Gradient Descent, Regularization

Fitting Models as Optimization

- Loss: compare fitted to true value

$$\ell(y, \hat{y}) = (y - \hat{y})^2$$

Fitting Models as Optimization

- Loss: compare fitted to true value **(w/ parameter)**

$$\ell(y, X\beta) = (y - X\beta)^2$$

Fitting Models as Optimization

- Loss: compare fitted to true value **(over data)**

$$\sum_{i=1}^{n} \ell(y_i, x_i\beta) = \sum_{i=1}^{n} (y_i - x_i\beta)^2$$

Fitting Models as Optimization

- Optimization problem:

$$\hat{\beta} = \arg\min \sum_{i=1}^{n} (y_i - x_i\beta)^2$$

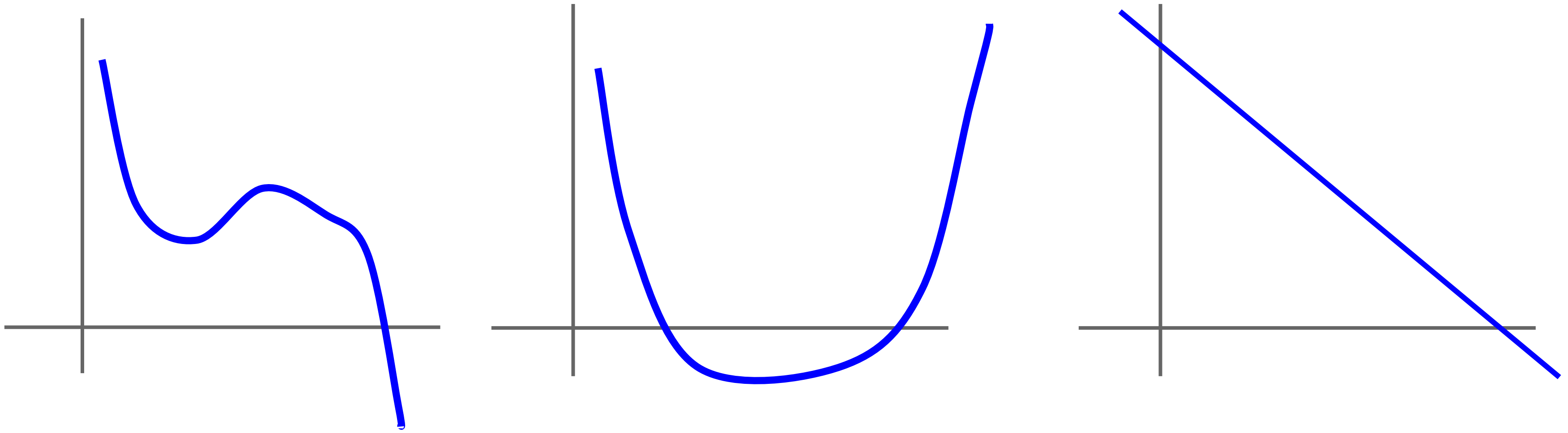
Gradient Descent

- Goal: minimize the loss
- Calculus approach to doing it:
  - Loss is a function of parameters
  - Take derivative (gradient) of that function (wrt parameters)
  - *Set equal to zero*
  - *Use algebra to solve it*

Gradient Descent

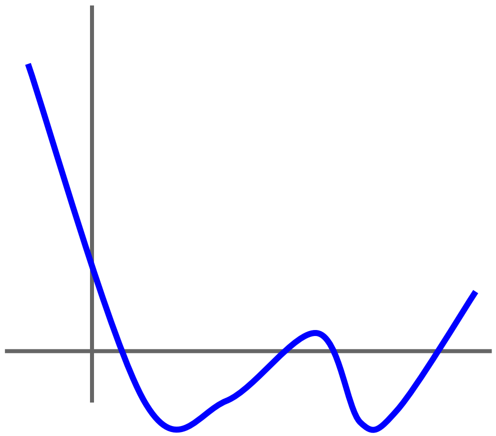
- Goal: minimize the loss
- (Approximate) calculus approach to doing it:
  - Loss is a function of parameters
  - Take derivative (gradient) of that function (wrt parameters)
  - *Derivative tells you where to go to decrease function*
  - *Follow the path to the solution*

When will Gradient Descent Work?

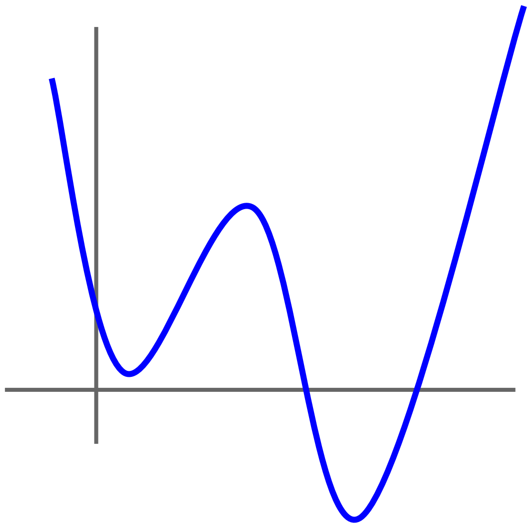


A B C

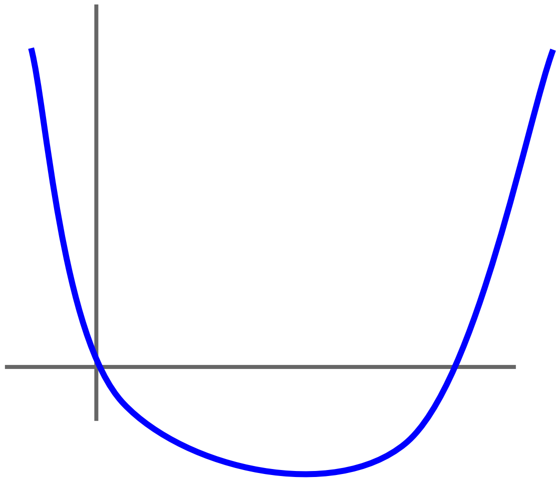
When will Gradient Descent Work?



A

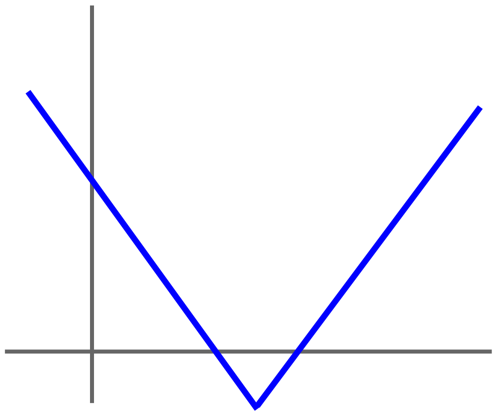


B

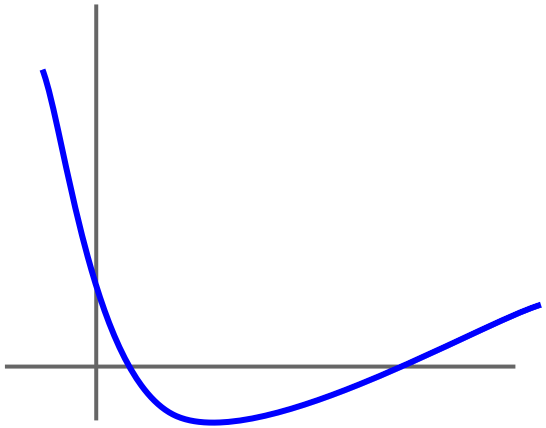


C

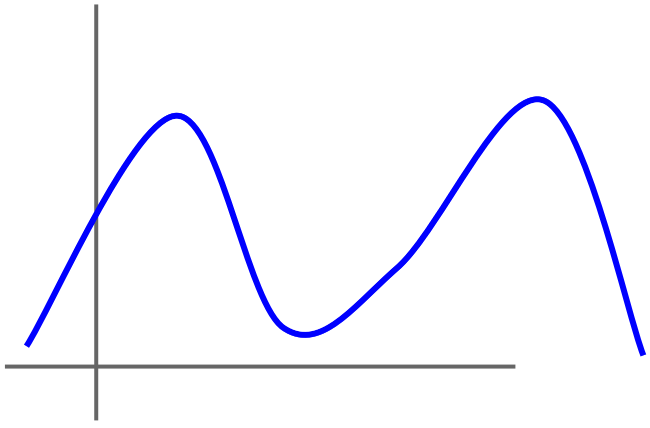
When will Gradient Descent Work?



A



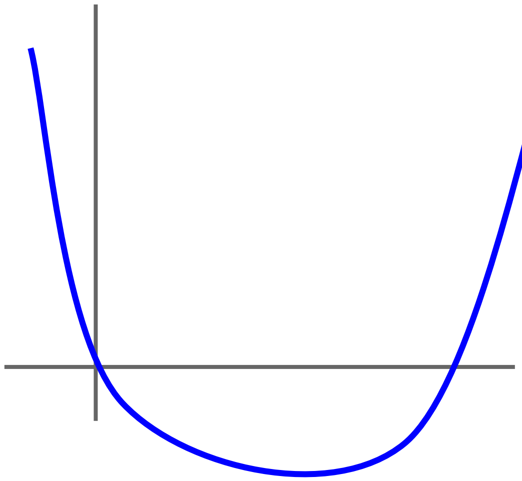
B

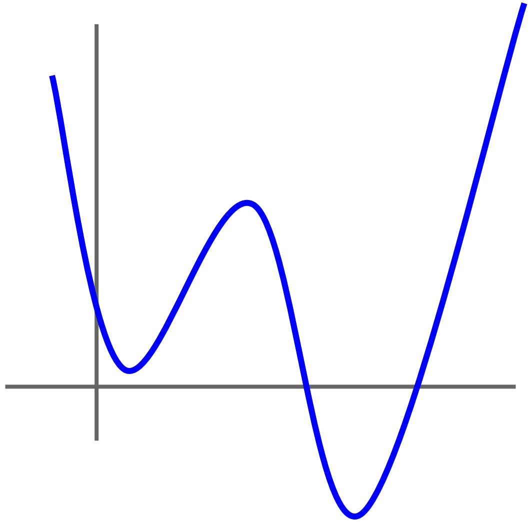


C

Gradient Descent

- Convex problems guaranteed to work
  - Can you roll a ball down the hill?
  - Will you end up in a unique spot if you do?





Do we only care about loss?

- What would happen if we are able to get the loss to zero on the training data? Is that good?

Do we only care about loss?

- Regularization lets us trade *within sample fit* and *complexity*.
    - Will a complex model generalize well?
    - Will a model that fits the training data perfectly generalize well?

Regularization: Complexity penalties

- Complexity penalties:

$$\hat{\beta} = \arg\min \sum_{i=1}^{n} (y_i - x_i\beta)^2 + \lambda \mathrm{penalty}(\beta)$$

- What do we do to optimize this? When is this possible?

Regularization: Complexity penalties

$$\hat{\beta} = \arg\min \sum_{i=1}^{n} (y_i - x_i \beta)^2 + \lambda \text{penalty}(\beta)$$

"Ridge" $\text{penalty}(\beta) = \sum_j \beta_j^2$

"Lasso" $\text{penalty}(\beta) = \sum_j |\beta_j|$

Regularization: Complexity penalties

"Ridge" $$\text{penalty}(\beta) = \sum_j \beta_j^2$$

"Lasso" $$\text{penalty}(\beta) = \sum_j |\beta_j|$$

- How do these measure complexity?

Regularization: Complexity penalties

"L-Zero" $\mathrm{penalty}(\beta) = \sum_j I(\beta_j \neq 0)$

- Why not use this penalty?

Other types of regularization

- Not going all the way in optimization
  - Stop before the optimum
  - Don’t use all the data; e.g. Stochastic Gradient Descent (SGD)
    - Footnote on SGD: developed decades ago when machines were too weak to use all data, these days we find ourselves back in this situation with “Big Data”