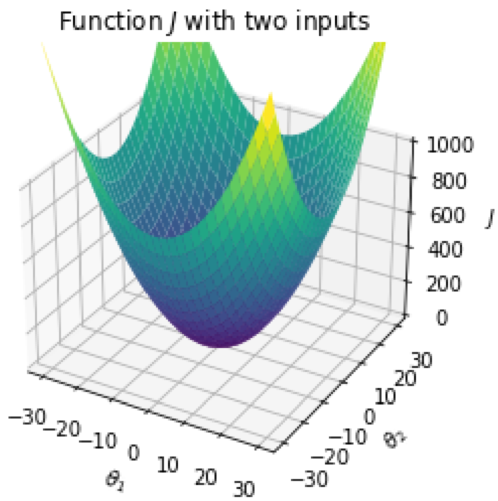
Explanatory Notes for 6.390

Shaunticlair Ruiz (Current TA)

Fall 2022

Multiple Dimensions

Now that we've handled the 1-D case, we'll move into 2-D: now, we have **two** parameters, θ_1 and θ_2, as the input to J.



The "height"of your plot in 3D, is, again, your output! You want to move **downhill**.

Multivariable Local Approximation (Review)

Again, we rely on **calculus**. We want to move up to having more parameters: more **dimensions**.

Before, in 1-D, we found that, if you **zoomed** in enough on a function (using a "**local** view"), we could **approximate** it as a **straight line**, and move up or down that slope.

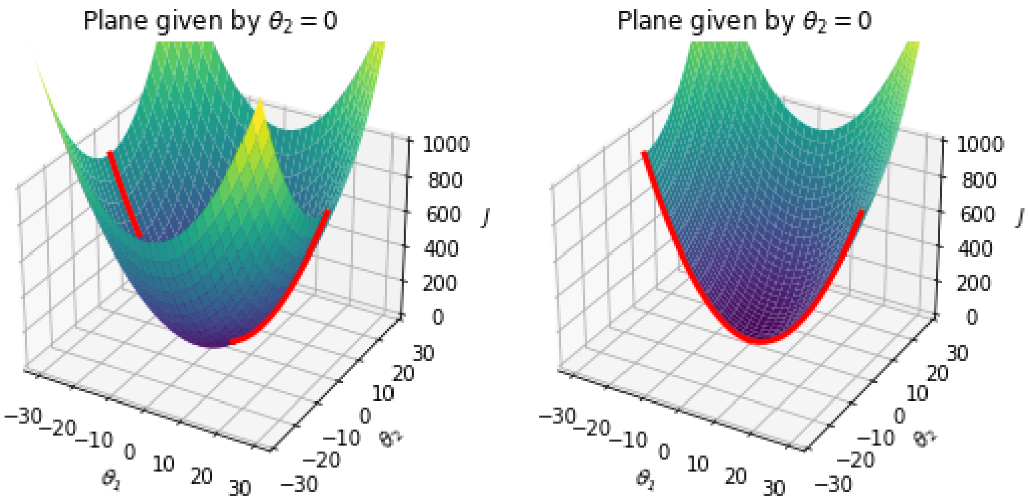
There are **two** ways we can **approximate** like we want to in 2-D:

Remember that, by 2-D, we mean two **parameters**/inputs to J. If we add in the **height** of our function, that means our plot will **look** like 3-D!

- First, we could turn it back into 1-D: we remove one variables.

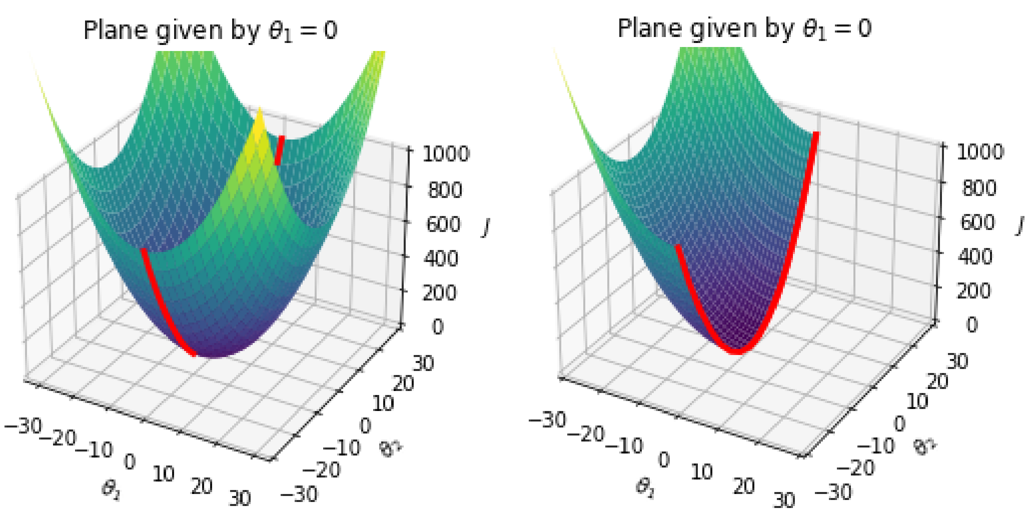
  We do this by turning one variable constant: take $\theta_2 = 0$. Now, we have one free variable θ_1. Same as 1-D.

> **Concept 1**
>
> We can **reduce** the number of **variables** we have to work with, by holding some of them **constant**. That way, we have a **simpler** problem to work with.
>
> This is **the same** as taking a single 2-D plane in a 3-D plot.



If we focus on a single plane of this surface, we end up with a **parabola**.

We can do the same the other way: we take $\theta_1 = 0$, and now we have a 1-D problem in θ_2.



We can slice along the other axis as well!

Along each **axis**, θ_1 and θ_2, you can **approximate** our function as **two** different straight lines. Which leads into our next point...

- Second way: if we take the two perpendicular **lines** we got from each dimension, we can combine them into a **plane**.

Concept 2

If we have **two input variables** (a 2-D problem), we can **approximate** our surface as a **plane** if we **zoom** in enough.

These **approximations** will allow us to **optimize**.

2-D: One dimension at a time

How do we **improve** our function J? Now that we have **two** dimensions, we have to store our change $\Delta\theta$ in a **vector**:

$$\Delta\theta = \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{bmatrix} \tag{1}$$

This **complicates** things: we have two different things to consider **at once**.

Well, the **simplest** way would be to treat it as a **1-D** problem, and do exactly what we did **before**.

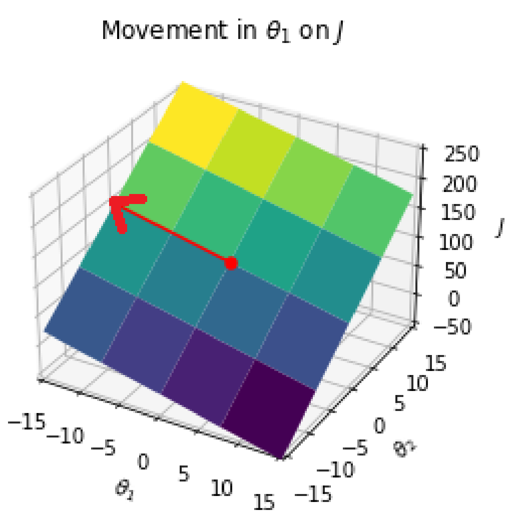
Note that we switched to **partial** derivatives, because we have **multiple** input variables θ_i.

$$\Delta\theta_1 = \frac{\partial J}{\partial \theta_1} \tag{2}$$

Writing this in our **new** notation, we get:

$$\Delta\theta = -\eta \begin{bmatrix} \partial J/\partial\theta_1 \\ 0 \end{bmatrix} \tag{3}$$

And then we would take a **step**, moving along the θ_1 **axis**.

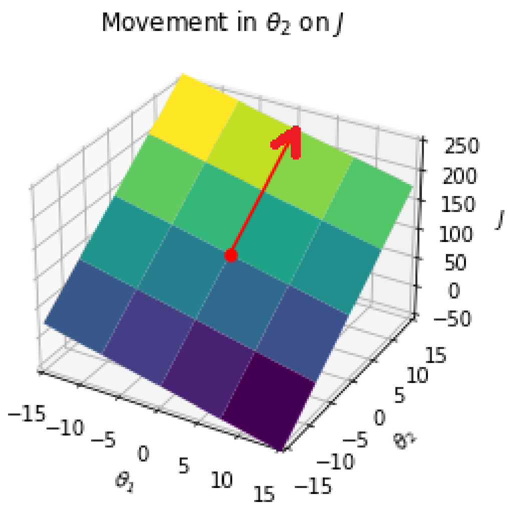


We can move along θ_1 just like on a line.

What if we treated this as a 1-D problem for the **other** variable, θ_2?

$$\Delta\theta = -\eta \begin{bmatrix} 0 \\ \partial J/\partial\theta_2 \end{bmatrix} \tag{4}$$

With this equation, we would be **moving** along the θ_2 axis.



We can do the same with θ_2.

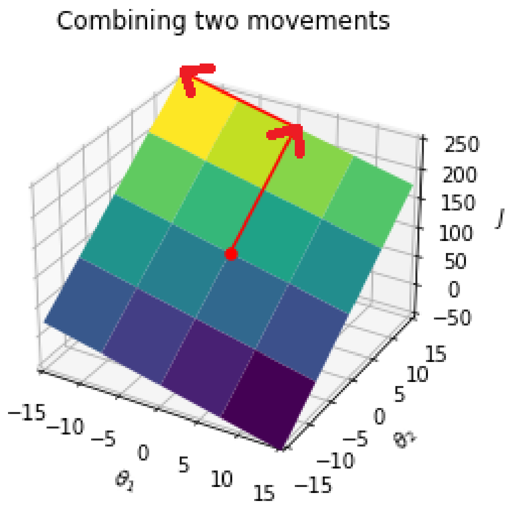
Why not move in **both** directions **at once**? We can **combine** our two derivatives: we'll add up our two steps.

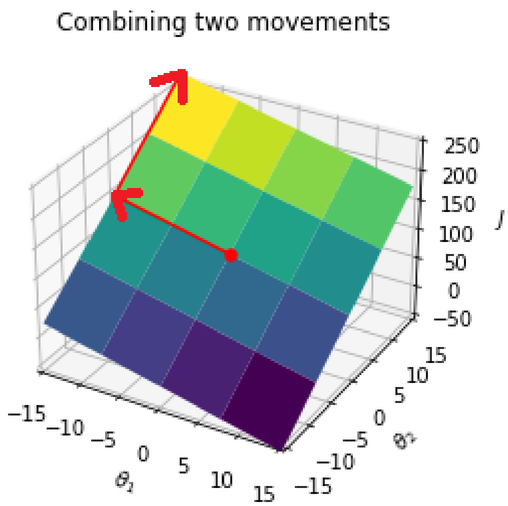
Linearity means that I can **add** them up without anything **weird** happening.

The relevant linearity rule: $L(x+y) = L(x) + L(y)$. In other words: taking two separate steps is the same as one big step.

$$\Delta\theta = -\eta \begin{bmatrix} \partial J/\partial\theta_1 \\ 0 \end{bmatrix} - \eta \begin{bmatrix} 0 \\ \partial J/\partial\theta_2 \end{bmatrix} \tag{5}$$

These can be combined because we're treating our function as a **flat** plane: if I move in the θ_1 direction first, it doesn't change the θ_2 slope, and vice versa.





Our plane being flat means we can take both operations, back-to-back! Notice that the order doesn't matter.

$$\Delta\theta = -\eta \begin{bmatrix} \partial J/\partial\theta_1 \\ \partial J/\partial\theta_2 \end{bmatrix} \tag{6}$$

So, let's use that to optimize:

Key Equation 3

In **2-D**, you can optimize your function J using this rule:

$$\theta_{new} = \theta_{old} - \eta \underbrace{\begin{bmatrix} \partial J / \partial \theta_1 \\ \partial J / \partial \theta_2 \end{bmatrix}}_{\text{Using } \theta_{old}}$$

This is our **gradient descent** rule for 2-D.

This sort of approach makes some **sense**: if $\frac{\partial J}{\partial \theta_1}$ is **bigger** than $\frac{\partial J}{\partial \theta_2}$, that means that you can get **more benefit** from moving in the θ_1 direction than θ_2.

So, in that case, your step will move more in the θ_1 direction: it's a more **efficient** way to get a **better** hypothesis!

But for now, we **don't know** that this is necessarily the **optimal** way to change θ - we'll explore that later.