

利用折半查找和拉格朗日插值算法 计算机车牵引 / 制动包络线的方法

黄 浩¹, 曹 霄¹, 何 伟²

(1. 株洲南车时代电气股份有限公司 技术中心, 湖南 株洲 412001;
2. 株洲变流技术国家工程研究中心有限公司, 湖南 株洲 412001)



作者简介: 黄 浩 (1981-), 男, 硕士, 主要从事机车主变流器逻辑与通信控制。

摘 要: 提出了一种利用折半查找算法查找电机转速数据, 在查找不成功时再使用拉格朗日插值算法计算机车牵引制动电机转矩值的方法, 通过将此算法与经典电机特性包络线计算算法在计算精度和时间复杂度上进行对比分析, 验证了该算法的性能较经典计算方法计算精度更高, 而时间复杂度又在可接受范围内。

关键词: 折半查找; 拉格朗日插值; 包络线; 牵引 / 制动; 电机转速

中图分类号: U260.13⁺1; U264.1 **文献标识码:** A **文章编号:** 1000-128X(2012)05-0099-04

Method for Calculating Locomotive Traction & Brake Limited Line Based on Binary Search and Lagrange Interpolation

HUANG Hao¹, CAO Xiao¹, HE Wei²

(1. Technology Center, Zhuzhou CSR Times Electric Co., Ltd., Zhuzhou, Hunan 412001, China;

2. Zhuzhou National Engineering Research Center of Converters Co., Ltd., Zhuzhou, Hunan 412001, China)

Abstract: A method was proposed by using binary search arithmetic searching motor speed, if missed then employing Lagrange interpolation method to calculating motor torque of locomotive's traction and brake. By comparing calculation accuracy and time complexity with the classical algorithm in envelope curve of motor characteristic, the presented method was proved have better calculation accuracy and acceptable time complexity.

Key words: binary search; Lagrange interpolation; envelope curve; traction and brake; motor speed

0 引言

电力机车电机牵引 / 制动特性包络线的设计是电力机车运行控制的关键, 针对不同类型电力机车, 其电机特性各不相同。如何根据电机厂商给出的电机特性表, 准确描述电机牵引 / 制动特性曲线, 是机车电机控制的重要环节。本文将提出一种先利用折半查找法定位电机转速及相对应的电机转矩, 再使用拉格朗日插值算法计算电力机车电机牵引 / 制动包络线的算法 (Binary Search and Lagrange Interpolation, 以下简称为BSLIA算法)。

1 算法设计

一般电机供应商提供的电机特性计算表均具有以下4个特点:

① 电机转速可视为从零到最大转速的离散型有限

元递增数列;

② 电机转矩可视为随电机转速增加而减小的离散型有限元递减数列;

③ 某个电机转速下存在唯一一个与之相对应的电机转矩, 即电机转速与电机转矩存在一一对应函数关系;

④ 电力机车牵引 / 制动包络线的组成可根据电机转速与相应的电机转矩两者的关系分为3段, 分别为直线段 (两者成正比例关系)、恒功曲线段 (两者成反比例关系) 和自然特性段 (两者成一元二次函数关系)。

折半查找 (Binary Search) 的过程定义: 在有序数列中, 先确定待查记录所在的范围, 然后逐步缩小范围直到找到或找不到该记录为止^[1]。根据电机特性计算表的特点①和②, 电机转速和电机转矩均满足折半查找过程的有序数列要求; 根据特性③, 利用折半查找算法可定位在机车运行过程中电机某转速下给定的转矩值。

收稿日期: 2012-02-22; 收修改稿日期: 2012-07-12

当速度传感器采集的电力机车实际电机转速（简称为AR）与电机供应商提供的电机特性计算表中电机转速（简称为SR）完全匹配时，折半查找算法可直接定位相应的电机转矩值供电机控制使用；当AR与SR不匹配时，折半查找算法可记录查找不成功时的最后位置（根据折半查找的特点，查找不成功时，查找到的最后位置的值一定是与被查找值在数列中大小最接近的值），由该位置可取出电机转速数列和电机转矩数列相对该位置的前续或后续的值。

根据电机转速计算电机转矩值采用如下方法：

对直线段和恒功曲线段采用拉格朗日线性插值算法^[2]，其线性插值公式为：

$$z = \sum_{i=0}^1 y[i] \prod_{\substack{j=0 \\ j \neq i}}^1 \frac{(x - x[j])}{(x[i] - x[j])} \quad (1)$$

对自然特性段采用拉格朗日二次（抛物线）插值算法^[2]，其二次插值公式为：

$$z = \sum_{i=0}^2 y[i] \prod_{\substack{j=0 \\ j \neq i}}^2 \frac{(x - x[j])}{(x[i] - x[j])} \quad (2)$$

式中： x 为当前电机实际转速，即折半查找待匹配值； $x[j]$ 为折半查找匹配不成功时定位的最接近 x 的值； $y[i]$ 为与 $x[i]$ 或 $x[j]$ 相对应的电机转矩值； z 为经过拉格朗日差值计算后的电机转矩值。

1.1 折半查找定位算法

针对电机特性计算表的折半查找定位算法设计如下：

分别将电机转速数列和电机转矩数列按顺序各存入一个一维数组，再根据当前速度传感器采集的电力机车实际电机转速为待查找值进行折半查找定位；若在电机转速数组中成功找到与当前实际电机转速相匹配值，则返回当前查找一维数组位置指针，并以该指针定位相应电机转矩数组位置指针，在电机转矩一维数组中取出相对应的电机转矩值供电力机车电机控制使用；若当前电机转速值在电机转速一维数组中找不到相匹配的电机转速，则记录下折半查找不成功时指针所指向电机转速数组的最后位置，以该指针值为定位点，取出电机转速数组和电机转矩数组位置该指针对应值及其前续或后续位置指针的电机转速值和相应电机转矩值（具体是取前续还是取后续视数组当前指针位置而定，后面有具体描述），供拉格朗日插值算法计算当前电机转速对应电机转矩值使用。

上述折半查找定位过程伪代码如下所示：

```
int Pivot;
int Binary_Search(Rotate_Speed, Length, Motor_Speed)
{
    low = 0; high = Length - 1; //置查找区间上下限
    while(low <= high)
    {
```

```
        if(low == high)
        {
            if(Rotate_Speed[high] == data) //找到
                待查元素
                return(low);
            else
            {
                Pivot = low; //没有找到相
                匹配值，记录下当前最后定位位置
                return(Pivot);
            }
        }
        mid = (low + high)/2;
        if(Rotate_Speed[mid] < data) //继续在前半
            区间进行查找
            low = mid + 1;
        else if(Rotate_Speed[mid] > data) //继续在后
            半区间进行查找
            high = mid - 1;
        else
            return(mid);
    }
    if(low > high)
    {
        Pivot = mid;
        return(Pivot);
    }
}
```

其中，定义Rotate_Speed[]为电机转速数组，Motor_Torque[]为电机转矩数组，Length为电机转速数组和电机转矩数组的长度（电机转速和电机转矩为一一对应关系，故两数组长度相等），Pivot为在查找定位无匹配值时的最后定位位置，Motor_Speed为当前电力机车实际电机转速。

1.2 直线段和恒功曲线段的拉格朗日线性插值算法

当折半查找算法未找到与当前实际电机转速Motor_Speed相匹配项时，在电机特性的直线段和恒功曲线段使用拉格朗日线性插值算法。程序伪代码如下所示：

```
float Lagrange1(Rotate_Speed, Motor_Torque,
Length, Pivot, Motor_Speed)
{
    float z, t;
    t = Motor_Speed;
    if(Pivot == (length - 1)) //Pivot为数组末尾位
        置，取其前继点插值计算
    {
        z = (y[pivot-1]*(t-x[pivot]))/(x[pivot-1]-x[pivot]) +
```

```
y[pivot]*(t-x[pivot-1])/(x[pivot]-x[pivot-1]));
    }
    else
    {
        if(x[pivot] < t) //取Pivot后继点插值计算
        {
            z=(y[pivot]*(t-x[pivot+1])/(x[pivot]-
x[pivot+1])+y[pivot+1]*(t-x[pivot])/(x[pivot+1]-x[pivot]));
        }
        else //取Pivot前继点插值计算
        {
            z=(y[pivot-1]*(t-x[pivot])/(x[pivot-1]-
x[pivot])+y[pivot]*(t-x[pivot-1])/(x[pivot]-x[pivot-1]));
        }
    }
}
```

其中, 定义电机特性直线段的起始点与恒功段的终止点分别为 $Line_{min}$ 和 $Power_{max}$ 。若在折半查找定位中未找到与当前电机转速相匹配的电机转矩值, 且当前电机转速 $Motor_Speed \in [Line_{min}, Power_{max}]$, 则采用拉格朗日线性插值算法。

1.3 自然特性段采用拉格朗日二次 (抛物线) 插值算法

设自然特性段的终止点为 $Nature_{max}$ (电机的最大转速)。若在折半查找定位中未找到与 $Motor_Speed$ 相匹配的电机转速值, 且当前电机转速 $Motor_Speed \in [Power_{max}, Nature_{max}]$, 则采用拉格朗日二次 (抛物线) 插值算法。程序伪代码如下所示:

```
float Lagrange2(Rotate_Speed, Motor_Torque, Length,
Pivot, Motor_Speed)
{
    int i,j,k,m;
```

```
float z,s,t;
z=0.0;
t=Motor_Speed;
if(t < x[length-3])
{
    k=length-4;
    m=length-2;
}
else if(t > x[length-3])
{
    k=length-3;
    m=length-1;
}
for(i=k;i<=m;i++)
{
    s=1.0;
    for(j=k;j<=m;j++)
        if(j != i)
            s=s*(t-x[j])/(x[i]-x[j]);
            z=z+s*y[i];
}
}
```

2 仿真试验对比分析

为验证本文设计BSLIA算法性能, 现以某厂HXD2型电力机车使用的某型牵引电机牵引/制动特性为例, 将BSLIA算法与经典电机特性包络线计算算法做性能对比。

某厂HXD2型电力机车所使用某型电机特性计算表中牵引和制动电机转速和电机转矩对应关系分别如表1和表2所示。

牵引工况下电机转速和电机转矩对应关系可分为

表1 电机厂商提供的牵引特性表

转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$	转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$	转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$	转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$
0	11 399.5	1 316.8	8 530.0	1 498.8	8 122.6	2 347.5	5 186.6
219.6	10 876.4	1 347.2	8 461.8	1 500.8	8 112.0	2 516.5	4 838.2
439.1	10 398.4	1 392.7	8 360.7	1 668.9	7 294.8	2 769.2	4 396.5
658.5	9 919.5	1 423.0	8 292.0	1 838.8	6 621.6	2 865.2	4 113.9
878.0	9 441.3	1 438.2	8 258.5	2 008.6	6 061.2	2 961.2	3 857.5
1 097.4	8 963.1	1 468.5	8 191.6	2 178.2	5 589.3	3 057.2	3 624.1

表2 电机厂商提供的制动特性表

转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$	转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$	转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$	转速 / $r \cdot \min^{-1}$	电机转矩 / $N \cdot m$
77.4	7 200.0	1 052.5	7 200.0	1 769.5	6 746.7	2 630.9	4 538.0
245.5	7 200.0	1 222.5	7 200.0	1 911.2	6 246.7	2 773.1	4 305.0
402.5	7 200.0	1 382.9	7 200.0	2 053.0	5 815.8		
572.5	7 200.0	1 453.8	7 200.0	2 194.8	5 440.4		
732.5	7 200.0	1 535.5	7 200.0	2 346.9	5 088.2		
892.5	7 200.0	1 624.6	7 200.0	2 488.8	4 797.7		

3 段,分别表示为:

①直线区间段:电机转速数列上下限分别为0~1 498.8 r/min;

②恒功曲线区间段:电机转速数列上下限为1 498.8~2 769.2 r/min;

③自然特性曲线区间段:电机转速数列上下限为2 769.2~3 057.2 r/min。

制动工况下电机转速和电机转矩对应关系可分为2 段,分别表示为:

①直线区间段:电机转速数列上下限分别为0~1 658.1 r/min;

②恒功曲线区间段:电机转速数列上下限为1 658.1~2 773.1 r/min。

经典算法计算电机转速和相应电机转矩值方法如下:直线段采用直线方程方式;恒功曲线段采用反比例函数方式;自然特性段采用一元二次函数方式。

在AMD公司 Athlon(tm)X2 DualCore 2.10GHz处理器和1G内存环境下,利用VMWare Workstation虚拟机中Linux Redhat 9.0编程环境的vi编译器分别实现了BSLIA算法和经典算法。

下面将从电机转矩的计算精度和算法时间复杂度两方面对比BSLIA算法和经典算法。对牵引和制动特性各随机取10个速度点进行模拟计算,算法消耗时间对每个速度点各模拟20次后取平均值。BSLIA算法与经典算法计算的电机牵引/制动转矩值对比如表3、表4所示。

对比分析表1和表3的牵引特性数据,当电机实际

表3 BSLIA算法与经典算法计算的电机牵引转矩对比

转速 / r · min ⁻¹	BSLIA 算法 电机转矩 /N · m	经典算法 电机转矩 /N · m	转速 / r · min ⁻¹	BSLIA 算法 电机转矩 /N · m	经典算法 电机转矩 /N · m
219.6	10 876.4	10 919.38	1 837.0	6 628.732	6 627.193
658.0	9 920.592	9 960.882	2 179.0	5 587.397	5 587.037
878.0	9 441.3	9 479.886	2 350.0	5 181.446	5 180.491
1 300.0	8 563.164	8 557.246	2 960.0	3 860.528	3 847.967
1 500.0	8 116.24	8 116.102	3 057.2	3 624.1	3 607.182

表4 BSLIA算法与经典算法计算的电机制动转矩对比

转速 / r · min ⁻¹	BSLIA 算法电机 转矩 /N · m	经典算法 电机转矩 /N · m	转速 / r · min ⁻¹	BSLIA 算法电机 转矩 /N · m	经典算法 电机转矩 /N · m
77.4	7 200.0	7 200.0	2 050.0	5 824.916	5 823.571
150.0	7 200.0	7 200.0	2 346.9	5 088.200	5 086.847
402.5	7 200.0	7 200.0	2 489.0	4 797.335	4 796.432
1 620.0	7 200.0	7 200.0	2 620.0	4 557.920	4 556.611
1 769.5	6 746.7	6 746.7	2 775.0	4 301.887	4 302.090

转速与表1中电机转速值完全相同时(如电机转速点219.6 r/min、878.0 r/min、3 057.2 r/min),BSLIA算法可通过折半查找表1直接定位。BSLIA算法计算的电机转矩与电机厂商提供值完全相同,计算精度优于经典算法。在直线区间段,BSLIA算法精度在电机转速点658.0 r/min、1 300.0 r/min计算的电机转矩值与表1中电机转速点658.5 r/min、1 316.8 r/min的电机转矩值很接近。对比表2中经典算法计算值后可发现BSLIA算法计算的电机转矩更接近实际值。在恒功曲线区间段,在电机转速点1 500.0 r/min、1 837.0 r/min、2 179.0 r/min和2 350.0 r/min,BSLIA算法和经典算法计算的电机转矩值非常接近,BSLIA算法精度略高,所计算的电机转矩值略大,实际使用效果将优于经典算法。在自然特性曲线区间段,在电机转速点2 960.0 r/min,采用本文的拉格朗日二次插值算法计算的电机转矩值与表1中电机转速点2 961.2 r/min的电机转矩值比较可发现,BSLIA算法较经典算法计算值更接近电机特性表1中的实际值。因此,在计算牵引特性时BSLIA算法较经典电机特性算法计算电机转矩值要精确。

对比分析表2和表4的制动特性数据可知,两算法

计算精度相差不大。这是因为表2中制动特性在直线区间段为恒转矩区间,电机转矩均为7 200 N · m,两算法计算精度相同。在恒功曲线区间段,分析方法与牵引特性的恒功区间段相似,BSLIA算法精度与经典算法精度非常接近,BSLIA算法精度略高。

由以上分析可知:当电机特性表中存在与电机实际转速值相同的数值时,BSLIA算法可直接取出与之相对应的电力转矩值;当电机特性表中不存在与电机实际转速值相同的数值时,通过拉格朗日插值算法计算的电机转矩值较经典算法精度要高,而且由于BSLIA算法选取的插值点数较少,线性插值选取2个点,二次插值选取3个点,不会导致拉格朗日插值多项式的次数很高,从而出现计算数值的不稳定(即尽管在已知的几个点取到给定的数值,但计算值却会与实际值之间存在很大偏差)^[3]。

BSLIA算法与经典算法的时间复杂度对比如图1和图2所示。可见虽然BSLIA算法较经典算法复杂,但是时间复杂度和经典算法差不多,有时甚至优于经典算法。理论分析认为,经典算法时间复杂度理论值为 $O(n)$,折(下转第106页)

6 关于襄渝线机车交路优化的探讨

①将HXD1型机车配属武汉局襄阳机务段,机车交路由襄阳北贯通至达州。调整机车配属,可以解决以下问题:

- 机车配属与车流方向一致,有利于襄渝线畅通。
- 便于日常调度组织,避免由重庆西调整单机到襄阳北开车,有利于提高机车运用效率。
- 襄渝线货物列车牵引定数可以提高到4 500 t,普超5 000 t,使襄阳北站5个方向货物列车牵引定数基本一致,减少襄阳北编组站因各线牵引定数不匹配的解编作业,提高襄阳北编组站的作业效率,满足入渝、入川物资日益增长的运输需求。

②襄阳北贯通至达州货物列车,预计需要使用HXD1型机车100台。

襄渝线襄阳北至安康东间货车49对,其中40对可以贯通至达州,其余9对为襄阳北—安康东间区段列

车。其中:襄阳北—达州间图定40对直通货车,按照1.6系数图定需要使用机车64台,加15%预备率后为74台。襄阳北—安康东间9对区段,按照1.0系数图定需要使用机车9台,加15%预备率后为11台。以上合计图定需要机车85台,考虑到襄渝线涉及武汉、西安、成都3个铁路局,日常调度指挥难度较大,因此,实际需要超配部分机车,按100台配备方可满足襄渝线货车开行需要。

参考文献:

- [1] 铁道部运输局装备部. 铁路机车概要 [M]. 北京: 中国铁道出版社, 2009.
- [2] 张曙光. HXD1型电力机车 [M]. 北京: 中国铁道出版社, 2009.
- [3] 王长明. 铁路机车运用管理 [M]. 北京: 中国铁道出版社, 2004.
- [4] 孙中央. 列车牵引计算规程实用教程 [M]. 北京: 中国铁道出版社, 1999.

(上接第102页)

半查找算法的时间复杂度为 $O(\log_2(n+1)-1)$,拉格朗日插值算法时间复杂度为 $O(n^2)$,因此BSLIA算法时间复杂度理论值为 $O(\log_2(n+1)+n^2-1)$ 。虽然在算法时间复杂度的理论值上BSLIA算法较经典算法高,但针对电机牵引特性表的实际应用时,BSLIA算法最多只需考虑二次插值的情况,无需考虑更高次的插值过程,且存在当电机特性表中有与电机实际转速值相同的数值时,BSLIA算法可直接取出与之相对应的电力转矩值而无需调用拉格朗日插值算法的情况,因此BSLIA算法在实际操作时较经典算法时间复杂度差不多,在能直接找到配置值时的时间复杂度甚至优于经典算法。

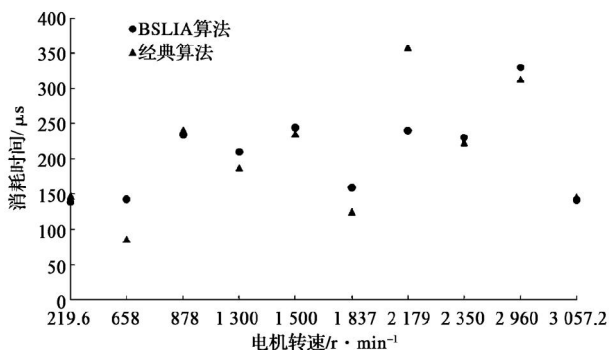


图1 牵引转矩计算时间比较

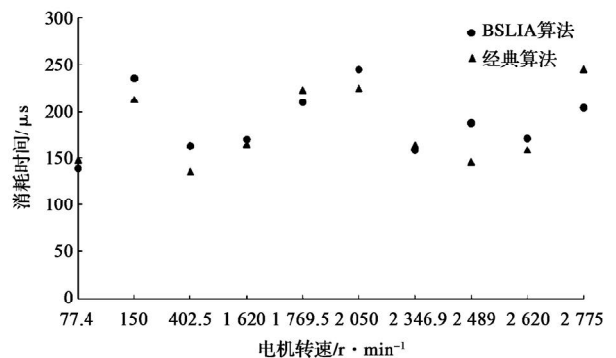


图2 制动转矩计算时间比较

3 结语

由上述仿真试验对比可知,经典算法虽具有算法实现简单、时间复杂度低的优点,但在电机转矩计算精度存在一定误差,该误差可能会对电力机车的电机控制造成不利影响。本文设计的BSLIA算法虽实现较复杂,但电机转矩计算精度较经典算法高,更有利于电力机车的电机控制,且BSLIA算法的时间复杂度也不高,完全在可接受范围之内。

参考文献:

- [1] 严蔚敏,吴伟民. 数据结构(C语言版) [M]. 北京: 清华大学出版社, 2003: 218-221.
- [2] 徐士良. 数值分析与算法 [M]. 北京: 机械工业出版社, 2007: 188-191.
- [3] 冯有前. 数值分析 [M]. 北京: 清华大学出版社, 2005: 64.