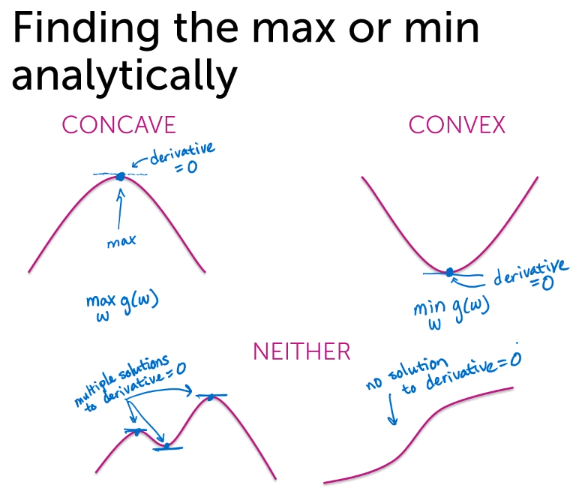
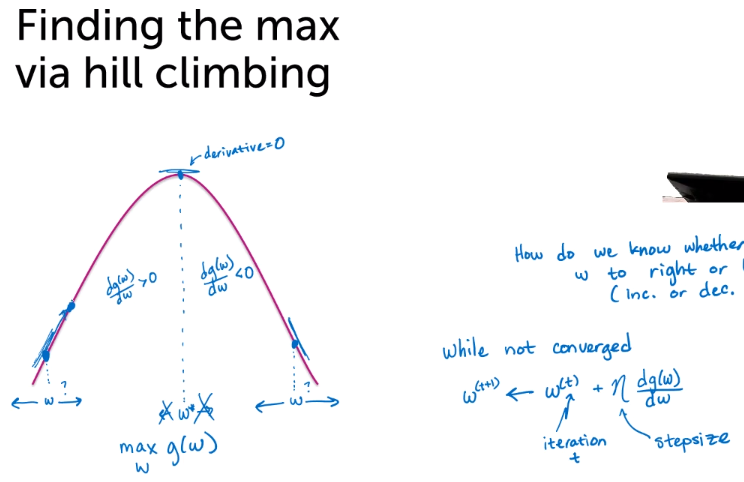
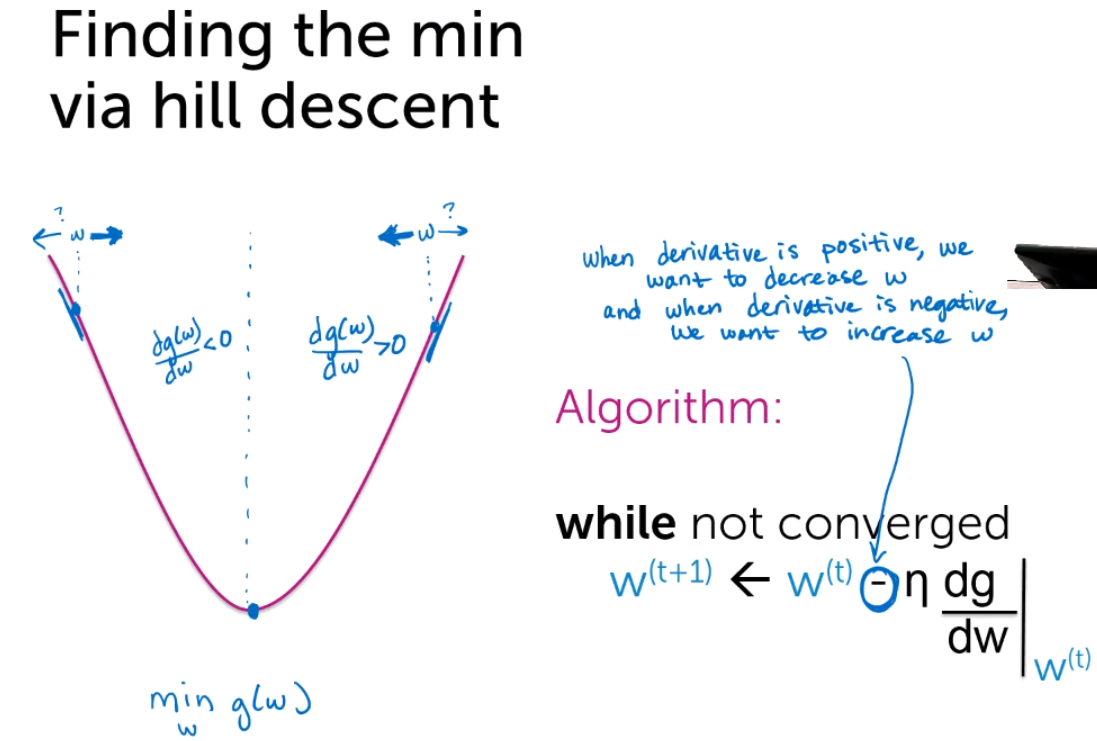
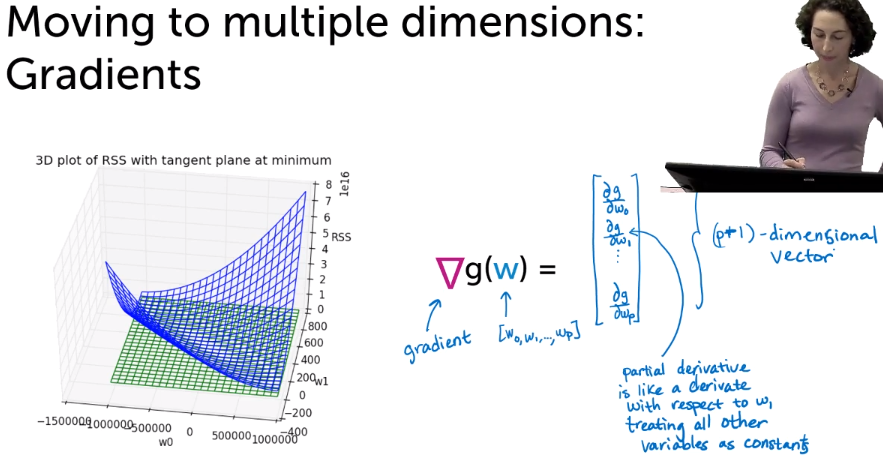
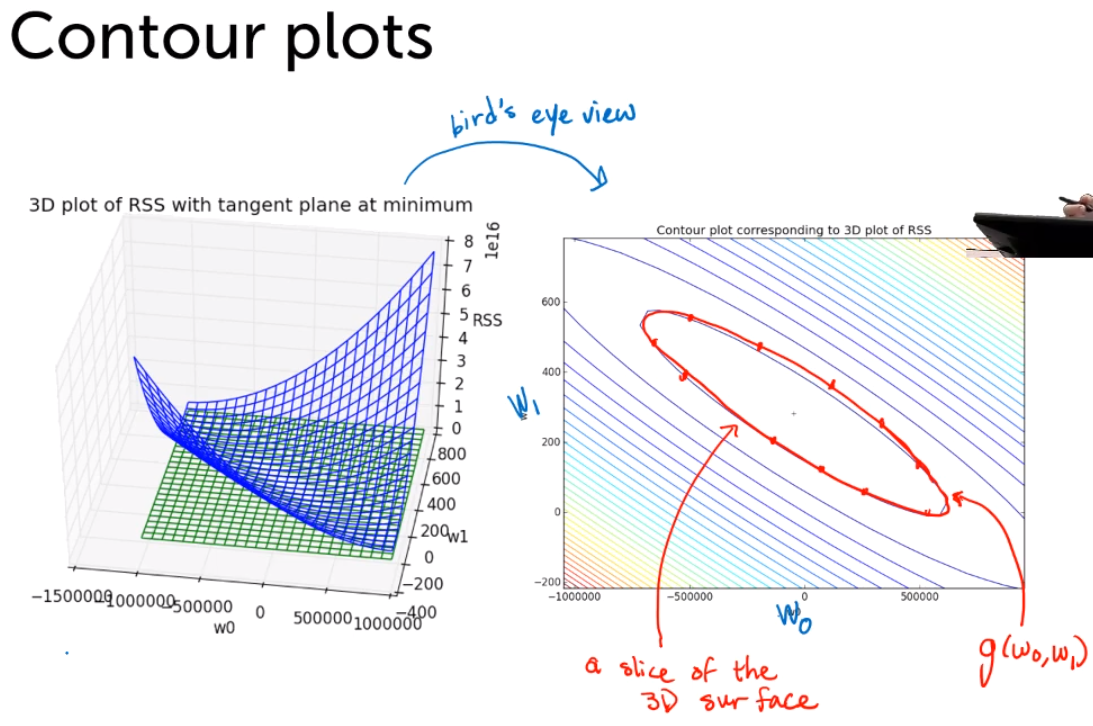
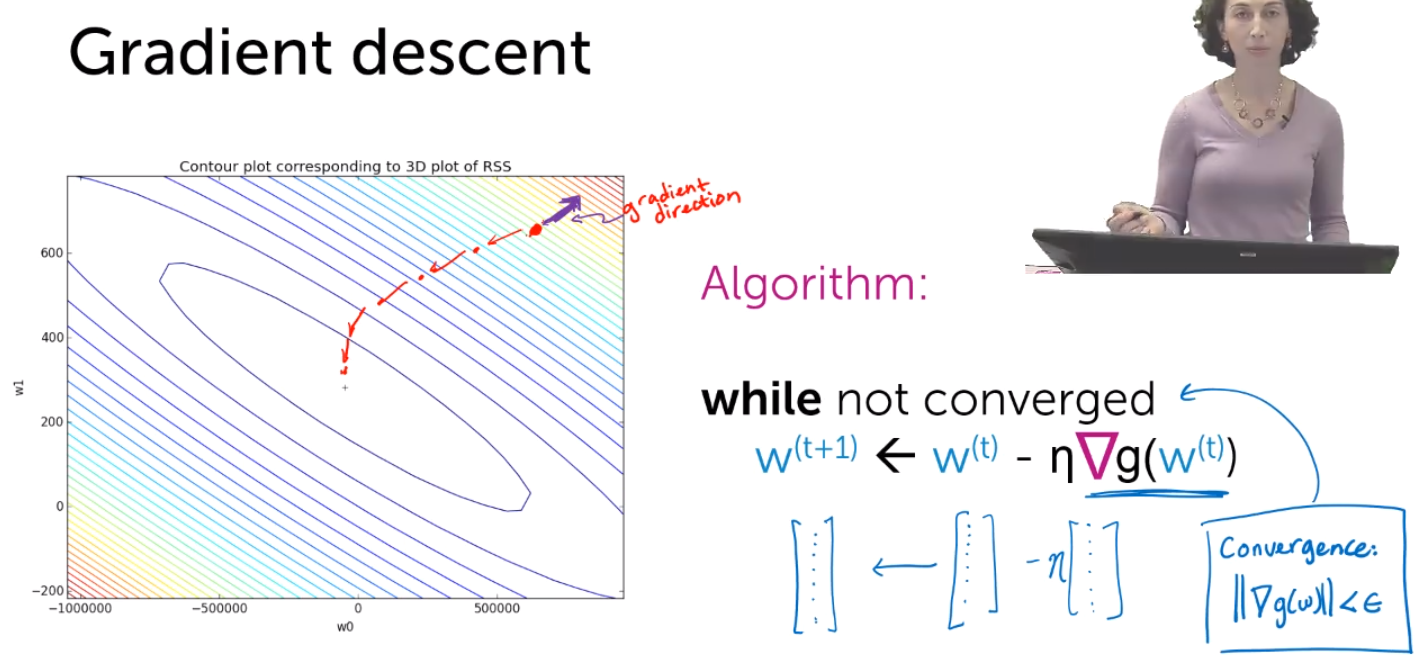
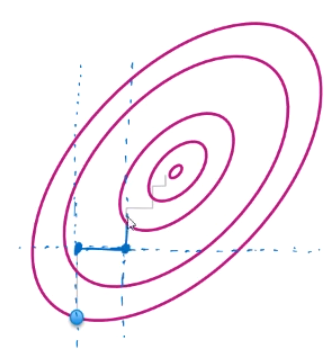
# Optimization



* Finding max/min:
  + Approach 1 - First order derivative = 0
    - For convex or concave functions, can take first order derivative that equals to zero and find unique solution
    - In reality we might have various local maximums/minimums
  + Approach 2 – hill climbing
    - If derivative>0, move to the left. If derv<0, move to right. If der=0 (or close to 0), stop!
    - 
* Gradient – similarly to hill climbing, but for multiple dimensions. We calculate partial derivatives of cost function with respect to each coefficient (dimension). Vector of all these derivatives shows whether we will increase or decrease cost function and how much
  + - 
    - 

## Coordinate descent

* Motivation:
  + optimize: RSS(w) + lambda |w| [residual sum of squares + L1 norm]. We cannot use gradient ascent because we cannot take a derivative of absolute value (there infinitely many solutions at zero)
  + so we use ‘coordinate descent’
* It is hard to minimize cost function with respect to all the coefficients. So, we fix all coefficients and optimize only with respect to 1 feature. Thus, at one step we would move only along one axis:
* 
* No step size in coordinate descent => very helpful
* Algorithms:
  + Pick a feature at randdom – random or stochastic coordinate ascent
  + Round robin – choose features one by one and then start from beginning