Chapter 2

Basic Approximaiton Problems

2.1 方程求根问题

定义2.1.1 (求根问题)**.** 已知$\mathbf{f}$是从$\mathbb{R}^n$到$\mathbb{R}^n$的一个非线性映射。对于任意给定的$\mathbf{y} \in \mathbb{R}^n$，问：$\mathbf{x}$取何值时，$\mathbf{f}$的取值恰好为$\mathbf{y}$，即使得

$$\mathbf{f}(\mathbf{x}) = \mathbf{y}$$

成立？更一般地，将两边同时减去$\mathbf{y}$，使其变为标准形式(为简单起见这里仍用f表示新的映射)：

$$\mathbf{f}(\mathbf{x}) = \mathbf{0} \tag{2.1}$$

例2.1.1. 一元非线性方程

$$f(x) = x^2 - 4\sin x = 0$$

的一个近似解为$x = 1.93375$。二元非线性方程组

$$f(x) = \begin{bmatrix} x^2 - y + 0.25 \\ -x + y^2 + 0.25 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

的解为$x = [0.5, 0.5]^T$.

2.1.1 存在唯一性

对于一般的$f(x) = 0$，我们无法给出该非线性方程组解的通用表达式。线性方程组只有$0/1/\infty$个解等情况，非线性方程组解的个数可能是任意个。一个经典的做法是以一维情形为突破线索：

例2.1.2 (几个典型的一维问题)**.** $e^x + 1 = 0$,无解
$e^x - x = 0$,有一个解
$x^2 - 4\sin x = 0$,有两个解
$\sin x = 0$,有无穷多个解

回顾数学分析中的方法，几个存在性定理是研究方程解存在性的几类典型方法：

引理2.1.1 (介值定理)**.** 若f是闭区间上[a,b]的连续函数，c介于f(a)和f(b)，之间，则必存在一个$x^1 \in [a, b]$，满足$f(x^1) = c$，取c为0即可证明f在[a,b]上一定有根。

其次，反函数定理$\mathbf{x} = f^{-1}(\mathbf{0})$。更进一步地，还有压缩映射理论(也是迭代法收敛性的基本定理)也可以获得解的存在性证明.

定义2.1.2 (不动点). f:$\mathbb{R}^n \to \mathbb{R}^n$是集合$S \in \mathbb{R}^n$上的压缩映射，即存在$\gamma \in [0,1]$,对任意集合S内的两个点x，z满足$||f(x) - f(z)|| \leq \gamma||x - z||$。满足$g(x) = x$的$x$称为g的不动点。

定理2.1.1 (压缩映射(不动点)定理). 若g在闭集S上是压缩映射，且$g(S) \in S$,则g在S内存在唯一的不动点。此时如果非线性方程有形如f=x-g，则我们可称，f=0在闭集S上是有唯一解的。

而关于根的进一步研究，需要借助topological degree of function f与重根理论。

2.1.2 单个方程求根方法

让我们通过讨论标量非线性方程的求根方法来回顾

二分法

假设我们有一个比较小的区间[a,b]，而函数f在此区间上有符号变化，则连续函数f在这个区间内一定有零点。设初始的函数为f，满足f(a)与f(b)的符号不相同，最终区间长度的误差上线为tol。我们可以得到如下二分法的算法。一个Matlab实现：

```
function x = bisect(f,a,b,tol)
  xlow = a;  plow  = f(xlow);
  xhigh = b; phigh = f(xhigh);
  while xhigh - xlow > 2*tol,
    xmid = (xlow + xhigh)/2; pmid = f(xmid);
    if  pmid*plow < 0,
        xhigh = xmid;   phigh = pmid;
    elseif  pmid*phigh < 0,
        xlow = xmid;     plow = pmid;
    else
        xlow = xmid;    xhigh = xmid;
    end
  end
  x = [xlow, xhigh];
end
```

不动点迭代

不动点迭代的思想为，将原问题变形为：$x = g(x)$的形式，再进行求解，我们可以构造格式$x_{k+1} = gx_k$来对非线性方程进行求解。值得一提的是，这样的构造方式并不唯一，相对应的，不同的构造方式也有不同的稳定性以及收敛速度。

例2.1.3. 非线性方程$f(x) = x^2 - x - 2 = 0$的根为$x^* = 2$和$x^* = -1$,等价的不动点问题可以包括：
1.$g(x) = x^2 - 2$;
2.$g(x) = \sqrt{x + 2}$;
3.$g(x) = 1 + 2/x$;
4.$g(x) = (x^2 + 2)/(2x - 1)$;

根据不同的迭代方程，都可求出相应的零点.

例 5.7　二分法迭代　用二分法求方程 $f(x)=x^3-4\sin x=0$ 的根. 取初始区间$[a,b]$为 $a=1, b=3$, 最重要的是使函数值在区间端点反号. 计算中点 $m=a+(b-a)/2=2$ 处的函数值知, $f(m)$ 与 $f(a)$ 的符号相反, 所以保留初始区间的左半部, 即取 $b=m$. 重复这个过程, 直到有根区间将方程的根隔离到所需的精确度. 迭代序列如下:

a	$f(a)$	b	$f(b)$
1.000000	−2.365884	3.000000	8.435520
1.000000	−2.365884	2.000000	0.362810
1.500000	−1.739980	2.000000	0.362810
1.750000	−0.873444	2.000000	0.362810
1.875000	−0.300718	2.000000	0.362810
1.875000	−0.300718	1.937500	0.019849
1.906250	−0.143255	1.937500	0.019849
1.921875	−0.062406	1.937500	0.019849
1.929688	−0.021454	1.937500	0.019849
1.933594	−0.000846	1.937500	0.019849
1.933594	−0.000846	1.935547	0.009491
1.933594	−0.000846	1.934570	0.004320
1.933594	−0.000846	1.934082	0.001736
1.933594	−0.000846	1.933838	0.000445
1.933716	−0.000201	1.933838	0.000445
1.933716	−0.000201	1.933777	0.000122
1.933746	−0.000039	1.933777	0.000122
1.933746	−0.000039	1.933762	0.000041
1.933746	−0.000039	1.933754	0.000001
1.933750	−0.000019	1.933754	0.000001
1.933752	−0.000009	1.933754	0.000001
1.933753	−0.000004	1.933754	0.000001

牛顿迭代

牛顿法的思想可以理解为：由于零点是其切线与x的交点，非零点则不是，于是在点x_k附近，使用$f(x_k)$处的切线来近似，使用这个切线的零点作为新的近似值，以此来靠近真正的零点。其算法流程可表示为：
x_0 =初始值
for k = 0,1,2,...
$x_{k+1} = x_k - f(x_k)/f'(x_k)$
end

例 5.10　牛顿法　用牛顿法求方程

$$f(x) = x^2 - 4\sin x = 0$$

的根.

函数 $f(x)$ 的导数为

$$f'(x) = 2x - 4\cos x,$$

因此迭代格式为

$$x_{k+1} = x_k - \frac{x_k^2 - 4\sin x_k}{2x_k - 4\cos x_k}.$$

取初始值 $x_0 = 3$，得到下面的迭代序列，其中 $h_k = -f(x_k)/f'(x_k)$ 表示迭代中 x_k 的变化量. 当 $|h_k| / |x_k|$ 或 $|f(x_k)|$，或者这两者都小于给定误差时，中止迭代.

k	x_k	$f(x_k)$	$f'(x_k)$	h_k
0	3.000000	8.435520	9.959970	−0.846942
1	2.153058	1.294772	6.505771	−0.199019
2	1.954039	0.108438	5.403795	−0.020067
3	1.933972	0.001152	5.288919	−0.000218
4	1.933754	0.000000	5.287670	0.000000

割线法

牛顿法有一个计算上的缺陷，即在每次迭代时都要计算函数及其导数的值，导数在计算中往往不方便或者计算量很大，因此在步长较小的情况下，我们可以用有限差分来近似代替导数，即用相邻两次迭代的函数值来代替其导数。这种方法叫割线法。

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

其算法流程可以表示为：
x_0 =初始值
for k = 0,1,2,...
$x_{k+1} = x_k - f(x_k)(x_k - x_{k-1}/[f(x_k) - f(x_{k-1})]$
end

如何计算多项式的所有零点？这是一个被人们研究了很久的问题。对于某些特殊情形的n阶多项式p(x)，有时要求求出它的所有n个零点，具体可以使用以下的思想来进行求值：用之前的方法，如牛顿法求出一个根x_1，考虑低一阶的收缩多项式$p(x)/(x - x_1)$，重复此过程，直到求出全部的根。形成给定多项式的友阵，利用之前讲过的计算特征值的方法计算特征值，即计算出多项式的根。一些专门计算多项式零点的方法，如Laguerre法、Traub法等。

2.1.3 非线性方程组求解

非线性方程组的求解要比单个的非线性方程求解困难很多：

- 由于这类问题所涉及的范围更加广泛，所以对解的存在性和个数的分析也更加复杂一些。
- 利用传统的数值方法一般无法既绝对安全又收敛保证地产生有解区间。
- 随着问题维数的增加，计算量也将显著增加。

解决非线性方程组的方法也有很多，如不动点迭代、牛顿法、割线修正法、稳健性牛顿法等，这里主要介绍前三种方法。

不动点迭代

给定函数g:$\mathbb{R}^n \to \mathbb{R}^n$，则不动点问题为：寻找$\mathbf{x} \in \mathbb{R}^n$，使

$$\mathbf{x} = g(\mathbf{x}).$$

在一维情形，不动点迭代的收敛与否与收敛速度由g导数的绝对值来决定。而在高维情形下，我们有类似的谱半径条件：

$$\rho(\mathbf{G}(\mathbf{x}^*)) < 1,$$

其中$\mathbf{G}(\mathbf{x})$表示g的雅可比矩阵在x点的值，即

$$\{\mathbf{G}(\mathbf{x})_{ij} = \frac{\partial g_i(\mathbf{x})}{\partial x_j}\}$$

若满足上述条件，当初始向量充分接近解时，不动点迭代收敛，谱半径越小，收敛速度越快。

牛顿法

对可微函数$\mathbf{f}$:$\mathbb{R}^n \to \mathbb{R}^n$，利用泰勒展开可得

$$\mathbf{f}(\mathbf{x}+\mathbf{s}) \approx \mathbf{f}(\mathbf{x}) + \mathbf{J}(\mathbf{x})\mathbf{s},$$

其中$\mathbf{J}(\mathbf{x})\mathbf{s}$是雅可比矩阵，$\{\mathbf{J}(\mathbf{x})\}_{ij} = \partial f_i(\mathbf{x})/\partial x_j$,即可通过这个迭代过程来求出非线性方程组的零点，其算法为：
$\mathbf{x}_0 =$ 初始值
for k=0,1,2,...
解$\mathbf{J}(\mathbf{x}_k)\mathbf{s}_k = -\mathbf{x}_k$求$\mathbf{s}_k$
$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$
end
每一步牛顿法都会解一个线性方程组，因此其运算量是很大的。

割线修正法

与割线法类似，它是将牛顿法中的雅可比矩阵中的偏导数用每个坐标方向上的有限差分近似代替，从而可以节省不少计算成本，其具体算法可表示为：
$\mathbf{x}_0 =$ 初始值
$\mathbf{B}_0 =$ 初始雅可比近似
for k=0,1,2,...
解$\mathbf{B}_k\mathbf{s}_k = -\mathbf{x}_k$求$\mathbf{s}_k$
$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$
$\mathbf{y}_k = \mathbf{f}(\mathbf{x}_{k+1}) - \mathbf{f}(\mathbf{x}_k)$
$\mathbf{B}_{k+1} = \mathbf{B}_k + [(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)\mathbf{S}_k^T]/(\mathbf{s}_k^T\mathbf{s}_k)$
end
这个方法即可加快迭代进程，提升计算的效率。

2.2 函数逼近问题

插值问题的来源于各类实际应用，如要求画出一条通过所有离散点的光滑曲线、利用列表函数求中间值、求列表函数的导数或积分、使用简单函数代替复杂函数，从而可以快速方便的求出需要

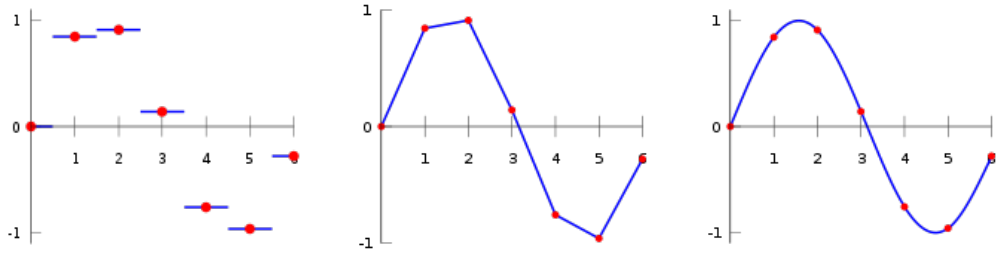
求得的值等等。当然，这些问题又是来自于各类科学与工程计算问题的实际需求中抽象出来的。让用数学的语言进行定义插值问题：

定义2.2.1 (插值问题). 已知数据$\{t_i, y_i\}_{i=0}^n$,寻找函数$f : \mathbb{R} \to \mathbb{R}$,满足：

$$f(t_i) = y_i, \qquad \forall i = 0, 1, ..., n$$

其中f称为插值函数，简称为插值。称$\{t_i\}_{i=0}^n$为插值节点。

如下几个例子描述了如何找到一些“简单”函数来“穿过”给定的数据点： 插值函数的选择主要



考虑因素是多方面的。首先，函数的简单程度是考虑的重要因素，因为越简单的函数越有利于计算机进行求值等各种运算，如多项式是计算机最擅长的数学工具。其次，也要考虑与拟合数据在性态方面的接近程度（如：光滑性，单调性，周期性等），比如分段多项式、三角函数、指数函数和有理函数等也是数值计算中常用的数学工具。

对于给定的数据点集$(t_i, y_i), i = 1, 2, ..., n$,选择基函数$\phi_1(t), \phi_2(t), ...\phi_n(t)$,在这组基函数所张成的函数空间中选择一个插值函数。将插值函数f写成这些基函数的线性组合：

$$f(t) = \sum_{j=1}^{n} x_j \phi_j(t),$$

其中x_j是待定的参数，则数据点(t_i, y_i)上的插值函数f应满足：

$$f(t_i) = \sum_{j=1}^{n} x_j \phi_j(t_i) = y_i,$$

将其写成$\mathbf{Ax} = \mathbf{y}$的矩阵形式，$\mathbf{A}$的元素为$a_{ij} = \phi_j(t_i)$。
利用上述形式，结合线性代数的知识我们有如下结论:

2.2.1 Interpolation with polynomials

最简单直接的基底是单项式基底，即对于n个数据点，进行选取k=n-1，则有单项式基底：

$$\phi_j(t) = t^{j-1}, \quad j = 1, 2, ..., n,$$

任何n-1次多项式，皆可用这个基底的线性组合来表示。此时，插值多项式的参数可由下述方程组求解：

$$\mathbf{Ax} = \begin{bmatrix} 1 & t_1 & ... & t_1^{n-1} \\ 1 & t_2 & ... & t_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_n & ... & t_n^{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \mathbf{y}$$

这里， $\mathbf{A}$是一个范德蒙矩阵，当$x_0 < x_1 < ... < x_n$时非奇异。单项式基底具有存在及唯一性，因此解此方程组需要的工作量是$\mathbf{O}(n^3)$, 计算成本高！

性质2.2.1 (存在唯一性)**.** 若基函数个数n与数据个数m相等，则得到的是一个方阵线性方程组。若矩阵$\mathbf{A}$非奇异，则一定有且仅有唯一解。而若矩阵$\mathbf{A}$奇异，则可以有许多参数的解，代表着数据点不能被精确拟合。

性质2.2.2 (病态性)**.** 基函数可以有许多选择的方式，相对应的会有许多矩阵$\mathbf{A}$的表达形式。$\mathbf{A}$若是单位阵，下三角矩阵，三对角矩阵等等特殊的矩阵，会大大提升求解参数的效率，降低求解的难度，在之后的具体例子里有所体现。

拉格朗日插值公式

对给定的数据点$(t_i, y_i), i = 1, 2, ..., n$,n-1次拉格朗日基函数可以表示为：

$$l_j(t) = \frac{\prod_{k=1,k\neq j}^{n}(t - t_k)}{\prod_{k=1,k\neq j}^{n}(t_j - t_k)}, j = 1, 2, ..., n,$$

显然，我们有结论：

$$l_j(t) = \begin{cases} 1, & i = j, \\ 0, & i \neq j \end{cases} i, j = 1, 2, \ldots, n,$$

说明对于拉格朗日插值来说，$\mathbf{A}$ 是单位矩阵$\mathbf{I}$，因此参数$x_1, x_2, \ldots, x_n$可以直接由y_i得到。拉格朗日插值求参数是很容易的，但是同单项式基底表达式相比，它基函数形式更加复杂，并且积分与微分的操作会困难许多。

牛顿插值公式

对给定的数据点$(t_i, y_i), i = 1, 2, ..., n$, n-1次牛顿基函数为：

$$\pi_j(t) = \prod_{k=1}^{j-1}(t - t_k), \qquad \forall j = 1, 2, ..., n,$$

注意到k始终比j小1，也就是说，多项式可以表示为：

$$Q_n(x) = x_1 + x_2(t - t_1) + x_3(t - t_1)(t - t_2) + \cdots + x_n(t - t_1)(t - t_2)\cdots(t - t_{n-1}). \tag{2.2}$$

这里，$\mathbf{A}$是下三角矩阵，解$\mathbf{Ax} = \mathbf{y}$复杂度$O(n^2)$ 同前两种方法相比，牛顿插值既可以节省计算量，又可以节约计算给定点的值的成本。牛顿法的另一大优势在于，它可以逐步进行计算。如果我们需要增加插值点的个数的时候，可以直接计算增加的点即可，这是其他两种方法做不到的。让我们考虑加入点(t_{n+1}, y_{n+1}),利用牛顿法可得新的插值多项式:

$$Q_{n+1}(t) = Q_n(t) + x_{n+1}\pi_{n+1}(t)$$

其中π的定义与之前的基函数相同，且

$$x_{n+1} = \frac{y_{n+1} - Q_n(t_{n+1})}{\pi_{n+1}(t_{n+1})}.$$

而之前两种插值方式只能将所有点再重新计算，再继续求解参数。

这里我们给出一例以体现三种不同插值公式的区别：

例2.2.1 (三种插值公式的基函数)**.** 分别绘制Langrange,monominal,以及Newton插值公式的基函数图像，并指出它们的区别。若给定数据点$(-1, 1), (0, 0), (1, 1)$，试计算相应的二次插值多项式。

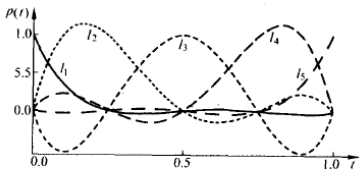


图 7.2 拉格朗日基函数

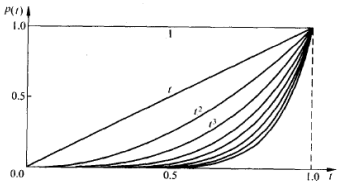


图 7.1 单项式基函数

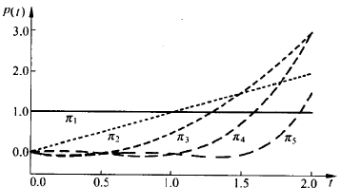


图 7.3 牛顿基函数

差商

差商的定义：$f[x_0,\cdots,x_n]=\dfrac{f[x_1,\cdots,x_n]-f[x_0,\cdots,x_{n-1}]}{x_n-x_0}$

差商的性质:

- $f[x_0,\cdots,x_n]=\dfrac{f^{(n)}(\xi)}{n!}$，其中$\min\{x_i\}<\xi<\max\{x_i\}$;
- 交换不变
- $\frac{\partial}{\partial x}f[x_0,\cdots,x_n,x]=f[x_0,x_1,\cdots,x_n,x,x]$

则可以得到插值多项式：

$$f(x)=P_n(x)+\omega_{n+1}(x)f[x_0,x_1,\cdots,x_n,x]$$

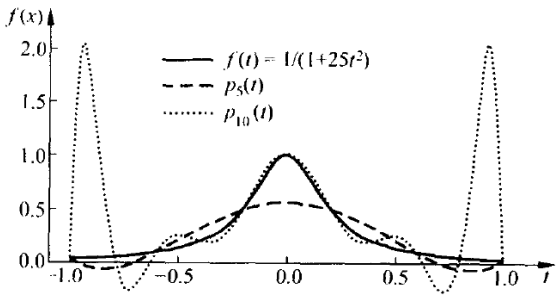
并且此时的误差为：

$$E_{n+1}=f(x)-P_n(x)=\omega_{n+1}(x)f[x_0,x_1,\cdots,x_n,x].$$

其中可以计算出二阶的误差为：$E_2=-\frac{h^2}{8}f''(\xi)$

2.2.2 Interpolation with piecewise polynomials

基函数和数据点的合适选择可以减轻高次多项式插值的困难，但有时候用一个多项式来拟合大量数据点，会出现令人不悦的振荡现象。



分片思想

分段多项式主要思想是: 对给定的数据点集$(t_i,y_i),i=1,\cdots,n,\quad t_1<t_2<\cdots<t_n$,作分段多项式插值时，在每个子区间$[t_i,t_{i+1}]$上用不同的多项式。典型的分段插值方法包括三次样条插值和三次Hermit插值，如下图所示：

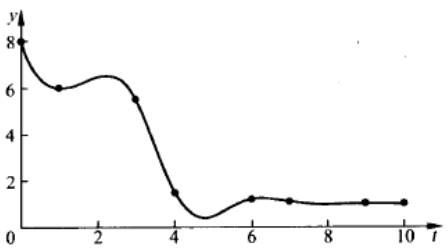


图 7.10 单调数据的三次样条插值

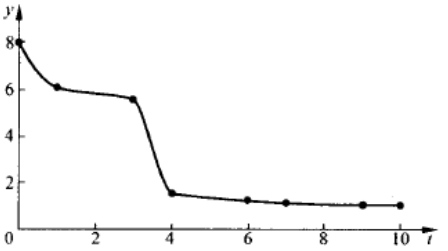


图 7.9 单调数据的三次埃尔米特插值

t	-1.0	-0.5	0.0	0.5	1.0
y	1.0	0.5	0.0	0.5	2.0

B-样条

采用B样条作为插值基函数有很多优点。

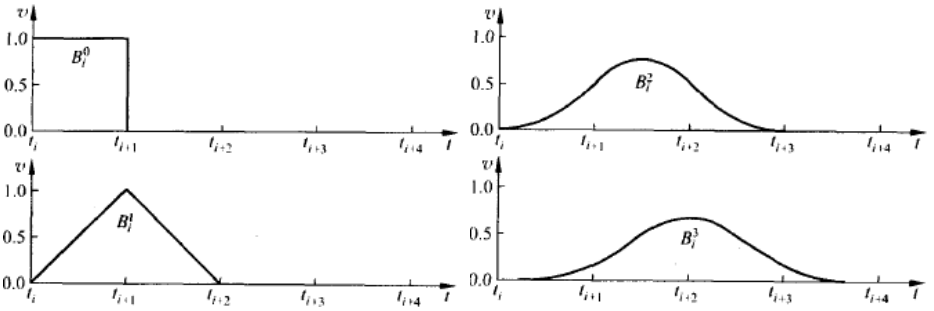
作为递推初始值，定义 0 次 B-样条

$$B_i^0(t)=\begin{cases}1, & t_i\leqslant t<t_{i+1},\\ 0, & \text{其他.}\end{cases}$$

对 $k>0$，定义 k 次 B-样条为

$$B_i^k(t)=v_i^k(t)B_i^{k-1}(t)+(1-v_{i+1}^k(t))B_{i+1}^{k-1}(t).$$

由于 B_i^0 是分段常数，v_i^k 是线性的，从定义可知，B_i^1 是分段线性的. 类似地，B_i^2 是分段二次

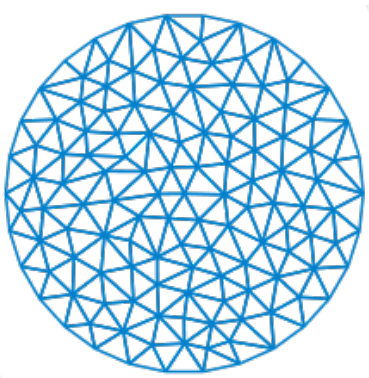


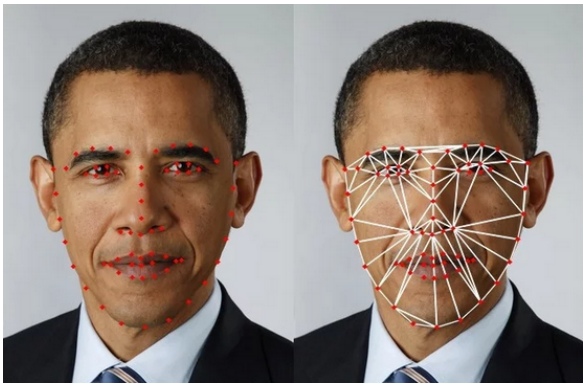
关于回归、插值、逼近、拟合的区别：插值：插值曲线要经过型值点;逼近：为复杂函数寻找近似函数，其误差在某种度量意义下最小;拟合：在插值问题中考虑**给定数据点的误差**，其误差在某种度量意义下最小;回归：最小二乘解,在$\|\cdot\|_2$度量意义下最小。

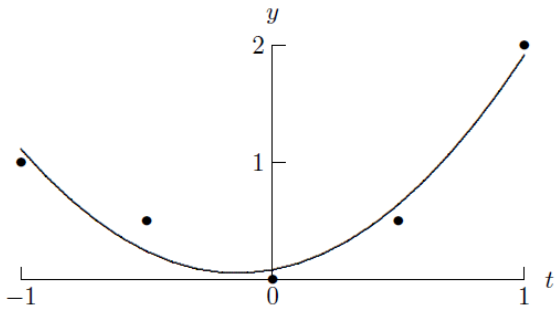
例2.2.2 (多变量逼近问题：平面网格三角剖分). 平面区域可被一系列直边三角形覆盖.当三角形的外接圆直径趋向于0时,空洞区域的面积趋向于0.

2.2.3 Least square approximation

例2.2.3 (数据拟合). 找一条二次曲线逼近如下观测点： 类似问题：多元线性回归、非多项式数据拟合等







2.2.4 Best approximation

2.3 数值求积问题

2.3.1 积分问题

积分的应用十分广泛，在各个领域都有一些应用，主要在以下几个方面：积分变换，如拉普拉斯变换，傅里叶变换、偏微分方程的有限元法和边界元法、积分方程和变分法、概率统计，其中总的许多基本概念，如概率分布、期望等由积分定义。此外，在古典和量子物理中的系统能量定义。

定义2.3.1 (积分). 对区间$[a,b]$上的函数f:$\mathbb{R}\to\mathbb{R}$,现代积分定义

$$\mathbf{I}(f)=\int_a^b f(x)dx$$

是以形如下式的黎曼和为基础的

$$R_n=\sum_{i=1}^{n}(x_{i+1}-x_i)f(\xi_i),$$

其中$a=x_1<x_2<\cdots<x_n<x_{n+1}=b$, $\xi\in[x_i,x_{i+1}]$.
记$h_n=\max\limits_{i=1,\ldots,n-1}\{x_{i+1}-x_i\}$,若

$$\lim_{h_n\to 0}R_n=R\neq\infty,$$

则称f在区间上黎曼可积。

杆的分散载荷问题. 图 8.1 中，若$\rho(x)$为载荷密度关于横坐标x的函数，则杆受到的总载荷W为曲线$\rho(x)$所围区域的面积，由积分

$$W=\int_0^L \rho(x)\mathrm{d}x$$

给出. 而且，等价的集中载荷的作用点为曲线所围区域的重心，其横坐标$\bar{x}$由积分

$$\bar{x}=\frac{1}{W}\int_0^L x\rho(x)\mathrm{d}x$$

给出.

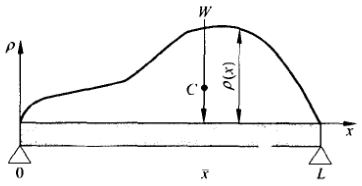


图 8.1 木杆上的分散载荷与集中载荷等价

很久以前就有人提出了近似计算不规则区域面积的方法，其中包括阿基米德(Archimedes)，他的方法是将区域用小正方形铺贴，然后计算区域内小方块的个数.

积分（模型/方法）的正则性（存在/唯一/条件数）是一个很重要的理论问题。
存在性: f 有界、连续$\rightarrow$ Riemann可积。唯一性: 极限存在性与剖分无关、与ξ_i取法无关。
问题的刚性: 数据扰动/误差传播分析。我们通过如下两种途径来分析误差的扰动：

被积函数的扰动

$$|I(\hat{f})-I(f)|\leq\int_a^b|\hat{f}(x)-f(x)|dx\leq(b-a)\|\hat{f}-f\|_\infty$$

积分区间的扰动

$$\left|\int_a^{\hat{b}}f(x)dx-\int_a^b f(x)dx\right|\leq\int_b^{\hat{b}}f(x)dx\leq(\hat{b}-b)\|f\|_\infty$$

2.3.2 数值积分公式

定积分的数值求解称为数值求积。定积分：

$$I(f)=\int_a^b f(x)dx,$$

数值求积的基本想法是，

定义2.3.2 (数值积分). 寻找一个$I(f)$的近似:

$$Q_n(f)=\sum_{i=1}^n\omega_i f(x_i),$$

其中$a < x_1 < \cdots < x_n < b$,称为节点，$\omega$称为权。数值求积的目标则是选择节点和权，利用合理的计算成本来求得达到满足要求的精度的近似$Q_n(f)$。

MATLAB: ¿¿ help quad提供了一些很好的

```
Q = quad(FUN,A,B,TOL) uses an absolute error tolerance of TOL
instead of the default, which is 1.e-6.  Larger values of TOL
result in fewer function evaluations and faster computation,
but less accurate results.  The quad function in MATLAB 5.3 used
a less reliable algorithm and a default tolerance of 1.e-3.
Example:
   Q = quad(@myfun,0,2);
where the file myfun.m defines the function:
   %------------------%
   function y = myfun(x)
   y = 1./(x.^3-2*x-5);
   %------------------%

or, use a parameter for the constant:
   Q = quad(@(x)myfun2(x,5),0,2);
where the file myfun2.m defines the function:
   %-----------------------%
   function y = myfun2(x,c)
   y = 1./(x.^3-2*x-c);
   %-----------------------%
See also integral, integral2, integral3, quadgk, quad2d, trapz,
```

2.3.3 插值型公式(Newton-Cotes系列)

求积公式的节点选择[a,b]上等距的节点，即：

$$x_i = a + i(b-a)/(n+1), i = 1, 2, \cdots, n.$$

梯形(Trapzoid)公式

取两个节点，取权值为两点函数值的平均值，则可以得到：

$$Q_n(f) = (b-a)\frac{f(a)+f(b)}{2}$$

误差：

$$E_2 = \left| \int_a^b f(x)dx - Q_n(f) \right| = \int_a^b \frac{(b-1)^2}{8} f''(\xi)dx = \frac{(b-a)^3}{8}\|f''\|_\infty$$

复化梯形公式($n+1$个节点)：

$$T_n = h\sum_{i=0}^{n-1}\frac{f(x_i)+f(x_{i+1})}{2} = \frac{b-a}{n}\left[\frac{1}{2}f(x_0) + f(x_1) + \cdots + f(x_{n-1}) + \frac{1}{2}f(x_n)\right]$$

求积公式可以由多项式插值得到. 事实上，利用被积函数 f 在点 $x_i(i=1,\cdots,n)$ 上的值，可以确定被积函数在这 n 个点上的 $n-1$ 次插值多项式(见第 7 章)，然后用插值多项式的积分作为原始函数积分的近似. 事实上，具体计算积分时，并不需要直接得到插值多项式. 相反，可以用插值多项式确定求积公式的节点和权，由节点和权可以计算积分区间上任何函数积分的近似值. 特别地，若使用拉格朗日插值，则对给定点集 $x_i(i=1,\cdots,n)$，可用其相应的拉格朗日基函数的积分确定权，

$$w_i = \int_a^b l_i(x)\mathrm{d}x, \qquad i = 1,\cdots,n.$$

可以看出，对任何被积函数这些权都是相同的. 因此，这种求积公式称为插值型的.

此时的误差为：$E_{rr}^{T}=-\frac{(b-a)\|f''\|_{\infty}}{12}h^2$

待定系数法

得到插值型求积公式的另一种方法是待定系数法，它是通过使积分公式对前 n 个多项式基函数精确成立，产生一个 n 元方程组，从而得到权. 例如，若使用单项式基底，得到的是力矩方程组

$$\begin{aligned}
&w_1\cdot 1+w_2\cdot 2+\cdots+w_n\cdot 1=\int_a^b 1\,\mathrm{d}x=b-a,\\
&w_1\cdot x_1+w_2\cdot x_2+\cdots+w_n\cdot x_n=\int_a^b x\,\mathrm{d}x=(b^2-a^2)/2,\\
&\qquad\vdots\\
&w_1\cdot x_1^{n-1}+w_2\cdot x_2^{n-1}+\cdots+w_n\cdot x_n^{n-1}=\int_a^b x^{n-1}\,\mathrm{d}x=(b^n-a^n)/n,
\end{aligned}$$

例：在区间$[a,b]$上构造三节点积分公式：

$$Q_3(f)=w_1f(x_1)+w_2f(x_2)+w_3f(x_3),$$

取三个节点分别为积分区间的两个端点及中点，即 $x_1=a$、$x_2=(a+b)/2$、$x_3=b$. 写成矩阵形式，力矩方程为范德蒙方程组

$$\begin{bmatrix}1 & 1 & 1\\ a & (a+b)/2 & b\\ a^2 & ((a+b)/2)^2 & b^2\end{bmatrix}\begin{bmatrix}w_1\\ w_2\\ w_3\end{bmatrix}=\begin{bmatrix}b-a\\ (b^2-a^2)/2\\ (b^3-a^3)/3\end{bmatrix}.$$

用高斯消去法求解，得到权

$$w_1=(b-a)/6,\qquad w_2=2(b-a)/3,\qquad w_3=(b-a)/6.$$

辛普生(Simpson)公式

$$S(f)=\frac{b-a}{6}\left[f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right]$$

误差：

$$E_3=\left|\int_a^b f(x)dx-S(f)\right|=-\frac{(b-a)^5}{2880}f^{(4)}(\eta),a<\eta<b$$

复化Simpson公式($n+1$个节点)：

$$S_n=\frac{h}{6}\sum_{i=0}^{n-1}\left[f(x_i)+4f(x_{i+\frac{1}{2}})+f(x_{i+1})\right].$$

此时的误差为：$E_{rr}^S = -\frac{(b-a)\|f^{(4)}\|_\infty}{2880}h^4$.

例 8.2 牛顿-柯特斯求积 为说明牛顿-柯特斯求积公式的应用，用上面三种求积公式近似计算积分

$$I(f) = \int_0^1 e^{-x^2}\,dx.$$

$$M(f) = (1-0)\exp(-0.25) \approx 0.778801,$$

$$T(f) = \frac{1}{2}(\exp(0) + \exp(-1)) \approx 0.683940,$$

$$T(f) = \frac{1}{6}(\exp(0) + 4\exp(-0.25) + \exp(-1)) \approx 0.747180.$$

图 8.2 画出了被积函数及每个公式中的插值多项式. 这个问题的精确值为 0.746824. 令人惊讶的是梯形公式误差的绝对值(0.062884)大约是中点公式(0.031977)

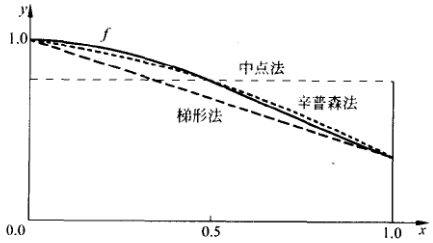


图 8.2 用牛顿-柯特斯求积公式计算 $f(x)=e^{-x^2}$ 的积分

的两倍，而辛普森公式的误差仅为 0.000356，若再考虑积分区间的长度，可以看到辛普森公式已相当精确. ■

2.3.4 提高积分精度的方法

复化同理于分段插值

自适应方法- 区间分半递归

Richardson外推- Romberg积分法（基于Newton-Cotes系列的外推）

其他典型积分相关问题。基于表格化数据求积分– 先插值，再积分。奇异积分– 取极限、变量代换。二/三重积分– 张量积、单纯型高维积分– Monte Carlo方法。积分方程– 通常是ill-conditioned。

2.3.5 最佳求积

固定节点的数值积分公式精度有限，并没有充分利用已有的“信息”。

Clenshaw-Curtis求积

一些研究发现：用Chebyshev多项式零点作为积分节点的时候可以证明：积分权重总是正数！其次，有效的自包含求法：不用事先求得权重。可以利用FFT和递推算法，代数精度不是最优阶！

高斯求积

基本思想:选择最佳的节点和权值，使其代数精度最大化。
以构造[-1,1]上的两点公式$G_2(f)$为例：

$$I(f) = \int_{-1}^{1} f(x)dx \approx \omega_1 f(x_1) + \omega_2 f(x_2) := G_2(f),$$

令其对前四阶单项式精确成立：

$$\begin{cases}\omega_1+\omega_2=\int_{-1}^{1}1dx=2,\\ \omega_1x_1+\omega_2x_2=\int_{-1}^{1}xdx=0,\\ \omega_1x_1^2+\omega_2x_2^2=\int_{-1}^{1}x^2dx=\frac{2}{3},\\ \omega_1x_1^3+\omega_2x_2^3=\int_{-1}^{1}x^3dx=\frac{2}{3},\end{cases}$$

可解得

$$x_1=-1/\sqrt{3},x_2=1/\sqrt{3},\omega_1=1,\omega_2=1$$

即得到了三阶两点高斯求积公式为：

$$G_2(f)=f(-1/\sqrt{3})+f(1/\sqrt{3}).$$

使用高斯求积公式也需要注意如下几点：如何通过Legendre多项式等典型的正交多项式零点计算；只在标准区间上给出，实际使用需放射变换；Gauss-Kronrod递推型高斯求积。

2.3.6 突破维数诅咒： Monte Carlo求积方法

软件包与参考材料

Exercise