曲线拟合工具箱 CurveFitting Toolbox 使用说明

目录

1 工具箱简介

曲线拟合工具箱 (CurveFitting Toolbox，以下简称工具箱) 提供了一系列用于样条曲线拟合的函数，包括 pp 格式 (piecewice-polynomial form) 形式与 B 格式 (B form) 的样条函数的生成与后处理操作。工具箱的主要函数及函数间的联系如下图所示：

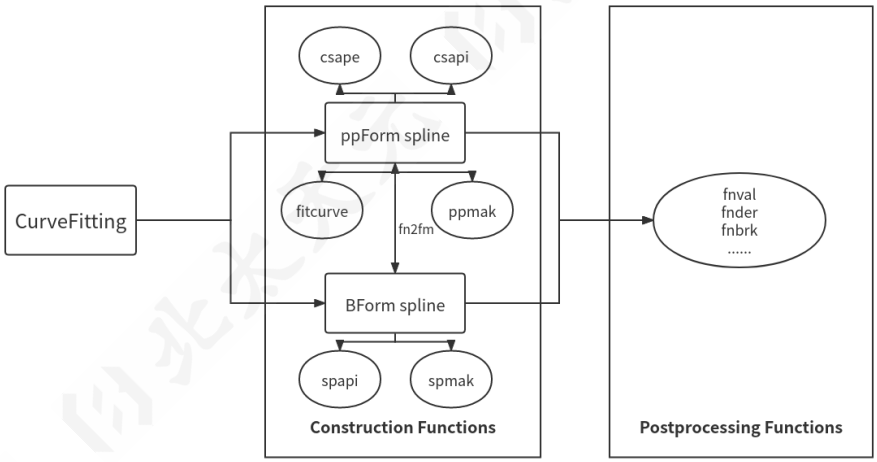


图 1: 工具箱整体框架

在工具箱中，我们为每个函数提供了使用帮助，可以通过在软件中使用 `help + 函数名` 来查看帮助。通过命令 `plugin_help("CurveFitting")`，我们可以看到工具箱中所有提供了帮助文档的函数：

```
插件 [CurveFitting] 提供的命令：
csapi     csape     ppmak     spapi     spmak     fitcurve  bspline   fnval     fnder     fnbrk     fn2fm
```

通过 `help csapi`，我们可以得到 `csapi` 函数的使用帮助。

```
>> help csapi

csapi 创建一个 ppForm 的三次样条曲线，其边界条件采用 not-a-knot 条件。目前仅支持一维情形。

  s = csapi(x,y) 创建一个三次样条曲线，x，y 为插值条件，x，y 必
  须为长度相同的一维向量。这里要求 x 已经完成单增排列。

  s 的结果是满足插值条件与 not-a-knot 边界条件的三次样条曲线，为 pp
  格式。
```

2 相关概念

在本节中，我们会对相关的数学概念做一个简单的介绍。对相关内容比较熟悉的读者可以跳过这一部分。

2.1 piecewice-polynomial 样条函数

定义 2.1 (piecewice-polynomial 样条函数). 给定非负整数 n, k 及一列分划 $[a, b]$ 的严格增序列 $\{x_i\}$,

$$a = x_1 < x_2 < \cdots < x_N = b,$$

称以下集合为 n 次 k 级光滑的 piecewice-polynomial 样条函数：

$$\mathbb{S}_n^k = \{s : s \in \mathcal{C}^k[a, b]; \forall i \in [1, N-1], s|_{[x_i, x_{i+1}]} \in \mathbb{P}_n\},$$

这里的序列 $\{x_i\}$ 被称为样条函数的插值节点。

定义 2.2 (三次样条函数). $\{x_i\}$ 上的三次样条函数 (cubic spline) 为 $\mathbb{S}_3^2$。

定义 2.3 (三次样条函数的边界类型). 我们对三次样条函数定义以下几种边界条件类型：

- complete 条件：$s'(f; a) = f'(a), s'(f; b) = f'(b)$；
- second 条件：$s''(f; a) = f''(a), s''(f; b) = f''(b)$；
- not-a-knot 条件：$s'''(f; x)$ 在 $x = x_2$ 和 $x = x_{N-1}$ 存在；
- periodic 条件：将 $s(f; b) = f(b)$ 替换为 $s(f; b) = s(f; a), s'(f; b) = s'(f; a), s''(f; b) = s''(f; a)$；
- mixed 条件：complete、second 与 not-a-knot 及其它边界条件的混合。

2.2 B 样条函数

定义 2.4 (B 样条). 设 $\{t_i\}$ 单调非减，则 B 样条 (B-splines) 被如下递归地定义：

$$B_i^{n+1}(x) = \frac{x - t_{i-1}}{t_{i+n} - t_{i-1}} B_i^n(x) + \frac{t_{i+n+1} - x}{t_{i+n+1} - t_i} B_{i+1}^n(x).$$

递归的起点是 0 次的 B 样条：

$$B_i^0(x) = \begin{cases} 1 & x \in (t_{i-1}, t_i], \\ 0 & \text{otherwise.} \end{cases}$$

例 2.5. 根据定义2.4，我们发现 1 次的 B 样条是所谓的 “帽子函数”：

$$B_i^1(x) = \begin{cases} \frac{x-t_{i-1}}{t_i-t_{i-1}} & x \in (t_{i-1}, t_i], \\ \frac{t_{i+1}-x}{t_{i+1}-t_i} & x \in (t_i, t_{i+1}], \\ 0 & \text{otherwise.} \end{cases}$$

定义 2.6 (B 样条节点)**.** 根据定义2.4,为了得到 n 次 B 样条 B_i^n,我们需要用到 $t_{i-1}, t_i, t_{i+1}, \cdots, t_{i+n}$，这些点被称为 B 样条的节点。

定义 2.7 (B 格式的样条函数)**.** 我们将一系列 n 次 B 样条的线性组合定义为次数为 n 的 B 格式的样条函数。

定理 2.8. 以下 B 样条是 $\mathbb{S}_n^{n-1}(t_1, t_2, \cdots, t_N)$ 的一组基：

$$B_{2-n}^n(x), B_{3-n}^n(x), \cdots, B_N^n(x).$$

3 函数功能说明及使用例

这一部分我们对工具箱中每个函数的使用方法进行详细说明，并给出具体用法的使用示例。我们用 pp 格式简单地表示 piecewice-polynomial 样条函数，用 B 格式表示 B 样条函数。值得一提的是，为了在软件中使用工具箱中的函数 (包括第 1 节中的 `help` 函数)，在使用前应该先调用命令 `plugin_help("CurveFitting")` 完成对工具箱的调用。

在正式开始进行说明前，有一点需要特别指出：在工具箱中，多项式的 “阶数”(Order) 的概念与我们通常理解的 “次数” 稍有不同，它表示的是多项式的系数的个数。例如，考虑三次多项式 $ax^3 + bx^2 + cx + d$，由于其有四个系数 (a, b, c, d)，在工具箱中我们称这一多项式的阶数为 4。我们限制多项式的阶数不超过 8。

3.1 `csapi`

`csapi` 函数根据给定的二维点列创建一个 pp 格式的三次样条曲线，其边界条件采用 not-a-knot 条件。该函数是 `csape` 函数的一种特殊情况。

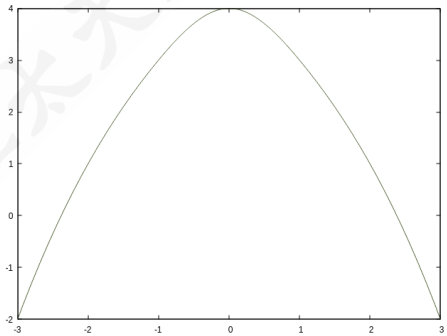
- `s = csapi(x,y)` 创建一个三次样条曲线，`x` 为插值节点，`y` 为节点上的值，`x`，`y` 必须为长度相同的一维向量，且要求 `x` 已经完成单增排列。

返回结果 `s` 是满足插值条件与 not-a-knot 边界条件的三次样条曲线，为 pp 格式。

例：在点列 $(-3, -2), (-2, 1), (-1, 3), (0, 4), (1, 3), (2, 1), (3, -2)$ 上进行三次样条插值，边界条件采用 not-a-knot 条件。

```
>> x = [-3,-2,-1,0,1,2,3];
>> y = [-2,1,3,4,3,1,-2];
>> s = csapi(x,y);
>> t = -3:0.01:3;plot(t,fnval(s,t));
```

在完成样条函数 s 的生成后，为了实现结果的可视化，我们调用了 fnval 函数。fnval 函数可以得到样条函数在指定点处的值，该函数的具体使用可以见3.8节。下面是以上命令的运行结果。



3.2 csape

csape 函数根据给定的二维点列与边界条件创建一个 pp 格式的三次样条曲线。

- s = csape(x,y) 创建一个三次样条曲线，使用 not-a-knot 边界条件。x 为插值节点，y 为节点上的值，x，y 必须为长度相同的一维向量，且要求 x 已经完成单增排列。等价于 s = csapi(x,y)。

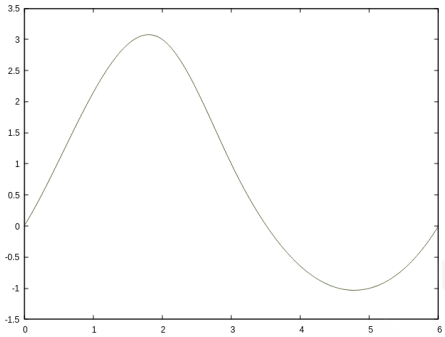
- s = csape(x,y,conds) 创建一个三次样条曲线, 边界条件由 conds 给出。x 为插值节点，y 为节点上的值，x，y 必须为长度相同的一维向量，且要求 x 已经完成单增排列。conds 可以是 ‘complete’,‘not-a-knot’,‘periodic’ 或 ‘second’ 中的一种，也可以是一个 1×2 的 int 型矩阵。当 conds 是一个 1×2 的 int 型矩阵时，0,1,2 分别表示在该端点使用 ‘not-a-knot’,‘complete’ 和 ‘second’ 边界条件。用这种方式进行插值时，边界条件的值（如果需要）被默认地设置为 0 。等价于 s = csape(x,[0,y,0],conds)。

- `s = csape(x,[e1,y,e2],conds)` 创建一个三次样条曲线, 边界条件由 `conds` 给出。`x` 为插值节点，`y` 为节点上的值，`x`，`y` 必须为长度相同的一维向量，且要求 `x` 已经完成单增排列。`e1,e2` 分别为左端和右端的边界条件的值。`conds` 的使用与 `s = csape(x,y,conds)` 中 `conds` 的用法相同。注意当采用的边界条件为 “not-a-knot” 条件时，对应的 `e` 将不会被使用。

返回结果 `s` 是满足插值条件与相应边界条件的三次样条曲线，为 pp 格式。

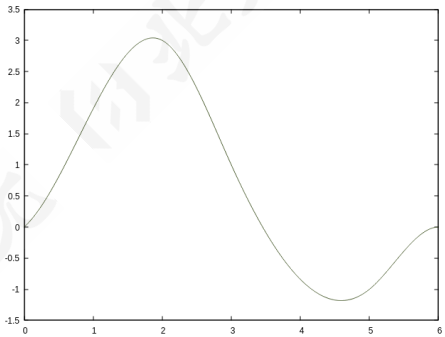
例 1： 在点列 $(0,0),(2,3),(3,1),(5,-1),(6,0)$ 上进行三次样条插值，边界条件采用 periodic 条件。

```
>> x = [0,2,3,5,6];
>> y = [0,3,1,-1,0];
>> s = csape(x,y,'periodic');
>> t = 0:0.01:6;plot(t,fnval(s,t));
```



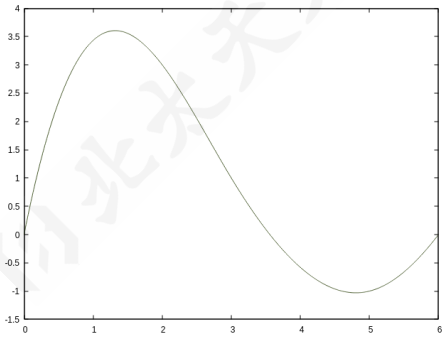
例 2： 在点列 $(0,0),(2,3),(3,1),(5,-1),(6,0)$ 上进行三次样条插值，边界条件采用 complete 条件，左端点处的导数值为 1，右端点处的导数值为 0。

```
>> x = [0,2,3,5,6];
>> y = [1,0,3,1,-1,0,0];
>> s = csape(x,y,'complete');
>> t = 0:0.01:6;plot(t,fnval(s,t));
```



例 3： 在点列 $(0,0),(2,3),(3,1),(5,-1),(6,0)$ 上进行三次样条插值，边界条件采用 mixed 条件，左端点处采用 not-a-knot 条件，右端点处采用 second 条件，对应的二阶导数值为 -1。

```
>> x = [0,2,3,5,6];
>> y = [0,0,3,1,-1,0,-1];
>> s = csape(x,y,[0,2]);
>> t = 0:0.01:6;plot(t,fnval(s,t));
```



3.3 ppmak

ppmak 函数根据给出的信息创建若干 pp 格式的样条曲线。

- [sp1,sp2,...spm] = ppmak(breaks,coefs) 根据节点和系数信息创建对应的样条曲线。要求 breaks 是严格单增的一维向量，coef 是一个矩阵，其列数 l = (length(breaks) - 1)×Order , Order 是样条曲线的阶数，最大不超过 8；行数决定了样条曲线的数量，即每行是一个以 breaks 为节点，以 coefs 对应行为系数的样条曲线。sp1,sp2,...spm 返回对应的样条曲线，为 pp 格式，这里要求 m 不超过 coefs 的行数。

- [sp1,sp2,...spm] = ppmak(breaks,coefs,d) 根据节点和系数信息创建对应的样条曲线。要求 breaks 是严格单增的一维向量，coefs 是一个矩阵，形式与 ppmak(breaks,coefs) 中不同：其列数是样条函数的阶数 Order，最大不超过 8；行数 r = (length(breaks) - 1)×d ，即每 (length(breaks) - 1) 行表示一条样条曲线，每行是一个多项式系数的降幂排列。sp1,sp2,...spm 返回对应的样条曲线，为 pp 格式，这里要求 m 不超过 d 。注意这里 coefs 的格式与 fnbrk 函数中的 'coefficients' 格式相同。

例：给定以下节点和相应的多项式，得到对应的样条曲线。

$x_1 = 1, x_2 = 3, x_3 = 5, x_4 = 7, x_5 = 8, x_6 = 10.$

sp1 :

$[1,3] : -0.1716(x-1)^3 + 1.5299(x-1)^2 - 2.8732(x-1) + 2.0000$

$[3,5] : -0.1716(x-3)^3 + 0.5000(x-3)^2 + 1.1866(x-3) + 1.0000$

$[5,7] : 0.2332(x-5)^3 - 0.5299(x-5)^2 + 1.1268(x-5) + 4.0000$

$[7,8] : -0.6757(x-7)^3 + 0.8696(x-7)^2 + 1.8062(x-7) + 6.0000$

$[8,10] : -0.6757(x-8)^3 - 1.1576(x-8)^2 + 1.5181(x-8) + 8.0000$

sp2 :

$[1,3] : -0.2083(x-1)^3 + 1.7500(x-1)^2 - 3.1667(x-1) + 2.0000$

$[3,5] : -0.2083(x-3)^3 + 0.5000(x-3)^2 + 1.3333(x-3) + 1.0000$

$[5,7] : 0.0417(x-5)^3 - 0.7500(x-5)^2 0.8333(x-5) + 4.0000$

$[7,8] : 0.1667(x-7)^3 - 0.5000(x-7)^2 - 1.6667(x-7) + 3.0000$

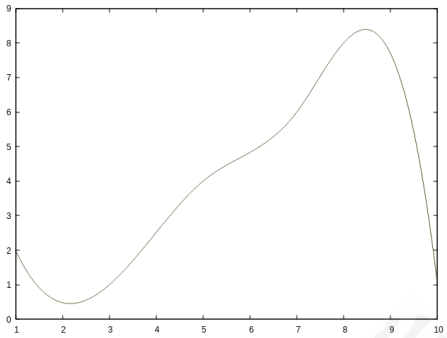
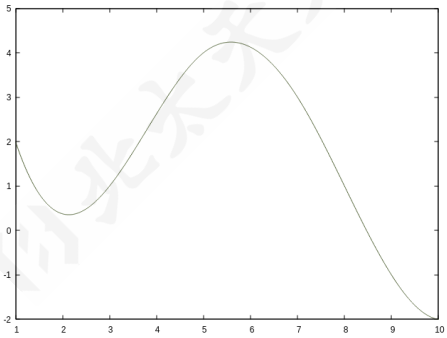
$[8,10] : 0.1667(x-8)^3 + 0.0000(x-8)^2 - 2.1667(x-8) + 1.0000$

这里我们使用 `ppmak` 的第二种用法：

```
>> breaks = [1,3,5,7,8,10];
>> coefs = [ -0.1716   1.5299  -2.8732   2.0000
  -0.1716   0.5000   1.1866   1.0000
   0.2332  -0.5299   1.1268   4.0000
  -0.6757   0.8696   1.8062   6.0000
  -0.6757  -1.1576   1.5181   8.0000
  -0.2083   1.7500  -3.1667   2.0000
  -0.2083   0.5000   1.3333   1.0000
   0.0417  -0.7500   0.8333   4.0000
   0.1667  -0.5000  -1.6667   3.0000
   0.1667   0.0000  -2.1667   1.0000];

>> [sp1,sp2] = ppmak(breaks,coefs,2);
>> t = 1:0.01:10;
>> plot(t,fnval(sp1,t))
>> plot(t,fnval(sp2,t))
```

得到的样条曲线 sp1, sp2 分别如下图所示：

3.4 spapi

spapi 函数根据给定的 B 样条节点或样条曲线次数以及插值条件创建一个 B 格式的样条曲线。

- s = spapi(knots,x,y) 创建一个样条曲线，knots 为 B 样条的节点，x 为插值节点，y 为插值节点上的值及相应阶导数值，x，y 必须为长度相同的一维向量。这里要求 x 已经完成非降排列，重复的值被视为在该点处的对应阶导数值。例如，若 1 在 x 中连续出现了三次，则其在 y 中对应的三个值依次为 1 处的值、1 处的导数值和 1 处的二阶导数值。s 的次数 k 满足 k + 1 = length(knots) - length(x)。

- s = spapi(k,x,y) 创建一个 k - 1 次样条曲线,x,y 为插值条件,其要求与 s = spapi(knots,x,y) 相同。

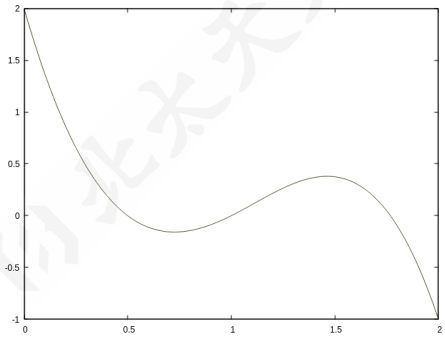
返回结果 s 是满足插值条件与指定 B 样条节点或指定次数的样条曲线，为 B 格式。

例 1： 在 B 样条节点 $\{0,0,0,0,1,2,2,2,2\}$ 上进行 B 样条插值，插值条件为

$$s(0)=2, s(1)=0, s'(1)=1, s''(1)=2, s(2)=-1$$

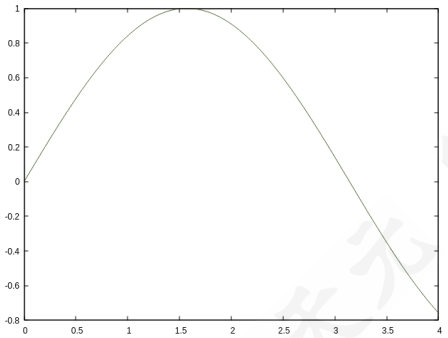
。

```
>> knots = [0,0,0,0,1,2,2,2,2];
>> x = [0,1,1,1,2];
>> y = [2,0,1,2,-1];
>> s = spapi(knots,x,y);
>> t = 0:0.01:2;plot(t,fnval(s,t));
```



例 2： 在 $[0,4]$ 上对函数 $f(x)=\sin(x)$ 进行 5 次 B 样条拟合，采样点为区间上均匀分布的 41 个点 $0, 0.1, 0.2, \cdots, 3.9, 4$。

```
>> x = 0:0.1:4;
>> s = spapi(6,x,sin(x));
>> t = 0:0.01:4;plot(t,fnval(s,t));
```



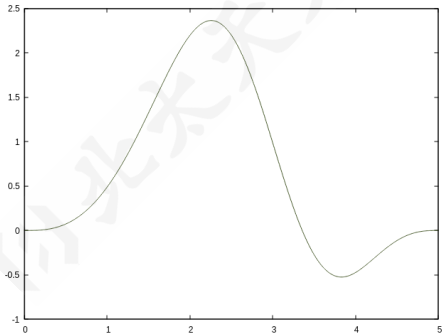
3.5 spmak

`spmak` 函数根据给出的信息创建一个 B 格式的样条曲线。

- `s = spmak(knots,coefs)` 根据 B 样条节点和系数信息创建对应的样条曲线。`knots` 表示 B 样条节点，这里要求 `knots` 是单调非减的一维向量；`coefs` 表示对应 B 样条的系数，是一维向量；样条曲线的次数 `k` 满足 `k + 1` = length(`knots`) - length(`coefs`)，且不能超过 7 。返回结果 `s` 为对应的样条曲线，为 B 格式。注意这里 `knots` 和 `coefs` 的格式与 `fnbrk` 函数中的 'knots' 与 'coefficients' 格式相同。

例： 给定 B 样条节点 $\{0,0,1,2,3,4,5,5\}$，可以得到三个 4 次的 B 样条。若这三个 B 样条的系数分别为 $1, 4$ 和 -2，则我们可以得到对应的 B 格式样条曲线。

```
>> knots = [0,0,1,2,3,4,5,5];
>> coefs = [1,4,-2];
>> s = spmak(knots,coefs);
>> t = 0:0.01:5;plot(t,fnval(s,t));
```



3.6 fitcurve

`fitcurve` 函数对二维曲线进行三次样条插值，采用 pp 格式。

- `s = fitcurve(x,y)` 创建一个二维三次样条曲线，使用 not-a-knot 边界条件。`x，y` 为插值条件，要求 `x，y` 必须为长度相同的一维向量，每个 `(x_i,y_i)` 表示一个插值点。

- `s = fitcurve(x,y,conds)` 创建一个二维三次样条曲线, 边界条件由 `conds` 给出。`x，y` 为插值条件，要求 `x，y` 必须为长度相同的一维向量，每个 `(x_i,y_i)` 表示一个插值点。`conds` 可以是 'not-a-knot' 或 'periodic'。

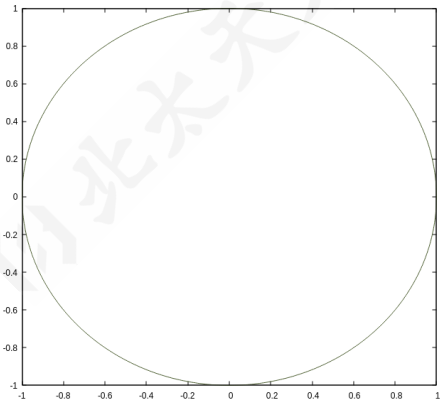
- `s = fitcurve([sx,x,ex],[sy,y,ey],conds)` 创建一个二维三次样条曲线, 边界条件由 `conds` 给出。`x,y` 为插值条件，要求 `x,y` 必须为长度相同的一维向量，每个 `(x_i,y_i)` 表示一个插值点。`(sx,sy)`、`(ex,ey)` 分别为曲线起点和终点的边界条件的取值。`conds` 可以是 'complete' 或 'second'，也可以是一个 1×2 的 int 型矩阵。当 `conds` 是一个 1×2 的 int 型矩阵时，$0, 1, 2$ 分别表示在该端点使用 'not-a-knot' , 'complete' 和 'second' 边界条件。注意当 'not-a-knot' 条件被使用时，对应的 `(sx,sy)`、`(ex,ey)` 将不会被使用。

- `[s,arclen] = fitcurve(...)` 的用法与 `s = fitcurve(...)` 相同，`arclen` 返回样条曲线在终点处的弧长参数，注意起点处的弧长参数默认为 0。

返回结果 `s` 是满足插值条件与相应边界条件的二维三次样条曲线，为 pp 格式。`arclen` 是样条曲线在终点处的弧长参数，为标量。

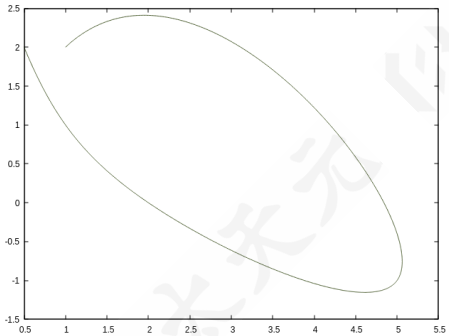
例 1：对单位圆进行三次样条插值，插值点为圆上均匀分布的 20 个点。

```
>> n = 0:19;
>> x = cos(0.1*n*pi); x=[x,x(1,1)];
>> y = sin(0.1*n*pi); y=[y,y(1,1)];
>> [s,arclen] = fitcurve(x,y,'periodic');
>> t = 0:0.01:arclen; t = [t,arclen];
>> z = fnval(s,t); plot(z(1,:),z(2,:));
```



例 2：关于二维点列 $(1,2),(5,-1),(2,0),(1,1),(0.5,2)$ 进行三次样条插值，要求起点处采用 complete 条件，对应的导数为 $(1,1)$；终点处采用 not-a-knot 条件。

```
>> [s,arclen] = fitcurve([1,1,5,2,1,0.5,0],[1,2,-1,0,1,2,0],[1,0]);
>> t = 0:0.01:arclen; t = [t,arclen];
>> z = fnval(s,t); plot(z(1,:),z(2,:));
```



3.7 bspline

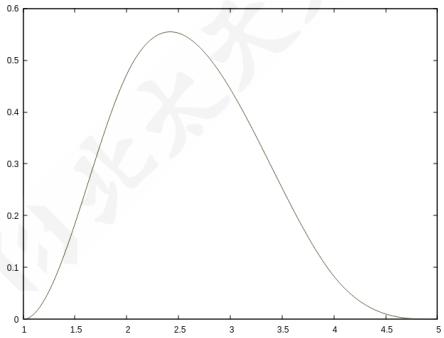
`bspline` 函数得到 B 样条节点对应的 B 样条，用 pp 格式表示。

- `s = bspline(knots)` 返回以 `knots` 为节点的 B 样条。`s` 的次数为 length(`knots`) - 2，这里要求 length(`knots`) 不超过 9。

返回结果 `s` 是以 `knots` 为节点的 B 样条，为 pp 格式。

例：对 B 样条节点 $\{1,1,2,4,5\}$，生成对应的 B 样条。

```
>> s = bspline([1,1,2,4,5]);
>> t = 1:0.01:5; plot(t,fnval(s,t));
```



3.8 fnval

`fnval` 函数返回样条函数在一点或一组点处的取值。

- `val = fnval(s,t)` 得到样条曲线 `s` 在 `t` 处的取值。`s` 可以是 pp 格式、B 格式，`t` 可以是标量、一维向量。`t` 不能超出 `s` 的有效范围。

返回结果 `val` 根据 `s` 的维数及 `t` 是向量与否，可以是标量、一维向量或二维矩阵。若 `val` 是二维矩阵，则它的每一行表示 `s` 在一个维度上 `t` 处的取值。

例：对3.2节中的例 2，计算该样条函数在 4 处的值。

```
>> x = [0,2,3,5,6];
>> y = [1,0,3,1,-1,0,0];
>> s = csape(x,y,'complete');
>> fnval(s,4)

ans =

  1x1 double matrix

  -0.8401
```

3.9 fnder

`fnder` 函数得到样条函数的导函数，仍以样条函数的格式返回。

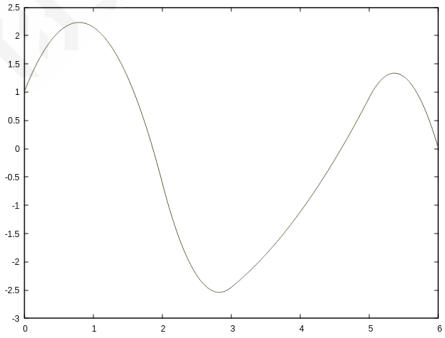
- `ds = fnder(s)` 得到样条函数 `s` 的导函数。`s` 可以是 pp 格式、B 格式，其次数至少是 1 次。

- `ds = fnder(s,order)` 得到样条曲线 `s` 的 `order` 阶导函数。`s` 可以是 pp 格式、B 格式，其次数至少是 `order` 次。`order` 是一个不超过样条函数次数的正整数。

返回结果 `ds` 是样条函数 `s` 的对应阶导函数，其类型与 `s` 一致。

例：对3.2节中的例 2，计算该样条函数的导函数。

```
>> x = [0,2,3,5,6];
>> y = [1,0,3,1,-1,0,0];
>> s = csape(x,y,'complete');
>> ds = fnder(s);
>> t = 0:0.01:6; plot(t,fnval(ds,t));
```



3.10 `fnbrk`

`fnbrk` 函数得到样条函数的相关信息。

`[out1,...,outn] = fnbrk(s,part1,...,partm)` 得到样条曲线 `s` 的 `m` 个相关信息。这里要求 $n \le m$ 。

对于任意格式的样条曲线 `s` ，`parti` 可以是如下形式：

- ‘form’ : `s` 的格式。返回 ‘ppform’ 或 ‘B-form’。
- ‘variables’ : `s` 的自变量的维数。返回一个整形标量。
- ‘dimension’ : `s` 的维数。返回一个整形标量。目前仅支持维数等于 1 的情形，即无法用于通过 `fitcurve` 生成的样条函数。
- ‘coefficients’ : `s` 的系数，对不同格式的样条曲线其含义和格式不同。目前仅支持自变量的维数等于 1 的情形。对 pp 格式，返回一个矩阵，其列数是样条函数的阶数 `Order`，行数 `r` = (length(`breaks`) - 1)，每行是一个多项式系数的降幂排列。对 B 格式，返回一个一维向量，其每个分量表示对应的 B 样条的系数。
- ‘interval’ : `s` 的有效区间。返回一个 1×2 的 double 型矩阵。
- ‘order’ : `s` 的阶数。返回一个整形标量。

对于 pp 格式的样条曲线 `s` ，`parti` 还可以是如下形式：

- ‘breaks’：s 的插值节点。返回一个向量。
- ‘pieces’：s 的多项式段数。返回一个整形标量。

对于 B 格式的样条曲线 s，parti 还可以是如下形式：

- ‘knots’：s 的 B 样条节点。返回一个向量。
- ‘number’：s 的系数的数量。返回一个整形标量。

例 1：对3.2节中的例 2，查看该样条函数的各项信息。

```
>> x = [0,2,3,5,6];
>> y = [1,0,3,1,-1,0,0];
>> s = csape(x,y,'complete');
>> [form,variables,dimension,coes,interval,order,breaks,pieces] = fnbrk(s,'form','variables','dimension','coefficients','interval','order',
'breaks','pieces')
```

```
form =
  1x6 char mat
   'ppform'

variables =
  1x1 int32 matrix
   1

dimension =
  1x1 int32 matrix
   1

coes =
  4x4 double matrix
  -0.6499   1.5497   1.0000   0.0000
   0.9489  -2.3495  -0.5995   3.0000
   0.1142   0.4973  -2.4516   1.0000
  -1.0914   1.1828   0.9086  -1.0000

interval =
  1x2 double matrix
    0.0000    6.0000

order =
  1x1 int32 matrix
   4

breaks =
  1x5 double matrix
    0.0000    2.0000    3.0000    5.0000    6.0000

pieces =
  1x1 int32 matrix
   4
```

例 2：对3.4节中的例 1，查看该样条函数的各项信息。

```
>> knots = [0,0,0,0,1,2,2,2,2];
>> x = [0,1,1,1,2];
>> y = [2,0,1,2,-1];
>> s = spapi(knots,x,y);
>> [form,variables,dimension,coes,interval,order,knots,number] = fnbrk(s,'form','variables','dimension','coefficients','interval','order',
'knots','number')
```

```
form =
  1x6 char mat
   'B-form'

variables =
  1x1 int32 matrix
   1

dimension =
  1x1 int32 matrix
   1

coes =
  1x5 double matrix
   2.0000   -0.3333   -0.3333    1.0000   -1.0000

interval =
  1x2 double matrix
   0.0000    2.0000

order =
  1x1 int32 matrix
   4

knots =
  1x9 double matrix
   0.0000    0.0000    0.0000    0.0000    1.0000    2.0000    2.0000    2.0000    2.0000

number =
  1x1 int32 matrix
   5
```

3.11 fn2fm

fn2fm 函数将样条函数在不同格式间转换。目前仅支持一维 B 格式样条函数向一维 pp 格式样条函数的转换。

- ns = fn2fm(s,form) 将样条函数 s 转换为 form 格式。目前 form 仅支持 'pp'。

返回结果 ns 是转换后的样条函数，类型为 form。

例：对3.4节中的例 1，将该样条函数转换为 pp 格式。

```
>> knots = [0,0,0,0,1,2,2,2,2];
>> x = [0,1,1,1,2];
>> y = [2,0,1,2,-1];
>> s = spapi(knots,x,y);
>> ns = fn2fm(s,'pp');
>> [breaks,coes] = fnbrk(ns,'breaks','coefficients');
>> breaks

breaks=

  1x3 double matrix

    0.0000    1.0000    2.0000



>> coes

coes=

  2x4 double matrix

   -2.0000    7.0000   -7.0000    2.0000
   -3.0000    1.0000    1.0000    0.0000
```