

## 8

## Algebraic Functions

## 代数函数

自变量有限次加、减、乘、除、有理指数幂和开方



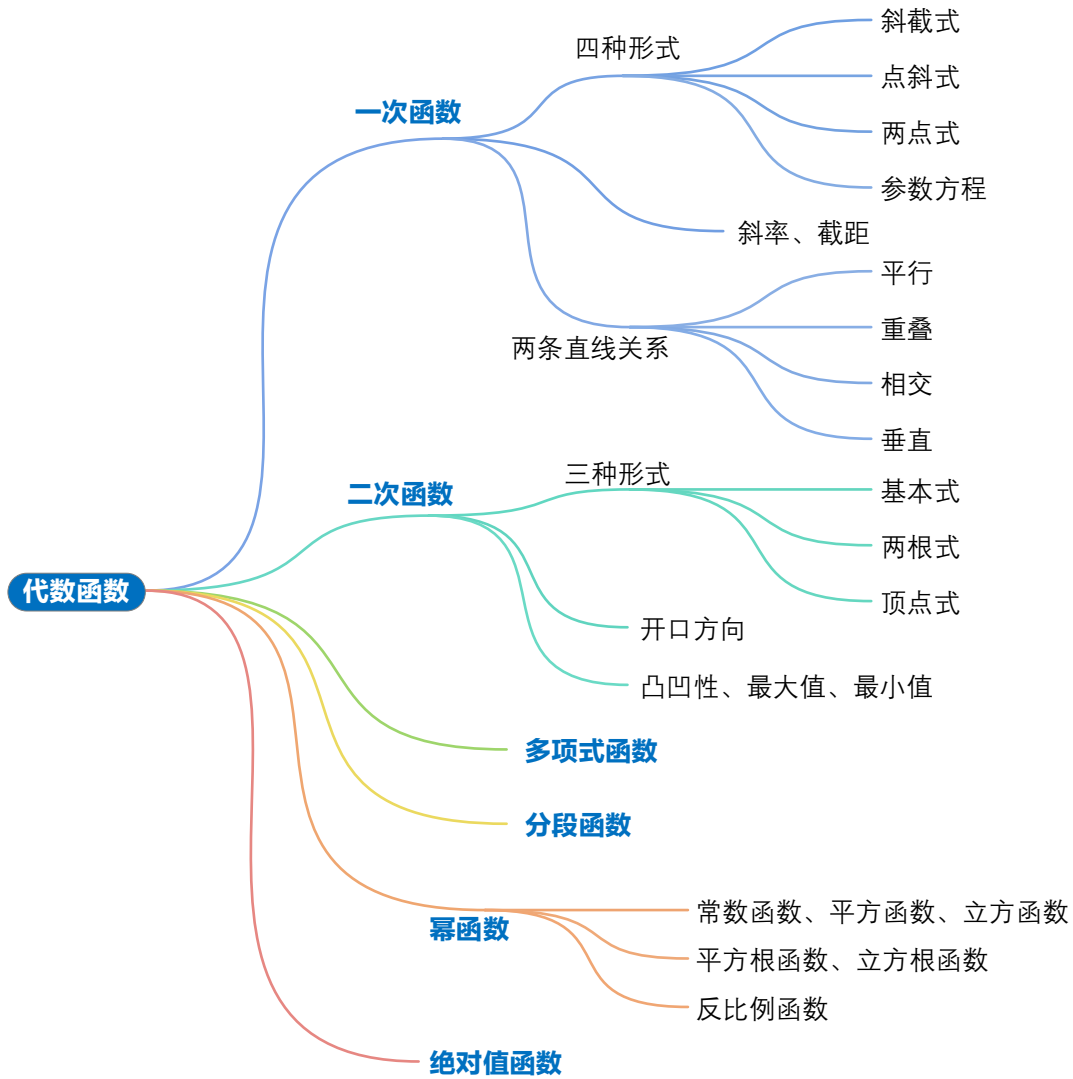
数学不分种族、不分地域；对于数学来说，其文化世界自成一国。

***Mathematics knows no races or geographical boundaries; for mathematics, the cultural world is one country.***

—— 大卫·希尔伯特 (David Hilbert) | 德国数学家 | 1862 ~ 1943



- ◀ matplotlib.pyplot.axhline() 绘制水平线
- ◀ matplotlib.pyplot.axvline() 绘制竖直线
- ◀ matplotlib.pyplot.contour() 绘制平面等高线
- ◀ matplotlib.pyplot.contourf() 绘制平面填充等高线
- ◀ matplotlib.pyplot.grid() 绘制网格
- ◀ matplotlib.pyplot.plot() 绘制线图
- ◀ matplotlib.pyplot.show() 显示图片
- ◀ matplotlib.pyplot.xlabel() 设定 x 轴标题
- ◀ matplotlib.pyplot.ylabel() 设定 y 轴标题
- ◀ numpy.absolute() 计算绝对值
- ◀ numpy.array() 创建 array 数据类型
- ◀ numpy.cbrt(x) 计算立方根
- ◀ numpy.ceil() 计算向上取整
- ◀ numpy.floor() 计算向下取整
- ◀ numpy.linspace() 产生连续均匀向量数值
- ◀ numpy.meshgrid() 创建网格化数据
- ◀ numpy.sqrt() 计算平方根



## 8.1 初等函数：数学模型的基础

初等函数是最朴实无华的数学模型，它们是复杂数学模型的基础。

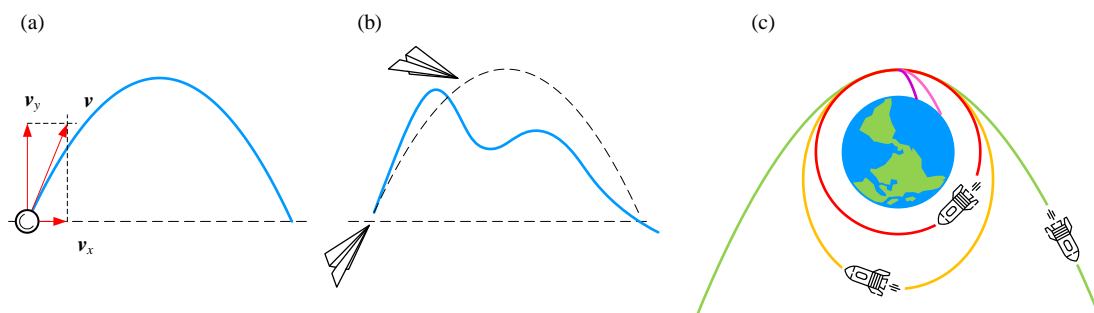


图 1. 抛物线

如图 1 (a) 所示，斜向上方抛起一个小球，忽略空气阻力影响，小球在空中划出一道曲线就可以近似地用抛物线描述。

同样的仰角，斜向上抛出一个纸飞机，纸飞机在空中的飞行轨迹就不得不考虑纸飞机外形、空气气流这些因素；如图 1 (b) 所示，抛物线已经不足以描述纸飞机的轨迹。

类似的，很多应用场景都需要对抛物线模型进行修正。比如，发射炮弹时，阻力与炮弹外形和飞行速度有密切关系；击打网球时，网球旋转可以改变轨迹；射击时，枪管膛线让子弹旋转飞行，这必然会影响到其行进的轨迹。

准确地说，抛射体受到的重力实际上是指向地心；也就是说物体在空中飞行时，是朝着地球中心的加速度，它的轨迹实际上是椭圆的一部分。

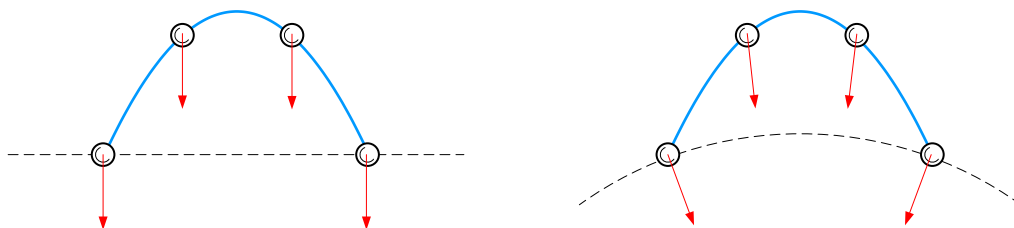


图 2. 引力方向

远程炮弹飞行需要考虑地心引力变化、地球自转；深空探测时，飞行器轨迹还需要考虑不同星体之间的引力作用，甚至来自太阳的光压等等因素。

认为抛物物体轨迹为抛物线至少基于几个假设前提。比如，忽略空气阻力的影响；再比如，假设大地平坦；同时假设物体受到的地球引力垂直大地。

假设前提是每个数学模型应用基础；数学模型毕竟是对现实世界各种现象的高度抽象概括，必须忽略一些次要因素，才能把握主要矛盾。

算力有限时，抛射一个实心小球，显然不会考虑小球的气动因素，更不会考虑引力场因素。但是，模拟不同击打技巧对网球飞行轨迹的影响，就不得不考虑网球旋转和空气流动这些因素。

模拟洲际导弹弹道时，气动布局、空气流体、地球自转、引力场等因素就不再是次要因素，必须考虑这些因素才能准确判断炮弹落点。

人类对抛物线、空气动力学，很大程度上来自于对弹道的研究。不得不承认，科学技术的确是把双刃剑，战争有些时候是人类自然科学知识进步的加速器。

## 8.2 一次函数：一条斜线

### 四种形式

**一次函数** (linear function) 有几种不同的形式构造。

- ◀ **斜截式** (slope-intercept form)，如图 3 (a) 所示；
- ◀ **点斜式** (point-slope form)，如图 3 (b) 所示；
- ◀ **两点式** (two-point form)，如图 3 (c) 所示；
- ◀ **参数方程** (parametric equation)，如图 3 (d) 所示。

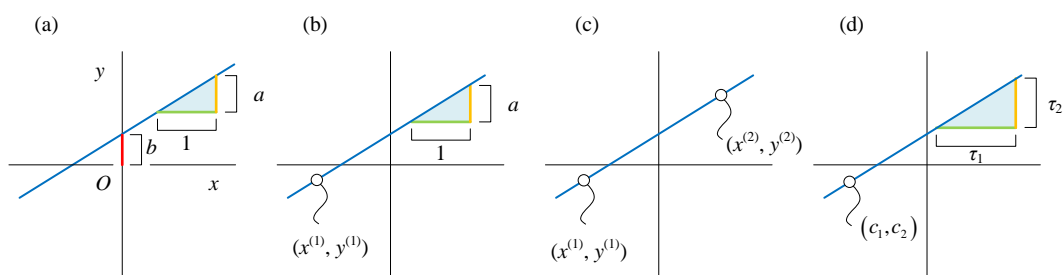


图 3. 一次函数的几种构造方法

### 斜截式

斜截式一次函数形式如下。

$$y = f(x) = ax + b \quad (1)$$

斜截式需要两个参数——斜率 ( $a$ ) 和  $y$  轴截距 ( $b$ )。

注意，斜率 ( $a$ ) 不能为 0；当  $a = 0$  时，(1) 为**常数函数** (constant function)。

对于 (1)，当  $b = 0$  时，函数为**比例函数** (proportional function)，而  $a$  也叫做**比例常数** (constant ratio 或 proportionality constant)。

一次函数有两种斜率：**正斜率** (positive slope) 和**负斜率** (negative slope)。图 4 (a) 所示一次函数斜率大于 0，单调递增；图 4 (b) 所示斜率小于 0，一次函数单调递减。

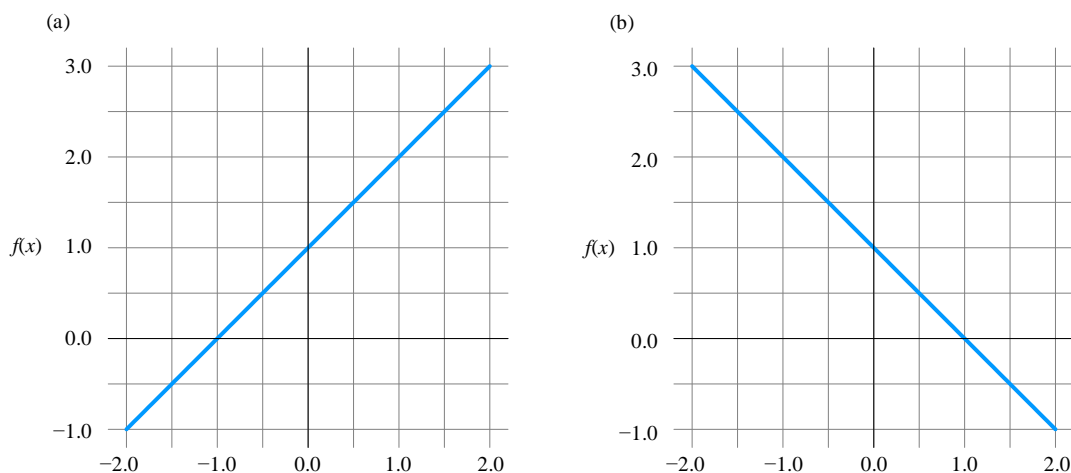


图 4. 一次函数

图 5 所示为一次函数随斜率变化。**零斜率** (zero slope) 对应常数函数，即**水平线** (horizontal line)。**无定义斜率** (undefined slope) 代表一条**竖直线** (vertical line)，此时图像虽然是一条直线，垂直于横轴，但它并不是函数。

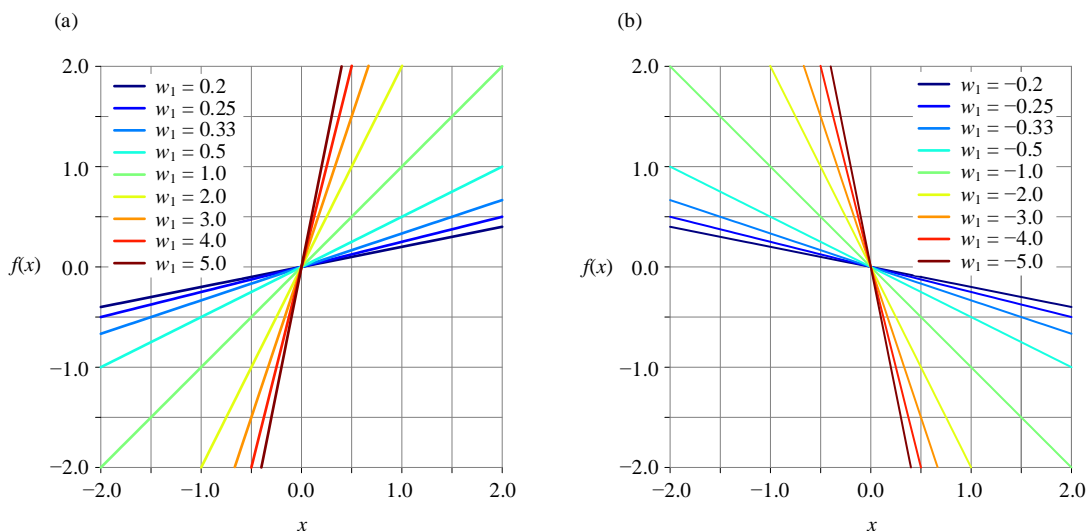


图 5. 一次函数  $y = w_1 x$  随斜率变化

图 6 所示为一次函数随截距变化情况；调整截距大小，相当于图像上下平移。

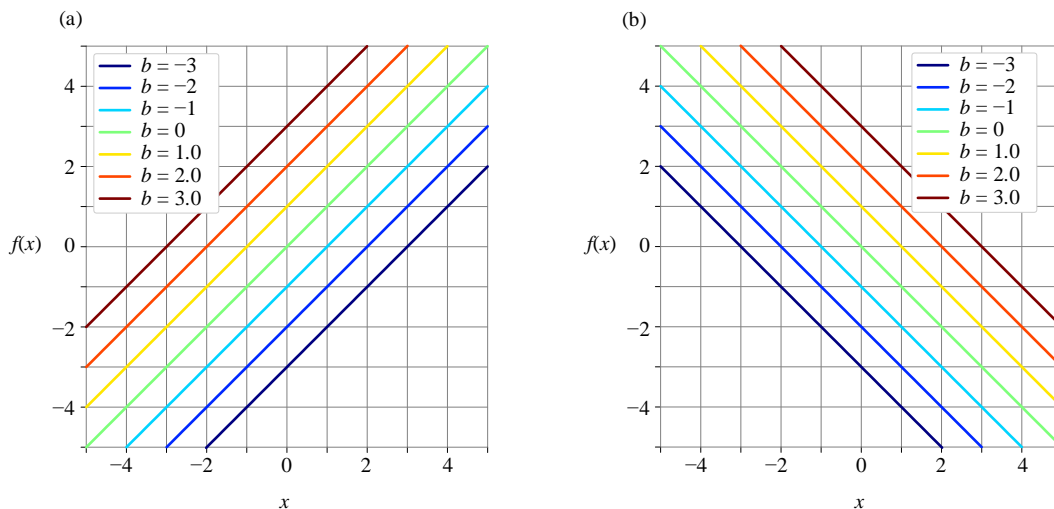


图 6. 一次函数随截距变化

如果两条直线斜率相同，它们相互**平行** (parallelize)，如图 7 (a)，或**重合** (coincide)，如图 7 (b)。图 7 (c) 所示为两条直线**相交** (intersect)，有唯一交点。

如果两个一次函数的斜率乘积为-1，则两者**垂直** (perpendicular)，如图 7 (d)。

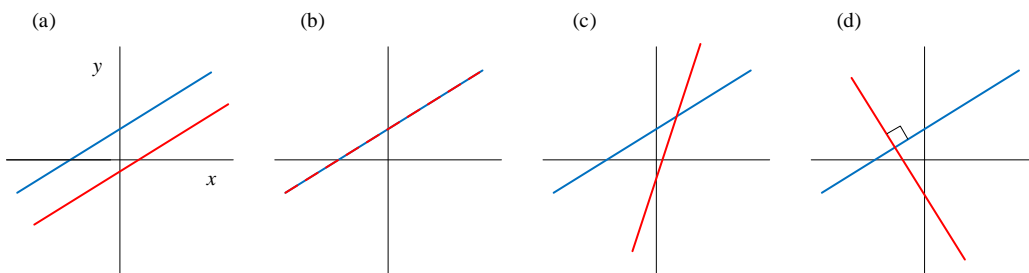


图 7. 两条之间的关系

## 点斜式

一次函数的第二种是点斜式。

$$y - y^{(1)} = f(x) - y^{(1)} = a(x - x^{(1)}) \quad (2)$$

也就是说，给定斜率  $a$  和直线上的一点  $(x^{(1)}, y^{(1)})$ ，便可以确定平面上一条直线。

## 两点式

第三种形式是两点式，即两点确定一条直线。如下一次函数通过  $(x^{(1)}, y^{(1)})$  和  $(x^{(2)}, y^{(2)})$  两点。

$$y - y^{(1)} = \underbrace{\frac{y^{(2)} - y^{(1)}}{x^{(2)} - x^{(1)}}}_{\text{Slope}} (x - x^{(1)}) \quad (3)$$

其中， $x^{(1)} \neq x^{(2)}$ 。

两点式可以写成。

$$(y - y^{(1)})(x^{(2)} - x^{(1)}) = (y^{(2)} - y^{(1)})(x - x^{(1)}) \quad (4)$$



一次函数虽然看着简单，大家千万不要小瞧；数据科学和机器学习中很多算法都离不开一次函数，比如简单线性回归 (Simple Linear Regression)。简单线性回归也叫一元线性回归模型，是指模型中只含有一个自变量和一个因变量。

简单线性回归采用的解析式便是一元函数的斜截式。

人们常用有限的样本数据去探寻数据之间的规律，并以此作为分析或预测的工具。如图 8 所示，给定的样本数据中  $x$  和  $y$  似乎存在某种线性关系；通过一些算法，我们可以找到图 8 中那条红色斜线，它就是简单线性回归模型。

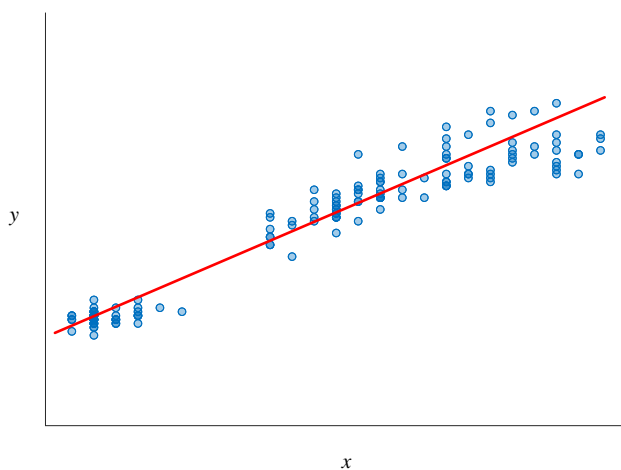


图 8. 简单线性回归

以下代码绘制图 5、图 6。本书中，一元函数自变量  $x$  取值一般是等差数列。

`numpy.linspace(start, end, num)` 可以用来生成等差数列，数列 `start` 和 `end` 之间（注意，包括 `start` 和 `end` 两个端点数值），数列的元素个数为 `num` 个；得到的结果数据类型为 `array`。



```
# Bk_Ch8_01

import numpy as np
import matplotlib.pyplot as plt

w_array = np.array([1/5, 1/4, 1/3, 1/2, 1, 2, 3, 4, 5])
x_array = np.linspace(-2, 2, 100)

ww, xx = np.meshgrid(w_array, x_array)

b = 0 # y intercept
ff = ww*xx + b

fig, ax = plt.subplots()

colors = plt.cm.jet(np.linspace(0, 1, len(w_array)))

for i in np.linspace(1, len(w_array), len(w_array)):
    plt.plot(x_array, ff[:, int(i)-1],
             color = colors[int(i)-1],
             label = '$w_1 = \{111:.2f\}$'.format(111 = w_array[int(i)-1]))

plt.xlabel('x')
plt.ylabel('f(x)')
plt.legend()
plt.xticks(np.arange(-2, 2.5, step=0.5))
plt.yticks(np.arange(-2, 2.5, step=0.5))
plt.axis('scaled')
ax.set_xlim(-2, 2)
ax.set_ylim(-2, 2)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.grid(linestyle='--', linewidth=0.25, color=[0.5, 0.5, 0.5])
```

## 8.3 二次函数：一条抛物线

**二次函数** (quadratic function) 是**二次多项式函数** (second order polynomial function 或 second degree polynomial function)。

二次函数图象是**抛物线** (parabola)，可以**开口向上** (open upward) 或**开口向下** (open downward)，**对称轴平行于纵轴** (the axis of symmetry is parallel to the y-axis)。

### 三种形式

二次函数解析式有三种形式。

◀ **基本式** (standard form)，如图 9 (a)；



◀ **两根式** (factored form), 如图 9 (b);

◀ **顶点式** (vertex form), 如图 9 (c)。

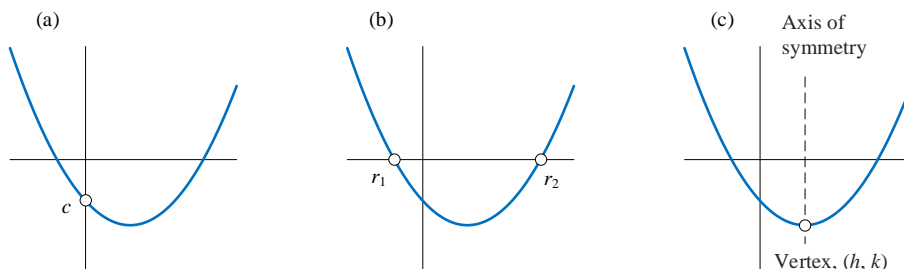


图 9. 二次函数的几种构造方法

## 基本式

二次函数的基本式如下。

$$f(x) = ax^2 + bx + c, \quad a \neq 0 \quad (5)$$

其中,  $a$  被称作**二次项系数** (quadratic coefficient), 注意  $a$  不为零;  $b$  被称作**一次项系数** (linear coefficient);  $c$  被称作**常数项** (constant term), 也是  $y$  轴截距 ( $y$ -intercept)。

图 10 (a) 所示二次函数开口向上, **顶点** (vertex) 位于  $y$  轴, 对称轴为  $y$  轴。图 10 (b) 所示二次函数开口向下。

图 10 (a) 二次函数为凸, 顶点位置对应函数**最小值** (minimum)。图 10 (b) 二次函数为凹, 顶点位置对应函数**最大值** (maximum)。

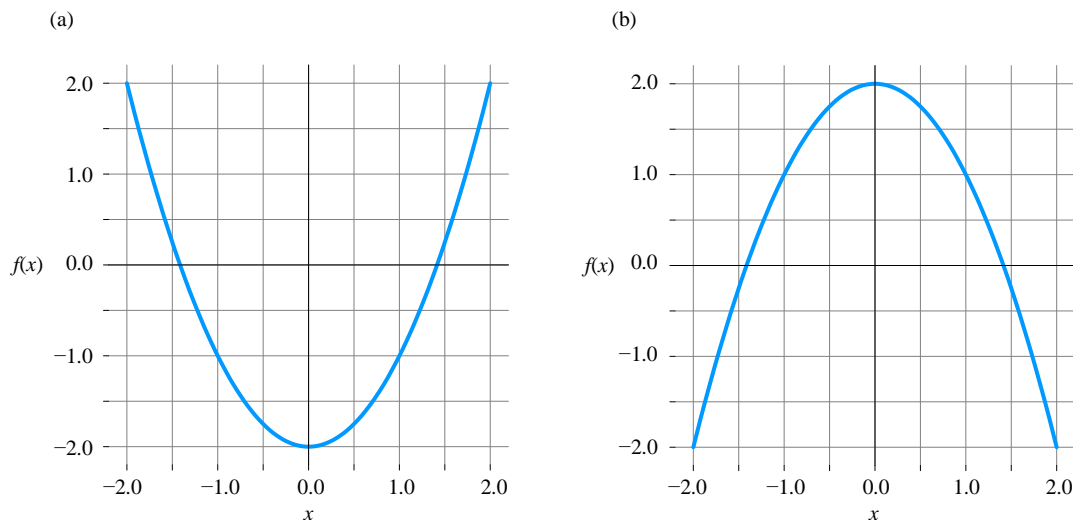
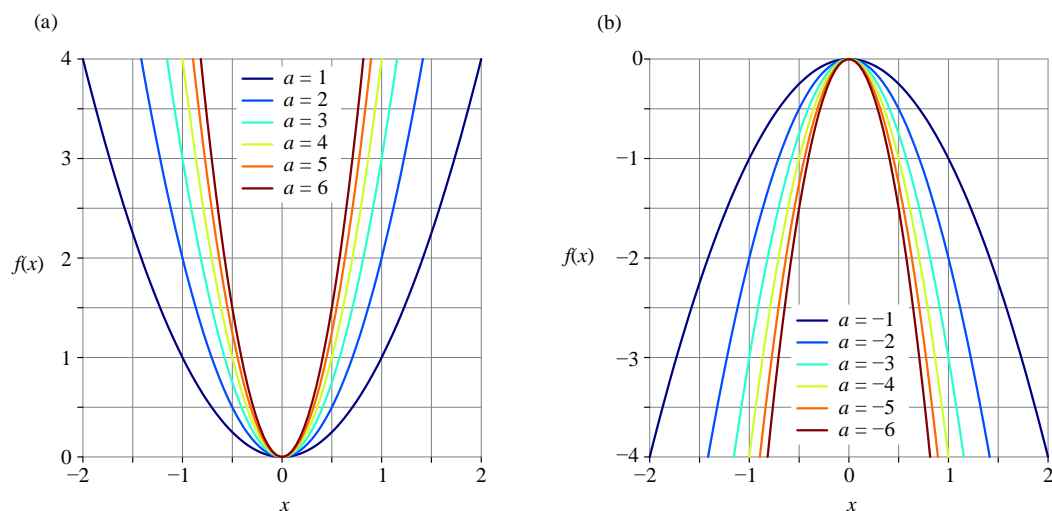


图 10. 不同开口方向二次函数

图 11 所示为二次函数图像开口大小随系数  $a$  变化;  $a$  的绝对值越大, 开口越小。

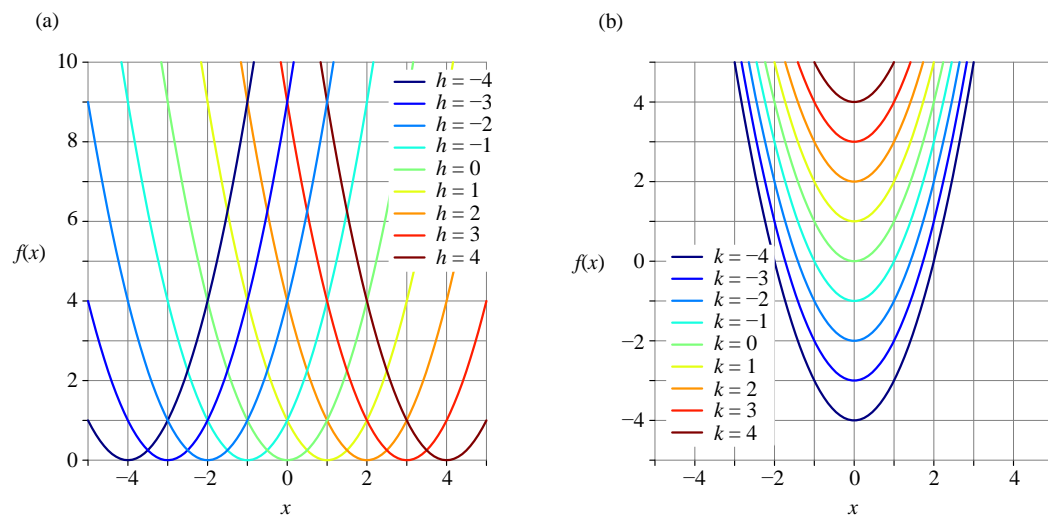
图 11. 二次函数随  $a$  变化

## 两根式

如果  $f(x) = 0$  存在两个实数根的话, 二次函数可以写成两根式。

$$f(x) = a(x - r_1)(x - r_2), \quad a \neq 0 \quad (6)$$

其中,  $r_1$  和  $r_2$  为二次方程的根。

图 12. 二次函数随  $h$  和  $c$  变化

## 顶点式

二次函数另外一个常见的形式是顶点式，具体形式如下。

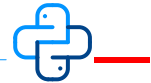
$$f(x) = a(x-h)^2 + k, \quad a \neq 0 \quad (7)$$

其中， $h$  和  $k$  分别为顶点的横纵坐标值。

图 12 (a) 所示为函数图像和  $h$  的关系， $h$  影响函数在水平方向位置；图 12 (b) 所示为函数图像和  $k$  的关系， $k$  影响函数在竖直方向位置。

前文提到，二次函数顶点可以是函数的最大值或最小值；该顶点也是函数**单调性** (monotonicity) 的拐点 (turning point)，二次函数**单调区间** (intervals of monotonicity) 分别为  $(-\infty, h)$  和  $(h, +\infty)$ 。

以下代码绘制图 11 和图 12。



```
# Bk Ch8 02

import numpy as np
import matplotlib.pyplot as plt

a_array = np.linspace(1,6,6)
x_array = np.linspace(-2,2,100)

aa, xx = np.meshgrid(a_array,x_array)
ww = aa*xx**2

fig, ax = plt.subplots()

colors = plt.cm.jet(np.linspace(0,1,6))

for i in np.linspace(1,6,6):
    plt.plot(x_array,ww[:,int(i)-1],
             color = colors[int(i)-1],
             label = '$a = {l1l:.0f}$'.format(l1l = a_array[int(i)-1]))

plt.xlabel('x'); plt.ylabel('f(x)')
plt.legend()
plt.xticks(np.arange(-2, 2.5, step=0.5)); plt.yticks(np.arange(0, 4.5, step=0.5))
plt.axis('scaled')
ax.set_xlim(-2,2); ax.set_ylim(0,4)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.grid(linestyle='--', linewidth=0.25, color=[0.5,0.5,0.5])
```

## 8.4 多项式函数

**多项式函数** (polynomial function) 相当于一次和二次函数的推广，具体形式如下。

$$y = f(x) = a_K x^K + a_{K-1} x^{K-1} + \dots + a_2 x^2 + a_1 x + a_0 = \sum_{i=0}^K a_i x^i \quad (8)$$

其中，最高次项系数  $a_K$  不为 0， $K$  为最高次项次数。

图 13 几幅分图分别展示常数函数、一次到五次函数图像；可以这样理解，任何五次多项式函数都是图 13 所示的图像分别乘以相应系数叠加而成。

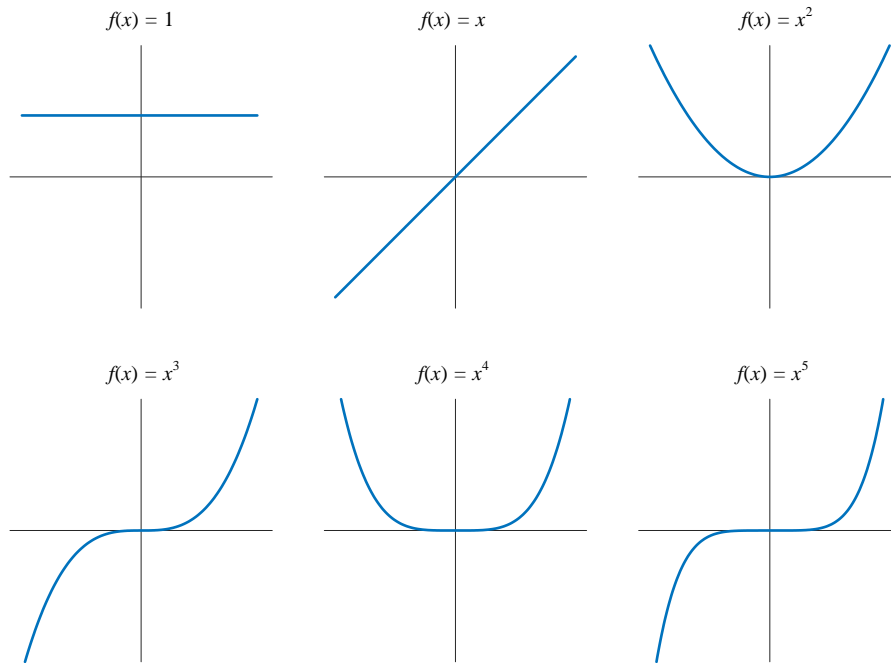


图 13. 常数函数到五次函数

**三次函数** (cubic function, polynomial function of degree 3) 的形式为。

$$y = f(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0 \quad (9)$$

举两个三次函数的例子。

$$\begin{aligned} y &= f(x) = x^3 - x \\ y &= f(x) = -x^3 + x \end{aligned} \quad (10)$$

这两个三次函数可以看做是  $x^3$  和  $x$  经过加减运算组合而成。

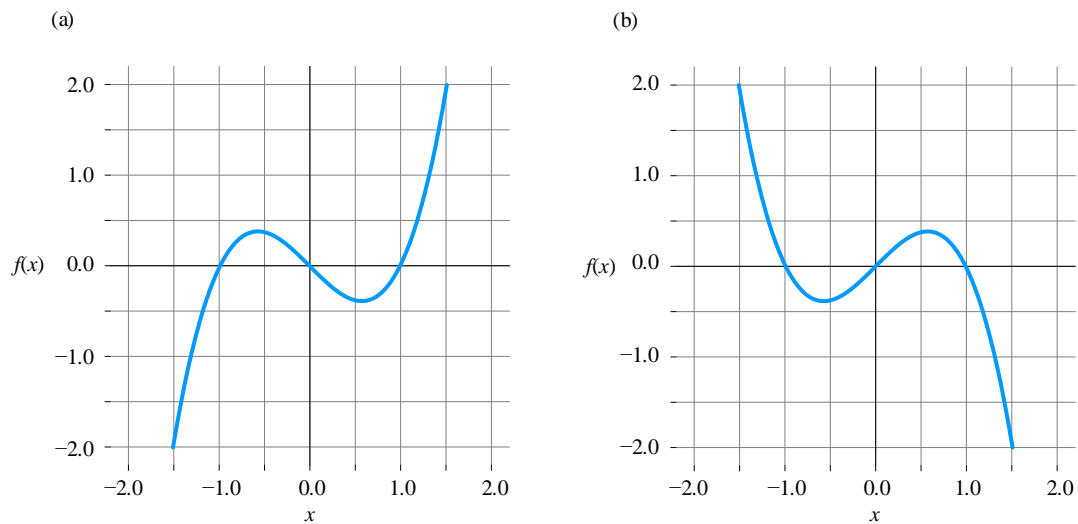


图 14. 两个三次函数

以下代码绘制图 4、图 10 和图 14。



```
# Bk_Ch8_03

import math
import numpy as np
from matplotlib import pyplot as plt

x = np.linspace(-2, 2, 100);

def plot_curve(x, y):

    fig, ax = plt.subplots()

    plt.xlabel("$\it{x}$")
    plt.ylabel("$\it{f}(\it{x})$")
    plt.plot(x, y, linewidth = 1.5)
    plt.axhline(y=0, color='k', linewidth = 1.5)
    plt.axvline(x=0, color='k', linewidth = 1.5)
    ax.grid(linestyle='--', linewidth=0.25, color=[0.5,0.5,0.5])
    plt.axis('equal')
    plt.xticks(np.arange(-2, 2.5, step=0.5))
    plt.yticks(np.arange(y.min(), y.max() + 0.5, step=0.5))
    ax.set_xlim(x.min(), x.max())
    ax.set_ylim(y.min(), y.max())
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.spines['bottom'].set_visible(False)
    ax.spines['left'].set_visible(False)
    plt.axis('square')

%% plot linear, quadratic, and cubic functions

plt.close('all')

# linear function
y = x + 1;
```

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

```

plot_curve(x, y)

# linear function
y = -x + 1;
plot_curve(x, y)

# quadratic function, parabola opens upwards
# y = np.power(x,2) - 2;
y = x**2 - 2;
plot_curve(x, y)

# quadratic function, parabola opens downwards
# y = -np.power(x,2) + 2;
y = -x**2 + 2;
plot_curve(x, y)

# cubic function
# y = np.power(x,3) - x;
y = x**3 - x;
plot_curve(x, y)

# cubic function
# y = -np.power(x,3) + x;
y = -x**3 + x;
plot_curve(x, y)

```

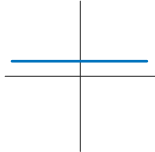
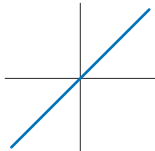
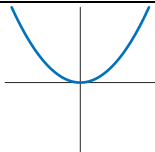
## 8.5 幂函数

**幂函数** (power function) 是形如下式的函数。

$$f(x) = k \cdot x^p \quad (11)$$

表 2 总结常用的幂函数。

表 1. 几个常用幂函数

幂函数	例子	图像
常数函数 (constant function)	$f(x) = 1 = x^0$	
恒等函数 (identity function)	$f(x) = x = x^1$	
平方函数 (square function)	$f(x) = x^2$	

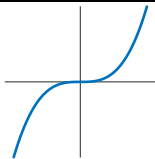
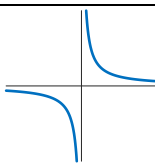
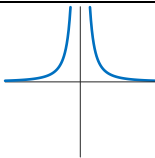
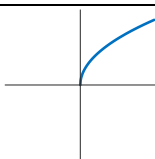
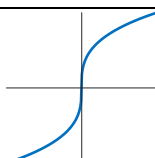
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

立方函数 (cubic function)	$f(x) = x^3$	
反比例函数 (reciprocal function)	$f(x) = \frac{1}{x} = x^{-1}$	
反比例平方函数 (reciprocal squared function)	$f(x) = \frac{1}{x^2} = x^{-2}$	
平方根函数 (square root function)	$f(x) = \sqrt{x} = x^{\frac{1}{2}}$	
立方根函数 (cubic root function)	$f(x) = \sqrt[3]{x} = x^{\frac{1}{3}}$	

## 平方根函数

图 15 (a) 所示红色曲线为**平方根函数** (square root function), 对应函数式为。

$$y = f(x) = \sqrt{x} = x^{\frac{1}{2}} \quad (12)$$

注意上式函数定义域  $x \geq 0$ , 即非负实数; 函数值域也是非负实数。`numpy.sqrt(x)` 可以用来计算平方根, 也可以用 `x**(1/2)` 来计算。

## 立方根函数

图 15 (b) 所示红色曲线为**立方根函数** (cubic root function), 对应函数式为。

$$y = f(x) = \sqrt[3]{x} = x^{\frac{1}{3}} \quad (13)$$

`numpy.cbrt(x)` 可以用来计算立方根。注意, `x**(1/3)` 不可以计算负数立立方根。

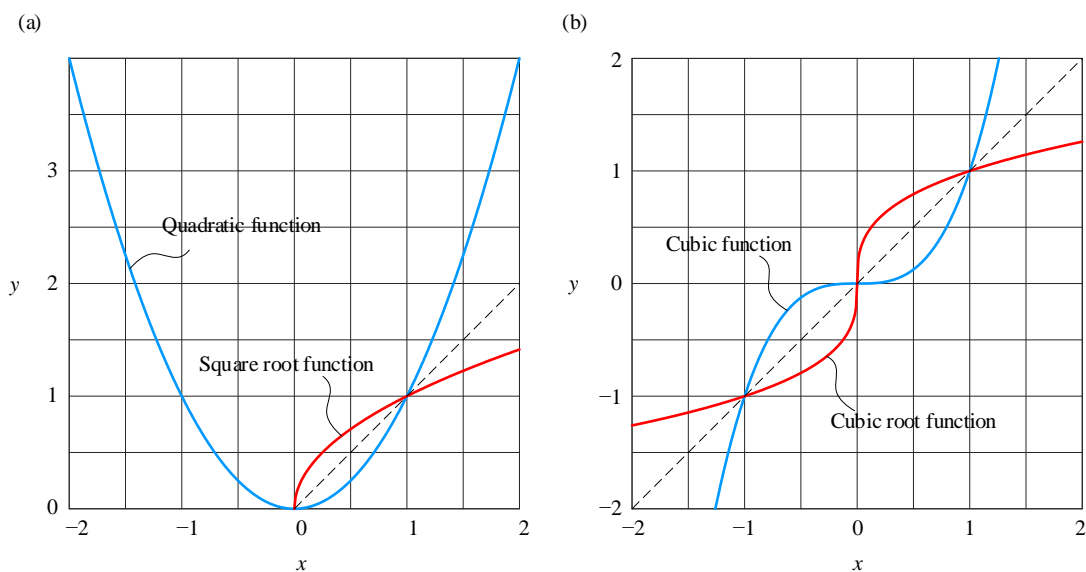


图 15. 平方根和立方根函数

### 奇偶性

如图 16 (a) 所示, 当  $p$  为偶数时, 幂函数为偶函数, 图像关于  $y$  轴对称;  $p$  值越大,  $x$  绝对值增大时, 函数值越快速接近正无穷。

如图 16 (b) 所示, 当  $p$  为奇数时, 幂函数为奇函数, 图像关于原点对称;  $p$  值越大,  $x$  绝对值增大时, 函数值越快速接近正无穷或负无穷。

请大家修改前文代码自行绘制图 16。

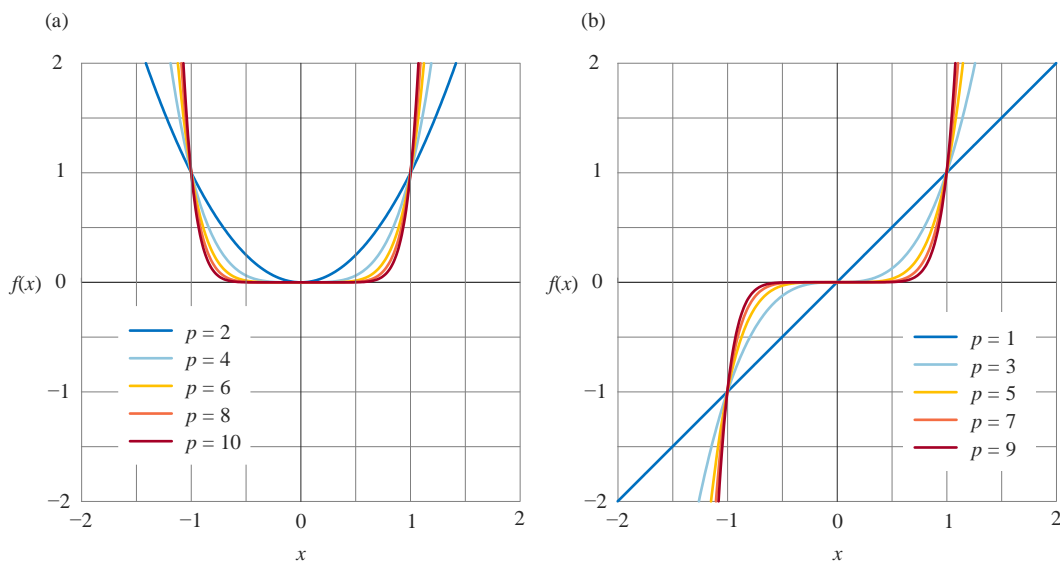
图 16. 幂函数  $p$  分别为偶数和奇数时, 图像特征

表 2. 用英文读乘幂

英文表达	中文表达
------	------

本 PDF 文件为作者草稿, 发布目的为方便读者在移动终端学习, 终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有, 请勿商用, 引用请注明出处。

代码及 PDF 文件下载: <https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教, 本书专属邮箱: [jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)



$x^n$	$x$ to the $n$ $x$ to the $n$ -th power $x$ raised to the $n$ -th power $x$ raised to the power of $n$ $x$ raised by the exponent of $n$
$a^2$	$a$ squared the square of $a$ $a$ raised to the second power $a$ to the second
$a^3$	$a$ cubed the cube of $a$ $a$ to the third
$2^5$	the fifth power of 2 2 raised to the fifth power 2 to the power of 5 2 to the fifth power 2 to the fifth 2 to the five
$y = 2^x$	$y$ equals 2 to the power of $x$
$\sqrt{2} = 2^{\frac{1}{2}}$	square root of two
$\sqrt[3]{2} = 2^{\frac{1}{3}}$	cube root of two
$\sqrt[2]{64} = 8$	The square root of sixty four is eight.
$\sqrt[3]{64} = 4$	The cube root of sixty four is four.
$\sqrt[6]{64} = 2$	The sixth root of sixty four is two.
$\sqrt[c]{a^b}$	$c$ -th root of $a$ raised to the $b$ power

## 反比例函数

反比例函数 (inversely proportional function) 的一般式

$$y = f(x) = \frac{k}{x} \quad (14)$$

如图 17 所示,  $|k| > 1$  将双曲线朝远离原点方向拉伸;  $|k| < 1$  将双曲线向靠近原点方向压缩。

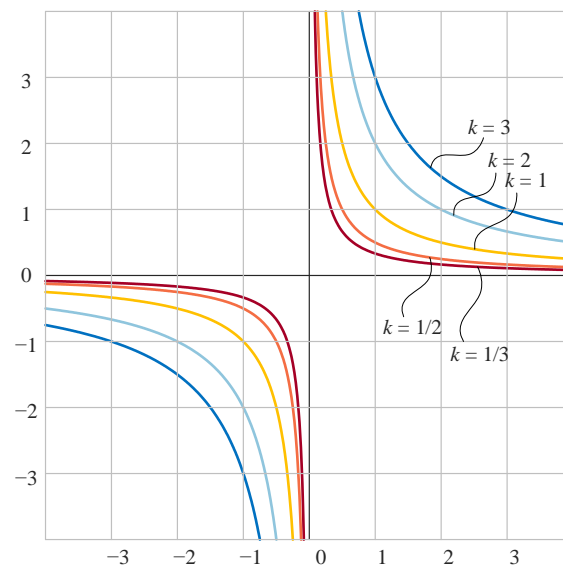


图 17.  $k$  取不同值时反比例函数图像  $f(x) = k/x$

本 PDF 文件为作者草稿, 发布目的为方便读者在移动终端学习, 终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有, 请勿商用, 引用请注明出处。

代码及 PDF 文件下载: <https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教, 本书专属邮箱: [jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

表 3. 用英文读比例函数

数学表达	英文表达
$y = \frac{k}{x}$	$x$ is inversely proportional to $y$
$h = \frac{k}{t^2}$	$h$ is inversely proportional to the square of $t$ $h$ varies inversely with the square of $t$

如图 17 所示反比例函数有两条**渐近线** (asymptote), **水平渐近线** (horizontal asymptote)  $y = 0$  和**竖直渐近线** (vertical asymptote)  $x = 0$ .

所谓渐近线是指与曲线极限相关的一条直线, 当曲线上某动点沿该曲线的一个分支移向无穷远时, 动点到该渐近线的垂直距离趋于零。图 17 中, 当  $x$  从右侧接近竖直渐近线, 函数值无约束地接近**正无穷** (positive infinity); 相反, 当  $x$  从左侧接近竖直渐近线, 函数值无约束地接近**负无穷** (negative infinity)。

反比例函数实际上是旋转双曲线。

### 有理函数

反比例函数移动之后可以得到最简单的**有理函数** (rational function), 解析式如下。

$$f(x) = \frac{k}{x-h} + a \quad (15)$$

$h$  左右移动竖直渐近线,  $a$  上下移动水平渐近线, 比如下例。

$$f(x) = \frac{1}{x-1} + 1 \quad (16)$$

如图 18 所示,  $y = 1$  为 (16) 对应反比例函数的水平渐近线;  $x = 1$  为竖直渐近线。

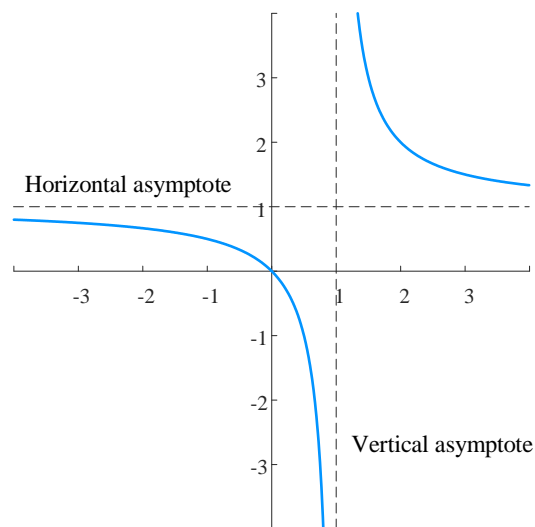


图 18. 反比例函数两条渐近线

## 8.6 分段函数：不连续函数

**分段函数** (piecewise function) 是一类不连续函数；分段函数是自变量  $x$  的不同的取值范围有不同的解析式的函数。

图 19 对应下例分段函数。

$$f(x) = \begin{cases} 4 & x < -2 \\ -1 & -2 \leq x < 3 \\ 3 & 3 \leq x \end{cases} \quad (17)$$

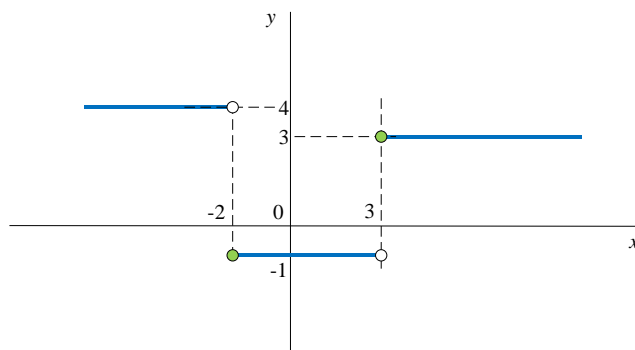


图 19. 分段函数



**插值** (interpolation) 指的是通过已知离散数据点，在一定范围内推导求得新数据点的方法。**线性插值** (linear interpolation) 是指插值函数为一次函数。

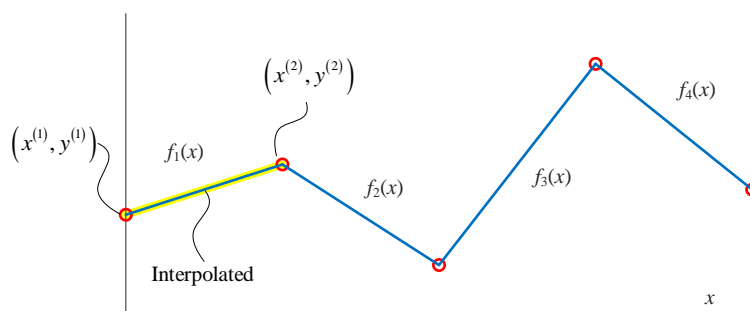


图 20. 一次函数两点式用于线性插值

插值函数是分段函数时，也称分段插值 (piecewise interpolation)，每两个相邻的数据点之间便是一个分段函数。

$$f(x) = \begin{cases} f_1(x) & x^{(1)} \leq x < x^{(2)} \\ f_2(x) & x^{(2)} \leq x < x^{(3)} \\ \dots & \dots \\ f_{n-1}(x) & x^{(n-1)} \leq x < x^{(n)} \end{cases} \quad (18)$$

如图 20 所示，所有红色的圆点为已知离散数据点。

相邻两点连接得到的线段解析式便是线性插值分段函数，两点式公式常用在线性插值。举个例子，给定的两点  $(x^{(1)}, y^{(1)})$  和  $(x^{(2)}, y^{(2)})$  可以确定分段函数  $f_1(x)$ 。

除了线性插值之外，本系列丛书还要介绍其他常见插值方法。

## 绝对值函数

**绝对值函数** (absolute value function) 就是分段函数。严格来讲，绝对值函数不属于代数函数。绝对值函数的一般式为。

$$f(x) = k|x - h| + a \quad (19)$$

举个最简单的例子。

$$f(x) = k|x| \quad (20)$$

对于  $f(x) = k|x|$  函数， $x = 0$  为  $f(x)$  的尖点，它破坏了函数的光滑。图 21 所示为  $k$  影响绝对值函数  $f(x) = k|x|$  的开口大小； $k$  的绝对值越大，绝对值函数开口越小。

`numpy.absolute()` 计算绝对值。请大家自行编写代码绘制图 21，并讨论  $k$  为不同负整数时函数图像特点。

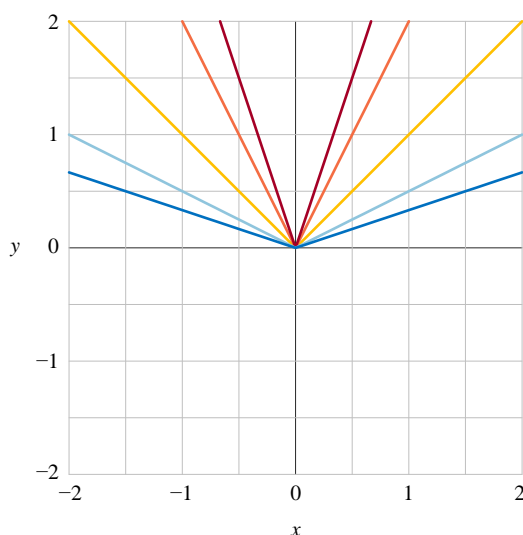
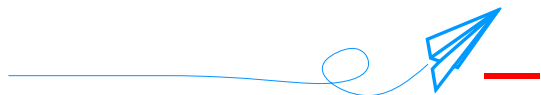


图 21.  $k$  影响绝对值函数  $f(x) = k|x|$



本章除了介绍几种常见的代数函数以外，还有一个要点——参数对函数形状、性质的影响。请大家思考这几个问题。

一次函数的斜率和截距，如何影响函数图像？

哪个参数影响二次函数的开口方向和大小？二次函数什么时候存在最大值或最小值？二次函数的对称轴位置？

用叠加这个思想，多项式函数  $f(x) = x^4 + 2x^3 - x^2 + 5$  相当于由哪些函数构造而成？它们各自的函数图像分别怎样？