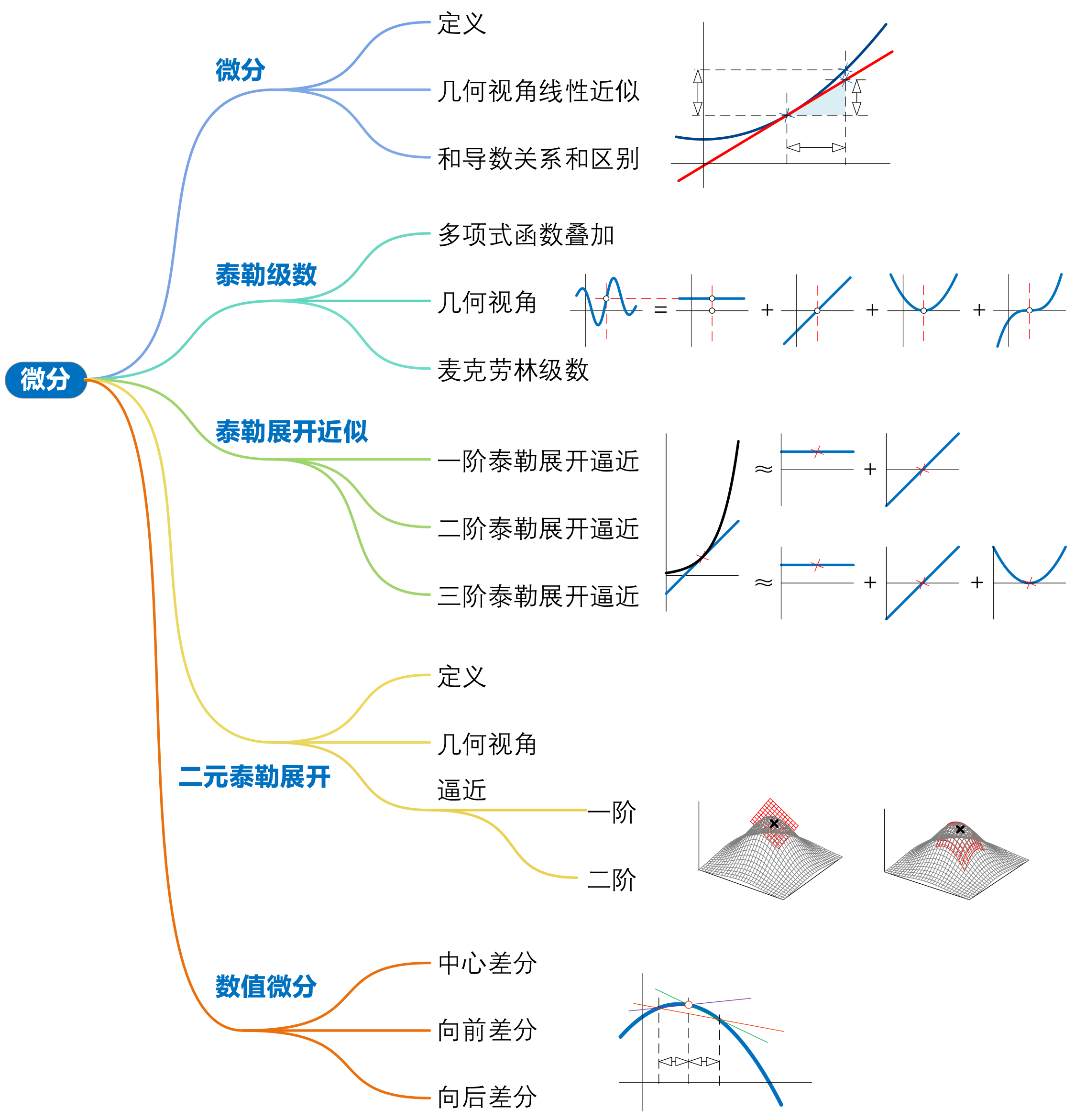
17 Differential 微分

微分是线性近似

> 我看的比别人更远，那是因为我站在一众巨人们的臂膀之上。
>
> ***If I have seen further than others, it is by standing upon the shoulders of giants.***
>
> —— 艾萨克·牛顿 (Isaac Newton) | 英国数学家、物理学家 | 1643 ~ 1727

- ◀ `numpy.meshgrid()` 获得网格数据
- ◀ `sympy.abc import x` 定义符号变量 x
- ◀ `sympy.abc import x` 定义符号变量 x
- ◀ `sympy.diff()` 求解符号导数和偏导解析式
- ◀ `sympy.evalf()` 将符号解析式中未知量替换为具体数值
- ◀ `sympy.lambdify()` 将符号表达式转化为函数
- ◀ `sympy.series()` 求解泰勒展开级数符号式
- ◀ `sympy.symbols()` 定义符号变量


微分
微分
定义
几何视角线性近似
和导数关系和区别
泰勒级数
多项式函数叠加
几何视角
麦克劳林级数
泰勒展开近似
一阶泰勒展开逼近
二阶泰勒展开逼近
三阶泰勒展开逼近
二元泰勒展开
定义
几何视角
逼近
一阶
二阶
数值微分
中心差分
向前差分
向后差分

17.1 几何角度看微分：线性近似

微分 (differential) 是函数的局部变化的一种线性描述。如图 1 所示，微分可以近似地描述当函数自变量取值出现足够小 Δx 变化时，函数值变化 Δy。

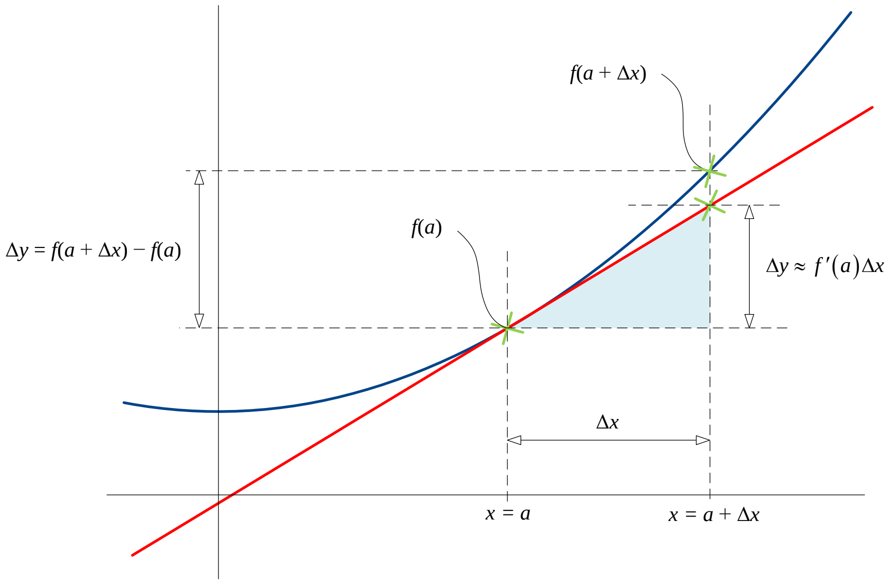


图 1. 对一元函数来说，微分是线性近似

假设函数 $f(x)$ 在某个区间内有定义。给定该区间内一点 a，当 a 变动到 $a + \Delta x$ (也在该区间内) 时，函数实际增量为 Δy：

$$\Delta y = f(a+\Delta x) - f(a) \tag{1}$$

而增量 Δy 可以近似为：

$$\Delta y = f(a+\Delta x) - f(a) \approx f'(a)\Delta x \tag{2}$$

其中，$f'(a)$ 为函数在 $x = a$ 处一阶导数。本书第 15 章讲过，函数 $f(x)$ 在某一点处一阶导数值是函数在该点处切线的斜率值。

整理上式，$f(a + \Delta x)$ 的近似写成：

$$f(a+\Delta x) \approx f'(a)\Delta x + f(a) \tag{3}$$

令

$$x = a + \Delta x \tag{4}$$

(3) 可以写成：

$$f(x) \approx f'(a)(x-a) + f(a) \tag{5}$$

上式就是一次函数的点斜式。一次函数通过 $(a, f(a))$ 这点，斜率为 $f'(a)$。

如图 1 所示，从几何角度，微分用切线这条斜线代替曲线。实践中，复杂的非线性函数可以通过局部线性化来简化运算。

图 2 和图 3 分别所示为高斯函数和其一阶导数函数在若干点处的切线。

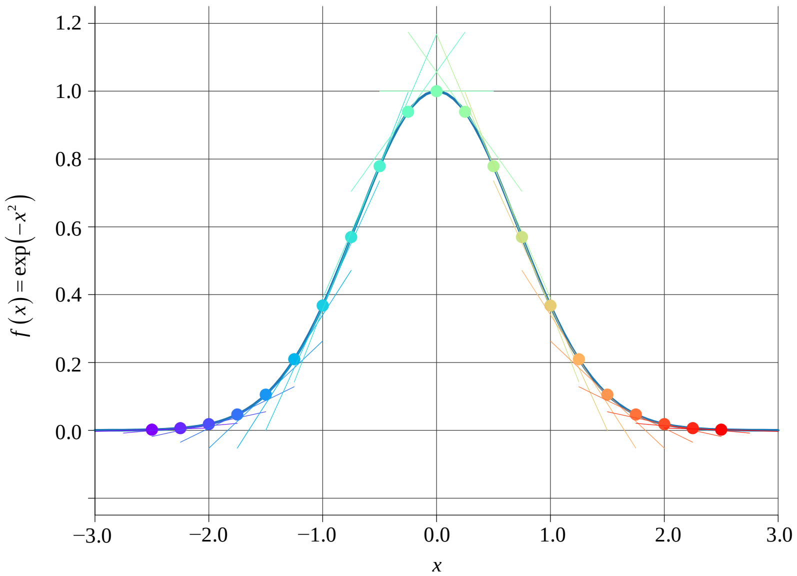


图 2. 高斯函数不同点处切线

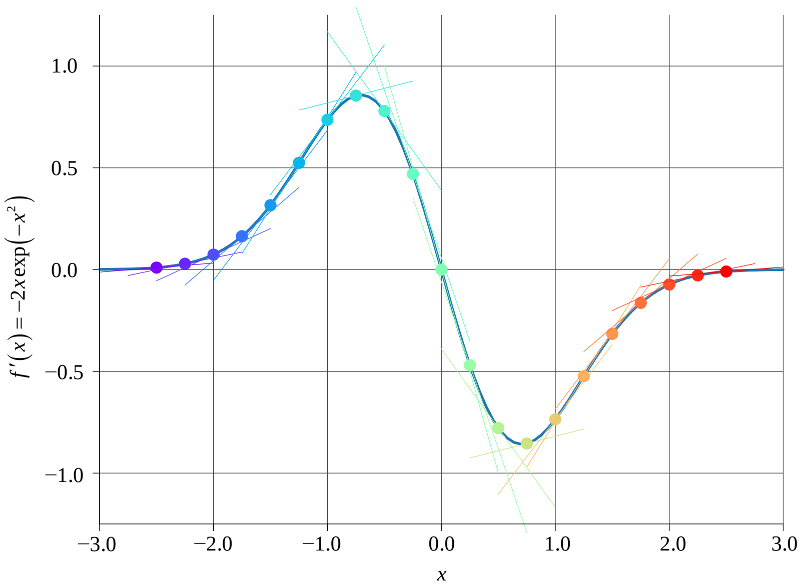


图 3. 高斯函数一阶导数不同点处切线

17.2 泰勒级数：多项式函数近似

英国数学家布鲁克·泰勒 (Sir Brook Taylor) 在 1715 年发表了泰勒级数 (Taylor's theorem)。泰勒级数是一种强大的函数近似工具。

布鲁克·泰勒 (Brook Taylor)
英国数学家 | 1685年 ~ 1731年
以泰勒公式和泰勒级数闻名

当展开点 (expansion point) 为 $x = a$ 时，一元函数 $f(x)$ 泰勒展开 (Taylor expansion) 形式为：

$$\begin{aligned} f(x) &= \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n \\ &= \underbrace{f(a)}_{\text{Constant}} + \underbrace{\frac{f'(a)}{1!}(x-a)}_{\text{Linear}} + \underbrace{\frac{f''(a)}{2!}(x-a)^2}_{\text{Quadratic}} + \underbrace{\frac{f'''(a)}{3!}(x-a)^3}_{\text{Cubic}} + \cdots \end{aligned} \tag{6}$$

其中，a 为展开点 (expansion point)。式中的阶乘是多项式求导产生的。展开点为 0 的泰勒级数又叫做麦克劳林级数 (Maclaurin series)。

如图 4 所示，泰勒展开相当于一系列多项式函数叠加，用来近似某个复杂函数。

⚠ 注意，图中常数函数图像对应的高度 $f(a)$ 提供了 $x = a$ 处 $f(x)$ 的函数值。而剩余其他多项式函数在展开点 $x = a$ 处函数值均为 0。

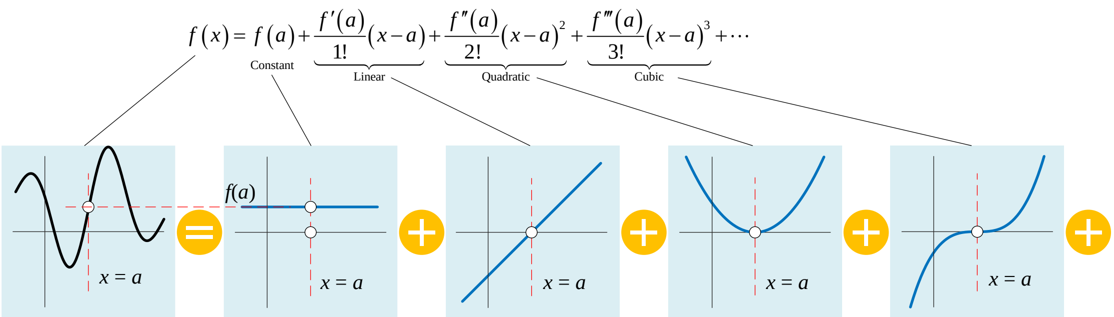


图 4. 一元函数泰勒展开原理

实际应用中，在应用泰勒公式近似计算时需要截断，也就是只取有限项。

上一节介绍微分时，(5) 实际上就是泰勒公式取前两项，即用“常数函数 + 一次函数”叠加近似原函数 $f(x)$：

$$f(x) \approx \underbrace{f(a)}_{\text{Constant}} + \underbrace{\frac{f'(a)}{1!}(x-a)}_{\text{Linear}} = f(a) + f'(a)(x-a) \tag{7}$$

在 (7) "常数函数 + 一次函数" 基础上，再增加"二次函数"成分，我们便得到二次近似：

$$f(x) \approx \underbrace{f(a)}_{\text{Constant}} + \underbrace{\frac{f'(a)}{1!}(x-a)}_{\text{Linear}} + \underbrace{\frac{f''(a)}{2!}(x-a)^2}_{\text{Quadratic}} \tag{8}$$

图 5 和图 6 分别所示为高斯函数和其一阶导数函数的二次近似。泰勒公式把复杂函数转换为多项式叠加。相较其他函数，多项式函数更容易计算微分、积分。本章后续将会介绍利用泰勒展开近似。

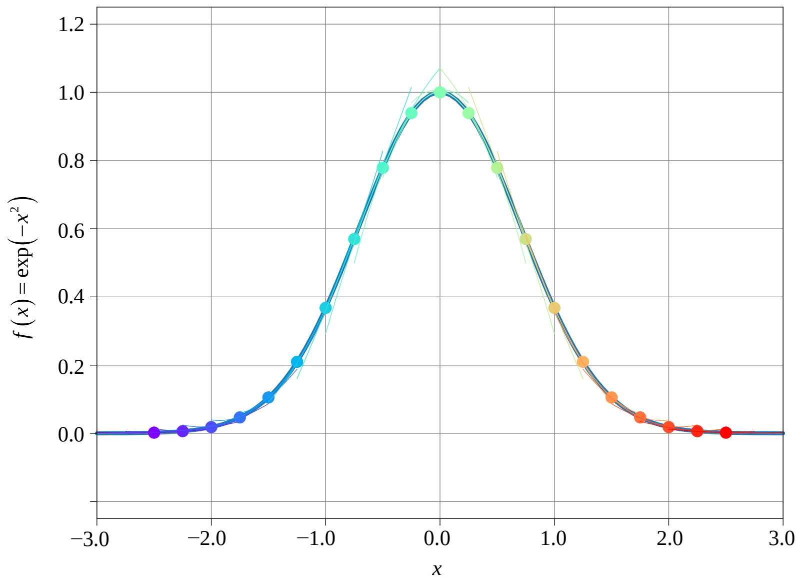


图 5. 高斯函数不同点处二次近似

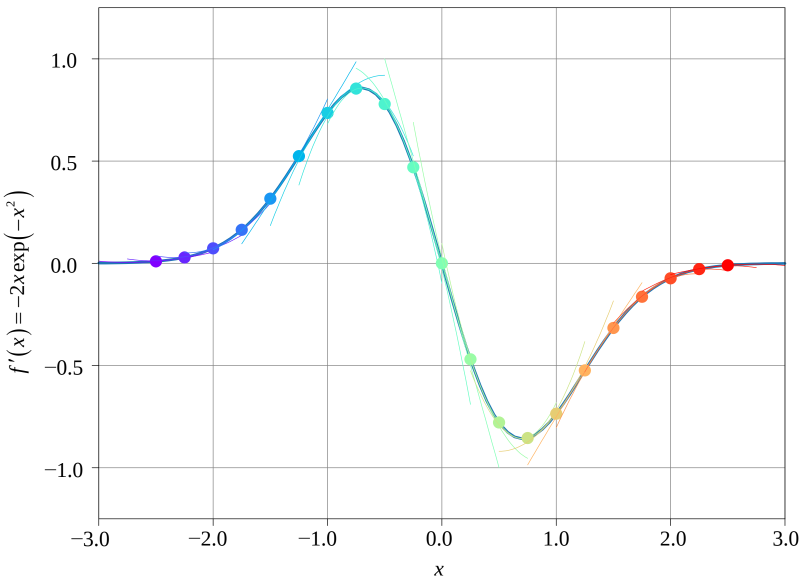


图 6. 高斯函数一阶导数不同点处二次近似

Bk3_Ch17_01.py 绘制图 5 和图 6。

在 Bk3_Ch17_01.py 基础上，我们做了一个 App 展示曲线上不同点一次和二次近似。请参考 Streamlit_Bk3_Ch17_01.py。

17.3 多项式近似和误差

再次强调，泰勒展开的核心是用一系列多项式函数叠加来逼近某个函数。实际应用中，泰勒级数常用来近似计算复杂非线性函数，并估计误差。

给定原函数 $f(x)$ 为自然指数函数：

$$f(x) = \exp(x) = \mathrm{e}^x \tag{9}$$

在 $x = 0$ 处，该函数的泰勒级数展开为：

$$\mathrm{e}^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \cdots = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \cdots \tag{10}$$

如前文所述，在具体应用场合，泰勒公式需要截断，只取有限项进行近似运算。一个函数的有限项的泰勒级数叫做泰勒展开式。

常数函数

在 $x = 0$ 点处，$f(x)$ 函数值为：

$$f(0) = \exp(0) = 1 \tag{11}$$

图 7 所示为用常数函数来近似原函数：

$$f_0(x) = \underbrace{1}_{\text{Constant}} \tag{12}$$

图 8 比较原函数和常数函数，并给出误差随 x 变化。常数函数为平行横轴的直线，它的估计能力显然明显不足。

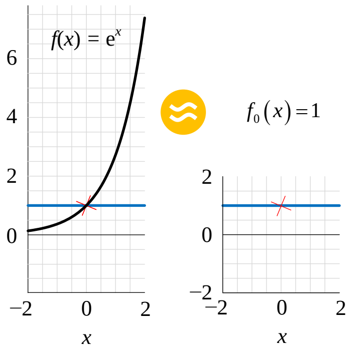


图 7. 常数函数近似

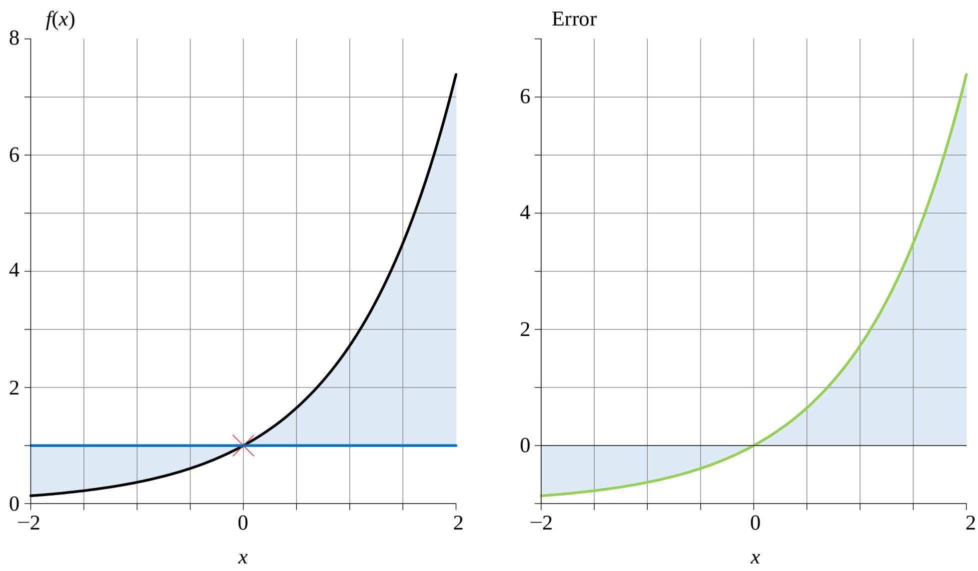


图 8. 常数函数近似及误差

一次函数

原函数 $f(x)$ 一阶导数为：

$$f'(x) = \exp(x) \tag{13}$$

$x = 0$ 处一阶导数为切线斜率：

$$f'(0) = \exp(0) = 1 \tag{14}$$

用一次函数来近似原函数：

$$f_1(x) = \underbrace{1}_{\text{Constant}} + \underbrace{x}_{\text{Linear}} \tag{15}$$

图 9 所示为"常数函数 + 一次函数"近似的原理。

叠加常数函数和一次函数，常被称作一阶泰勒展开 (first-order Taylor polynomial/expansion/approximation 或 first-degree Taylor polynomial)。一阶泰勒展开是最常用的逼近手段。

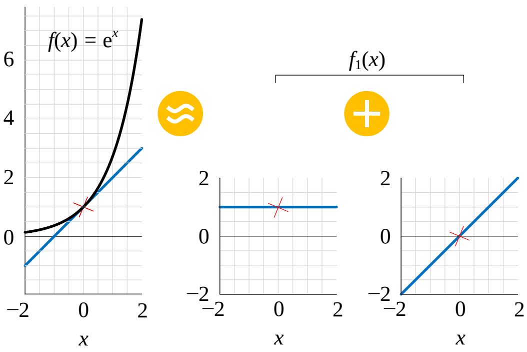


图 9. “常数函数 + 一次函数”近似

原函数和泰勒多项式的差被称作为泰勒公式的余项，即误差：

$$R(x) = f(x) - f_1(x) = f(x) - \left(\underbrace{1}_{\text{Constant}} + \underbrace{x}_{\text{Linear}} \right) \tag{16}$$

图 10 所示为一阶泰勒展开近似和误差；离展开点 $x = a$ 越远，误差越大。

也就是说，非线性函数在 $x = a$ 附近可以用这个一次函数近似。当 x 远离 a，这个近似就变得越不准确。

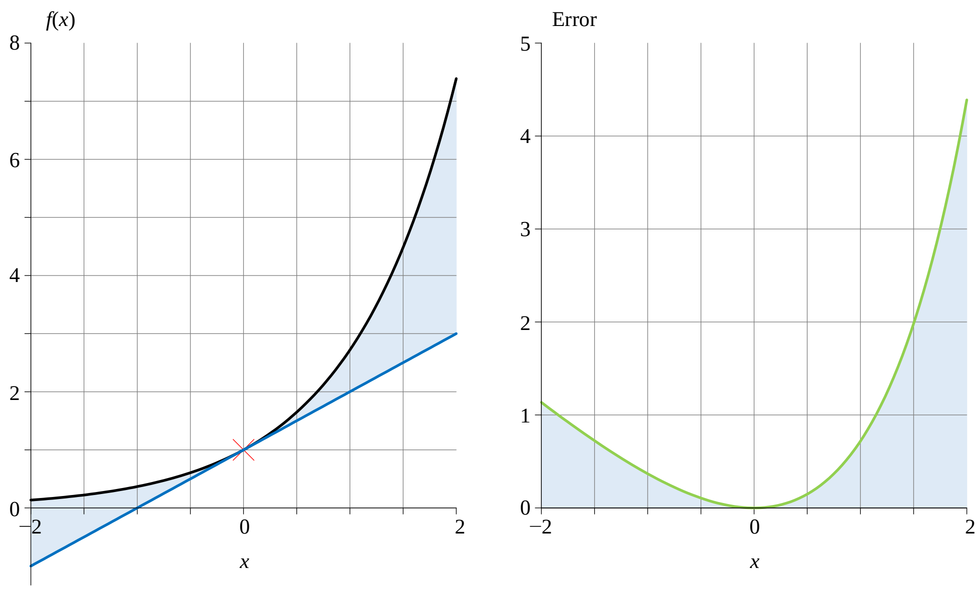


图 10. 一阶泰勒展开近似及误差

二次函数

用二次多项式函数近似原函数：

$$f_2(x) = \underbrace{1}_{\text{Constant}} + \underbrace{x}_{\text{Linear}} + \underbrace{\frac{x^2}{2}}_{\text{Quadratic}} \tag{17}$$

上式也叫二阶泰勒展开 (second-order Taylor polynomial/expansion/approximation)。图 11 所示，二阶泰勒展开叠加了三个成分，“常数函数 + 一次函数 + 二次函数”。

图 12 给出的是二阶泰勒展开近似及误差。相较图 10，图 12 中误差明显变小。

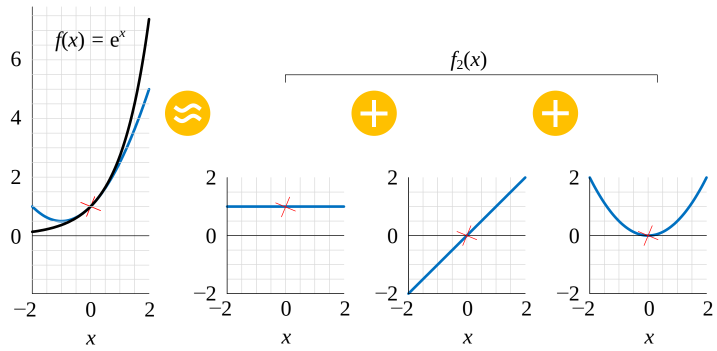


图 11. “常数函数 + 一次函数 + 二次函数”近似原函数

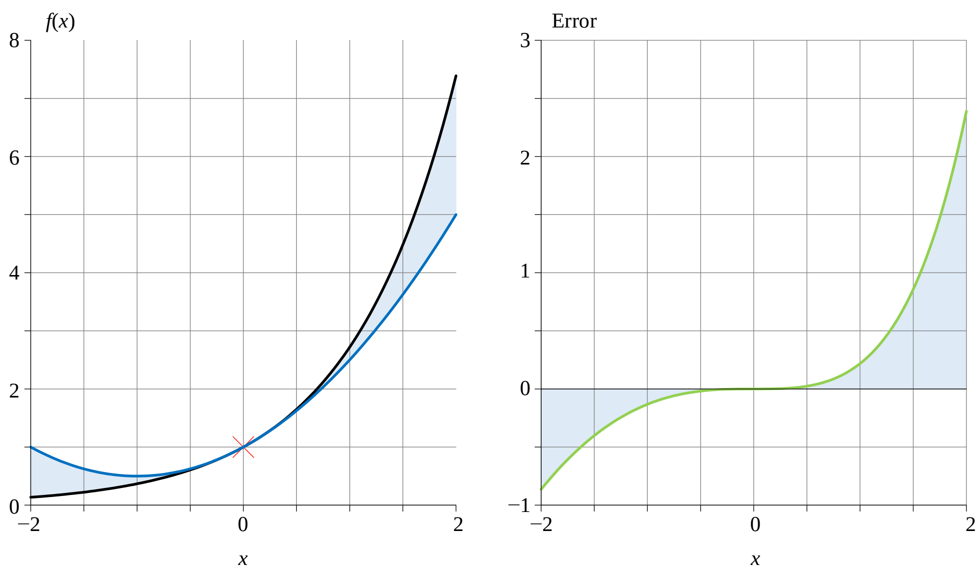


图 12. 二阶泰勒展开近似及误差

三次函数

用三次多项式函数近似原函数：

$$f_3(x) = \underbrace{1}_{\text{Constant}} + \underbrace{x}_{\text{Linear}} + \underbrace{\frac{x^2}{2}}_{\text{Quadratic}} + \underbrace{\frac{x^3}{6}}_{\text{Cubic}} \tag{18}$$

图 13 所示为"常数函数 + 一次函数 + 二次函数 + 三次函数"叠加近似原函数。比较图 12 和图 14，增加三次项后，逼近效果提高，误差进一步减小。

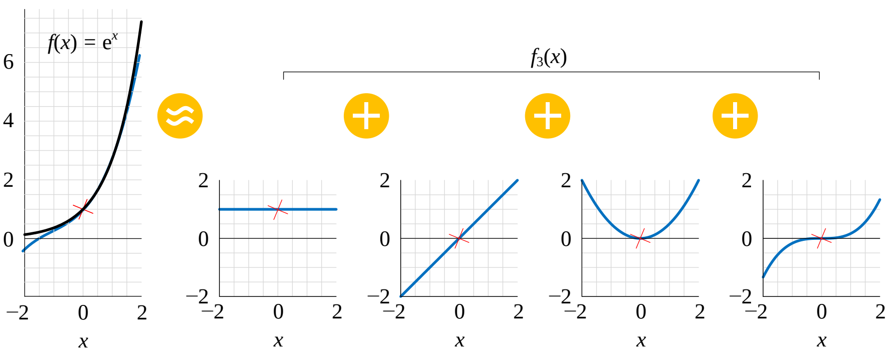


图 13. "常数函数 + 一次函数 + 二次函数 + 三次函数"近似原函数

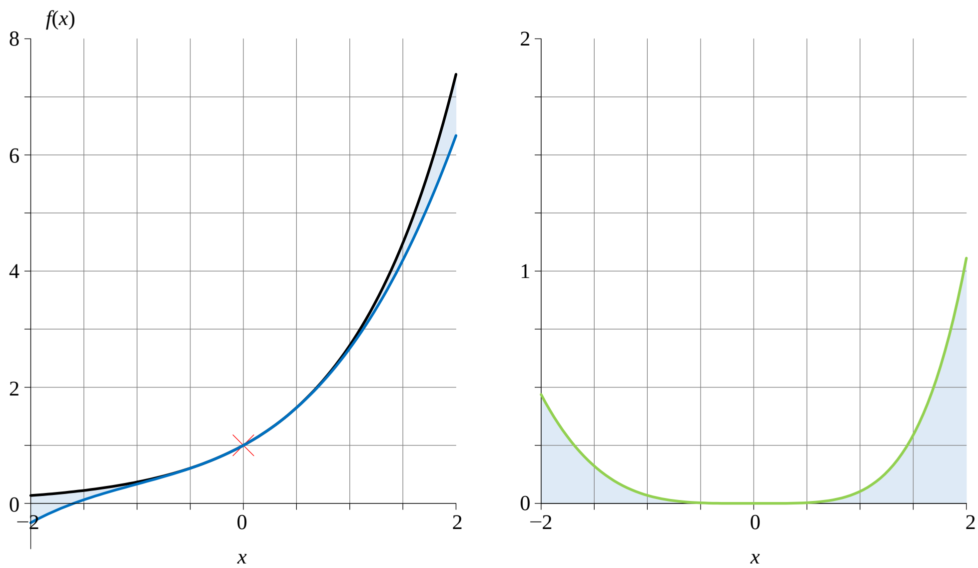


图 14. 三函数近似原函数及误差

四次函数

图 15 所示为用四次多项式函数近似原函数：

$$f(x) = \exp(x) \approx f_4(x) = \underbrace{1}_{\text{Constant}} + \underbrace{x}_{\text{Linear}} + \underbrace{\frac{x^2}{2}}_{\text{Quadratic}} + \underbrace{\frac{x^3}{6}}_{\text{Cubic}} + \underbrace{\frac{x^4}{24}}_{\text{Quartic}} \tag{19}$$

一般来说，泰勒多项式展开的项数越多，也就是说多项式幂次越高，逼近效果越好；但是，实际应用中，线性逼近和二次逼近用的最为广泛。对于误差分析本书不做探讨，对于误差分析感兴趣的同学可以自行学习。

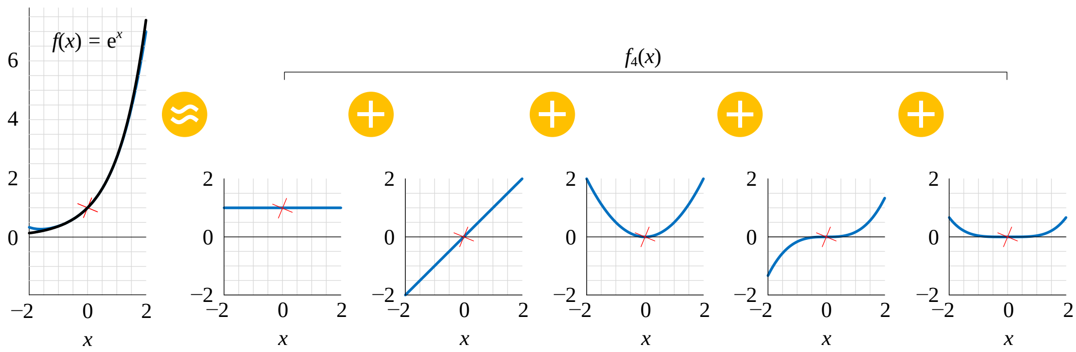


图 15.“常数函数 + 一次函数 + 二次函数 + 三次函数 + 四次函数”近似原函数

Bk3_Ch17_02.py 绘制本节图像；请大家改变展开点位置，比如 $x = 1$、$x = -1$，并观察比较近似及误差。

在 Bk3_Ch17_02.py 基础上，我们做了一个 App 比较不同泰勒展开项数对逼近结果的影响。请参考 Streamlit_Bk3_Ch17_02.py。

17.4 二元泰勒展开：用多项式曲面近似

上一节介绍的一元泰勒展开也可以扩展到多元函数。本节以二元函数为例介绍多元函数泰勒展开。

给定二元函数 $f(x_1, x_2)$，它的泰勒展开可以写成：

$$
\begin{aligned}
f(x_1,x_2) = & \underbrace{f(a,b)}_{\text{Constant}} + \underbrace{f_{x1}(a,b)(x_1-a) + f_{x2}(a,b)(x_2-b)}_{\text{Plane}} \\
& + \underbrace{\frac{1}{2!}\left[f_{x1x1}(a,b)(x_1-a)^2 + 2f_{x1x2}(a,b)(x_1-a)(x_2-b) + f_{x2x2}(a,b)(x_2-b)^2\right]}_{\text{Quadratic}} + \cdots
\end{aligned}
\tag{20}
$$

⚠ 注意，式中假定两个混合偏导相同，即 $f_{x1x2}(a, b) = f_{x2x1}(a, b)$。

将 (20) 写成矩阵运算形式：

$$
f(x_1,x_2) = f(a,b) + \begin{bmatrix} f_{x1}(a,b) \\ f_{x2}(a,b) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} x_1-a \\ x_2-b \end{bmatrix} + \frac{1}{2!}\begin{bmatrix} x_1-a \\ x_2-b \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} f_{x1x1}(a,b) & f_{x2x1}(a,b) \\ f_{x1x2}(a,b) & f_{x2x2}(a,b) \end{bmatrix} \begin{bmatrix} x_1-a \\ x_2-b \end{bmatrix} + \ldots \tag{21}
$$

如图 16 所示，从几何角度，二元函数泰勒展开相当于，水平面、斜面、二次曲面、三次曲面等多项式曲面叠加。

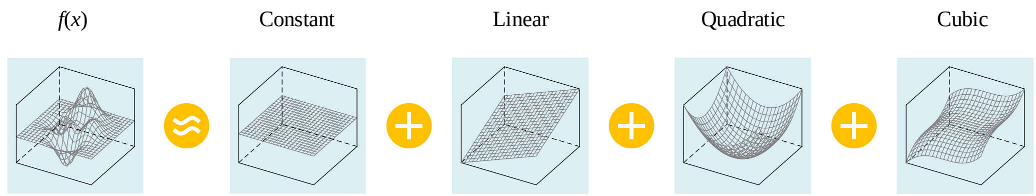


图 16. 二元函数泰勒展开原理

举个例子

给定如下二元高斯函数：

$$
y = f(x_1,x_2) = \exp\left(-\left(x_1^2 + x_2^2\right)\right) \tag{22}
$$

二元函数 $f(x_1, x_2)$ 一阶偏导：

$$
\begin{aligned}
f_{x1}(x_1,x_2) &= \frac{\partial f}{\partial x_1}(x_1,x_2) = -2x_1\exp\left(-\left(x_1^2 + x_2^2\right)\right) \\
f_{x2}(x_1,x_2) &= \frac{\partial f}{\partial x_2}(x_1,x_2) = -2x_2\exp\left(-\left(x_1^2 + x_2^2\right)\right)
\end{aligned}
\tag{23}
$$

图 17 所示为两个一阶偏导函数的平面等高线图；图中 × 为展开点位置，水平面位置坐标 (−0.1, −0.2)。

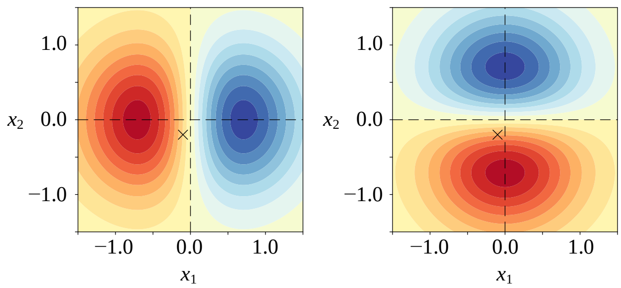


图 17. 一阶偏导数曲面等高线

(22) 中二元函数 $f(x_1, x_2)$ 二阶偏导：

$$
\begin{aligned}
f_{x1x1}\left(x_1,x_2\right) &= \frac{\partial^2 f}{\partial x_1^2}\left(x_1,x_2\right) = \left(-2+4x_1^2\right)\exp\left(-\left(x_1^2+x_2^2\right)\right) \\
f_{x2x2}\left(x_1,x_2\right) &= \frac{\partial^2 f}{\partial x_2^2}\left(x_1,x_2\right) = \left(-2+4x_2^2\right)\exp\left(-\left(x_1^2+x_2^2\right)\right) \\
f_{x1x2}\left(x_1,x_2\right) &= \frac{\partial^2 f}{\partial x_2 \partial x_1}\left(x_1,x_2\right) = 4x_1x_2\exp\left(-\left(x_1^2+x_2^2\right)\right) \\
f_{x2x1}\left(x_1,x_2\right) &= \frac{\partial^2 f}{\partial x_1 \partial x_2}\left(x_1,x_2\right) = 4x_1x_2\exp\left(-\left(x_1^2+x_2^2\right)\right)
\end{aligned}
\tag{24}
$$

显然，两个混合偏导相同，即，

$$
f_{x1x2}\left(x_1,x_2\right) = f_{x2x1}\left(x_1,x_2\right)
\tag{25}
$$

图 18 所示为两个二阶偏导函数的平面等高线图。

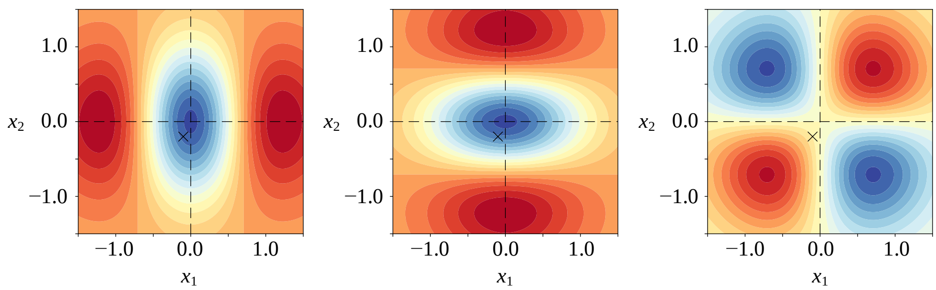


图 18. 二阶偏导数曲面等高线

展开点 × (−0.1, −0.2) 处函数值、一阶、二阶偏导数具体值为：

$$
f\left(-0.1,-0.2\right)=0.951, \quad \begin{cases} f_{x1}\left(-0.1,-0.2\right)=0.190 \\ f_{x2}\left(-0.1,-0.2\right)=0.380 \end{cases}, \quad \begin{cases} f_{x1x1}\left(-0.1,-0.2\right)=-1.864 \\ f_{x2x2}\left(-0.1,-0.2\right)=-1.750 \\ f_{x1x2}\left(-0.1,-0.2\right)= f_{x2x1}\left(-0.1,-0.2\right)=0.076 \end{cases}
\tag{26}
$$

常数函数

类似前文，我们本节也采用逐步分析。首先用二元常函数数来估计 $f(x_1, x_2)$：

$$f\left(x_1, x_2\right) \approx \underbrace{f\left(a, b\right)}_{\text{Constant}} \tag{27}$$

这相当于用一个平行于 x_1x_2 平面的水平面来近似 $f(x_1, x_2)$。

图 19 所示为用常数函数估计二元高斯函数，常数函数对应解析式为：

$$f\left(x_1, x_2\right) \approx f\left(-0.1, -0.2\right) = 0.951 \tag{28}$$

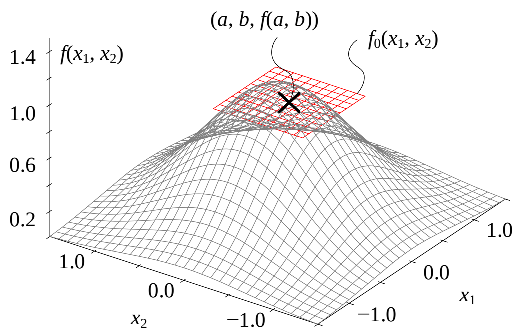


图 19. 用常数函数估计二元高斯函数

一次函数

用一次泰勒展开估计 $f(x_1, x_2)$：

$$f\left(x_1, x_2\right) \approx \underbrace{f\left(a, b\right)}_{\text{Constant}} + \underbrace{f_{x1}\left(a, b\right)\left(x_1 - a\right) + f_{x2}\left(a, b\right)\left(x_2 - b\right)}_{\text{Plane}} \tag{29}$$

相当于“水平面 + 斜面”叠加来近似 $f(x_1, x_2)$。

图 20 所示为一阶泰勒展开估计原函数，二元一次函数对应解析式为：

$$\begin{aligned} f\left(x_1, x_2\right) &\approx f\left(-0.1, -0.2\right) + f_{x1}\left(-0.1, -0.2\right)\left(x_1 - \left(-0.1\right)\right) + f_{x2}\left(-0.1, -0.2\right)\left(x_2 - \left(-0.2\right)\right) \\ &= 0.951 + 0.190\left(x_1 + 0.1\right) + 0.380\left(x_2 + 0.2\right) \end{aligned} \tag{30}$$

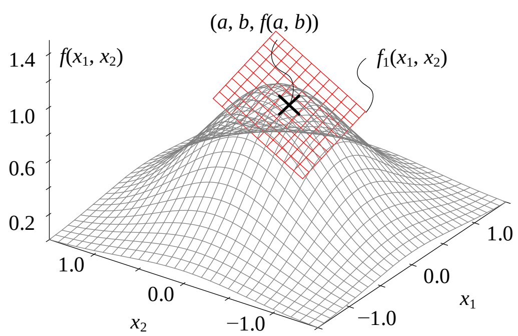


图 20. 用二元一次函数估计二元高斯函数

二次函数

用二次泰勒展开估计 $f(x_1, x_2)$：

$$\begin{aligned} f(x_1,x_2) \approx & \underbrace{f(a,b)}_{\text{Constant}} + \underbrace{f_{x1}(a,b)(x_1-a)+f_{x2}(a,b)(x_2-b)}_{\text{Plane}} \\ & + \underbrace{\frac{1}{2!}\left[f_{x1x1}(a,b)(x_1-a)^2+2f_{x1x2}(a,b)(x_1-a)(x_2-b)+f_{x2x2}(a,b)(x_2-b)^2\right]}_{\text{Quadratic}} \end{aligned} \tag{31}$$

相当于“水平面 + 斜面 + 二次曲面”叠加来近似 $f(x_1, x_2)$。

图 21 所示为二阶泰勒展开估计原函数，二元二次函数对应解析式为：

$$\begin{aligned} f_2(x_1,x_2) = & 0.951+0.190(x_1+0.1)+0.380(x_2+0.2) \\ & +\frac{1}{2}\left[-1.864(x_1+0.1)^2+0.152(x_1+0.1)(x_2+0.2)-1.750(x_2+0.2)^2\right] \end{aligned} \tag{32}$$

请大家用本节代码自行展开整理上式。

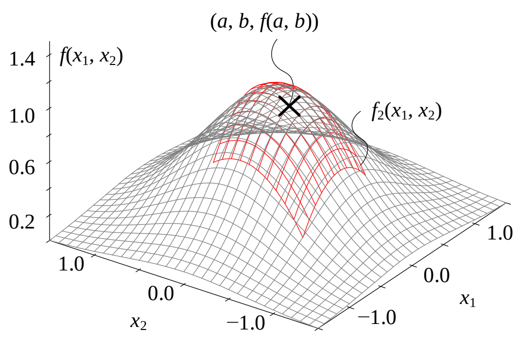


图 21. 用二次函数估计二元高斯函数

Bk3_Ch17_03.py 绘制图 19、图 20、图 21 三幅图。建议大家自己用 Streamlit 把这个代码改成一个 App。

17.5 数值微分：估算一阶导数

并不是所有函数都能得到导数的解析解，很多函数需要用数值方法近似求得导数。数值方法就是“近似”。

三种方法

本节介绍三种一次导数的数值估算方法：向前差分 (forward difference)、向后差分 (backward difference)、中心差分 (central difference)，具体如图 22 所示。

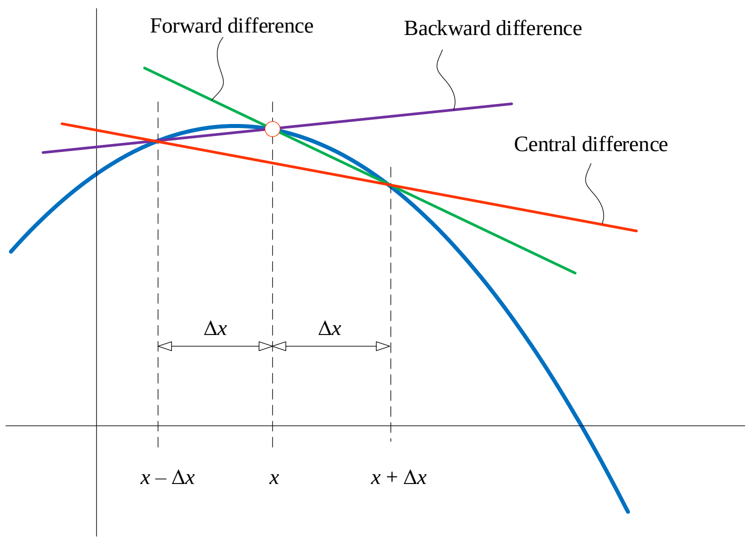


图 22. 三种一次导数的数值估计方法

一阶导数向前差分的具体公式为：

$$f'(x) \approx \frac{f(x+\Delta x)-f(x)}{\Delta x} \tag{33}$$

一阶导数向后差分的公式如下：

$$f'(x) \approx \frac{f(x)-f(x-\Delta x)}{\Delta x} \tag{34}$$

一阶导数的中心差分形式如下：

$$f'(x) \approx \frac{f(x+\Delta x)-f(x-\Delta x)}{2\Delta x} \tag{35}$$

举个例子

给定如下高斯函数：

$$f(x) = \exp\left(-x^2\right) \tag{36}$$

我们可以很容易计算得到它的一阶导数函数解析式为：

$$f(x) = -2x\exp\left(-x^2\right) \tag{37}$$

图 23 所示为高斯函数和它的一阶导数图像。同时，我们用三种不同的数值方法在 x 取不同值时估算 (37)。

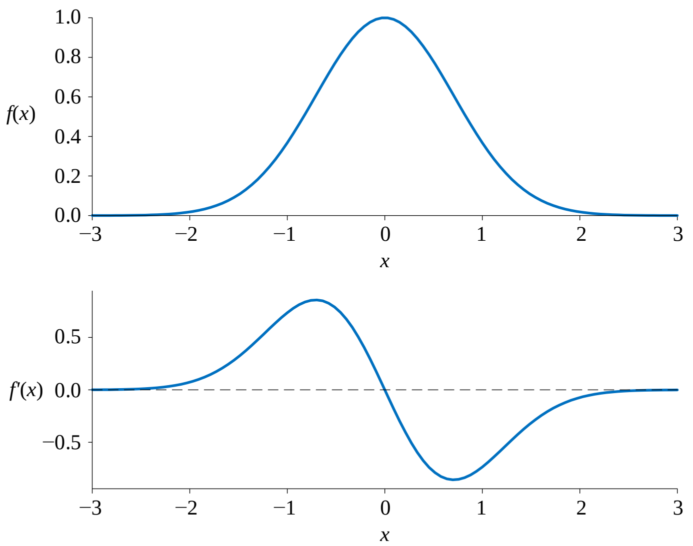


图 23. 高斯函数和它的一阶导数图像

设定 $\Delta x = 0.2$，图 24 对比中心差分、向前差分、向后差分结果。图 24 中，中心差分的结果相对好一些。

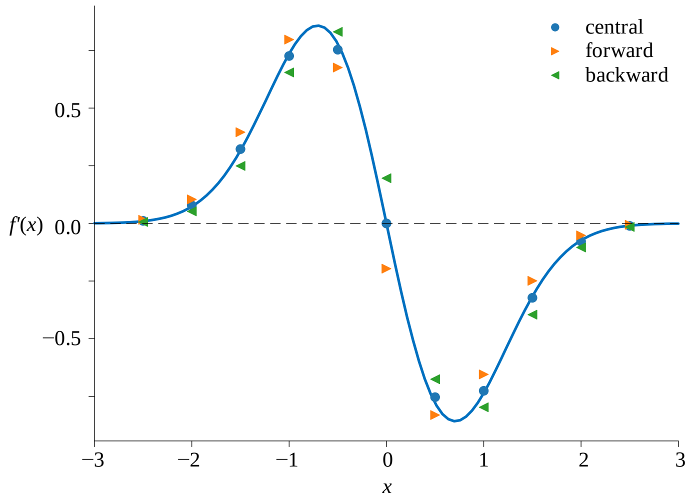


图 24. 对比中心差分、向前差分、向后差分结果，$\Delta x = 0.2$

图 25 所示为 $x = 1$，步长 Δx 取不同值时，中心差分、向前差分、向后差分结果对比。很明显当 Δx 减小时，中心差分更快地收敛于解析解，具有更高的精度。

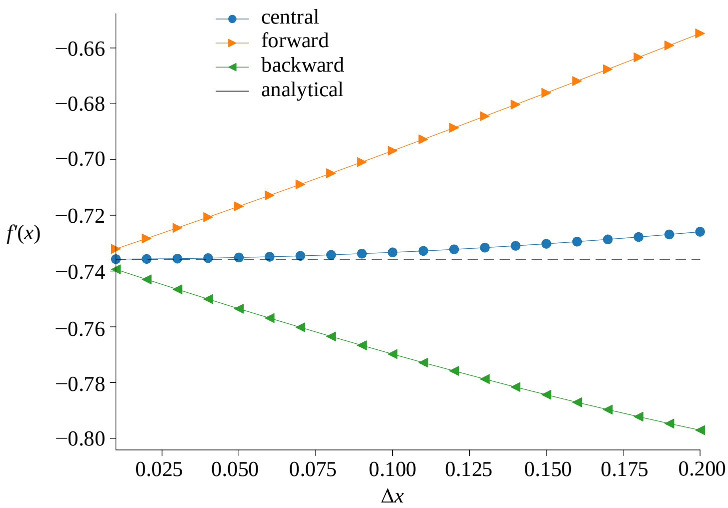


图 25. 步长 Δx 不同时，中心差分、向前差分、向后差分结果

Bk3_Ch17_04.py 完成三种差分运算，并绘制图 24 和图 25。

本章有三个关键点——微分、泰勒展开、数值微分。

再次强调，导数是函数的变化率，而微分是函数线性近似。从几何视角来看，微分用切线近似非线性函数。

泰勒展开是一系列多项式函数叠加，用来近似某个复杂函数；最为常用的是一阶泰勒展开和二阶泰勒展开。

并不是所有的函数都能很容易求得导数，对于求导困难的函数，我们可以采用数值微分的方法。本章介绍了三种不同的方法。