let m training examples, n features

| Gradient Descent | Normal Equation |
|---|---|
| ⇒ Need to choose 'α' | ⇒ No need to choose 'α' |
| ⇒ Needs many iterations | ⇒ Don't need to iterate |
| ⇒ works well even when n is large<br>↙<br>If $n \approx 10^6$ or larger | ⇒ Need to comput $(X^T X)^{-1}$ ** ⇒ Slow if n is very large<br>n = 100, 1000, 10,000<br>⇓ fast ⇓ reasonable ⇓ might be slow |

** If $X \Rightarrow \mathbb{R}^{m \times n+1}$ then $X^T \Rightarrow \mathbb{R}^{n+1 \times m}$

Set, $\underset{n \times m}{X^T}\ \underset{m \times n}{X} = A \Rightarrow \underbrace{\mathbb{R}^{n+1 \times n+1}}_{\text{Square matrix}}$ $O(n^3) \Rightarrow$ time complexity of inverting a matrix

same

∴, for large value of n inverting a matrix is computationally expensive.

What if $(X^T X)$ is non-invertable? (i.e. singular matrix/dengenerate matrix)

octave: $\underset{\downarrow \text{ pseudo inverse}}{\text{pinv}}(X'{*}X){*}X'{*}y$

pseudo inverse → this can [illegible]

If $X^T X$ is non-invertaible

⇒ Redundant features exist (i.e linearly dependent)

Eg) $x_1$ = size (feet$^2$); $x_2$ = size (m$^2$)