

Module 5.4 : Momentum based Gradient Descent

Some observations about gradient descent

- It takes a lot of time to navigate regions having a gentle slope

Some observations about gradient descent

- It takes a lot of time to navigate regions having a gentle slope
- This is because the gradient in these regions is very small

Some observations about gradient descent

- It takes a lot of time to navigate regions having a gentle slope
- This is because the gradient in these regions is very small
- Can we do something better ?

Some observations about gradient descent

- It takes a lot of time to navigate regions having a gentle slope
- This is because the gradient in these regions is very small
- Can we do something better ?
- Yes, let's take a look at 'Momentum based gradient descent'

Intuition

- If I am repeatedly being asked to move in the same direction then I should probably gain some confidence and start taking bigger steps in that direction

Intuition

- If I am repeatedly being asked to move in the same direction then I should probably gain some confidence and start taking bigger steps in that direction
- Just as a ball gains momentum while rolling down a slope

Intuition

- If I am repeatedly being asked to move in the same direction then I should probably gain some confidence and start taking bigger steps in that direction
- Just as a ball gains momentum while rolling down a slope

Update rule for momentum based gradient descent

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

Intuition

- If I am repeatedly being asked to move in the same direction then I should probably gain some confidence and start taking bigger steps in that direction
- Just as a ball gains momentum while rolling down a slope

Update rule for momentum based gradient descent

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

- In addition to the current update, also look at the history of updates.

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_2 = \gamma \cdot update_1 + \eta \nabla w_2 = \gamma \cdot \eta \nabla w_1 + \eta \nabla w_2$$

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_2 = \gamma \cdot update_1 + \eta \nabla w_2 = \gamma \cdot \eta \nabla w_1 + \eta \nabla w_2$$

$$update_3 = \gamma \cdot update_2 + \eta \nabla w_3 = \gamma(\gamma \cdot \eta \nabla w_1 + \eta \nabla w_2) + \eta \nabla w_3$$

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_2 = \gamma \cdot update_1 + \eta \nabla w_2 = \gamma \cdot \eta \nabla w_1 + \eta \nabla w_2$$

$$\begin{aligned} update_3 &= \gamma \cdot update_2 + \eta \nabla w_3 = \gamma(\gamma \cdot \eta \nabla w_1 + \eta \nabla w_2) + \eta \nabla w_3 \\ &= \gamma \cdot update_2 + \eta \nabla w_3 = \gamma^2 \cdot \eta \nabla w_1 + \gamma \cdot \eta \nabla w_2 + \eta \nabla w_3 \end{aligned}$$

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_2 = \gamma \cdot update_1 + \eta \nabla w_2 = \gamma \cdot \eta \nabla w_1 + \eta \nabla w_2$$

$$update_3 = \gamma \cdot update_2 + \eta \nabla w_3 = \gamma(\gamma \cdot \eta \nabla w_1 + \eta \nabla w_2) + \eta \nabla w_3$$

$$= \gamma \cdot update_2 + \eta \nabla w_3 = \gamma^2 \cdot \eta \nabla w_1 + \gamma \cdot \eta \nabla w_2 + \eta \nabla w_3$$

$$update_4 = \gamma \cdot update_3 + \eta \nabla w_4 = \gamma^3 \cdot \eta \nabla w_1 + \gamma^2 \cdot \eta \nabla w_2 + \gamma \cdot \eta \nabla w_3 + \eta \nabla w_4$$

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_2 = \gamma \cdot update_1 + \eta \nabla w_2 = \gamma \cdot \eta \nabla w_1 + \eta \nabla w_2$$

$$update_3 = \gamma \cdot update_2 + \eta \nabla w_3 = \gamma(\gamma \cdot \eta \nabla w_1 + \eta \nabla w_2) + \eta \nabla w_3$$

$$= \gamma \cdot update_2 + \eta \nabla w_3 = \gamma^2 \cdot \eta \nabla w_1 + \gamma \cdot \eta \nabla w_2 + \eta \nabla w_3$$

$$update_4 = \gamma \cdot update_3 + \eta \nabla w_4 = \gamma^3 \cdot \eta \nabla w_1 + \gamma^2 \cdot \eta \nabla w_2 + \gamma \cdot \eta \nabla w_3 + \eta \nabla w_4$$

⋮

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$

$$w_{t+1} = w_t - update_t$$

$$update_0 = 0$$

$$update_1 = \gamma \cdot update_0 + \eta \nabla w_1 = \eta \nabla w_1$$

$$update_2 = \gamma \cdot update_1 + \eta \nabla w_2 = \gamma \cdot \eta \nabla w_1 + \eta \nabla w_2$$

$$update_3 = \gamma \cdot update_2 + \eta \nabla w_3 = \gamma(\gamma \cdot \eta \nabla w_1 + \eta \nabla w_2) + \eta \nabla w_3$$

$$= \gamma \cdot update_2 + \eta \nabla w_3 = \gamma^2 \cdot \eta \nabla w_1 + \gamma \cdot \eta \nabla w_2 + \eta \nabla w_3$$

$$update_4 = \gamma \cdot update_3 + \eta \nabla w_4 = \gamma^3 \cdot \eta \nabla w_1 + \gamma^2 \cdot \eta \nabla w_2 + \gamma \cdot \eta \nabla w_3 + \eta \nabla w_4$$

⋮

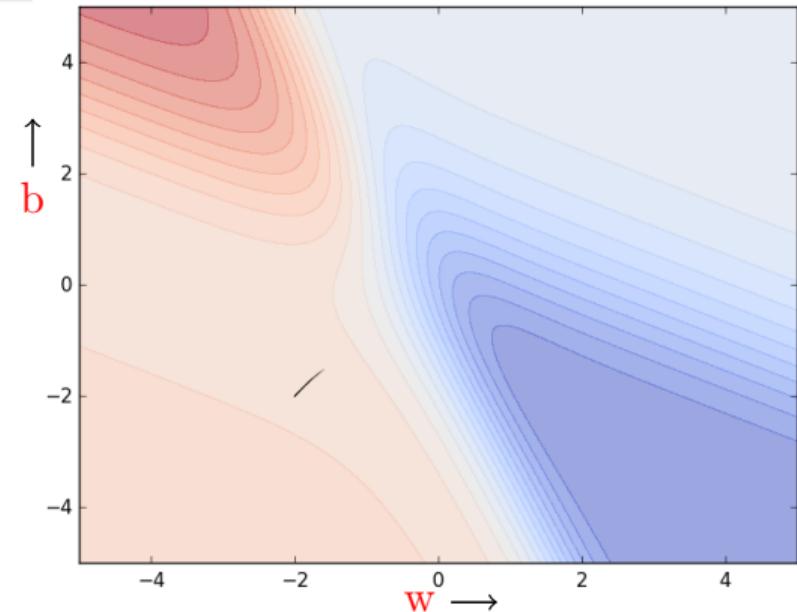
$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t = \gamma^{t-1} \cdot \eta \nabla w_1 + \gamma^{t-2} \cdot \eta \nabla w_1 + \dots + \eta \nabla w_t$$

```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

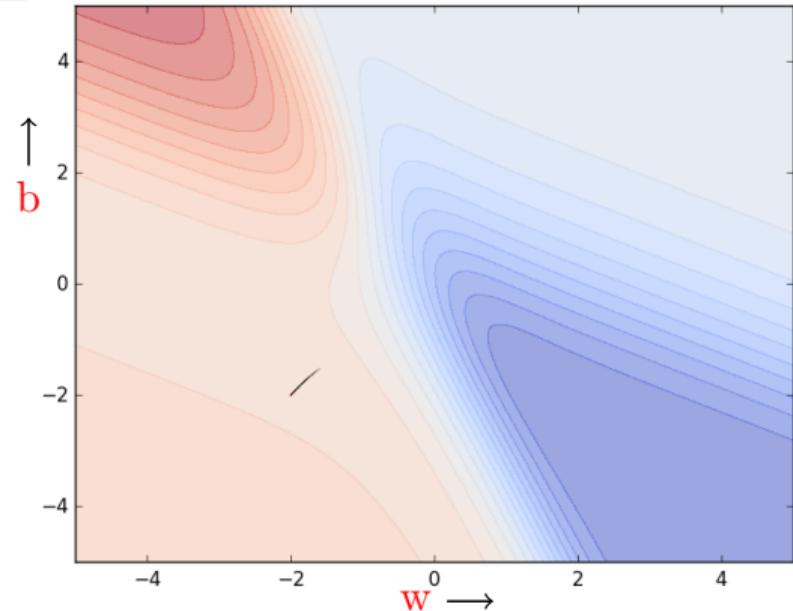


```
def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db

        w = w - v_w
        b = b - v_b

        prev_v_w = v_w
        prev_v_b = v_b
```

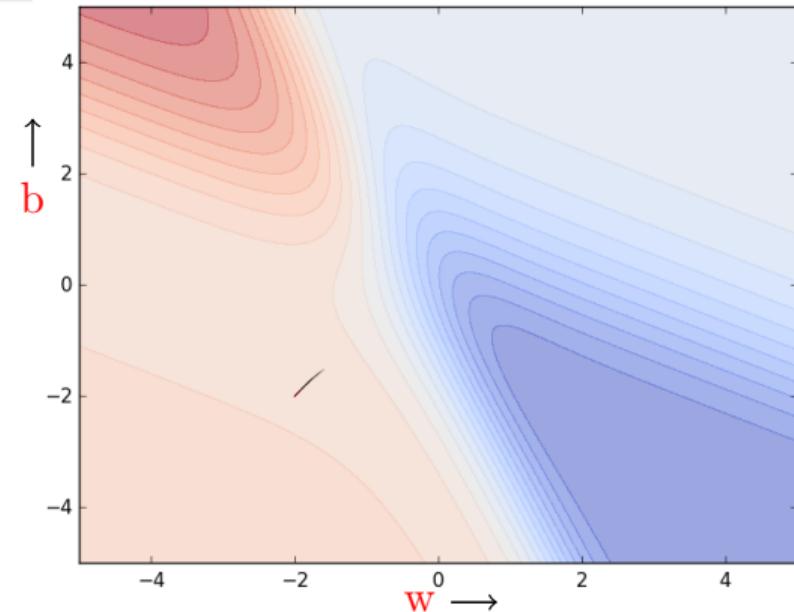


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

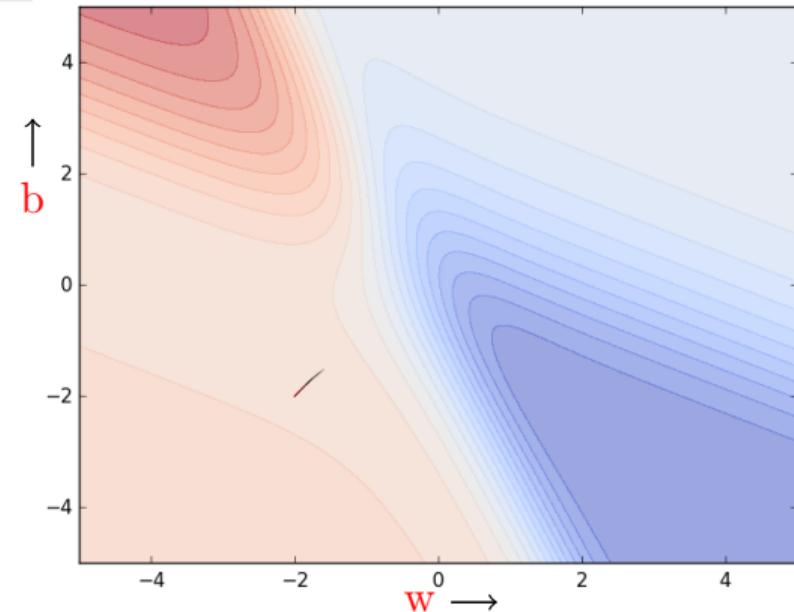


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

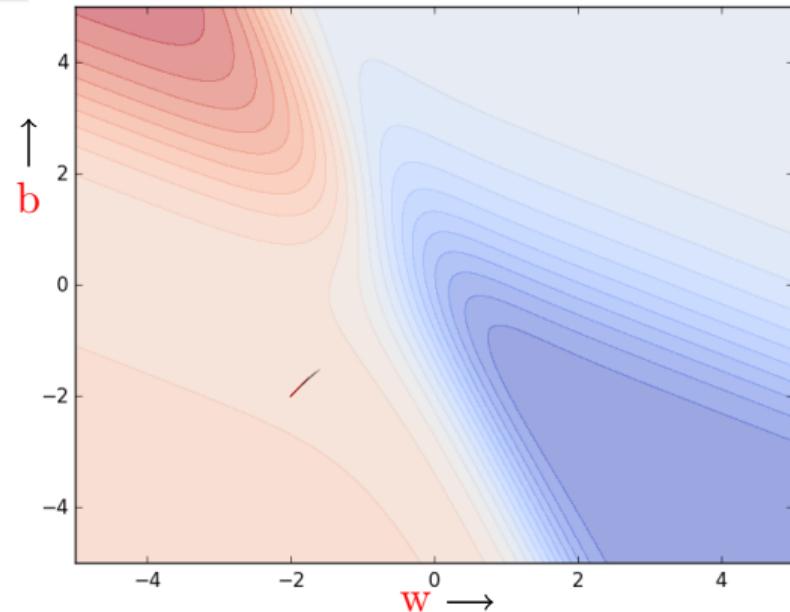


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

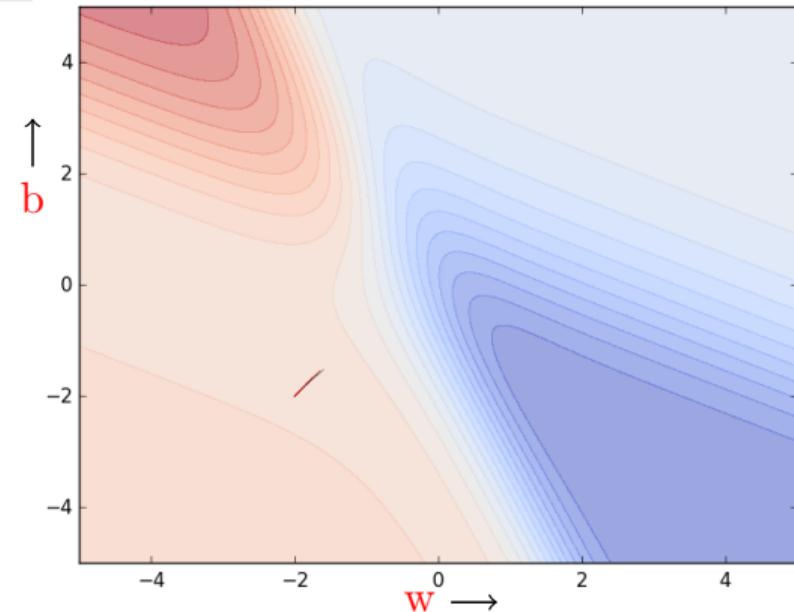


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

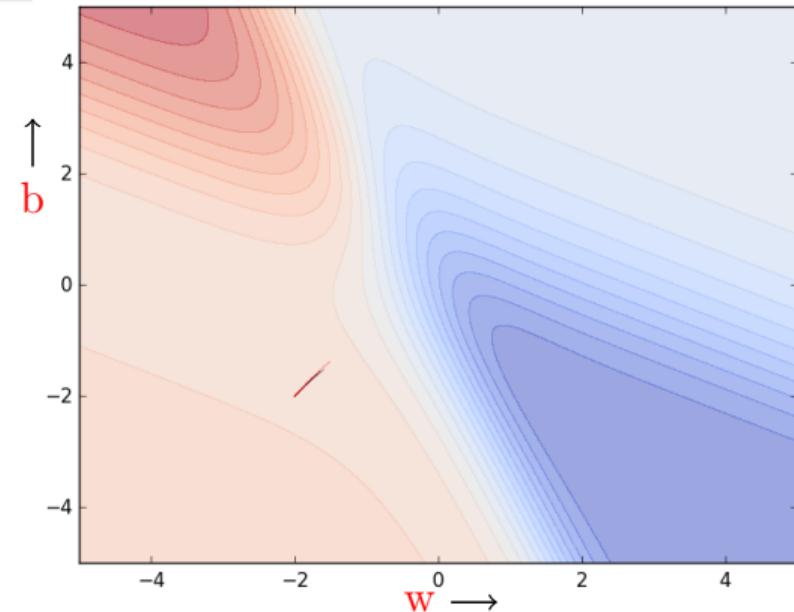


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

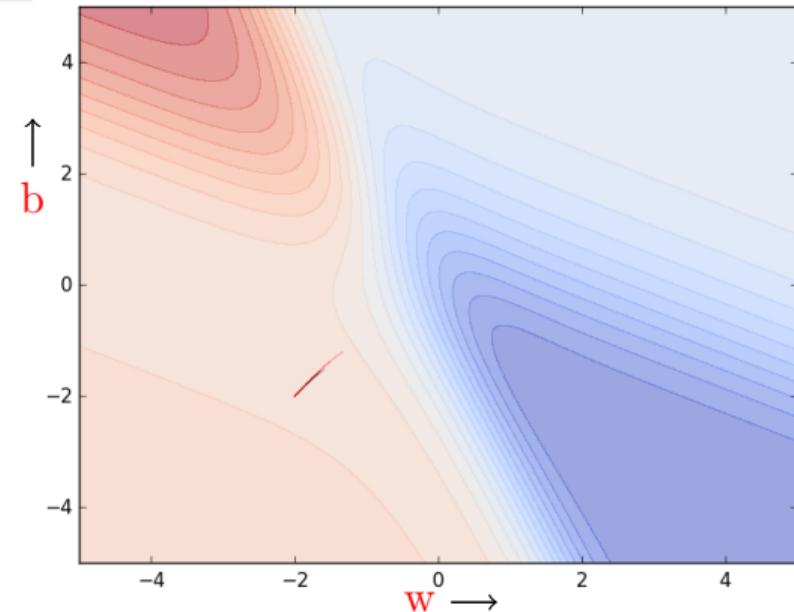


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

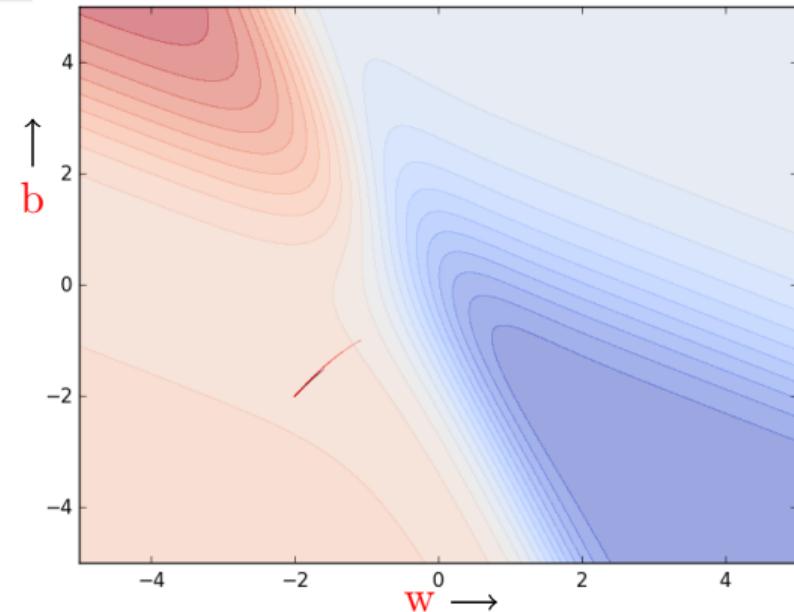


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

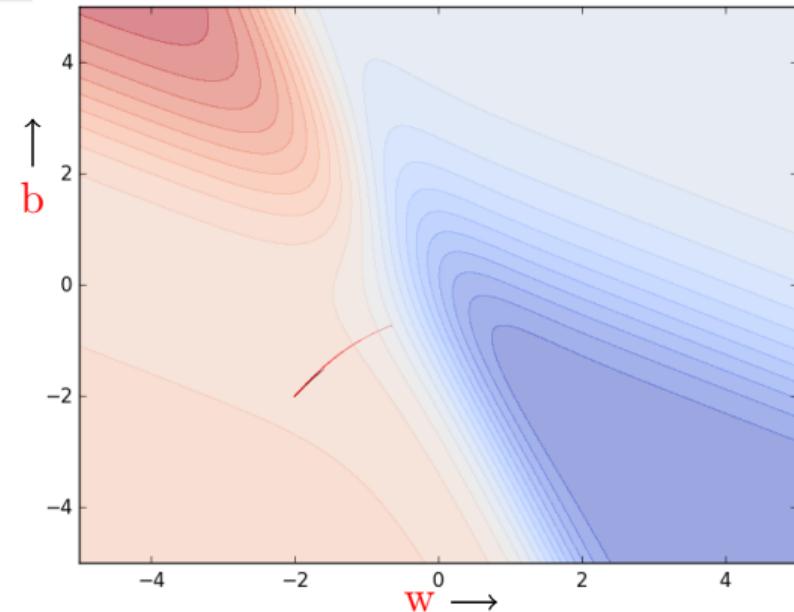


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

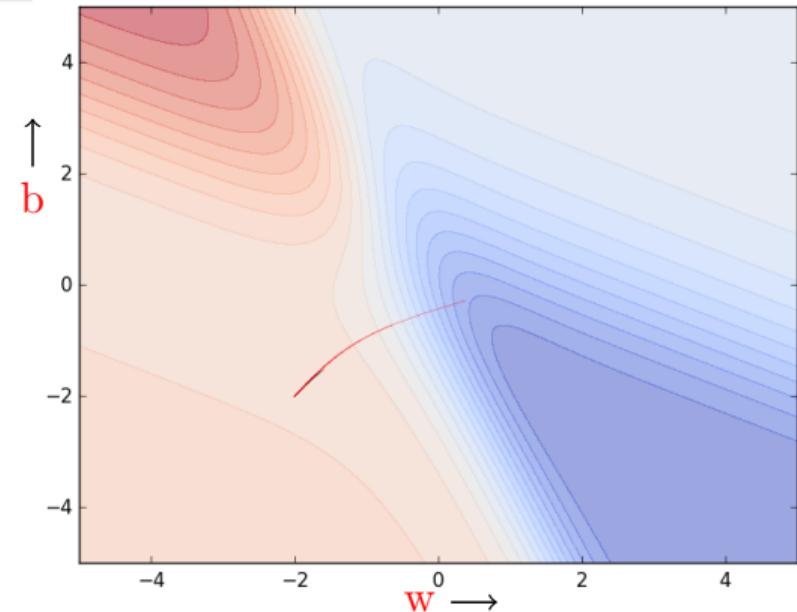


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

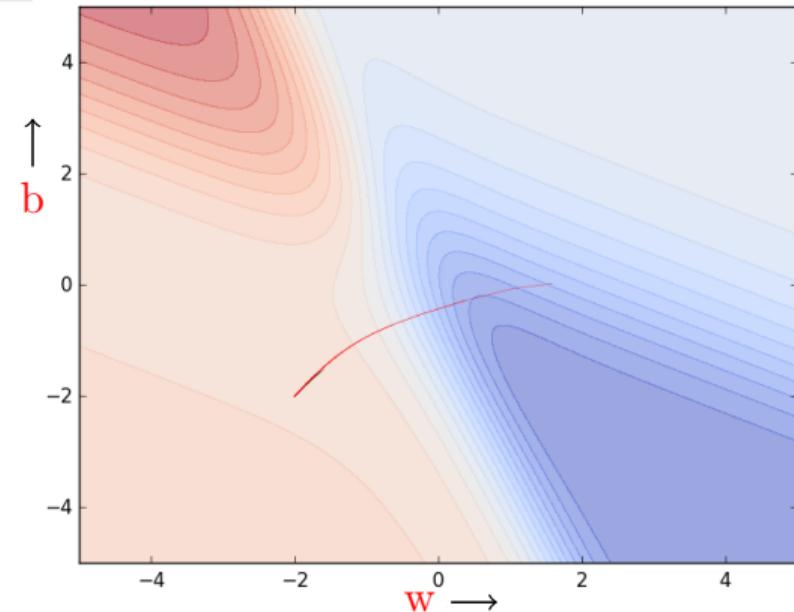


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

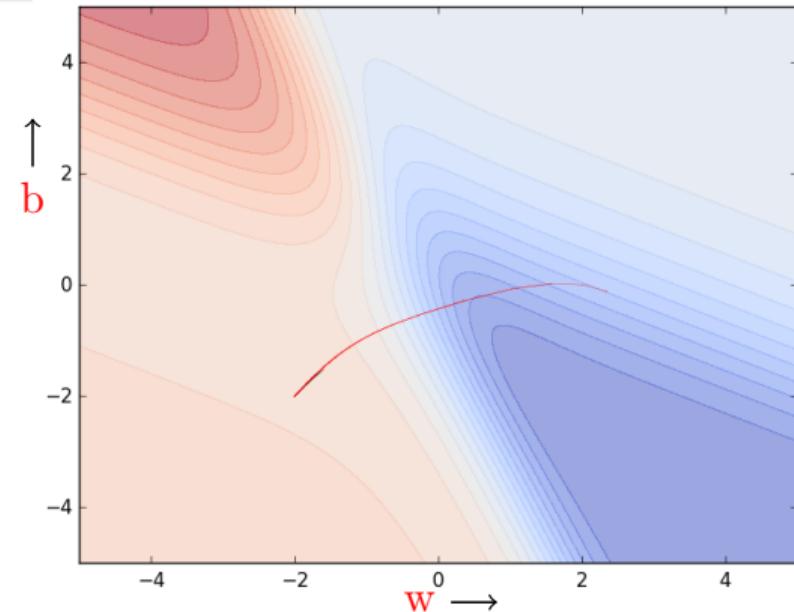


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

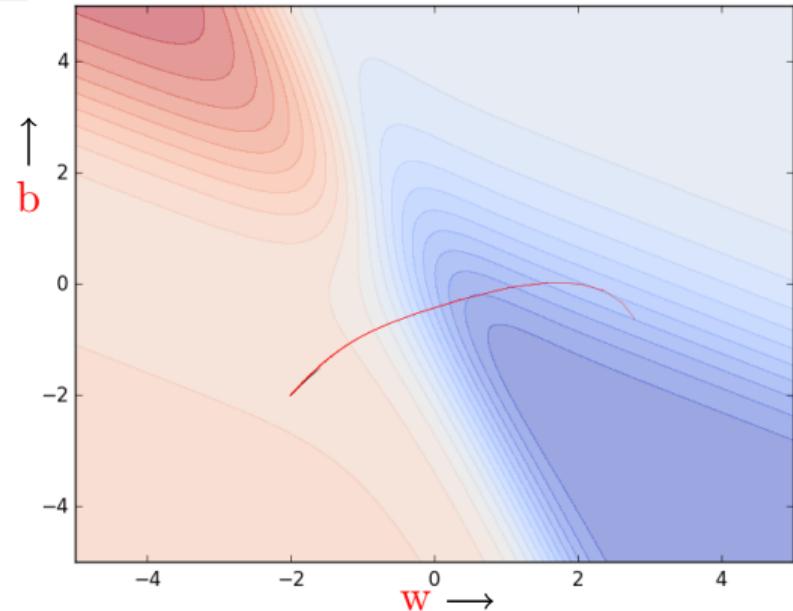


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

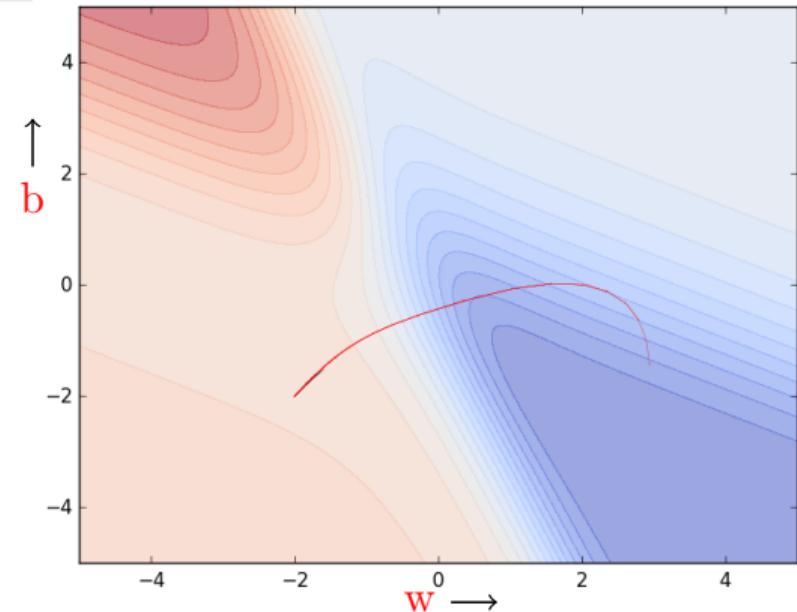


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

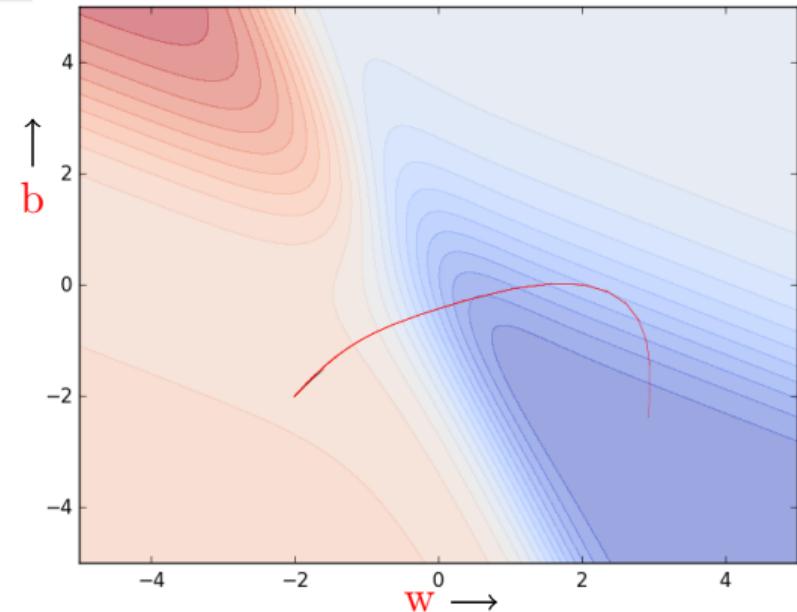


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

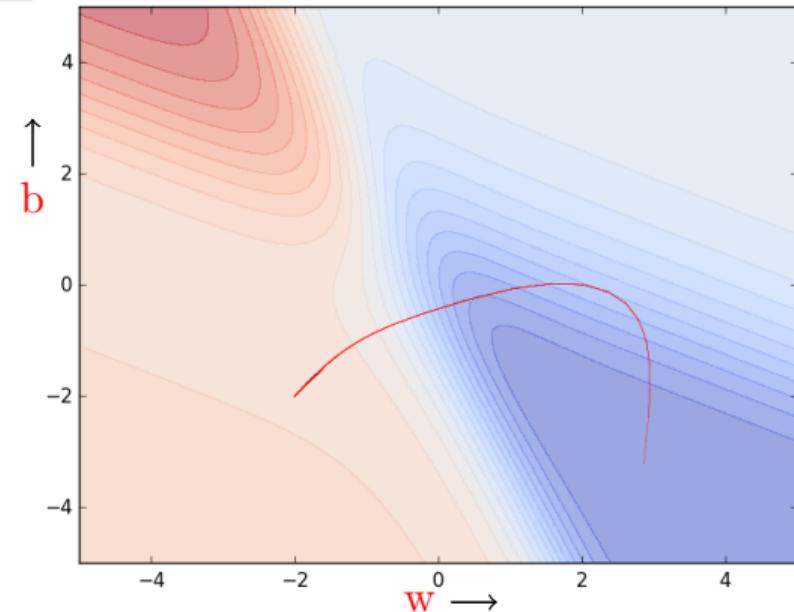


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

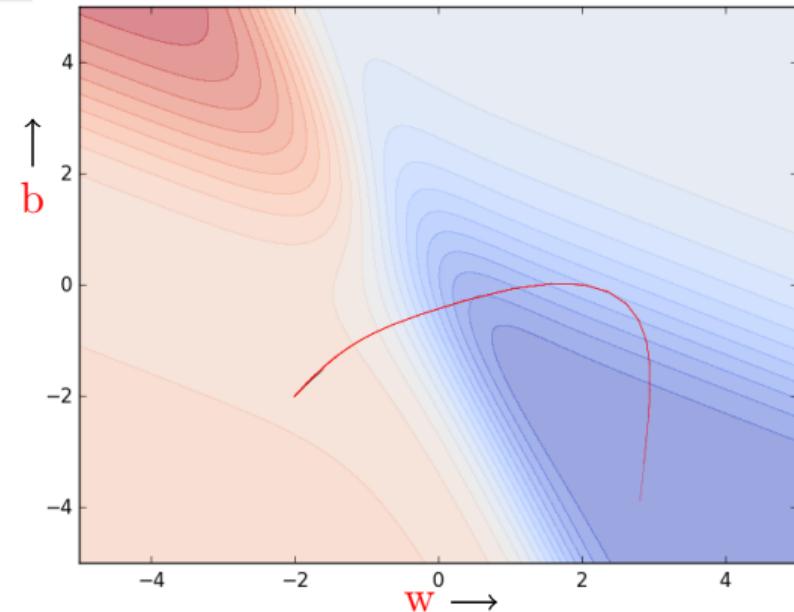


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

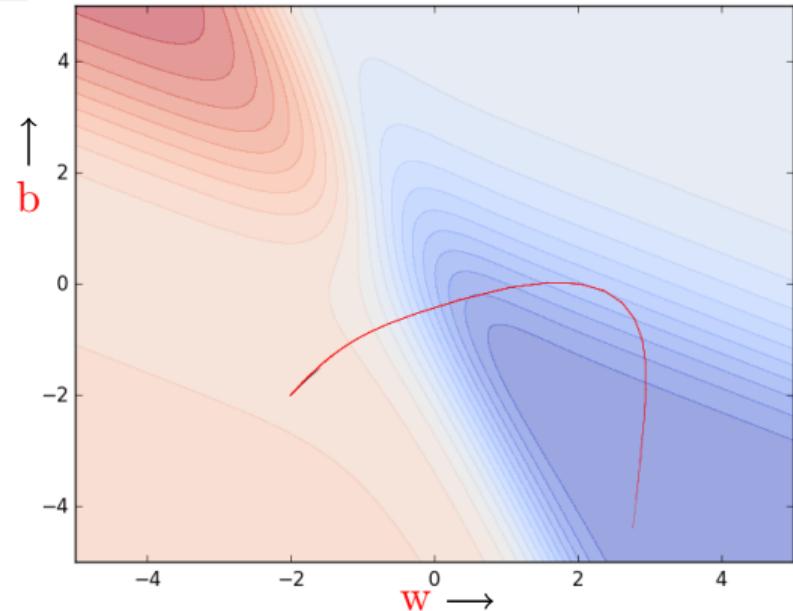


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```

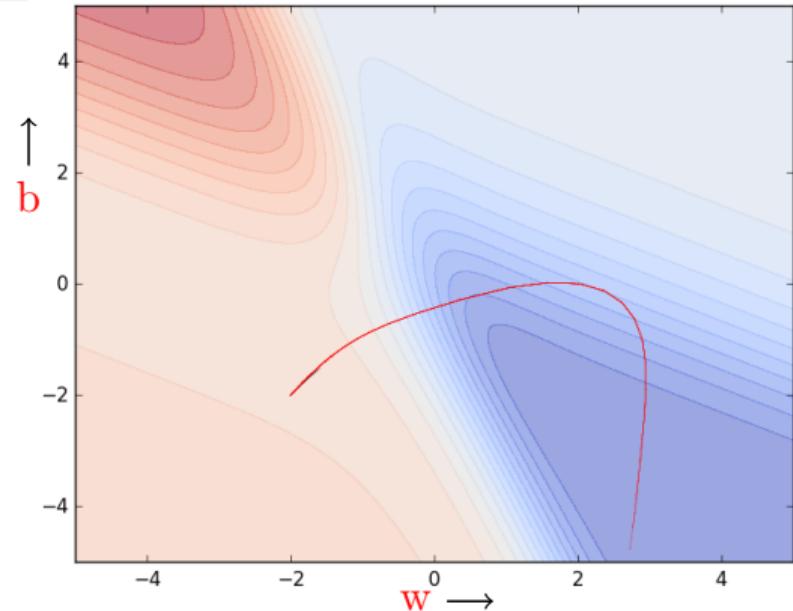


```

def do_momentum_gradient_descent() :
    w, b, eta = init_w, init_b, 1.0
    prev_v_w, prev_v_b, gamma = 0, 0, 0.9
    for i in range(max_epochs) :
        dw, db = 0, 0
        for x,y in zip(X, Y) :
            dw += grad_w(w, b, x, y)
            db += grad_b(w, b, x, y)

        v_w = gamma * prev_v_w + eta* dw
        v_b = gamma * prev_v_b + eta* db
        w = w - v_w
        b = b - v_b
        prev_v_w = v_w
        prev_v_b = v_b

```



Some observations and questions

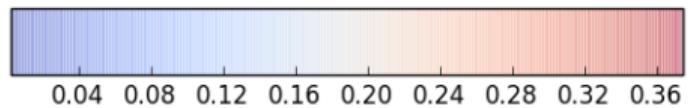
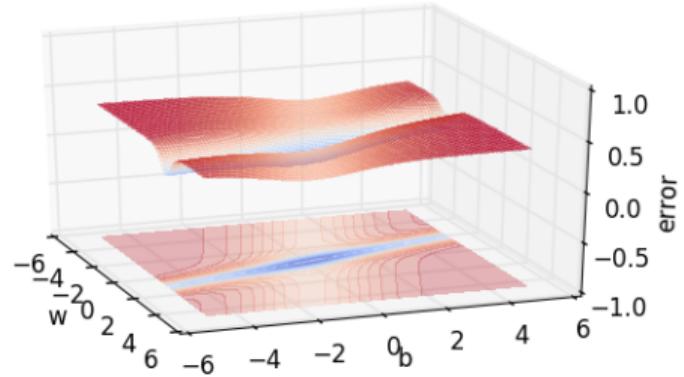
- Even in the regions having gentle slopes, momentum based gradient descent is able to take large steps because the momentum carries it along

Some observations and questions

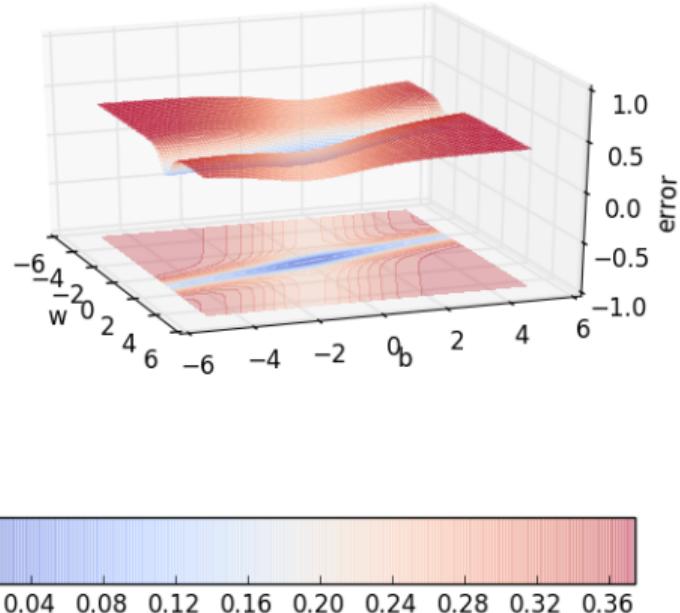
- Even in the regions having gentle slopes, momentum based gradient descent is able to take large steps because the momentum carries it along
- Is moving fast always good? Would there be a situation where momentum would cause us to run pass our goal?

Some observations and questions

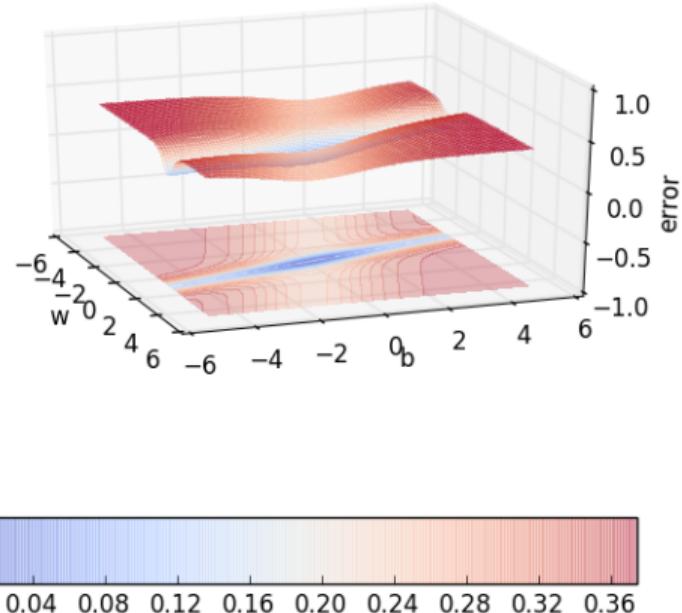
- Even in the regions having gentle slopes, momentum based gradient descent is able to take large steps because the momentum carries it along
- Is moving fast always good? Would there be a situation where momentum would cause us to run pass our goal?
- Let us change our input data so that we end up with a different error surface and then see what happens ...

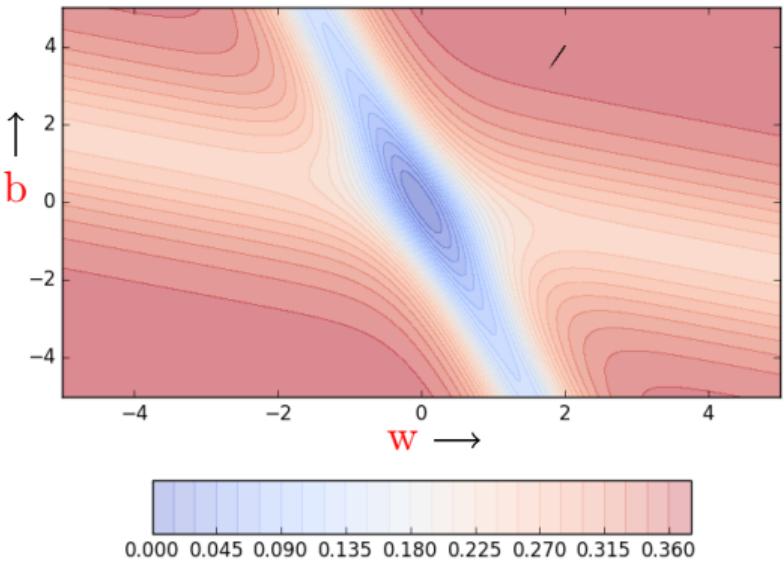


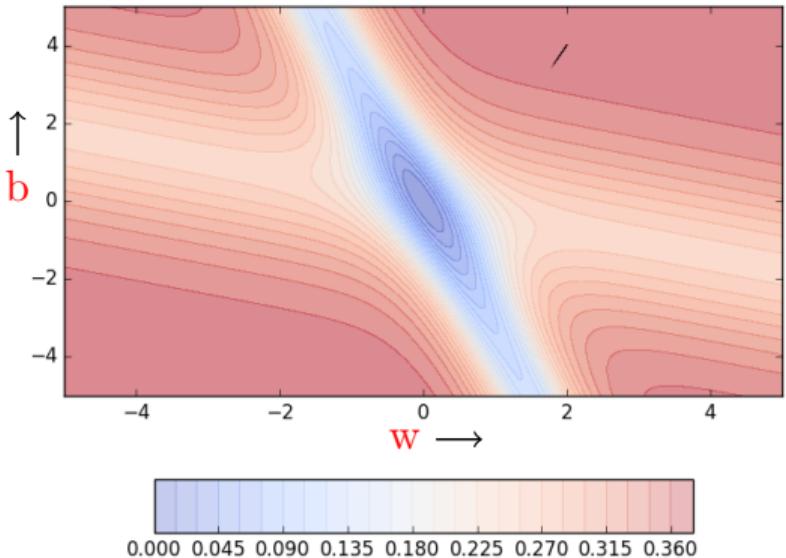
- In this case, the error is high on either side of the minima valley

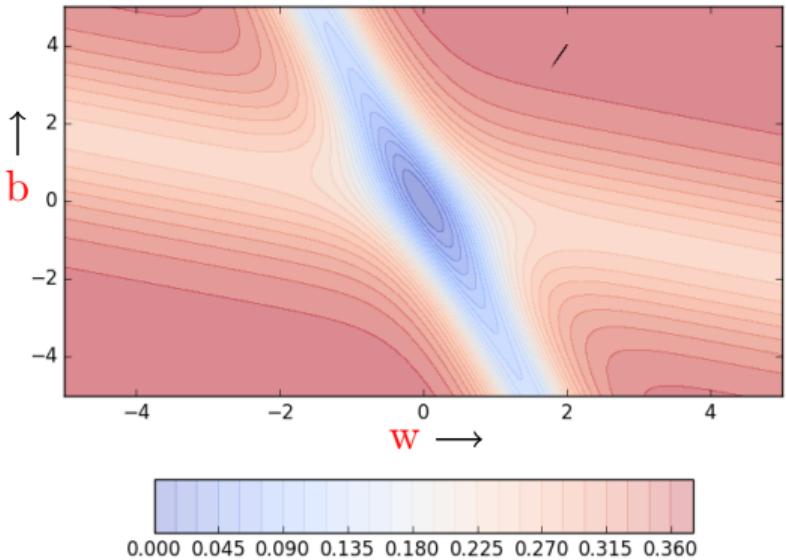


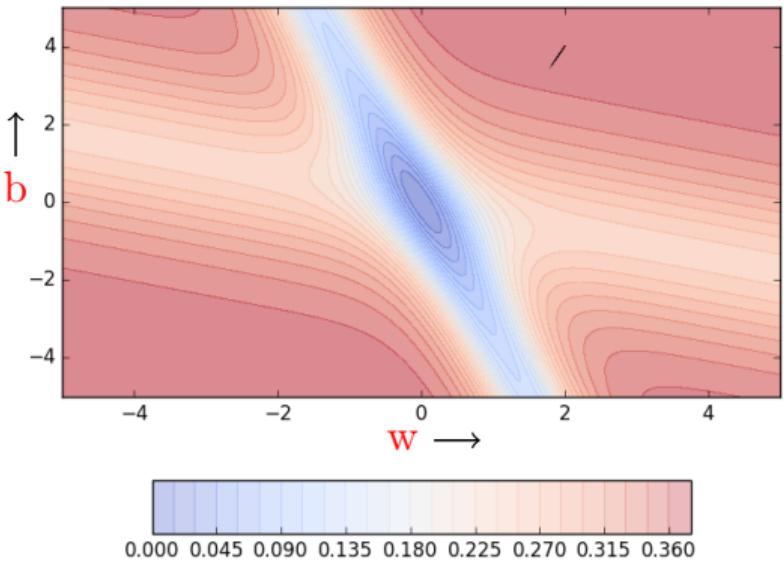
- In this case, the error is high on either side of the minima valley
- Could momentum be detrimental in such cases... let's see....

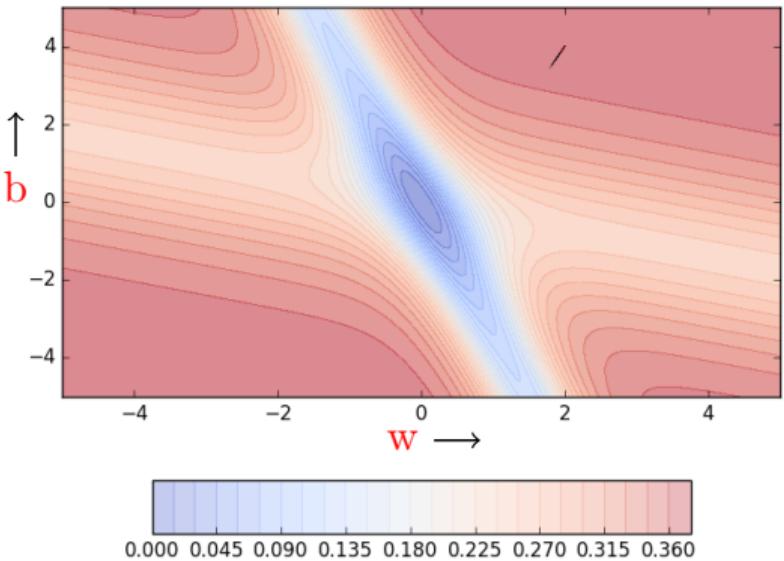


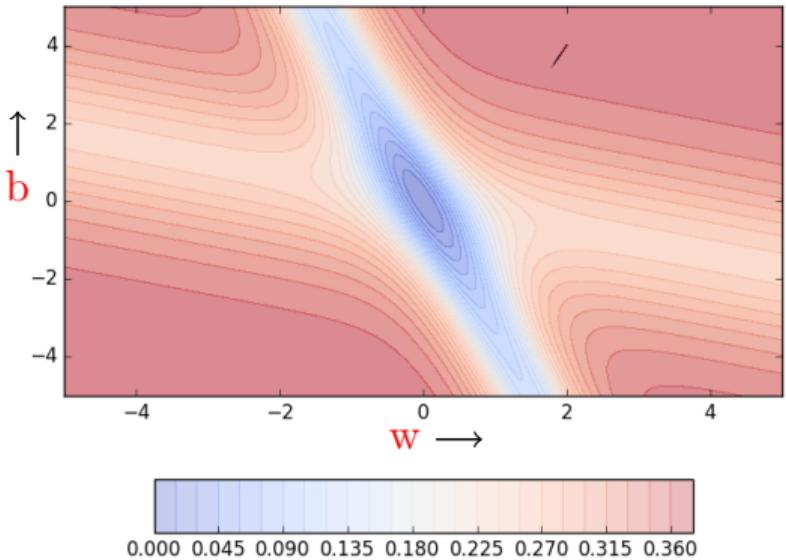


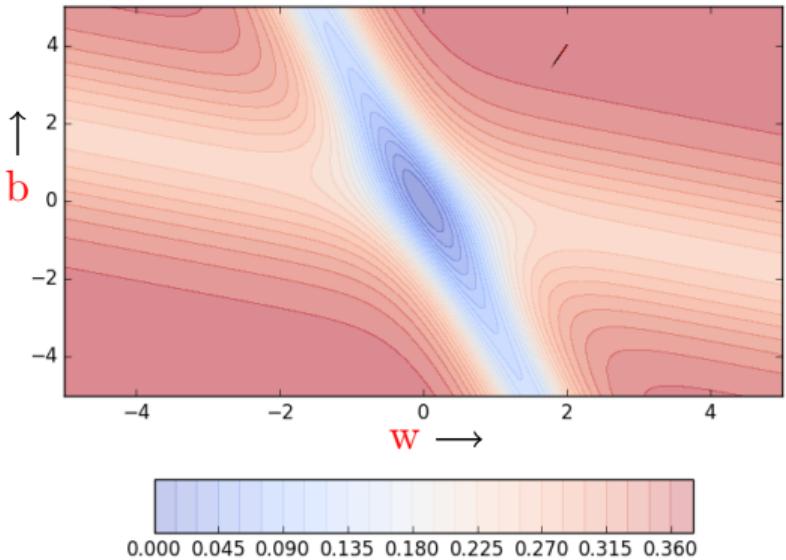


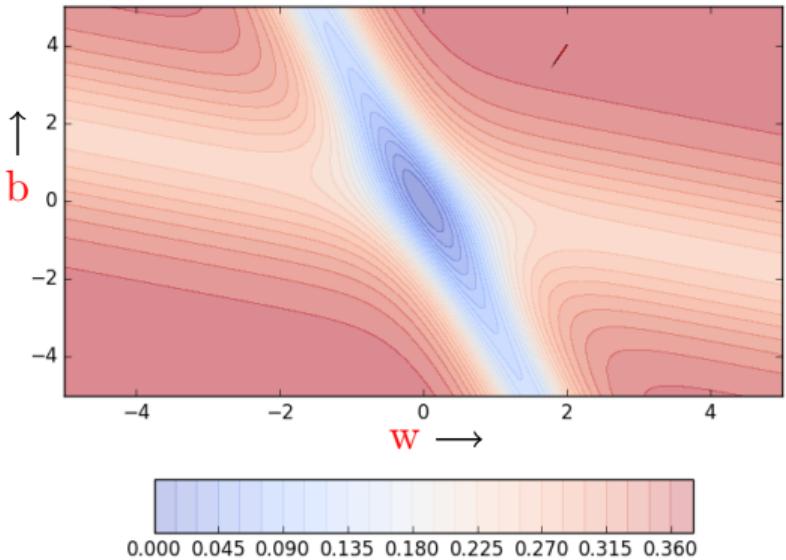


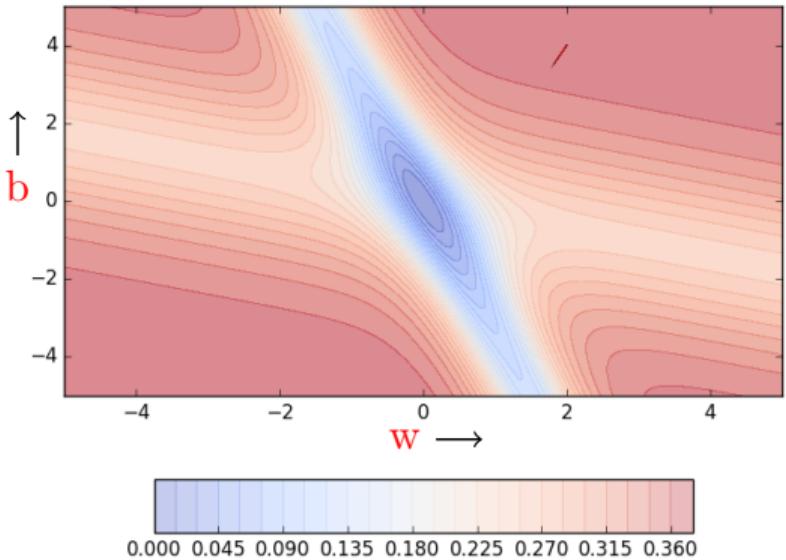


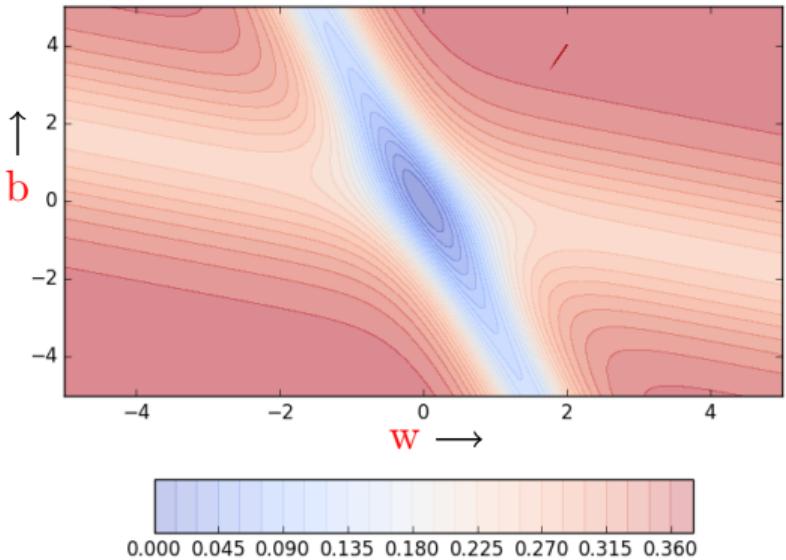


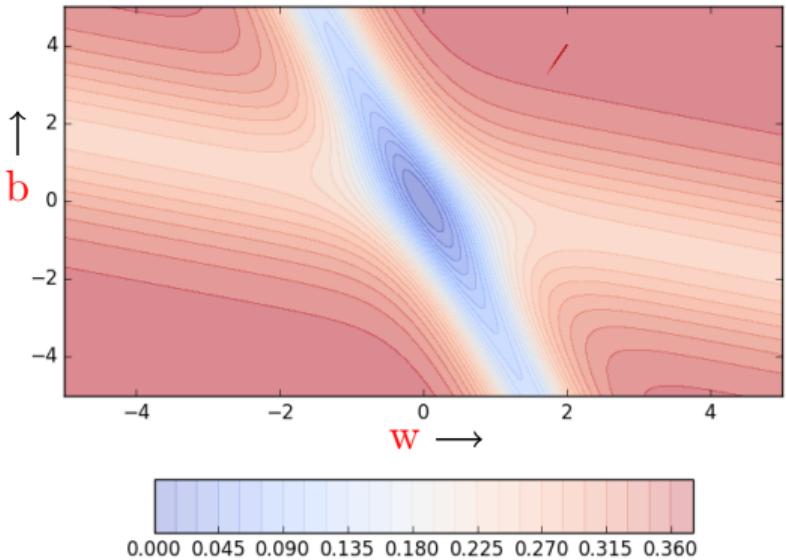


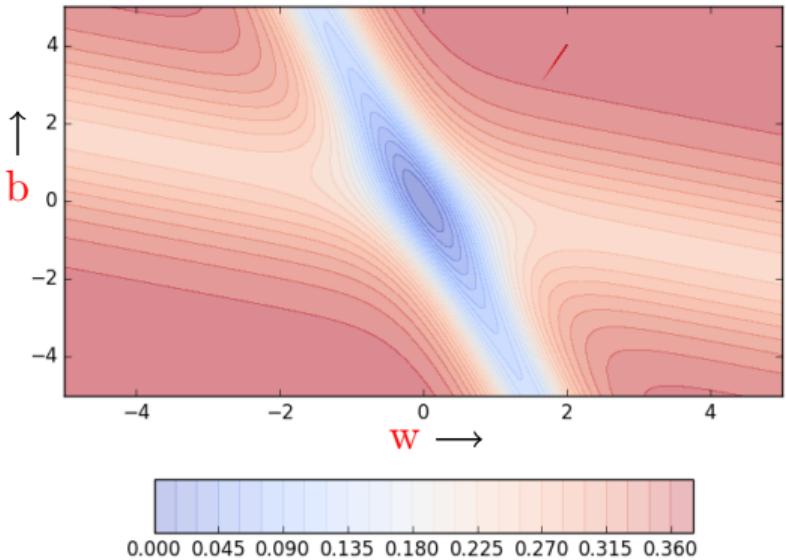


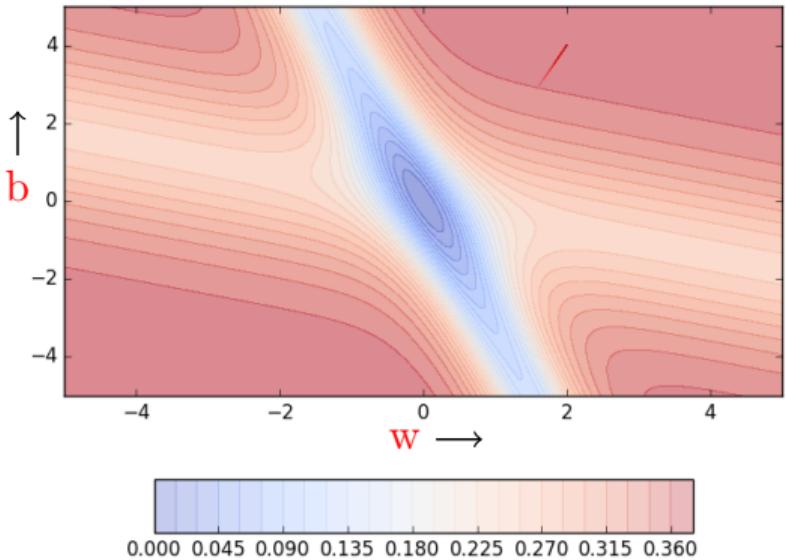


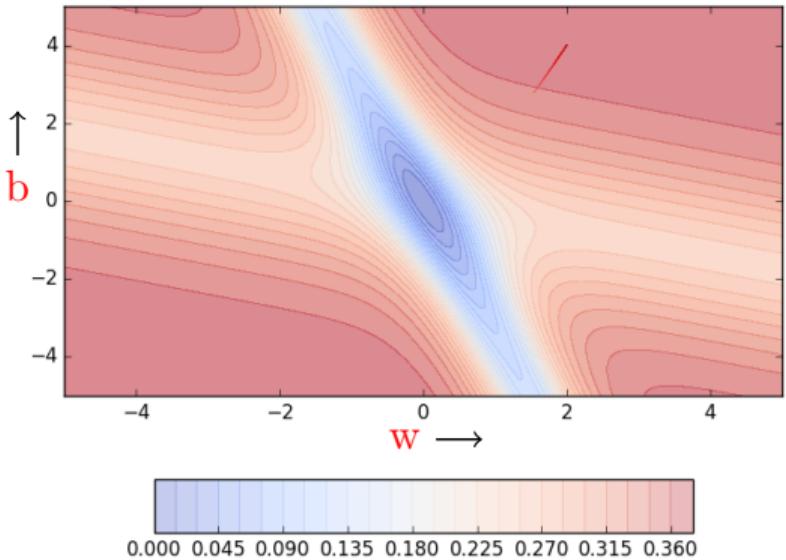


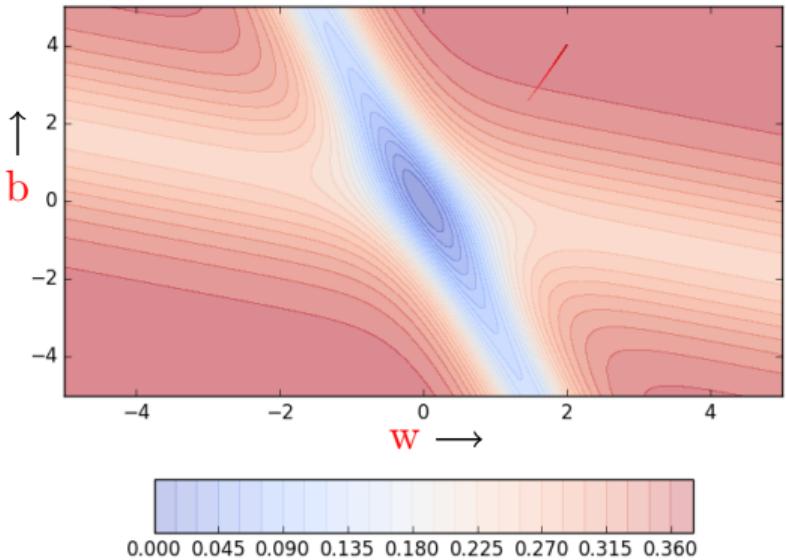


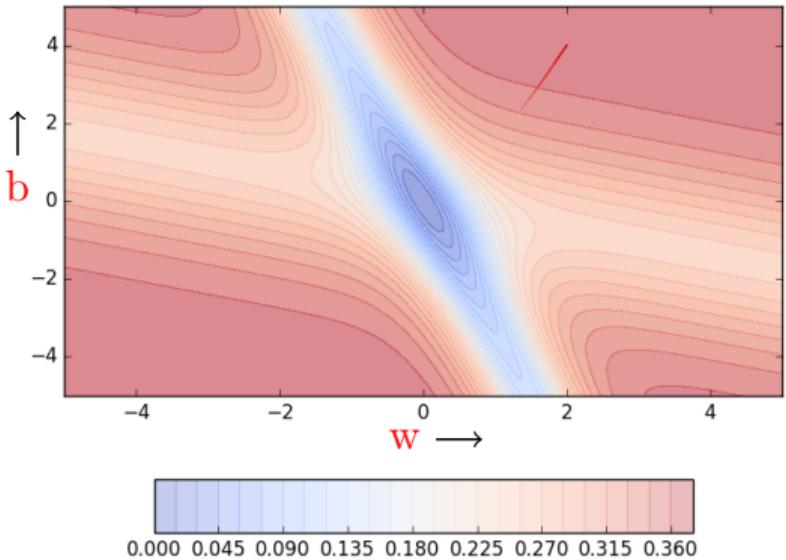


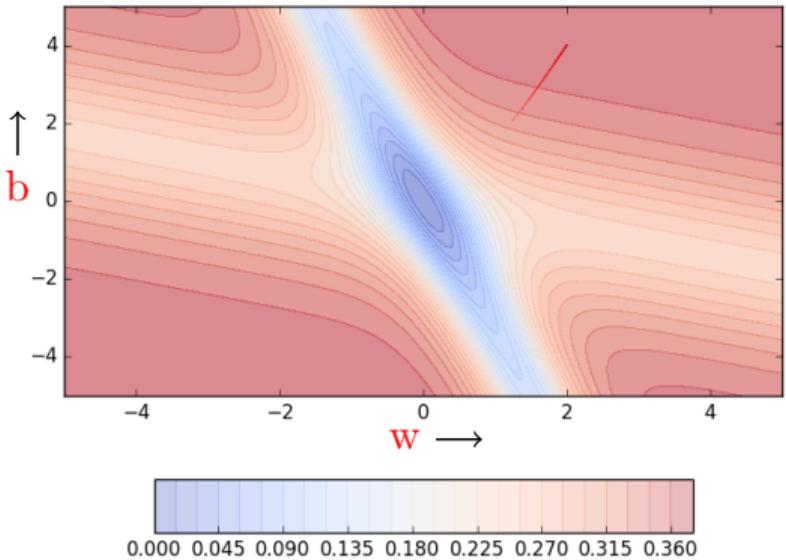


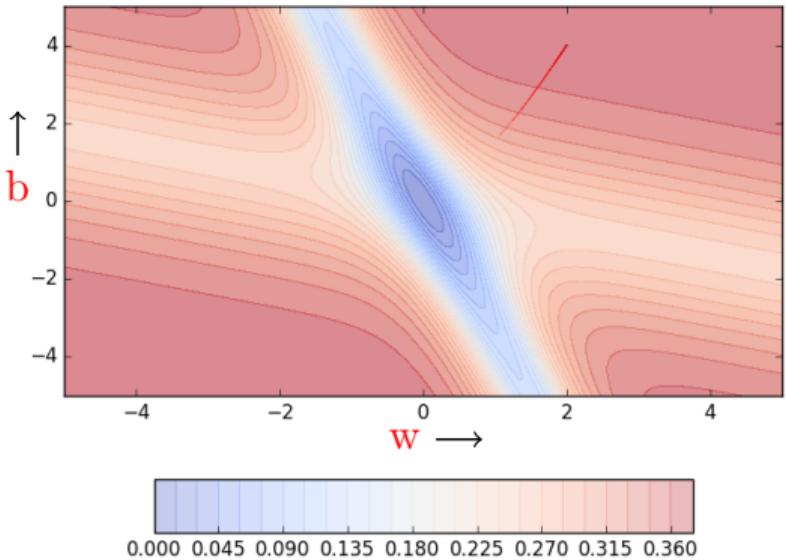


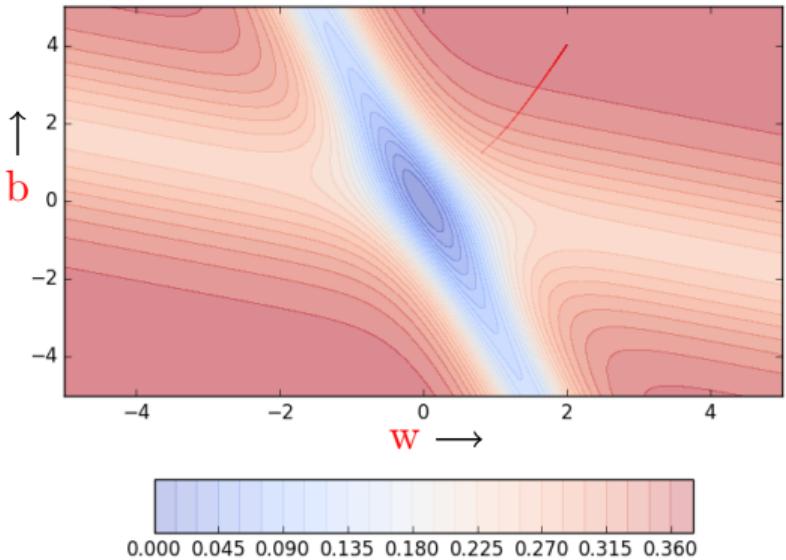


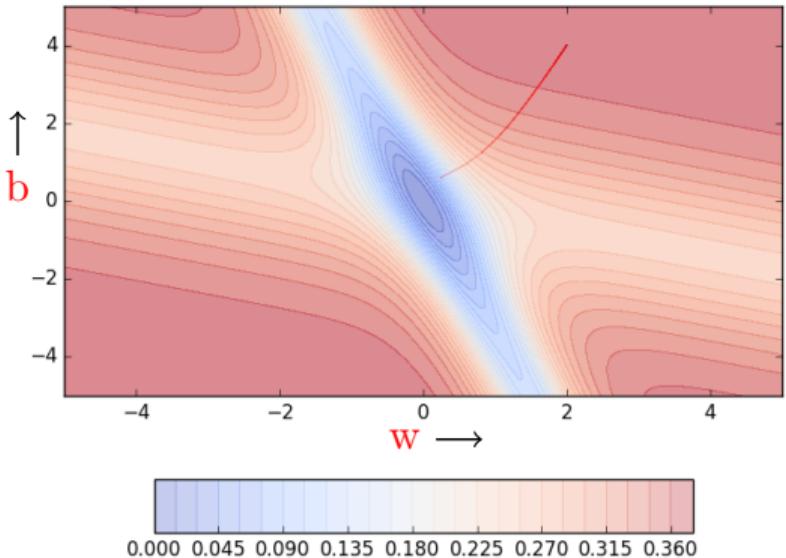


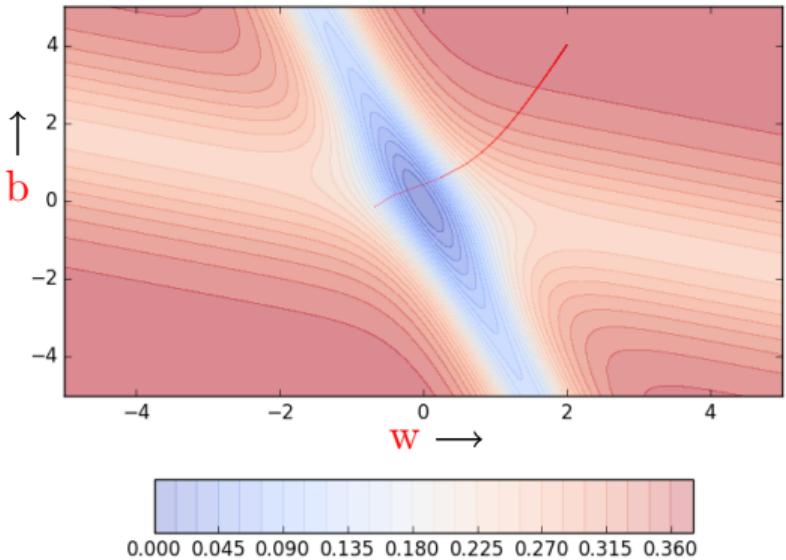


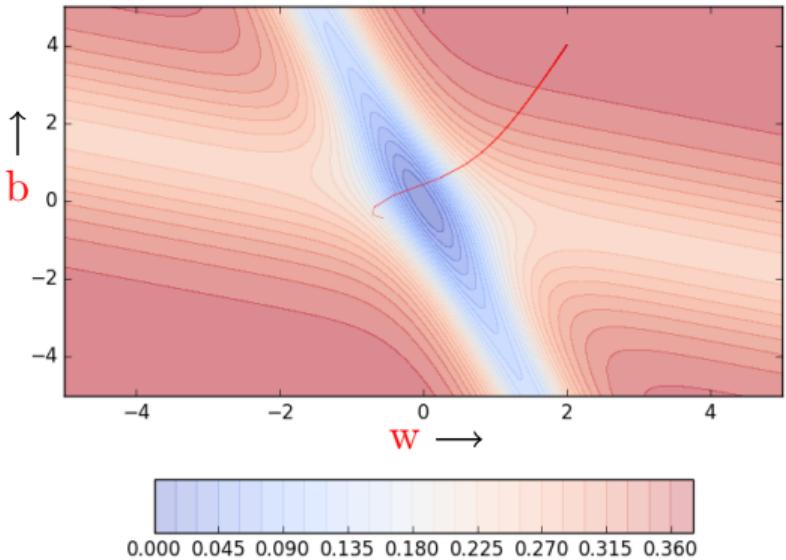


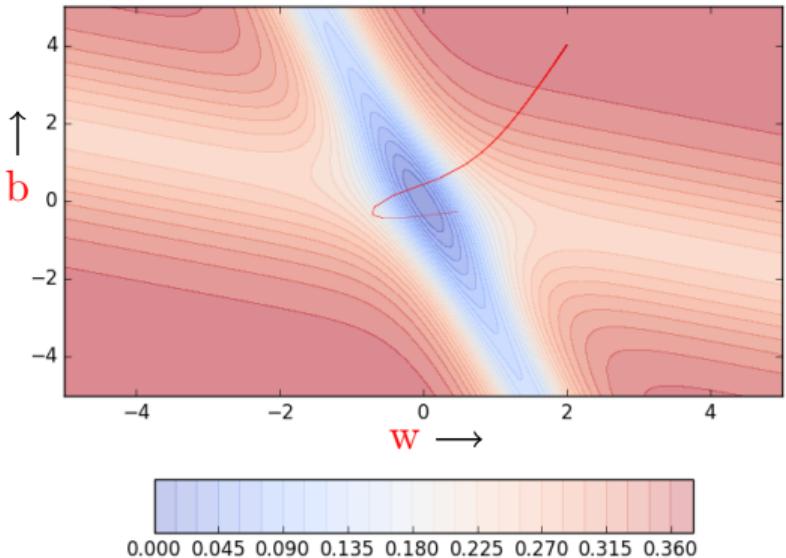


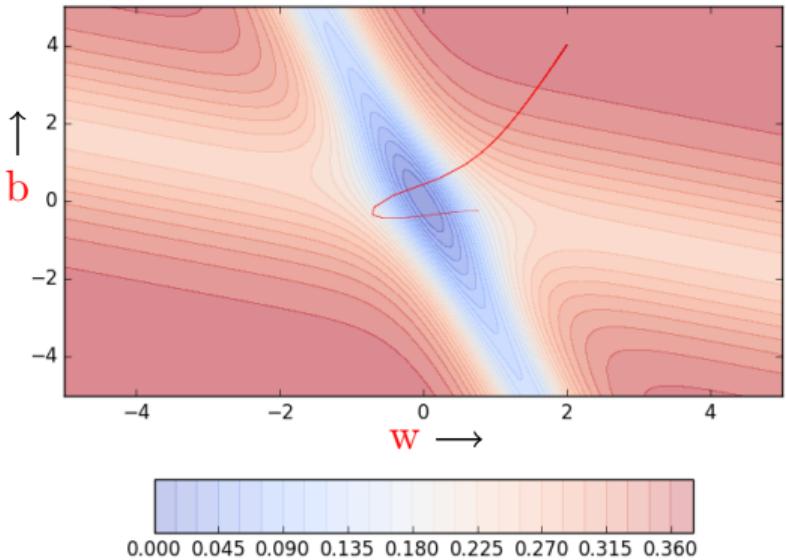


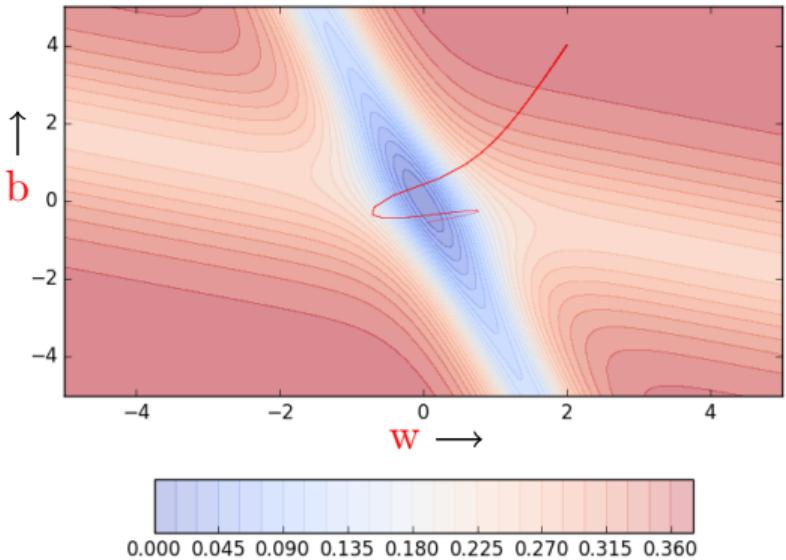


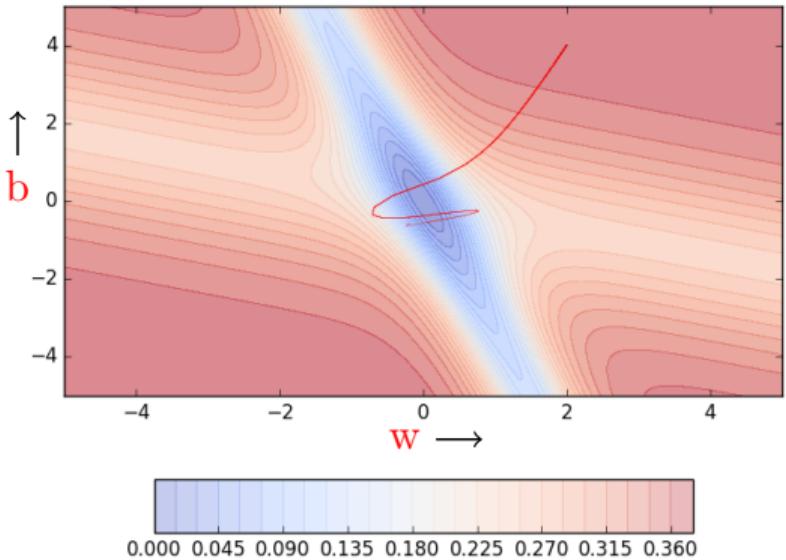


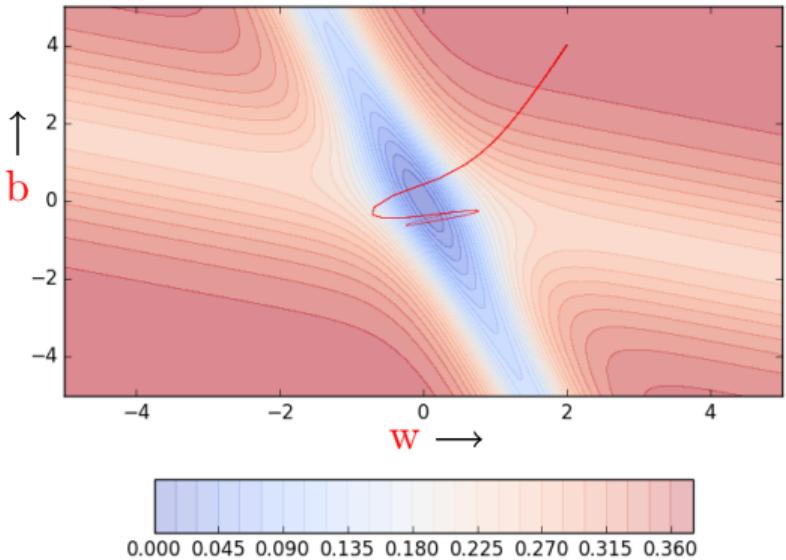


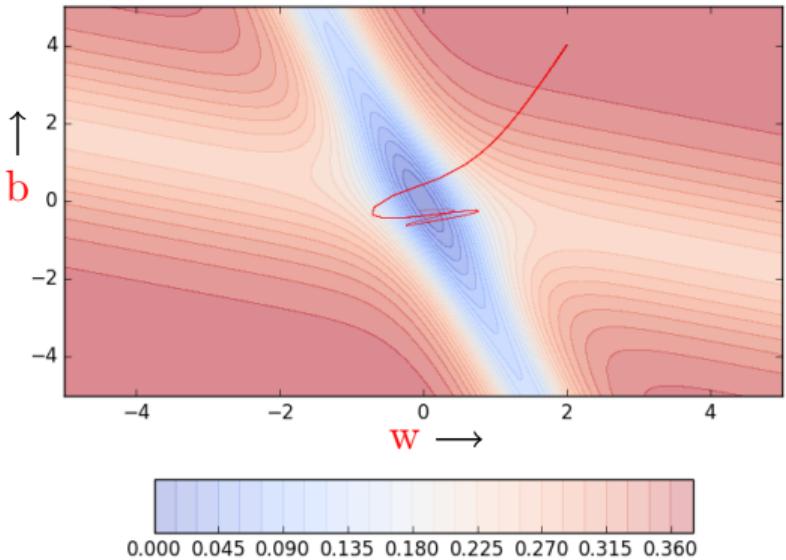


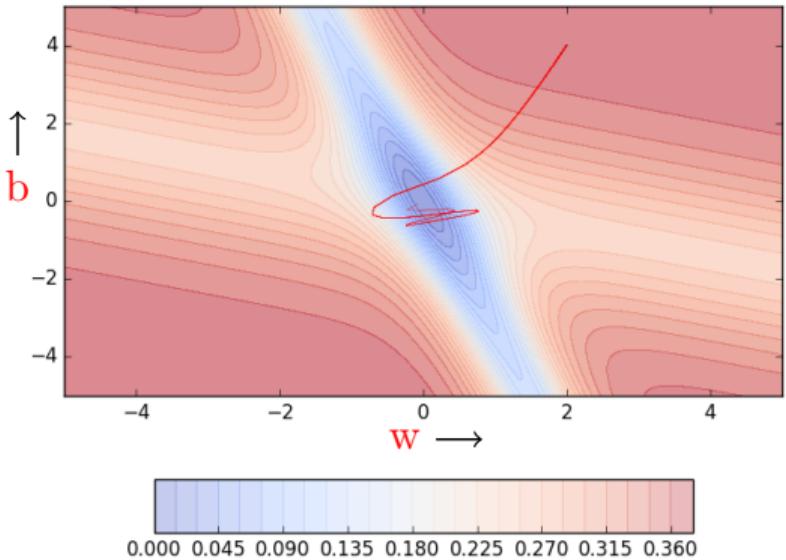


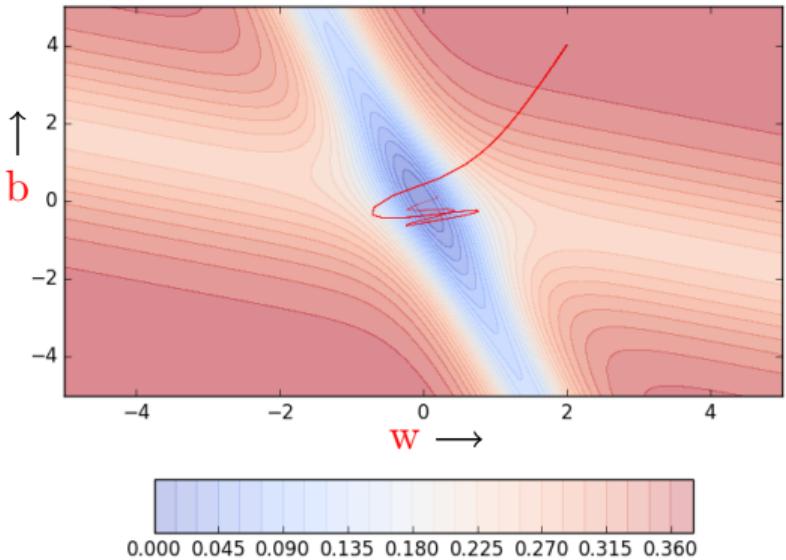


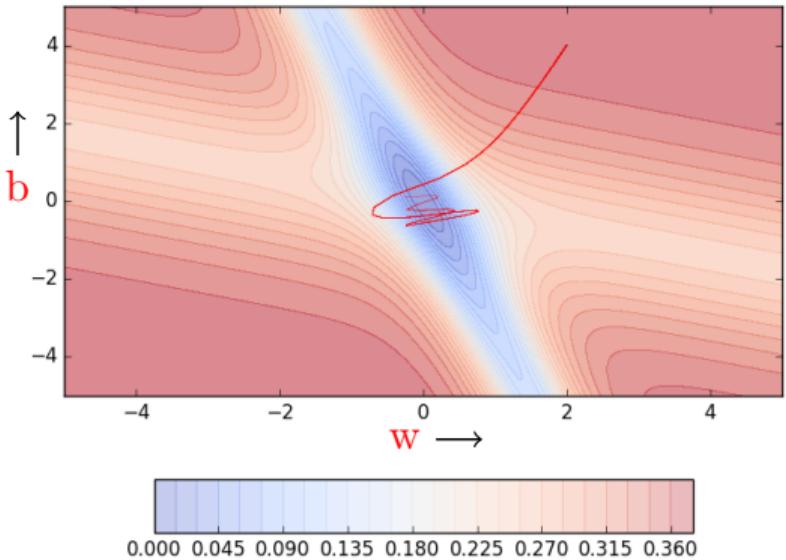


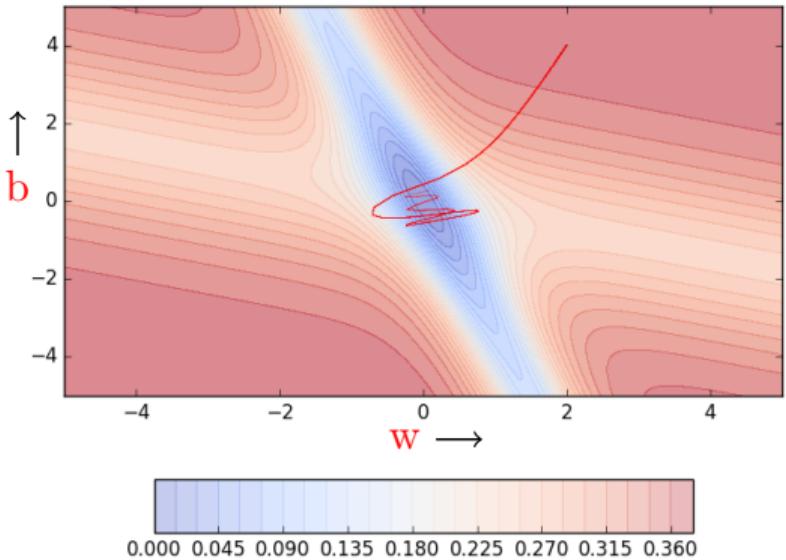


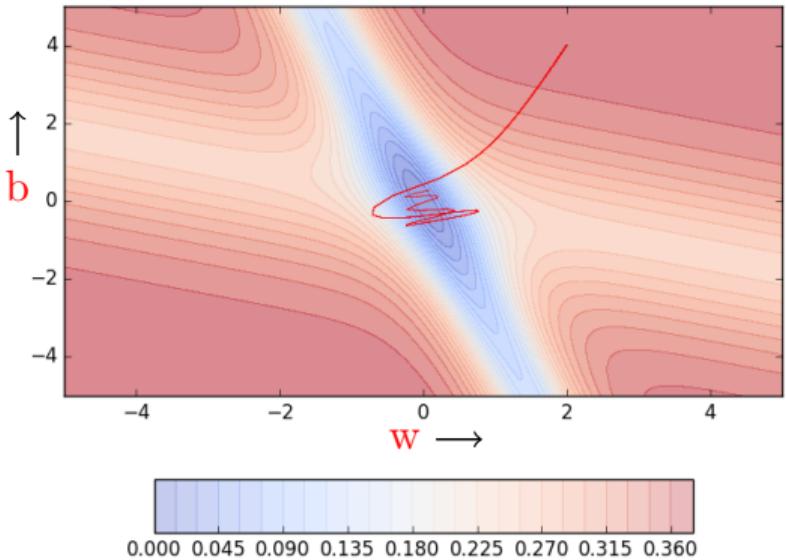


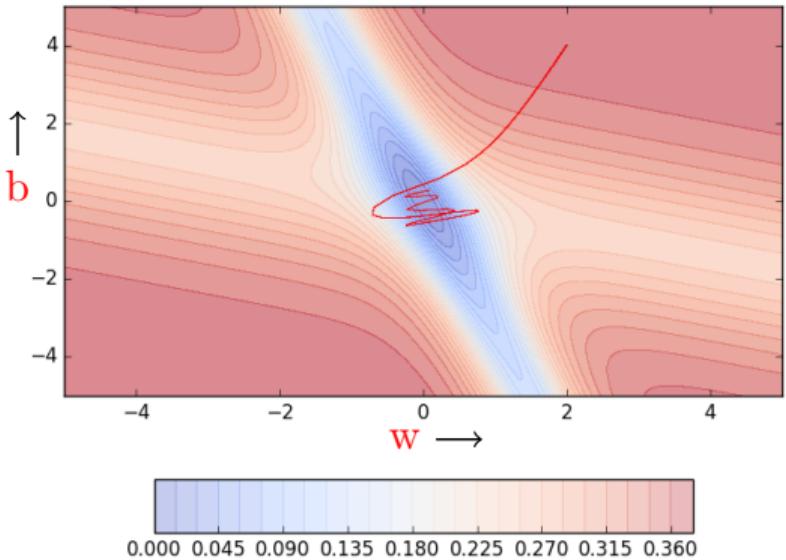


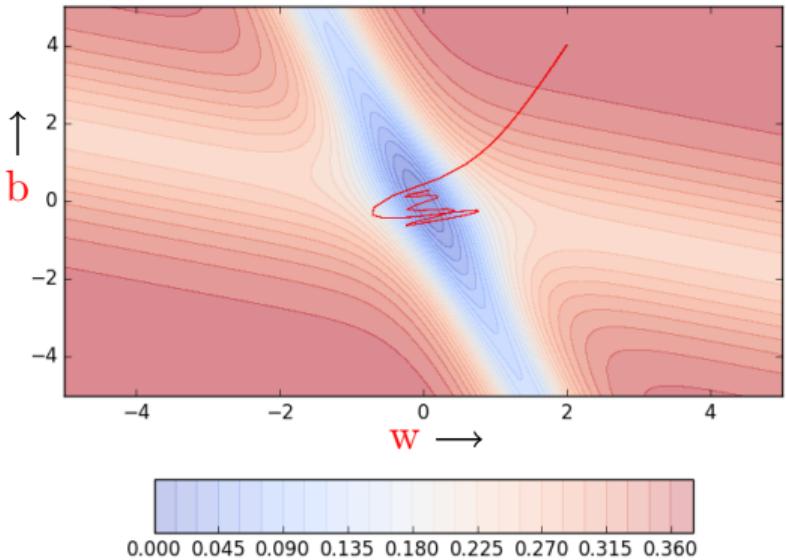


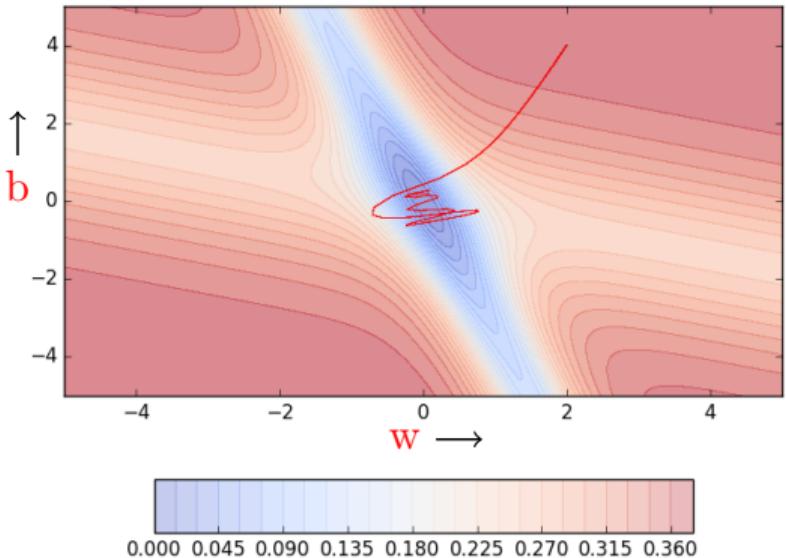




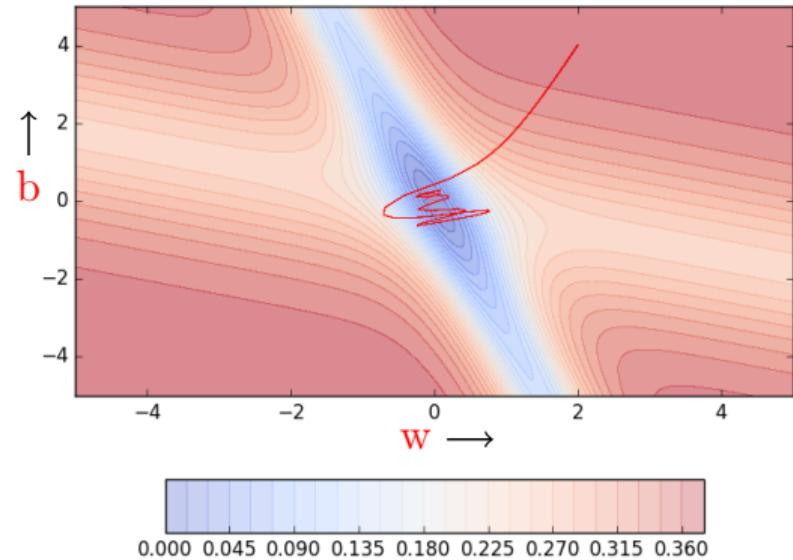




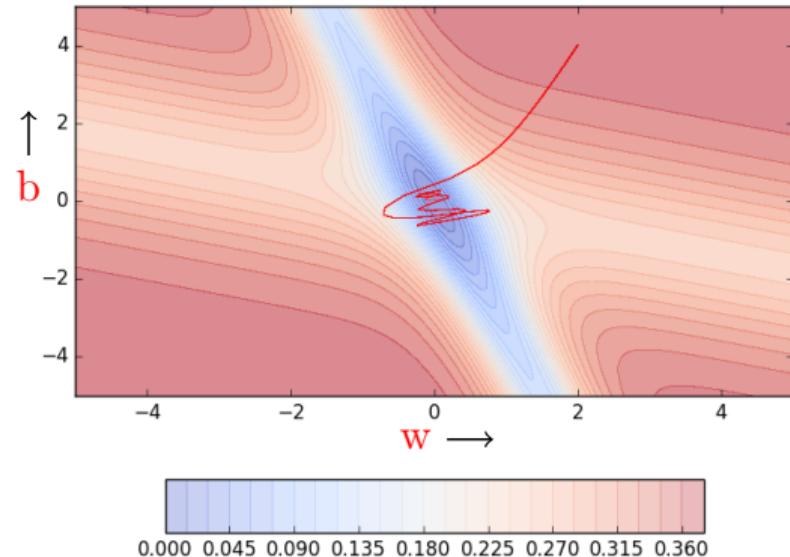




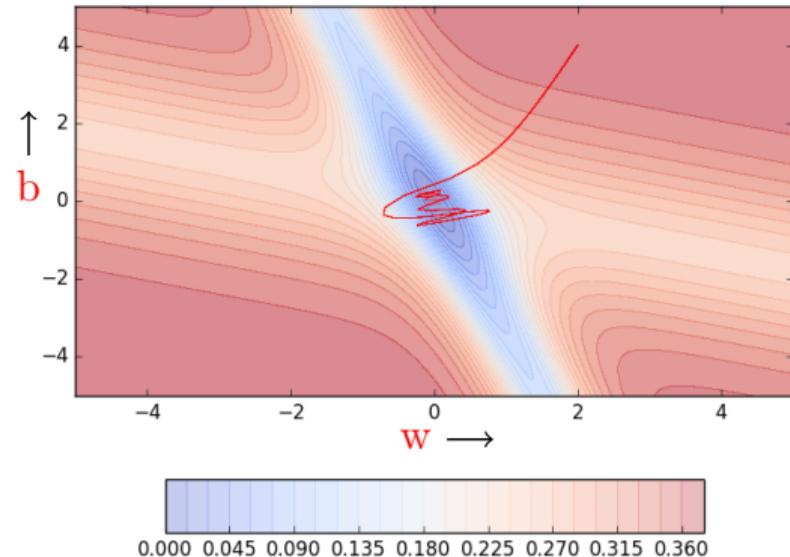
- Momentum based gradient descent oscillates in and out of the minima valley as the momentum carries it out of the valley



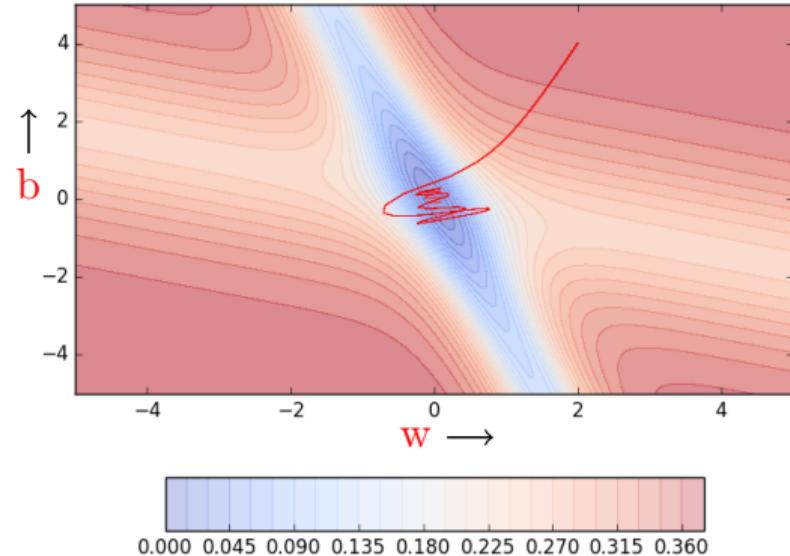
- Momentum based gradient descent oscillates in and out of the minima valley as the momentum carries it out of the valley
- Takes a lot of *u*-turns before finally converging



- Momentum based gradient descent oscillates in and out of the minima valley as the momentum carries it out of the valley
- Takes a lot of *u*-turns before finally converging
- Despite these *u*-turns it still converges faster than vanilla gradient descent



- Momentum based gradient descent oscillates in and out of the minima valley as the momentum carries it out of the valley
- Takes a lot of *u*-turns before finally converging
- Despite these *u*-turns it still converges faster than vanilla gradient descent
- After 100 iterations momentum based method has reached an error of 0.00001 whereas vanilla gradient descent is still stuck at an error of 0.36



Let's look at a 3d visualization and a different geometric perspective of the same thing...

