不如 OLS 方法

VI. Conclusion

本文从原理和实际测试结果出发，详细分析了 RJMCMC+SA/AIC, OLS 等几种重要的模型选择方法，从而对 RBF 网络模型自动选择技术给出了综合性地说明。几种方法各有利弊，在不同的应用场景中使用不同的选择方法可以取得更好的效果。

Acknowledgement

在这次的课程设计中，张静阳、全雨晗、李一卓等几位同学与我进行了多次深入的讨论，这让我对相关知识有了更细致的了解，在遇到不解的问题时他们也给予我许多参考意见。感谢他们的无私帮助。

当然也要感谢欧智坚老师以及两位助教的辛苦付出，这次作业有一定难度，不过在完成之后也能让我们有比较大的收获，感谢您们的悉心教导。

References

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, Dec 1974.

[2] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[3] Christophe Andrieu, Nando de Freitas, and Arnaud Doucet. Reversible jump MCMC simulated annealing for neural networks. *CoRR*, abs/1301.3833, 2013.

[4] S. Chen, C. F. N. Cowan, and P. M. Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2(2):302–309, Mar 1991.

[5] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[6] Wikipedia. Metropolis–Hastings algorithm — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm`, 2017.

[7] Peter J. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[8] Christophe Andrieu, Nando de Freitas, and Arnaud Doucet. Robust full bayesian learning for neural networks, 1999.

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, May 1983.

[10] S. Chen, S. A. Billings, and W. Luo. Orthogonal least squares methods and their application to non-linear system identification. Address: London, 1989.