统计推断在数模转换中的应用

组号 01 姓名 沈思齐 学号 5130309335 姓名 仇一 学号 5130309333

摘要：运用 Matlab 软件对传感器进行校准定标。在分析比较多项式拟合、插值拟合等不同拟合方式得到的样本数据曲线后，选用光滑样条插值拟合样本。通过遗传算法，对于样本曲线的选点进行优化，降低校准定标的总成本。

关键字：校准定标，插值拟合，遗传算法

1 引言

本课题来源于工业中校准定标问题。

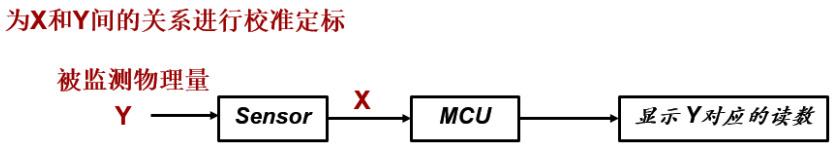


图 1 检测模块工作流程[1]

在工业生产中，我们往往需要通过传感器测量相关的参数，但是传感器使用种种原理将带测量转换为直接可观测的量的过程中往往要经过一系列的非电信号到电信号转换与机械传动，因此被检测物理量与直接测出量之间绝大数时候并不呈线性，当非线性带来的误差不能被接受的时候就需要重新定标。

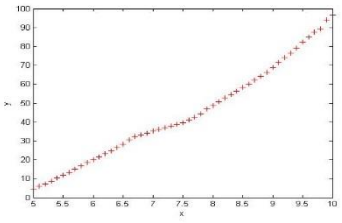


图 2 个体样品特性曲线

然而对于器件一致性差，样本容量大的情况，传统的密集选点法并不能高效地完成校准定标的工作，并且在测量中将付出极大的成本，这就要求我们寻求更为优化的方法完成校准定标的工作。

2 样本分析

本课题共提供了 469 组样本的密集选点测定数据，其中每个样本包含 51 个 X 取值从 5.0 到 10.0 等间距分布的测定点。这里随机的抓取了 3 组样本（76, 122, 161 组），绘制 Y-X 图像如下。

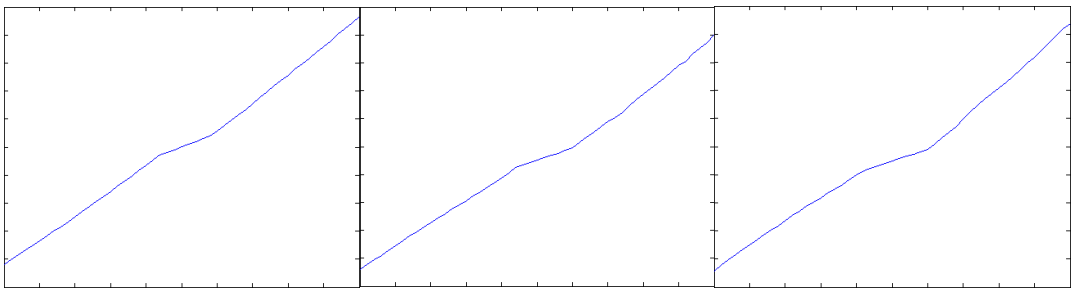


图 3 样本数据 Y-X 曲线

从样本的图像中可以看出：

（1） 各样本曲线均单调递增

（2） 各曲线可大致分为三段，中段与其他两段之间有明显的斜率降低，但各段内部线性关系都较好，不过仍有略微的波动。

（3） 各曲线中段的始末位置不同，且斜率的变化程度也有不同。

总的来说，样本没有跳变并且较为光滑，因此对于后续的操作限制较小。

3 拟合方法选取

正如之前阐述的，本课题的主要目的是优化校准定标工作，而降低定标成本其中一个重要的部分就是降低定标成本。这将通过减少测定的次数以实现。

在减少测量点的过程中，曲线会丢失一部分原有的信息，我们将通过已测点的数据推断估算其他点的数值。不可避免的，在推断中一定会引入误差，而对拟合方法的研究也就是对减小引入误差的研究。

3.1 拟合

严格地说，狭义的拟合只是早期用于离散函数逼近的重要方法。它是指已知某函数的若干离散函数值{f1,f2,…,fn}，通过调整拟合目标函数中若干待定系数(λ1, λ2,…, λ3)，使得该函数与已知离散点的差别最小，通常通过考察 SSE,RMSE,R-square 这些参数评价拟合结果好坏。注意拟合的曲线往往并不经过所有的离散点，当离散点的数量多于参数的数量时则存在矛盾方程，一定有点不在拟合曲线上。常见的拟合目标函数形式有多项式(polynomial)、傅里叶函数(Fourier)、Weibull 分布、指数函数、对数函数、幂函数、高斯函数(Gauss)等等。由于多项式的性质优良，并且拟合速度较快，故多项式拟合较为常用，傅里叶函数则在波的处理中大量使用，后几种拟合则大多只在明确函数关系的情况下选用。

3.2 插值

此外可以使用的还有一些插值方法。插值是指在离散数据的基础上补插连续函数，使得这条连续曲线通过全部给定的离散数据点(注意这点与拟合的区别)，并由这些连续函数确定函数的估计值，常见的插值方法有 Lagrange 插值、Newton 插值、Hermite 插值等，它们均为插值多项式。然而 Runge 在研究多项式插值的时候，发现有的情况下，并非取节点越多多项式就越精确。著名的例子是 f（x）=1/(1+25x^2).它的插值函数在两个端点处发生剧烈的波动，造成较大的误差。[2]究其原因，是舍入误差造成的。因此在上述基础上产生了分段插值、样条插值等通过分区段操作提高精度从而回避了高次插值可能带来的弊端的插值方法。

3.3 拟合方法简单评价

由于拟合方法多种多样，我们事先并不能知道哪一种拟合方法是最优的，因此这里简单地试进行拟合以获取一些直观印象。

3.3.1 多项式拟合

这里我们使用的是整一组的数据进行拟合而不是用了其中一部分，因为我们认为该课题的目标是使得校准定标结果在更多的点上准确乃至整个连续的区段上更准确，需要观察整个区段上的曲线形态，因此观察 50 个数据进行拟合得到的曲线的性质是可行的。

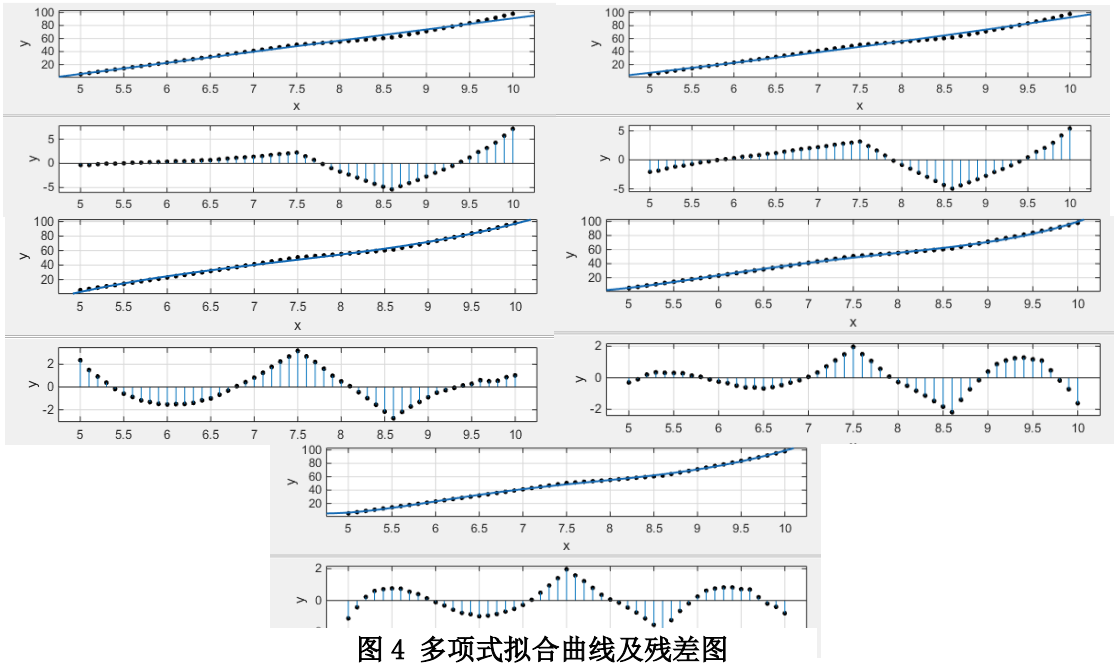


图 4 多项式拟合曲线及残差图

以上五张图依次展示了一至五次多项式拟合的结果以及其残差。从图表中可以看到在数据前半段，一次多项式拟合的表现很好，这也与之前对图像的描述一致。但是很遗憾，五个多项式均不能很好地拟合，都有比较大的残差。更为高次的多项式由于 Runge 现象的可能存在，以及对于最小取点数量的限制，我们不予采用。

3.3.2 其他拟合函数

在实验了指数函数、高斯函数、幂函数等一系列之前提到的拟合目标函数后，我们发现它们的表现都不够令人满意。篇幅所限，不一一贴出数据图。

3.3.3 关于插值方法的评价

这里我们没有对插值方法进行评价，由于选取了全部点，插值方法将会将全部的 50 个点置于函数上，由于我们没有办法知道其余位置的真实值，因此得到的函数的方差将会是零。

在后续实验中，我们使用的是光滑样条插值(smoothing spline)的无参数拟合，它原本主要用于去噪等用途。光滑样条插值可以调节需要的精度，并且在默认情况下也有极好的精度，对于本课题中样本本身没有跳变的曲线较为合适。如下图，仅在 7.5 和 8.5 即中段的始末处有 0.2 左右的残差，其他地方符合得都相当好。

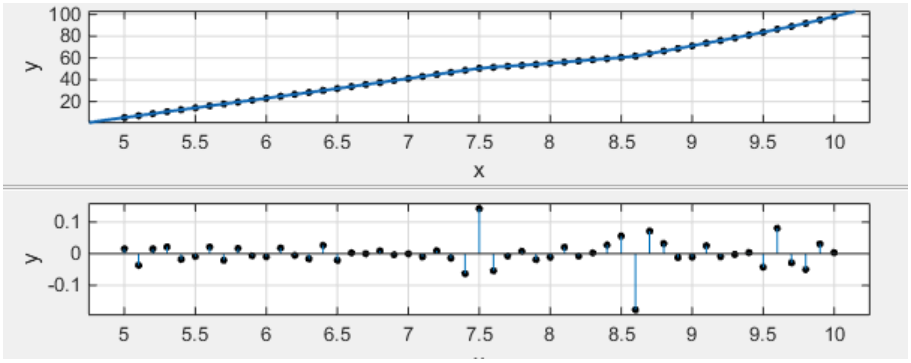


图 5 光滑样条拟合曲线及残差图

$$p\sum_i w_i\left(y_i - s(x_i)\right)^2 + (1-p)\int\left(\frac{d^2s}{dx^2}\right)^2 dx \qquad (1\text{-}1)$$

公式（1-1）为我们选取的光滑样条插值的特征量的表达式，其中 p 为光滑参数(smoothing parameter)，w 为权重。算式左侧表现为与残差平方有关，右侧表现为与曲线光滑程度有关，p 越小则对曲线光滑的要求越高，反之则对曲线与数据点的接近程度要求越高。事实上光滑参数 p=1 时，即为三次样条插值(cubic spline)，p=0 时将给出直线。在本次实验中，考虑到曲线特征，我们使用的是 p 的默认值 0.99，这样一方面使曲线与测定点的符合程度较好，另一方面可以利用遗传算法选取性质好的点，对物理世界中本该光滑的曲线进行拟合，在对于拐点的处理中得到较好的效果。

4 特征点选取

如果说拟合是让已选取的点发挥其最大效用，那么特征点选取则是从整一个样本中选取最具有价值、最能体现整个函数特性的点集，因此它是优化中非常重要的一环。

4.1 特征点选取优化的必要性

最佳特征点的选取并不是一件简单的事情，仅仅$\boldsymbol{C_{51}^{6}}$就达到了 100M 的量级，即使是当今的计算机也无法暴力穷举所有的可能性，更何况这样做耗费的资源也违背了我们进行优化的初衷。因此，如何选取特征点成了课题的核心问题之一。

对于这种备选方案数量巨大的问题，我们称其为 NP-hard 问题(即算法复杂度不能用问题的阶数 n 来表示)。现今解决这类问题的主要方式还是以启发式搜索算法为主，较有名的有遗传算法(Genetic Algorithm),模拟退火算法(Simulated Annealing),以及神经网络算法，粒子群算法等等。我们在本课题中选用了遗传算法。

4.2 遗传算法

4.2.1 遗传算法原理

遗传算法是模仿自然界生物进化机制而产生的全局搜索与优化方法，它借鉴了生物学中包括遗传杂交、基因突变、染色体易位等一系列的自然现象。遗传算法中问题的一组解被称为一个个体，其特征通过结构体中的一组参数进行表征，这些参数被称为染色体。

遗传算法将会在满足限制条件(仅包括自变量范围和其线性约束，不包括非线性约束)的情况中随机产生个体，每个个体都将使用用户定义的适应度函数得到其适应度数值。适应度数值越高的个体将有更多的机会进行交配，从而将其特征延续，使各代的适应度在进化中逐渐提高。

4.2.2 遗传算法流程

(1)选择初始生命种群。尽管遗传算法是全局优化算法，但是合理地选取初始种群的特征及数量能有效地提高计算效率。

(2)循环。

评价种群中的个体适应度。适应度函数是算法筛选的依据，不当的适应度函数可能导致结果收敛于局部最优。

以比例原则选择产生下一个种群，常见的选择方法有轮盘法，竞争法等，通过合理的选择方法同样可以避免早熟的情况。

改变种群，即进行交叉变异等操作，使得种群有机会跳出局部最优解。

(3)满足终止条件时终止。一般的终止条件有限定进化次数，限定计算资源耗费，适应度函数饱和等。[3]

4.2.3 为什么选择遗传算法

我们选择遗传算法，一方面是因为它是一个较为完善的全局优化算法，在 Matlab 中可以

直接调用工具箱中写好的函数，另一方面是因为课题要求中有倾向性地介绍了该算法。

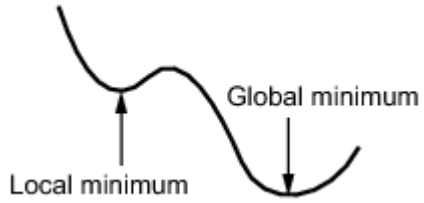


图 6 局部最优与全局最优示意图

遗传算法通过变异等方式跳出局部最优解的区间，而达到全局优化的目的。同时，按比例原则的遗传可以继承上一代中良好的特征，而区别于简单随机。在 Matlab 的 global optimization 工具箱中包含了遗传算法，并且我们可以通过调整改变变异交叉的方式参数、人口参数、精英保留等等参数使算法达到我们要求的精度。

遗传算法作为一种从自然界物理现象中发展而来的算法，相比其他打包的优化方法来说更为直观。对于该课题数据的静态数据，算法能较快地求解出最优解。因此，对于此课题选择遗传算法是可行的。

当然遗传算法也有缺陷。遗传算法对算法的精度、可行度、计算复杂度等方面不能定量的分析。遗传算法的计算量远远大于模式化搜索，相比其他一些优化算法也较大，导致优化所需时间较长。

4.2.4 遗传算法参数的设定

出于便捷直观的目的，在后期的参数调节中我们直接使用了 Optimization 工具箱 GUI，其界面图如下，可以直接在界面上进行参数的设置，并且支持在运行中改变部分设置。

界面左侧为待优化问题的解决方法和适应度函数以及一些变量的限制，右侧为对遗传算法本身的参数的设置。我们后期的工作也主要集中于对遗传算法中参数的设置调整。

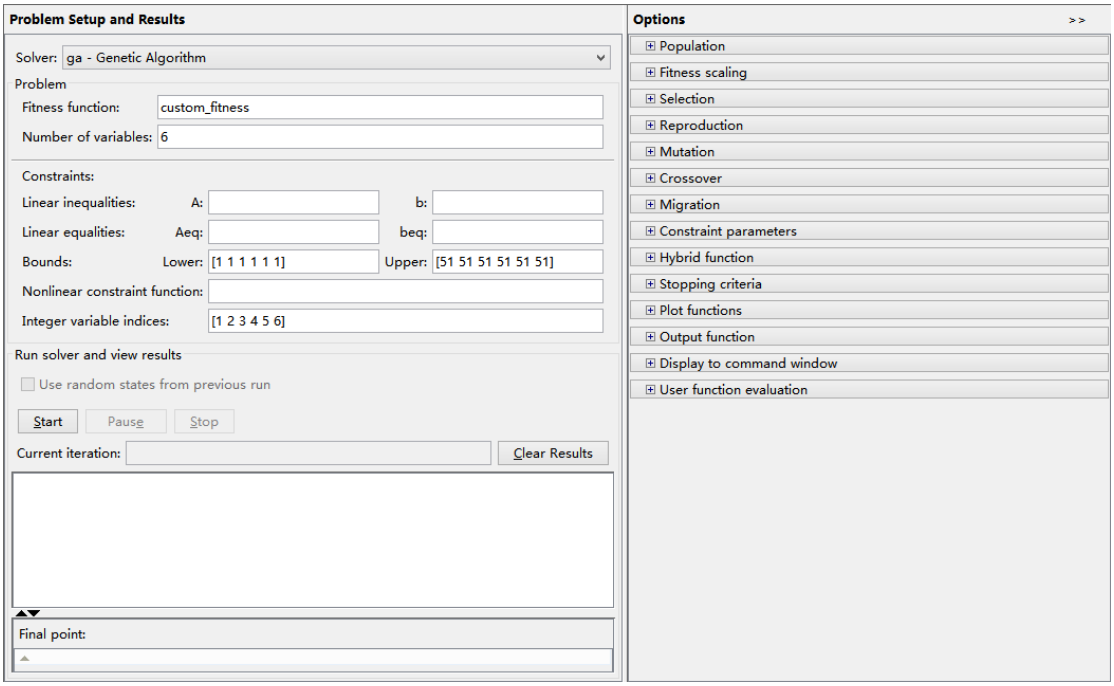


图 7 optimization 工具箱

为了详细了解遗传算法的机理以便更好的调整算法，我们对其每一项设置都进行了分析。参考 Matlab 的说明文档并结合自己在实践中的心得，我们归纳整理所有选项和其分选项的内容如下表（由于有的名词在中文中找不到对应的官方翻译，故一些名词仅按照其作用进行了意译），可供不了解遗传算法运作机理以及希望了解算法中函数选择的丰富性的同学参考。

表 1 遗传算法设置一览表

Population	i. Population type ii. Population size iii. Creation function	➢ 人口种类：实验中使用了Double Vector。matlab作为支持矩阵运算的语言，对于向量的支持还是较好的。Bit String对本问题来说没有明显优势还显得麻烦。 ➢ 人口数量： 单个种群采用default:max(min(10*numberOfVariables,100),40)。理论上，大种群能更好地模拟自然界中的遗传，但出于速度和内存的考虑使用默认值即可。 此外，为了丰富内容，实验中将人口数量设成了向量的形式，每一分量对应着一个子种群的人口数量，并对分量的大小在默认值附近进行调节，模拟了不同大小种群中的遗传过程，以便在后续操作中引入迁徙的步骤。 ➢ 创造函数：决定第一代的种群。使用uniform，即均匀随机产生。由于对于变量做了整数的限制，feasiable将无视该限制，constraint dependent将退化为uniform，故直接使用uniform，从1：51中随机产生整数。
Fitness scaling	Scaling function	➢ 评价函数：不同于适应度函数，该函数将适应度函数处理后转交给选择函数。实验中采用了Proportional，即比例形式，将使适应度好的个体在选择时有显著的优势。这样操作将会使种群成熟得更快，减少了计算时间。 在实践中注意到本问题中早熟并不是主要的问题，主要问题是不能在限定的代数中找到更优解，因此使用Proportional是合理的。例如对一组曲线进行操作的时候使用rank（进行排名之后丢弃具体适应度），则有很大概率在成本为74的时候就因为到达无更佳值的循环最大代数而停止，Proportional则能进行到73左右。 另外，Top的方式对于位于最佳的一定比例的平等对待，完全抛弃差的一部分，不能很好的利用遗传算法变异的优势，并且在对一组曲线进行实验时代数达到了上述方式的两倍，故不予采用。
Selection	Selection function	➢ 选择函数：根据处理后的适应度决定哪些个体产生子代。采用了Stochastic uniform,该方法为将各个体的处理后适应度在数轴上给予相同长度，随机选点开始后进行等步长跳跃，落入区域对应的个体将获得一次产生后代的机会。在直观上该方法等同于几何概型实验来决定概率。 另外Roulette轮盘赌的方式与此接近，不过观察随机数落入区域的方式进行。Tournament则保留最佳的一定比例，并从剩余的个体中随机产生需要的父代。选择Stochastic uniform并不是因为其有显著优势，而是因为它是默认的方式并且速度上更快且支持有整数限制的变量。
Reproduction	i. Elite count ii. Crossover fraction	➢ 精英数量：采用default:0.05*max(min(10*numOfVariables,100),40)。实验中实际为5%。精英即能确保保留进入下一代的个体。精英个体的存在有助于种群最佳解的稳定，避免了最佳解被抛弃而导致的额外工作量。不过同样，精英比例若太高将影响遗传算法的效果，故依旧采用默认值。 ➢ 交配率：交配即基因片段在不同个体之间重新组合而不产生新的基因，可以保留优良性状的点用于重新组合成更优解。实验中采用default=0.8，而没有做过多改动。 注意到遗传算法中除了Elite,Crossover的部分，剩余的为Mutation部分，即发生突变的部分。故在设定上述两个参数的同时也设定了突变率。突变率过高将使解难以收敛，突变率过低则容易错过最优解，收敛于局部最优。
Mutation	Mutation function	➢ 变异函数：决定了个体变异的方式。 采用默认的Constraint dependent,其使用Adaptive feasible方法，因为本实验中个体值为整型矢量，故无法采用Gaussian,Uniform这些方法。Adaptive feasible将向有益的方向变异，尽管并不是完全随机的变异，但是将使解更快的收敛到多个局部最优，并从中获得全局最优。
Crossover	Crossover function	➢ 交配函数：决定交配过程中双亲的基因如何重新组合。 采用默认的Constraint dependent,其使用的是常用的Scattered,将产生一个大小与变量数量相等的二进制串，每一位随机取1或0，取1的位数选取双亲中A的基因，取0的位数选取B的基因。 另外Single point,Two point功能大致相同，以Single point为例，其随机产生一个小于变量数量的整数，在该整数以前采用A的基因，之后采用B的基因，Two point则是选定了需要采用A基因的区间。 另外Intermediate,Heuristic,Arithmetic功能则大致相同。将A,B基因组分别映射到向量空间中的一个点，并做其连线，Arithmetic将在连线上随机取点，Intermediate则有Ratio(<1)参数控制，取点A+rand*Ratio*(B-A)，即在以A为起点的一定长度的连线上取点。Heuristic方法，取点A+rand*Ratio*(B-A)，其中Ratio>1, 即点将有机会取到连线外朝向更优方向的点，有助于快速找到局部最优。
Migration	i. Direction ii. Fraction iii. Interval	➢ 迁徙方向：迁徙即某一子种群中最优秀的部分取代另一子种群中最糟糕的部分。方向有Forward,即向下一子种群迁徙，还有Both,向两侧迁徙，实验中选择了Both。 ➢ 迁徙比例：迁徙人口的比例，大小为较小的子种群的人口数量乘上Fraction。采用默认值default=0.2。 ➢ 迁徙间隔：决定迁徙发生的间隔代数。采用默认值default=20。过大的迁徙率和过短的迁徙间隔起不到明显的效果并且会增加算法运行之间，故采用了较为中庸的默认值。
Constraint parameters	i. Initial penalty ii. Penalty factor	➢ 本部分主要用于解决对于非线性约束中遗传算法产生的困难，通过调节惩罚参数可以尽量将不符合非线性约束的点从种群中剔除出去。本实验没有用到。
Hybrid function	Hybrid function	➢ 在运行完遗传算法之后，对所得解用另一优化函数进行处理。可选的函数有fminsearch,patternsearch,fminunc,fmincon。本实验中仅使用遗传算法，不进行算法混合。
Stopping criteria	i. Generations ii. Time limit iii. Fitness limit iv. Stall generations v. Stall time limit vi. Stall test vii. Function tolerance viii. Constraint tolerance	➢ 最大代数：定义最大的迭代次数，使用default=100*numOfVariables ➢ 最大时间：使用default=inf。由于没有特别大的事件限制，在运行时以确保能得到优良的解为主要目的。实际运行一次的时间事实上大概在两天左右，也不方便使用时间参量进行限制。 ➢ 适应度限制：当适应度降低到适应度限制以下时即停止操作，由于在实际运行中不能得到最低限度的解（误差成本为零的情况），故仍希望其进一步进行优化。使用default=-inf,设置成12*numOfVariables也是一样的效果。 ➢ 平摊代数：该代数表示计算最佳值变化的平均值所采用的代数，当平摊代数内平均变化量小于设定值，算法终止。使用default=50。 ➢ 平摊时间：与上一参数相似，只不过以时间衡量。采用default=Inf。 ➢ 平摊测试：决定上述两参数中计算平均变化量时对各代数特征值的加权。Average change为算数平均。Geometic weighted以(1/2)^n作为加权值。采用Average change。 ➢ 函数容忍值：即之前提到的平摊中的设定值。使用default=1e-6。 ➢ 限制容忍值：允许个体对非线性限制的最大违背值。实验中不使用。
Plot functions	/	运行时的绘图功能，附表中对各功能进行了解释。
Output function	/	输出函数。本实验中直接读取最佳值点即可，不需另写输出函数。
Display to command window	/	设定运行时在matlab命令行窗口中显示的内容。由于采用了GUI，无需从命令行中获取信息。
User function evaluation		决定适应度函数以何种方式进行计算。有serial,每个个体的适应的分别计算。vectorized，将一代的个体的适应度函数按照一组矢量来计算。Parrallel,并行计算，进行包括多核处理器与多处理器网络上的计算。 因为在适应度函数中需要进行曲线拟合，因此使用vectorized对本问题的帮助不大。 由于现在电脑使用的大多是多核处理器，实验中我们使用的电脑是i5处理器，支持parrallel，因此在运行过程中开启了并行计算

表 2 Plot 功能表

Best fitness	最佳值
Best individual	最佳值对应的点
Distance	一代中个体间平均距离
Expectation	不同适应度产生子代数量的期望值
Genealogy	家谱
Range	所有个体的范围（Best, Worst&Mean scores）
Score diversity	个体的适应度分布（统计）
Scores	所有个体的适应度（具体个体）
Selection	产生子代的个体分布（统计）
Stopping	终止条件满足状况
Max constraint	（非线性的，用不着）

由上述表可见，Matlab 在 GA 工具箱中提供了大量算法的选项，最核心的包括选择、变异、交配的函数。我们在这些选项中经过选择的项大多是默认项，这也从侧面印证了在默认项的选择上，工具箱的设计者是以在时间耗费不至于高昂的前提下以尽可能得找出最优解为目的的。

4.2.5 遗传算法运行结果

Matlab 作为用矩阵完成运算的语言，相比其他语言效率并不高。在使用遗传算法这样语句量很大的语言是更是如此，这就限制了实验的可重复性，我们在实验中对于同一设置一般运行 2 遍以排除偶然性。

选点个数在上文中始终没有提到，因为我们并不能通过分析直接确定选点个数的最优值，而需通过实验在一个小范围内筛选。

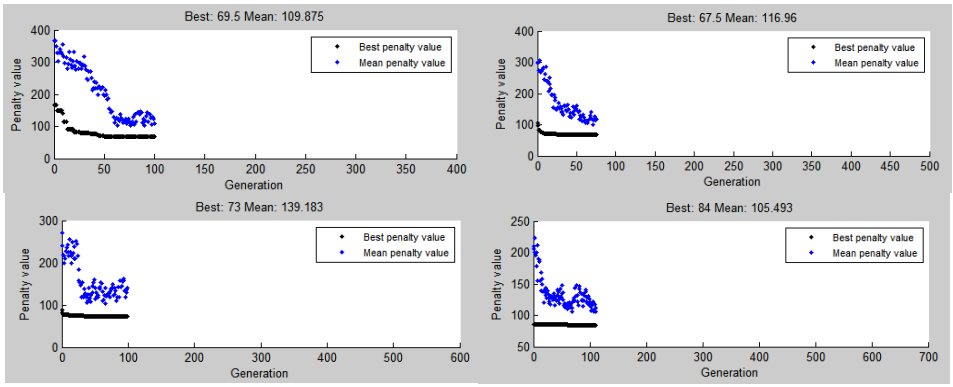
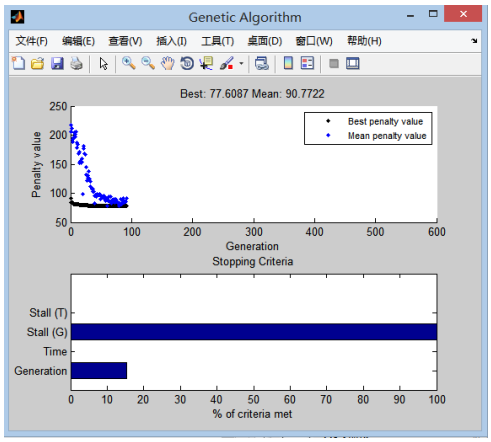


图 8 不同选点数量的单样本最佳适应度曲线图

上图分别为选取了一组点时使用 4,5,6,7 个取点时的实验情况，可以从中推断所有样本处理时的情况。注意到随着样本增加会使同一组点的普适性降低，适应所有样本远比适应某单个样本困难，因此会导致平均误差成本的上升。4 个点的方案在 1 组样本时成本便高于 5 组，明显不可行。而 7 个点的方案，尽管误差成本在一个样本的时候降为了零，但在后续实验中发现 6 个点的误差成本也并不是特别大，7 个点方案增加了测量成本，得不偿失，故也舍弃。

在对所有 469 组样本运行过程序之后。我们得到取 5 个点的方案平均成本在 97.1 左右，而 6 个点的方案成本在 89.6 左右，故可认为 6 个点为最佳取点方案。



最佳的一次运行结果如左图，成本为 89.6087。运行时设置可参照之前设置表格。（由于疏忽，在运行 6 个点的方案时，测量成本仍使用了 5 个点时的值，故应在显示值上加 12，但平移变化并不影响实验进行。）

图 9 全样本最佳适应度曲线图

5 总结

从上述分析中可得，对于课题中提供的样本，低次多项式拟合不能很好地给出函数的曲线，高次多项式由于 Runge 现象造成的波动和样本点数量的限制所以不予采用。在观察拟合后残差等操作后，我们决定在实验中使用光滑样条拟合。

取点方案随着实验的进行而调整，经过实践，我们采用成本最低的方案——测量 6 点。

在样本点选择中，我们使用了遗传算法进行选择的优化。通过对样本量的扩大和选择、变异、交配函数的调整，从而使遗传算法获得较好的效果。

本例可以说明遗传算法在全局优化中的良好表现，该算法可以降低对于数据处理的数学操作难度，而使用模拟自然界遗传的方式进行评估筛选，同时，该算法可以通过其启发式的特性避免了穷举造成的时间耗费并获得较佳效果。

6 参考文献

[1] 袁焱. 统计推断在数模转换系统中的应用课程讲义[EB/OL].ftp://202.120.39.248.

[2] 朱琪. 高次插值的龙格现象的测试[D]. 湖南:湖南科技学院,2005:26-11.

[3] 维基百科.遗传算法[J/OL]
.http://zh.wikipedia.org/wiki/%E9%81%97%E4%BC%A0%E7%AE%97%E6%B3%95.

源代码

Run.m(只在前期实验中使用)

```
nvars=6;
LB=[1 1 1 1 1 1];
UB=[51 51 51 51 51 51];
i=1;
data=dataform;
options=gaoptimset('MutationFcn',@mutationadaptfeasible, ...
    'PlotFcns',{@gaplotbestf,@gaplotstopping},'Display','iter');
fitnessfcn=@(x)custom_fitnessv(x,data);
[x,fval]=ga(fitnessfcn,nvars,[],[],[],[],LB,UB,[],1:6,options);
```

Custom_fitnessv.m

```
function value=custom_fitnessv(x)
value=0;

dataform = importdata('mydata.mat');
i=1:469;
i=i*2;
k=0:0.1:5;
data=dataform(i,:);

for index=1:469
fitobject=fit(k(x)',data(index,x)','smoothingspline');
sum=0;
for j=1:51
dif=abs(data(index,j)-fitobject(k(j)));
if dif<=0.5
    sum=sum+0;
elseif dif<=1
    sum=sum+0.5;
elseif dif<=2
    sum=sum+1.5;
elseif dif<=3
    sum=sum+6;
elseif dif<=5
    sum=sum+12;
else sum=sum+25;
```

```
end
end
value=value+sum+84;
end
value=value/469;
end
```