Lecture 10

Feature Engineering, Gradient Descent

DSC 40A, Summer 2024

Announcements

- Homework 4 is due **tonight**.
  - *Please* remember to select pages in your Gradescope submission.
  - We're going to start penalizing for submissions without pages selected.

The Midterm Exam is on Thursday, August 22nd!

- The Midterm Exam is on **Thursday, August 22nd in class**.
- 80 minutes, on paper, no calculators or electronics.
  - **You are allowed to bring one two-sided index card (4 inches by 6 inches) of notes that you write by hand (no iPad).**
- Content: Lectures 1-9, Homeworks 1-4, Groupworks 1-3.
- Prepare by practicing with old exam problems at practice.dsc40a.com.
  - Problems are sorted by topic!
  - Come by office hours to review.
  - Nishant holds OH this afternoon, Jack tomorrow AM virtually.
- Some time for review in discussion tomorrow.

Agenda

- Feature engineering and transformations.
- Minimizing functions using gradient descent.

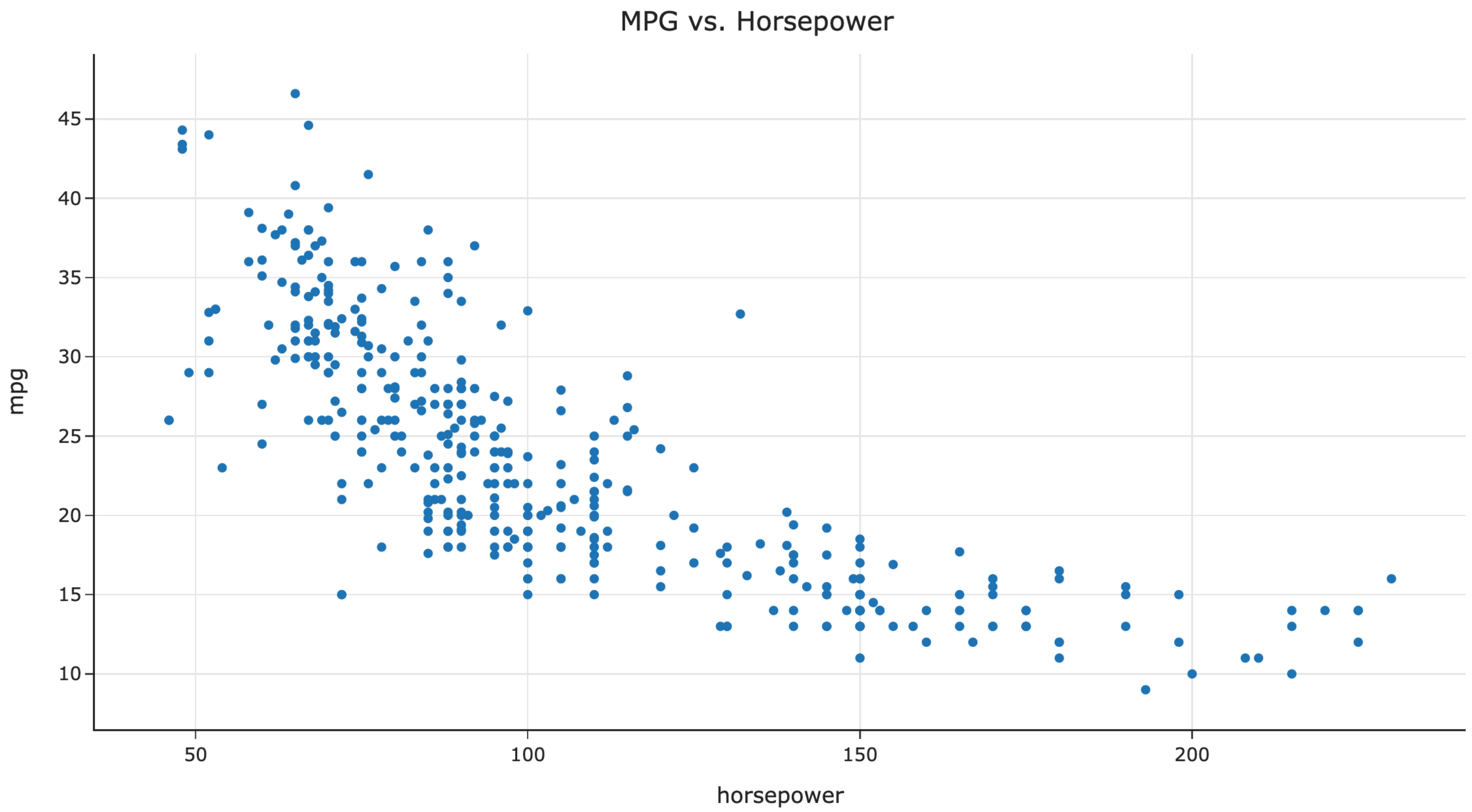
Question 🤔

Answer at q.dsc40a.com

Remember, you can always ask questions at q.dsc40a.com!

If the direct link doesn't work, click the "🤔 Lecture Questions" link in the top right corner of dsc40a.com.

Feature engineering and transformations



Question: Would a linear hypothesis function work well on this dataset?

Linear in the parameters

- We can fit rules like:

$$w_0 + w_1 x + w_2 x^2 \qquad w_1 e^{-{x^{(1)}}^2} + w_2 \cos(x^{(2)} + \pi) + w_3 \frac{\log 2x^{(3)}}{x^{(2)}}$$

  - This includes arbitrary polynomials.
  - These are all linear combinations of (just) features.

- We can't fit rules like:

$$w_0 + e^{w_1 x} \qquad w_0 + \sin(w_1 x^{(1)} + w_2 x^{(2)})$$

  - These are **not** linear combinations of just features!

- We can have any number of parameters, as long as our hypothesis function is **linear in the parameters**, or linear when we think of it as a function of the parameters.

Example: Amdahl's Law

- Amdahl's Law relates the runtime of a program on p processors to the time to do the sequential and nonsequential parts on one processor.

$$H(p) = t_{\text{S}} + \frac{t_{\text{NS}}}{p}$$

- Collect data by timing a program with varying numbers of processors:

Processors	Time (Hours)
1	8
2	4
4	3

Example: Fitting $H(x) = w_0 + w_1 \cdot \frac{1}{x}$

Processors	Time (Hours)
1	8
2	4
4	3

How do we fit hypothesis functions that aren't linear in the parameters?

- Suppose we want to fit the hypothesis function:

$$H(x) = w_0 e^{w_1 x}$$

- This is **not** linear in terms of w_0 and w_1, so our results for linear regression don't apply.
- **Possible solution**: Try to apply a **transformation**.

Transformations

- **Question**: Can we re-write $H(x) = w_0 e^{w_1 x}$ as a hypothesis function that **is** linear in the parameters?

Transformations

- **Solution**: Create a new hypothesis function, $T(x)$, with parameters b_0 and b_1, where $T(x) = b_0 + b_1 x$.
- This hypothesis function is related to $H(x)$ by the relationship $T(x) = \log H(x)$.
- $\vec{b}$ is related to $\vec{w}$ by $b_0 = \log w_0$ and $b_1 = w_1$.
- Our new observation vector, $\vec{z}$, is $\begin{bmatrix} \log y_1 \\ \log y_2 \\ \dots \\ \log y_n \end{bmatrix}$.
- $T(x) = b_0 + b_1 x$ is linear in its parameters, b_0 and b_1.
- Use the solution to the normal equations to find $\vec{b}^*$, and the relationship between $\vec{b}$ and $\vec{w}$ to find $\vec{w}^*$.

Once again, let's try it out! Follow along in this notebook.

Non-linear hypothesis functions in general

- Sometimes, it's just not possible to transform a hypothesis function to be linear in terms of some parameters.
- In those cases, you'd have to resort to other methods of finding the optimal parameters.
    - For example, $H(x) = w_0 \sin(w_1 x)$ **can't** be transformed to be linear.
    - But, there are other methods of minimizing mean squared error:

$$R_{\text{sq}}(w_0, w_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - w_0 \sin(w_1 x))^2$$

    - One method: **gradient descent**, the topic we're going to look at next!
- Hypothesis functions that are linear in the parameters are much easier to work with.

Question 🤔

Answer at q.dsc40a.com

Which hypothesis function is **not** linear in the parameters?

- A. $H(\vec{x}) = w_1(x^{(1)}x^{(2)}) + \frac{w_2}{x^{(1)}}\sin\left(x^{(2)}\right)$
- B. $H(\vec{x}) = 2^{w_1}x^{(1)}$
- C. $H(\vec{x}) = \vec{w} \cdot \text{Aug}(\vec{x})$
- D. $H(\vec{x}) = w_1 \cos(x^{(1)}) + w_2 2^{x^{(2)} \log x^{(3)}}$
- E. More than one of the above.

Roadmap

- This is the end of the content that's in scope for the Midterm Exam.
- Now, we'll introduce **gradient descent**, a technique for minimizing functions that can't be minimized directly using calculus or linear algebra.
- After the Midterm Exam, we'll switch gears to **probability**.

The modeling recipe

1. Choose a model.

2. Choose a loss function.

3. Minimize average loss to find optimal model parameters.

Minimizing functions using gradient descent

Minimizing empirical risk

- Repeatedly, we've been tasked with **minimizing** the value of empirical risk functions.
    - Why? To help us find the **best** model parameters, h^* or w^*, which help us make the **best** predictions!
- We've minimized empirical risk functions in various ways.
    - $R_{\text{sq}}(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$
    - $R_{\text{abs}}(w_0, w_1) = \frac{1}{n}\sum_{i=1}^{n}|y_i - (w_0 + w_1 x)|$
    - $R_{\text{sq}}(\vec{w}) = \frac{1}{n}\|\vec{y} - X\vec{w}\|^2$

Minimizing arbitrary functions

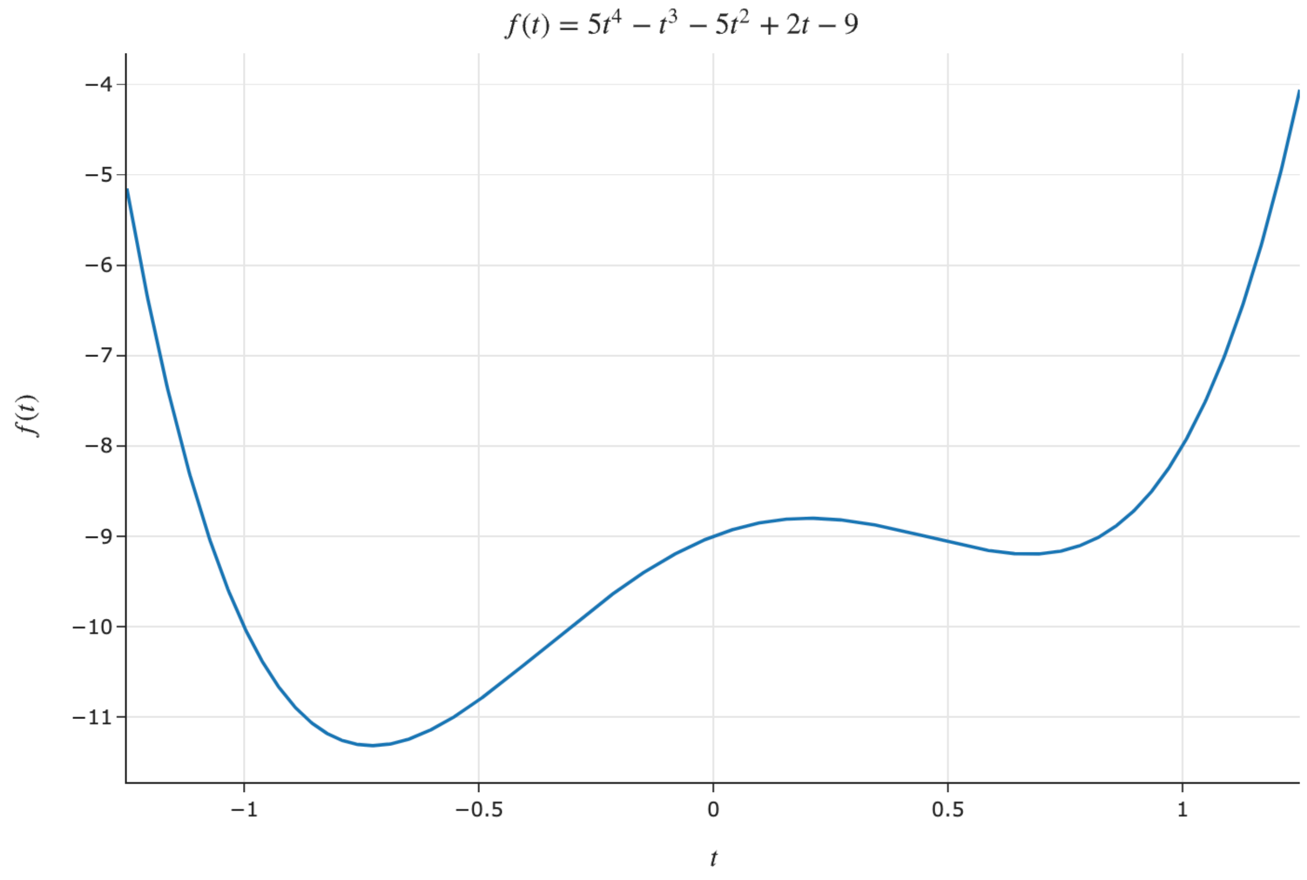
- Assume $f(t)$ is some **differentiable** single-variable function.
- When tasked with minimizing $f(t)$, our general strategy has been to:
  i. Find $\frac{df}{dt}(t)$, the derivative of f.
  ii. Find the input t^* such that $\frac{df}{dt}(t^*) = 0$.
- However, there are cases where we can find $\frac{df}{dt}(t)$, but **it is either difficult or impossible to solve** $\frac{df}{dt}(t^*) = 0$.

$$f(t) = 5t^4 - t^3 - 5t^2 + 2t - 9$$

- Then what?

What does the derivative of a function tell us?

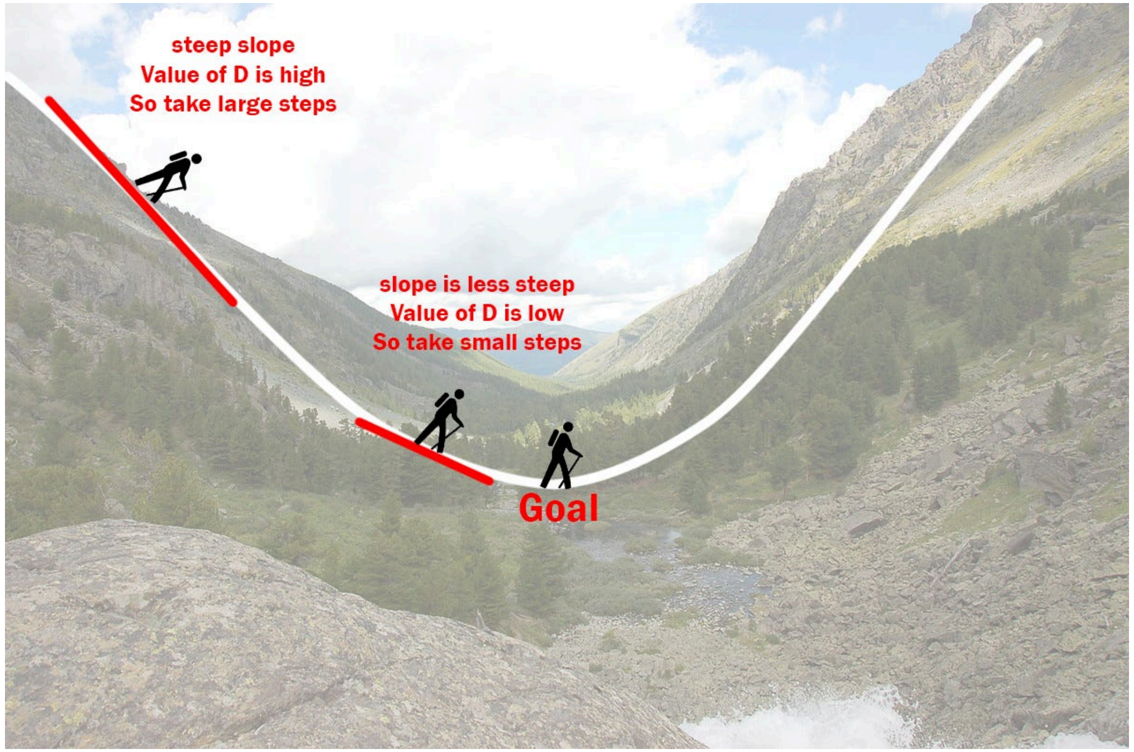
- **Goal**: Given a **differentiable** function $f(t)$, find the input t^* that minimizes $f(t)$.
- What does $\frac{d}{dt} f(t)$ mean?



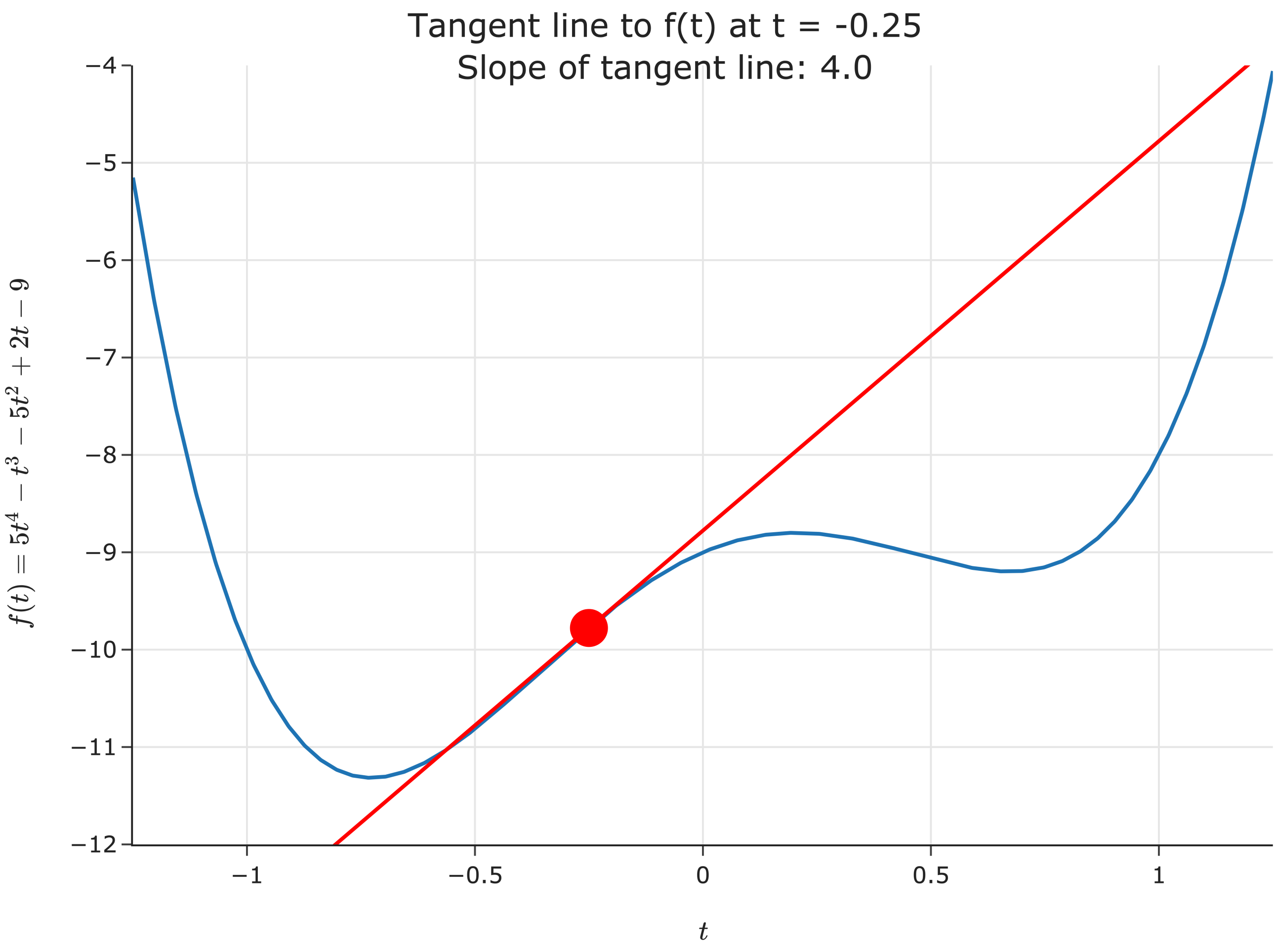
See dsc40a.com/resources/lectures/lec10 for an animated version of the previous slide!

Let's go hiking!

- Suppose you're at the top of a mountain 🏔️ and need to get **to the bottom**.
- Further, suppose it's really cloudy ☁️, meaning you can only see a few feet around you.
- **How** would you get to the bottom?



Searching for the minimum

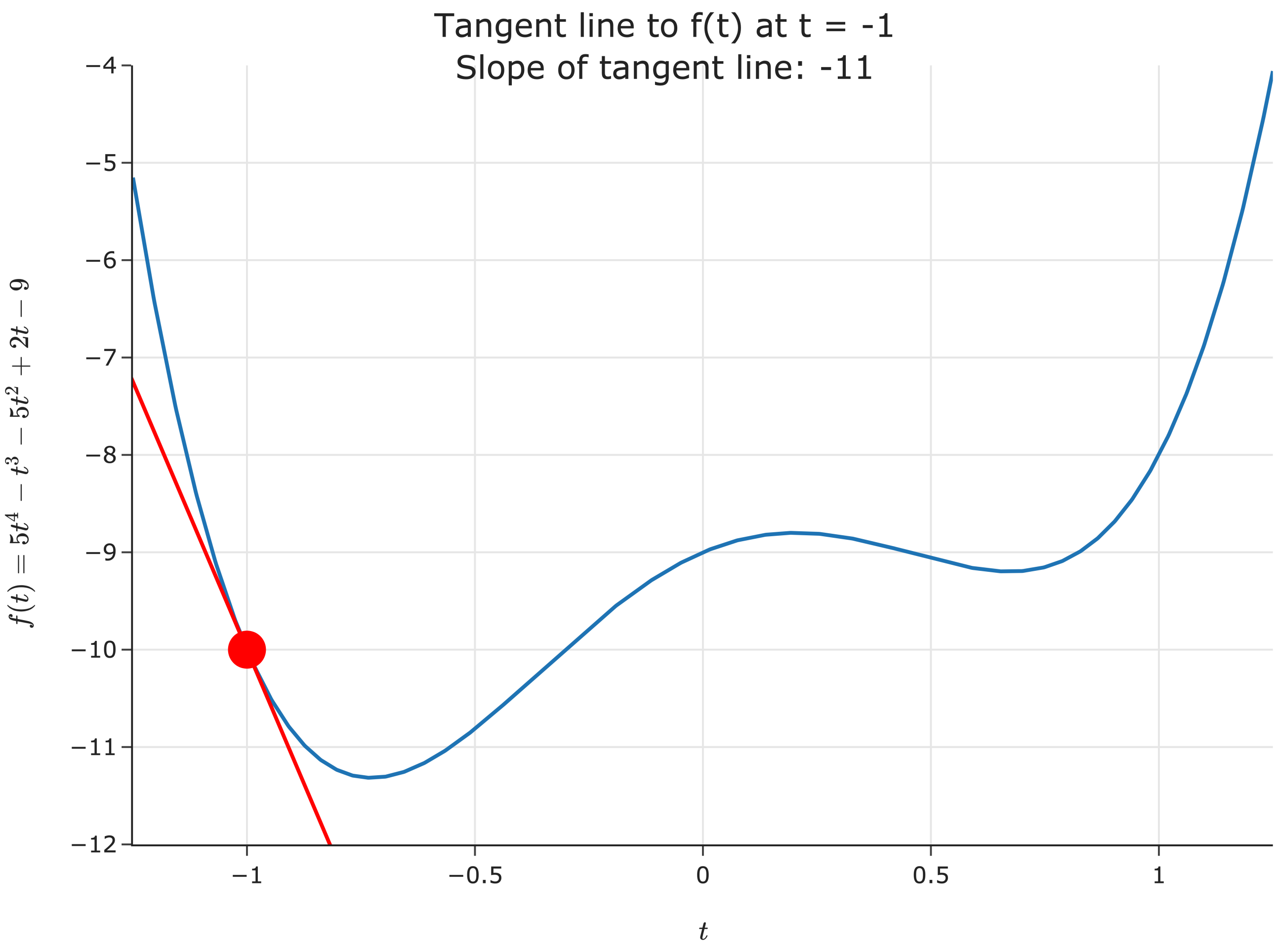


Suppose we're given an initial *guess* for a value of t that minimizes $f(t)$.

If the **slope of the tangent line at $f(t)$** is **positive** 📈:

- Increasing t **increases** f.
- This means the minimum must be to the **left** of the point $(t, f(t))$.
- Solution: **Decrease** t ⬇️.

Searching for the minimum



Suppose we're given an initial *guess* for a value of t that minimizes $f(t)$.

If the **slope of the tangent line at $f(t)$** is **negative** 📉:

- Increasing t **decreases** f.
- This means the minimum must be to the **right** of the point $(t, f(t))$.
- Solution: **Increase** t ⬆️.

Intuition

- To minimize $f(t)$, start with an initial guess t_0.
- Where do we go next?
  - If $\frac{df}{dt}(t_0) > 0$, **decrease** t_0.
  - If $\frac{df}{dt}(t_0) < 0$, **increase** t_0.
- One way to accomplish this:

$$t_1 = t_0 - \frac{df}{dt}(t_0)$$

Gradient descent

To minimize a **differentiable** function f:

- Pick a positive number, α. This number is called the **learning rate**, or **step size**.
- Pick an **initial guess**, t_0.
- Then, repeatedly update your guess using the **update rule**:

$$t_{i+1} = t_i - \alpha \frac{df}{dt}(t_i)$$

- Repeat this process until **convergence** – that is, when t doesn't change much.
- This procedure is called **gradient descent**.

What is gradient descent?

- Gradient descent is a numerical method for finding the input to a function f that minimizes the function.
- Why is it called **gradient** descent?
  - The gradient is the extension of the derivative to functions of multiple variables.
  - We will see how to use gradient descent with multivariate functions next class.
- What is a **numerical** method?
  - A numerical method is a technique for approximating the solution to a mathematical problem, often by using the computer.
- Gradient descent is **widely used** in machine learning, to train models from linear regression to neural networks and transformers (includng ChatGPT)!

See dsc40a.com/resources/lectures/lec10 for animated examples of gradient descent, and see this notebook for the associated code!

Lingering questions

Next class, we'll explore the following ideas:

- When is gradient descent *guaranteed* to converge to a global minimum?
    - What kinds of functions work well with gradient descent?
- How do I choose a step size?
- How do I use gradient descent to minimize functions of multiple variables, e.g.:

$$R_{\text{sq}}(w_0, w_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (w_0 + w_1 x_i))^2$$

Gradient descent and empirical risk minimization

- While gradient descent can minimize other kinds of differentiable functions, its most common use case is in **minimizing empirical risk**.
- For example, consider:
  - The constant model, $H(x) = h$.
  - The dataset $-4, -2, 2, 4$.
  - The initial guess $h_0 = 4$ and the learning rate $\alpha = \frac{1}{4}$.
- **Exercise**: Find h_1 and h_2.