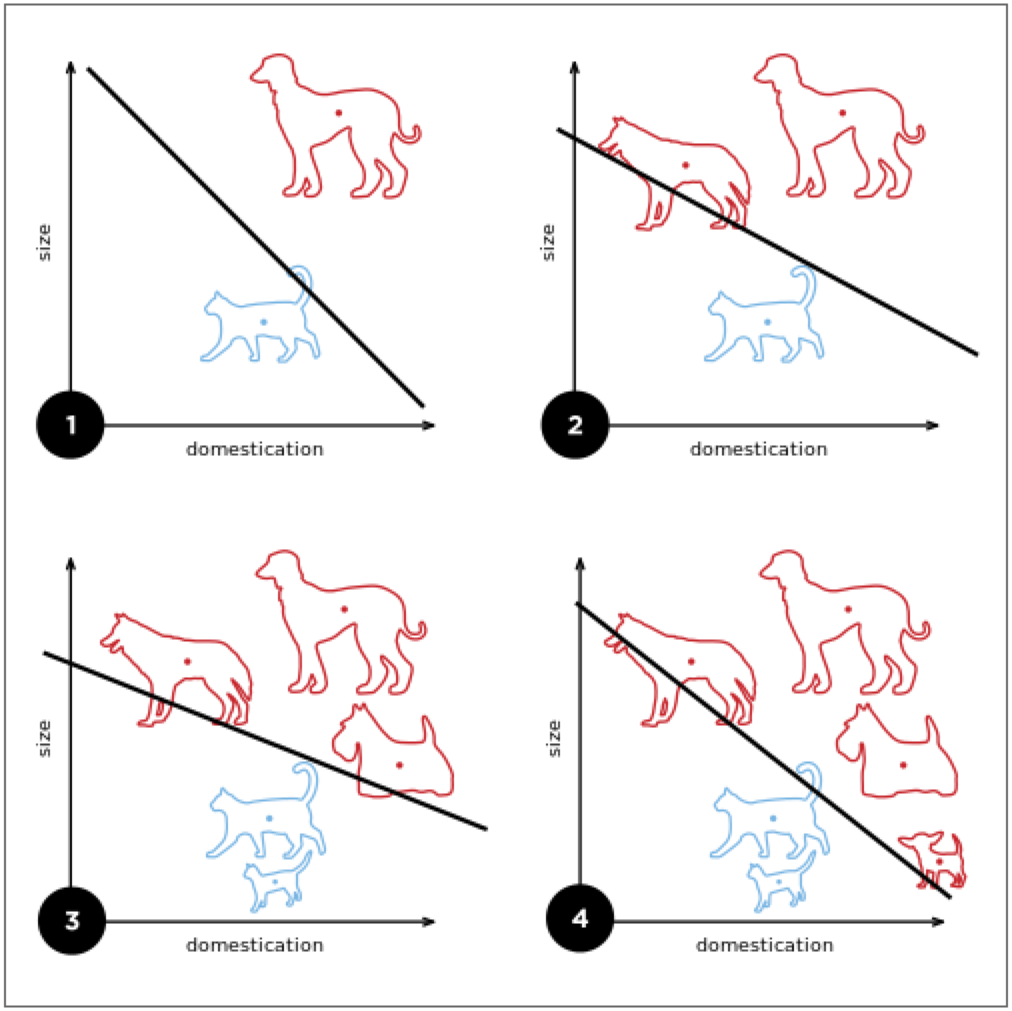
Perceptron

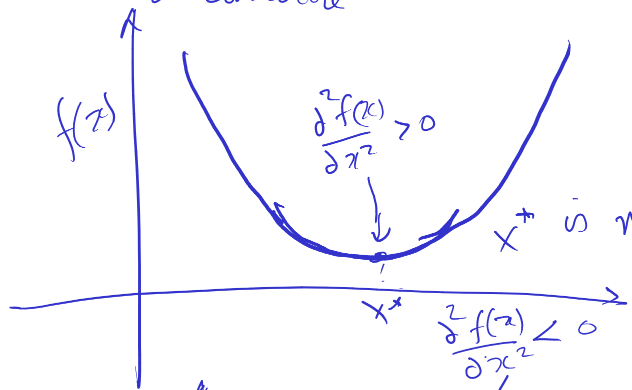


1
size
domestication
2
size
domestication
3
size
domestication
4
size
domestication

Second derivative

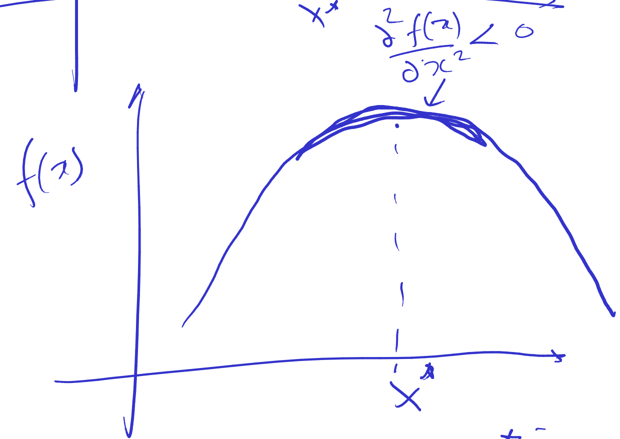
What is the role of $\underbrace{\text{second derivative}}_{\text{is curvature}}$ in optimization?

$$\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x^*} > 0$$



x^* is minimum $\left.\frac{\partial f}{\partial x}\right|_{x^*} = 0$

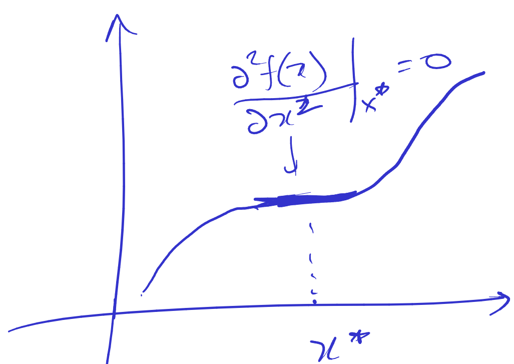
$$\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x^*} < 0$$



$\left.\frac{\partial f}{\partial x}\right|_{x^*} = 0$

x^* is maximum

$$\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x^*} = 0$$



x^* is an inflection point

Second derivative

How do the following functions differ?

$$f(x, y) = 2x^2 + 4y^2 - xy - 6x - 8y + 6$$

$$f(x, y) = -2x^2 - 4y^2 - xy - 6x - 8y + 6$$

$$f(x, y) = 2x^2 - 4y^2 - xy - 6x - 8y + 6$$

$$f(x, y) = 2x^2 + 4y^2 - xy - 6x - 8y + 6$$

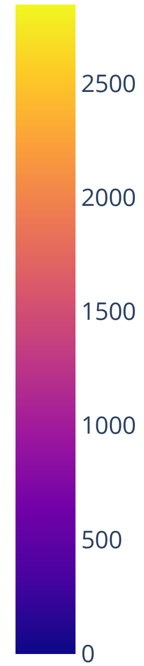
```
In [3]: def f(x, y):
            return 2*x**2 + 4*y**2 - x*y - 6*x - 8*y  + 6

        plot_surface(f)
```

$\frac{\partial^2 f}{\partial y^2} = 8 > 0$

$\frac{\partial^2 f(x,y)}{\partial x^2} = 4 > 0$

Please look at the notebook for visualization.

The surface looks is a bowl because second derivatives are positive.


2500
2000
1500
1000
500
0

$\frac{\partial^2 f}{\partial x^2} = -4 < 0$

$$f(x, y) = -2x^2 - 4y^2 - xy - 6x - 8y + 6$$

$\frac{\partial^2 f}{\partial y^2} < -4$

```
In [4]: def f(x, y): return - 2*x**2 - 4*y**2 - x*y - 6*x - 8*y  + 6
        plot_surface(f)
```

Please look at the notebook for visualization.

The surface looks is an upside down bowl because second derivatives are negative.


0
−500
−1000
−1500
−2000
−2500
−3000

$$\frac{\partial^2 f}{\partial x^2} = 4 > 0$$

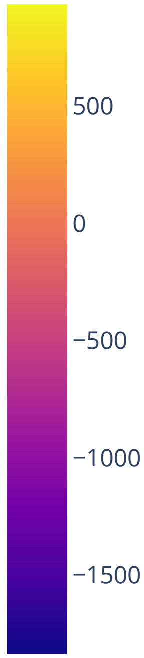
$$f(x,y) = 2x^2 - 4y^2 - xy - 6x - 8y + 6$$

$$\frac{\partial^2 f}{\partial y^2} = -8 < 0$$

```
In [5]: def f(x, y): return  2*x**2 - 4*y**2 - x*y - 6*x - 8*y  + 6
        plot_surface(f)
```

Please look at the notebook for visualization.

The surface looks like a saddle because one of the double derivatives is positive while other is negative.


500
0
−500
−1000
−1500

$a_{12} + a_{21} = -1$

$a_{12} = -\frac{1}{2}$

$a_{21} = -\frac{1}{2}$

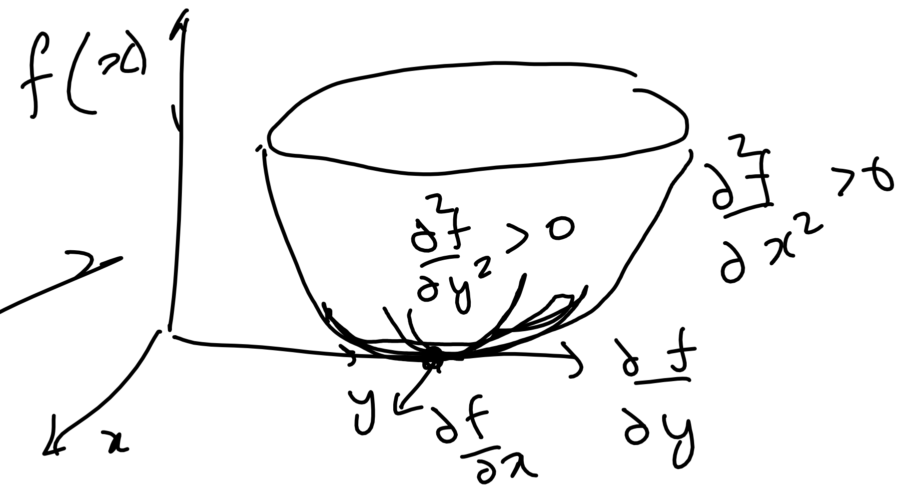
Hessians

$$f_1(x) = \underbrace{2x^2 + 4y^2 - xy}\ - 6x - 8y + 6$$

$$\frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x} = 4x - y - 6\right)$$

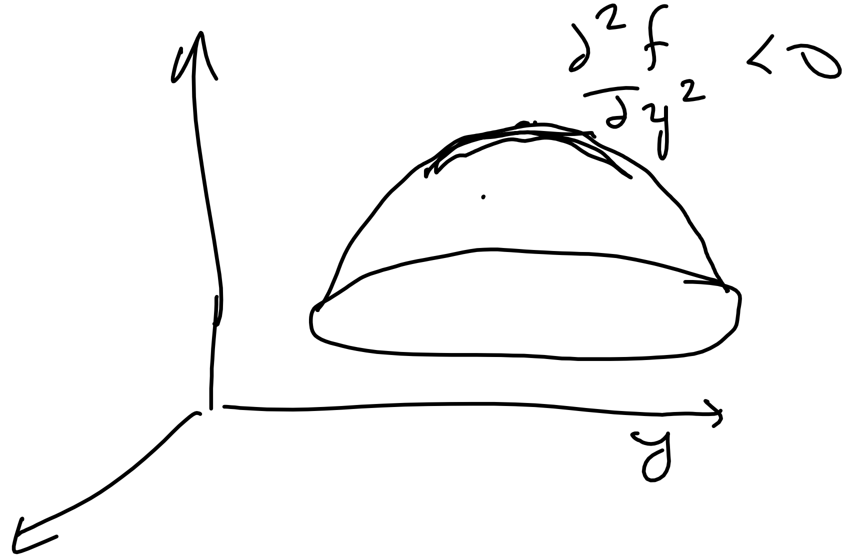
$$\frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y} = 8y - x - 8\right)$$

$$\left[\ \frac{\partial^2 f(x)}{\partial x^2} = 4 \qquad \frac{\partial^2 f(x)}{\partial y^2} = 8 \qquad \frac{\partial^2 f(x)}{\partial x \partial y} = -1 \qquad \frac{\partial^2 f(x)}{\partial y \partial x} = -1\ \right]$$

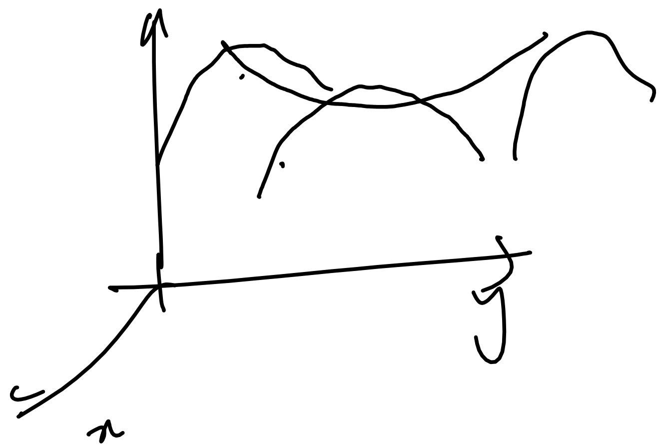


$$\frac{\partial^2 f}{\partial x^2} > 0,\ \frac{\partial^2 f}{\partial y^2} > 0 \Rightarrow \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)\bigg|_{x^*} \text{ is minimum}$$

$$\frac{\partial^2 f}{\partial x^2} < 0 \;,\; \frac{\partial^2 f}{\partial y^2} < 0 \Rightarrow$$



$$\frac{\partial^2 f}{\partial x^2} < 0 \;,\; \frac{\partial^2 f}{\partial y^2} > 0 \Rightarrow$$



$$H_{xx}\, f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} = \frac{\partial}{\partial x} \frac{\partial f}{\partial x} = \frac{\partial}{\partial x} \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix}$$

$$f(\underline{x}) = \underbrace{\underline{x}^T A \underline{x}} + \underline{b}^T \underline{x} + c$$

$$\frac{\partial}{\partial \underline{x}} f(\underline{x}) = 2\underline{x}^T A + b^T + 0^T$$

$$g(x) = 2\underline{x}^T A + b^T$$

$$\frac{\partial}{\partial \underline{x}} g(\underline{x}) = 2x^T [\underline{a}_1, a_2, \ldots, a_n]$$

$$= 2[\underline{x}^T \underline{a}_1, \underline{x}^T \underline{a}_2, \ldots \underline{x}^T \underline{a}_n]$$

$$\frac{\partial}{\partial \underline{x}} g(x) = 2 \begin{bmatrix} a_1^T \\ \underline{a}_2^T \\ \vdots \\ \underline{a}_n^T \end{bmatrix} = 2A^T$$

$$\frac{\partial}{\partial \underline{x}} g(x) = 2A$$

$$\frac{\partial}{\partial \underline{x}} \underline{x}^T A \underline{x}$$

$$= \underline{x}^T (A + A^T)$$

$$= 2x^T A$$

$$\frac{\partial}{\partial \underline{x}} \underline{b}^T \underline{x} = b^T$$

$$\frac{\partial}{\partial \underline{x}} \underline{x}^T \underline{b} = \underline{b}^T$$

$$\frac{\partial}{\partial \underline{x}} x^T A$$

$$\frac{\partial}{\partial x} A^T x$$

$$A = \begin{bmatrix} 2 & -1/2 \\ -1/2 & 4 \end{bmatrix}, \quad a_1 = \begin{bmatrix} 2 \\ -1/2 \end{bmatrix}, \quad a_2 \begin{bmatrix} -1/2 \\ 4 \end{bmatrix}$$

Symmetric

$$A^T = A$$

$$\frac{\partial^2 f(x)}{\partial x^2} = 2A = H_{xx} f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

$$\frac{\partial^2 f}{\partial x^2} > 0, \quad \frac{\partial^2 f}{\partial y^2} > 0 \Rightarrow ?$$

Any square matrix can be either

① Positive definite $\quad 2A \succ 0$

a sq matrix $A \in \mathbb{R}^{n \times n}$ is PD if for all $\underline{x} \in \mathbb{R}^n$

$\underline{x}^T A \underline{x} > 0$

Positive semi definite $A \succeq 0$

$\forall x \; \underline{x}^T A \underline{x} \geq 0$

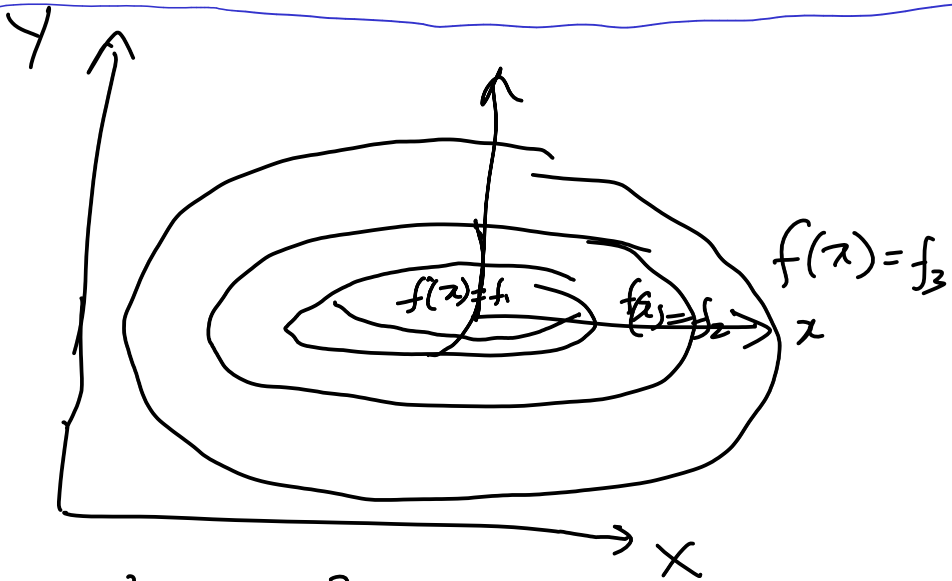
② Negative definite $\quad A \prec 0 \Rightarrow \underline{x}^T A \underline{x} < 0$

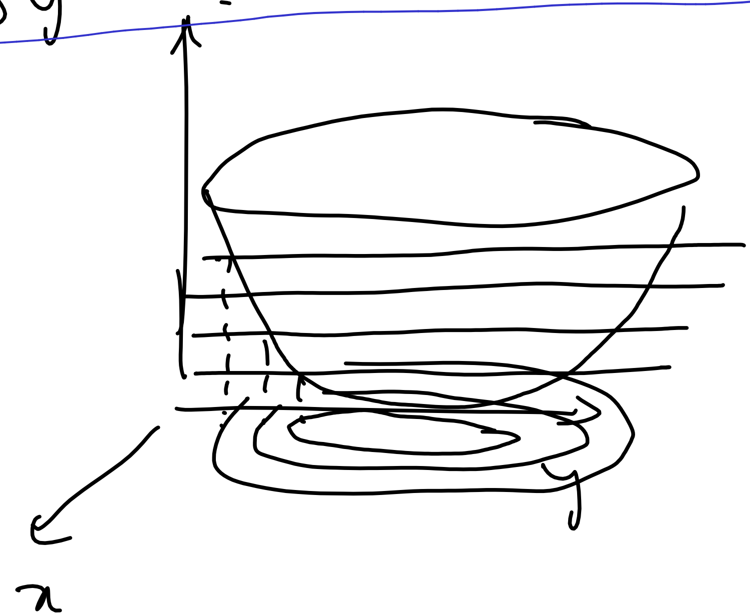
Negative semi definite $\quad A \preceq 0 \Rightarrow x^T A x \leq 0$

$\frac{\partial^2 f}{\partial x^2} < 0 \quad \frac{\partial^2 f}{\partial y^2} < 0 \qquad A = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & 0 \\ 0 & \frac{\partial f}{\partial y^2} \end{bmatrix}$

3 Indefinite

$$\frac{\partial^2 f}{\partial x^2} < 0, \quad \frac{\partial^2 f}{\partial y^2} > 0$$





$$\frac{\partial^2 f}{\partial x^2} > 0 \quad \frac{\partial^2 f}{\partial y^2} > 0$$

$\forall x \quad \hat{x}^T A \hat{x} > 0$

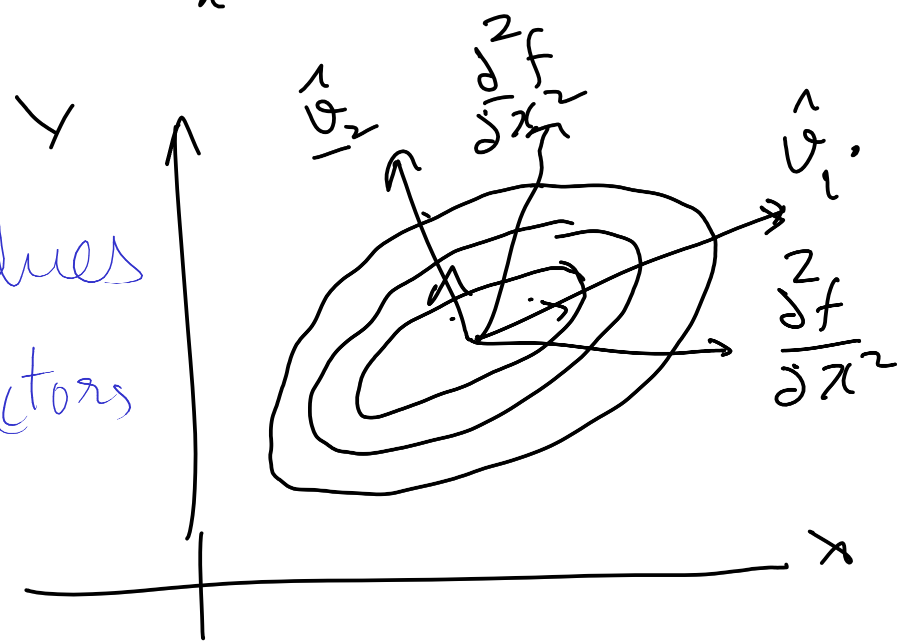
eigen values

eigen vectors

$$A \underline{v} = \lambda \underline{v}$$

A is PD if $\lambda_i > 0$

all eigen values > 0



$$f(\underline{x}) = \underline{x}^T A \underline{x} + \underline{b}^T \underline{x} + c$$

$$H_{xx} f(x) = 2A$$

$$\left.\frac{\partial f(x)}{\partial \underline{x}}\right|_{x^*} = 0$$ at extreme points for all function

The extreme point is

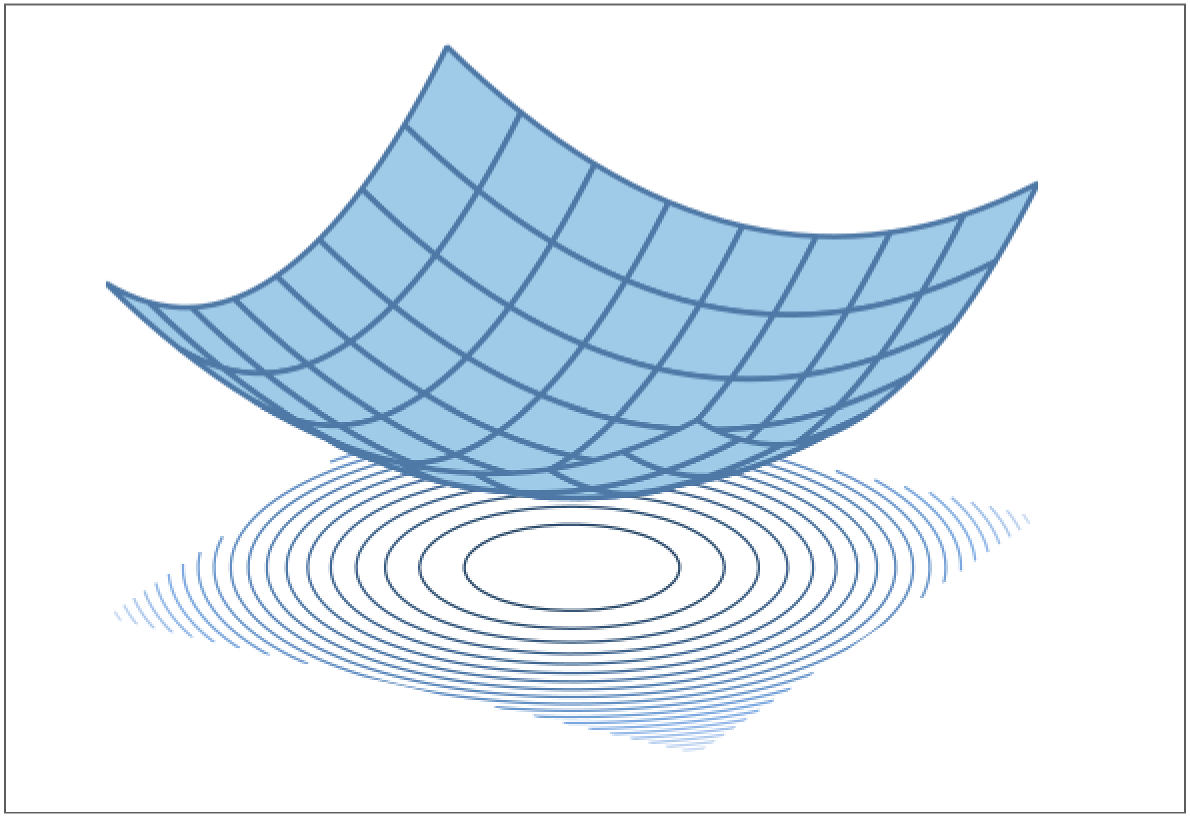
① a minimum point if $\underbrace{H_{xx} f(x)}$ > 0 is PD or PSD

→ all eigen values > 0

② a max " if $H_{xx} f(x) < 0$. is ND or NSD

③ either inflexion point or a saddle point

Contour Plots



```
In [7]: def f(x, y): return  2*x**2 + 4*y**2 - x*y - 6*x - 8*y  + 6
        plot_contour(f)
```

Isosurface

$$S(f,c) = \{(x,y) : f(x,y) = c\}$$

$$S(f,c) = \{ \underline{x} : f(\underline{x}) = c \}$$

$$f(x,y) = x^2 + y^2$$

$$f(\underline{x}) = \underline{x}^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \underline{x}$$

$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

$$\underbrace{S(f,c) = \{(x,y) : x^2 + y^2 = c\}}_{\text{implicit equation of circle with center } (0,0) \text{ radius } (\sqrt{c})}$$

$$c > 0$$

$$\underline{x} = \underline{g}(c, \theta) =$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \sqrt{c} \cos\theta \\ \sqrt{c} \sin\theta \end{bmatrix}$$

parameteric equation of circle

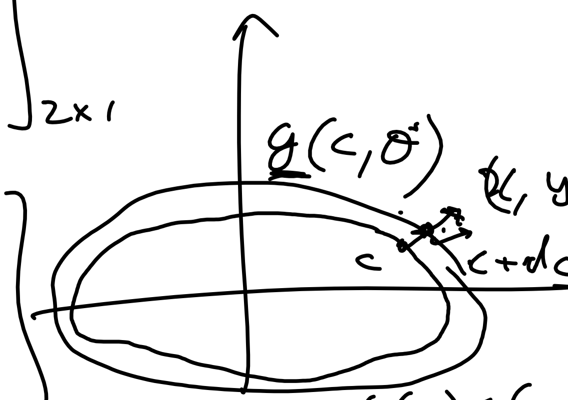
$$\nabla_x^T f(x) = 2x^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \underline{\underline{2\underline{x}^T}}$$

$$f(\underline{x}) = \underline{x}^T \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{I} \underline{x}$$

$$\underbrace{\left[\frac{\partial \underline{g}(c,\theta)}{\partial c}\right]}_{\perp\, S(f,c)} = \frac{\partial}{\partial c}\begin{bmatrix} \sqrt{c}\cos\theta \\ \sqrt{c}\sin\theta \end{bmatrix}_{2\times 1} = \begin{bmatrix} \frac{1}{2\sqrt{c}}\cos\theta \\ \frac{1}{2\sqrt{c}}\sin\theta \end{bmatrix}_{2\times 1}$$

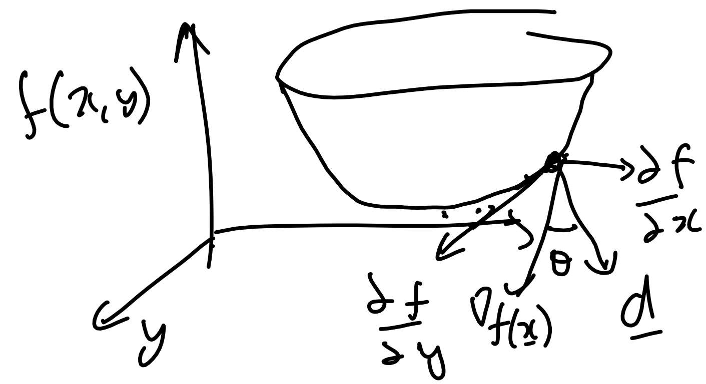
$$= \begin{bmatrix} \frac{\sqrt{c}\cos\theta}{2c} \\ \frac{\sqrt{c}\sin\theta}{2c} \end{bmatrix}_{2\times 1} = \frac{1}{2c}\begin{bmatrix} \sqrt{c}\cos\theta \\ \sqrt{c}\sin\theta \end{bmatrix}_{2\times 1}$$

$$= \underbrace{\frac{1}{2c}}\,\underline{x} \;\propto\; \underbrace{\nabla_x f(x) = 2\underline{x}}$$



$$\frac{\partial}{\partial \underline{x}} f(\underline{x}) = \nabla_x^T f(\underline{x}) = \left[\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right]$$

Directional derivative



$$D_{\underline{d}} f(\underline{x}) = \lim_{h \to 0} \frac{\boxed{f(\underline{x} + h\,\underline{d})} - f(\underline{x})}{h}$$

$$\frac{\partial f(\underline{x})}{\partial x} = \lim_{h \to 0} \frac{f\left(\underline{x} + h\begin{bmatrix}1\\0\end{bmatrix}\right) - f(\underline{x})}{h}$$

$$\frac{\partial f(\underline{x})}{\partial y} = \lim_{h \to 0} \frac{f\left(\underline{x} + h\begin{bmatrix}0\\1\end{bmatrix}\right) - f(\underline{x})}{h}$$

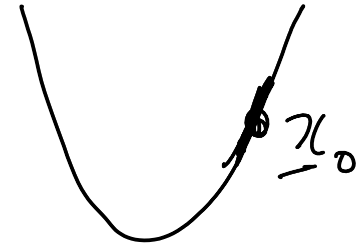
$$\frac{d f(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

$$D_{\underline{d}} f(x) = \underbrace{\nabla_{\underline{x}}^T f(x)}\ \underline{d} = \underbrace{\begin{bmatrix} \frac{\partial f(x)}{\partial x} & \frac{\partial f(x)}{\partial y} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}}$$ dot product

$$= \|\nabla_x f(x)\| \, \|\underline{d}\| \cos\theta$$

$$\underline{a} \cdot \underline{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 3 \end{bmatrix} \qquad \underline{a}^T \underline{b} = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Taylor series



$$f(x) = f(x_0) + \frac{df(x)}{dx}\Big|_{x_0}(x-x_0) + \frac{1}{2!}\frac{d^2f(x)}{dx^2}\Big|_{x_0}(x-x_0)^2 + \dots + \frac{1}{n!}\frac{d^nf(x)}{dx^n}\Big|_{x_0}(x-x_0)^n + \dots \infty$$

Vector valued function:

$$\underline{f}(\underline{x}) = \underline{f}(\underline{x}_0) + \nabla_{\underline{x}}^T f(\underline{x})\Big|_{\underline{x}_0}\underbrace{(\underline{x}-\underline{x}_0)}_{\substack{0.001 \\ 10^{-3}}} + \frac{1}{2!}(\underline{x}-\underline{x}_0)^T H_{\underline{x}} f(x)\Big|_{x_0}\underbrace{(\underline{x}-\underline{x}_0)}_{\substack{0.000001 \\ 10^{-6}}} + \dots + \infty$$

To minimize a function

$$\min_{\underline{x}} f(\underline{x})$$

$$\frac{\partial f(\underline{x})}{\partial \underline{x}} = 0$$ Solve this equation

But the equation might not be "solvable"

$f(x) = x^3 - 4x^2 + 8x + 6$

$\frac{\partial f(x)}{\partial x} = x^2 - 8x + 6$ solvable

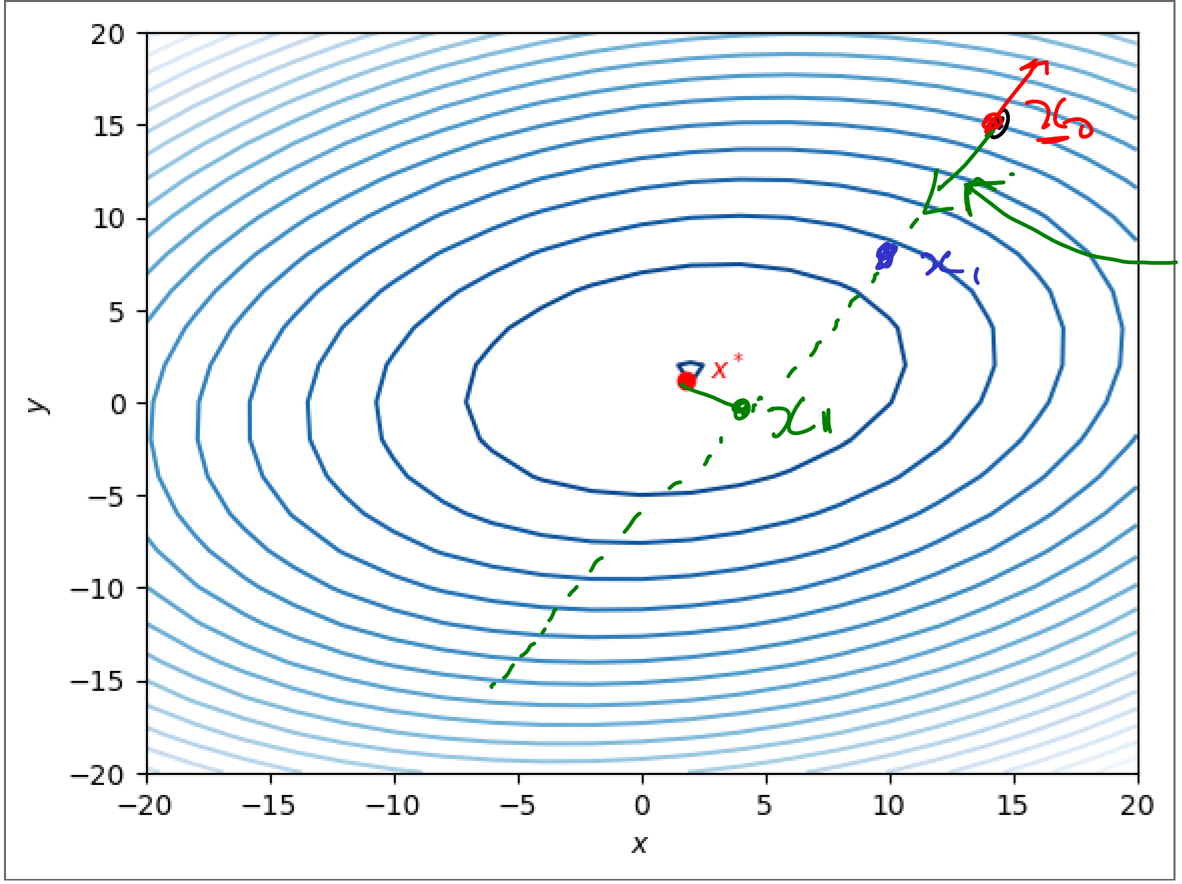
$\frac{\partial f(x)}{\partial x} = x \exp(x+2) + \sin(x+3)$ (No solution)

closed form analytic

This method works only when $\frac{\partial f(\underline{x})}{\partial \underline{x}} = 0$ has closed form solutions

Iterative methods: Gradient descent

Random starting point $= x_0$



$\nabla_x f(\underline{x})\big|_{x_0}$

$\min_{\underline{x}} f(\underline{x})$

$-\nabla_x f(\underline{x})\big|_{x_0}$

$\underline{x}_1 = \underline{x}_0 - \alpha \nabla_x f(\underline{x})$

$\in \mathbb{R}$

$\alpha = \arg\min_{\alpha} f(x_0 - \alpha \nabla_x f(x))$

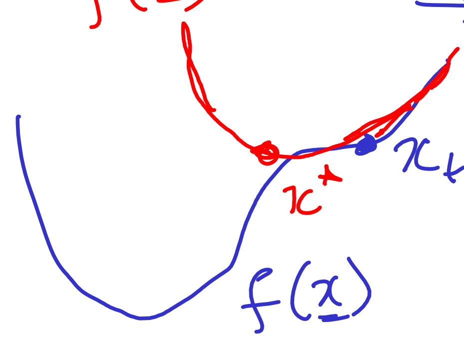
$\alpha = 0.01$ = step size

$\underline{x}_{t+1} = \underline{x}_t - \alpha_t \nabla_x f(\underline{x})$

Starting at random point $\underline{x}_0$ $\qquad \min_{\underline{x}} f(\underline{x})$

$$\underline{x}_{t+1} = \underline{x}_t - \alpha_t \nabla_x f(\underline{x}) \Big|_{\underline{x}_t}$$

$$f(\underline{x}_{t+1}) = \underbrace{\boxed{f(\underline{x}_t) + \nabla_x^T f(\underline{x}) (\underline{x}_{t+1} - \underline{x}_t) + \frac{1}{2} (\underline{x}_{t+1} - \underline{x}_t)^T H f(x) (\underline{x}_{t+1} - \underline{x}_t)}}_{\hat{f}(\underline{x})} + \underbrace{O\left((\underline{x}_{t+1} - \underline{x}_t)^3\right)}_{\to 0}$$



$$\underline{x}_{t+1} = \arg\min_{\underline{x}_{t+1}} \hat{f}(x)$$

$$\underline{x}_{t+1} - \underline{x}_t = -\left[H_x f(x_t)\right]^{-1} \nabla_x f(\underline{x})$$

$$\underset{\in \mathbb{R}^{n+1}}{\underline{x}_{t+1}} = \underset{\in \mathbb{R}^{n\times 1}}{\underline{x}_t} - \underbrace{\left[H_x f(x)\right]^{-1}}_{\in \mathbb{R}^{n\times n}} \underbrace{\nabla_x f(\underline{x})}_{\in \mathbb{R}^{n\times 1}}$$

Newton's method for optimization

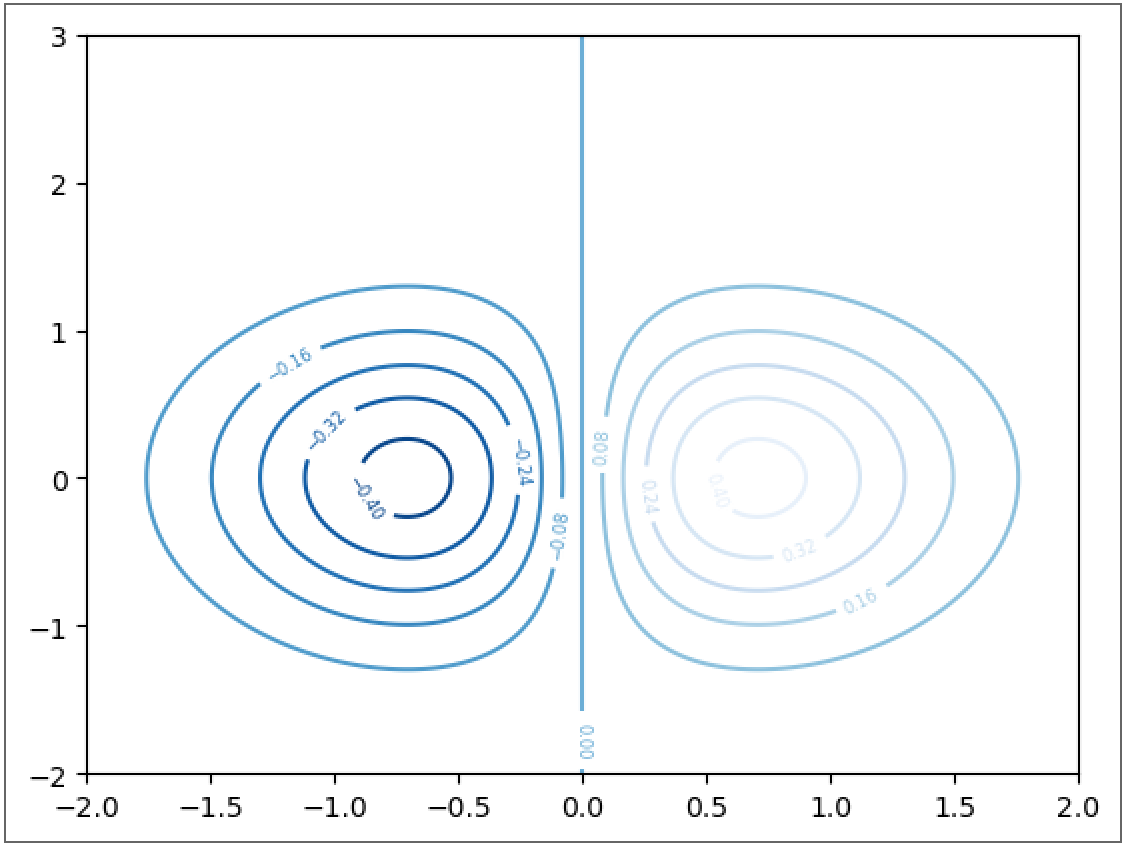
$n \to 10^6 \qquad 10^6 \times 10^6$

$x_{t+1} = x_t - \alpha_t \nabla_x f(x)$ ← For most NN

But how about other kinds of functions say:

$$\arg\min_x f(x) = x\exp(-(x^2+y^2))$$

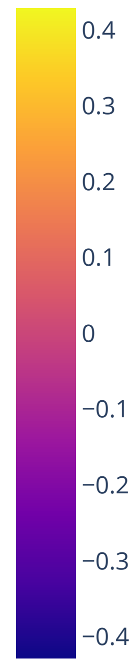
```
In [9]: def f(x,y): return  x * np.exp(-(x**2 + y**2))
        plot_contour(f)
```




```
In [10]: def plot_surface_3d(func):
             x, y = np.mgrid[-2:2:201j,
```

```
                        -2:3:201j]
    f = func(x,y)
    fig = go.Figure(data=[go.Surface(z=f, x=x, y=y,
                                     contours = {
                                         "x": {"start": -2, "end": 2, "size": 0.2},
                                         "z": {"start": -2, "end": 2, "size": 0.2}
                                     },
                                     )])
    fig.update_traces(contours_z=dict(show=True, usecolormap=True, project_z=True))
    fig.show()
```

```
In [11]: plot_surface_3d(f)
```

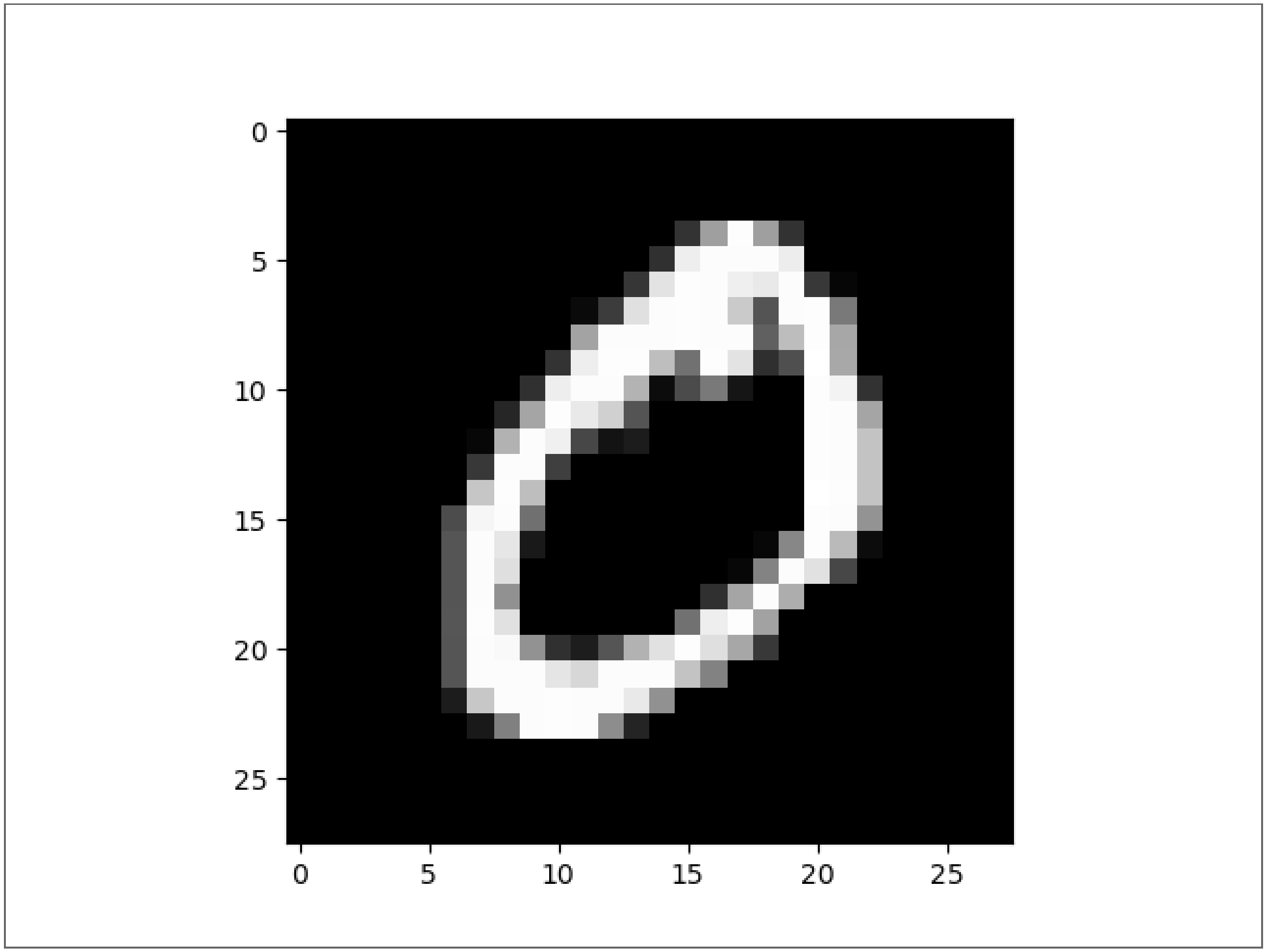


Taylor series

Taylor series vectorized

In [15]:
```
zero_images_anim
```

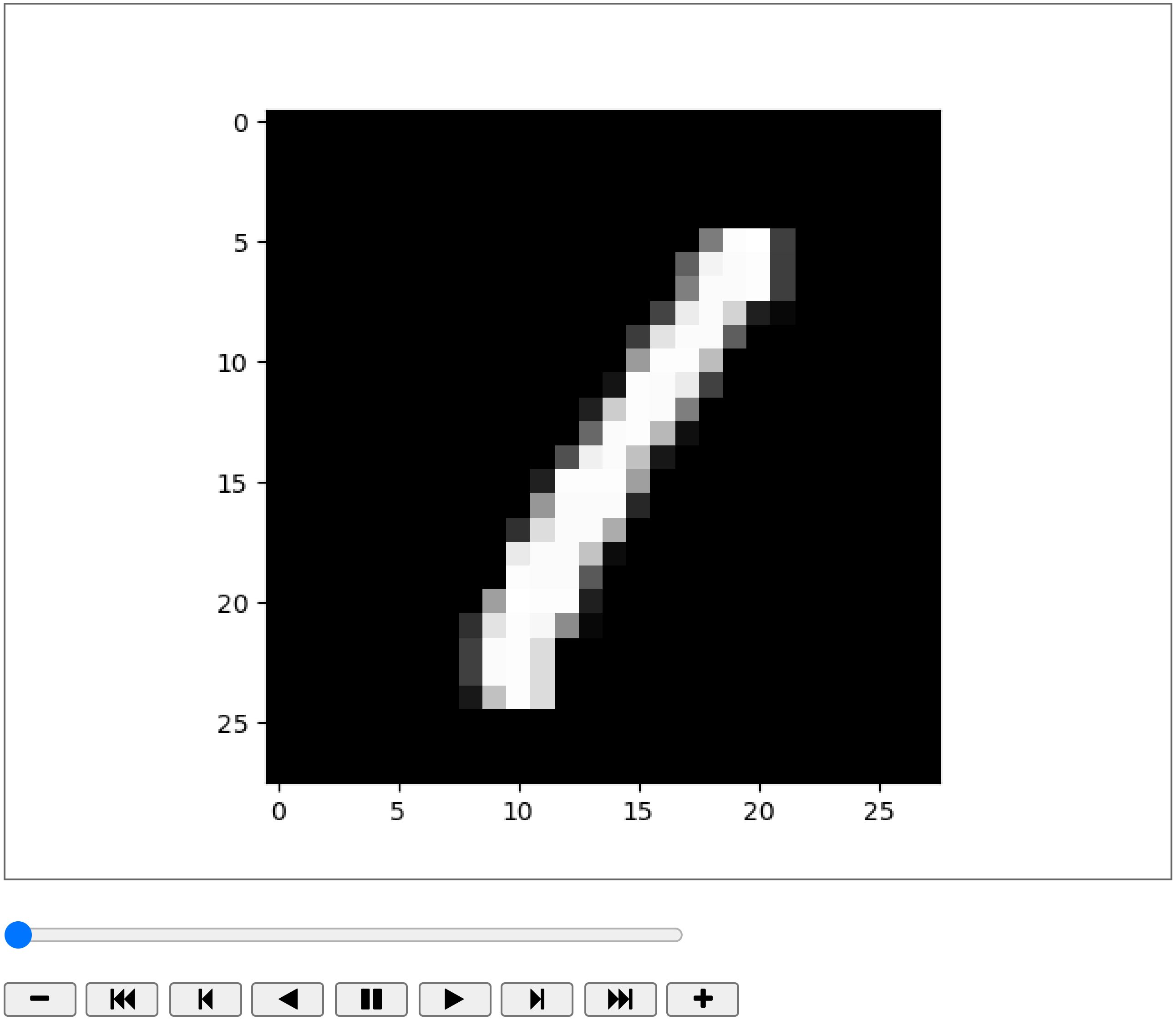
Out[15]:



Once Loop Reflect

In [17]:
```
one_images_anim
```

Out[17]:



◉ Once ○ Loop ○ Reflect

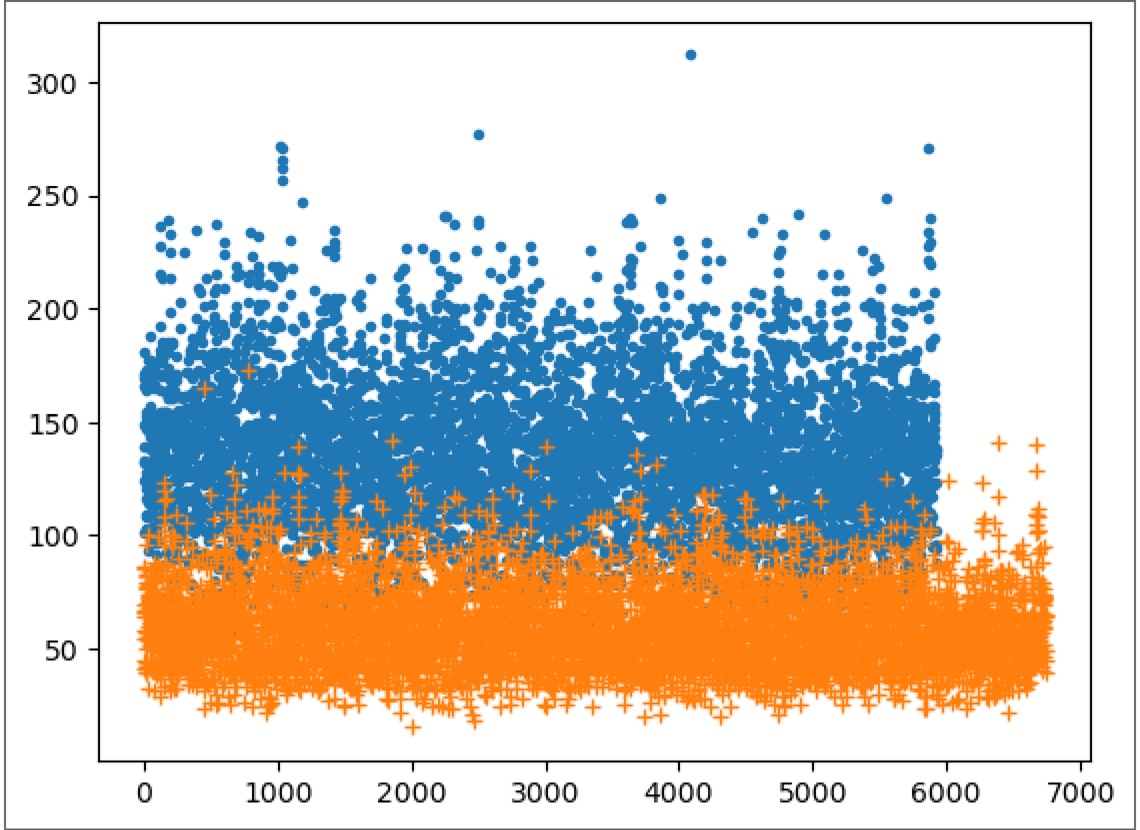
What is a feature

Any property of data sample that helps with the task.

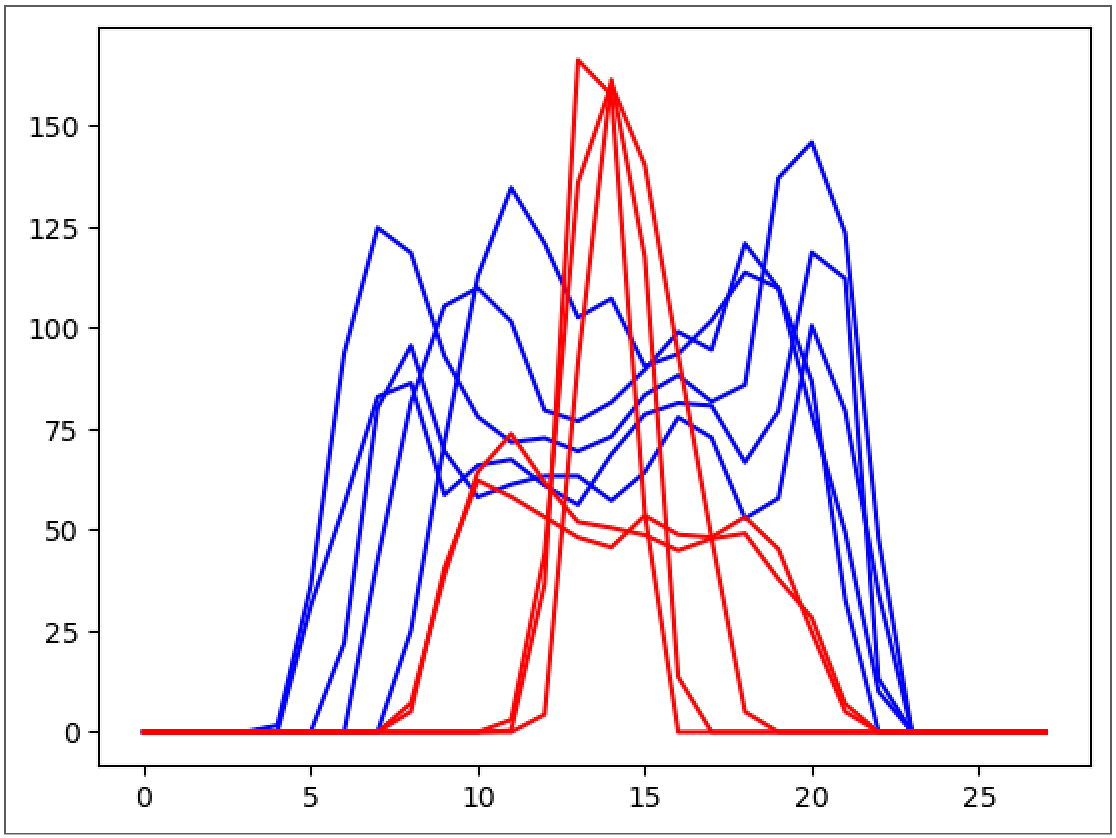
In [18]:
```
def feature_n_pxls(imgs):
            n, *shape = imgs.shape
    return np.sum(imgs[:, :, :].reshape(n, -1) > 128, axis=1)

n_pxls_zero_images = feature_n_pxls(zero_images)
n_pxls_one_images = feature_n_pxls(one_images)
fig, ax = plt.subplots()
ax.plot(n_pxls_zero_images, '.')
ax.plot(n_pxls_one_images, '+')
```

Out[18]:

```
[<matplotlib.lines.Line2D at 0x7f838d398490>]
```


300
250
200
150
100
50
0
1000
2000
3000
4000
5000
6000
7000

```
In [19]: fig, ax = plt.subplots()
         for i in range(5):
    ax.plot(zero_images[i].mean(axis=0), 'b-')
for i in range(5):
    plt.plot(one_images[i].mean(axis=0), 'r-')
```



In [20]:
```
wts = zero_images[0].mean(axis=0)
mean = (np.arange(wts.shape[0]) * wts).sum() / np.sum(wts)
var = ((np.arange(wts.shape[0]) - mean)**2 * wts).sum() / np.sum(wts)
var
```

Out[20]:

```
22.811061800377757
```

In [21]:

```python
def feature_y_var(img):
    wts = img.mean(axis=0)
    mean = (np.arange(wts.shape[0]) * wts).sum() / np.sum(wts)
    var = ((np.arange(wts.shape[0]) - mean)**2 * wts).sum() / np.sum(wts)
    return var
feature_y_var(zero_images[0]), feature_y_var(one_images[0])
```

Out[21]:

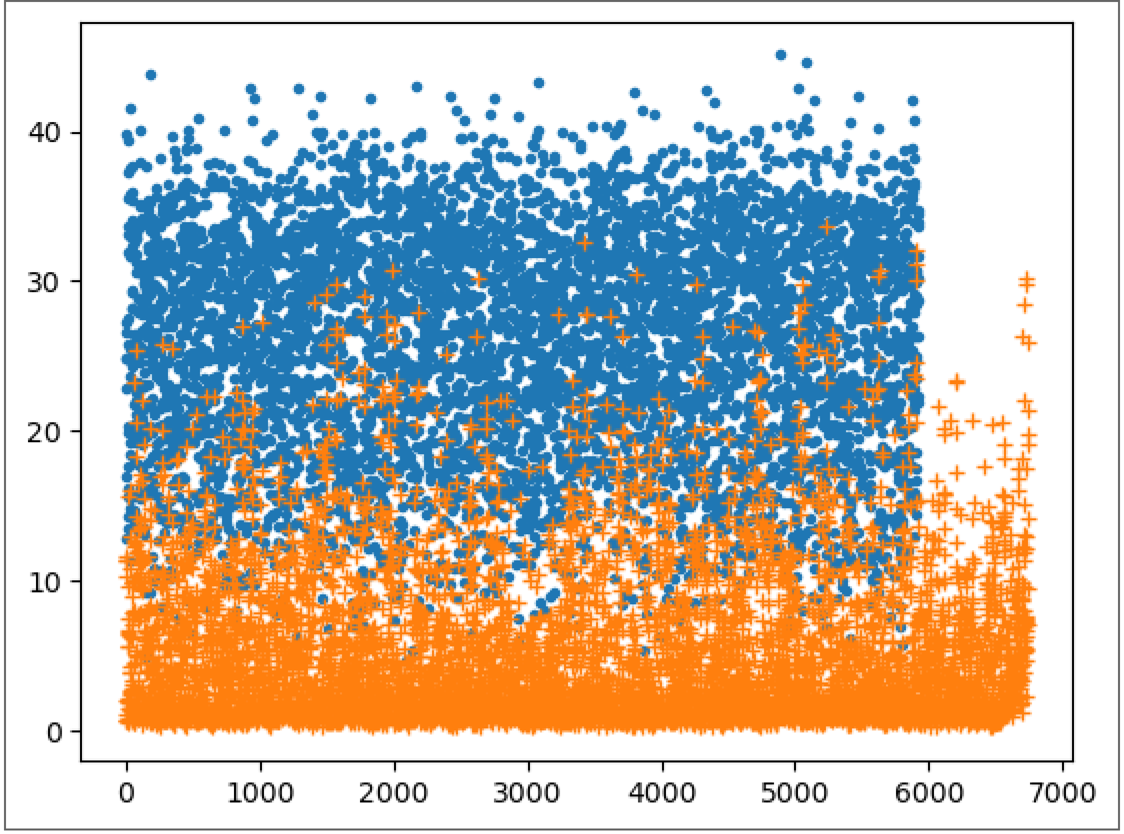
```
(22.811061800377757, 11.384958735403274)
```

In [22]:

```python
def feature_y_var(imgs):
    wts = imgs.mean(axis=-2)
    arange = np.arange(wts.shape[-1])
    mean = (arange * wts).sum(axis=-1) / wts.sum(axis=-1)
    mean = mean[:, np.newaxis]
    var = ((arange - mean)**2 * wts).sum(axis=-1) / wts.sum(axis=-1)
    return var

fig, ax = plt.subplots()
ax.plot(feature_y_var(zero_images), '.')
ax.plot(feature_y_var(one_images), '+')
```

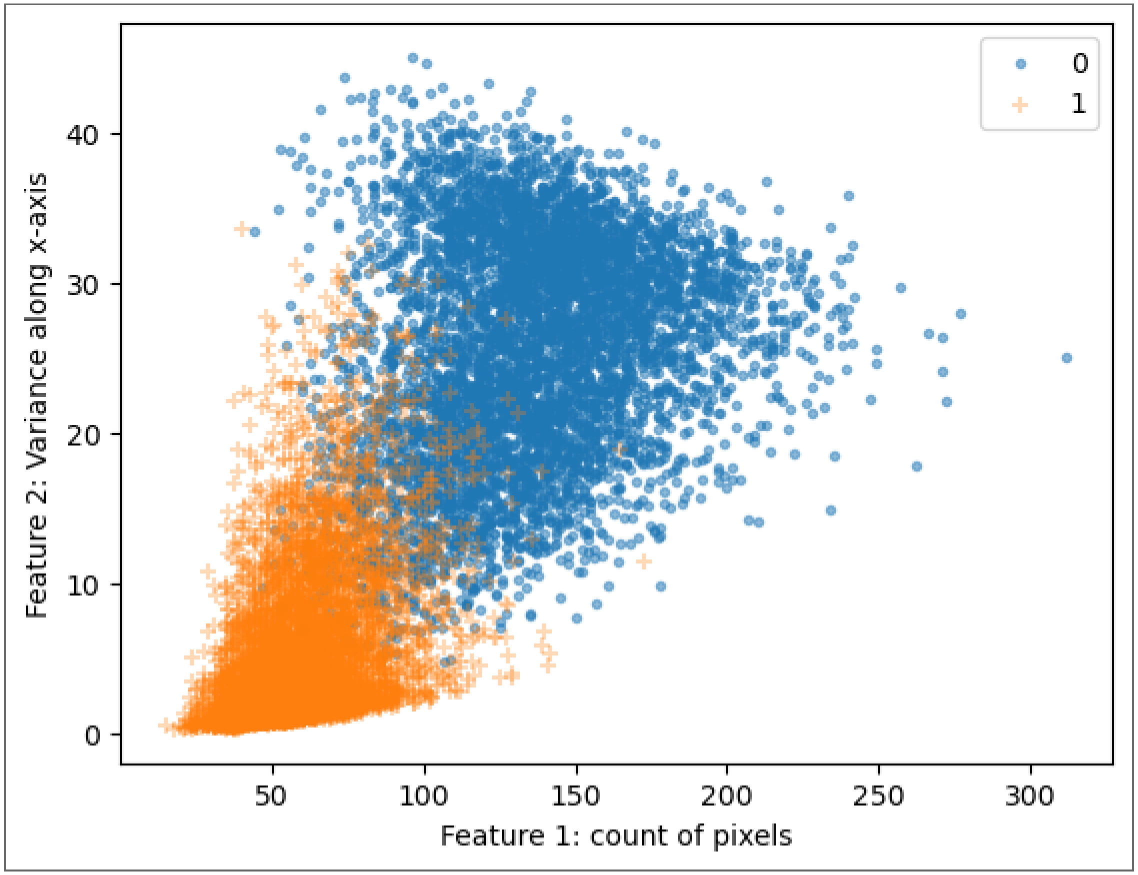
Out[22]:

```
[<matplotlib.lines.Line2D at 0x7f838d58a1a0>]
```



0
10
20
30
40
0
1000
2000
3000
4000
5000
6000
7000

In [24]:
```
fig, ax = plt.subplots()
draw_features(ax, zero_features, one_features)
plt.show()
```



In [27]:
```
def error(X, Y, bfm):
    # YOUR CODE HERE
    raise NotImplementedError()

def grad_error(Xw, Yw, bfm):
    # YOUR CODE HERE
```

```python
    raise NotImplementedError()

def train(X, Y, lr = 0.1):
    # YOUR CODE HERE
    raise NotImplementedError()

OPTIMAL_BFM, list_of_bfms, list_of_errors = train(X, Y)
fig, ax = plt.subplots()
ax.plot(list_of_errors)
ax.set_xlabel('t')
ax.set_ylabel('loss')
plt.show()
```

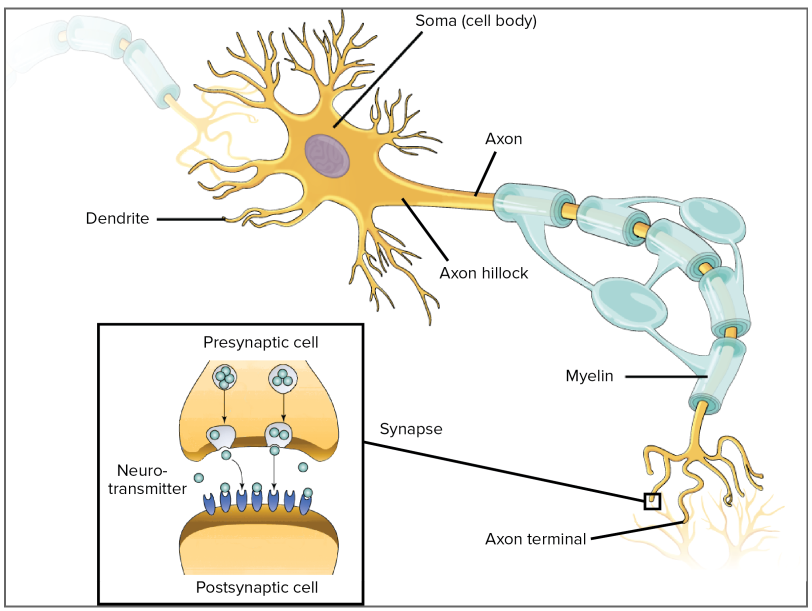
```
---------------------------------------------------------------------------
NotImplementedError                       Traceback (most recent call last)
Cell In[27], line 13
      9 def train(X, Y, lr = 0.1):
     10     # YOUR CODE HERE
     11     raise NotImplementedError()
---> 13 OPTIMAL_BFM, list_of_bfms, list_of_errors = train(X, Y)
     14 fig, ax = plt.subplots()
     15 ax.plot(list_of_errors)

Cell In[27], line 11, in train(X, Y, lr)
      9 def train(X, Y, lr = 0.1):
     10     # YOUR CODE HERE
---> 11     raise NotImplementedError()

NotImplementedError:
```

Single Layer Neural Networks

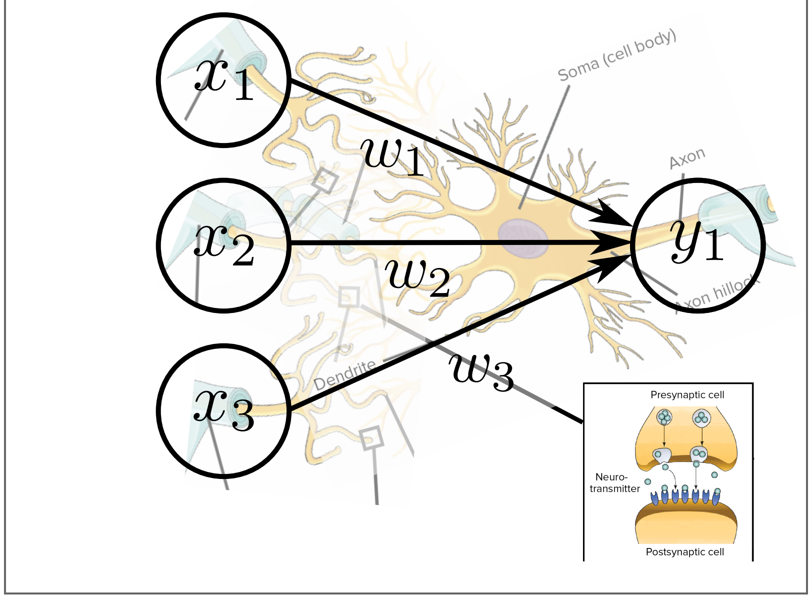
<u>Read Chapter 2 and 3 of UDL Book</u>

Notes Single Layer Neural Networks are simplest kind of neural networks. But before we dive into single layer neural networks, may be we should focus on the name *neural* networks. The name neural networks comes from biological neurons.

Similarities between Artificial neuron and Biological neuron



$$y_1 = \sigma(w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4)$$



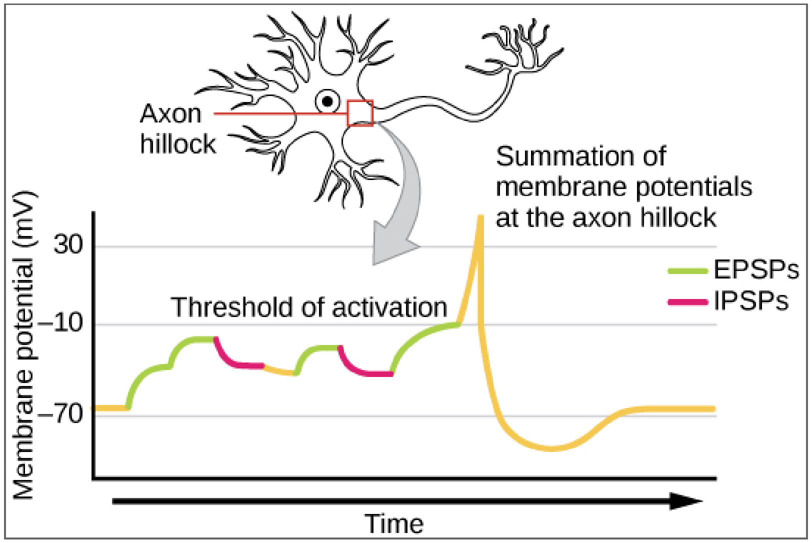
1. The excitation or firing of a biological neuron can be equated to a high positive value of units (x_1, x_2, x_3) in artificial neurons.
2. The synapse in biological neuron determines which input excitations will have excitatory or inhibitary impact on output excitations. Synapses can strengthen, weaken, disconnect or form new connections during biological learning. Similarly to excitatory synapes, positive weights can cause positive input values to contribute to positive output values.
3. Usually multiple excitatory inputs are required excite the output neuron.

References:

1. **https://openstax.org/books/biology/pages/35-2-how-neurons-communicate**

Differences

1. Biological neuron is all or None
2. Biological neuron has a time component



notes

1. The activity of the biological neuron is an "all-or-none" process. Articial activations are typically continuous range. Even when sigmoid or softmax nonlinearities.
2. Biological neuron has time dynamics. The input activations are integrated over time.