汽车四轮转向控制的参数辨识

张所鑫

摘 要 在汽车工程领域，转向系统的数学建模和参数辨识对于确保系统性能和延长其使用寿命至关重要。传统的参数设计方法在精度和实时性方面存在局限性，无法满足现代汽车技术对动态性能和安全性的高标准。本研究首先通过对汽车四轮转向系统的机理分析，构建了一个详细的数学模型。随后，本研究采用 MATLAB 软件作为仿真平台，对比分析了增广最小二乘法（ALS）在递推最小二乘法（RLS）和极大似然法（MLE）等传统参数辨识方法中的优越性，研究结果表明，在已知噪声分布的前提下，即使在样本点较少的情况下，增广最小二乘法也能显著提高参数辨识的效果，验证了其在汽车转向系统参数辨识中的优越性。该研究为高精度和高实时性的汽车转向系统参数辨识提供了一种新的有效方法论。

关键词 车辆工程; 四轮转向控制; 增广最小二乘法; 参数识别

中图法分类号 TP DOI 号: 10.12345/j.aas.c2023100

Parameter Identification for Four-Wheel Steering Control of Automobiles

Suoxin Zhang

Abstract In the field of automotive engineering, the mathematical modeling and parameter identification of steering systems are crucial for ensuring system performance and extending service life. Traditional parameter design methods are limited in precision and real-time capabilities, failing to meet the high standards for dynamic performance and safety required by modern automotive technology. This study begins with a mechanistic analysis of the automobile's four-wheel steering system to construct a detailed mathematical model. Subsequently, using MATLAB software as a simulation platform, the study compares and analyzes the superiority of the Augmented Least Squares (ALS) method over traditional parameter identification methods such as Recursive Least Squares (RLS) and Maximum Likelihood Estimation (MLE). The findings indicate that, given a known noise distribution and even with a limited number of samples, the Augmented Least Squares method significantly improves the accuracy of parameter identification, demonstrating its superiority in the parameter identification of automotive steering systems. This research provides a new and effective methodology for high-precision and real-time parameter identification in automotive steering systems.

Keywords Vehicle Engineering; Four-Wheel Steering Control; Augmented Least Squares; Parameter Identification

中国地质大学（武汉）自动化学院 231215 班

1 研究汽车四轮转向控制的重要性

汽车转向系统在现代汽车工程中占据着举足轻重的地位，对驾驶操控性、行车安全性和驾驶舒适性具有决定性影响。卓越的转向系统能够使驾驶员更轻松地掌控车辆，提供卓越的行驶稳定性，降低驾驶员的疲劳感，从而显著提高行车安全。然而，传统的设计方法常常无法提供足够精确的参数设定，且无法实时监测和调整参数。

在现代汽车工程领域，面临着日益复杂的挑战。车辆需要适应各种驾驶情境，同时还需要适应不断变化的道路条件和车辆状态。这些要求使得传统设计方法无法满足，因此需要更智能、更自适应的解决方案。

通过研究主流的参数辨识方法，并选择其中最有效的方法，实现实时监测和调整参数，有望提供更高精度的参数估计，满足不同驾驶条件下的需求。这样不仅能提高汽车的操控性和安全性，还可以为汽车工程领域提供创新的解决方案。对该问题的研究具有重要性，它将为汽车工程领域提供新的视角和方法，推动该领域的技术进步和发展[1]。

2 简化的汽车四轮转向模型

在四轮转向分析中，通常情况下可以把汽车简化为一个二自由度的两轮车模型，如图一所示。二自由度模型忽略了悬架的作用，将整车简化为两轮，认为轮胎侧偏特性为线性（车辆的侧向加速度限制在 0.4g 以下），并且忽略纵向的驱动或阻力，即纵向车速不变，车辆只有沿 y 轴的侧向运动和绕质心的横摆运动[2]。

模型的运动微分方程可以表示为式(1):

$$\begin{cases} Mu(r+\beta) = F_{y1}\cos\theta_f + F_{y2}\cos\theta_r \\ I_z r = F_{y1}L_f\cos\theta_f - F_{y2}L_r\cos\theta_r \end{cases} \tag{1}$$

其中，M 为整车质量；V 为车速；u 为沿 x 方向的前进速度；v 为沿 y 方向的侧向速度；β 为质心处的侧偏角，$\beta = \frac{v}{u}$；r 为横摆角速度；I_z 为绕质心的横摆转动惯量；δ_f 和 δ_r 分别为前后轮转角；L_f 和 L_r 分别为质心至前后轴的距离；F_{y1} 和 F_{y2} 分别为前后轮侧偏力。

横摆角速度 r 与前后轮转角、质心处的侧偏角 β 与前后轮转角关系如下：

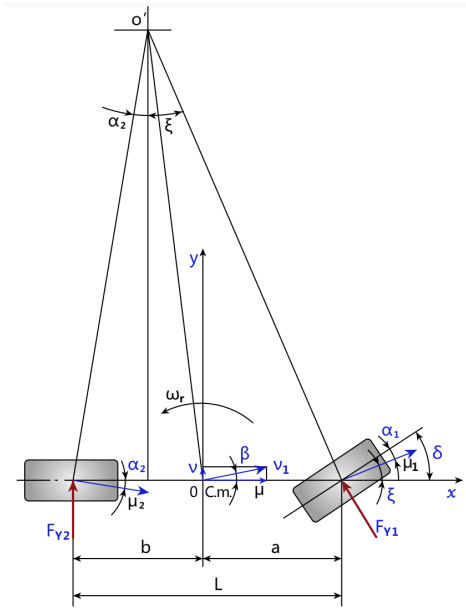


图 一 二自由度四轮转向车辆模型

转角输入-横摆角速度输出的传递函数关系：

$$r(s) = \frac{a_1 s + a_0}{ms^2 + hs + f}\delta_f + \frac{b_1 s + b_0}{ms^2 + hs + f}\delta_r \tag{2}$$

转角输入-执行侧偏角输出的传递函数关系：

$$\beta(s) = \frac{c_1 s + c_0}{m's^2 + hs + f}\delta_f + \frac{d_1 s + d_0}{m'^2 s^2 + hs + f}\delta_r \tag{3}$$

采用横摆角速度反馈，必须找到一个比较量，本文采用稳态横摆角速度作为比较量，并且用稳态角速度增益（转向灵敏度）来表示，即

$$\frac{\partial r}{\partial \delta_f} = \frac{u}{(1+ku^2)L} \tag{4}$$

其中,L 为车辆轴距，K 为车辆稳定因子，i 为前后轮转向比。

3 四轮转向控制原理

当给出一个前轮转角阶跃输入后，在忽略后轮转角的情况下，得到相应的横摆角速度响应，然后和稳态横摆角速度相比较，得出一个需要调整的值；该值经过一定的关系后，求出当前需要的后轮横摆角。整个过程动态进行，后轮根据需要不断接近最优值。该控制调节过程如图二所示：

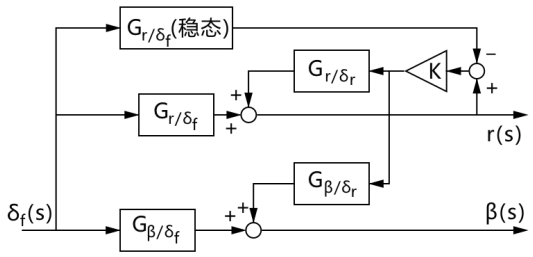


图 二　基于横摆角速度反馈的四轮转向控制原理

本文所选用的汽车模型参数见表一:

表 1　汽车模型参数

变量	值	单位	变量	值	单位
M	1889	Kg	I_z	5523	kg·m^2
L_f	1.5	m	L_r	1.8	m
C_f	-38875	N/rad	C_r	-39566	N/rad

在车辆低速行驶时，各传递函数如下:

$$G_{r/\delta_f}(s)=\frac{-10.7s+8.2}{s^2+2.5s+3.3}$$

$$G_{r/\delta_r}(s)=\frac{-12.4s-14.75}{s^2+2.5s+3.3}$$

$$G_{\beta/\delta_f}(s)=\frac{0.6s-9.8}{s^2+2.5s+3.3}$$

$$G_{\beta/\delta_r}(s)=\frac{0.6s+13.1}{s^2+2.5s+3.3}$$

稳态横摆角速度增益:

$$\frac{r}{G_f}=4.48$$

4　RLS、ALS、MLE 的原理及异同点

4.1　递推最小二乘法 (RLS)

参数递推估计是指对被辨识的系统每取得一次新的测量数据后就在前一次估计结果的基础上，利用新引入的测量数据对前一次估计的结果进行修正，从而递推地得出新的参数估值。这样，随着新测量数据的引入，一次接一次地进行参数估计, 直到估计值达到满意的精确程度为止。最小二乘递推算法的基本思想可以概括为:

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\delta \tag{5}$$

其中, $\hat{\theta}$ 为当前估计值, $\hat{\theta}(k-1)$ 为上次估计值,δ 为修正项。

即当前估计值 $\hat{\theta}$ 是在旧的估计值 $\hat{\theta}(k-1)$ 的基础上，利用新的观测数据对旧的估计值进行修正而成的。

4.2　增广最小二乘法 (ALS)

当噪声均值为 0 时，最小二乘参数估计算法为无偏估计；当噪声的均值不为 0 时, 最小二乘参数估计算法为有偏估计。为了解决最小二乘参数估计的有偏性，将噪声模型的辨识同时考虑进去，这种方法被称为增广最小二乘法。该算法可以看成是对一般最小二乘参数辨识算法的简单推广或扩充，因此又称为扩充最小二乘算法。

4.3　极大似然法 (MLE)

极大似然估计（Maximum Likelihood Estimation, MLE）则是一种在参数估计中寻找能使得观测数据出现概率最大化的参数值的方法。在应用 MLE 时，首先需要定义似然函数，该函数是关于参数的函数，表示在给定参数下观测数据出现的概率。然后，通过优化方法找到最大化似然函数的参数值，这些参数值即为所求的参数估计值。

当噪声均值不为零时，可以通过将噪声的统计特性包含到似然函数中, 从而进行无偏的参数估计。这种方法可以有效地处理数据中的系统偏差，并提供更为准确的估计结果。

4.4　异同点

4.4.1　相同点

(1) **参数估计:** 它们都用于从数据中估计模型参数。

(2) **目标最小化:** 三者都旨在最小化某种形式的误差函数或目标函数。

(3) **迭代过程:** 都采用迭代方法来逐步改进参数估计。

(4) **数据驱动:** 参数的估计是基于实际观测数据进行的。

(5) **模型拟合:** 它们的目标是改善模型对数据的拟合度。

4.4.2 不同点

(1) **更新机制：** RLS 采用递归方式实时更新参数，适合在线处理和动态系统。而 ALS 和 MLE 通常是批处理方法。

(2) **噪声和偏差处理：** ALS 专门处理噪声中的非零均值和系统偏差问题，RLS 通常假设噪声具有零均值或已知统计特性，MLE 可以适应多种噪声分布，但需要先验知识确定合适的概率模型。

(3) **理论基础：** RLS 基于最小化误差平方和的原则，ALS 在最小二乘的基础上增加了噪声模型，MLE 基于统计概率，通过最大化似然函数来估计参数。

(4) **计算复杂度：** RLS 的递归特性通常提供了较低的计算复杂度，ALS 可能因为噪声模型的增加而有更高的计算需求，MLE 的计算复杂度取决于似然函数的形式，可能需要复杂的数值方法解决。

(5) **适用条件：** RLS 适用于需要快速适应新数据的场景，ALS 适用于存在明显偏差或噪声特性已知的情况，MLE 适用于模型和噪声特性可以明确定义的情况。

4.5 对应算法及程序框图

4.5.1 递推最小二乘法 (RLS)

递推最小二乘法是一种动态的数据处理方法，它用于在线更新参数估计，使之能够适应时间变化的数据流，其核心思想是最小化所有已接收数据的加权平方误差和。

1. 计算增益向量 $\mathbf{K}_t$，这个向量决定了新数据点如何影响参数的更新。

$$\mathbf{K}_t = \frac{\mathbf{P}_{t-1}\mathbf{x}_t}{\lambda + \mathbf{x}_t^\top \mathbf{P}_{t-1}\mathbf{x}_t},$$

2. 更新参数估计 $\hat{\boldsymbol{\theta}}_t$，这个估计利用增益向量和新数据点来修正。

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} + \boldsymbol{K}_t(y_t - \boldsymbol{x}_t^\top \hat{\boldsymbol{\theta}}_{t-1}),$$

3. 更新误差协方差矩阵 $\boldsymbol{P}_t$，这个矩阵衡量了当前估计的不确定性。

$$\boldsymbol{P}_t = \frac{1}{\lambda}(\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{x}_t^\top)\boldsymbol{P}_{t-1},$$

为了方便理解，这里列出递推最小二乘法的伪代码如下：

Algorithm 1 递推最小二乘法（RLS）

procedure RLS($\theta(0), P(0), x, y, \lambda, N$){初始化参数估计和误差协方差矩阵}
  for $k \leftarrow 1$ **to** N **do**{对所有观测进行迭代}
    $e(k) \leftarrow y(k) - \theta^T(k-1)x(k)$｛计算时刻 k 的预测误差｝
    $K(k) \leftarrow P(k-1)x(k)[\lambda + x^T(k)P(k-1)x(k)]^{-1}$｛计算时刻 k 的卡尔曼增益｝
    $P(k) \leftarrow \frac{1}{\lambda}[P(k-1) - K(k)x^T(k)P(k-1)]$｛更新时刻 k 的误差协方差矩阵｝
    $\theta(k) \leftarrow \theta(k-1) + K(k)e(k)$｛更新时刻 k 的参数估计｝
  end for
  return $\theta(N)$｛返回最终的参数估计｝
end procedure=0

对应程序框图如图三：

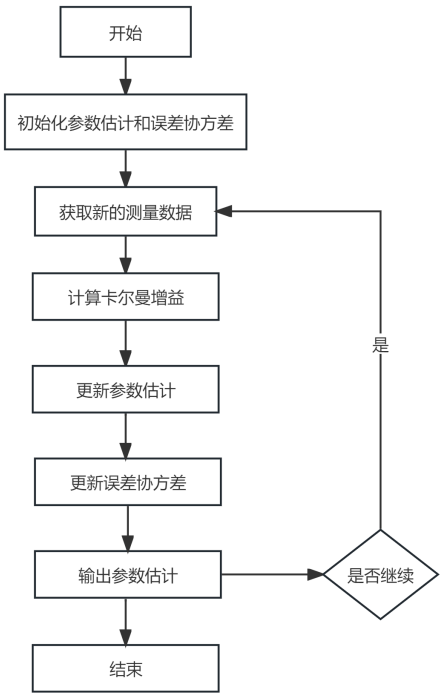


图 三 递推最小二乘法的程序框图

4.5.2 增广最小二乘法 (ALS)

增广最小二乘法在传统最小二乘法的基础上进行了改进。传统的最小二乘法在处理数据较少或模型过于复杂的情况下容易产生过拟合问题，即模型对训练数据的预测性能很好，但对新的、未知的数据预测性能却很差。增广最小二乘法通过引入一个正则化项，可以有效地控制模型的复杂度，提高模型对新数据的预测能力。在增广最小二乘法中，参数估计的目标不仅是最小化预测误差，还包括对参数大小的惩罚，以防止其数值过大，从而避免过拟合。

具体来说，增广最小二乘法通过求解下列优化问题来估计模型参数 $\hat{\boldsymbol{\theta}}$：

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left\{ \sum_{t=1}^{T} (y_t - \boldsymbol{x}_t^{\top}\boldsymbol{\theta})^2 + \lambda \|\boldsymbol{\theta}\|^2 \right\} \tag{6}$$

其中，y_t 是第 t 个观测的目标值，$\boldsymbol{x}_t$ 是相应的特征向量，$\boldsymbol{\theta}$ 是模型参数向量，而 λ 是正则化系数，它控制着正则化项的权重。在这个公式中，第一项是传统的最小二乘误差项，它衡量了模型预测值与实际观测值之间的差异；第二项是正则化项，通常采用参数向量的 L2 范数，即参数向量各元素的平方和。

为了方便理解，这里列出增广二乘法的伪代码如下：

Algorithm 2 增广最小二乘法（ALS）

procedure ALS($\theta(0), x, y, n, N$){初始化参数估计}
  for $k \leftarrow 1$ **to** N **do**{对所有观测进行迭代}
    $e(k) \leftarrow y(k) - \theta^T(k-1)x(k)$｛计算时刻 k 的预测误差｝
    $x_{aug}(k) \leftarrow [x^T(k), n^T(k)]^T$｛构造时刻 k 的扩充输入向量｝
    $S(k) \leftarrow \sum_{i=1}^{k} x_{aug}(i)x_{aug}^T(i)$
    $C(k) \leftarrow \sum_{i=1}^{k} x_{aug}(i)y(i)$
    $\theta(k) \leftarrow S(k)^{-1}C(k)$｛利用扩充向量更新参数估计｝
  end for
  return $\theta(N)$｛返回最终的参数估计｝
end procedure=0

对应程序框图如图四：

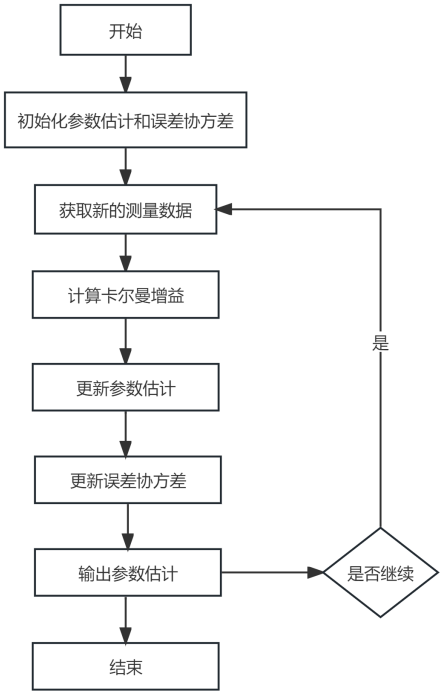


图 四 增广最小二乘法的程序框图

4.5.3 极大似然估计法 (MLE)

极大似然估计是一种基于概率模型的参数估计方法。对于给定的样本集合，目标是找到模型参数的值，使得这些样本出现的概率最大。首先定义似然函数，它是样本集合在模型参数下的联合概率密度函数。然后，通过最大化似然函数来求解模型参数。在实际应用中，通常对似然函数取对数，得到对数似然函数，以简化计算，因为对数函数是单调递增的，不会改变最大化问题的解。当参数是单一变量时，极大似然估计量是以下微分方程的解：

$$\frac{d}{d\theta} \log L(\theta; Y) = 0.$$

当参数是多变量时，参数向量 θ 的极大似然估计量是以下梯度方程的解：

$$\nabla \log L(\theta; Y) = \mathbf{0}.$$

以正态分布为例，假设样本集合 $Y = \{y_1, y_2, \ldots, y_n\}$ 服从均值为 μ，方差为 σ^2 的

正态分布。似然函数和对数似然函数分别为:

$$L(\mu,\sigma^2;Y)=\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2\right),$$

$$\log L(\mu,\sigma^2;Y)=-\frac{n}{2}\log(2\pi)-\frac{n}{2}\log(\sigma^2)-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2.$$

对 μ 和 σ^2 分别求导，并令导数为零，可得到一组方程:

$$\frac{\partial}{\partial\mu}\log L(\mu,\sigma^2;Y)=\frac{1}{\sigma^2}\sum_{i=1}^{n}(y_i-\mu)=0,$$

$$\frac{\partial}{\partial\sigma^2}\log L(\mu,\sigma^2;Y)=-\frac{n}{2\sigma^2}+\frac{1}{2(\sigma^2)^2}\sum_{i=1}^{n}(y_i-\mu)^2=0.$$

解这组方程，可以得到参数的极大似然估计值:

$$\hat{\mu}_{MLE}=\frac{1}{n}\sum_{i=1}^{n}y_i,$$

$$\hat{\sigma}^2_{MLE}=\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{\mu}_{MLE})^2.$$

这些估计值在样本数量趋于无限时，会趋近于参数的真实值。对数似然函数在参数空间中通常是凸的，这意味着得到的解通常是全局最优解。在正态分布的例子中，当 μ 和 σ^2 分别趋向无穷大时，似然函数趋向于零，从而确保了解的存在性和唯一性。

为了方便理解，这里列出极大似然估计法的伪代码如下:

Algorithm 3 极大似然估计（MLE）

procedure MLE(y, f, Θ) {使用模型 f 对数据 y 进行参数估计}
  $\hat{\theta} \leftarrow$ initial guess {初始参数猜测}
  $L \leftarrow$ calculateLikelihood($y, f, \hat{\theta}$) {计算初始似然}
  while not converged **do**
    $\hat{\theta}, L \leftarrow$ updateEstimate($y, f, \hat{\theta}$) {更新估计}
    if not L improved **then**
      break {若似然未改善则停止}
    end if
  end while
  return $\hat{\theta}$ {返回估计参数}
end procedure=0

对应程序框图如图五所示。

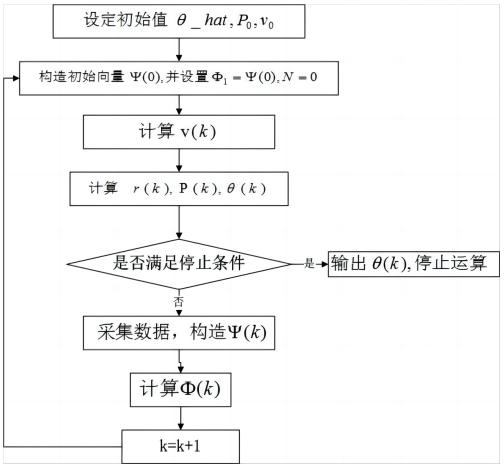


图 五 极大似然估计法的程序框图

5 对汽车转向系统的辨识

5.1 输入信号的选择

在系统辨识过程中，如果数学模型的结构被正确选定，那么模型参数的辨识精度将主要取决于系统的输入信号。因此，选择适当的输入信号是确保理想辨识结果的一个关键。理论分析显示，使用白噪声作为辨识输入信号能够提供较好的辨识效果。然而，在工程实践中，利用白噪声作为输入信号往往是不可行的，因为真实的工业设备无法按照白噪声的特性进行操作。因此，通常采用与白噪声具有相近特性的 M 序列信号来作为一种可行的替代方案[3]。

5.2 辨识对象及参数的选择

在研究的单输入-单输出（SISO）系统中，系统的动态可以通过离散时间差分方程组来描述。具体方程如下:

$$\begin{cases}z(k)+a_1z(k-1)+a_2z(k-2)=\\ \qquad b_1u(k-1)+b_2u(k-2)+V(k),\\ V(k)=c_1v(k)+c_2v(k-1)+c_3v(k-2)\end{cases}$$

其中，$z(k)$ 代表在第 k 个时间步的系统输出，$u(k)$ 为系统输入，而 $V(k)$ 则是噪声项，由噪声过程 $v(k)$ 经过噪声模型转换得到。

由于文章篇幅关系，本文只选择传递函数 $G_{\beta/\delta_f}(s)=\frac{0.6s-9.8}{s^2+2.5s+3.3}$ 进行辨识，其他传递函数的

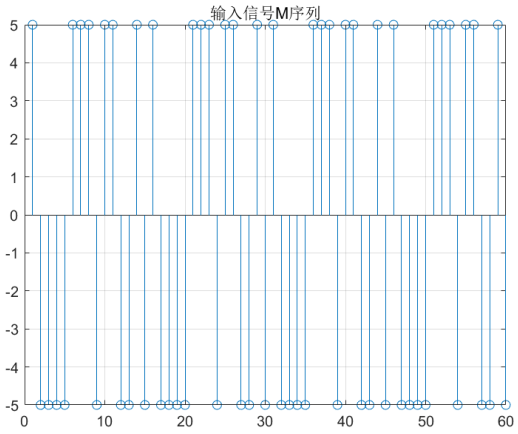
辨识方法一样。通过对该传递函数利用向后差分法进行离散化处理（利用 MATLAB 的 c2d 函数）即可得到 a_1、a_2、b_1、b_2，这里假设 c_1、c_2 和 c_3 分别为 0.1，0.4，0.3，则差分方程为:

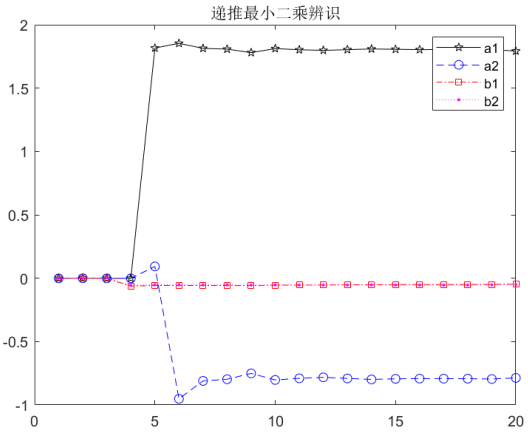
$$\begin{aligned} z(k) =& 1.7503z(k-1) - 0.7794z(k-2) \\ & - 0.0432u(k-1) - 0.0481u(k-2) \\ & + 0.1v(k) + 0.4v(k-1) + 0.3v(k-2) \end{aligned}$$

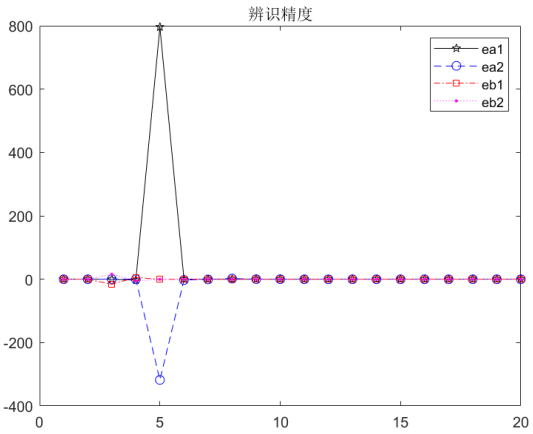
5.3 系统辨识结果

5.3.1 RLS 的辨识结果

输入信号为一个包含 60 个样本点的高斯白噪声序列，该序列的均值（期望值）为 0，方差为 0.1，以模拟实际测量中的噪声环境，辨识结果如下:







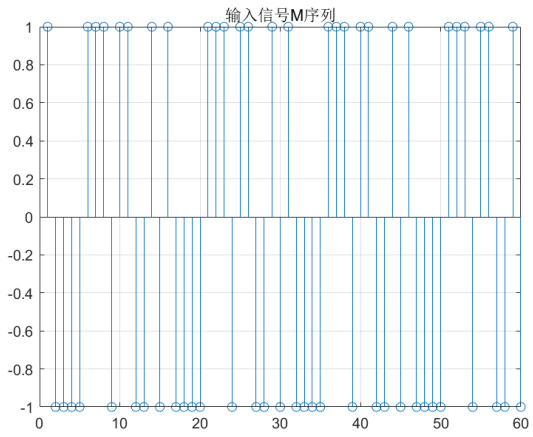
最终迭代步骤所得到的参数估计值被采纳作为系统模型的辨识结果，并与真实值对比得出误差，见表 2:

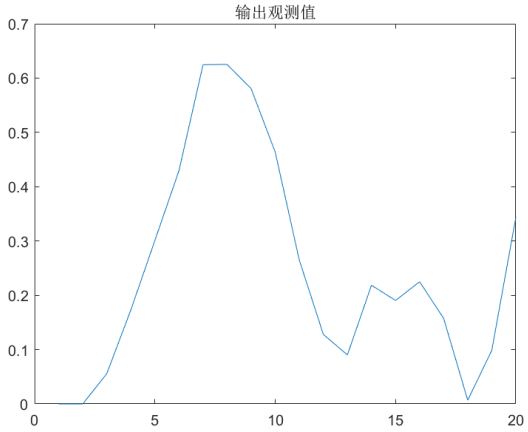
表 2 递推最小二乘法的参数估计

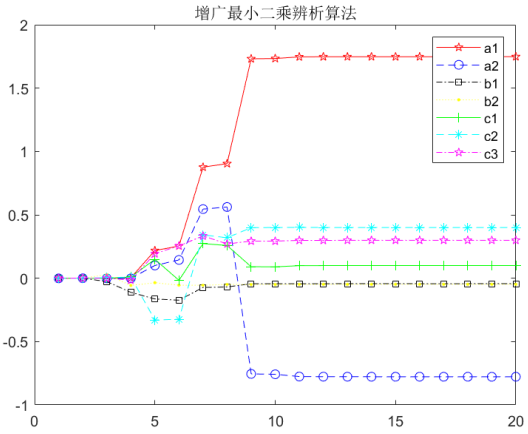
参数	a_1	a_2	b_1	b_2
真值	1.75	-0.7794	-0.0432	-0.0481
估计值	1.7883	-0.7951	-0.0482	-0.0512
误差	0.0219	0.0201	0.1157	0.0644

5.3.2 ALS 的辨识结果

输入信号为一个包含 60 个样本点的高斯白噪声序列，该序列的均值（期望值）为 0，方差为 0.1，以模拟实际测量中的噪声环境，辨识结果如下。







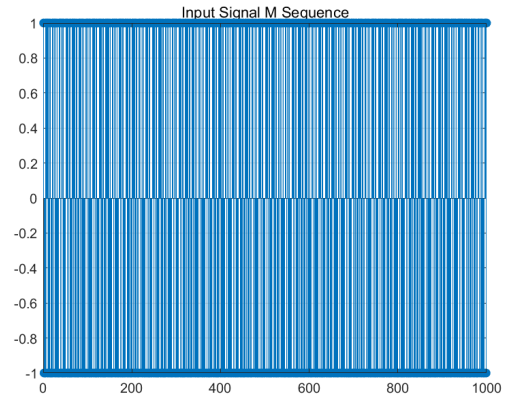
最终迭代步骤所得到的参数估计值被采纳作为系统模型的辨识结果，并与真实值对比得出误差，见表 3：

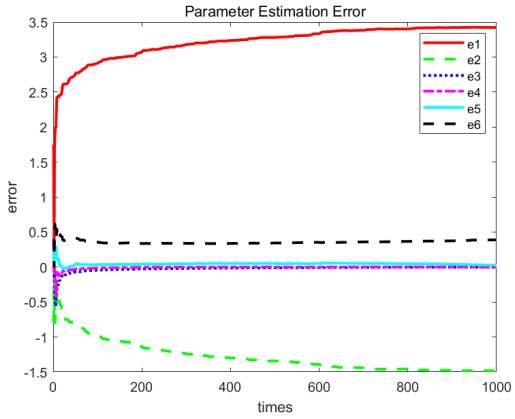
表 3 增广最小二乘法的参数估计

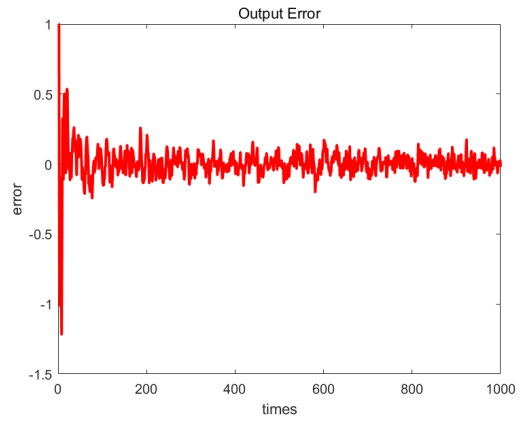
参数	a1	a2	b1	b2	c1	c2	c3
真值	1.7503	-0.7794	-0.0432	-0.0481	0.1	0.4	0.3
估计值	1.7495	-0.7789	-0.0432	-0.0481	0.1	-0.3999	0.2998
误差	0.0008	0.0005	0	0	0	0.0001	0.0002

5.3.3 MLE 的辨识结果

由于极大似然法在输入信号样本量小的情况下（60 个样本点）辨识效果非常差，所以提高样本量至 1000，重新辨识得到结果如下：







最终迭代步骤所得到的参数估计值被采纳作为系统模型的辨识结果，并与真实值对比得出误差，见表 4：

表 4 极大似然法的参数估计

参数	a_1	a_2	b_1	b_2	c_1	c_2
真值	1.75	-0.7794	-0.0432	-0.0481	0.1	0.4
估计值	1.6769	-0.7069	-0.0380	-0.0483	0.0910	0.0466
误差	0.0418	0.0930	0.1204	0.0042	0.0900	0.8835

5.4 结果分析

根据表 1、表 2 和表 3 的结果不难看出，在 a_1 和 a_2 的参数估计中，ALS 的辨识效果最佳，RLS 次之，而 MLE 的估计值与真值之间存在较大偏差，尤其是 a_2 的估计误差高达 0.0930。对于 b_1 和 b_2 的估计，ALS 的结果最为理想，其估计值与真值完全吻合。考虑到 MLE 和 RLS 在 b_1 的辨识结果相近，而在 b_2 的估计上，MLE 明显优于 RLS，因此可以认为在这一点上 MLE 的表现优于 RLS。

在 c_1 和 c_2 的参数估计方面，ALS 的精度依旧是最高的，误差几乎可以忽略不计。而 MLE 在 c_1 的估计误差还在可接受的范围内，但 c_2 的估计误差非常大，这表明 MLE 在处理含有高斯白噪声的数据时可能存在局限性。高斯白噪声的随机性可能会显著影响似然函数的形态，从而对参数估计的精度产生不利影响。

综上所述，在本次辨识中，ALS 的表现优于 MLE 和 RLS，最核心的原因可能与其对噪声的处理能力有关。

具体来说，输入的噪声序列具有 0 均值和 0.1 的方差，这意味着噪声虽然随机但其分布特性是已知的。ALS 通过增广模型，能够在估计过程中考虑到噪声的统计特性，从而在参数估计时减少噪声的影响，这对于样本数量相对较少（60 个样本点）的情况尤为重要。

而 MLE 虽然也基于概率模型，但其在估计参数时对噪声的实际分布假设较为敏感。如果噪声的真实分布与假设模型稍有偏差，MLE 的估计结果就可能产生较大误差。此外，MLE 在寻找参数估计的最优值时可能容易受到局部最优解的影响，尤其是在样本数量不是非常大时。

对于 RLS，它的更新方式是逐步的，依赖于每个新样本来递推更新参数估计。这种方法在样本数量较少时会不够稳定，特别是当噪声水平较高时，每次更新都会引入额外的误差，导致估计结果偏离真实值。

所以，在已知噪声分布特性且样本点较少的情况下，应该优先选择 ALS 来辨识参数。而在样本点较多的情况下，MLE 往往能有更好的效果。

参考文献

[1] 李伟, 王洪民, 唐峥. 基于递推最小二乘法的转向系统参数辨识 [J]. 重庆交通大学学报（自然科学版）,2019,38(8):124-128. DOI:10.3969/j.issn.1674-0696.2019.08.21.

[2] 周淑文, 张思奇, 许晓东. 四轮转向汽车稳定性控制 [J]. 控制工程,2007,14(z1):78-80. DOI:10.3969/j.issn.1671-7848.2007.z1.027.

[3] https://blog.csdn.net/keilzc/article/details/122846510

6 附录

6.1 心得体会

在初次接触到课程报告的要求时，我必须承认自己感到有些手足无措。课堂上，我们专注于理论学习，而没有对实际应用有过多的了解。但经过课程实验，我逐渐领悟到了其中的奥秘。实验中的系统辨识，本质上是对给定的差分方程系数的识别。这启示我，只要能从实际系统中导出差分方程，就能实现对系统的辨识。

幸运的是，在《现代控制理论》课程中，我们有一个选做的作业，即将理论知识应用于具体系统。我当时选择的是汽车转向系统，这为我提供了不少基础知识。基于这个基础，我开始寻找与汽车转向系统相关的论文。当找到参考文献的第一篇论文时，我得到了思路。文中介绍了如何将实际系统转化为可辨识的参数方程——即通过机理建模得到传递函数，再通过离散化（比如使用 MATLAB 的 c2d 函数实现向后差分法）来完成。一旦得到差分方程，接下来的步骤就相对简单了，因为实验中我已经编写了相应的代码，只需根据差分方程对参数进行调整即可。这个过程也让我意识到查找文献的重要性，如果仅凭自己思考，我可能只能完成到机理建模的步骤，因为我还未学会如何将传递函数离散化。在分析结果时，我通过对比其他文献的结论，才得出了自己的观点。这个过程也加深了我对 RLS、ALS、MLE 这三种辨识方法的理解。

此外，这次的课程报告我是用 LaTeX 编写的，一部分原因是担心自己会忘记这项技能。我之前只在数学建模比赛中使用过 LaTeX，所以当老师要求报告格式要达到发表论文的标准，我就觉得这是一个练习的好机会。然而，在使用 LaTeX 的过程中，我遇到了不少问题，如缺少宏包、图形浮动等，我不得不将错误信息逐一复制到浏览器中搜索解决方案。尽管如此，当论文最终完成时，我为自己一周来的努力感到无比的满足和自豪。

在这里我也要感谢万老师这个学期的倾心授课，耐心答疑。之前到老师办公室问问题的时候，得知了老师每次上课都会提前准备好，这样就能在课前 30 分钟到，而这三十分钟，就能用来解答同学们的问题。这让我感触很深，尤其是这学期保研的压力让我几乎喘不过气。我必须每一科都发挥得很好，才能有机会保研。但是，倘若我以前就能向老师这样，什么都提前准备，不管是考试，还是作业，我都赶在前面完成，我应该也不会落到这么紧张的地步吧！总而言之，遇见万老师实属人生一大幸事，万老师的个人魅力，深深影响了我，再次感谢万老师的辛苦付出！

6.2 向后差分离散化传递函数代码

```
%%
% 定义连续时间传递函数
num = [0.6 -9.8]; % 分子系数
den = [1 2.5 3.3]; % 分母系数
G_s = tf(num, den); % 创建传递函数

% 设定采样时间
T = 0.1; % 假设采样周期为0.1秒

% 使用后向差分法进行离散化
G_z = c2d(G_s, T, 'tustin');

% 显示离散时间传递函数
disp(G_z);

% 如果需要，将离散时间传递函数转换为差分方程的系数
[num_d, den_d] = tfdata(G_z, 'v'); % 提取离散时间传递函数的系数
a = -den_d(2:end); % 注意MATLAB返回的分母包含了z^0的系数，我们需要取反和移除它
b = num_d(2:end); % 分子的系数也需要调整，移除z^0的系数

% 输出差分方程的系数
disp('差分方程的系数 a:');
disp(a);
disp('差分方程的系数 b:');
disp(b);
```

6.3 递推最小二乘法代码

```
%% 递推最小二乘法
clear all
clc
randn('seed',100) ;
```

```
v=sqrt(0.1)*rand(1,60);%产生一组60个N（0,0.1）的高斯分布的随机噪声，若N(0,0.5)则写为v=sqrt(0.5)*rand(1,16)
%M序列产生程序
L=60;%M序列的周期
y1=1;y2=1;y3=1;y4=0;%四个移位寄存器的输出初始值
for i=1:L;
x1=xor(y3,y4);
x2=y1;
x3=y2;
x4=y3;
y(i)=y4;
if y(i)>0.5,u(i)=-5;%M序列幅值为5
else u(i)=5;
end
y1=x1;y2=x2;y3=x3;y4=x4;
end
figure(1);
stem(u),grid on
title('输入信号M序列')
%递推最小二乘辨识程序
z(2)=0;z(1)=0;
%观测值由理想输出值加噪声
for k=3:60;%循环变量从3到60
  z(k)=1.75*z(k-1)-0.7794*z(k-2)-0.0432*u(k-1)-0.0481*u(k-2)+0.1*v(k)+0.4*v(k-1)+0.3*v(k-2);%观测值，此处为白噪声
end
c0=[0.001 0.001 0.001 0.001]';
p0=10^3*eye(4,4);
E=0.000000005;%相对误差
c=[c0,zeros(4,59)];%被辨识参数矩阵的初始值及大小
e=zeros(4,60);%相对误差的初始值及大小
lamt=1;
for k=3:60;
    h1=[-z(k-1),-z(k-2),u(k-1),u(k-2)]';
    k1=p0*h1*inv(h1'*p0*h1+1*lamt);%求出k的值
    new=z(k)-h1'*c0;
    c1=c0+k1*new;%求被辨识参数c

    p1=1/lamt*(eye(4)-k1*h1')*p0;
    e1=(c1-c0)./c0;%求参数当前值与上一次的值的差值
    e(:,k)=e1;%把当前相对变化的列向量加入误差矩阵的最后一列
    c(:,k)=c1;%把辨识参数c列向量加入辨识参数矩阵的最后一列
    c0=c1;%新获得的参数作为下一次递推的旧参数
    p0=p1;
    if norm(e1)<=E
       break;%若参数收敛满足要求，中止计算
    end
end
%分离参数
a1=c(1,:)
a2=c(2,:)
b1=c(3,:)
b2=c(4,:)
ea1=e(1,:);ea2=e(2,:);eb1=e(3,:);eb2=e(4,:);

figure(2);
i = 1:60;
plot(i, -a1, '-kp', i, -a2, '--bo', i, b1, '-.rs', i, b2, ':m.');
xlim([0 20])
legend('a1', 'a2', 'b1', 'b2');
title('递推最小二乘辨识');


figure(3);
plot(i, ea1, '-kp', i, ea2, '--bo', i, eb1, '-.rs', i, eb2, ':m.');
xlim([0 20])
legend('ea1', 'ea2', 'eb1', 'eb2');
title('辨识精度');
```

6.4 增广最小二乘法代码

```
%% 增广递推最小二乘辨识法
clear all
clc
% M序列，噪声信号产生及其显示程序
L=60;% 四位移位寄存器产生的M序列的周期
y1=1;y2=1;y3=1;y4=0; %四个移位寄存器的输出初始值
for i=1:L;
    x1=xor(y3,y4);
    x2=y1;
```

```
    x3=y2;
    x4=y3;
    y(i)=y4;
    if y(i)>0.5,u(i)=-1;
    else u(i)=1;
    end
    y1=x1;y2=x2;y3=x3;y4=x4;
end
figure(1)
stem(u),grid on%画出M序列输入信号
title('输入信号M序列')
randn('seed',100)
v=sqrt(0.1)*randn(1,60);%产生一个N (0, 0.1
    ) 的随机噪声
%增广最小二乘辨识
z(2)=0;z(1)=0;
theat0=[0.001 0.001 0.001 0.001 0.001
    0.001 0.001]';
p0=10^4*eye(7,7);%初始状态p0
theat=[theat0,zeros(7,59)];%被辨析参数矩阵
    的初始值及大小
for k=3:60;
    z(k)=1.75*z(k-1)-0.7794*z(k-2)-0.0432*
        u(k-1)-0.0481*u(k-2)+0.1*v(k)-0.4*v
        (k-1)+0.3*v(k-2)
    h1=[-z(k-1),-z(k-2),u(k-1),u(k-2),v(k)
        ,v(k-1),v(k-2)]';
    x=h1'*p0*h1+1;
    x1=inv(x);
    k1=p0*h1*x1;%k
    d1=z(k)-h1'*theat0;
    theat1=theat0+k1*d1;%辨析参数c
    theat0=theat1;%给下次用
    theat(:,k)=theat1;%把辨析参数c列加入辨
        析参数矩阵
    p1=p0-k1*k1'*[h1'*p0*h1+1];%find p(k)
    p0=p1;%给下一次用
end%循环结束
%分离变量
a1=theat(1,:)
a2=theat(2,:)
b1=theat(3,:)
b2=theat(4,:)
c1=theat(5,:)
c2=theat(6,:)
c3=theat(7,:)
i=1:60;
figure(2);
plot(i,z);
xlim([0 20])
title('输出观测值');
figure(3);
plot(i,-a1,'-rp',i,-a2,'--bo',i,b1,'-.ks',
    i,b2,':y.',i,c1,'-g+',i,-c2,'--c*',i,
    c3,'-.mp')%画出各个被辨析参数
legend('a1', 'a2', 'b1', 'b2','c1','c2','
    c3');
xlim([0 20])
title('增广最小二乘辨析算法')%标题
```

6.5 极大似然法代码

```
%% 极大似然参数估计
clear all
close all
% 产生仿真数据
% 产生仿真数据
n = 2;
total = 1000; % 修改为1000以与Newton-
    Raphson法中的序列长度一致
sigma = 0.1; % 噪声变量的均方根
% M 序列作为输入
z1 = 1; z2 = 1; z3 = 1; z4 = 0;
for i = 1:total
    x1 = xor(z3, z4);
    x2 = z1;
    x3 = z2;
    x4 = z3;
    z(i) = z4;
    if z(i) > 0.5
        u(i) = -1;
    else
        u(i) = 1;
    end
    z1 = x1; z2 = x2; z3 = x3; z4 = x4;
end
figure(1);
stem(u, 'filled'), grid on;
title('Input␣Signal␣M␣Sequence');
% 系统输出
y(1) = 0; y(2) = 0;
v = sigma * randn(total, 1); % 噪声
y(1) = 1; y(2) = 0.01;
```

```
for k = 3:total
    y(k) = 1.7503 * y(k - 1) -0.7794 * y(
        k - 2) -0.0432*u(k - 1) -0.0481 *
        u(k - 2) + 0.1*v(k) +0.4 * v(k -
        1) + 0.3 * v(k - 2);
end
%初始化
theta0=0.001* ones(6,1); %参数
e1(1)=1.7503-theta0(1); e2(1)=-0.7794-
    theta0(2); %误差初始化
e3(1)=-0.0432-theta0(3); e4(1)=-0.0481-
    theta0(4);
e5(1)=0.1-theta0(5); e6(1)=0.4-theta0(6);
a_hat(1)=theta0(1); a_hat(2)=theta0(2); %
    参数分离
b_hat(1)=theta0(3); b_hat(2)=theta0(4);
c_hat(1)=theta0(5); c_hat(2)=theta0(6);
P0=eye(6,6); %矩阵 P 初始化
for i=1:n
    yf(i)=0.1;
    uf(i)=0.1;
    vf(i)=0.1;
    fai0(i,1)=-yf(i);
    fai0(n+i,1)=uf(i);
    fai0(2* n+i,1)=vf(i);
end
e(1)=1.0;
e(2)=1.0;
%递推算法
for i=n+1:total
pusai=[-y(i-1);-y(i-2);u(i-1);u(i-2);e(i
    -1);e(i-2)];
C=zeros(n* 3,n* 3);
Q=zeros(3* n,1);
Q(1)=-y(i-1);
Q(n+1)=u(i-1);
Q(2* n+1)=e(i-1);
for j=1:n
    C(1,j)=-c_hat(j);
    C(n+1,n+j)=-c_hat(j);
    C(2* n+1,2* n+j)=-c_hat(j);
if j>1
    C(j,j-1)=1.0;
    C(n+j,n+j-1)=1.0;
    C(2* n+j,2* n+j-1)=1.0
    end
end
fai=C* fai0+Q;
K=P0* fai* inv(fai'* P0* fai+1);
P=[eye(6,6)-K* fai']* P0;
e(i)=y(i)-pusai'* theta0;
theta=theta0+K* e(i);
P0=P;
theta0=theta;
fai0=fai;
    a_hat(1)=theta(1); a_hat(2)=theta(2);
    b_hat(1)=theta(3); b_hat(2)=theta(4);
    c_hat(1)=theta(5); c_hat(2)=theta(6);
    e1(i)=1.7503-a_hat(1); e2(i)=-0.7794-
        a_hat(2);
    e3(i)=-0.0432-b_hat(1); e4(i)=-0.0481-
        b_hat(2);
    e5(i)=0.1-c_hat(1); e6(i)=0.4-c_hat(2)
        ;
end
% 绘制参数估计误差
figure(2);
plot(e1, '-r', 'LineWidth', 2); hold on;
plot(e2, '--g', 'LineWidth', 2); hold on;
plot(e3, ':b', 'LineWidth', 2); hold on;
plot(e4, '-.m', 'LineWidth', 2); hold on;
plot(e5, '-c', 'LineWidth', 2); hold on;
plot(e6, '--k', 'LineWidth', 2);
title('Parameter␣Estimation␣Error');
xlabel('times');
ylabel('error');
legend('e1', 'e2', 'e3', 'e4', 'e5', 'e6')
    ;
hold off;

% 绘制输出误差
figure(3);
plot(e, '-r', 'LineWidth', 2); % 修改为红
    色实线
title('Output␣Error');
xlabel('times');
ylabel('error');
```