东南大学

毕业设计（论文）报告

题目 B 样条曲线的近似合并

数 学 院（系） 信息与计算科学 专 业

学 号 07210114

学生姓名 杜小东

指导教师 朱平

顾问老师

起讫日期 2013.11—2014.6

设计地点 九龙湖校区

2014 年 5 月 29 日

B 样条曲线的近似合并

07210114　杜小东
指导教师　朱平

摘　要

本文讲解了合并两个相邻的 B 样条曲线为一个 B 样条曲线的近似的问题。基本思想是对两条 B 样条曲线进行均匀或非均匀节点采样，然后再将采样的节点进行 B 样条逼近，最后再调整起始点和末端点的位置，这样就得到了近似合并后的 B 样条新曲线。为了得到合并后的曲线，需要找到 B 样条逼近的算法。

第一部分，我们引出合并问题的由来以及近似合并的在计算机辅助几何设计（CAGD）几何通信方面的用处，然后讲述了论文的核心载体 B 样条曲线及其性质，简单介绍了前辈们对于合并的一些研究。

第二部分，我们通过二次和三次 B 样条自由曲线讲解了 B 样条曲线的两种算法：deBoor 算法和节点插图算法，对给定的大量数据，则介绍了两种整体逼近算法，而这也是 B 样条合并的核心方法；

最后，我们完整呈现了 B 样条从采取样点到整体逼近的全过程。

关键词： B 样条曲线 近似合并 逼近 采样

Approximate merging of B-spline curves

07210114 Du xiaodong
Zhu ping

Abstract

This paper explained the merger of two adjacent B-spline curve for a B-spline curve approximation problem. The basic idea is to sample two uniform or non-uniform B-spline curves node, and then approximate sampling node. Finally adjust the position of the starting point and end point. Thus we get the new approximate combined B-spline curve. In order to get the merged curve, we need to find the B-spline approximation algorithm.

The first part of our merger leads to the origin and the approximate combined in Computer Aided Geometric Design(CAGD) geometry communications usefulness, and then the paper describes the nature of the core carrier B-spline curve.And then, for a brief introduction to some research predecessors combined.

The second part, we explain the two algorithms B-spline curve by quadratic and cubic B-spline curve freedom:deBoor algorithm and node illustrations algorithm. Given a lot of data, then introduces two overall approximation algorithm,which B-spline method is the core of the merger.

Finally, We complete the overall presentation of the B-spline approximation of samples to be taken from the whole process.

Keywords: b-spline curve, merging , approximate, sampling

目 录

第一章　引言

1.1 B 样条曲线

B 样条函数

递归定义：B 样条基函数 $N_i^m(u)$ 是定义在参数节点 $\cdots < u_0 < u_1 < u_2 < \cdots$ 上的 m 次分段多项式，由以下公式决定：

$$N_i^r(u) = \frac{u - u_i}{u_{i+r} - u_i} N_i^{r-1}(u) + \frac{u_{i+r+1} - u}{u_{i+r+1} - u_{i+1}} N_{i+1}^{r-1}(u),$$

其中

$$N_i^0(u) = \begin{cases} 1, & u \in [u_i, u_{i+1}), \\ 0, & u \notin [u_i, u_{i+1}). \end{cases}$$

B 样条函数的性质：

1. 单位分解性：$\sum_i N_i^m(u) \equiv 1$;
2. 非负性：$0 \le N_i^m(u) \le 1$;
3. 局部支撑性：$N_i^m(u) = 0, \quad u \notin [u_i, u_{i+m+1}]$;
4. 连续性：$N_i^m(u)$ 是 $m-1$ 次连续可微的;
5. 微分差分公式：$N'_{i,k}(t) = \frac{k-1}{t_{i+k-1} - t_i} N_{i,k-1}(t) - \frac{k-1}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t)$;
6. 可退化性：若取节点 $t_{i-k+2} = t_{i-k+3} = \cdots = t_i < t_{i+1} = t_{i+2} = \cdots = t_{i+k-1}, \tau = (t - t_i)/(t_{i+1} - t_i) \in [0, 1]$, 则 $N_{j,k}(t) = B_{j-i+k-1}(\tau), t_i \le t \le t_{i+1}, j = i-k+1, \cdots, i$, 即定义在两端均为 $k-1$ 重节点的子区间上的 k 个 k 阶 B 样条基必定退化为 $k-1$ 次 Bernstein 基。

B 样条曲线

定义：B 样条曲线是分段参数多项式曲线 $\mathbf{P}(u) = \sum_{i=0}^{n} N_{i,k}(u) \cdot \mathbf{d}_i$, 其中 $\mathbf{d}_i (i = 0, 1, \cdots, n)$ 为控制顶点，也称 deBoor 点，控制多边形也称为 deBoor 多边形。$N_{i,k}(u) (i = 0, 1, \cdots, n)$ 是如上的 B 样条基函数，定义在参数节点序列 $\mathbf{U} : u_0 \le u_1 \le \cdots \le u_{n+k+1}$ 上的 k 次多项式样条。

$$\begin{cases} N_{i,0}(u) = \begin{cases} 1, & u_i \le u < u_{i+1}, \\ 0, & else, \end{cases} \\ N_{i,k}(u) = \frac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \frac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} N_{i+1,k-1}(u), \\ \frac{0}{0} = 0. \end{cases}$$

$N_{i,k}(u)$ 的双下标中的第二个下标 k 表示次数，第一个下标 i 表示序号。该递推式表明，确定 $N_{i,k}(u)$ 需要用到共 $k+2$ 个节点。称区间 $[u_i, u_{i+k+1}]$ 为 $N_{i,k}(u)$ 的支撑区间。

B 样条曲线的局部性质

K 次 B 样条的支撑区间包含 $k+1$ 个节点区间，在对参数区间上任一点 $u \in [u_i, u_{i+1}]$ 处，至多只有 $k+1$ 个非零的 k 次 B 样条 $N_{j,k}(u)(j = i-k, i-k+1, \cdots, i)$, 其他 k 次 B 样条在该处均为零。

考察 B 样条曲线定义在区间 $u \in [u_i, u_{i+1}]$ 上的那一曲线段，表示为：$\mathbf{p}(u) = \sum_{j=i-k}^{i} N_{j,k}(u) \cdot \mathbf{d}_j, \quad u \in [u_i, u_{i+1}]$。

局部性表述：k 次 B 样条曲线上参数为 $u \in [u_i, u_{i+1}]$ 的一点 $\mathbf{p}(u)$ 至多与 $k+1$ 个控制顶点 $\mathbf{d}_j(j = i-k, i-k+1, \cdots, i)$ 有关，与其他控制顶点无关；移动该曲线的第 i 个控制顶点 $\mathbf{d}_j$ 至多将影响到定义在第 i 个 k 次 B 样条支撑区间 (u_i, u_{i+1}) 上那部分曲线形状，对曲线的其余部分不发生影响。

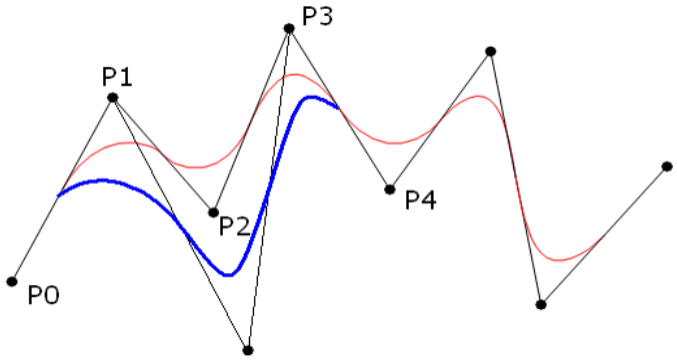

图 1.1 B 样条曲线的局部修改性

1.2 Bezier 和 B 样条合并问题

曲线曲面是计算机辅助几何设计（CAGD）系统中的基本工具，CAGD 的大多数操作都是以曲线曲面为对象的。而无论是根据给定的几何信息构造满足几何约束条件的曲线曲面，还是为压缩几何信息的数据量而近似转换曲线曲面，它们都被广泛应用于实际生产中，因而一直是人们关注的热点。正如 Hoschek 提到的一样，近似转换包括以下两个问题：

▷ 降阶逼近：找到和给定的 m 次曲线近似的 n 次参数曲线 $(n < m)$;

▷ 近似合并：合并尽可能多的 n 次曲线段来得到一条 m 次曲线段 $(n \leq m)$。

Bezier、球和 B 样条曲线曲面的降阶方法已被广泛研究。而近似合并，人们所做的工作还不多。利用 Bezier 曲线细分后的矩阵表示，根据原 Bezier 曲线与合并后的 Bezier 曲线间的距离函数取最小值的方法，给出把两相邻 Bezier 曲线合并成一条 Bezier 曲线

的方法。Chiew-Lan Tai 根据两相邻 n 次 B 样条曲线能够精确合并成一条 n 次 B 样条曲线的条件，利用控制顶点的扰动法，给出把两相邻 n 次 B 样条曲线合并成一条 n 次 B 样条曲线的方法，为使所得合并后的 B 样条曲线没有多余的节点，利用了在不改变原 B 样条形状下的节点调整方法，在合并过程中，还考虑了合并 B 样条曲线与原 B 样条曲线在端点插值的情形。

合并时数据减少的主要方法之一是：通过合并尽可能多的曲线段成一个曲线，可以减少所需要的几何通信的数据量。本文的主要目的是考虑 B 样条曲线的近似合并。

1.3 节点调整和约束优化

胡事民教授在他《Approximate merging of B-Spline curves via knot adjustment and constrained optimization》一文里提到：

给定两个相邻 k 阶 B 样条曲线 $P(t), R(s)$, 他们对应的节点向量 $T = \{t_0, t_1, \cdots, t_k, \cdots, t_n, \cdots, t_{n+k}\}$ 和 $S = \{s_0, s_1, \cdots, s_k, \cdots, s_m, \cdots, s_{m+k}\}$, 控制点分别是 $P_i(i = 0, 1, \cdots, n)$ 和 $R_j(j = 0, 1, \cdots, m)$。找到一个 k 阶 B 样条曲线 $F(u)$, 控制点和节点向量分别是 $F_i(i = 0, 1, \cdots, n+m-k+2)$ 和 $U = \{t_0, t_1, \cdots, t_k, \cdots, t_n, t_{n+1} = s'_{k-1}, s'_k, \cdots, s'_{m+k}\}, s'_i = f(s_i)$ 对于一些线性函数 f,$d(F, F^-)$ 和 $F(u)$ 之间的合适的距离函数最小化。其中:

$$\bar{F}(u) = \begin{cases} P(u), & t_{k-1} \le u \le t_{n+1} \\ R(f^{-1}(u)), & s'_{k-1} \le u \le s'_{m+1} \end{cases}$$

方法的基本思想是首先找到的两个 B-样条曲线的精确合并的条件，扰乱曲线的约束优化的控制点来满足精确合并的条件。然后利用提出的节点调整法将生成的曲线参数化。这样的节点调整需要有效地制造一个没有多余节点的合并的 B 样条曲线。

首先推出 $P(t)$ 和 $R(s)$ 合并的条件。为了在 $P(t_{n+1}) = R(s_{k-1})$ 处合并，需要 $P^{(l)}(t_{n+1}) = R^{(l)}(s_{k-1}), l = 0, 1, \cdots, k-2$, 例如：

$$\sum_{i=n-k+1}^{n-l} P_i^l N_{i+1,k-l}^T(t_{n+1}) = \sum_{i=0}^{k-1-l} R_i^l N_{i+1,k-l}^S(s_{k-1}), l = 0, 1, \cdots, k-2$$

其中 $N_{i,k}^T N_{i,k}^S(s)$ 是定义在 T 和 S 处的 B 样条基函数。P_i^l 和 R_i^l 可以被写作 $\sum_{j=n-k+1}^{n} a_{ij}^{(l)} P_j$ 和 $\sum_{j=0}^{k-1} b_{ij}^{(l)} R_j$，其中 $a_{ij}^{(l)}, b_{ij}^{(l)}$ 可以通过算式下面算式得到。

$a_{ij}^{(l)}, n-k+1 \le i \le n-l, n-k+1 \le j \le n, 0 \le l \le k-2$,

$b_{ij}^{(l)}, 0 \le i \le k-1-l, 0 \le j \le k-1, 0 \le l \le k-2$.

1. 对于 $n-k+1 \le j \le n, p_0, p_1, \cdots, p_n$ 是标量，有

$$p_i = \delta_{ij} = \begin{cases} 0, & i \ne j \\ 1, & i = j \end{cases}$$

$$p_i^l = \begin{cases} p_i, & l = 0 \\ \frac{k-l}{t_{i+k}-t_{i+l}}(p_{i+1}^{l-1} - p_i^{l-1}), & l > 0 \end{cases}$$

2.$a_{ij}^{(l)} = p_i^l, n-k+1 \le i \le n-l, 0 \le l \le k-2.$
3. 同理可得 $b_{ij}^{(l)} = p_i^l, 0 \le i \le k-1-l, 0 \le l \le k-2.$
合并条件可以写作：

$$\sum_{j=n-k+1}^{n}(\sum_{i=n-k+1}^{n-l} a_{ij}^{(l)} N_{i+1,k-l}^{T}(t_{n+1}))P_j - \sum_{j=0}^{k-1}(\sum_{i=0}^{k-1-l} b_{ij}^{(l)} N_{i+1,k-1}^{S}(s_{k-1}))R_j = 0, l = 0, 1, \cdots, k-2.$$

该文提出了一种合并两个 B 样条曲线的近似合并。推导出曲线精确的合并条件。该曲线是通过约束优化修改，以满足精确融合条件。借助节点调整算法，合并后的曲线的控制点可以直接从这些曲线的确切满足合并条件下得到。生成的合并曲线没有多余的节点。在文中用了离散系数规范，但与“平方差不可分割规范”合并也是有可能的。将来，可以进一步调查多个 B 样条曲线和曲面的合并。

与胡教授的节点调整算法不同，借助 B 样条的节点向量，我们采取了插入节点算法。即将 n 次曲线的节点向量进行 n 次均匀采样，然后再对样点进行 B 样条逼近，这样就得到了合并后的 B 样条曲线。下一章主要讲采样后的核心算法：B 样条曲线逼近。

第二章　B 样条曲线逼近

2.1 二次和三次 B 样条曲线

1. 二次 B 样条曲线

$$s_i(t)=\frac{1}{2}[1 \quad t \quad t^2]\begin{bmatrix}1 & 1 & 0\\ -2 & 2 & 0\\ 1 & -2 & 1\end{bmatrix}\begin{bmatrix}\mathbf{P}_{i-2}\\ \mathbf{P}_{i-1}\\ \mathbf{P}_i\end{bmatrix}, t\in[0,1]; i=2,3,\cdots,n$$

二次 B 样条曲线 $S(t)$ 也称二次自由曲线，它将一次 B 样条曲线按折线段的走向加工成外形光滑的二次参数曲线，并作局部修正。$S(t)$ 的第 i 段曲线段的几何特征：

a. $s_i(0)=\frac{1}{2}(\mathbf{P}_{i-2}+\mathbf{P}_{i-1})$,

b. $s_i(1)=\frac{1}{2}(\mathbf{P}_{i-1}+\mathbf{P}_i)$,

c. $s_i\frac{1}{2}=\frac{1}{8}(\mathbf{P}_i-2\mathbf{P}_{i-1}+\mathbf{P}_{i-2})+\frac{1}{2}(\mathbf{P}_{i-1}-\mathbf{P}_{i-2})+\frac{1}{2}(\mathbf{P}_{i-1}+\mathbf{P}_{i-2})=\frac{1}{2}(\frac{s_i(1)+s_i(0)}{2}+\mathbf{P}_{i-1})$,

d. $s_i'(0)=\mathbf{P}_{i-1}-\mathbf{P}_{i-2}$,

e. $s_i'(1)=\mathbf{P}_i-\mathbf{P}_{i-1}$,

f. $s_i'(\frac{1}{2})=\frac{1}{2}(\mathbf{P}_i-\mathbf{P}_{i-2})$.

2. 三次 B 样条曲线

$$s_i(t)=\frac{1}{6}[1 \quad t \quad t^2 \quad t^3]\begin{bmatrix}1 & 4 & 1 & 0\\ -3 & 0 & 3 & 0\\ 3 & -6 & 3 & 0\\ -1 & 3 & -3 & 1\end{bmatrix}\begin{bmatrix}\mathbf{P}_{i-3}\\ \mathbf{P}_{i-2}\\ \mathbf{P}_{i-1}\\ \mathbf{P}_i\end{bmatrix}, t\in[0,1]$$

分别记线段 $\overline{\mathbf{P}_{i-3}\mathbf{P}_{i-1}}, \overline{\mathbf{P}_{i-2}\mathbf{P}_i}$ 的中点为 $\mathbf{M}_{i-2}, \mathbf{M}_{i-1}$, 则三次 B 样条曲线第 i 段曲线段有以下几何特征：

a. $s_i(0)=\frac{1}{6}(\mathbf{P}_{i-3}+4\mathbf{P}_{i-2}+\mathbf{P}_{i-1})=\frac{1}{3}\mathbf{M}_{i-2}+\frac{2}{3}\mathbf{P}_{i-2}$,

b. $s_i(1)=\frac{1}{3}\mathbf{M}_{i-1}+\frac{2}{3}\mathbf{P}_{i-1}$,

c. $s'(0)=\frac{1}{2}(\mathbf{P}_{i-1}-\mathbf{P}_{i-3})$,

d. $s'(1)=\frac{1}{2}(\mathbf{P}_i-\mathbf{P}_{i-2})$,

e. $s''(0)=\mathbf{P}_{i-1}-2\mathbf{P}_{i-2}+\mathbf{P}_{i-3}$,

f. $s''(1)=\mathbf{P}_i-2\mathbf{P}_{i-1}+\mathbf{P}_{i-2}$.

2.2 B 样条曲线算法

1.deBoor 算法

给定控制顶点 $\mathbf{d}_i(i=0,1,\cdots,n)$, 次数 k 以及确定的节点向量 $\mathbf{U}=[u_0,u_1,\cdots,u_{n+k+1}]$ 后，就定义了一条 k 次 B 样条曲线。如果给出定义域内一参数值 $u\in[u_i,u_{i+1}]\subset[u_k,u_{n+1}]$, 欲计算该 B 样条曲线上对应的一点 $p(u)$, 可采用 deBoor 算法：

$$\mathbf{p}(u)=\sum_{j=0}^{n}N_{j,k}(u)\cdot\mathbf{d}_j=\sum_{j=i-k}^{i-l}N_{j,k-l}(u)\cdot\mathbf{d}_j^l=\cdots=\mathbf{d}_{i-k}^k,u\in[u_i,u_{i+1}]\subset[u_k,u_{n+1}],$$

$$\mathbf{d}_j^l=\begin{cases}\mathbf{d}_j, & l=0,\\ (1-a_j^l)\mathbf{d}_j^{l-1}+a_j^l\mathbf{d}_{j+1}^{l-1}, & j=i-k,i-k+1,\cdots,i-l;l=1,2,\cdots,k,\\ a_j^l=\frac{u-u_{j+l}}{u_{j+k+1}-u_{j+l}}.\end{cases}$$

从递推公式看，用参数 u 经过 k 级递推得到最后一个中间顶点 $\mathbf{d}_{i-k}^k$, 就是曲线上一点 **p**(u)。利用 deBoor 算法计算 B 样条曲线上的点，可以避免先进行 B 样条基的计算，从而减少了计算量。

2. 节点插入算法

插入一个节点: 记已给一条 k 次B样条曲线 $\mathbf{p}(u)=\sum_{j=0}^{n}N_{j,k}(u)\cdot\mathbf{d}_j$, 其中B样条基由节点矢量 $\mathbf{U}=[u_0,u_1,\cdots,u_{n+k+1}]$ 完全决定。现在对定义域某个节点区间内插入一个节点 $u\in[u_i,u_{i+1}]\subset[u_k,u_{n+1}]$, 于是得到新的节点向量 $\overline{\mathbf{U}}=[u_0,u_1,\cdots,u_i,u,u_{i+1},\cdots,u_{n+k+1}]$, 重新编号后成为 $\overline{\mathbf{U}}=[\overline{u_0},\overline{u_1},\cdots,\overline{u_i},\overline{u_{i+1}},\overline{u_{i+2}},\cdots,\overline{u_{n+k+2}}]$。这些新节点向量 $\overline{\mathbf{U}}$ 决定了一组新的 B 样条基 $\overline{N}_{i,k}(u)(i=0,1,\cdots,n+1)$。原样条曲线就可以用这组新的 B 样条基与未知新顶点 $\mathbf{d}_i^1(i=0,1,\cdots,n+1)$ 表出 $\mathbf{p}(u)=\sum_{j=0}^{n+1}\overline{N}_{j,k}(u)\cdot\overline{\mathbf{d}_j}$。控制顶点增加一个，曲线的形状和连续性均保持不变。

重复插入同一节点: 把同一节点 u 重复插入 l 次，即插入重节点。假定该节点在老节点向量中可能已经具有重复度 r, 即 $u=u_i=u_{i-1}=\cdots=u_{i-r+1}$, 则应使 $r+l\le k$。这时将有新节点向量 $\overline{\mathbf{U}}=[\overline{u_0},\overline{u_1},\cdots,\overline{u_{n+k+l+1}}]$. 决定一组新 B 样条基 $\overline{N}_{j,k}(u)(j=0,1,\cdots,n+l)$。将原曲线用这组新基与未知新顶点 $\overline{\mathbf{d}_j}(j=0,1,\cdots,n+l)$ 表示 $\mathbf{p}(u)=\sum_{j=0}^{n+l}\overline{N}_{j,k}(u)\cdot\overline{\mathbf{d}_j}$。重复插入同一节点 l 次的算法实质上就是执行求 B 样条曲线一点的 deBoor 算法 l 级递推。

这些新顶点可由 $k-r+1$ 个顶点 $\mathbf{d}_{i-k},\mathbf{d}_{i-k+1},\cdots,\mathbf{d}_{i-r}$ 按如下递推公式执行 l 级递推得到:

$$\mathbf{d}_j^s=\begin{cases}\mathbf{d}_j, & s=0,\\ (1-\alpha_j)\mathbf{d}_j^{s-1}+\alpha_j\mathbf{d}_{j+1}^{s-1}, & s=1,2,\cdots,l;j=i-k,\cdots,i-r-l.\end{cases}$$

其中 $\alpha=\frac{u-u_{j+s}}{u_{j+k+l}-u_{j+s}}$。

3. 节点插入结论

a. 插入节点是一个局部过程。对 k 次 B 样条曲线插入一个节点仅与 $k-r+1$ 个老顶点

有关，且这些老顶点中仅除首末顶点外的其余 $k-r-1$ 个被新顶点所替代。其他顶点不受影响。

b. 当插入多个节点，最后结果与插入的先后次序无关。

c. 插入节点将改变 B 样条基在所插入节点处的可微性，每插入一次，可微性降一阶。

d. 插入节点前后的新老 B 样条控制多边形都定义同一条 B 样条曲线。在插入节点处，曲线的参数连续性保持不变。

e. 当插入节点无限加密时，其控制多边形序列将收敛到所定义的 B 样条曲线。其收敛速度要比 Bézier 曲线的升阶和 B 样条曲线的升阶的收敛速度快得多。

4.B 样条曲线全局插值

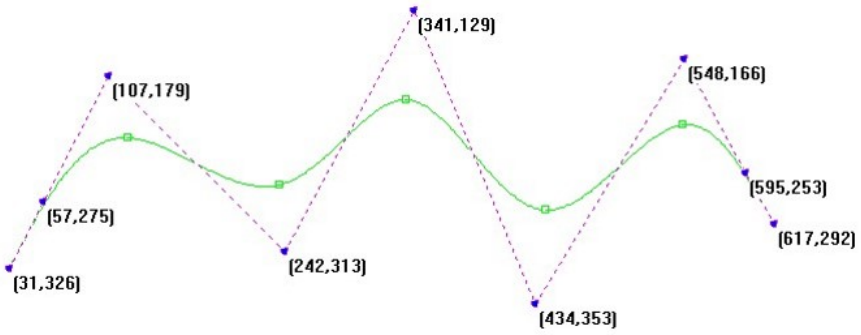

图 2.1 B 样条曲线全局插值

给定一组点 $\{\mathbf{Q}_k\}, k=0,\cdots,m$, 希望找到一条 p 次 B 样条曲线插值这些点，其首末端点分别与首末数据点一致，使数据点分别与曲线定义域内的节点一一对应，即 $\mathbf{Q}_i$ 点有节点值 $u_{k+1}(i=0,1,\cdots,m)$。B 样条插值曲线有 n 个控制顶点 $\mathbf{P}_i(i=0,1,\cdots,n)$ 与节点向量 $\mathbf{U}=\{u_0,\cdots,u_{n+p+1}\}$, 其中 $n=m+p-1$。

2.3 B 样条曲线逼近

对给定的大量数据，做插值曲线将产生不必要的被动。此时，需要求逼近这些数据点的最小二乘 B 样条曲线。插值不存在曲线误差问题。而逼近，曲线误差 E 与被拟合的数据点一起给出。通常，预先不知道需要用多少控制顶点才能满足误差精度。因此，逼近一般是一个迭代过程。利用 B 样条曲线对数据点做整体逼近主要有两种算法，两种算法的中心问题都是怎样给定一个控制顶点数目。

算法一：
(1) 由最少的或一个小数目的控制顶点开始；
(2) 用整体拟合方法对数据点拟合一条逼近曲线；
(3) 检查逼近曲线对数据点的误差，如果误差处处小于给定误差界 E，返回；否则，增加控制顶点的数目，转到步骤 (2)。

算法二：
(1) 由最大的或一个大数目的控制顶点开始，以致第一次逼近就满足精度误差 E；
(2) 用整体拟合方法对数据拟合一条逼近曲线；
(3) 检查逼近曲线对数据点的误差是否处处满足精度误差 E；
(4) 如果不满足且步骤 (3) 未执行过则转到步骤 (1), 如 (3) 已执过，返回上次结果；否则减少控制顶点的数目，转到步骤 (2)。

最小二乘逼近：

需要预知数据点的参数值 $\widetilde{u}_i$ 和节点向量 **U**。设给定 $m+1$ 个数据点 $\mathbf{q}_0, \mathbf{q}_1, \cdots, \mathbf{q}_m (m > n)$, 逼近曲线次数 $k \geq 1$, 试图寻找一条 k 次 B 样条曲线 $\mathbf{p}(u) = \sum_{j=0}^{n} N_{j,k}(u) \cdot \mathbf{d}_j, u \in [0,1]$ 满足 $\mathbf{q}_0 = \mathbf{p}(0), \mathbf{q}_m = \mathbf{p}(1)$; 其余数据点 $\mathbf{q}_i (i = 1, 2, \cdots, m-1)$ 在最小二乘意义上被逼近, 即目标函数 $f = \sum_{i=1}^{m-1} [\mathbf{q}_i - \mathbf{p}(\widetilde{u}_i)]^2$ 是关于 $n-1$ 个控制顶点 $\mathbf{d}_j (j = 1, 2, \cdots, n-1)$ 的一个最小值。参数值 $\widetilde{u}_i (i = 0, 1, \cdots, m)$ 可以由前面的累计弦长参数化决定。现在来确定节点向量 $\mathbf{U} = [u_0, u_1, \cdots, u_{n+k+1}]$ 以决定样条基函数。一般来说，需要满足端点插值性，所以 $u_0 = u_1 = \cdots = u_k = 0, u_{n+1} = u_{n+2} = \cdots = u_{n+k+1} = 1$。中间 n-k 个节点的选取应当反映数据点参数值 $\widetilde{u}_i$ 的分布情况，可按如下决定:

设 c 是一个正实数,$i = int(c)$。令
$\widetilde{u}_i$ 的分布情况，可按如下决定:

$c = \frac{m+1}{n-k+1}$
则定义域的内节点为:

$i = int(jc), \alpha = jc - i,$

$u_{k+j} = (1-\alpha)\widetilde{u}_{i-1} + \alpha \widetilde{u}_i, j = 1, 2, \cdots, n-k.$
按如上决定的内节点保证每个节点区间包含至少一个 $\widetilde{u}_i$。

由于逼近，不要求曲线精确通过数据点 $\mathbf{q}_i (i = 1, 2, \cdots, m-1)$, 且 $\mathbf{p}(\widetilde{u}_i)$ 不是在曲线上与 $\mathbf{q}_i$ 的最近点。
设 $\mathbf{r}_i = \mathbf{q}_i - \mathbf{q}_0 N_{0,k}(\widetilde{u}_i) - \mathbf{q}_m N_{n,k}, i = 1, 2, \cdots, m-1$, 那么目标函数有 $f = \sum_{i=1}^{m-1} [\mathbf{q}_i - \mathbf{p}(\widetilde{u}_i)]^2 = \sum_{i=1}^{m-1} [\mathbf{r}_i - \sum_{j=1}^{n-1} \mathbf{d}_j \cdot N_{j,k}(\widetilde{u}_i)]^2$.
使用最小二乘法使其最小：

$(N^T N) D = N^T R$

$$N=\begin{bmatrix} N_{1,k}(\widetilde{u}_1) & \cdots & N_{n-1,k}(\widetilde{u}_1) \\ \vdots & \ddots & \vdots \\ N_{1,k}(\widetilde{u}_{m-1}) & \cdots & N_{n-1,k}(\widetilde{u}_{m-1}) \end{bmatrix}, D=\begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_{n-1} \end{bmatrix}, R=\begin{bmatrix} \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_{m-1} \end{bmatrix}$$

误差检查最大距离：$\max\limits_{0\le i\le m}|\mathbf{q}_i-\mathbf{p}(\widetilde{u}_i)|$ 或 $\max\limits_{0\le i\le m}(\min\limits_{0\le u\le 1}|\mathbf{q}_i-\mathbf{p}(u)|)$。后者称为最大范数距离。一般来说，后者是用户所要求的。

在算法一中，逼近如下：由最少 $k+1$ 个控制顶点开始，拟合得一逼近曲线，然后用范数使检验是否小于给定误差 E。对每个节点区间，维持一个记录，以表明是否收敛。如果对所有的 i，都满足，则该节点区间已经收敛。在每次拟合与误差检查之后，在每个非收敛节点区间的中点插入一个节点，相应就增加一个顶点。如果生成奇异方程组也需要处理。

第三章　B 样条曲线合并

B 样条曲线进行合并，本质上是一种给定误差进行逼近的过程。对于逼近，一定会有逼近的点，即我们需要对所给的两条或多条曲线进行采取样点，接下来就可以对样点进行逼近了。逼近后的曲线，在每个点都不一定是完全重合的，为了保持曲线的可靠性，需要对一前一后的两个端点位置进行赋值校正，这就是 B 样条曲线合并的全过程。

3.1 节点采样

由于多条曲线与两条曲线合并的方法完全一样，本文仅介绍两条给定控制顶点和节点向量的的 B 样条曲线进行合并。因为 B 样条曲线本质上是分段多项式,n 次曲线采样 n 个点, 插值一定可以重建，所以将每个节点区间进行 n 等分，在节点向量中插入这些等分节点。接下来我将通过一个实际的例子来讲述整个过程。
首先，我们给定两条 B 样条曲线（图 3.1）：

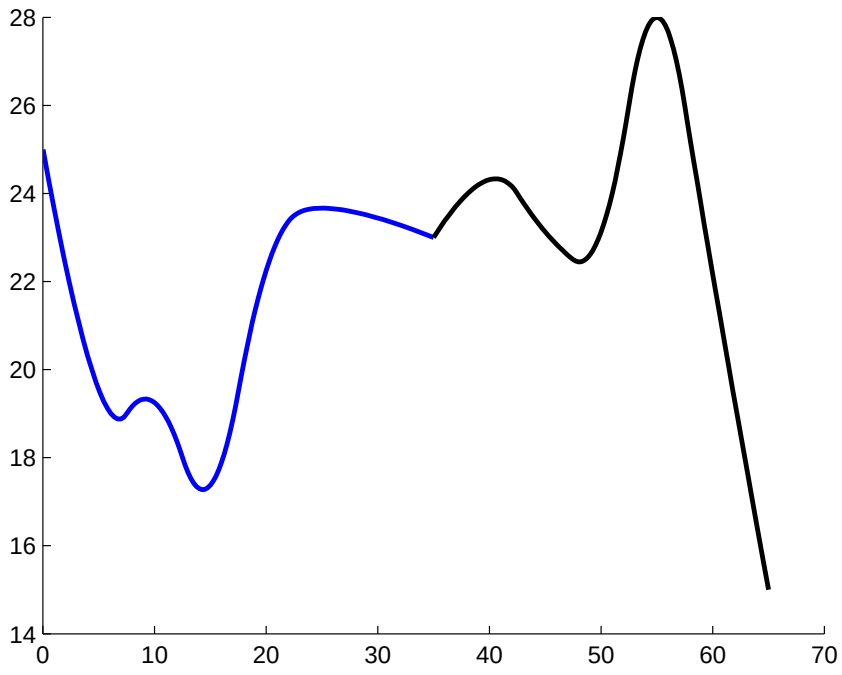

图 3.1 原始的两条 B 样条曲线

两条曲线的控制顶点分别是 $A1, A2$：

$$A1 = \begin{bmatrix} x: & 0 & 5 & 10 & 15 & 20 & 25 & 35 \\ y: & 25 & 18 & 20 & 16 & 23 & 24 & 23 \end{bmatrix}$$

$$A2 = \begin{bmatrix} x: & 35 & 40 & 45 & 50 & 55 & 60 & 65 \\ y: & 23 & 25 & 23 & 22 & 30 & 22 & 15 \end{bmatrix}$$

在图 3.2 上显示控制顶点：

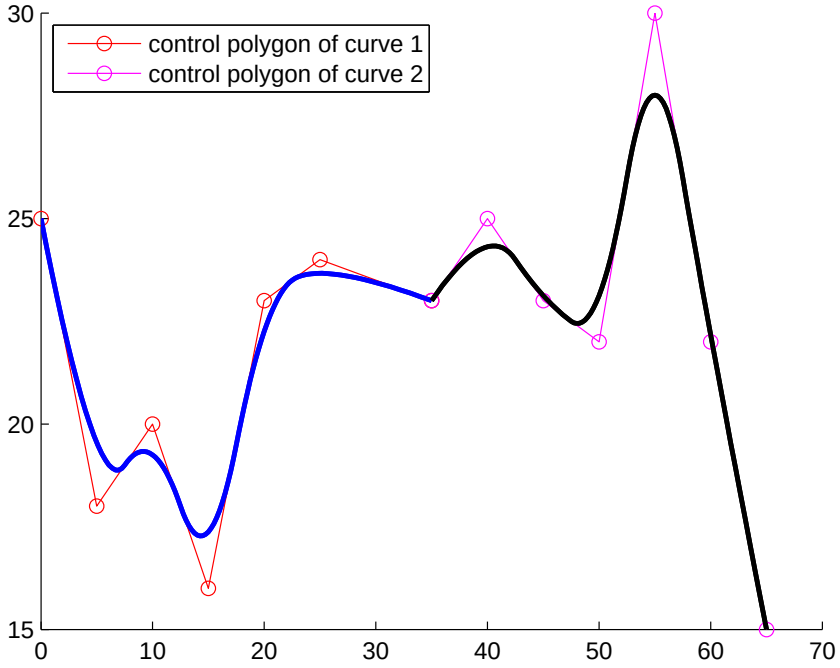

图 3.2 两条 B 样条曲线及其控制多边形

它们的节点向量分别是：

knots1=[0 0 0 1/5 2/5 3/5 4/5 1 1 1];

knots2=[0 0 0 1/5 2/5 3/5 4/5 1 1 1];

由 length(knots$i$)-length(Ai)=3, (i=1,2) 得到两条 B 样条曲线均为 2 次 B 样条曲线。分别对它们的节点向量进行采样，2 次曲线采样 2 个点, 然后将每个节点区间进行 2 等分，在节点向量中插入这些等分节点，得到新的节点向量：

knots13=[0 1/10 1/5 3/10 2/5 1/2 3/5 7/10 4/5 9/10 1];

knots23=[0 1/10 1/5 3/10 2/5 1/2 3/5 7/10 4/5 9/10 1];

这样我们可以得到如图 3.3 所示的采样点 (图中绿色和黄色的 * 标记), 样点坐标为 Ay=

x	y	x	y
0	25.0000	35.0000	23.0000
4.3750	20.0000	39.3750	24.2500
7.5000	19.0000	42.5000	24.0000
10.0000	19.2500	45.0000	23.1250
12.5000	18.0000	47.5000	22.5000
15.0000	17.3750	50.0000	23.1250
17.5000	19.5000	52.5000	26.0000
20.0000	22.2500	55.0000	28.0000
22.5000	23.5000	57.5000	26.0000
26.8750	23.6250	60.6250	21.2500
35.0000	23.0000	65.0000	15.0000

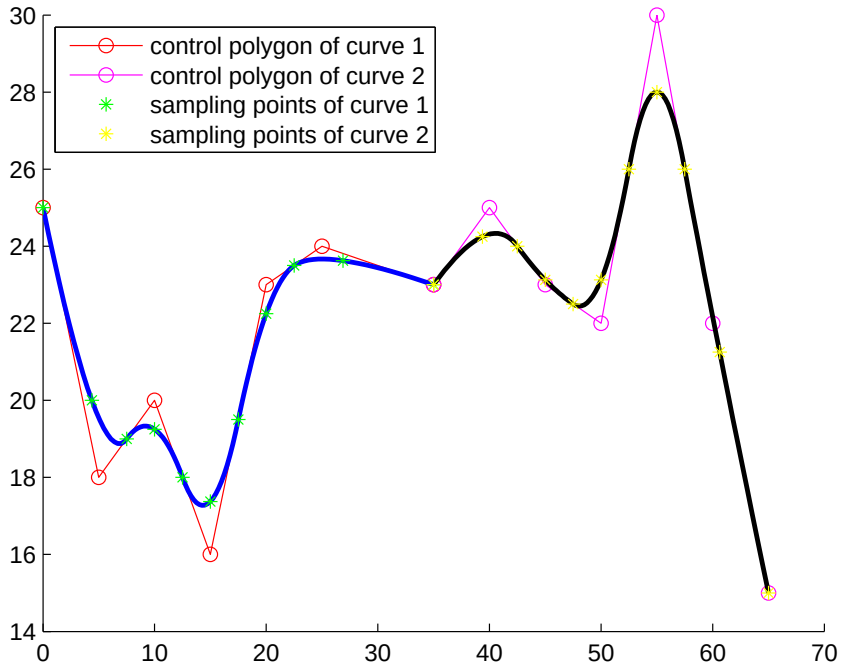

图 3.3 两条 B 样条曲线均匀采样点

3.2 样点逼近

将所选样点进行逼近，接下来的所有逼近的误差都设为 $e = 1 \times 10^{-6}$，逼近后的曲线设置为 3 次 B 样条曲线，即节点数 − 控制点数 =4。对当前两条曲线，我们先设置合并后的 B 样条曲线的控制点为 10 个，则由逼近程序算出来的节点向量和控制顶点如下：

节点向量：mknots110= [0, 0, 0, 0, 0.14487501066996, 0.28387813840892, 0.41915984868222, 0.55560490261852, 0.69659652559125, 0.84422690561212, 1, 1, 1, 1];

控制顶点：mX110=

x	y
0.05091956182936	24.913173534056
0.9098063460802	22.535402011053
9.8722390147356	15.801318665629
17.937236533338	20.544673211937
27.545724993608	24.865167607182
39.087853421893	23.638956371753
49.896502150356	21.760633610066
58.430035167441	32.326891446225
62.553791438335	14.861575249215
64.992966899276	15.124444695882

与原来的曲线差距很大，这是因为控制点数取得太少，接下来取控制点为 15 个，则由逼近程序算出来的节点向量和控制顶点如下：

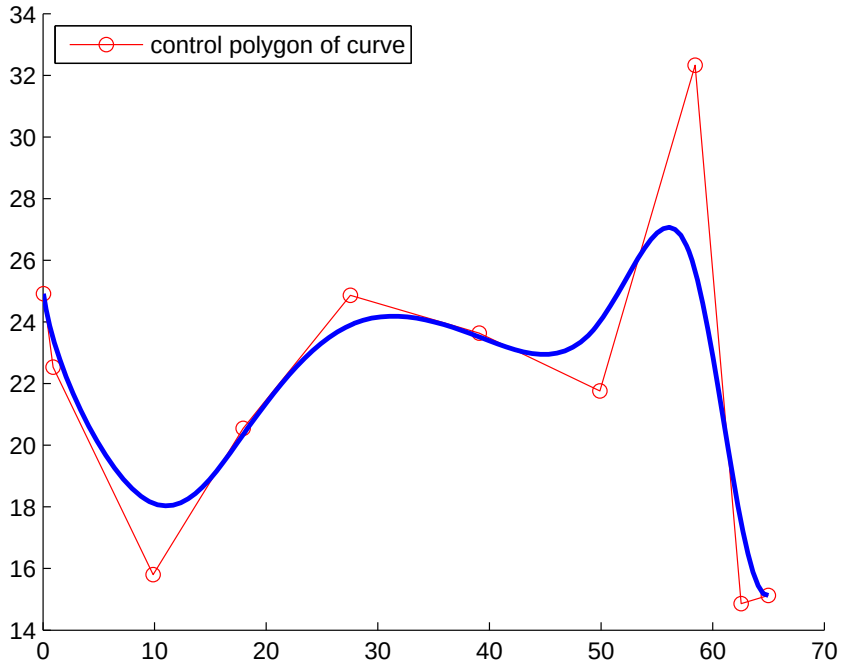

图 3.4　10 控制点样点逼近

节点向量：mknots115= [0, 0, 0, 0, 0.087189359065146, 0.16892500156618, 0.2480550388634, 0.32669189642223, 0.40591390164656, 0.4869439497054, 0.56883644689745, 0.65046692711676, 0.73331603656949, 0.81822567526098, 0.90634304036331, 1, 1, 1, 1];

控制顶点：mX115=

x	y
0.0005158945627477	25.001897495802
1.2355231024696	23.397097284274
4.1239088622059	18.641117960498
10.843366047414	19.338508884231
16.689342080667	16.243566904455
20.334827173427	24.302131532171
27.081148714731	23.989182587336
33.578793945732	21.49142739603
39.520074930174	25.946675696897
46.424920998368	21.416364487809
51.820067373303	24.365513852045
56.677563640819	30.562129688326
61.189947152537	17.513079644156
63.167849794748	19.782033886869
65.000403814646	14.998891063483

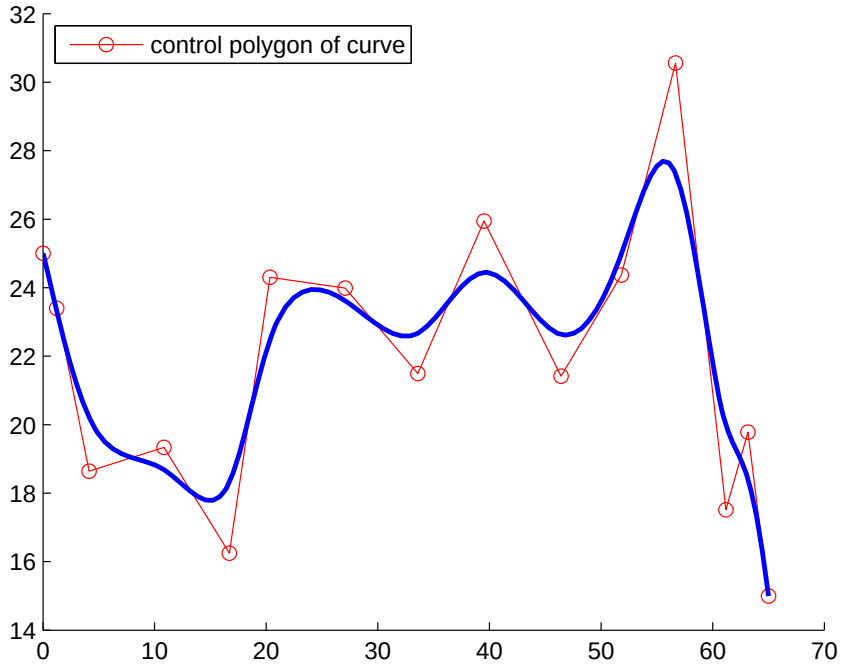

图 3.5　15 控制点样点逼近

将两端边界值进行优化，即将两端点的值赋为原始曲线的值，具体做法是将端点的控制顶点赋为原始曲线控制顶点的端点值，然后去除控制多边形和采样点得到结果如图 3.6 所示：

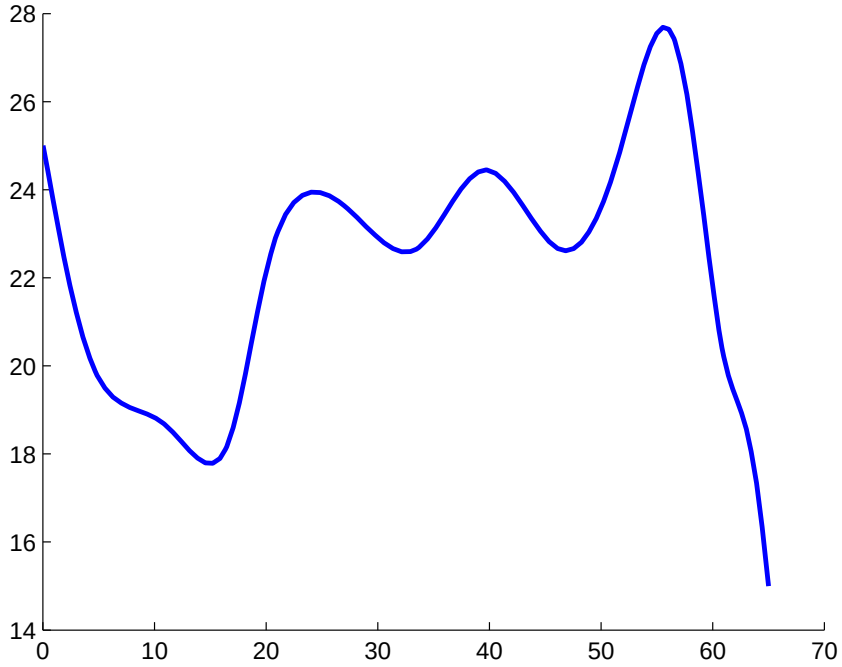

图 3.6 合并后的 B 样条曲线

自此，我们得到了两条 B 样条曲线合并后新的 B 样条曲线。

3.3 更多实例

给定的两条将要合并的 B 样条曲线（图 3.7）：
两条曲线的控制顶点分别是 $A3, A4$：

$$A3 = \begin{bmatrix} x: & -21 & -10 & 0 & 20 & 25 & 40 & 50 & 60 \\ y: & 80 & 90 & 70 & 66 & 84 & 96 & 80 & 95 \end{bmatrix}$$

$$A4 = \begin{bmatrix} x: & 60 & 70 & 85 & 90 & 110 & 120 & 135 \\ y: & 95 & 85 & 100 & 110 & 104 & 89 & 90 \end{bmatrix}$$

它们的节点向量分别是：
knots3=[0 0 0 1/6 1/3 1/2 2/3 5/6 1 1 1];
knots4=[0 0 0 0 1/4 2/4 3/4 1 1 1 1];

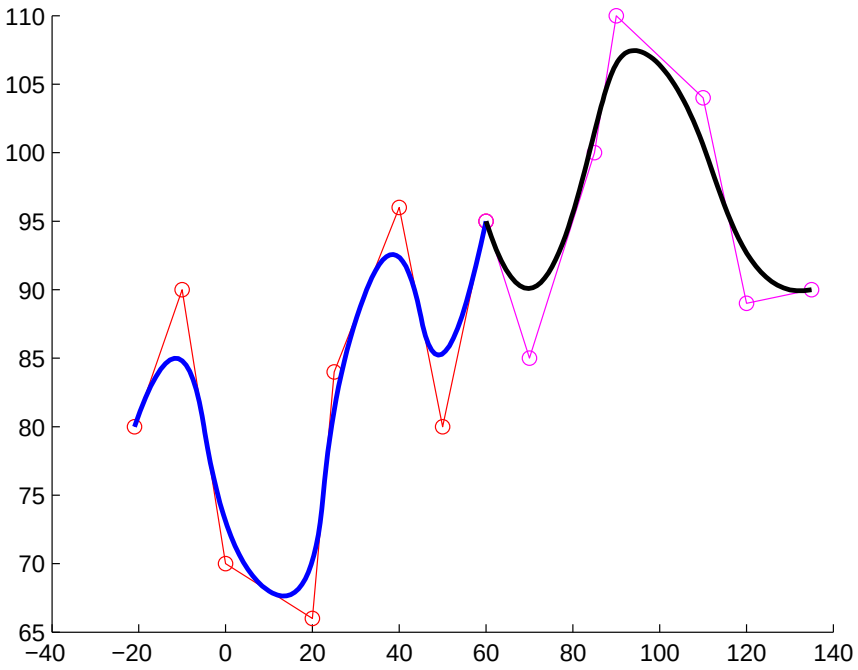

图 3.7 原始的两条 B 样条曲线

由 length(knots3)-length(A3)=3 得到第一条 B 样条曲线为 2 次 B 样条曲线，由 length(knots4)-length(A4)=4 得到第二条 B 样条曲线为 3 次 B 样条曲线。分别对它们的节点向量进行采样，2 次曲线采样 2 个点, 然后将每个节点区间进行 2 等分，3 次曲线采样 3 个点，然后将每个节点区间进行 3 等分，在节点向量中插入这些等分节点，得到新的节点向量：

knots33=[0 1/12 1/6 1/4 1/3 5/12 1/2 7/12 2/3 3/4 5/6 11/12 1];

knots44=[0 1/12 1/6 1/4 1/3 5/12 1/2 7/12 2/3 3/4 5/6 11/12 1];

这样我们可以得到如图 3.8 所示的采样点 (图中绿色和黄色的 * 标记), 样点坐标为 Ay:

x	y	x	y
-21.0000	80.0000	60.0000	95.0000
-11.5000	85.0000	69.1512	90.1080
-5.0000	80.0000	76.5432	92.5309
1.2500	72.0000	82.0833	97.9167
10.0000	68.0000	85.8951	102.6173
18.1250	68.7500	88.9660	105.7994
22.5000	75.0000	92.5000	107.3333
26.2500	83.2500	97.3765	107.1080
32.5000	90.0000	103.1790	105.0864
39.3750	92.5000	109.1667	101.2500
45.0000	88.0000	115.1235	96.0000
51.2500	85.7500	122.9321	91.4167
60.0000	95.0000	135.0000	90.0000

将所选样点进行逼近，我们先设置合并后的 B 样条曲线的控制点为 20 个，则由逼近程序算出来的节点向量和控制顶点如下：

节点向量：mknots110= [0, 0, 0, 0, 0.14487501066996, 0.28387813840892, 0.41915984868222, 0.55560490261852, 0.69659652559125, 0.84422690561212, 1, 1, 1, 1];

控制顶点：mX110=

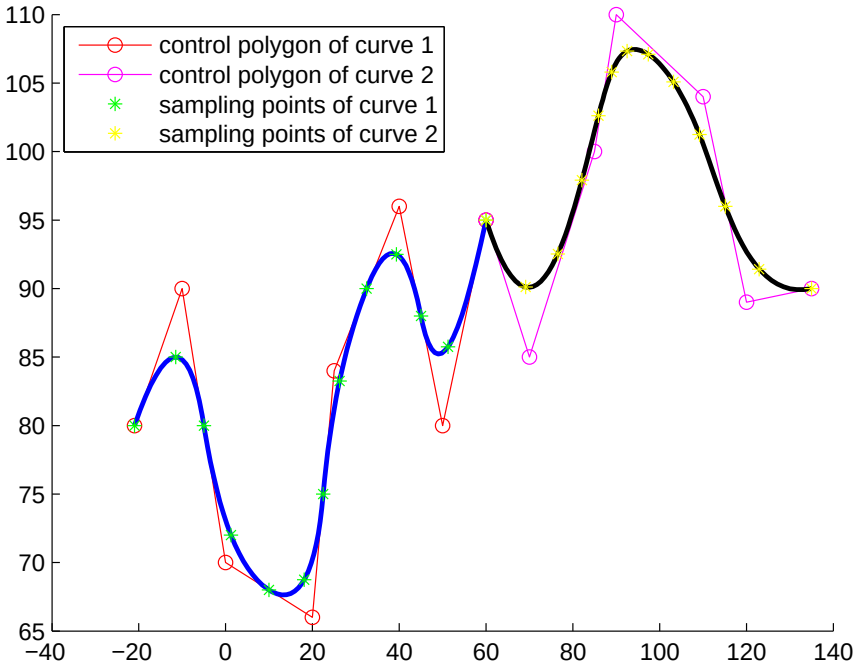

图 3.8 两条 B 样条曲线均匀采样点

x	y
0.05091956182936	24.913173534056
0.9098063460802	22.535402011053
9.8722390147356	15.801318665629
17.937236533338	20.544673211937
27.545724993608	24.865167607182
39.087853421893	23.638956371753
49.896502150356	21.760633610066
58.430035167441	32.326891446225
62.553791438335	14.861575249215
64.992966899276	15.124444695882

将两端边界值进行优化，然后去除控制多边形和采样点得到结果如图 3.9 所示：

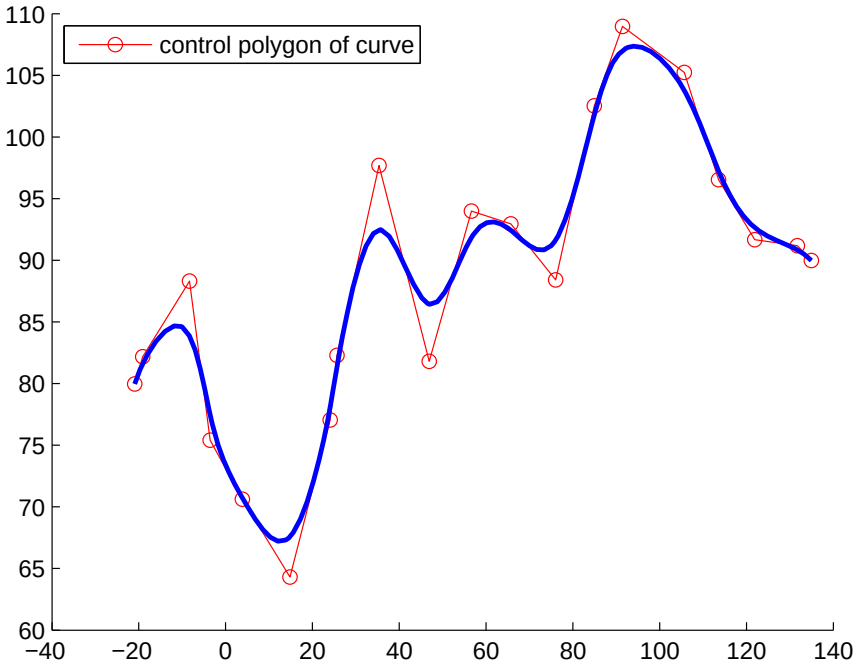

图 3.9 20 控制点样点逼近

第四章　总结与展望

正如前面所讲的那样，把多条 B 样条曲线合并成一条新的 B 样条曲线，首先要对各个曲线进行均匀或非均匀样点采取，然后再将这些样点进行整体逼近，最后再修复端点处的值，所得曲线即为合并后的新曲线。可以说，整个过程意思简介明了，思路清晰可见，难点在于样点的采取及 B 样条逼近算法的实现。

如引言所述，B 样条在 CAGD 通讯方面起着重要作用，能够减少传输数据量而保持数据的高度完整性。这条性质是及其可观的。我想，在日后的网络数据传输方面，B 样条或许能够扮演一个重要的角色，承担起大数据共享的巨大责任。

当然，如果 B 样条用于网络数据传输，势必将会给现今成熟安全的网络传输带来各种不可预料的问题。但是，我相信人类不是止步不前的种族，假如 B 样条真有引领新一代的互联网标准的潜力，疯狂的大神们不会介意靠上去的。届时，人们就可真正体验到飞一般的网络世界。

致　谢

大学美丽的时光已经灯火阑珊，在此我首先对我伟大的东南大学、我的父母、亲人们，我的老师和同学们表达我由衷的谢意。感谢我的家人对我大学四年学习的默默支持；感谢我的母校给了我在校园飞翔的机会，让我能继续学习和提高；感谢所有的老师和同学们四年来的关心和鼓励。老师们课堂上的激情洋溢，课堂下的谆谆教诲；同学们在学习中的认真热情，生活上的热心帮助，所有这些都让我充满了感动。这次毕业论文设计我得到了很多老师和同学的帮助，其中我的论文指导老师朱平老师对我的关心和支持尤为重要。每次遇到难题，我最先做的就是向朱老师寻求帮助，而朱老师总会不辞辛劳，不嫌弃我，然后一起商量解决的办法。

我做毕业设计的每个阶段，从选题到查阅资料，论文提纲的确定，中期论文的修改，后期论文格式调整等各个环节中都给予了我悉心的指导。这几个月以来，朱平老师不仅在学业上给我以精心指导，同时还在思想给我以无微不至的关怀，在此谨向朱老师致以诚挚的谢意和崇高的敬意。同时，本篇毕业论文的写作也得到了刘同学、侯同学等的热情帮助，他们提醒着我是该做什么的时候了。感谢在整个毕业设计期间和我密切合作的同学，和曾经在各个方面给予过我帮助的伙伴们，在此，我再一次真诚地向帮助过我的老师和同学表示感谢！

参考文献

1 CHEN Jun,WANG Guo-jin, *Approximate merging of B-spline curves and surfaces* Appl. Math. J. Chinese Univ. 2010, 25(4): 429-436.

2 Chiew-Lan Tai,Shi-Min Hu,Qi-Xing Hua, *Approximate merging of B-spline curves via knot adjustment and constrained optimization*Computer-Aided Design 35 (2003) 893–899.

3 杨雅迪, 秦新强等. *C-B 样条曲线的光顺逼近算法研究* [J]. 计算机工程与应用.205-207.

4 秦新强, 岳丽, 胡钢, 李凯. *带参数均匀 B 样条曲线的近似合并*上海交通大学学报.2010.1084-1088.

5 陈军. *曲线曲面的几何约束造型与近似合并* [D]; 浙江大学;2010 年.

6 周联. *曲线曲面造型中的三类几何逼近* [D]; 浙江大学;2010 年.

7 梁锡坤. *B 样条类曲线曲面理论及其应用研究* [D]; 合肥工业大学;2003 年.

8 徐惠霞. *B 样条多重乘积理论与有理曲线曲面多项式逼近技术的研究* [D]; 浙江大学;2008 年.

9 成敏. *曲线曲面的求值及降阶、等距变换的研究* [D]; 浙江大学;2003 年.

10 王志国. *曲线曲面形状修改和变形关键技术研究* [D]; 南京航空航天大学;2006 年.

附录 A　Matlab 程序

curves.m:

```
function [p1,p2]=curves(X1,X2,knots1,knots2,knots1s,knots2s)
x1=X1(1,:);x2=X2(1,:);
y1=X1(2,:);y2=X2(2,:);
hold on
l1 = plot(x1,y1,'r-o');
l2 = plot(x2,y2,'m-o');
sp1=spmak(knots1,X1);
sp2=spmak(knots2,X2);
fnplt(sp1,[knots1(1),knots1(10)],'b');
fnplt(sp2,[knots2(1),knots2(10)],'k');
p1=fnval(sp1,knots13);
p2=fnval(sp2,knots23);
l3 = plot(p1(1,:),p1(2,:),'*g');
l4 = plot(p2(1,:),p2(2,:),'*y');
legend([l1,l2,l3,l4],'Location','NorthWest','control polygon of curve 1','control polygon of curve 2','sampling points of curve 1','sampling points of curve 2');
end
```

main.m:

```
X11=load('data\data11.txt');
X12=load('data\data12.txt');
knots11=[0 0 0 1/5 2/5 3/5 4/5 1 1 1];
knots113=[0 1/10 1/5 3/10 2/5 1/2 3/5 7/10 4/5 9/10 1];
knots12=[0 0 0 1/5 2/5 3/5 4/5 1 1 1];
knots123=[0 1/10 1/5 3/10 2/5 1/2 3/5 7/10 4/5 9/10 1];
figure(1)
[p11,p12]=curves(X11,X12,knots11,knots12,knots113,knots123);
X21=load('data\data21.txt');
X22=load('data\data22.txt');
knots21=[0 0 0 1/6 1/3 1/2 2/3 5/6 1 1 1];
knots22=[0 0 0 0 1/4 2/4 3/4 1 1 1 1];
knots213=[0 1/12 1/6 1/4 1/3 5/12 1/2 7/12 2/3 3/4 5/6 11/12 11];
knots224=[0 1/12 1/6 1/4 1/3 5/12 1/2 7/12 2/3 3/4 5/6 11/12 1];
figure(2)
[p21,p22]=curves(X21,X22,knots21,knots22,knots213,knots224);
%%%%%%merge1
%%%%%%10points
figure(5110);
mknots110=load('merge\mknots110.txt');
mX110=load('merge\mX110.txt');
merge(mX110,mknots110);
%%%%%%15points
figure(5115)
mknots115=load('merge\mknots115.txt');
mX115=load('merge\mX115.txt');
merge(mX115,mknots115);
%%%%%%merge2
%%%%%%20points
figure(5220)
mknots220=load('merge\mknots220.txt');
mX220=load('merge\mX220.txt');
merge(mX220,mknots220);
```

merge.m:

```
function []=merge(mX,mknots)
x=mX(:,1);
y=mX(:,2);
hold on
plot(x,y,'r-o');
legend('Location','NorthWest','control polygon of curve');
sp=spmak(mknots,mX');
fnplt(sp,mknots,'b');
end
```