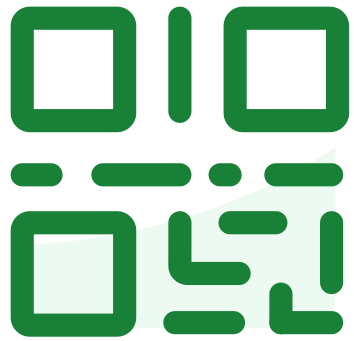


Do not edit
How to change the
design



**Join at slido.com
#1136566**

① The Slido app must be installed on every computer you're presenting from

slido

Homework 2 Overview

This homework is worth \$1.7 billion!



1136566

In homework 2, you'll apply ML techniques to data from Arena AI.

- Arena AI is a crowdsourced LLM ranking platform
 - Valuated at \$1.7B (up from \$600M a year ago!)
- Every month, 10 million users have 700+ million conversations on Arena AI and provide feedback on AI responses

Task: By analyzing Arena AI data,

- Predict the "strength" of models and win rates against other models
- Determine what hidden factors in prompts or responses affect preferences

what is the Gateway building at Berkeley? ▾

what is the Gateway building at Berkeley?

Assistant A

The Gateway building at UC Berkeley refers to a major new academic facility associated with the university's data science and computing initiatives.

Key points about it:

- **Purpose:** It's designed to house the College of Computing, Data Science, and Society (CDSS), which is one of Berkeley's newer academic units focused on data science, computing, and related interdisciplinary fields.
- **Location:** It's being built on the north side of the Berkeley campus, near Hearst Avenue, at the site formerly occupied by an older building (the previous Hearst Field Annex / near the northwest area of campus).

Assistant B

There isn't a single, widely recognized building simply called "The Gateway" at UC Berkeley, and the name can refer to a few different things depending on the context. Here are the most likely referents:

1. **Campus entrance/planning context** – "Gateway" is often used colloquially to describe entry points to the Berkeley campus, most notably **Sather Gate**, the historic archway and entrance on Telegraph Avenue that leads into the central campus core.
2. **Residential or academic buildings** – Some UC Berkeley colleges or departments have buildings with "Gateway" in their name as part of newer construction or renovation projects (e.g., housing complexes, interdisciplinary research spaces, or school-specific facilities). These are typically named in connection with a specific program or donor.

← A is better ⚡ Both are good ⚡ Both are bad B is better →

Ask followup...

+ ⚡ Battle ▾ T Text ▾

The Arena formerly known as Chatbot



1136566

Fun fact: Arena AI was a research project that Professor Gonzalez led at Berkeley!

You'll be reading the original paper as part of the homework

- And a bonus paper that vibe checks data from the arena
- Paper reading is a valuable skill that helps you better understand methods and ideas
- How do I read a paper?
 - [How to Read a Paper](#) by S. Keshav (Cambridge Professor, Cal Alum 🐻)
 - Practice helps!
 - We will guide you through the process

Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference

Published as a conference paper at ICLR 2025

VIBECHECK: DISCOVER & QUANTIFY QUALITATIVE DIFFERENCES IN LARGE LANGUAGE MODELS

Lisa Dunlap Krishna Mandal Trevor Darrell Jacob Steinhardt Joseph Gonzalez
UC Berkeley UC Berkeley UC Berkeley UC Berkeley UC Berkeley

ABSTRACT

Large language models (LLMs) often exhibit subtle yet distinctive characteristics in their outputs that users intuitively recognize, but struggle to quantify. These “vibes” – such as tone, formatting, or writing style – influence user preferences, yet traditional evaluations focus primarily on the singular vibe of correctness. We introduce **VibeCheck**, a system for automatically comparing a pair of LLMs by discovering identifying traits of a model (“vibes”) that are well-defined, differentiating, and user-aligned. VibeCheck iteratively discovers vibes from model outputs and then utilizes a panel of LLM judges to quantitatively measure the utility of each vibe. We validate that the vibes generated by VibeCheck align with those found in human discovery and run VibeCheck on pairwise preference data from real-world user conversations with Llama-3-70b vs GPT-4. VibeCheck reveals that Llama has a friendly, funny, and somewhat controversial vibe. These vibes predict model identity with 80% accuracy and human preference with 61% accuracy. Lastly, we run VibeCheck on a variety of models and tasks including summarization, math, and captioning to provide insight into differences in model behavior. VibeCheck discovers vibes like Command X prefers to add concrete intros and conclusions when summarizing in comparison to TNG, Llama-405b often overexplains its thought process on math problems compared to GPT-4o, and GPT-4 prefers to focus on the mood and emotions of the scene when captioning compared to Gemini-1.5-Flash. Code and vibe visualizer found at <https://bench-mark.org/>

1 INTRODUCTION

vibe check : A process by which a group obtains a subjective assessment of another person, place, or thing.
– *Urban Dictionary*

How a large language model writes a story, explains a concept, or edits an essay can be evaluated along many different dimensions such as creativity, formatting, and writing style. However, most evaluations focus on one dimension: “correctness”. State-of-the-art in evaluation methods remain largely focused on measuring accuracy for question answering and analytical reasoning tasks (Hendrycks et al., 2021a; Wang et al., 2019b; Hendrycks et al., 2021c), and methods which aim to provide a more holistic view

arXiv:2403.04132v1 [cs.AI] 7 Mar 2024

arXiv:2410.12851v7 [cs.CL] 19 Apr 2025

1



Paper Reading Homework Requirements



1136566

- You will write short-answer responses to questions.
- The questions are designed to help you understand:
 - What problem the paper is trying to solve
 - The "state of the world" at the time the authors wrote the paper
 - The solutions, key insights, or contributions the authors present
 - How the authors arrived at their solutions/insights
 - Limitations and future extensions

Homework Summary



1136566

Coding

- Apply ML frameworks you've been learning about in lecture!
- PCA, GMMs, logistic regression, feature exploration

Written questions

- Written questions to practice your math muscles
- Linear regression, GMMs, MLE/MAP, bias-variance tradeoff

Reading papers

- Reading research papers!
- [Chatbot Arena](#) and [VibeCheck](#)



1136566

Important HW2 Reminders

- Due in 10 days on Friday, October 9 at 11:59 PM
 - No part 2
- You are responsible for 3 assignments on Pensive:
 - HW 2 Coding
 - Submit the exported zip file; the autograder will automatically upload to HW2 Coding PDF
 - Watch the setup and submission walkthroughs on Askademia
 - HW 2 Written
 - Submit your responses to the **written** questions
 - HW 2 Paper Write-up
 - Submit your responses to the **paper** questions
- **Start early!**

Good luck and have fun :)



How is homework going?



1136566

Getting the Most out of UC Berkeley

You are **supposed to struggle** with homework

➤ It is an **exercise** for your brain

Things may be **broken** and it may **not always be your fault**

➤ Learning how to **work around** these challenges is *important*

Build your community ask for help and help others

➤ Opportunity to learn how to **learn from and teach others**

➤ Go to **office hours, homework parties**

➤ How we can help?

Lecture 10

Optimization and Gradient Descent

The core algorithm of modern machine learning

EECS 189/289, Fall 2026 @ UC Berkeley

Joseph E. Gonzalez and Narges Norouzi

Key Factors in Generative AI Revolution



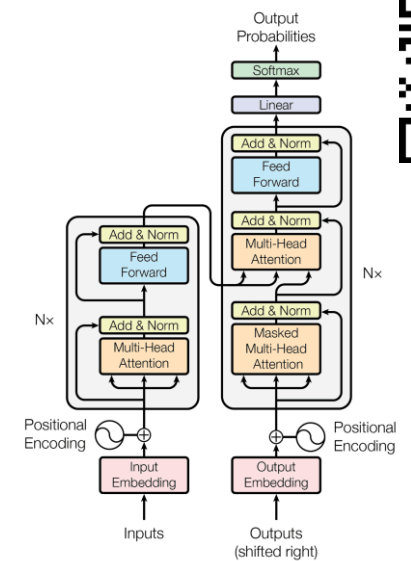
Data

$$\ln p(\text{Word}_n \mid \{\text{Word}_i\}_{i=1}^{n-1})$$

Loss Functions

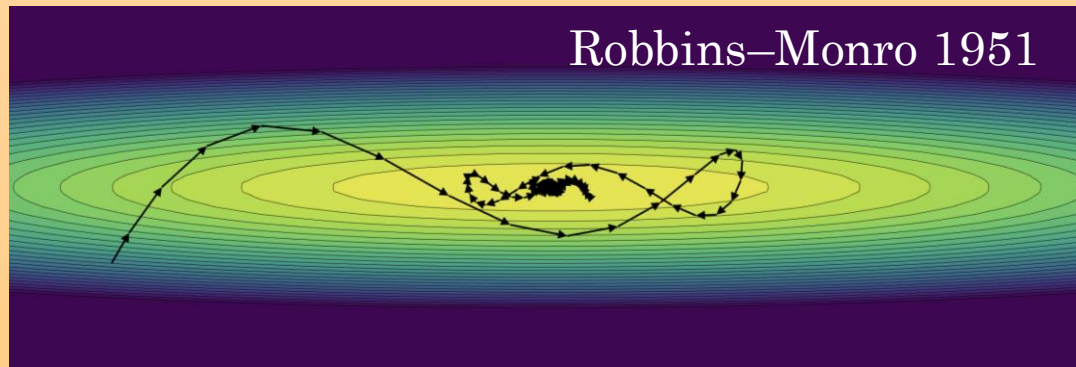


Compute



1136566

Neural Architectures



Robbins-Monro 1951

Stochastic Gradient Descent

 PyTorch



TensorFlow

Automatic Diff.

The General Optimization Problem



1136566

The General Optimization Problem

Most problems in ML can be reduced to the following general **optimization problem**

Objective Function

$$w^* = \arg \min_{w \in \Theta} f(w)$$

Solution
(Learned Parameters)

Constraints



1136566

Optimization Constraints

The **constraints** define the valid set of parameters.

$$w^* = \arg \min_{w \in \Theta} f(w)$$

Constraints

For the rest of this lecture, we will focus on the **unconstrained setting**.

$$\Theta = \mathbb{R}^D$$

The most common case in ML is **unconstrained optimization** in which $\Theta = \mathbb{R}^D$.

In some cases, there may be integer (e.g., $\Theta = \mathbb{Z}^D$) or probability constraints (recall PCA and the EM Algorithm):

$$\Theta = \left\{ w \mid w \in \mathbb{R}^D, 0 \leq w \leq 1, \sum w_i = 1 \right\}$$



1136566

The Objective Function

The **shape** of the objective function plays a big role in the complexity of solving the optimization problem.

Objective Function

$$w^* = \arg \min_{w \in \Theta} f(w)$$

The **Objective Function** in ML is often referred to as the **loss** or **error** function and depends on the **data**.

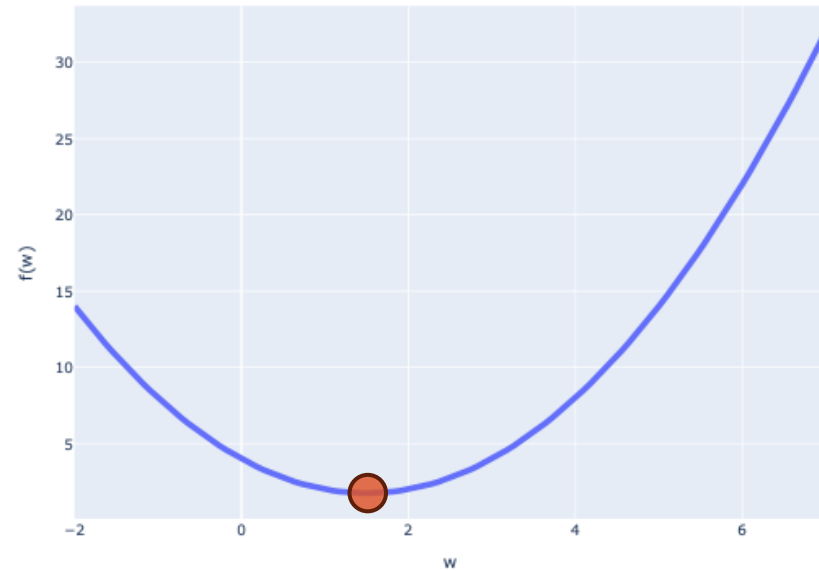
$$f(w) = E[w; \mathcal{D}]$$

Examples Optimization Problems

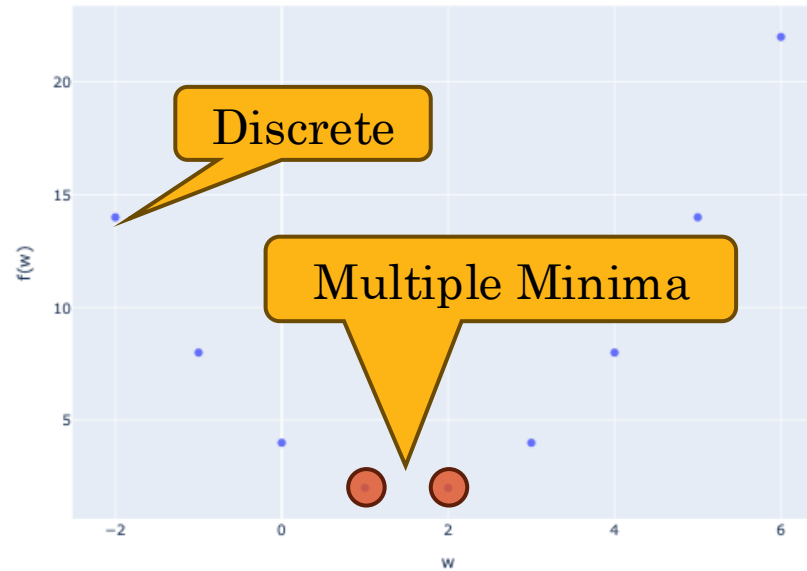


1136566

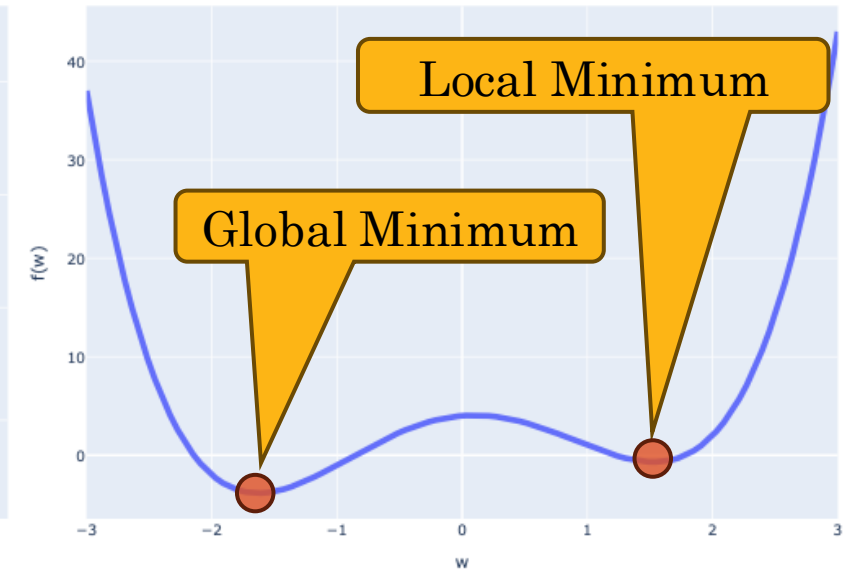
$$\arg \min_{w \in \mathbb{R}} w^2 - 3w + 4$$



$$\arg \min_{w \in \mathbb{Z}} w^2 - 3w + 4$$



$$\arg \min_{w \in \mathbb{R}} w^4 - 5w^2 + w + 4$$



How do we solve these optimization problems?

What properties of the objective function make it “easier” to solve?



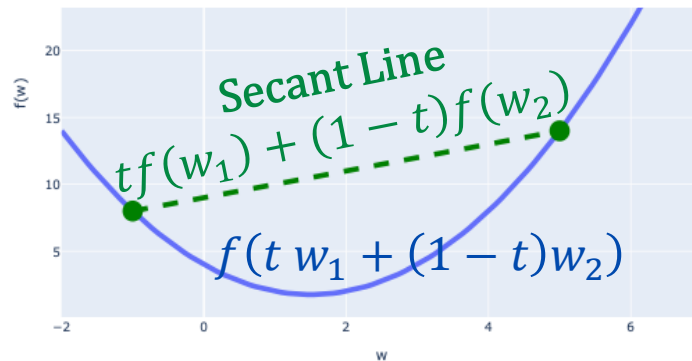
1136566

Convex Functions

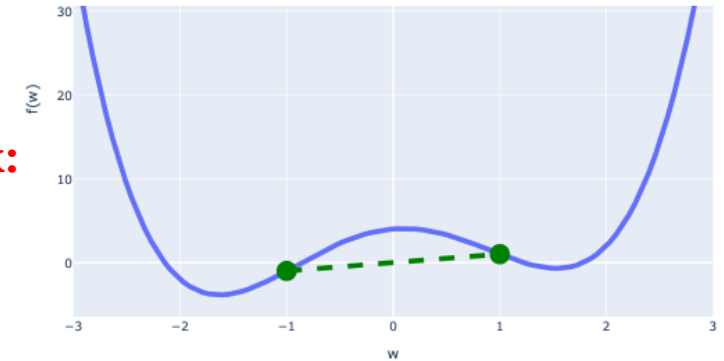
Definition: A function is **convex** on the domain Θ if for any pair of points $\forall w_1, w_2 \in \Theta$ and for all $0 \leq t \leq 1$:

$$f(t w_1 + (1 - t)w_2) \leq t f(w_1) + (1 - t)f(w_2)$$

Convex:



Non-Convex:



In general, there are **efficient algorithms** to reliably minimize convex functions.



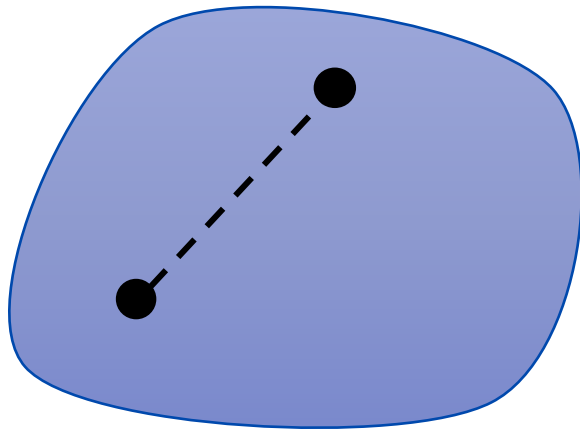
1136566

Convex (Constraint) Sets

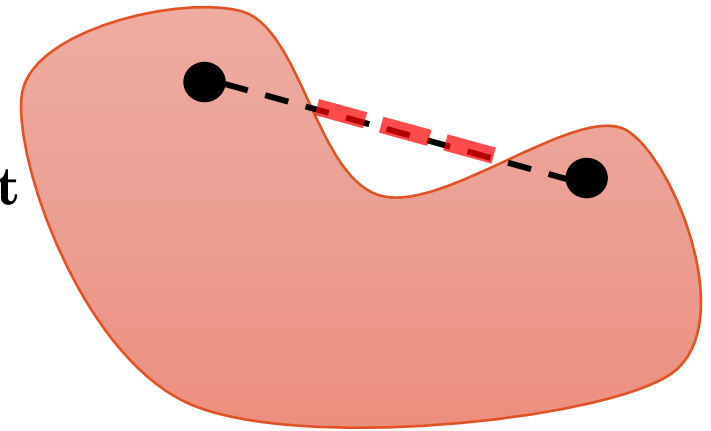
Definition: A set S is convex if it contains the **line segment** between **any two points** in the set:

$$\forall w_1, w_2 \in S; 0 \leq t \leq 1 : t w_1 + (1 - t) w_2 \in S$$

Convex Set



Non-convex Set



In general, there are **efficient algorithms** to reliably minimize **convex functions** over **convex sets**.

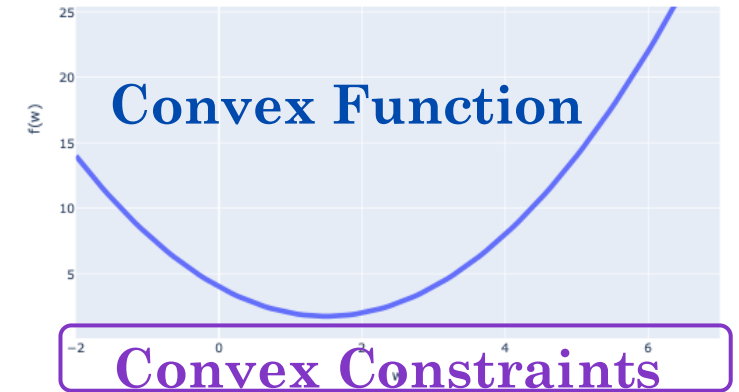


1136566

Optimization Problems

Optimization of a **convex objective function** over a **convex constraint set**.

- *ML problems that we can solve efficiently*
- Primary focus of [EECS-127](#) and [the cvx book](#)



Many *classic* ML problems are **convex**.

- **Least Squares Regression** and **Logistic Regression**

Most of **modern ML (deep learning)** is *not convex*.

- Apply convex optimization **techniques** to non-convex problems
- In this class, introduce basic **optimization concepts**

Minimizing the Error & Optimizing in Weight-Space



1136566

Error Minimization

Our goal is to **minimize the prediction error** of our model:

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^D} \frac{1}{N} \sum_{n=1}^N \mathbf{Loss}(f_{\mathbf{w}}(x_n), t_n)$$

- where $f_{\mathbf{w}}(x_n)$ is the prediction of the model $f_{\mathbf{w}}$ with parameters $\mathbf{w}$ and $\mathbf{Loss}(\hat{t}, t)$ is the loss function (e.g., L2-loss or CE-loss)

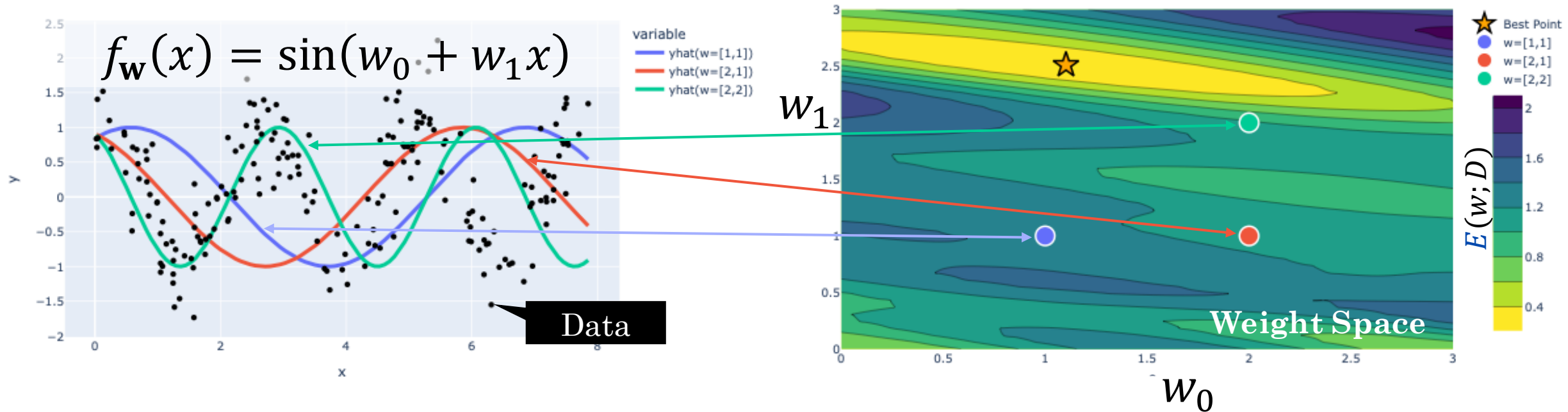
Because we are only interested in the **error as a function of the parameters $\mathbf{w}$** we typically rewrite the objective:

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^D} E(\mathbf{w}; \mathcal{D})$$

- where $\mathcal{D}$ indicates that the error E depends on the data.

Points on the Error Surface

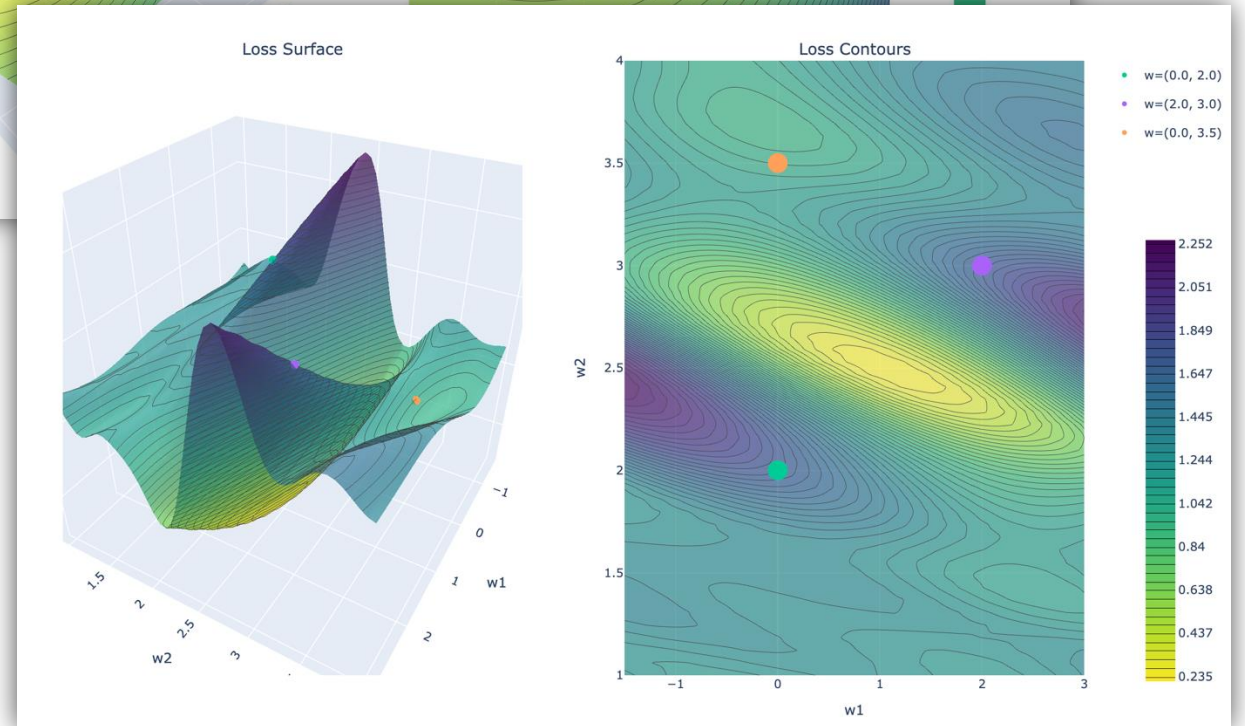
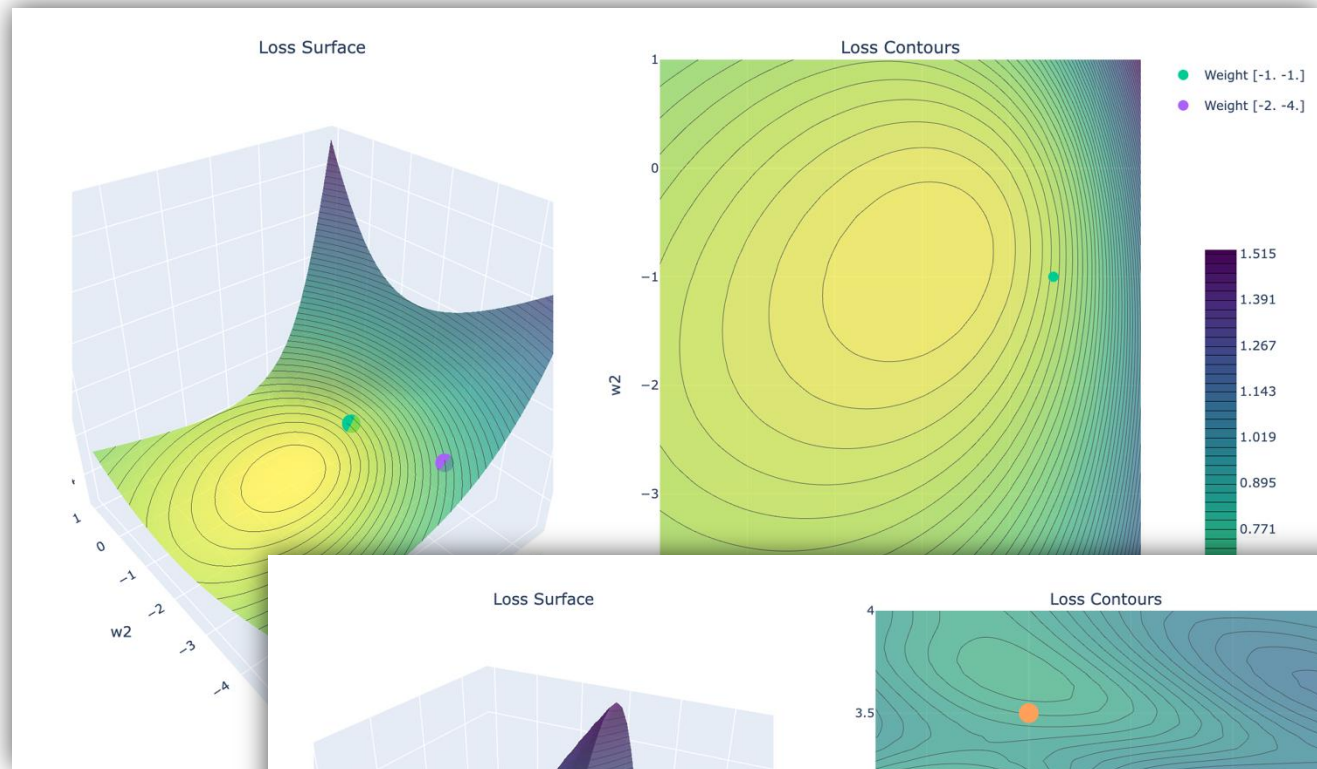
Each **parameterization of a model** corresponds to a point on the error surface $E(w; \mathcal{D})$.



Goal: find **weights** with lowest error: $\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^D} E(w; \mathcal{D})$

Demo

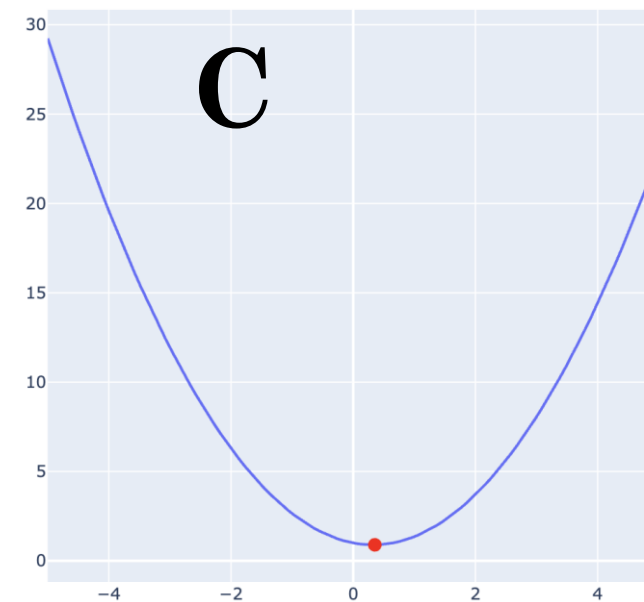
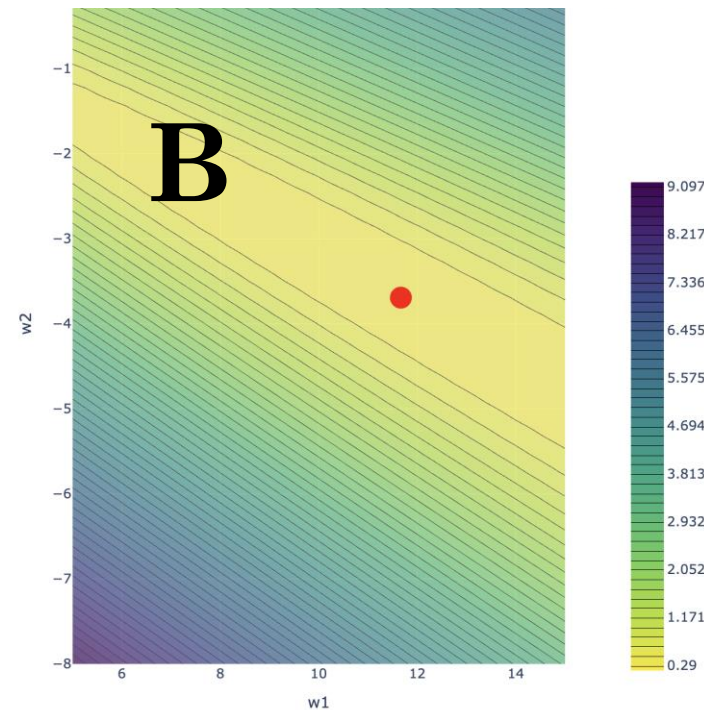
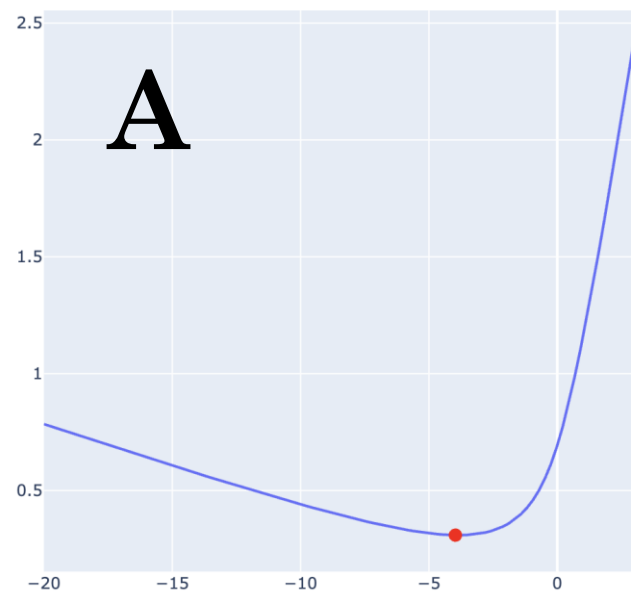
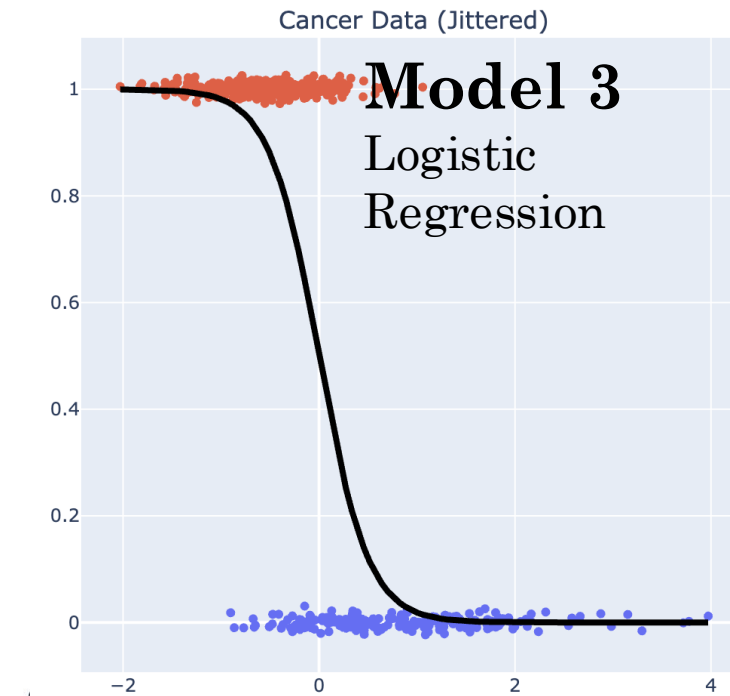
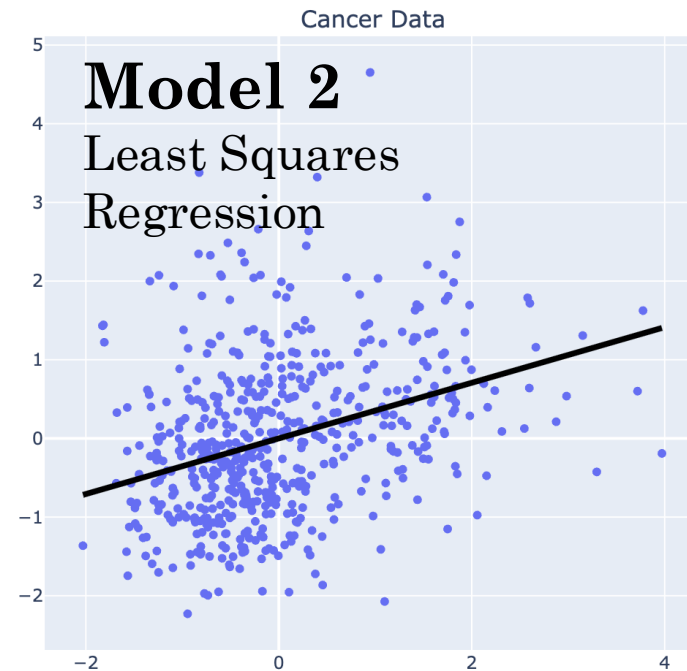
Understanding the Error Surface



1136566



1136566





Match the model with the loss.

Calculus and Optimization

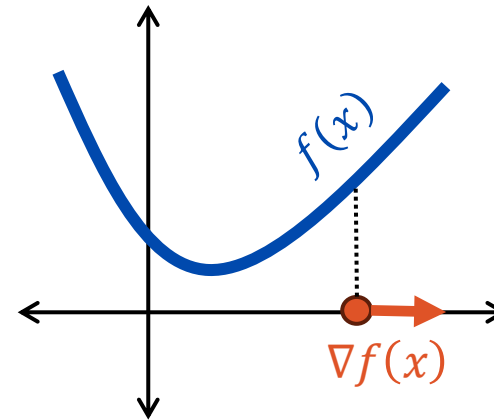


1136566

The Gradient

The gradient of a **scalar-valued function** $f: \mathbb{R}^d \rightarrow \mathbb{R}$ is the **column vector-valued function** $\nabla f: \mathbb{R}^d \rightarrow \mathbb{R}^d$ of partial derivatives of f

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{bmatrix}$$



Important Properties

- The gradient $\nabla f(x)$ points in the **direction of steepest ascent**.
- The gradient $\nabla f(x) = 0$ at local minima, maxima, and saddle points.
 - We already used this property to solve for the MLE/MAP (...many times...)

Gradient Exercise

Consider the function:

$$E(w) = (w_0 - 1)^2 + (w_1 - 2)^2 + 1$$

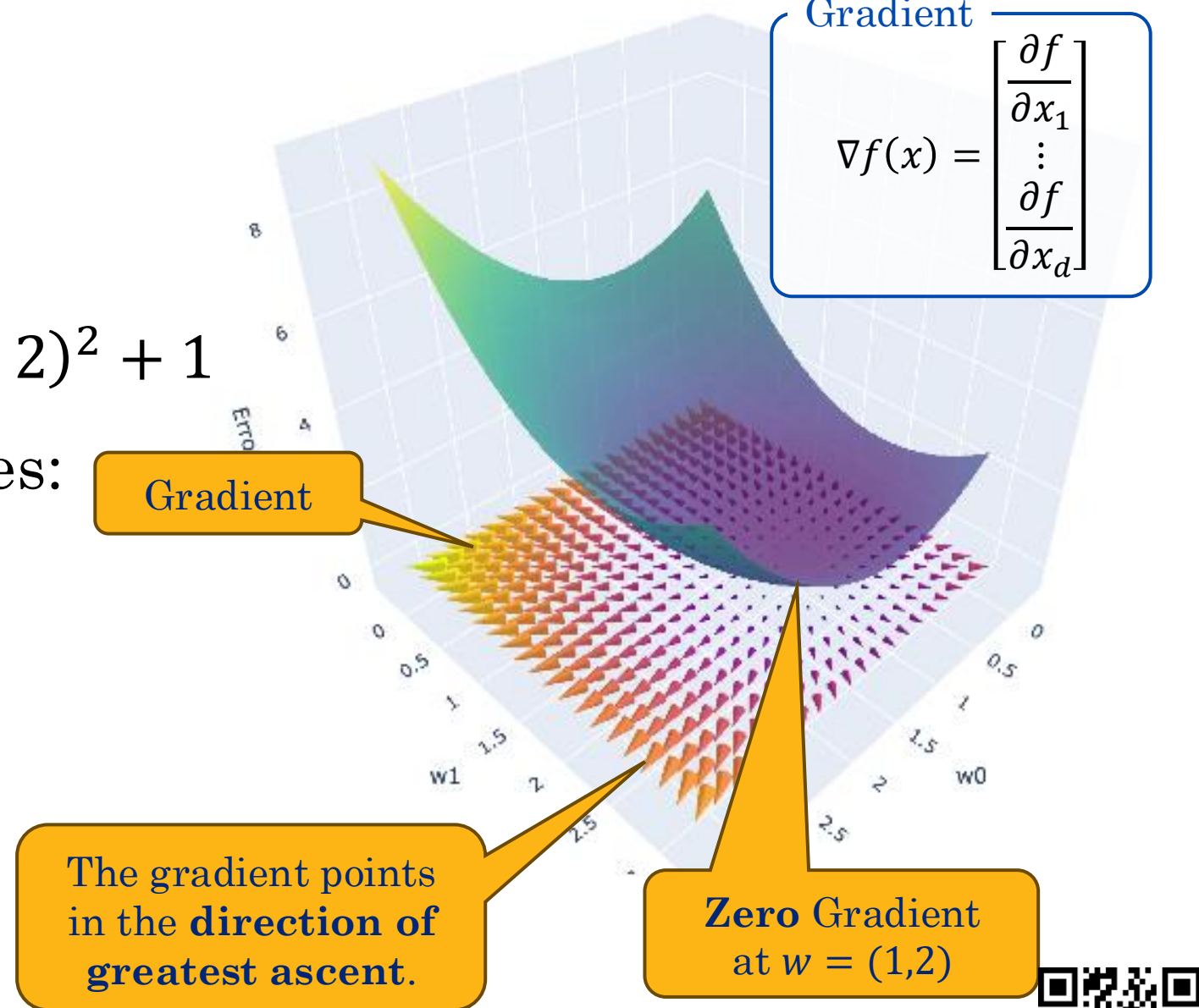
Compute the partial derivatives:

$$\frac{\partial}{\partial w_0} E(w) = 2(w_0 - 1)$$

$$\frac{\partial}{\partial w_1} E(w) = 2(w_1 - 2)$$

Then the gradient is:

$$\nabla E(w) = \begin{bmatrix} 2(w_0 - 1) \\ 2(w_1 - 2) \end{bmatrix}$$





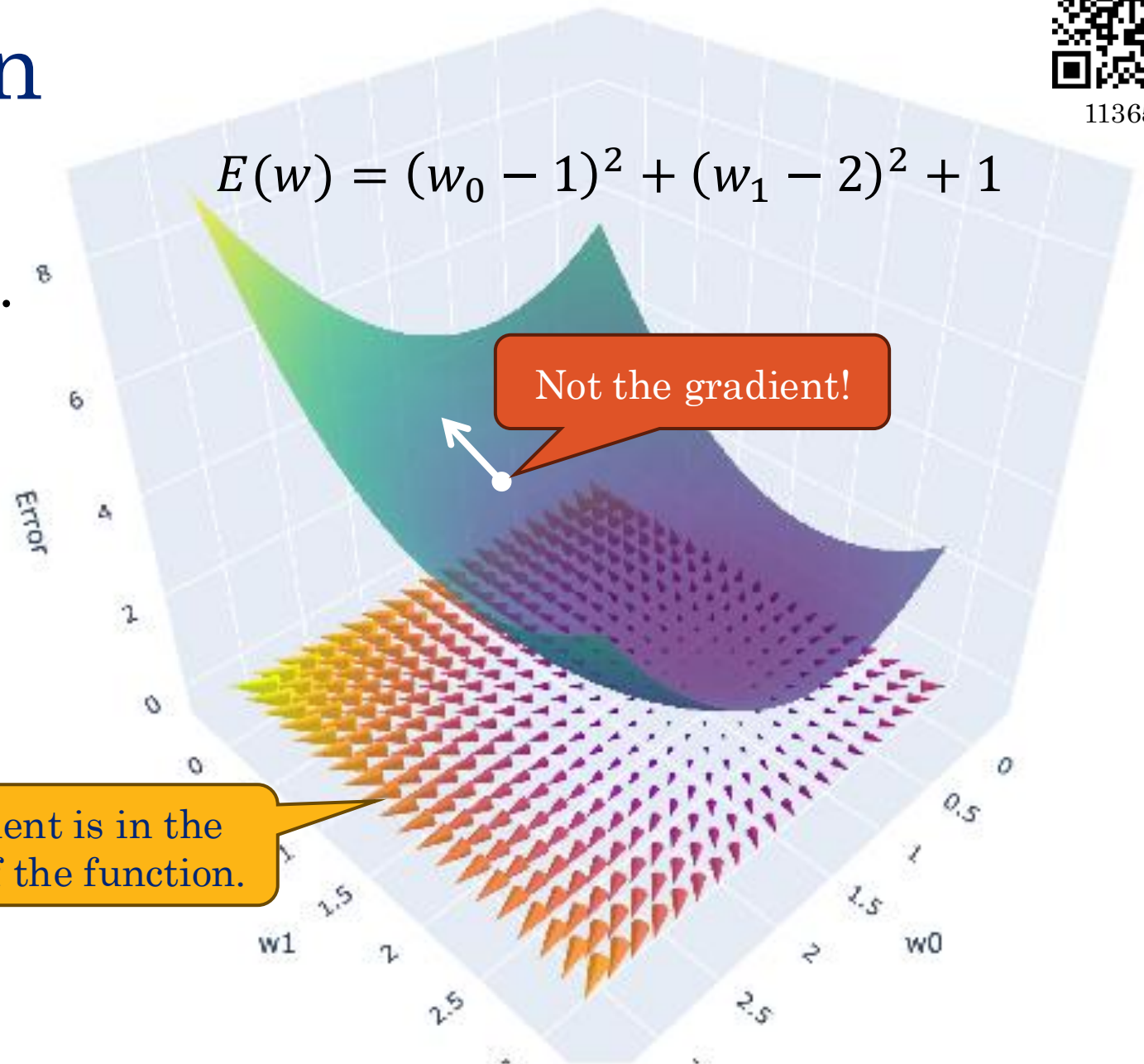
If w is a D -dimensional vector and $E(w)$ is a scalar-valued function, what is the dimensionality of the gradient $\nabla E(w)$ with respect to w ?

Common Confusion

The gradient does not lie on the surface of the function.

Example: $E: \mathbb{R}^2 \rightarrow \mathbb{R}$

The gradient $\nabla E: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ is **2-dimensional (not 3)**



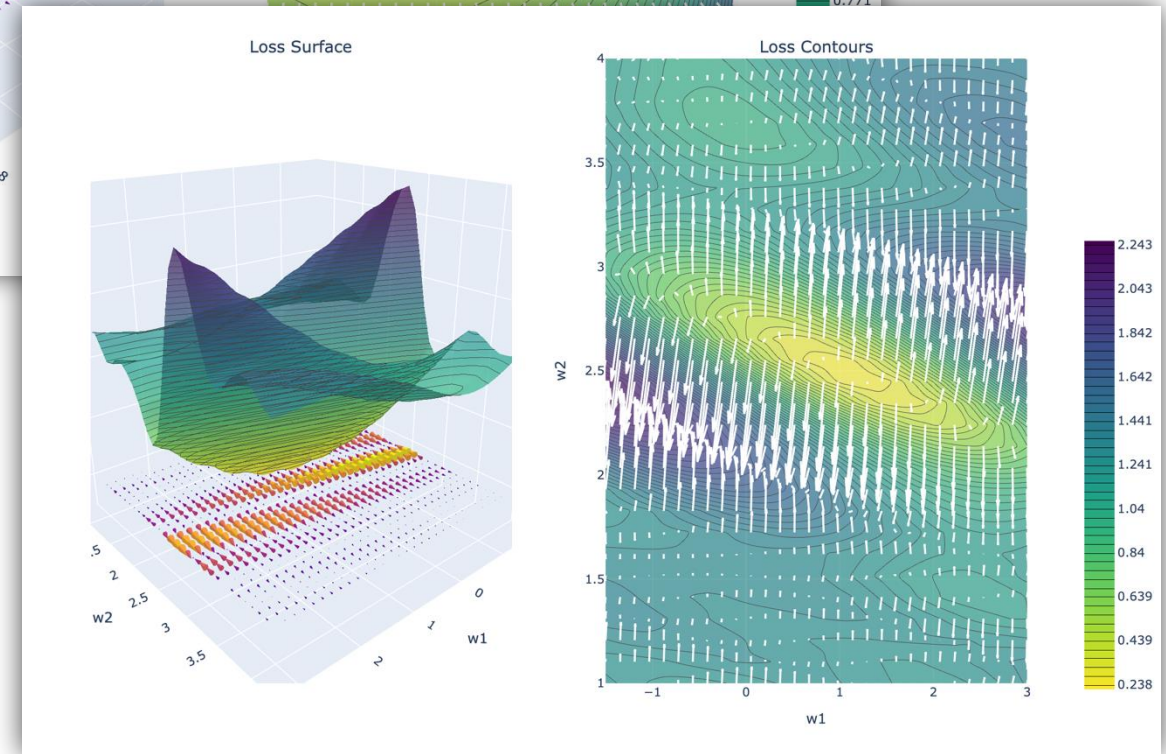
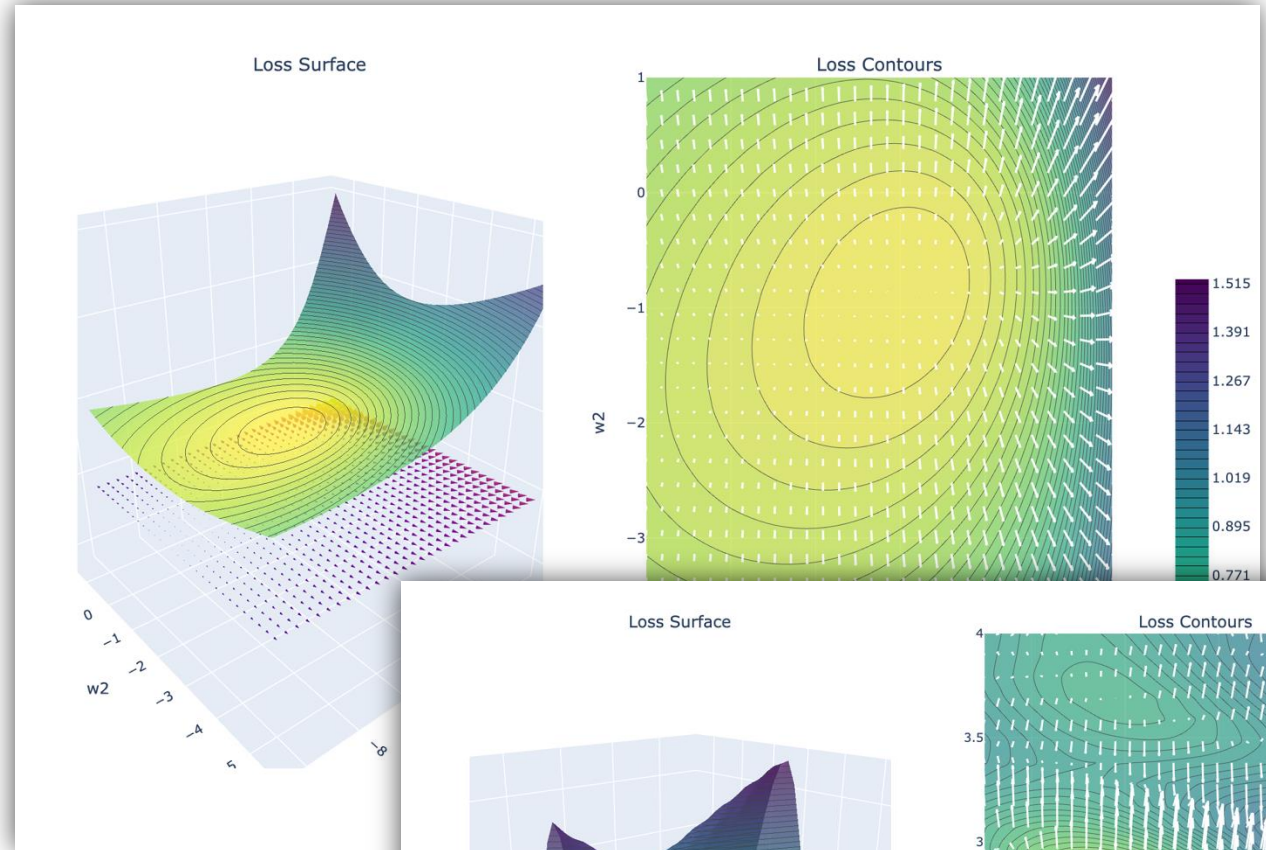
1136566

Demo

The Gradient of the Error Function



1136566



Stationary Points and Zero Gradient

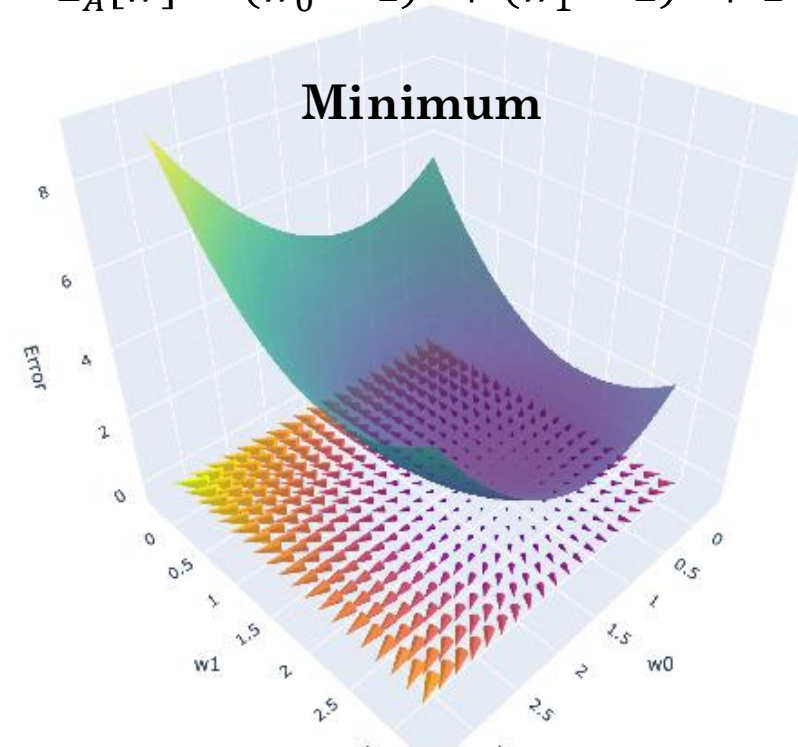


1136566

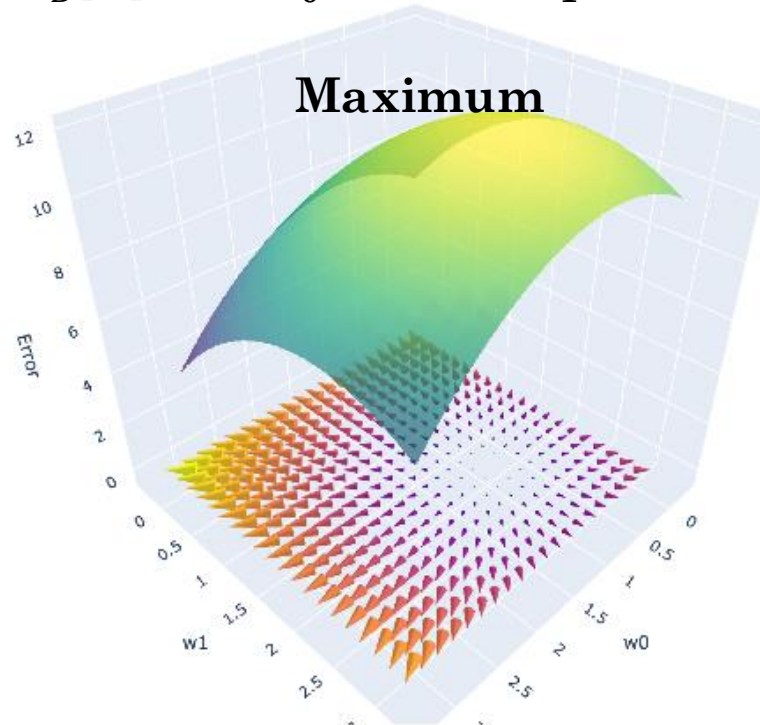
Anywhere the gradient is zero is a **stationary point**.

- Gradient is zero at local minima but zero gradient does not imply a minimum

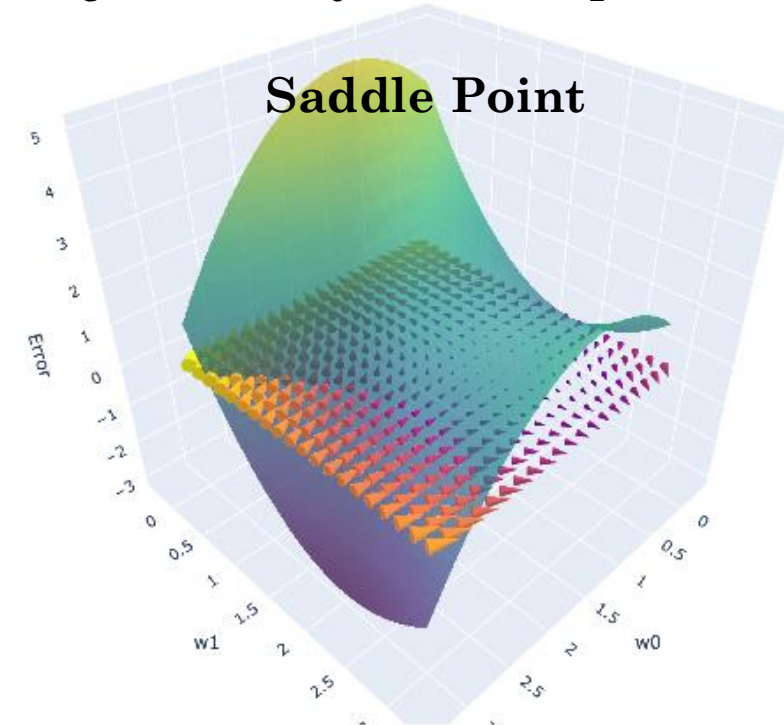
$$E_A[w] = (w_0 - 1)^2 + (w_1 - 2)^2 + 1$$



$$E_B[w] = -(w_0 - 1)^2 - (w_1 - 2)^2 + 12$$



$$E_C[w] = -(w_0 - 1)^2 + (w_1 - 2)^2 + 1$$



Plots in demo notebook.

Analytical Optimization using Gradients



1136566

In some situations, (e.g., **ordinary least squares**) we can find the stationary points by solving:

$$\nabla E(w) = 0$$

- For these problems we often know that there is one stationary point and that it is the minimum.

However, for many machine learning applications (e.g., logistic regression and neural networks) **we cannot solve for the stationary points analytically.**

Iterative optimization

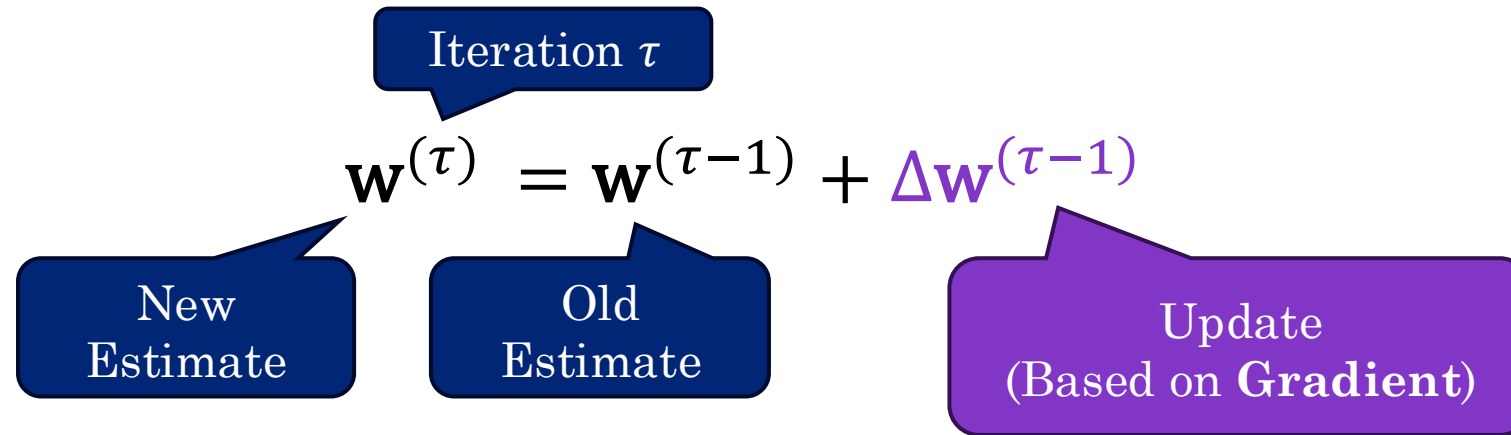
Gradient Descent



1136566

Iterative Optimization Algorithms

Iterative optimization algorithms start with an **initial estimate** $\mathbf{w}^{(0)}$ and then **iteratively refine** that estimate:



until **stopping criterion** is achieved.

The choice of initial estimate $\mathbf{w}^{(0)}$, how the update $\Delta \mathbf{w}^{(\tau-1)}$ is computed, and the **stopping criterion** define the **iterative optimization algorithm**.



1136566

Loss Game Demo

Interactive version of slides.



The Loss Game

Minimize the error by turning the knobs.

Machine: Parabola (3 knobs) New puzzle Copy link

1 Turn the knobs 2 Probe 3 Follow the gradient 4 Gradient descent

LEVEL 1: TURN THE KNOBS

This machine computes the error $E(w)$ of a model with parameters w . The height of the tower is the error.

Goal: turn the knobs until the tower drops below the gold line.

Each time you let go of a knob, the machine evaluates $E(w)$ once. Every evaluation requires a pass over all the training data, so evaluations are expensive. Try to win with as few as possible.

w_0 w_1 w_2

+2.30 +2.40 -3.10

Error $E(w)$
24.6813

Error evaluations
1

Gradient evaluations
0

Best error so far
24.6813

Reset Letting go of a knob evaluates the error once.

ERROR CURVE

$E(w)$, log scale

error evaluation number

goal

WEIGHT SPACE show

The error surface is hidden. The machine is a black box: all you can see is the height of the error tower.

DATA SPACE show

The training data and model are hidden.

SCOREBOARD FOR THIS PUZZLE reveal answer

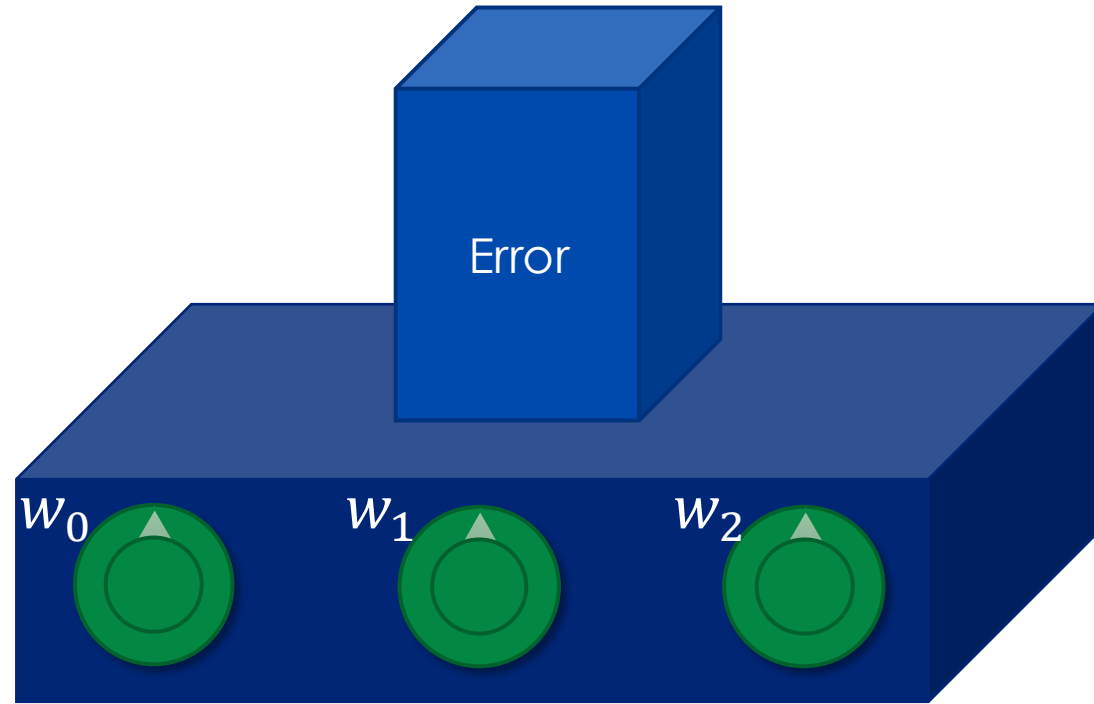
Level	Your best (fewest evaluations)
1. Turn the knobs	not solved yet
2. Probe	not solved yet
3. Follow the gradient	not solved yet
4. Gradient descent	not solved yet

https://eecs189.org/fa26/resources/assets/lectures/lec10/loss_game

Intuition



1136566



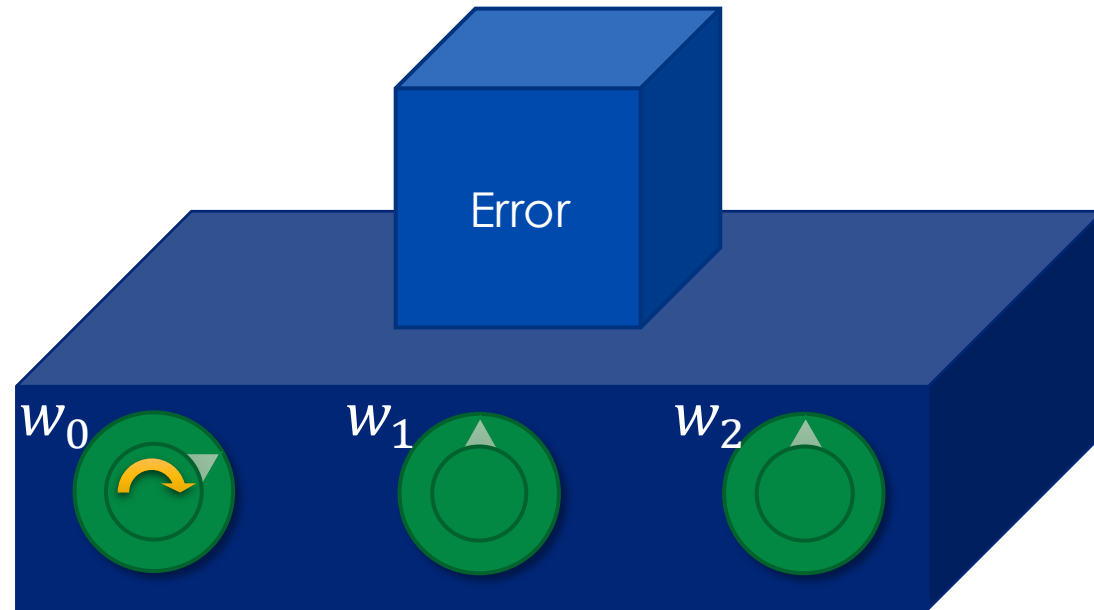
Goal: Minimize the loss by turning the knobs.

Try the new [loss game](#).

Intuition



1136566

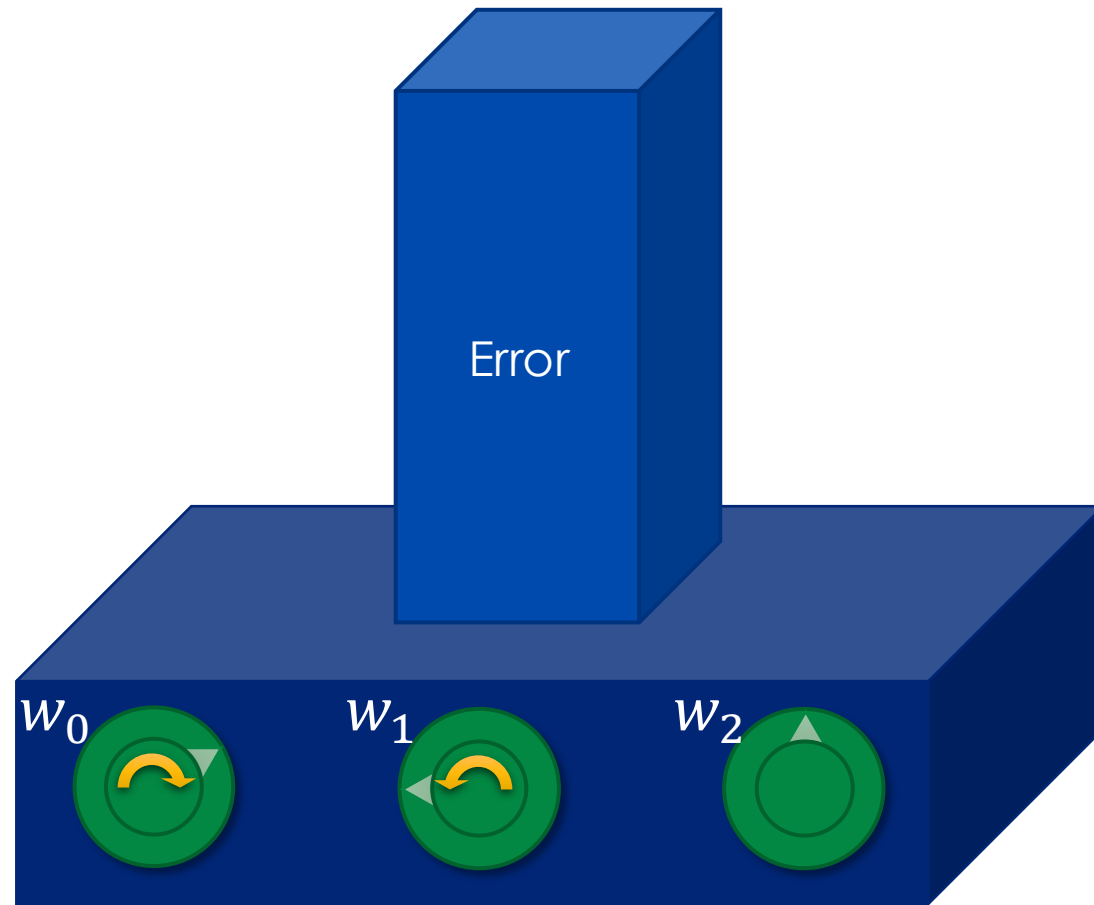


Try the new [loss_game](#).

Intuition



1136566

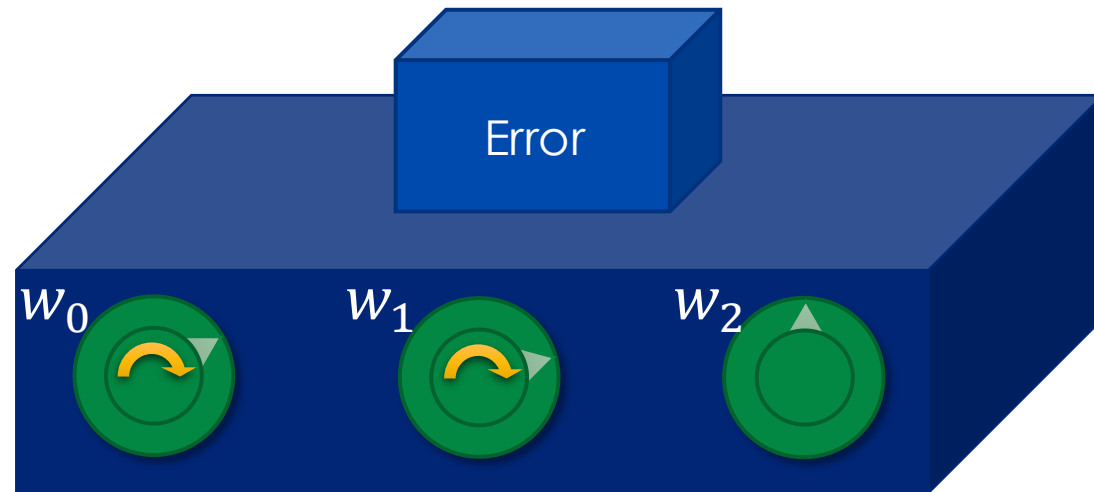


Try the new [loss_game](#).

Intuition



1136566

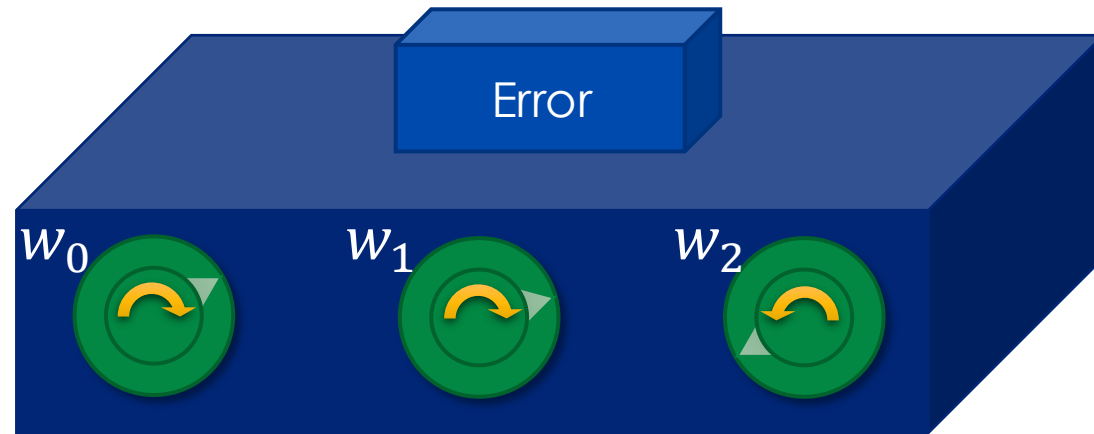


Try the new [loss_game](#).

Intuition



1136566

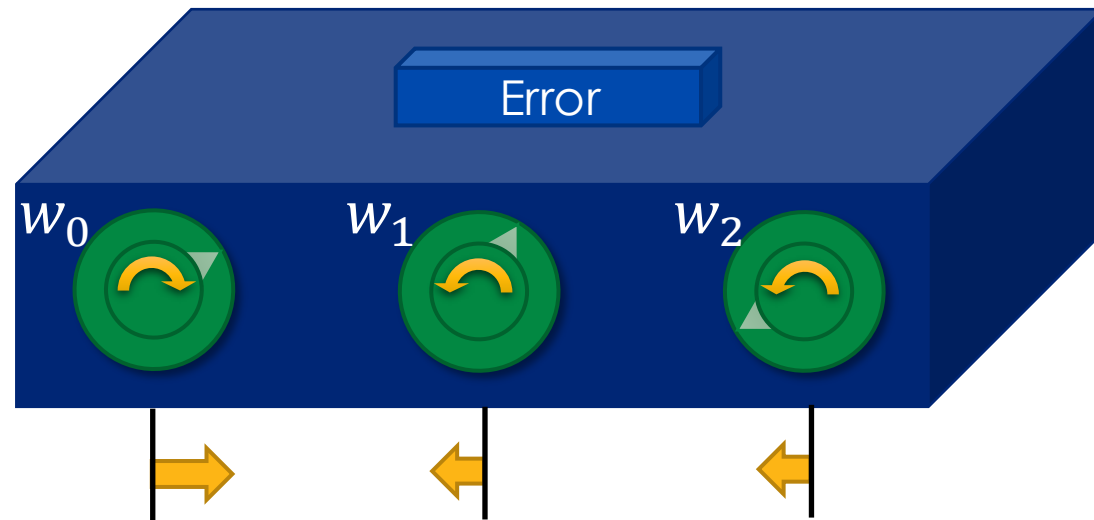


Try the new [loss_game](#).

Intuition



1136566



What if we knew which way to turn the knob
and an idea of how far?

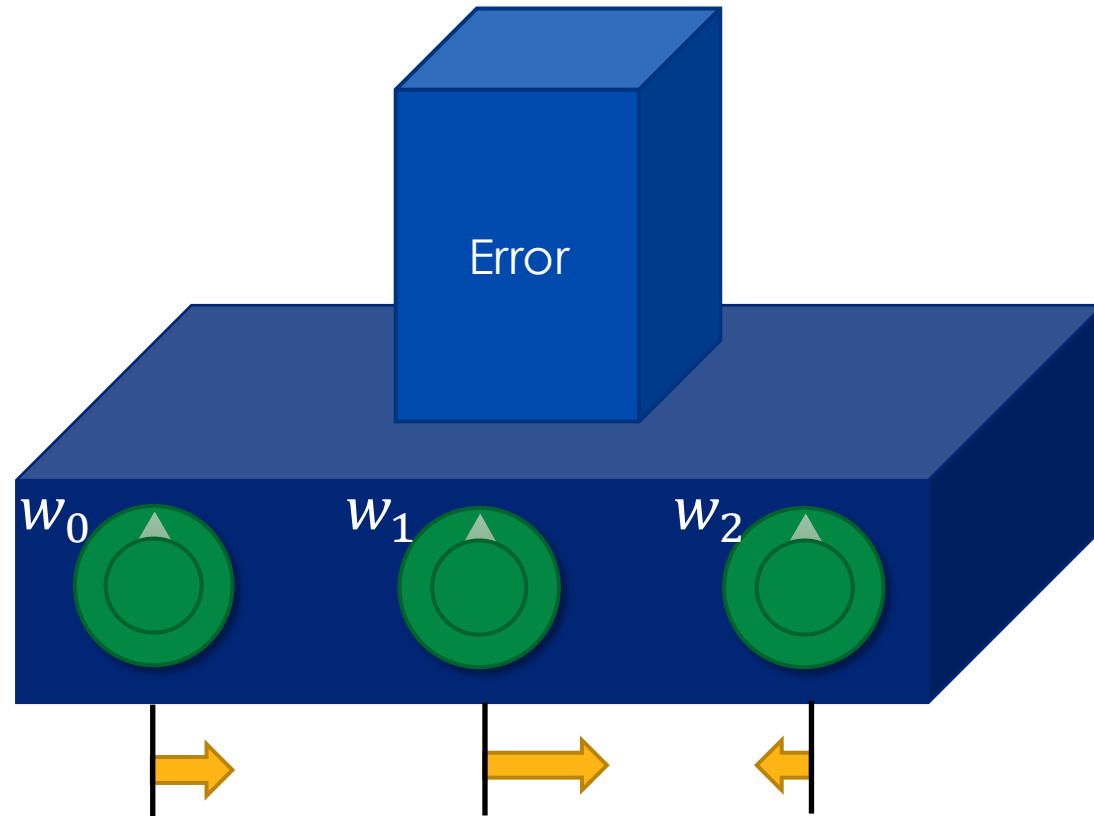
This is the Gradient!

Try the new [loss_game](#).

Intuition



1136566

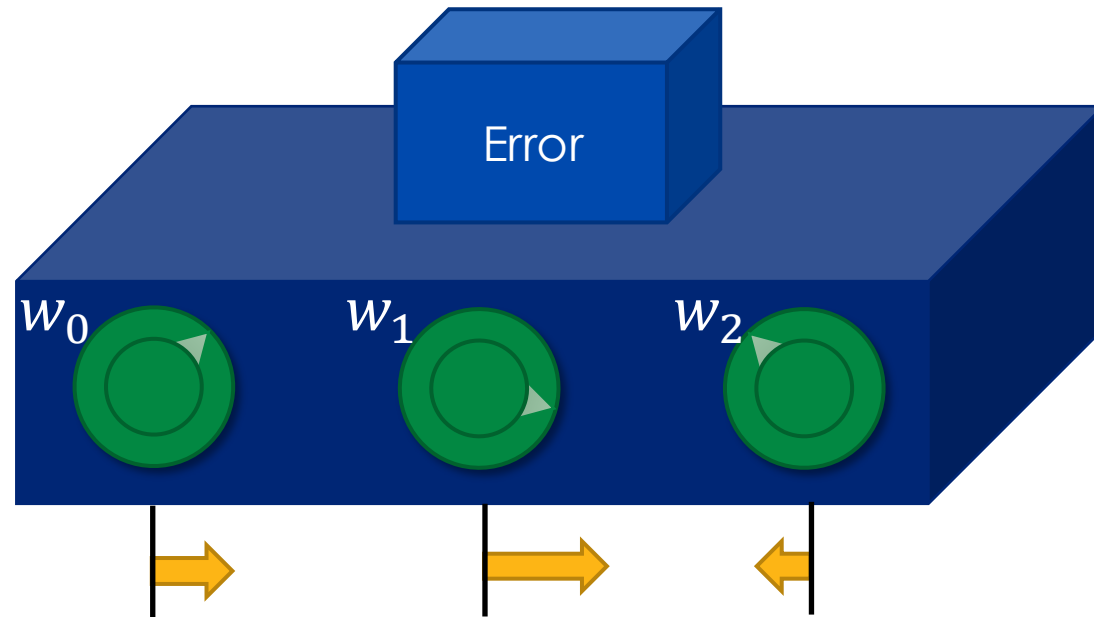


Try the new [loss game](#).

Intuition



1136566

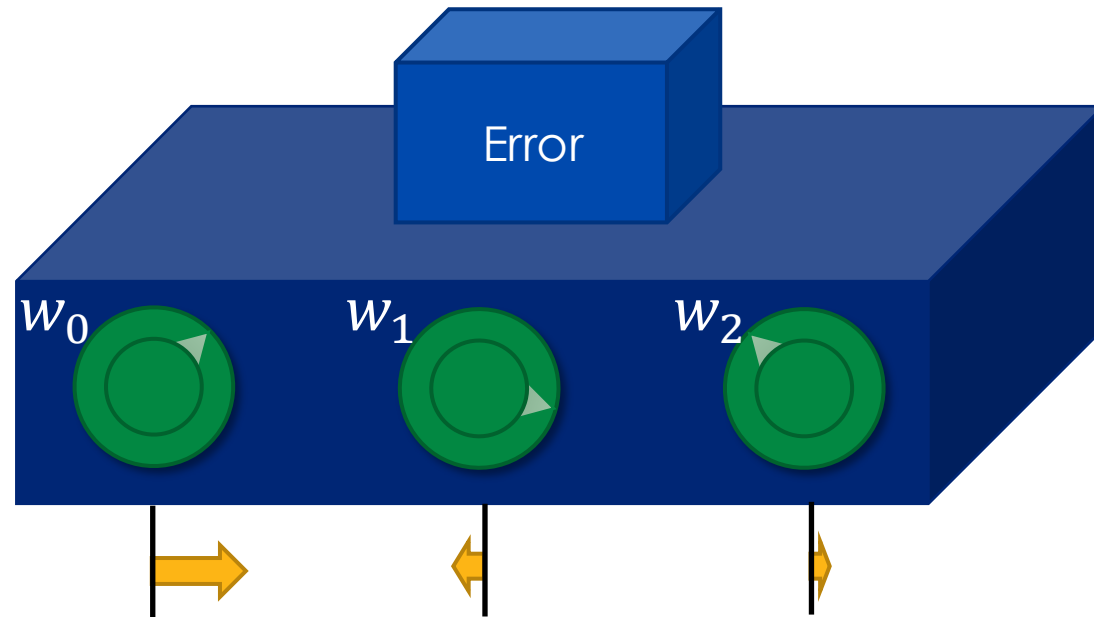


Try the new [loss_game](#).

Intuition



1136566

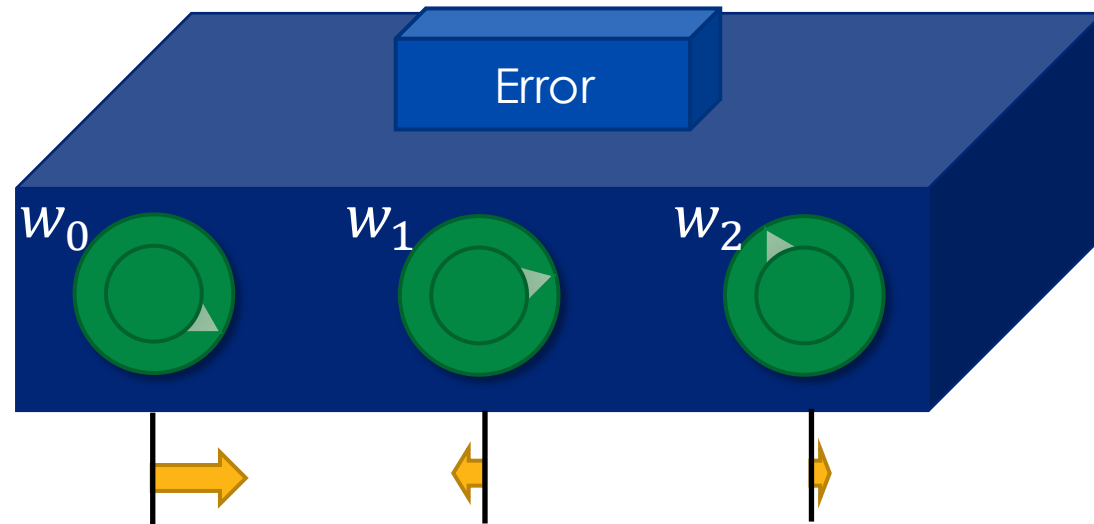


Try the new [loss_game](#).

Intuition



1136566

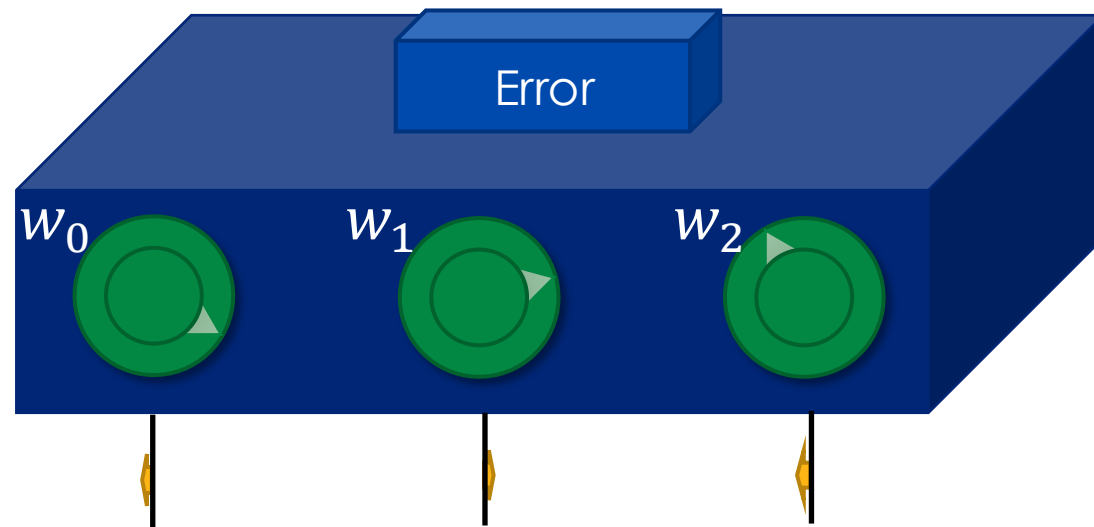


Try the new [loss_game](#).

Intuition



1136566



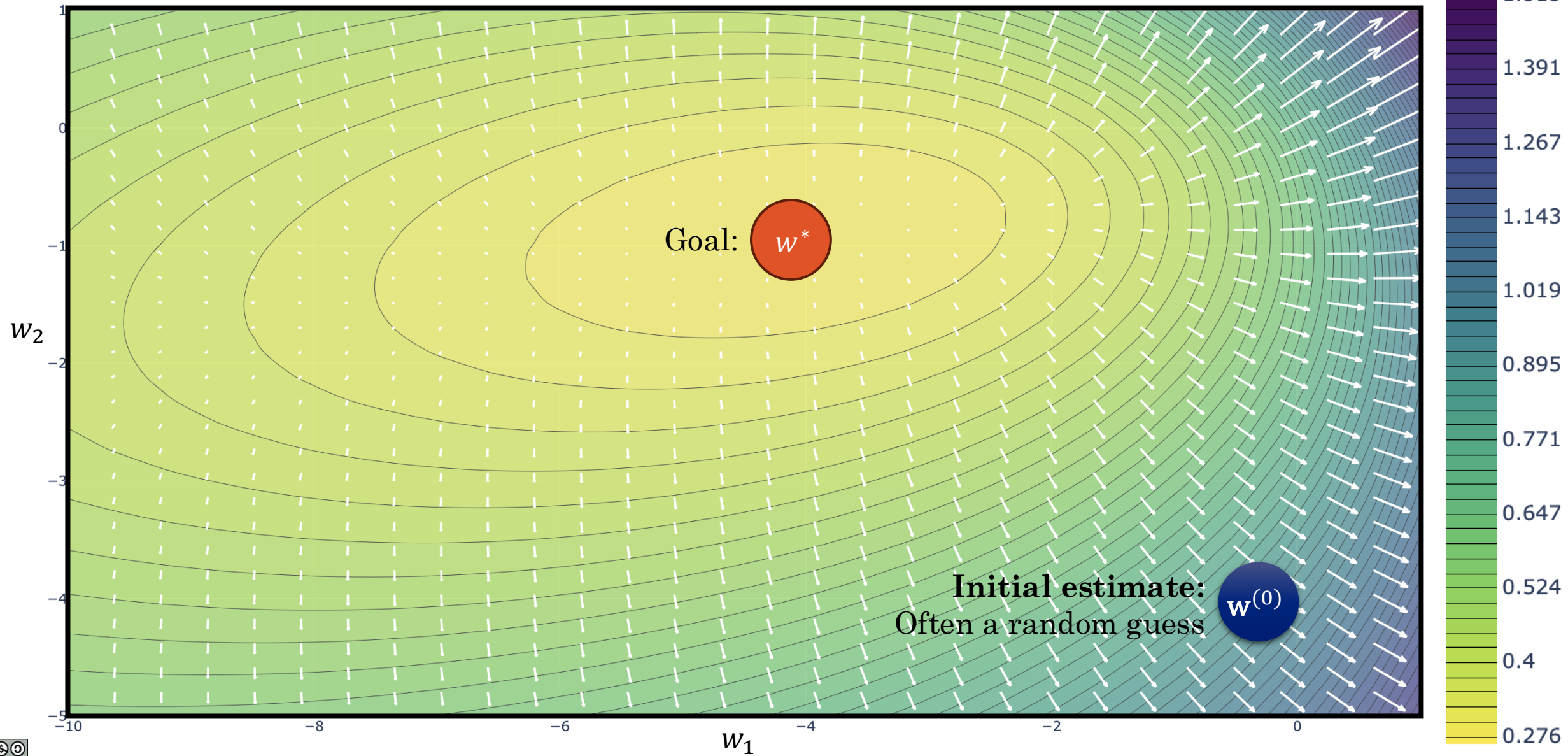
This is Gradient descent!

Try the new [loss_game](#).

Gradient Descent Intuition



1136566

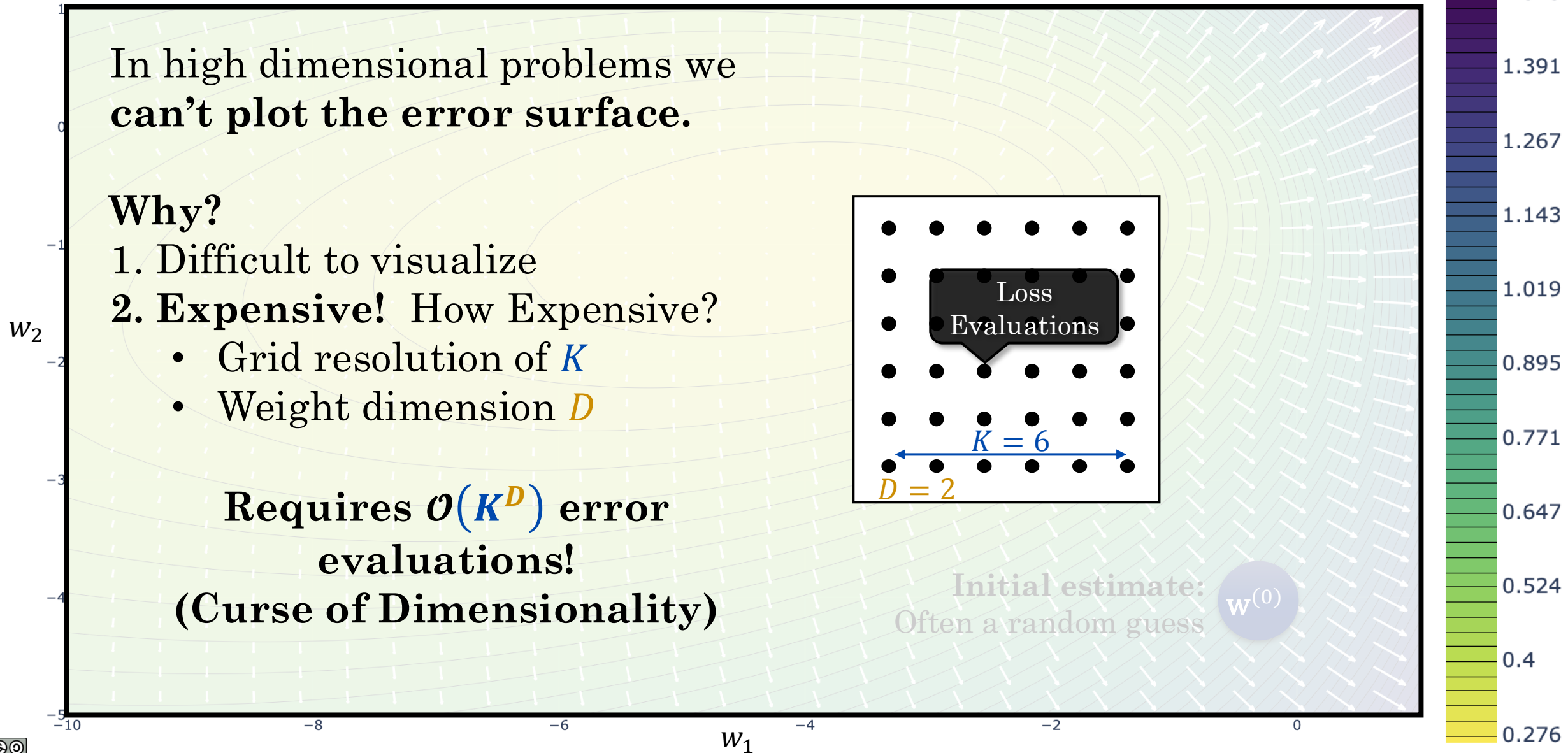


Gradient Descent Intuition



1136566

Error

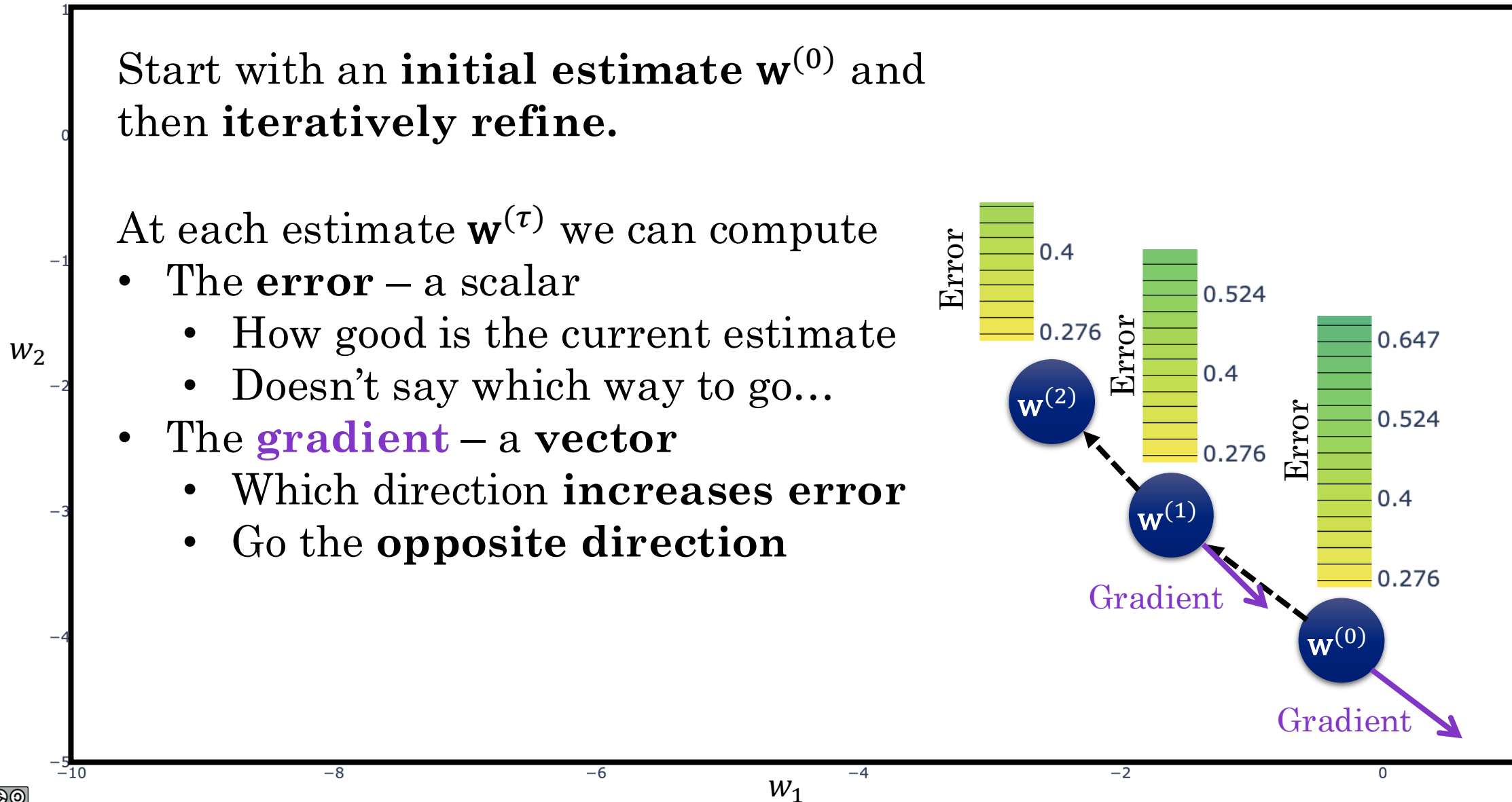
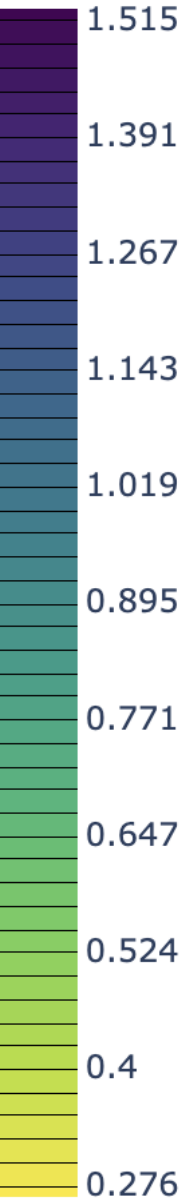


Gradient Descent Intuition



1136566

Error



Start with an **initial estimate** $w^{(0)}$ and then **iteratively refine**.

At each estimate $w^{(\tau)}$ we can compute

- The **error** – a scalar
 - How good is the current estimate
 - Doesn't say which way to go...
- The **gradient** – a **vector**
 - Which direction **increases error**
 - Go the **opposite direction**



1136566

The (Batch) Gradient Descent Alg.

Update the weights by moving in the **opposite direction** of the gradient:

```
 $\mathbf{w}^{(0)} = \text{Initialize}()$   
for  $\tau$  in range(1,  $\tau_{\text{max}}$ ):  
     $\mathbf{w}^{(\tau)} = \mathbf{w}^{(\tau-1)} - \eta \nabla E(\mathbf{w}^{(\tau-1)})$   
    if  $\text{Converged}(\mathbf{w}^{(\tau)}, \mathbf{w}^{(\tau-1)}, \epsilon)$ : terminate
```

$\text{Initialize}()$ typically a small random value (more on this later)

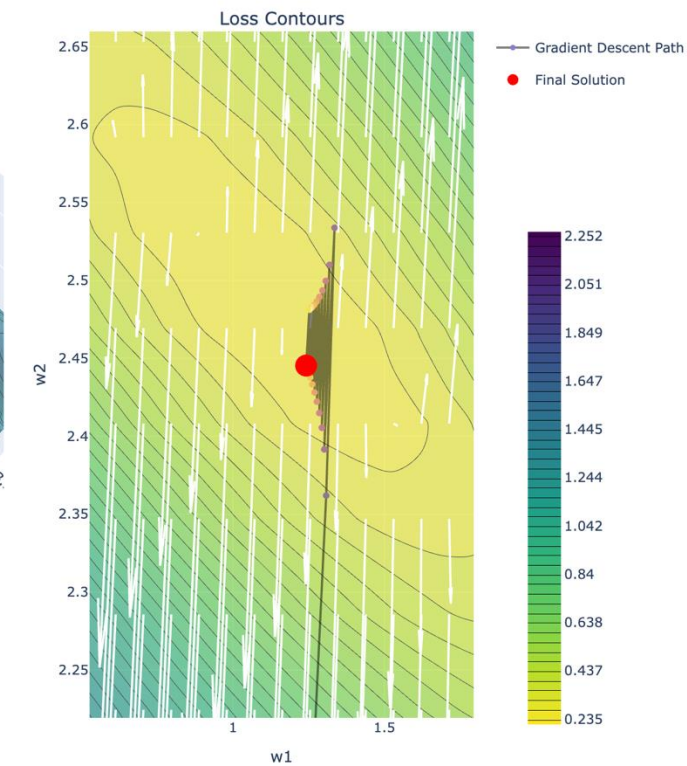
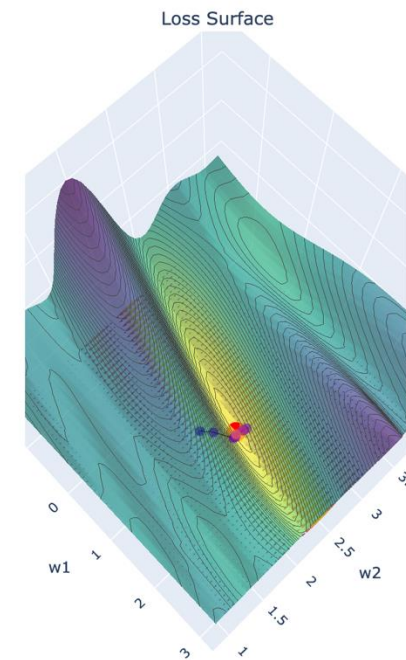
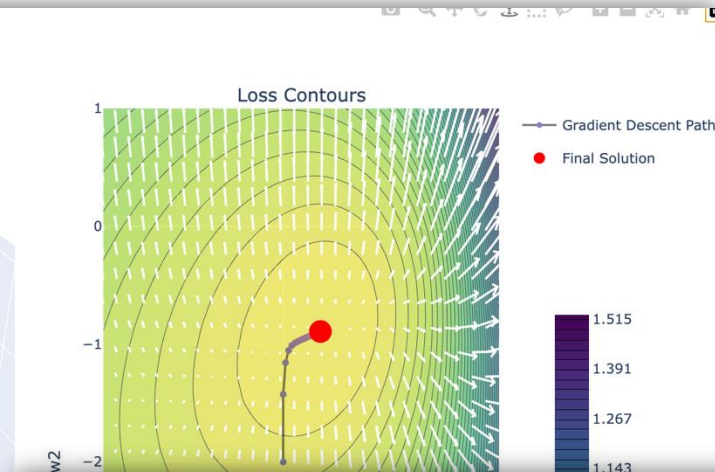
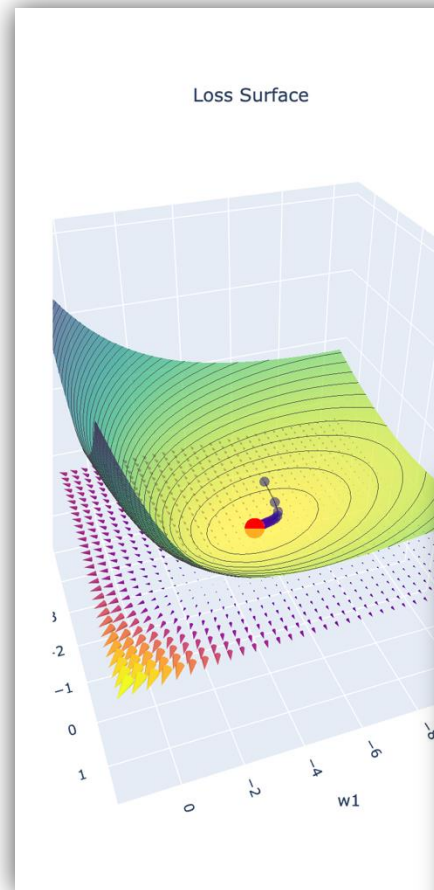
$\text{Converged}(\mathbf{w}^{(\tau)}, \mathbf{w}^{(\tau-1)}, \epsilon)$ stopping condition (e.g., $\|\Delta \mathbf{w}^{(\tau-1)}\| < \epsilon$)

Hyper Parameters:

- η is the learning rate. A common starting range is $0 < \eta < 1$, but the stable range depends on the curvature of E (next lecture).
- τ_{max} is the maximum number of iterations

Demo

Batch Gradient Descent



1136566



1136566

Loss (Error) Curves

The **loss** or **error curve** plots the error as a function of the number of **iterations (steps)** of gradient descent.

Training and Validation Error

Validation curves are typically **above** the training curves.

You can **overoptimize** the objective (**overfit**).

Choose parameters with **lowest validation error**.



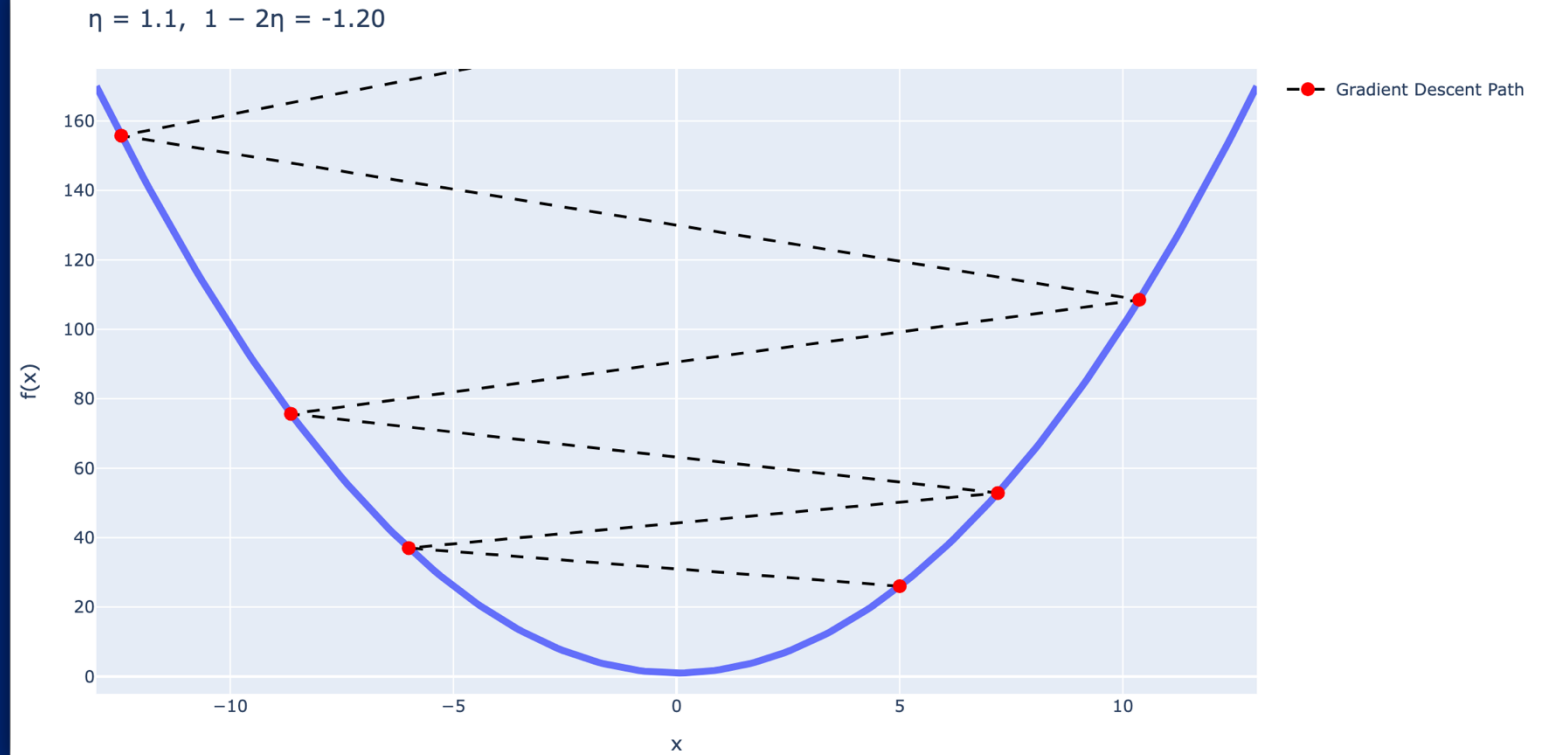


Assuming a simple quadratic equation (e.g., x^2) is gradient descent guaranteed to converge?



Demo

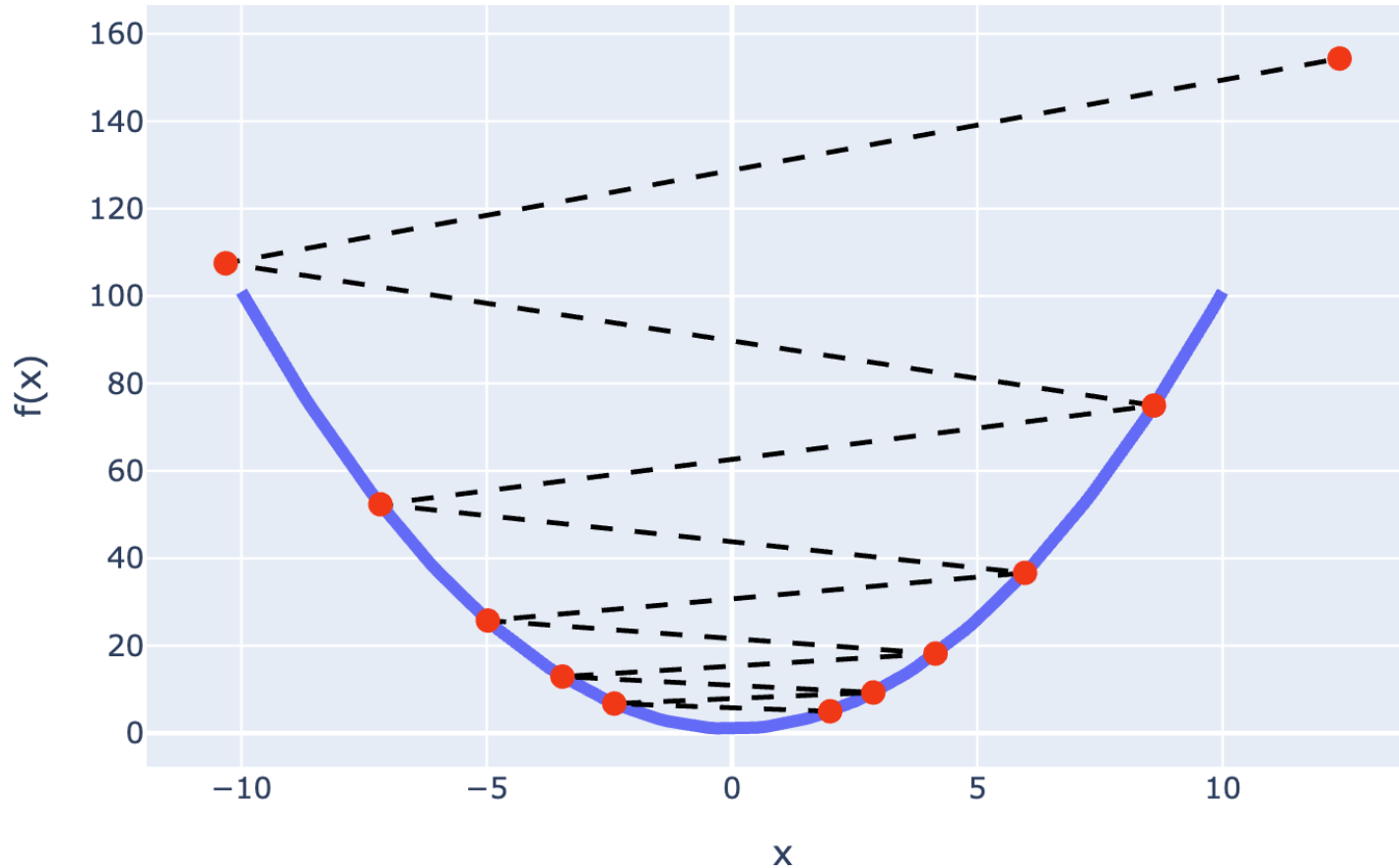
Gradient Descent on a Parabola



What are the conditions for convergence?



1136566



—●— Gradient Descent Path

Learning rate
seemed to affect
convergence even on
a simple parabola.

Convergence Assuming Quadratic Approximation



1136566

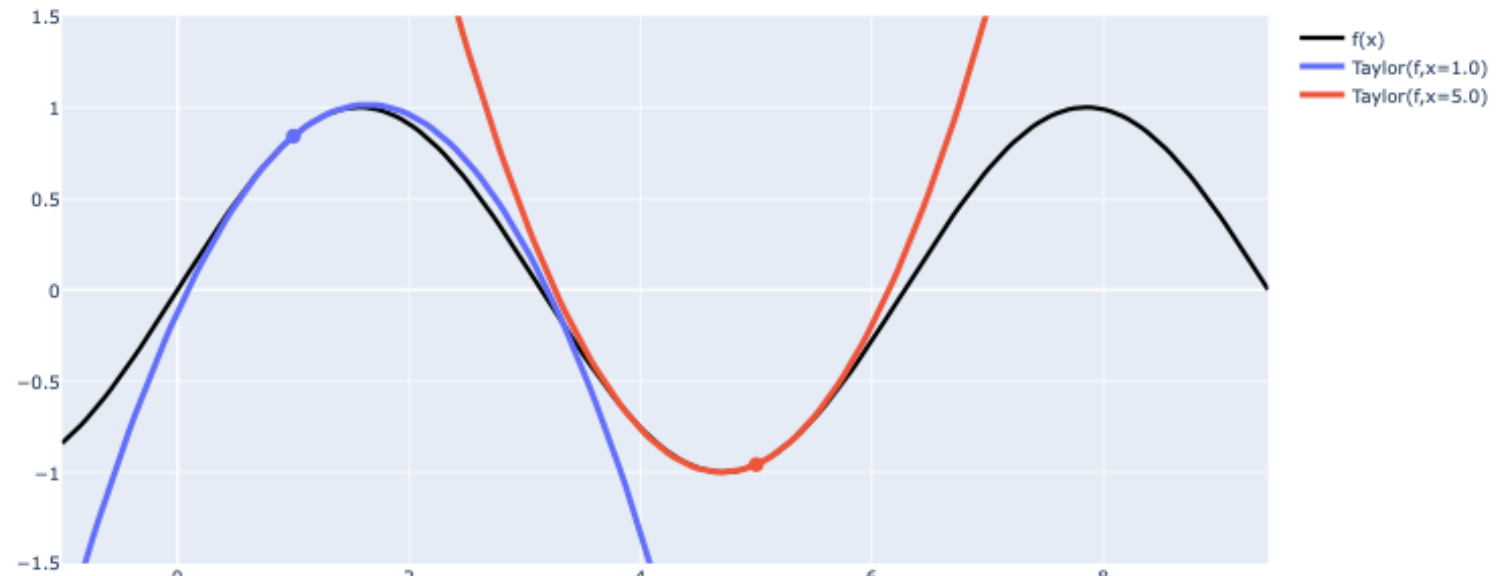
Quadratic Approximations

The 2nd order (quadratic) Taylor approximation of a function $f: \mathbb{R} \rightarrow \mathbb{R}$ evaluated at $\hat{x}$ is:

$$\tilde{f}(x) = f(\hat{x}) + \left(\frac{\partial}{\partial x} f(x) \Big|_{x=\hat{x}} \right) (x - \hat{x}) + \frac{1}{2} \left(\frac{\partial^2}{\partial x^2} f(x) \Big|_{x=\hat{x}} \right) (x - \hat{x})^2$$

For example: $f(x) = \sin(x)$

$$\begin{aligned} \tilde{f}_{\hat{x}=1}(x) &= \sin(1) + \cos(1)(x - 1) \\ &\quad + \frac{1}{2}(-\sin(1))(x - 1)^2 \\ \tilde{f}_{\hat{x}=5}(x) &= \sin(5) + \cos(5)(x - 5) \\ &\quad + \frac{1}{2}(-\sin(5))(x - 5)^2 \end{aligned}$$





Taylor Expansion of the Error Surface

The 2nd-order Taylor expansion of the **error surface** at $\hat{w}$:

$$\tilde{E}_{\hat{w}}(w) = E(\hat{w}) + (w - \hat{w})^T b + \frac{1}{2} (w - \hat{w})^T H (w - \hat{w})$$

- $b = \nabla E(\hat{w})$ is the **Gradient** of E evaluated at $\hat{w}$ (vector)
- $H = \nabla \nabla E(\hat{w})$ is the **Hessian** of E evaluated at $\hat{w}$ (a matrix)

Matrix Equivalent of the
Second Derivative



1136566

Hessians (the 2nd derivative)

The Hessian of a **scalar-valued function** $E: \mathbb{R}^D \rightarrow \mathbb{R}$ is the **matrix-valued function** $\nabla^2 E: \mathbb{R}^D \rightarrow \mathbb{R}^{D \times D}$ of the 2nd partial derivatives of E

$$H = \nabla^2 E = \begin{bmatrix} \frac{\partial^2}{\partial w_1^2} E(w) & \cdots & \frac{\partial^2}{\partial w_1 \partial w_D} E(w) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial w_D \partial w_1} E(w) & \cdots & \frac{\partial^2}{\partial w_D^2} E(w) \end{bmatrix}$$

The Hessian

- describes the local **curvature** of E
- determines the **stationary point type** (minimum, maximum, or saddle point)

Hessian Exercise

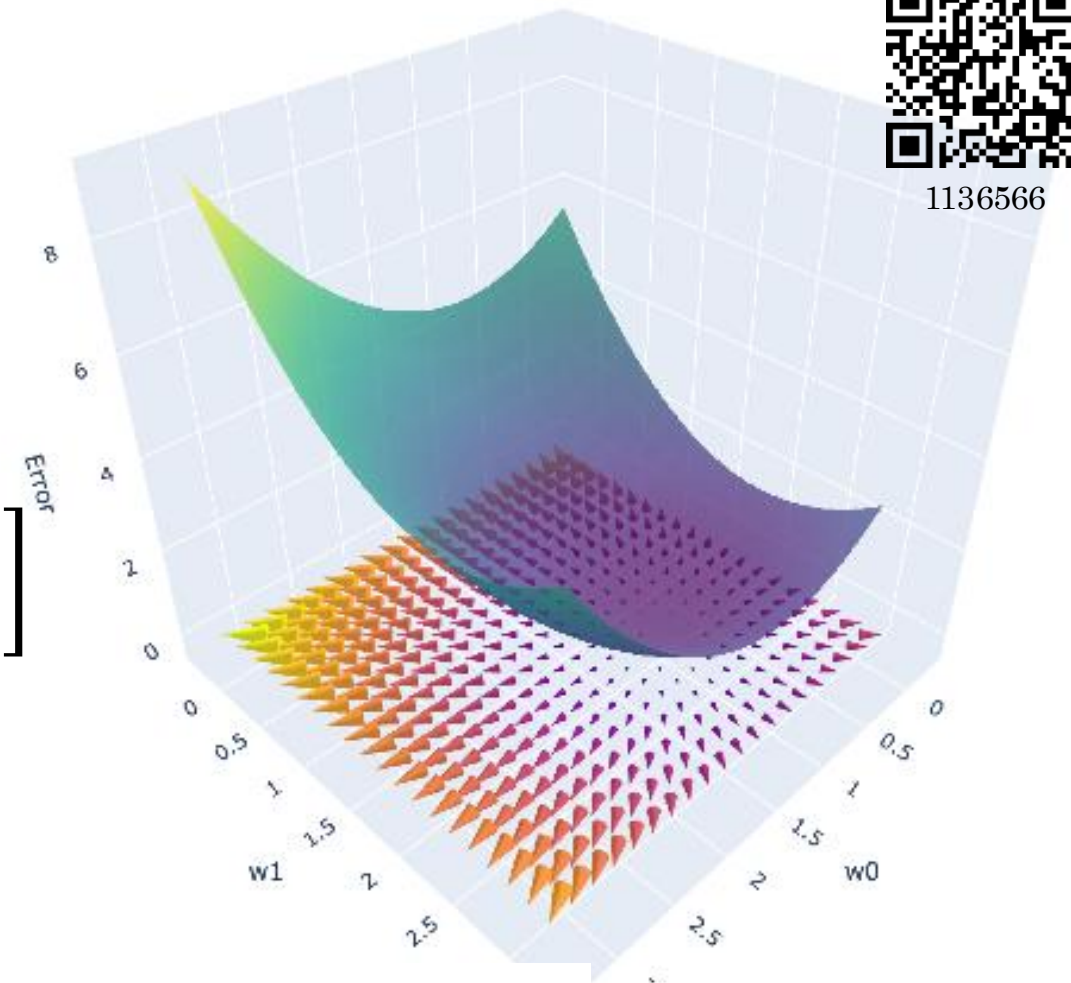
Consider the function:

$$E(w) = (w_0 - 1)^2 + (w_1 - 2)^2 + 1$$

Then the gradient is: $\nabla E(w) = \begin{bmatrix} 2(w_0 - 1) \\ 2(w_1 - 2) \end{bmatrix}$

The Hessian is:

$$H(w) = \begin{bmatrix} \frac{\partial^2}{\partial w_0^2} E(w) & \frac{\partial^2}{\partial w_0 \partial w_1} E(w) \\ \frac{\partial^2}{\partial w_0 \partial w_1} E(w) & \frac{\partial^2}{\partial w_1^2} E(w) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$



1136566

Hessian

$$\nabla^2 E = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} E(x) & \dots & \frac{\partial^2}{\partial x_1 \partial x_D} E(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial x_D \partial x_1} E(x) & \dots & \frac{\partial^2}{\partial x_D^2} E(x) \end{bmatrix}$$



Taylor Expansion of the Error Surface

The 2nd-order Taylor expansion of the error surface at $\hat{w}$:

$$\tilde{E}_{\hat{w}}(w) = E(\hat{w}) + (w - \hat{w})^T b + \frac{1}{2} (w - \hat{w})^T H (w - \hat{w})$$

- $b = \nabla E(\hat{w})$ is the **Gradient** of E evaluated at $\hat{w}$ (vector)
- $H = \nabla \nabla E(\hat{w})$ is the **Hessian** of E evaluated at $\hat{w}$ (a matrix)

If $\hat{w}$ is a **stationary point** (w^*) then $\nabla E(w^*) = 0 = b$

$$\tilde{E}_{w^*}(w) = E(w^*) + \frac{1}{2} (w - w^*)^T H (w - w^*)$$

Use a change of variables to better understand this part.



Eigen Decomposition of the Hessian

We can construct the **Eigen decomposition of the Hessian**

$$Hu_i = \lambda_i u_i$$

- u_i are the complete set of **orthonormal eigenvectors**: $\forall_{i,j}: u_i^T u_j = \delta_{ij}$
- λ_i are the real-valued **eigenvalues**

Then we apply a **change of variables**:

$$w - w^* = \sum_i \alpha_i u_i \quad \text{where} \quad \alpha_i = (w - w^*)^T u_i$$

This allows us to rewrite:

$$\tilde{E}_{w^*}(w) = E(w^*) + \frac{1}{2} (w - w^*)^T H (w - w^*) = E(w^*) + \frac{1}{2} \sum_i \alpha_i^2 \lambda_i$$

Show it!



1136566

- u_i are the complete set of **orthonormal eigenvectors**: $\forall_{i,j}: u_i^T u_j = \delta_{ij}$
- λ_i are the real-valued **eigenvalues**

Then we apply a **change of variables**:

$$w - w^* = \sum_i \alpha_i u_i \quad \text{where} \quad \alpha_i = (w - w^*)^T u_i$$

This allows us to rewrite:

$$\tilde{E}_{w^*}(w) = E(w^*) + \frac{1}{2} (w - w^*)^T H (w - w^*) = E(w^*) + \frac{1}{2} \sum_i \alpha_i^2 \lambda_i$$

Show it!

Derivation:

$$\begin{aligned} (w - w^*)^T H (w - w^*) &= \left(\sum_i \alpha_i u_i \right)^T H \left(\sum_j \alpha_j u_j \right) = \left(\sum_i \alpha_i u_i \right)^T \left(\sum_j \alpha_j H u_j \right) \\ &= \left(\sum_i \alpha_i u_i \right)^T \left(\sum_j \alpha_j \lambda_j u_j \right) = \sum_{ij} \alpha_i \alpha_j u_i^T u_j \lambda_j = \sum_{ij} \alpha_i \alpha_j \delta_{ij} \lambda_j = \sum_i \alpha_i^2 \lambda_i \end{aligned}$$

Eigenvector Defn.

Orthonormality



1136566

Taylor Expansion at the Stationary Pts.

If $\hat{w}$ is a **stationary point** (w^*) (where $b = \nabla E|_{w=\hat{w}} = 0$)

$$\tilde{E}_{w^*}(\alpha) = E(w^*) + \frac{1}{2} \sum_i \alpha_i^2 \lambda_i$$

At $\alpha = 0$ we get the error at the stationary point $E(w^*)$.

- If we move in **direction i** (corresponding to u_i) then α_i^2 **increases**.
 - The change in error then depends on the sign of λ_i

Therefore, if the **eigenvalues are**

- **Slido!**



If the eigen values at w^* are all positive then



Taylor Expansion at the Stationary Pts.

If $\hat{w}$ is a **stationary point** (w^*) (where $b = \nabla E|_{w=\hat{w}} = 0$)

$$\tilde{E}_{w^*}(\alpha) = E(w^*) + \frac{1}{2} \sum_i \alpha_i^2 \lambda_i$$

At $\alpha = 0$ we get the error at the stationary point $E(w^*)$.

- If we move in **direction** i (corresponding to u_i) then α_i^2 **increases**.
 - The change in error then depends on the sign of λ_i

Therefore, if the **eigenvalues** are

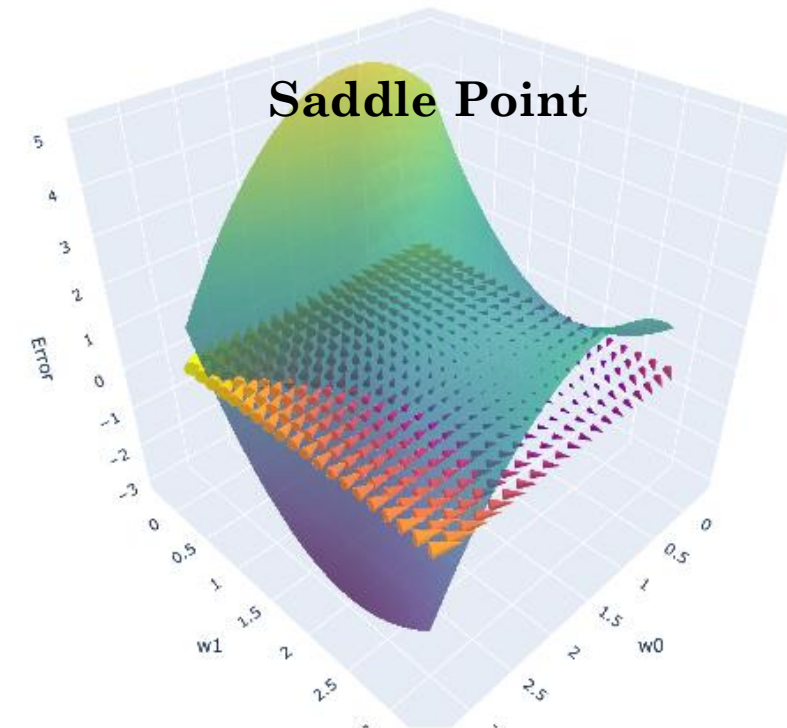
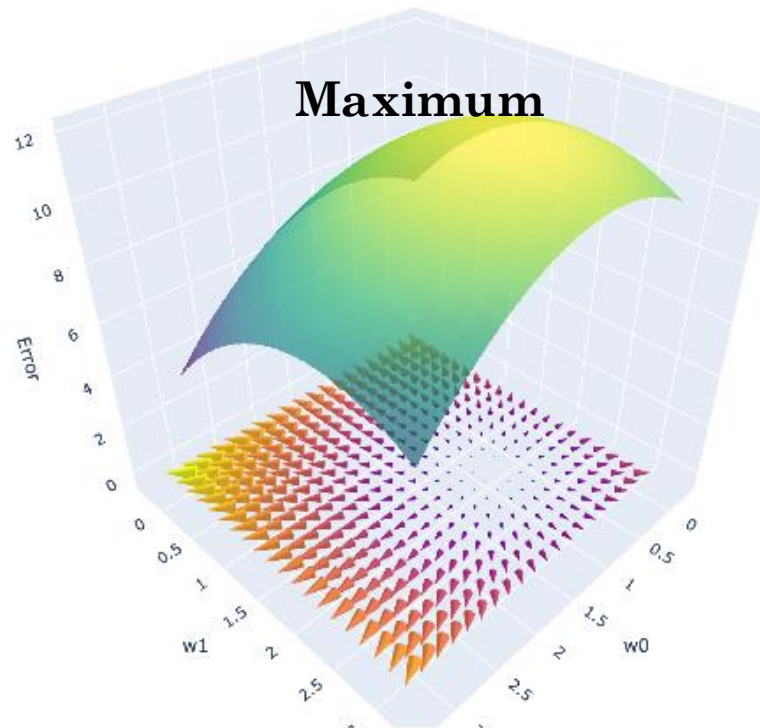
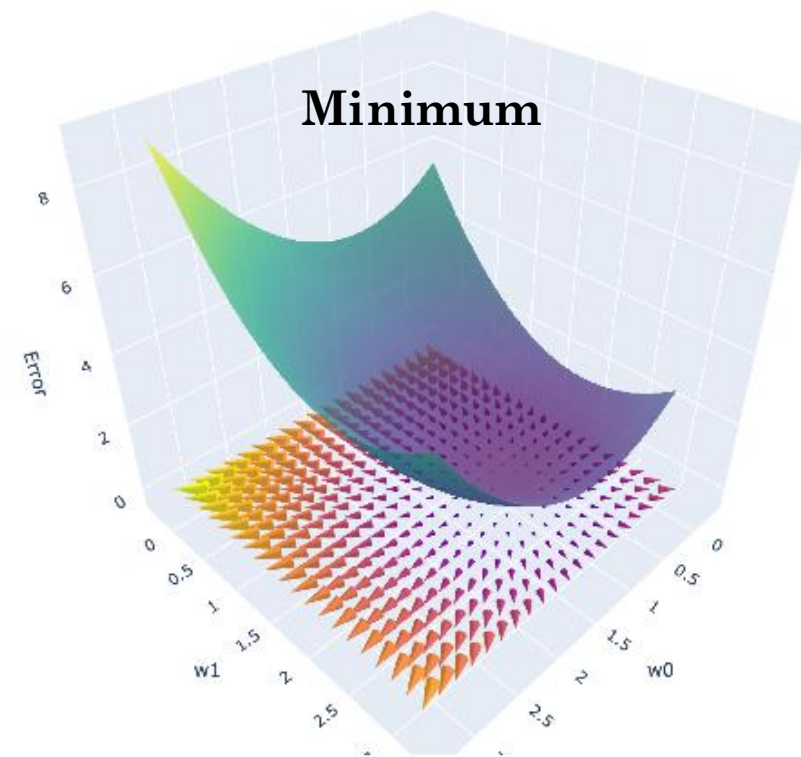
- **all positive** then w^* is a **local minimum**.
- **all negative** then w^* is a **local maximum**.
- **mixed** then w^* is a **saddle point**.

Stationary Points and the Hessian



1136566

Eigenvalues of the Hessian determine the curvature at the stationary points



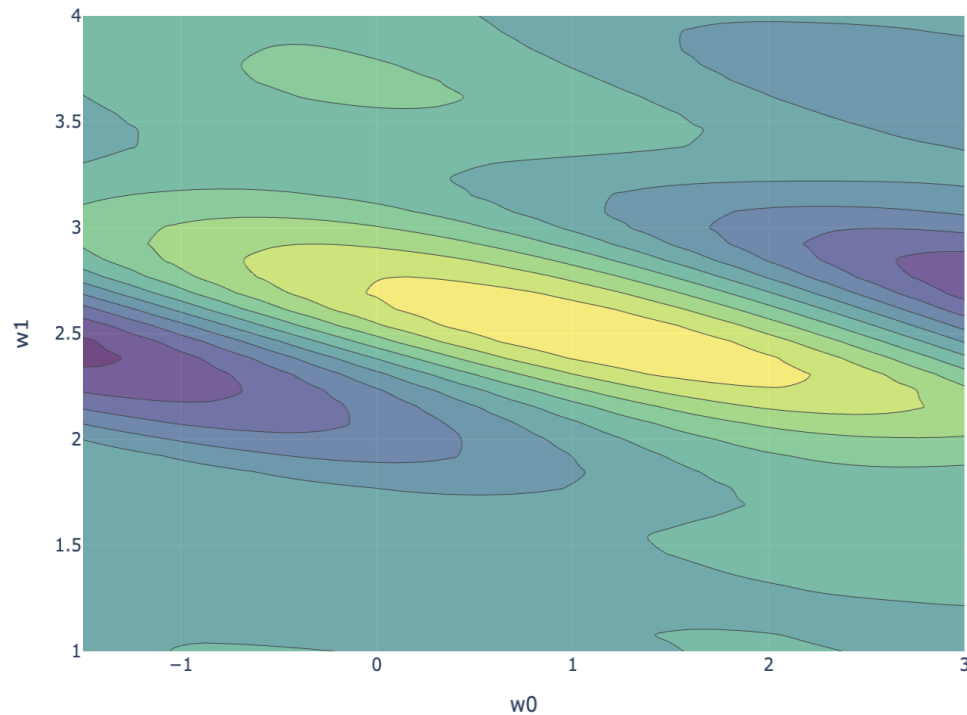
Eigenvectors of the Hessian



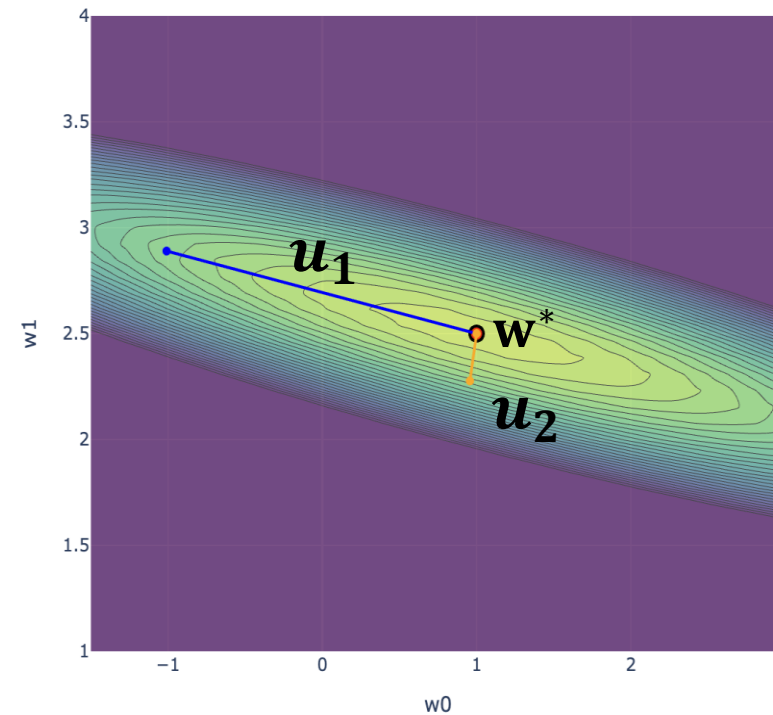
1136566

Elliptical contours of constant error align with the eigenvectors of the Hessian matrix.

Loss Surface



Taylor Approximation



Plots in demo notebook.



**The eigenvector u_1
corresponds to the:**

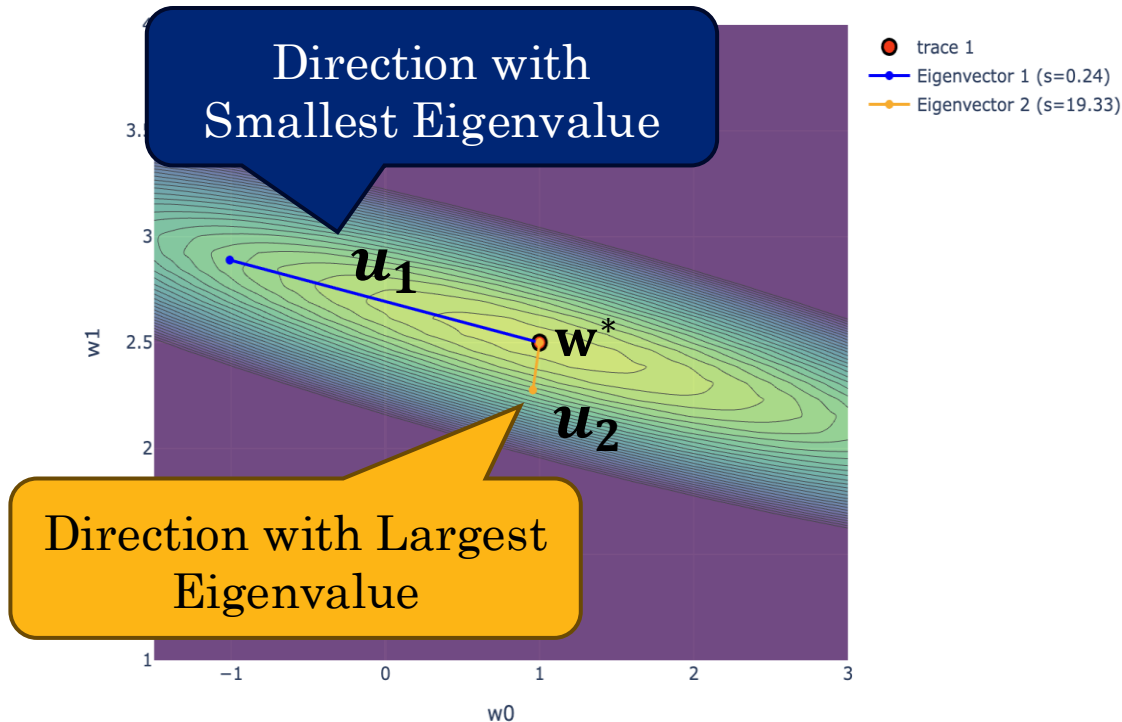


1136566

Eigenvectors of the Hessian

Elliptical contours of constant error align with the eigenvectors of the Hessian matrix.

Taylor Approximation



If we plot the vectors so length represents a constant c increase in error and

$$\tilde{E}_{w^*}(\alpha) = E(w^*) + \frac{1}{2} \sum_i \alpha_i^2 \lambda_i$$

Then we would want:

$$\frac{1}{2} \alpha_i^2 \lambda_i = c \Rightarrow \alpha_i = \sqrt{\frac{2c}{\lambda_i}}$$

lengths proportional to $\sqrt{1/\lambda_i}$

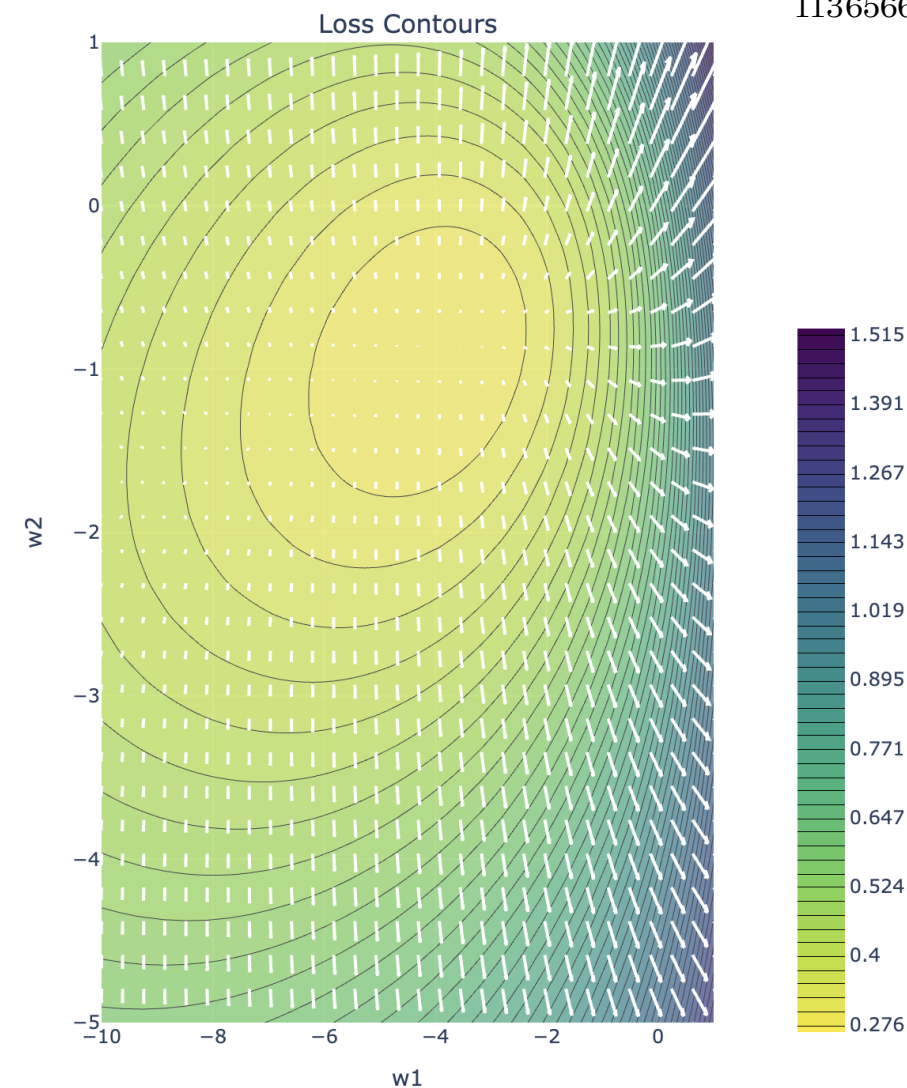
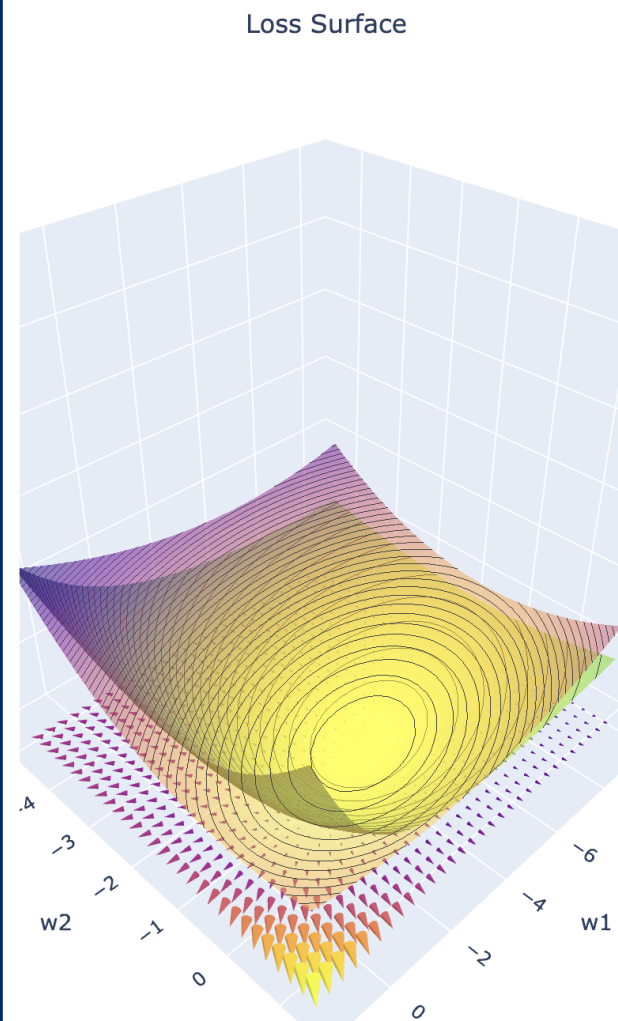
Plots in demo notebook.

Optional Demo

Taylor Expansions and the Hessian



1136566



Lecture 10

Optimization and Gradient Descent

Credit: Joseph E. Gonzalez and Narges Norouzi

Reference Book Chapters: Chapter 7