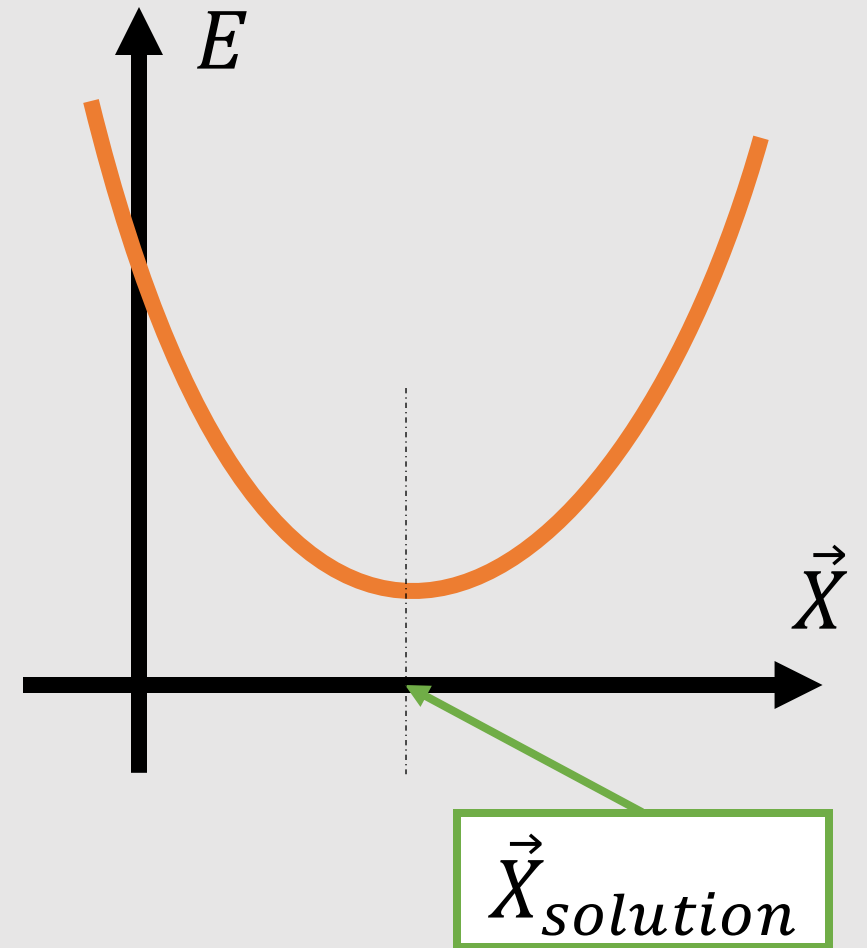


Numerical Optimization

What is Optimization?

- Find input parameter \vec{X} where a cost function $W(\vec{X})$ is minimized

$$\vec{X}_{solution} = \underset{\vec{X}}{\operatorname{argmin}} W(\vec{X})$$



Optimization Solve Many Problems

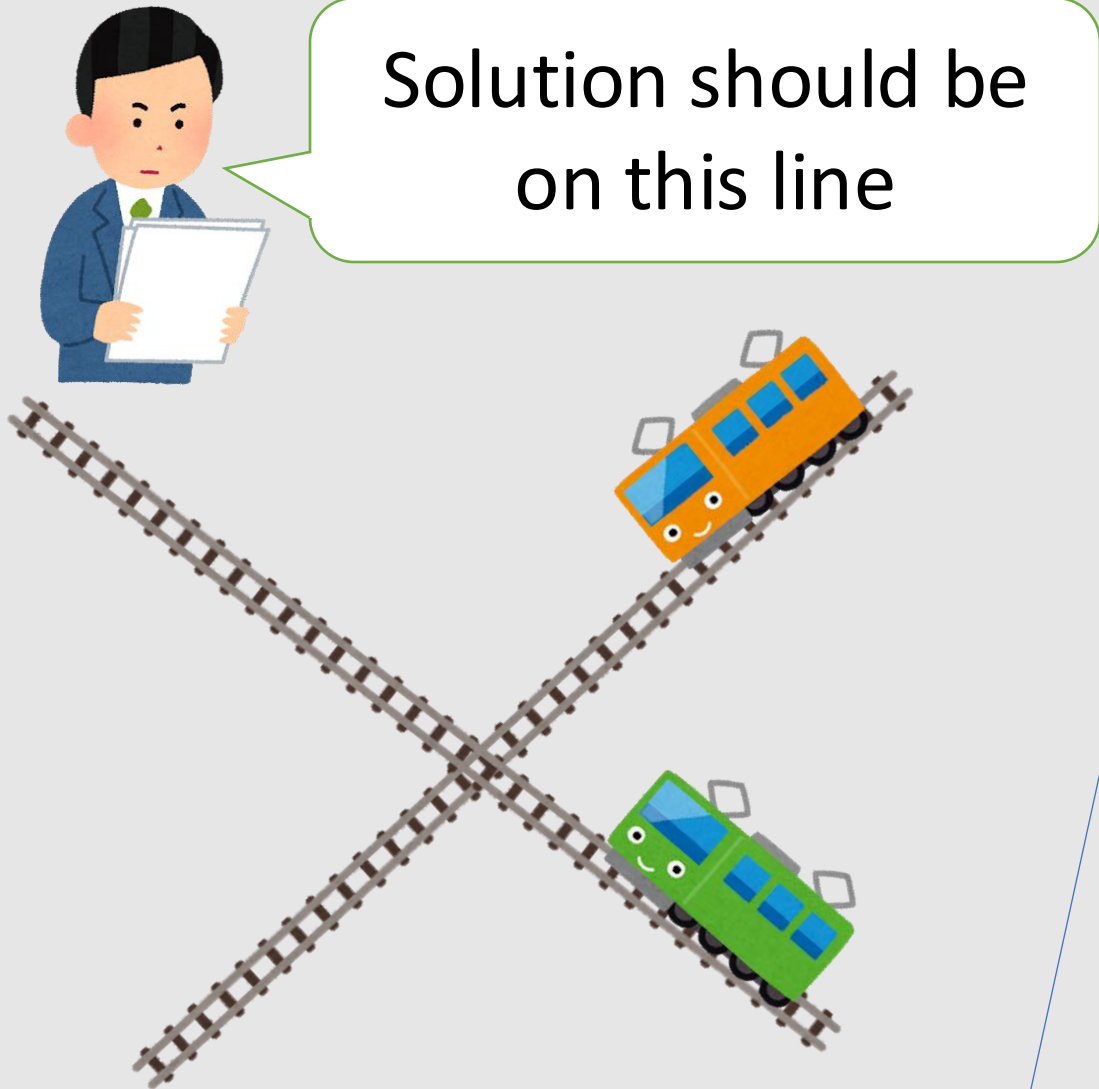
- What typical computer science paper looks like:

a sketch or a parameter sample, and (iii) the reconstruction error of a parameter sample from itself in an auto-encoder fashion. Thus, the combined loss function is defined as:

$$\begin{aligned}\mathcal{L}(\mathbf{P}, \mathbf{M}, \mathbf{S}) = & \omega_1 \|P - f_{L2P}(f_{S2L}(S))\|_2 + \omega_2 \|M - f_{L2M}(f_{S2L}(S))\|_2 \\ & + \omega_3 \|M - f_{L2M}(f_{P2L}(P))\|_2 + \omega_4 \|P - f_{L2P}(f_{P2L}(P))\|_2,\end{aligned}\tag{1}$$

where $\{\omega_1, \omega_2, \omega_3, \omega_4\}$ denote the relative weighting of the individual errors. We set these weights such that the average gradient of

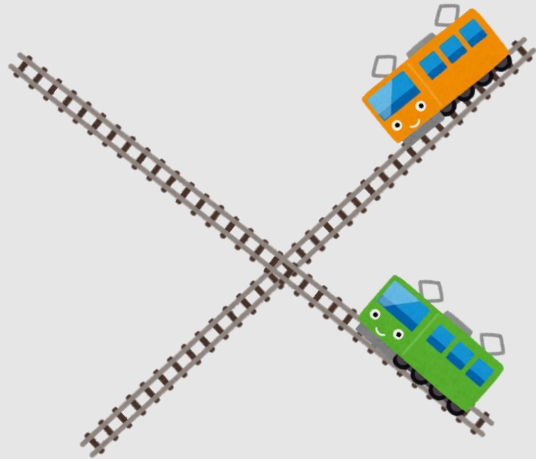
Solving Constraints v.s. Optimization



Solving Constraints v.s. Optimization



Solution should be
on this line



Linearization

$$Ax = b$$

Solution should be at the
bottom of this hole

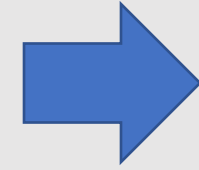


There are many
weapons to fight



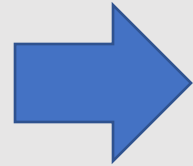
Three Optimization Approaches

- Stochastic Optimization



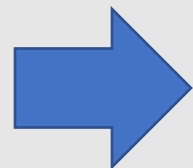
Requires value $W(\vec{X})$

- Gradient Descent



Requires gradient $\nabla W(\vec{X})$

- Newton Method

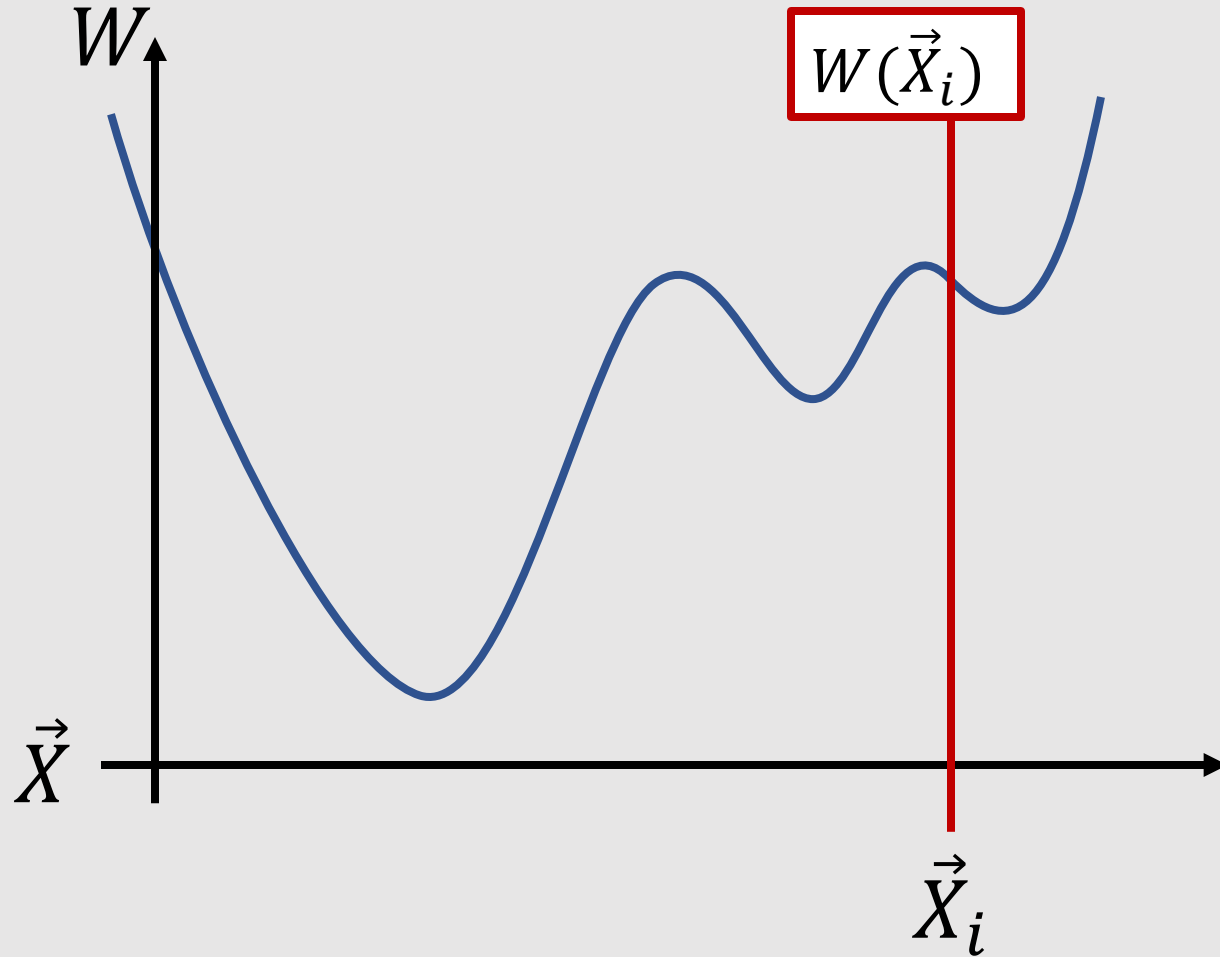


Requires gradient & hessian
 $\nabla W(\vec{X}), \nabla^2 W(\vec{X})$

Stochastic Optimization



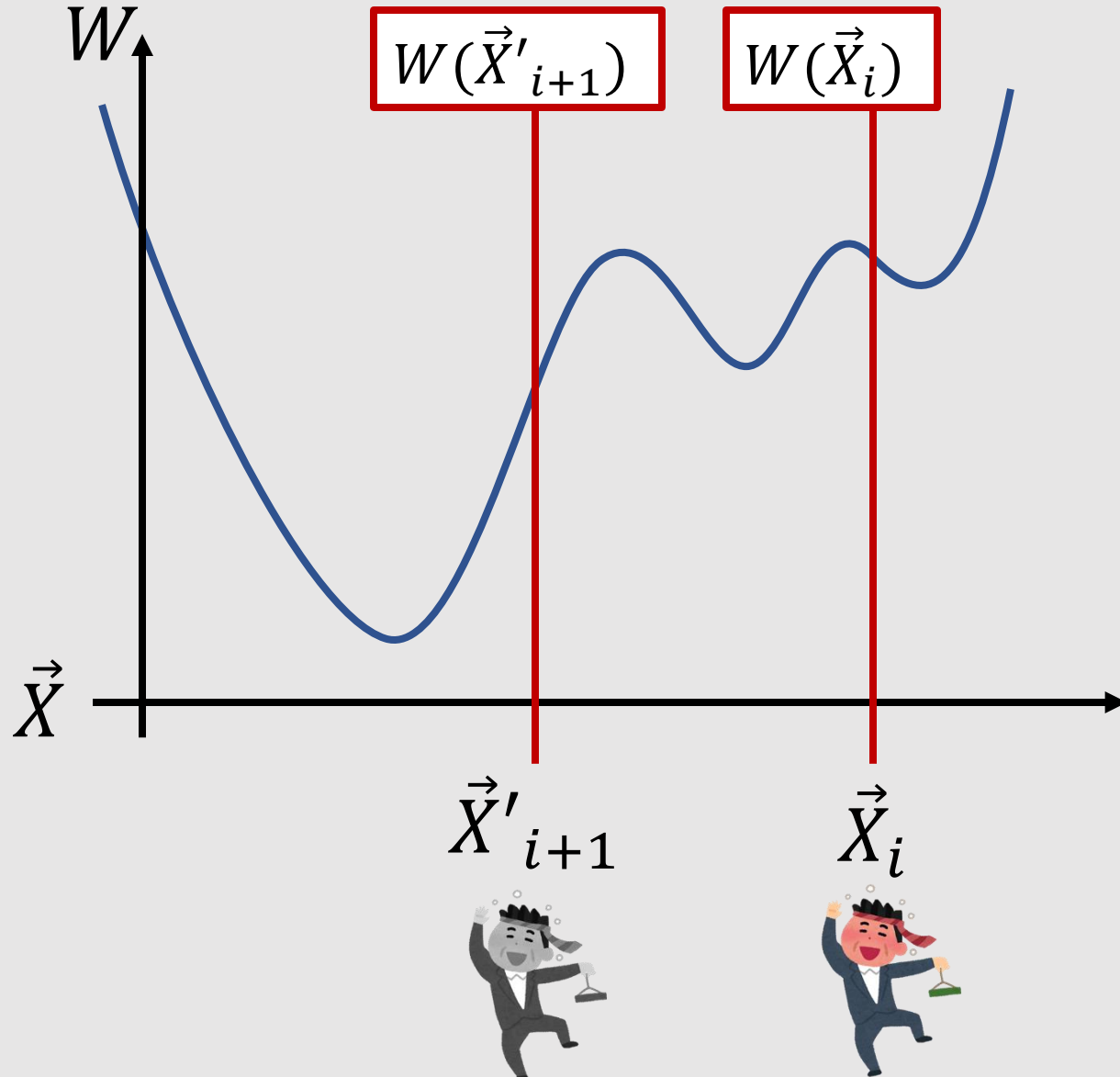
Find Minimum by Random Sampling 1



1. Starting from an initial guess \vec{X}_0
2. Evaluate $W(\vec{X}_i)$

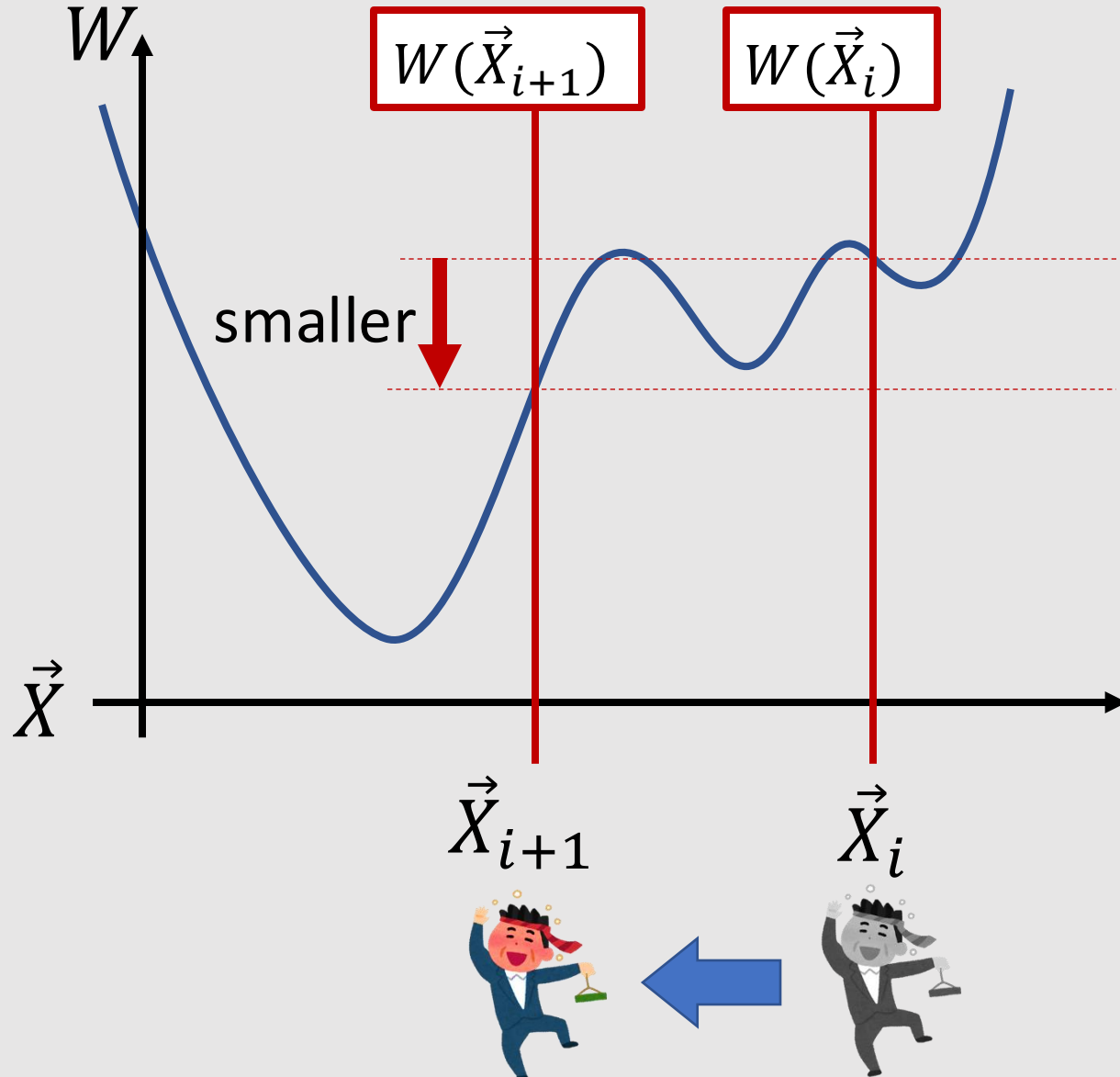
1. Starting from an initial guess \vec{X}_0
2. Evaluate $W(\vec{X}_i)$

Find Minimum by Random Sampling 2



1. Starting from an initial guess \vec{X}_0
2. Evaluate $W(\vec{X}_i)$
3. Make a candidate
$$\vec{X}'_{i+1} = \vec{X}_i + \text{Random}$$
4. Evaluate $W(\vec{X}'_{i+1})$

Find Minimum by Random Sampling 3

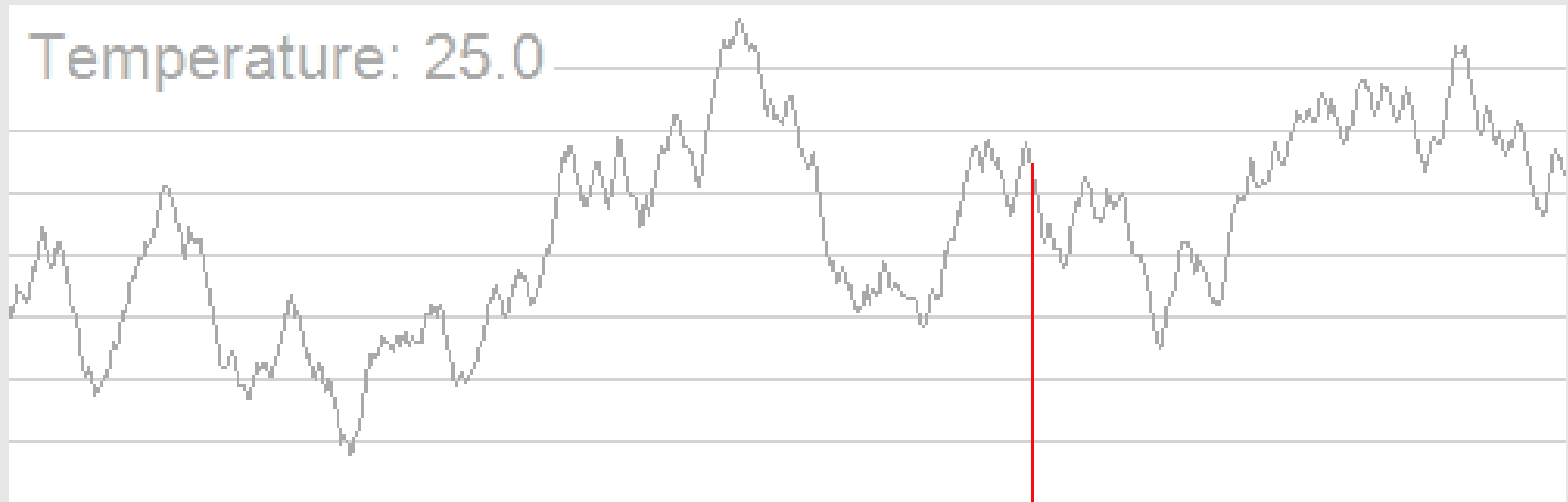


1. Starting from an initial guess \vec{X}_0
2. Evaluate $W(\vec{X}_i)$
3. Make a candidate
 $\vec{X}'_{i+1} = \vec{X}_i + \text{Random}$
4. Evaluate $W(\vec{X}'_{i+1})$
5. Move \vec{X} to the candidate
if $W(\vec{X}'_{i+1}) < W(\vec{X}_i)$
6. Go to 3

Simulated Annealing Method

Gradually make the random update small during iteration

➡ Make the optimization robust to local minima

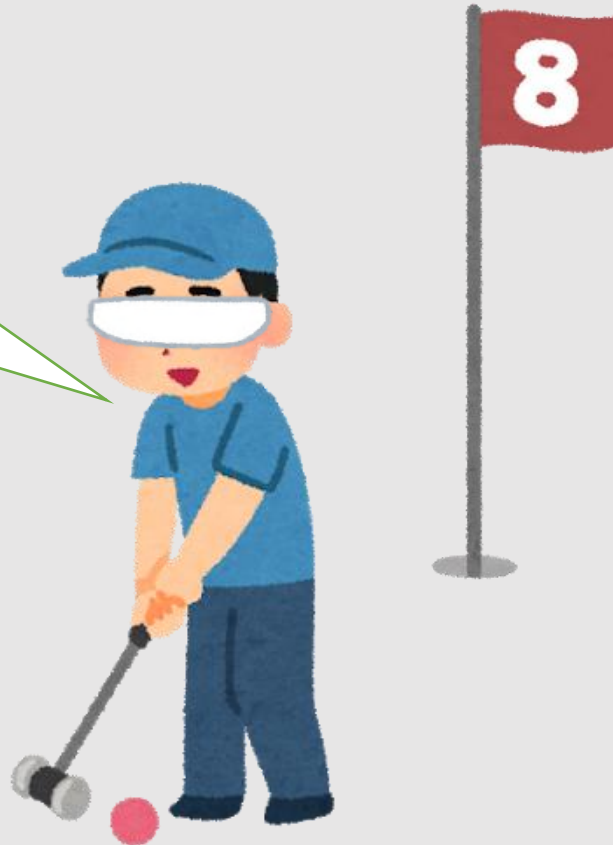


Credit: Kingpin13 @ Wikipedia

Stochastic Optimization: Blinded Golf

- Optimizer do not know the direction & strength to hit

Swing in the
random direction!



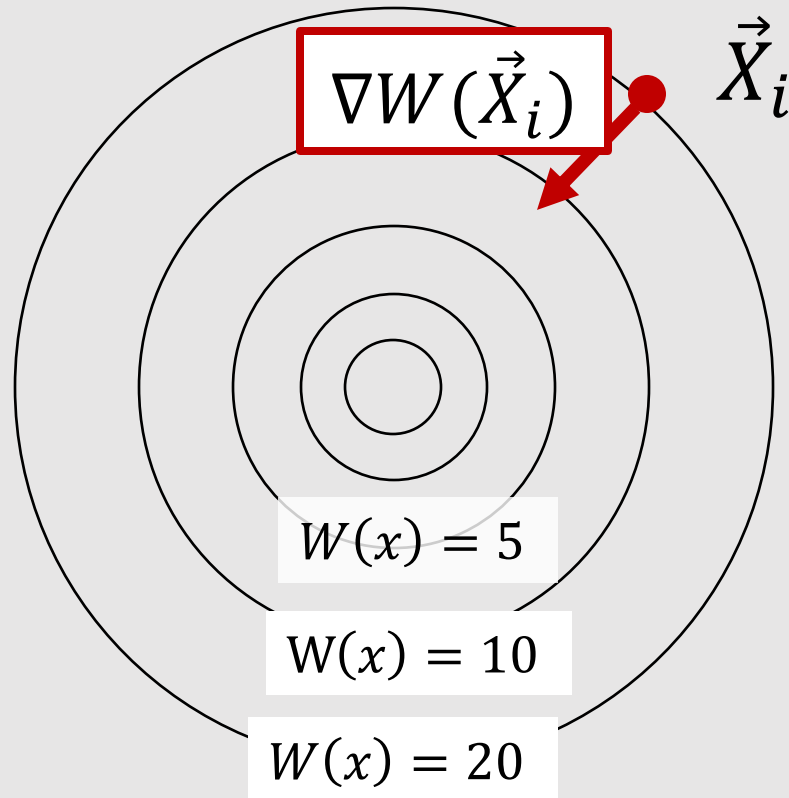
Gradient Descent Method

最急降下法



Gradient Descent Method

- A.k.a “steepest descent method” or “hill climbing method”



$$\vec{X}_{i+1} = \vec{X}_i - \alpha \nabla W(\vec{X}_i)$$

Learning rate



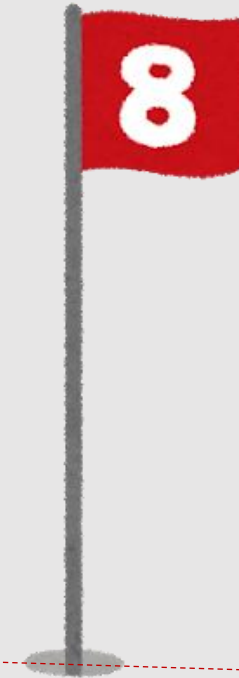
Gradient Descent: Blinded Golf with a Guide

- Optimizer know the direction, but do not know strength to hit



Aim that direction!

OK, but how hard?



Japanese Version of “Pinata”

- Breaking a watermelon with a stick on a beach



Credit: BeenAroundAWhile @ Wikipedia



Move left and hit!

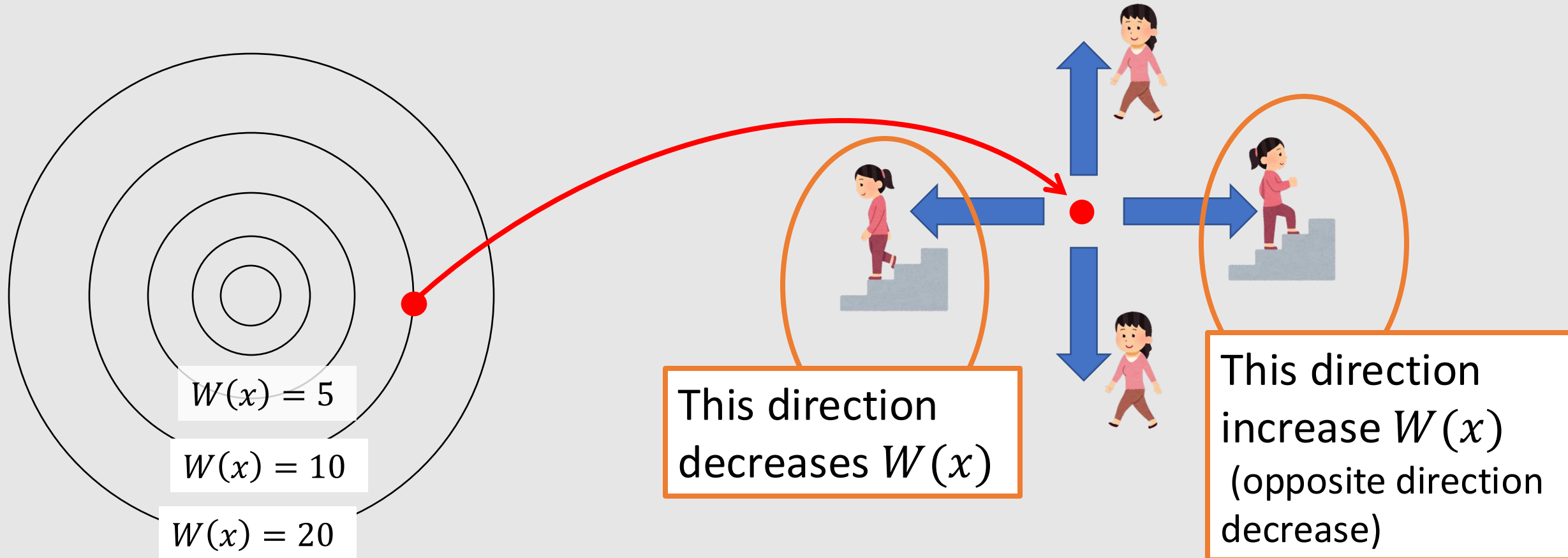


Newton-Raphson Method



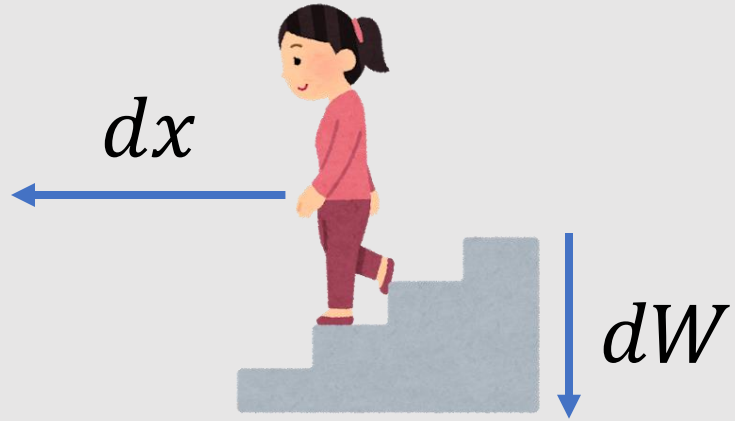
What is **not** Minimum

- A point is **not minimum** if there is a direction changing $W(x)$

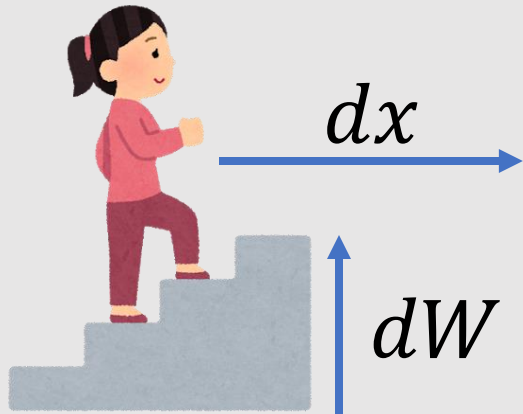


What is **not** Minimum

- A point is **not minimum** if $\exists dx \neq 0$ s.t. $\nabla W(x) \cdot dx \neq 0$



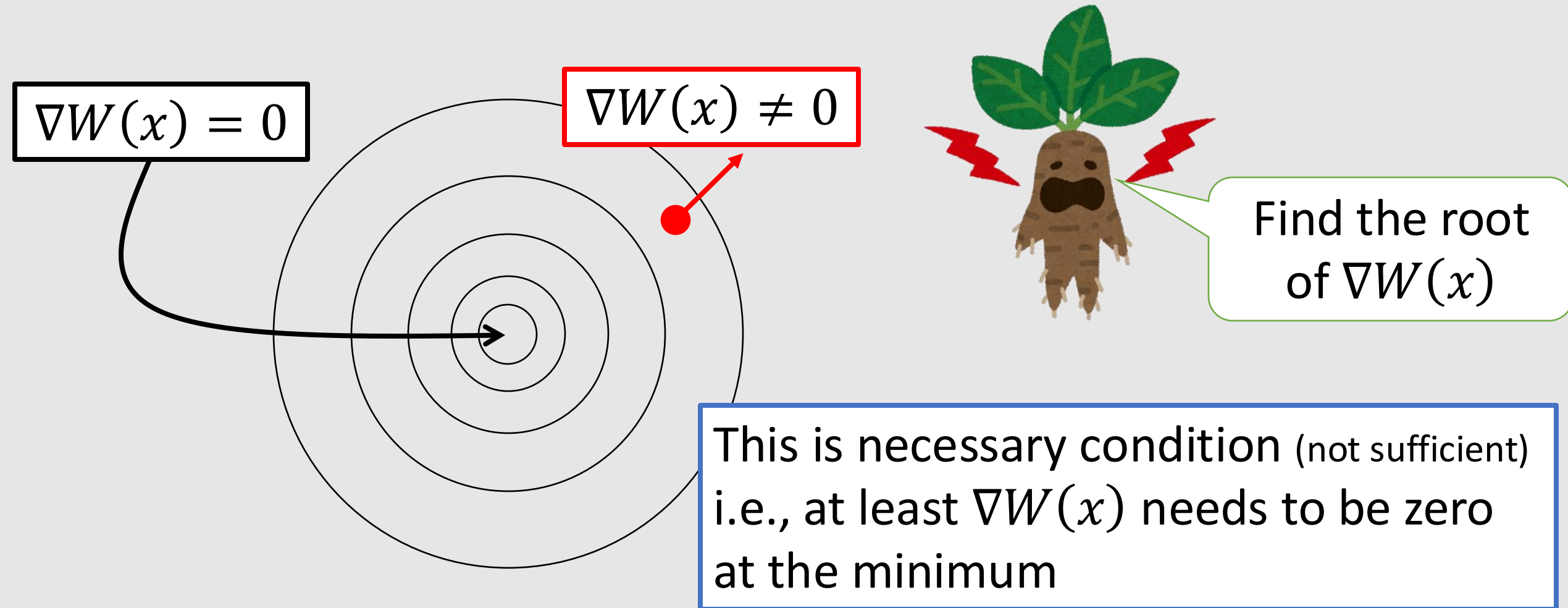
$$dW = \nabla W(x) \cdot dx < 0$$



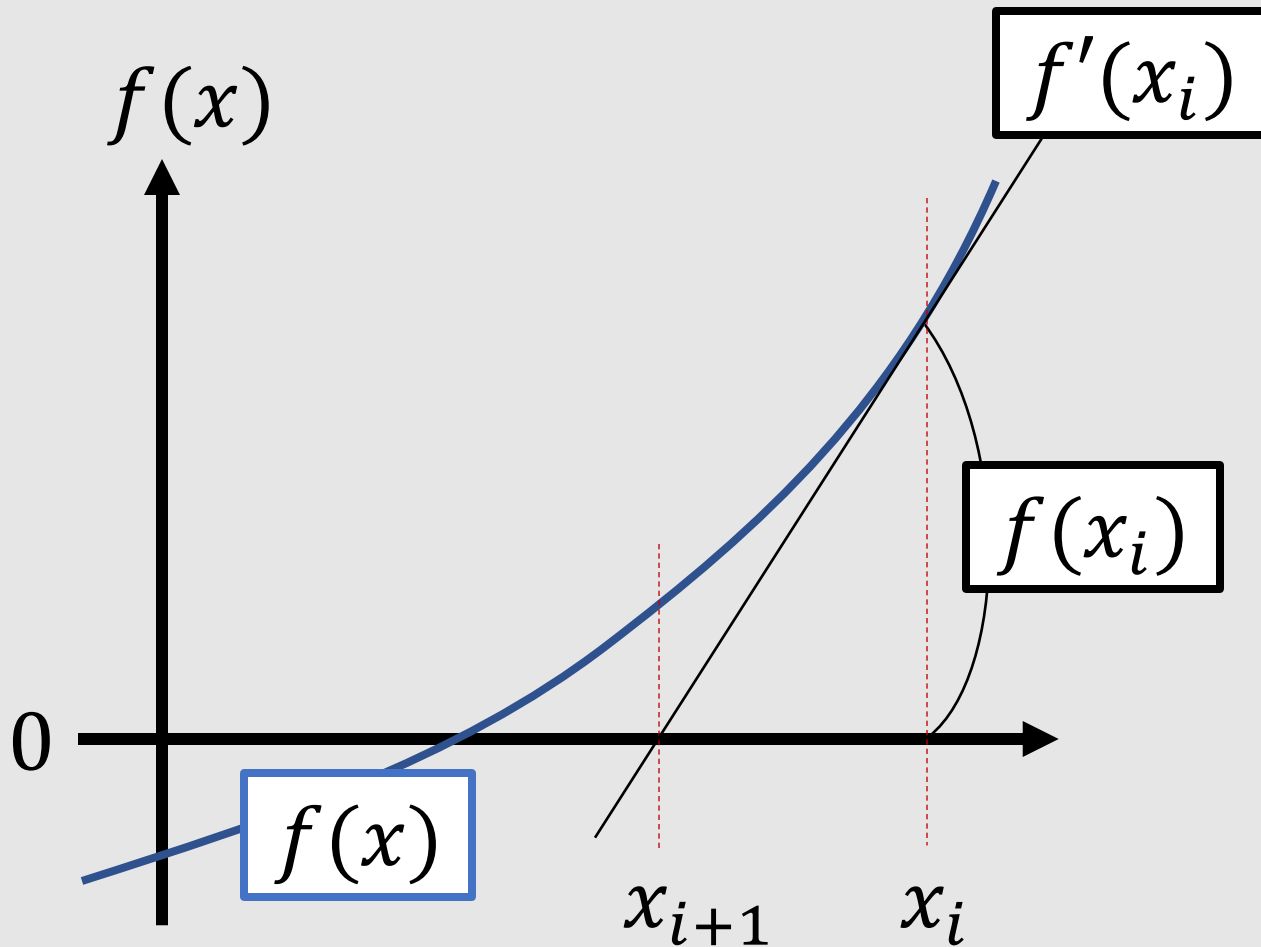
$$df = \nabla f(x) \cdot dx > 0$$

What **Might be** Minimum: Zero Gradient

$$\nabla W(x) = 0$$



Finding the **Root** of a Scalar Function

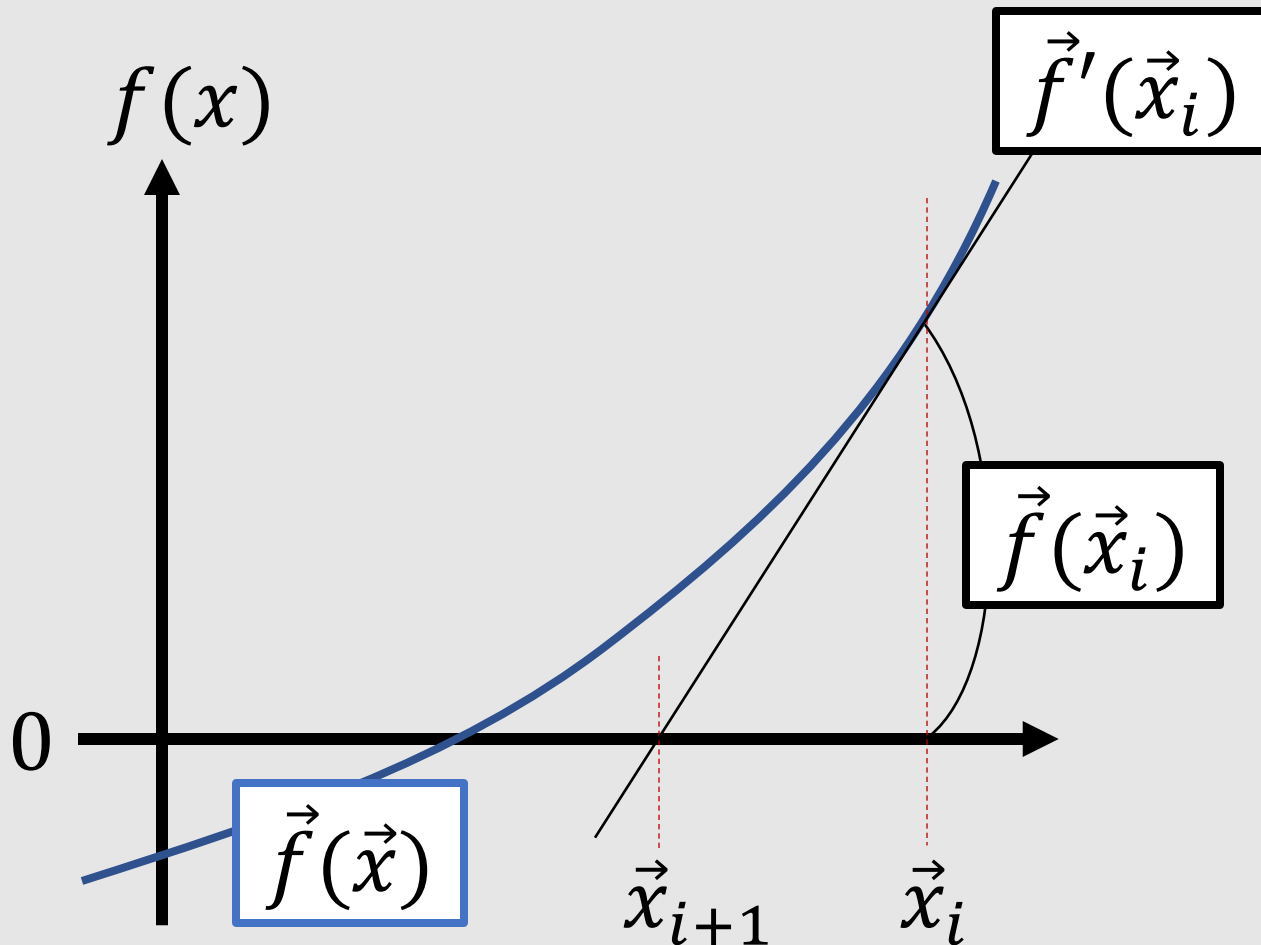


To find x where $f(x) = 0$

Iterate:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Finding the **Root** of a **Multivariate** Function



To find \vec{x} where $\vec{f}(\vec{x}) = 0$

Iterate:

$$\vec{x}_{i+1} = \vec{x}_i - \left[\nabla \vec{f}(\vec{x}_i) \right]^{-1} f(\vec{x}_i)$$

Jacobian matrix

* $\nabla \vec{f}(\vec{x}_i)$ need to be invertible

Finding the **Root of Gradient** $\nabla W(x) = 0$

- Gradient of gradient is called **hessian**

$$\vec{f} = \nabla W$$

To find \vec{x} where $\vec{f}(\vec{x}) = 0$

Iterate:

$$\vec{x}_{i+1} = \vec{x}_i - [\nabla \vec{f}(\vec{x}_i)]^{-1} \vec{f}(\vec{x}_i)$$

To find \vec{x} where $\nabla W(\vec{x}) = 0$

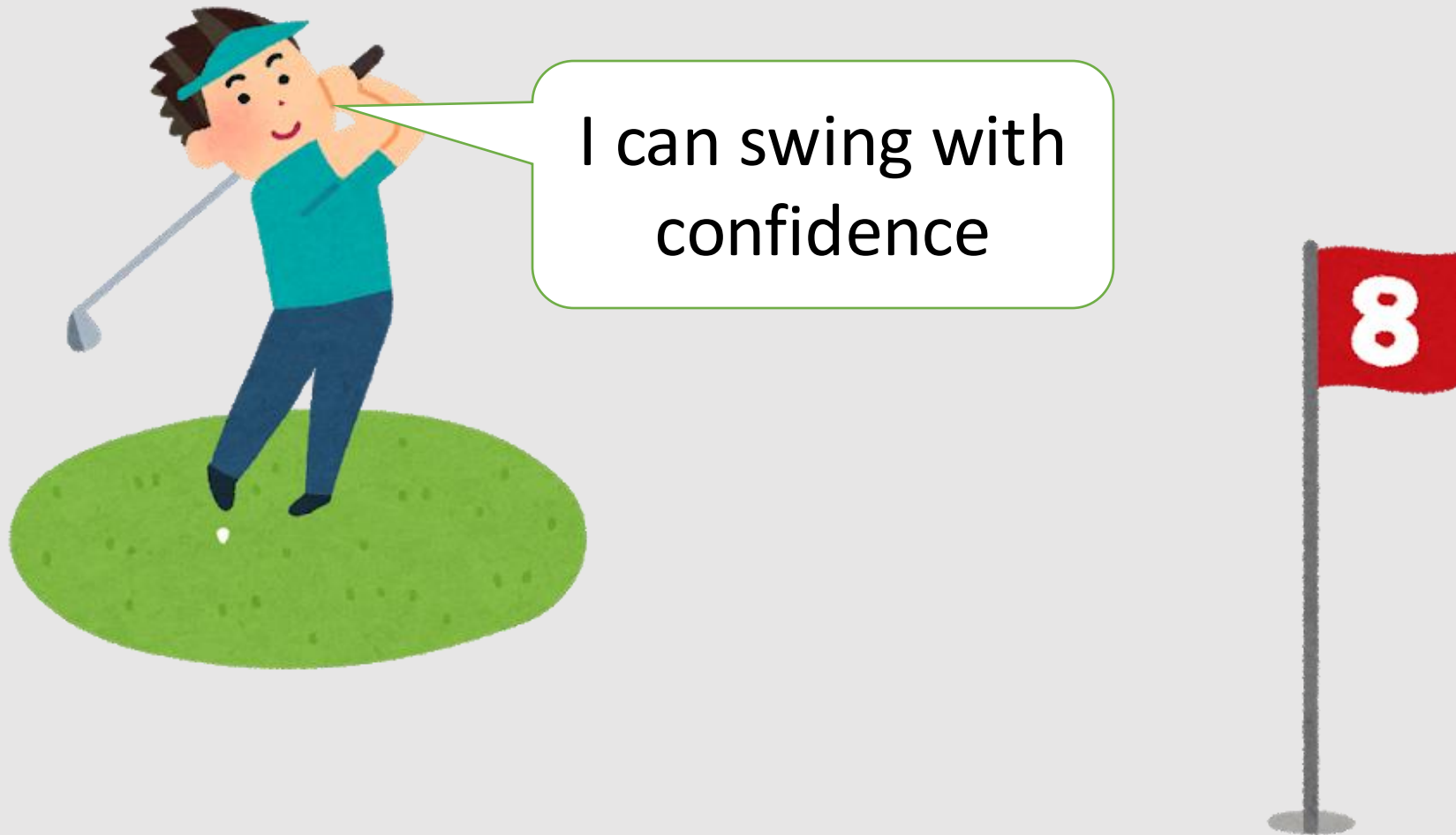
Iterate:

$$\vec{x}_{i+1} = \vec{x}_i - [\nabla^2 W(\vec{x}_i)]^{-1} \nabla W(\vec{x}_i)$$

hessian

Gradient Descent: Golf without Blindfold

- Optimizer know the direction & strength to hit



Comparison of Three Approaches



Stochastic Optimization

- ☺ Only evaluation of a function is necessary
- ☹ Very slow
- ☹ Not scalable
- ☹ Heuristics



Gradient Descent

- ☺ Only gradient is necessary
- ☺ Very scalable
- ☹ Slow
- ☹ Parameter tuning



Newton Method

- ☺ Very fast for almost quadratic problem
- ☹ Require Hessian
- ☹ Complicated Code

Advanced Topics

- Stochastic Optimization

- Metropolis Hasting Method
- Meta-heuristic Optimization (Particle Swarm, Evolutionary Algorithm)



- Gradient Descent

- Stochastic Gradient Descent



- Newton Method

- Levenberg–Marquardt method
- L-BFGS method



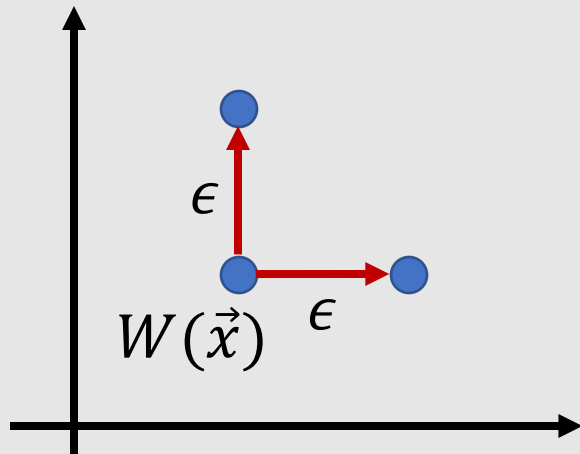
Typical Mistakes in Optimization

- Don't use **numerical difference** in gradient or Newton method

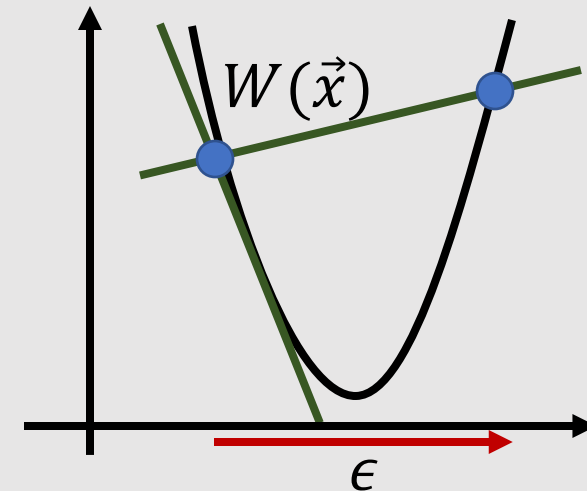
$$(\nabla W)_i = \frac{W(\vec{x} + \epsilon \vec{e}_i) - W(\vec{x})}{\epsilon}$$



Not scalable for large DoFs



Inaccurate around convergence



End