

由于突发恶疾住院若干天，出院之后沉迷アトリ -My Dear Moments-和萨特《存在与虚无》，故本篇记录时隔一整月。

本篇继续讨论函数插值的问题

函数插值法

在前述的两种插值法中，我们都构造了一个函数，使得至少在各个节点处与潜在的函数 $f(x)$ 同值。但如果我们想要通过插值函数 $H(x)$ 来刻画 $f(x)$ 更加深刻的性质，如一阶导数，那么前述的两种插值方法将难以胜任。

Hermite插值

我们试图找到某插值函数 $H(x)$ 使得在 $(n+1)$ 个点的点集上，有：

$$\forall i \in [0, n] : H(x_i) = y_i, H'(x_i) = y'_i \quad (5.1)$$

直观地，在 $H(x)$ 中蕴含了更多的信息，这样一来就可能需要更多的自由度来承载这些信息，当然我们只有 $(2n+2)$ 个约束条件，因此天然地，Hermite插值不可以也不会超过 $(2n+1)$ 次。

基函数构造法

与在Lagrange插值法中作的工作相似，我们同样希望找到一组基，使得它满足这一问题中的要求：我们将上述的问题分开处理：

首先找到某一类函数 $h_i(x)$ ，使得对于样本点集中的点 (x_j, y_j) 有：

$$h_i(x_j) = 0, j \neq i$$

$$h_i(x_j) = 1, j = i$$

同时满足：它的一阶导数在样本点集上恒为0：

$$h'_i(x_j) \equiv 0$$

这样，我们即可使用 $h(x)$ 来控制插值函数的函数值而不影响其导数。

同样地，我们试图找到 $H_i(x)$ ，使得对于样本点集中的点 (x_j, y_j) 有：

$$H'_i(x_j) = 0, j \neq i$$

$$H'_i(x_j) = 1, j = i$$

同时，它的原值应当在样本点集上恒为0：

$$H_i(x_j) \equiv 0$$

这样，我们即可使用 $h(x)$ 来控制插值函数的一阶导数而不影响其原值。

幸运的是，数学家们创造性地发现了这组基的形式：

$$\begin{cases} h_i(x) = [a + b(x - x_i)]l_i^2(x) \\ a = 1 \\ b = -2l'_i(x_i) \end{cases}$$

$$\begin{cases} H_i(x) = cl_i^2(x) \\ c = 1 \end{cases}$$

其中, $l_i(x)$ 是关于 x_i 的Lagrange基:

$$l_i(x) = \frac{\prod_{j=0, j \neq i}^n (x - x_j)}{\prod_{j=0, j \neq i}^n (x_i - x_j)}$$

因此, 在该方法中, 每一个点-导数三元组 (x_i, y_i, y'_i) 生成一项

$$y_i[1 - 2l'_i(x_i)(x - x_i)]l_i^2(x) + y'_i(x - x_i)l_i^2(x)$$

由此可给出Hermite插值表达式:

$$H(x) = \sum_{i=0}^n \{y_i[1 - 2l'_i(x_i)(x - x_i)]l_i^2(x) + y'_i(x - x_i)l_i^2(x)\}$$

Hermite插值法拥有误差表达式

$$R_n(x) = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \omega_{n+1}^2(x)$$

这样的“函数值-导数”分别控制的方法有着更广阔的应用空间, 如给出点集中部分拥有导函数值而其余没有, 而这种形式的问题需要构造额外的基函数(如使用原基函数则待定参数数量过多), 而构造这一基函数的过程需要较强的分析要求, 故不建议使用基函数构造法. (课后作业习题9涉及到这一方法, 可以试着做一下)

对于上述这种“不完全”的情况, 误差表达式会有所变化, 但总得来说, 如果给出了 n 组原值条件, 其中有 m 组限制了导数值(不妨设它们就是第 $0 \rightarrow (m-1)$ 个), 则误差可以表示为

$$R_n(x) = \frac{f^{(n+m+2)}(\xi)}{(n+m+2)!} \omega_{n+1}(x) \prod_{i=0}^{(m-1)} (x - x_i)$$

待定系数法

这种方法直接将 $H(x)$ 显式地假设出来, 再代入点集解方程组(5.1). 严格地来讲这种算法不能称之为方法, 但在数据量较小时很好用.

降阶法

而面对稍多的数据时, 待定系数法就不那么好用了. 因此我们试图通过某种方式来减少需要处理的自由度:

我们先通过前述的插值法求出仅在原函数值上提供保证的插值函数:

$$N(x), s.t. \forall i \in [0, n], N(x_i) = y_i$$

明显地，Hermite插值函数与 $N(x)$ 存在如下的关系：

$$H(x) - N(x) = P(x) \prod_{i=0}^n (x - x_i)$$

而此时，由于 $H(x)$ 是 $2n + 1$ 阶多项式，而 $N(x)$ 是 n 阶多项式，故 $H(x) - N(x)$ 是 $2n + 1$ 阶多项式，故 $P(n)$ 是 n 阶多项式。

此时再利用 $H'(x_i) = y_i$ 对 $P(n)$ 待定系数直球求解即可。

样条插值法

样条

对于一水平弹性木条，在若干点 $\{x_0, x_1, x_2, \dots, x_n\}$ 处施加垂直于其的外力，产生弯矩 $M(x)$ ，由Euler-Bernoulli梁方程，每两点之间的弯矩 $M(x)$ 与材料杨氏模量 E 、产生的惯性矩 I 与产生曲线的曲率 $k(x)$ 有如下关系：

$$M(x) = EI k(x) = EI \frac{y''}{(1 + y'^2)^{1.5}}$$

一般的梁难以发生大的弯曲，故 y' 接近于0，有：

$$M(x) \approx EI y''$$

而每两点之间没有额外的力，故产生的弯矩是线性的，故 y'' 是一分段一次函数，故可以得知 y 是一拥有连续二阶导数的分段三次函数。

样条插值

因此设置第 i 点和第 $(i + 1)$ ， $i \in [0, n]$ 点之间的函数表达式是 $\phi_i = a_i + b_i x + c_i x^2 + d_i x^3$ ，总共有 n 段曲线，产生 $4n$ 个自由度。接下来数数我们有什么条件：

1. 插值条件，在 $i \in [0, n]$ 时 $\phi(x_i) = y_i$ ， $(n + 1)$ 个
2. 插值条件中有 $(n - 1)$ 个被两段曲线共用，属自由度的冗余，消除 $(n - 1)$ 个自由度(书上称为函数值连续条件)
3. 一阶导数连续，在 $(n - 1)$ 个共用点上 $\phi'_i(x_{i+1}) = \phi'_{i+1}(x_{i+1})$ ， $(n - 1)$ 个
4. 二阶导数连续，在 $(n - 1)$ 个共用点上 $\phi''_i(x_{i+1}) = \phi''_{i+1}(x_{i+1})$ ， $(n - 1)$ 个

如此，我们拥有 $(4n - 2)$ 个条件，明显不能解决 $4n$ 个自由度的问题。

因此我们可以通过某些人为的限定来为其附加2个限定条件，而最佳的方式就是将其附加在未被连续性约束的两个边界点 x_0, x_n 上。

常用的方式有：

1. 令 $\phi'(x_0) = \phi'(x_n) = 0$
2. 令 $\phi''(x_0) = \phi''(x_n) = 0$ ，此时与工程中使用的样条之情况(边界弯矩为0)符合，称为自然三次样条

我们用自然三次样条为例，指出如何求解样条：

首先假定任意第 $i \in [1, n-1]$ 段函数均有参数列表 a_i, b_i, c_i, d_i ，我们采取较为巧妙的"自底向上"方式以降低计算难度：

在每一段上， $\phi_i(x)$ 的二阶导数均是线性函数，易知在每一段上：

$$\phi_i''(x) = \phi''(x_i) \frac{x_{i+1} - x}{x_{i+1} - x_i} - \phi''(x_{i+1}) \frac{x_i - x}{x_{i+1} - x_i}$$

为简化表达，我们约定新符号 $h_i = x_{i+1} - x_i$ ，使用之前在“力学背景”中提到的弯矩 M_i 来代表 $\phi(x_i)$ ：

$$\phi_i''(x) = M_i \frac{x_{i+1} - x}{h_i} - M_{i+1} \frac{x_i - x}{h_i}$$

为利用函数值条件，对上式执行两次积分，并适当调整积分常数的位置，得到：

$$\phi_i(x) = \frac{M_i}{6h_i}(x_{i+1} - x)^3 - \frac{M_{i+1}}{6h_i}(x_i - x)^3 + C(x - x_i) + D(x_{i+1} - x)$$

代入条件 $\phi_i(x_i) = y_i, \phi_i(x_{i+1}) = y_{i+1}$ ，解出上述积分常数：

$$\phi_i(x) = \frac{M_i}{6h_i}(x_{i+1} - x)^3 - \frac{M_{i+1}}{6h_i}(x_i - x)^3 + \left(\frac{y_{i+1}}{h_i} - \frac{M_{i+1}h_i}{6}\right)(x - x_i) - \left(\frac{y_i}{h_i} - \frac{M_ih_i}{6}\right)(x - x_{i+1})$$

有了这一方程，我们就只需求解出各个弯矩 M_i ，即可得解。

我们还有一个未用的条件，即 $(n-1)$ 个导数的连续性条件：

对上述方程左右求导并令 $\phi_{i-1}(x_i) = \phi_i(x_i)$ ，有

$$h_{i-1}M_{i-1} + 2(h_i + h_{i-1})M_i + h_iM_{i+1} = \frac{6}{h_i}(y_{i+1} - y_i) - \frac{6}{h_i}(y_i - y_{i-1})$$

上述方程被称为三弯矩方程，其中 $i \in [1, n-1]$ 。我们令它是自然样条，即 $M_0 = M_n = 0$ ，得到一方程组（为了简化表达，我们记 $v_i = RHS$ ， $u_i = 2(h_i + h_{i-1})$ ）：

$$\begin{pmatrix} u_1 & h_1 & & & \\ h_1 & u_2 & h_2 & & \\ & h_2 & u_3 & h_3 & \\ & & \ddots & \ddots & \ddots \\ & & & h_{n-3} & u_{n-2} & h_{n-2} \\ & & & & h_{n-2} & u_{n-1} & h_{n-1} \end{pmatrix} \begin{pmatrix} M_1 \\ M_2 \\ M_3 \\ \vdots \\ M_{n-2} \\ M_{n-1} \end{pmatrix} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{pmatrix}$$

使用前述的"追赶法"解决之，即可得到样条插值函数。

函数逼近

函数逼近问题给出一目标函数 $f(x)$ ，希望提出一(容易处理的)近似函数 $\phi(x)$ 来代替 $f(x)$ ，以近似刻画并简化对 $f(x)$ 的诸多运算。

而插值问题是给出一系列点 $\{(x_i, y_i)\}$ ，希望提出一拟合函数 $\phi(x)$ 以近似刻画这些点所隐含的函数关系 $f(x)$ 。

函数逼近法的总体过程

1. 提出近似函数 $\phi(x)$ 的形式并预留一些待定系数 $\{a_0, a_1, a_2, \dots\}$
2. 使用某种误差估计手段写出误差 E 关于上述待定系数 $\{a_0, a_1, a_2, \dots\}$ 的关系
3. 通过最小化误差 E 求出 $\{a_0, a_1, a_2, \dots\}$

由于其有着便于计算的性质，我们通常使用多项式函数 $\phi(x) = \sum_{i=0}^n a_i x^i$ 来进行拟合，同时使用最小二乘法来描述误差。

在某些情况下，可能会将自变量 x 映射为其它的函数，如设置原型 $y = a_0 + a_1 e^x + a_2 e^{2x}$ 逼近，则可以对逼近点列进行变换如 (e^{x_0}, y_0) 。

最小二乘法

“最小二乘”是一种描述误差的尺度，即使用均方误差(假使在点集 P 上考察)，其“误差泛函”表示如下：（注：仅仅是形式不同的事物不需要讨论多次，故此处的求和符号可以为离散求和，也可以为连续积分）

$$E[\phi_A(x)] = \sum_{(x,y) \in P} (f(x) - \phi(x))^2$$

我们找出某种手段来最小化这个 E 即可。幸运的是，我们在针对“形式固定-系数待定”的多项式的讨论中不需要面对上述令人生畏的泛函，而可以将其转写为：

$$E(a_0, a_1, a_2, \dots, a_n) = \sum_{(x,y) \in P} [y - (\sum_{i=0}^n a_i x^i)]^2$$

特别地，这种最小二乘意义下的误差是一串平方和，这意味着它必定有一个最小值，以下简单**证明**（这证明是我自己写的，虽然还挺漂亮的，但是可能有逻辑断裂可以跳过不看）

假定有一 $(n+1)$ 个相互独立的自变量之函数： $f(a_0, a_1, a_2, \dots, a_n) = \sum_{i=0}^n a_i^2$

易知 $\forall i \in [0, n], i = 0$ 时函数 f 取得极值，而此时其Hessian矩阵为 $diag(2, 2, 2, 2, \dots, 2)$ ，明显是正定的，因此它在此处取最小值。

我们设置 $A = (a_0, a_1, a_2, \dots, a_n)$ ，那么对于任一和 A 无关的 $(n+1)$ 个向量 x_i 与标量 $y_i (i \in [0, n])$ ，我们产生一个新的 $(n+1)$ 维向量 $(y_0 + Ax_0, y_1 + Ax_1, y_2 + Ax_2, \dots, y_n + Ax_n)$ ，易知这个向量仅仅是将向量 A 在其空间内平移、放缩而未旋转，未实际改变该空间的拓扑性质，因此 f 在新的自变量列 $(y_0 + Ax_0, y_1 + Ax_1, y_2 + Ax_2, \dots, y_n + Ax_n)$ 下仍拥有一唯一最小值点。

而此时， f 恰可以被写为 $f(a_0, a_1, a_2, \dots, a_n) = \sum_{(x,y) \in P} [y - (\sum_{i=0}^n a_i x^i)]^2$

■

因此这提示了我们最小二乘法的解决策略：对于误差函数 $E(a_0, a_1, a_2, \dots, a_n) = \sum_{(x,y) \in P} [y - (\sum_{i=0}^n a_i x^i)]^2$ ，我们求：

$$\left\{ \begin{array}{l} \frac{\partial E}{\partial a_0} = -2 \sum_{(x,y) \in P} (y - \sum_{j=0}^n a_j x^j) \\ \frac{\partial E}{\partial a_1} = -2 \sum_{(x,y) \in P} (y - \sum_{j=0}^n a_j x^j) x \\ \dots \\ \frac{\partial E}{\partial a_k} = -2 \sum_{(x,y) \in P} (y - \sum_{j=0}^n a_j x^j) x^k \\ \dots \\ \frac{\partial E}{\partial a_n} = -2 \sum_{(x,y) \in P} (y - \sum_{j=0}^n a_j x^j) x^n \end{array} \right.$$

用更为通俗的方式展开之：

$$\left\{ \begin{array}{l} \frac{\partial E}{\partial a_0} = -2 \sum_{(x,y) \in P} [y - (a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n)] \\ \frac{\partial E}{\partial a_1} = -2 \sum_{(x,y) \in P} [y - (a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n)] x \\ \dots \\ \frac{\partial E}{\partial a_k} = -2 \sum_{(x,y) \in P} [y - (a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n)] x^k \\ \dots \\ \frac{\partial E}{\partial a_n} = -2 \sum_{(x,y) \in P} [y - (a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n)] x^n \end{array} \right.$$

令其全部为0以找到最小值，整理得到方程组

$$\left\{ \begin{array}{l} \sum_{(x,y) \in P} y = \sum_{(x,y) \in P} a_0 + \sum_{(x,y) \in P} x a_1 + \sum_{(x,y) \in P} x^2 a_2 + \dots + \sum_{(x,y) \in P} x^n a_n \\ \sum_{(x,y) \in P} x y = \sum_{(x,y) \in P} x a_0 + \sum_{(x,y) \in P} x^2 a_1 + \sum_{(x,y) \in P} x^3 a_2 + \dots + \sum_{(x,y) \in P} x^{n+1} a_n \\ \dots \\ \sum_{(x,y) \in P} x^k y = \sum_{(x,y) \in P} x^k a_0 + \sum_{(x,y) \in P} x^{k+1} a_1 + \sum_{(x,y) \in P} x^{k+2} a_2 + \dots + \sum_{(x,y) \in P} x^{k+n} a_n \\ \dots \\ \sum_{(x,y) \in P} x^n y = \sum_{(x,y) \in P} x^n a_0 + \sum_{(x,y) \in P} x^{n+1} a_1 + \sum_{(x,y) \in P} x^{n+2} a_2 + \dots + \sum_{(x,y) \in P} x^{2n} a_n \end{array} \right.$$

写为矩阵形式则为：

$$MA = b$$

$$M = \begin{pmatrix} \sum_{(x,y) \in P} 1 & \sum_{(x,y) \in P} x & \sum_{(x,y) \in P} x^2 & \cdots & \sum_{(x,y) \in P} x^n \\ \sum_{(x,y) \in P} x & \sum_{(x,y) \in P} x^2 & \sum_{(x,y) \in P} x^3 & \cdots & \sum_{(x,y) \in P} x^{n+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{(x,y) \in P} x^k & \sum_{(x,y) \in P} x^{k+1} & \sum_{(x,y) \in P} x^{k+2} & \cdots & \sum_{(x,y) \in P} x^{k+n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{(x,y) \in P} x^n & \sum_{(x,y) \in P} x^{n+1} & \sum_{(x,y) \in P} x^{n+2} & \cdots & \sum_{(x,y) \in P} x^{2n} \end{pmatrix}$$

$$b = \begin{pmatrix} \sum_{(x,y) \in P} y \\ \sum_{(x,y) \in P} xy \\ \cdots \\ \sum_{(x,y) \in P} x^k y \\ \cdots \\ \sum_{(x,y) \in P} x^n y \end{pmatrix}$$

此方程组被称为**正则方程组**。它依据求和方式的不同，可以有两种表示形式，分别对应离散点集上逼近和连续点集上逼近两种情况：

1. 离散点集逼近：

$$M = \begin{pmatrix} \sum_{i=0}^m 1 & \sum_{i=0}^m x & \sum_{i=0}^m x^2 & \cdots & \sum_{i=0}^m x^n \\ \sum_{i=0}^m x & \sum_{i=0}^m x^2 & \sum_{i=0}^m x^3 & \cdots & \sum_{i=0}^m x^{n+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{i=0}^m x^k & \sum_{i=0}^m x^{k+1} & \sum_{i=0}^m x^{k+2} & \cdots & \sum_{i=0}^m x^{k+n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{i=0}^m x^n & \sum_{i=0}^m x^{n+1} & \sum_{i=0}^m x^{n+2} & \cdots & \sum_{i=0}^m x^{2n} \end{pmatrix}$$

$$b = \begin{pmatrix} \sum_{i=0}^m y \\ \sum_{i=0}^m xy \\ \cdots \\ \sum_{i=0}^m x^k y \\ \cdots \\ \sum_{i=0}^m x^n y \end{pmatrix}$$

2. 连续区间逼近

$$M = \begin{pmatrix} \int_a^b 1dx & \int_a^b xdx & \int_a^b x^2dx & \cdots & \int_a^b x^ndx \\ \int_a^b xdx & \int_a^b x^2dx & \int_a^b x^3dx & \cdots & \int_a^b x^{n+1}dx \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \int_a^b x^kdx & \int_a^b x^{k+1}dx & \int_a^b x^{k+2}dx & \cdots & \int_a^b x^{k+n}dx \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \int_a^b x^ndx & \int_a^b x^{n+1}dx & \int_a^b x^{n+2}dx & \cdots & \int_a^b x^{2n}dx \end{pmatrix}$$

$$b = \begin{pmatrix} \int_a^b f(x)dx \\ \int_a^b x f(x)dx \\ \dots \\ \int_a^b x^k f(x)dx \\ \dots \\ \int_a^b x^n f(x)dx \end{pmatrix}$$

非多项式逼近，一般函数族

在上述的讨论中，我们设置逼近函数为 $\phi(x) = \sum_{i=0}^n a_i x^i$ ，即我们使用了函数族 $\{1, x, x^2, \dots, x^n, \dots\}$ 的某一线性组合来逼近目标函数 $f(x)$ ，且逼近点集上的各个点有着相同的"重要程度".

我们将这个过程推广开来：我们使用一般的函数族 $\{\phi_0(x), \phi_1(x), \phi_2(x), \dots, \phi_n(x), \dots\}$ 来刻画目标函数 $f(x)$ ，且不同的逼近锚点 (x, y) 拥有不同的非负非全零(实际上权函数有严格的约束，但我们可以理解为非负非全零(反正不会考))的"重要性"权函数 $\omega(x)$ ，那么，我们设置逼近函数为 $\phi(x) = \sum_{i=0}^n a_i \phi_i(x)$ ，在带权点集 P 下，其"最小二乘"尺度下的误差可以写作：

$$E(a_0, a_1, a_2, \dots, a_n) = \sum_{(x,y) \in P} \omega(x) [y - (\sum_{i=0}^n a_i \phi_i(x))]^2$$

(推导过程简单，略) 为了能使矩阵变得好看一点，我们简记 $(A, B) = \sum_{(x,y) \in P} \omega(x) A(x) B(x)$ 则正则方程组可以写作

$$MA = b$$

$$M = \begin{pmatrix} (\phi_0, \phi_0) & (\phi_0, \phi_1) & (\phi_0, \phi_2) & \dots & (\phi_0, \phi_n) \\ (\phi_1, \phi_0) & (\phi_1, \phi_1) & (\phi_1, \phi_2) & \dots & (\phi_1, \phi_n) \\ \vdots & \vdots & \vdots & & \vdots \\ (\phi_i, \phi_0) & (\phi_i, \phi_1) & (\phi_i, \phi_2) & \dots & (\phi_i, \phi_n) \\ \vdots & \vdots & \vdots & & \vdots \\ (\phi_n, \phi_0) & (\phi_n, \phi_1) & (\phi_n, \phi_2) & \dots & (\phi_n, \phi_n) \end{pmatrix}$$

$$b = \begin{pmatrix} (\phi_0, f) \\ (\phi_1, f) \\ \dots \\ (\phi_2, f) \\ \dots \\ (\phi_n, f) \end{pmatrix}$$

为了使得上述方程恰定，矩阵 M 须满秩，这对函数族提出了要求(*iff*指“当且仅当”)：

$$\sum_{i=0}^n a_i \phi_i(x) = 0 \text{ iff } \forall i \in [0, n], a_i = 0$$

这一性质被称为函数族"线性无关".

正交函数族

解线性方程组是痛苦的，故我们对上述方程组可以作出更进一步的幻想：“如果矩阵 M 是一个对角矩阵就再好不过了”。这样的幻想要求函数族 ϕ_i 和权函数 $\omega(x)$ 具有这样的性质：

$$\forall i, j \in [0, n] : i \neq j \iff \sum_{(x,y) \in P} \omega(x) \phi_i(x) \phi_j(x) = 0$$

$$\forall i \in [0, n] : \sum_{(x,y) \in P} \omega(x) \phi_i(x) \phi_i(x) = a_i > 0$$

如有此性质，则称函数族 $\{\phi_i\}$ 在域 P 上关于权函数 $\omega(x)$ 正交. 易知一正交函数族所导出的正则方程组之系数矩阵必满秩，故正则方程组必有唯一解.

特别地，若对于任意的 i ，上述 $a_i = 1$ ，则称其为**正交-归一化函数族**. 正交-归一化函数族不仅在函数逼近中出现，在诸多领域中均有应用(如量子化学中的本征波函数)

既然正交函数族如此好用，那么是否可以将设置出的简单的非正交函数族如 $\{x^i\}$ 转化为正交函数族呢？

构造正交函数族，Gram-Schmidt正交化方法

有一线性无关函数族 $\{\phi_i(x)\}$ ，重构一新的函数族 $\{\Phi_i(x)\}$ ：

$$\Phi_0(x) = \phi_0(x)$$

$$\Phi_i(x) = \begin{vmatrix} (\phi_0, \phi_0) & (\phi_0, \phi_1) & \dots & (\phi_0, \phi_{i-1}) & \phi_0(x) \\ (\phi_1, \phi_0) & (\phi_1, \phi_1) & \dots & (\phi_1, \phi_{i-1}) & \phi_1(x) \\ \vdots & \vdots & & \vdots & \vdots \\ (\phi_i, \phi_0) & (\phi_i, \phi_1) & \dots & (\phi_i, \phi_{i-1}) & \phi_i(x) \end{vmatrix}$$

此时

$$(\Phi_i, \Phi_j) = \begin{cases} 0, i \neq j \\ \Delta_i, i = j \end{cases}$$

其中

$$\Delta_i = \begin{vmatrix} (\phi_0, \phi_0) & (\phi_0, \phi_1) & \dots & (\phi_0, \phi_{i-1}) & (\phi_0, \phi_i) \\ (\phi_1, \phi_0) & (\phi_1, \phi_1) & \dots & (\phi_1, \phi_{i-1}) & (\phi_1, \phi_i) \\ \vdots & \vdots & & \vdots & \vdots \\ (\phi_i, \phi_0) & (\phi_i, \phi_1) & \dots & (\phi_i, \phi_{i-1}) & (\phi_i, \phi_i) \end{vmatrix}$$

课本151页给出了Gram-Schmidt正交化方法的一应用实例，在153页起给出了若干常用的正交函数族.