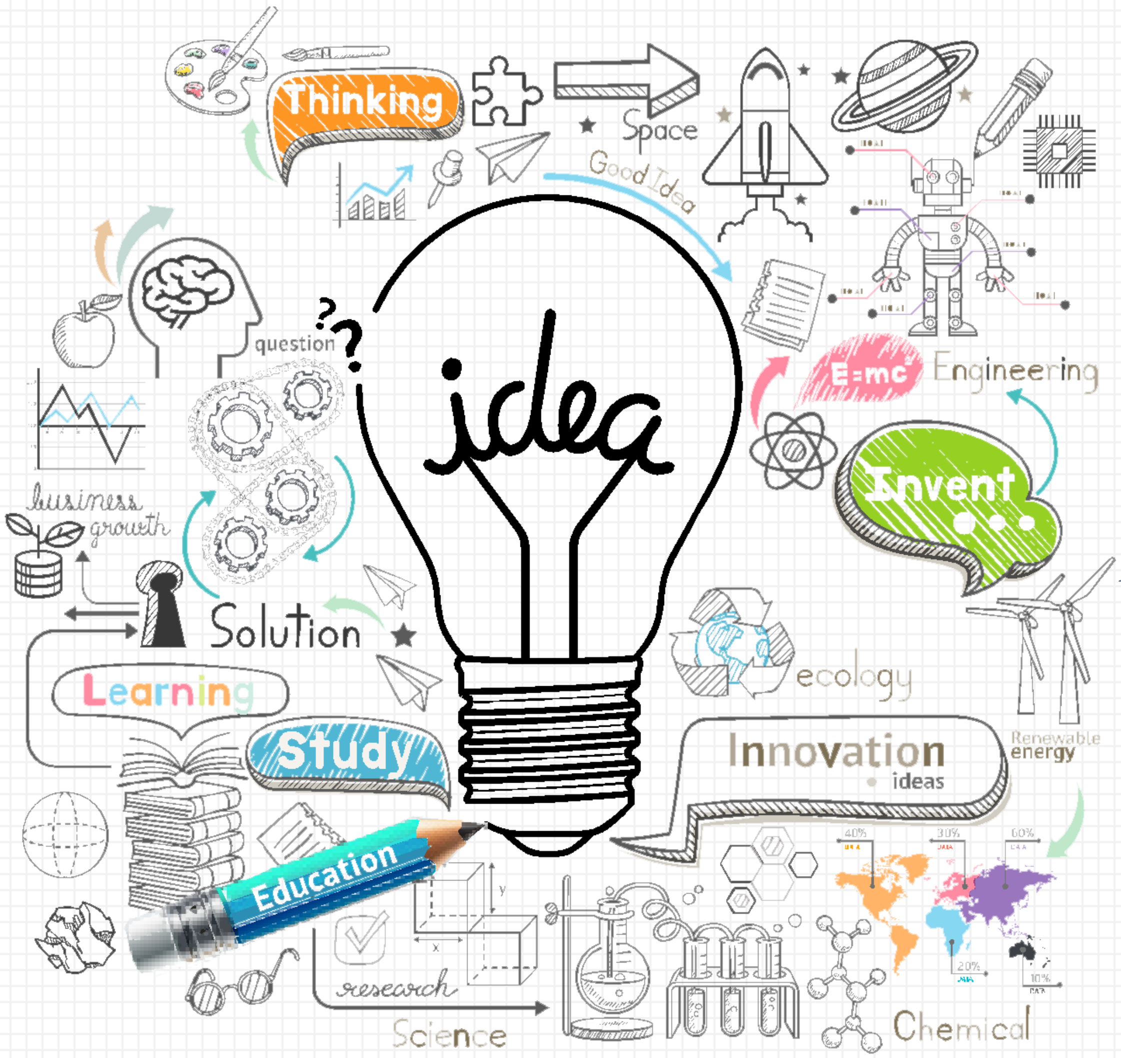
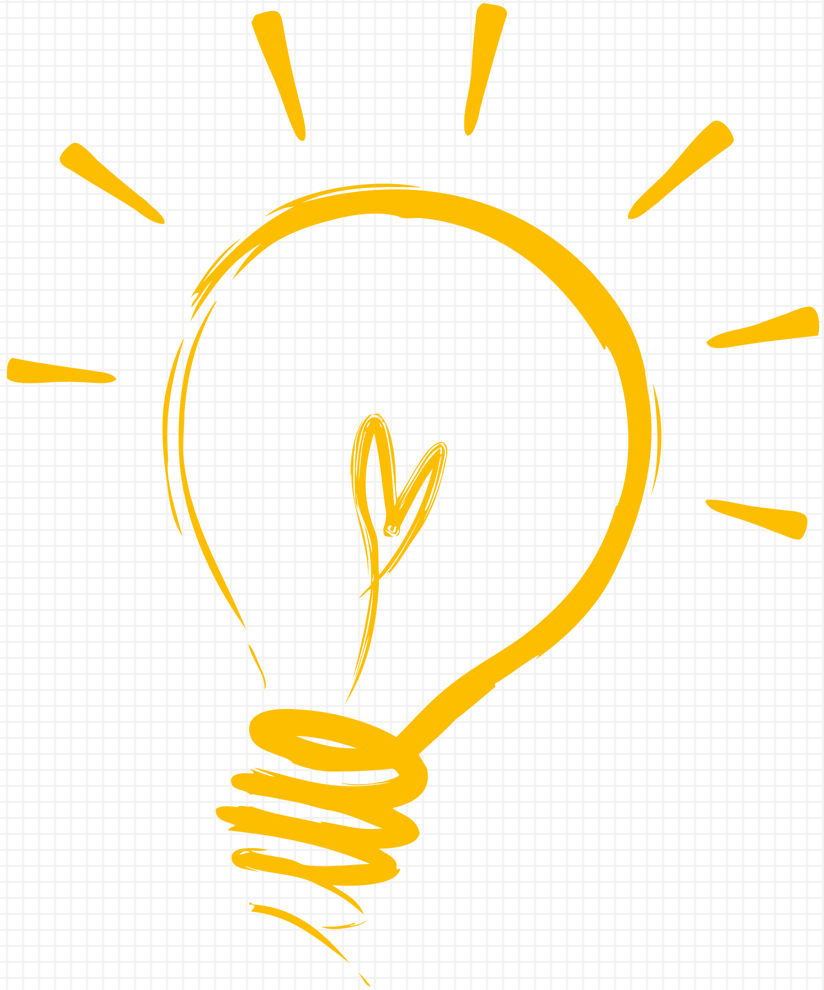


数据挖掘算法

梯度下降

Gridient Desent



CONTENTS

di yi zhang jie

第一章节

算法背景

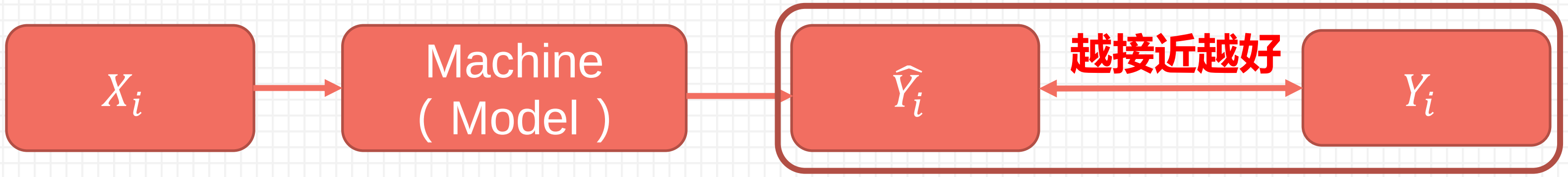
1.1 算法背景

近年来，随着计算机技术的快速发展，机器学习、深度学习越来越成为热门研究方向。

其实很多机器学习、深度学习的问题都可以看成一个优化问题，即给定机器（模型）一个输入，通过训练使得机器（模型）的输出与实际输出越接近越好。

我们可以定义一个Loss函数来表示机器的输出与实际输出的“距离”，通过优化该Loss函数使机器（模型）满足我们的要求，之后再将训练好的机器（模型）投入到实际应用中。

Train Data：$\{X_i, \ Y_i\}$，i = 1~N　　　Test　Data：$\{X_k, \ Y_k\}$，k = 1~M



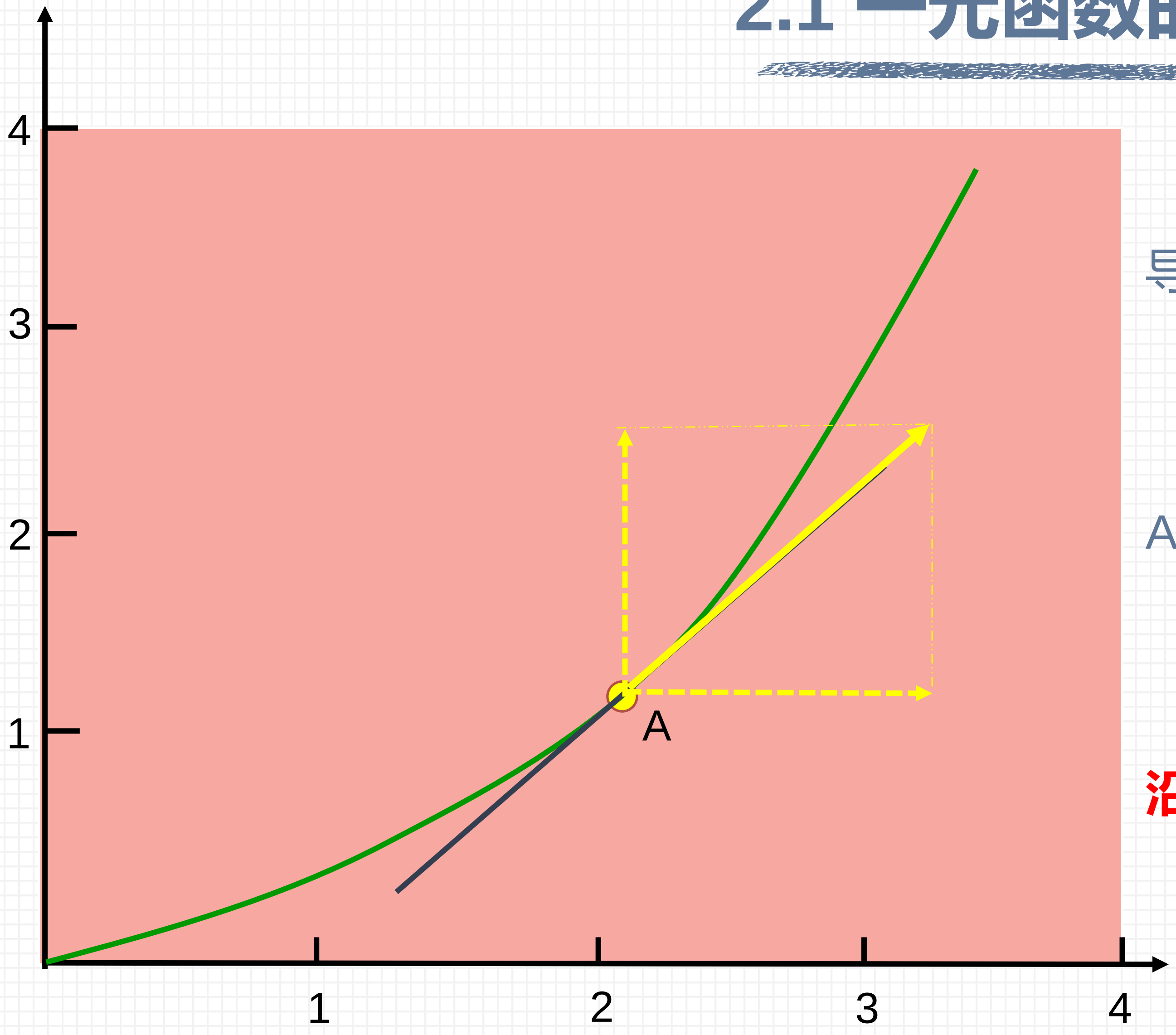
不妨令Loss = $(\hat{Y}_i - Y_i)^2$，通过某些方法调整Model的参数使得Loss逐渐变小直到满足要求，**梯度下降便是一种常用的优化方法。**

di yi zhang jie

第二章节

算法原理

2.1 一元函数的导数



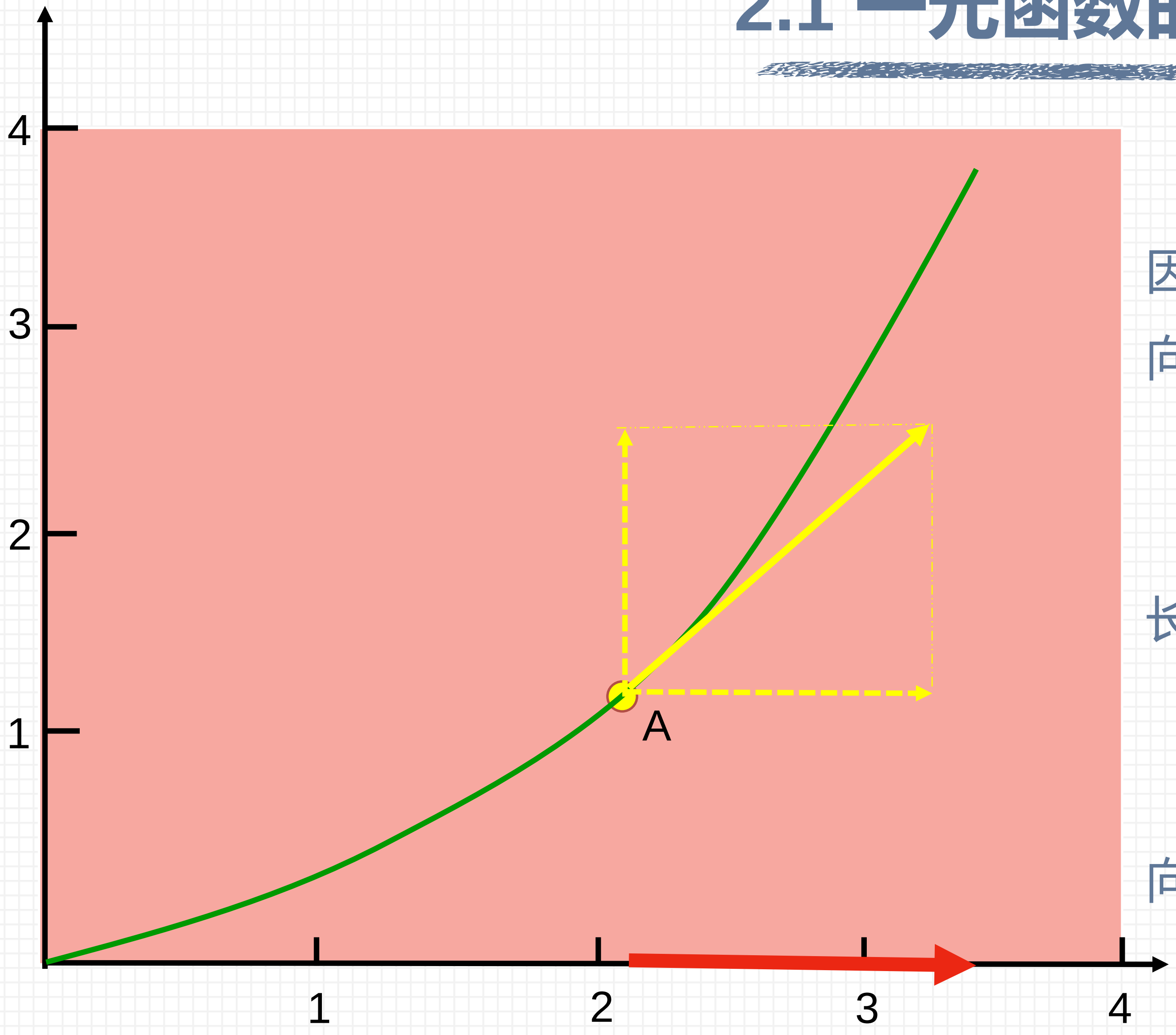
以一元二次函数的右半面为例，探讨某个点导数的意义。

以往我们会说点A处的导数大于0，函数在点A处是增长的。

事实上，我们应该这么说：“**函数在点A处，沿X轴正向**是增长的。”

也就是说，$\frac{dy}{dx}$**的意义暗含了方向性**。

2.1 一元函数的导数

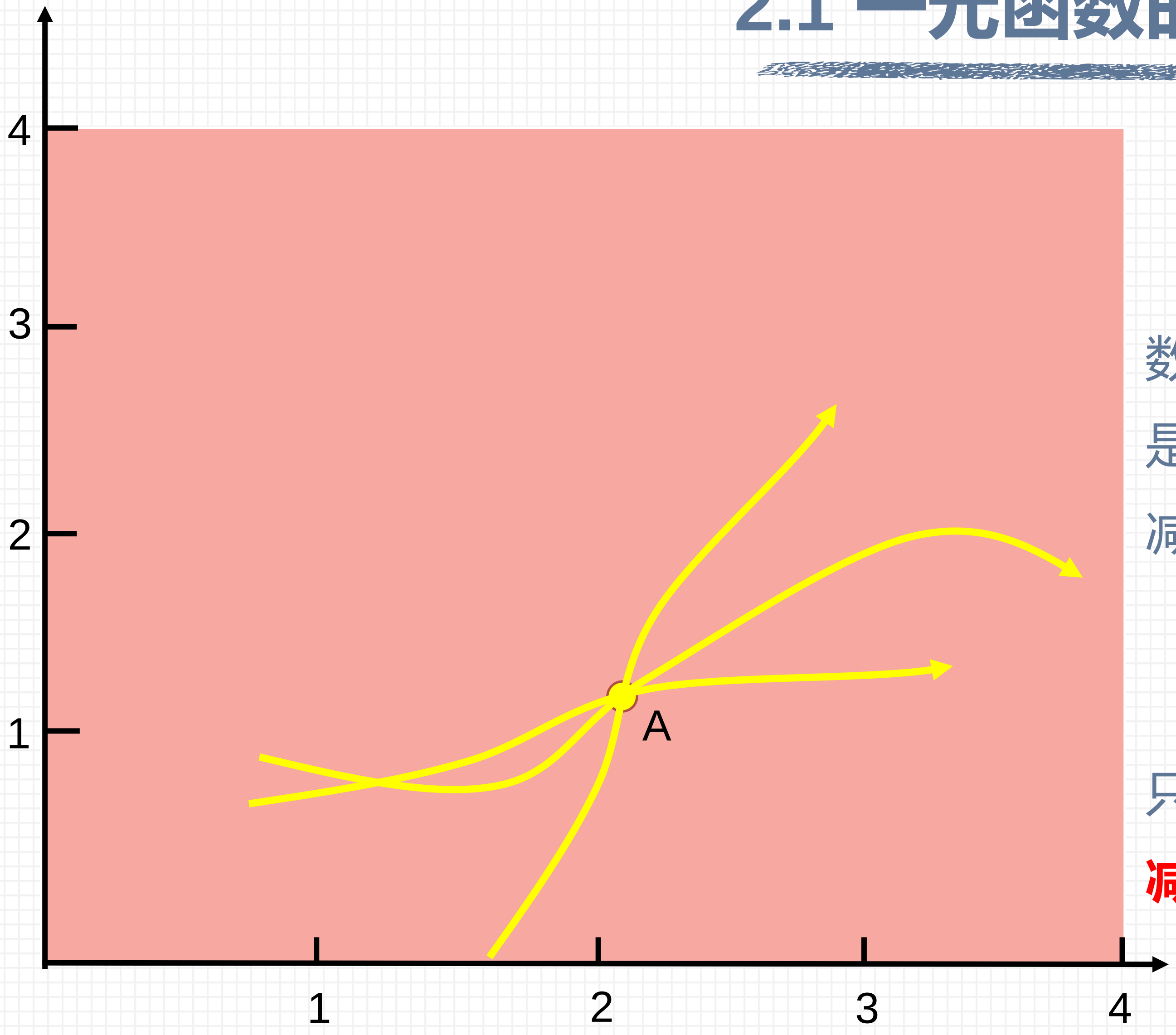


另外，函数在某个点处的导数是一个数值，、因此这时某个点的导数可以用一个一维向量表示，向量的长短表示导数大小。

比如导数值-2，就可以用一个以该点为起点，长度为2，方向指向x负向的向量表示。

导数值5，可用以该点为起点，长度为5，方向指向x正向的向量表示。

2.1 一元函数的导数



有了方向与大小，我们能得到什么?

一个很简单问题，如左图，已知某个一元函数经过点A，函数的具体形式未知，图像未知，但是函数在点A处的导数值为1，问如何微调x，使y减小?

答：点A的导数大于0，那么根据前边的思路，只需将**x值向左移动就可以使函数值y在小领域内减小**。

到这里，你已经感受到了梯度下降的思想。

2.2 二元函数的导数

导数的分解动画示意

二元函数的导数值，可以用平面上的一个向量表示。同时根据向量的分解，二元函数的某一点处的任意方向导数，总可以得到如下分解：

$$\boldsymbol{z} = a\frac{\partial z}{\partial x} + b\frac{\partial z}{\partial y}$$

$\frac{\partial z}{\partial x}$这里指的是沿x轴正向的单位向量，a是函数沿x轴正向的导数值。

N元函数的导数值，可以用N为空间的一个向量表示，同理。

2.3 梯度及性质

Def 1 方向导数：设三元函数 $f(x,y,z)$ 在点 $P_0(x_0,y_0,z_0)$的某个领域有定义，l 是从 P_0 出发的任意方向射线，记 $P(x,y,z)$是 P_0 小领域内的任意一点,记 ρ 为 P 到 P_0 的距离，若极限

$$\lim_{\rho\to0^+}\frac{f(P)-f(P_0)}{\rho}$$

存在，称这个极限为函数 f 在 P_0 沿方向 l 的方向导数。

容易看到，若函数 f 在点P_0存在关于x的偏导数，则函数 f 在点P_0沿x轴正向的方向导数恰好为

$$\frac{\partial f}{\partial l}=\frac{\partial f}{\partial x}$$

当l为x轴的负方向，则有

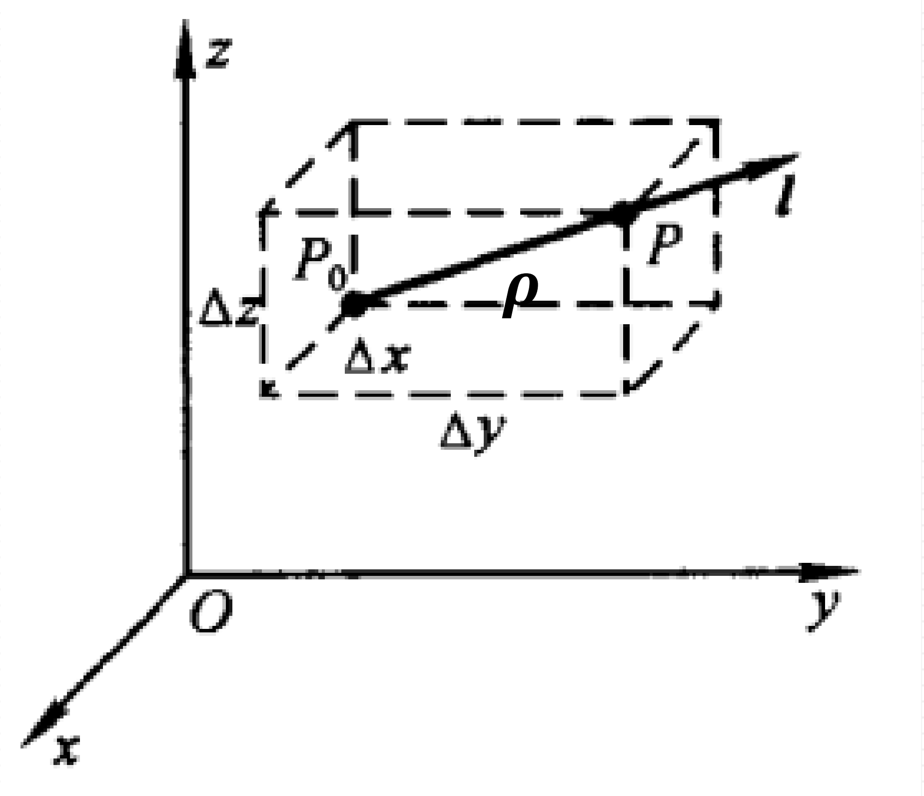
$$\frac{\partial f}{\partial l}=-\frac{\partial f}{\partial x}$$

2.3 梯度及性质

The 1 ：函数 $f(x,y,z)$ 在点 $P_0(x_0,y_0,z_0)$可微，则f 在点 P_0 任意方向 l 的方向导数均存在，且

$$f_l(P_0)=f_x(P_0)\cos\alpha+f_y(P_0)\cos\beta+f_z(P_0)\cos\gamma$$

其中$\cos\alpha\ \cos\beta\ \cos\gamma$分别为方向 l 与x,y,z轴的夹角余弦。



证明:

$$\Delta x=\rho\cos\alpha$$
$$\Delta y=\rho\cos\beta$$
$$\Delta z=\rho\cos\gamma$$

f可微则有

$$f(P)-f(P_0)=f_x(P_0)\Delta x+\ f_y(P_0)\Delta y+f_z(P_0)\Delta z+o(\rho)$$

左右同时除以ρ，取极限即可。

2.3 梯度及性质

Def 2 梯度：设三元函数 $f(x,y,z)$ 在点 $f(x_0,y_0,z_0)$的某个领域有定义，若f 对所有自变量均存在偏导数，那么称向量 $\operatorname{grad} f(P_0)=(f_x(P_0),f_y(P_0),f_z(P_0))$,为函数 f在点P_0的梯度。

The 2：函数 $f(x,y,z)$ 在点 $P_0(x_0,y_0,z_0)$**沿梯度方向增长最快。**

证明：取l方向上的单位向量$l_0(\cos\alpha,\cos\beta,\cos\gamma)$

$$f_l(P_0)=f_x(P_0)\cos\alpha+f_y(P_0)\cos\beta+f_z(P_0)\cos\gamma$$

$$f_l(P_0)=(f_x(P_0),f_y(P_0),f_z(P_0))\cdot(\cos\alpha,\cos\beta,\cos\gamma)=\operatorname{grad} f(P_0)\cdot l_0$$

$$|f_l(P_0)|=|\operatorname{grad} f(P_0)\cdot l_0|=|\operatorname{grad} f(P_0)|\cos\theta$$

当$\theta=0$时， $|f_l(P_0)|$达到最大值： $|\operatorname{grad} f(P_0)|$，也就是说

1、f 在点P_0的梯度方向是 f 的值增长最快的方向

2、沿这一方向的函数变化率就是 f 在点P_0的梯度的模

2.4 算法流程

```
For  t  in epoch:
    loss = 0
    # 将dataset数据带入model，计算误差（误差平方和）
    for d in dataset :
        loss +=   error(model(x) , y)

        # 计算loss对参数的梯度,并更新参数
```

$$error = \frac{1}{2}(\varpi_i x_i + b_i - y_i)^2, loss = \frac{1}{2n}\sum(\varpi_i x_i + b_i - y_i)^2 = \frac{1}{2n}(\varpi x + b - y)^T(\varpi x + b - y)$$

$$\frac{\partial loss}{\partial \varpi} = \frac{1}{n}(\varpi x + b - y) \cdot x \ , \quad \frac{\partial loss}{\partial b} = \frac{1}{n}(\varpi x + b - y)$$

$$w \ = w \ - lr * \frac{\partial loss}{\partial \varpi} = w \ - lr * \frac{1}{n}(\varpi x + b - y) \cdot x$$

$$b = b \ - lr * \frac{\partial loss}{\partial b} = w \ - lr * \frac{1}{n}(\varpi x + b - y) \cdot 1$$

di san zhang jie

第三章节

算法分析

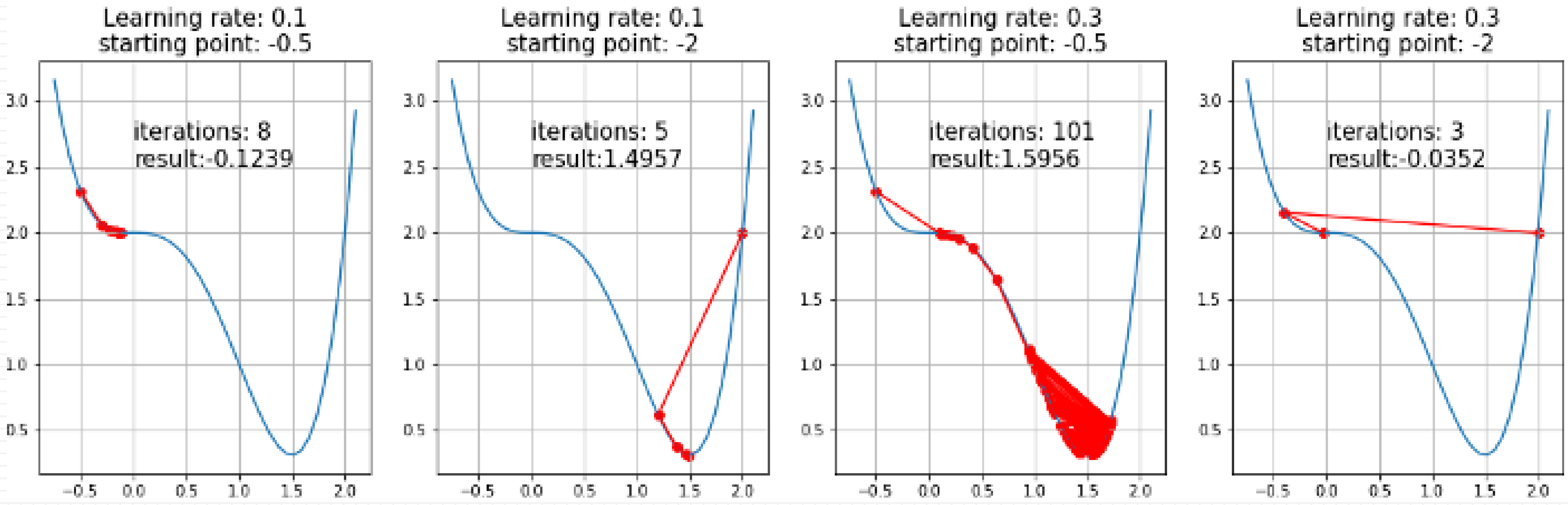
3.1 算法分析

优点：

1）原理比较简单，实现也是很容易。

缺点：

1）收敛速度慢、非凸优化难以收敛至全局最优。



https://towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21 di si zhang jie

第四章节

算法拓展

4.1 算法拓展

由于标准的梯度下降算法更新参数时，需要计算所有的训练样本的误差，然后反向求导，这样在动辄上万参数，需要大量样本训练的神经网络来说，耗时稍长。因此提出了随机梯度下降、小批次梯度下降、自适应梯度下降等等。

https://pytorch.org/docs/stable/optim.html di wu zhang jie

第五章节

案例实操

5.1 案例实操

略。

THANKS

谢谢观看