Learning From Data
Lecture 22
Neural Networks and Overfitting

Approximation vs. Generalization
Regularization and Early Stopping
Minimizing E_{in} More Efficienty ☺

M. Magdon-Ismail
CSCI 4100/6100

Neural Networks and Fitting the Data

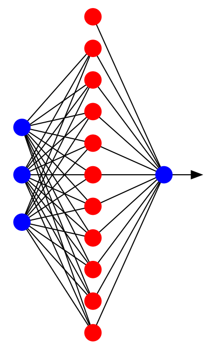
Forward Propagation:

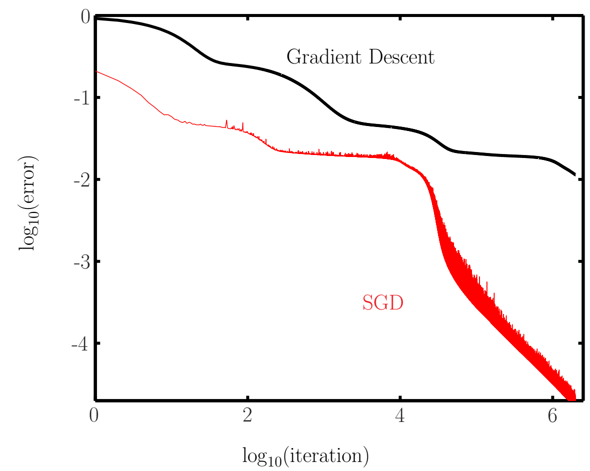
$$\mathbf{x} = \mathbf{x}^{(0)} \xrightarrow{\mathrm{W}^{(1)}} \mathbf{s}^{(1)} \xrightarrow{\theta} \mathbf{x}^{(1)} \xrightarrow{\mathrm{W}^{(2)}} \mathbf{s}^{(2)} \cdots \xrightarrow{\mathrm{W}^{(L)}} \mathbf{s}^{(L)} \xrightarrow{\theta} \mathbf{x}^{(L)} = h(\mathbf{x})$$

$$\mathbf{s}^{(\ell)} = (\mathrm{W}^{(\ell)})^{\mathrm{T}}\mathbf{x}^{(\ell-1)} \qquad \mathbf{x}^{(\ell)} = \begin{bmatrix} 1 \\ \theta(\mathbf{s}^{(\ell)}) \end{bmatrix}$$

(Compute h and E_{in})

Choose $\mathrm{W} = \{\mathrm{W}^{(1)}, \mathrm{W}^{(1)}, \ldots, \mathrm{W}^{(L)}\}$ to minimize E_{in}



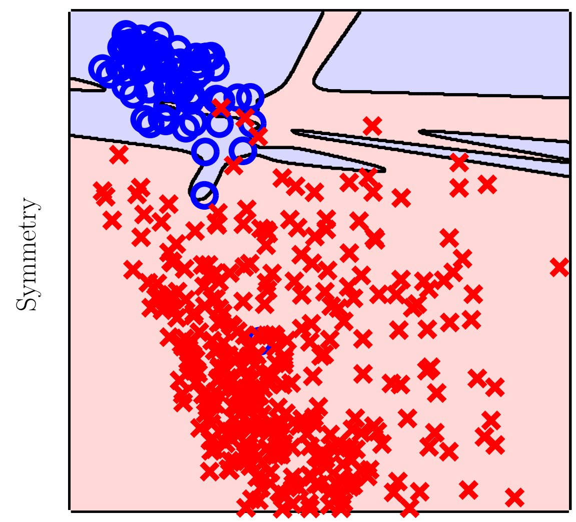


Gradient descent:

$$\mathrm{W}(t+1) \leftarrow \mathrm{W}(t) - \eta\nabla E_{\text{in}}(\mathrm{W}(t))$$

Compute gradient $\longrightarrow$ need $\dfrac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}}$ $\longrightarrow$ need $\boldsymbol{\delta}^{(\ell)} = \dfrac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}$

$$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$



Average Intensity

Backpropagation:

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

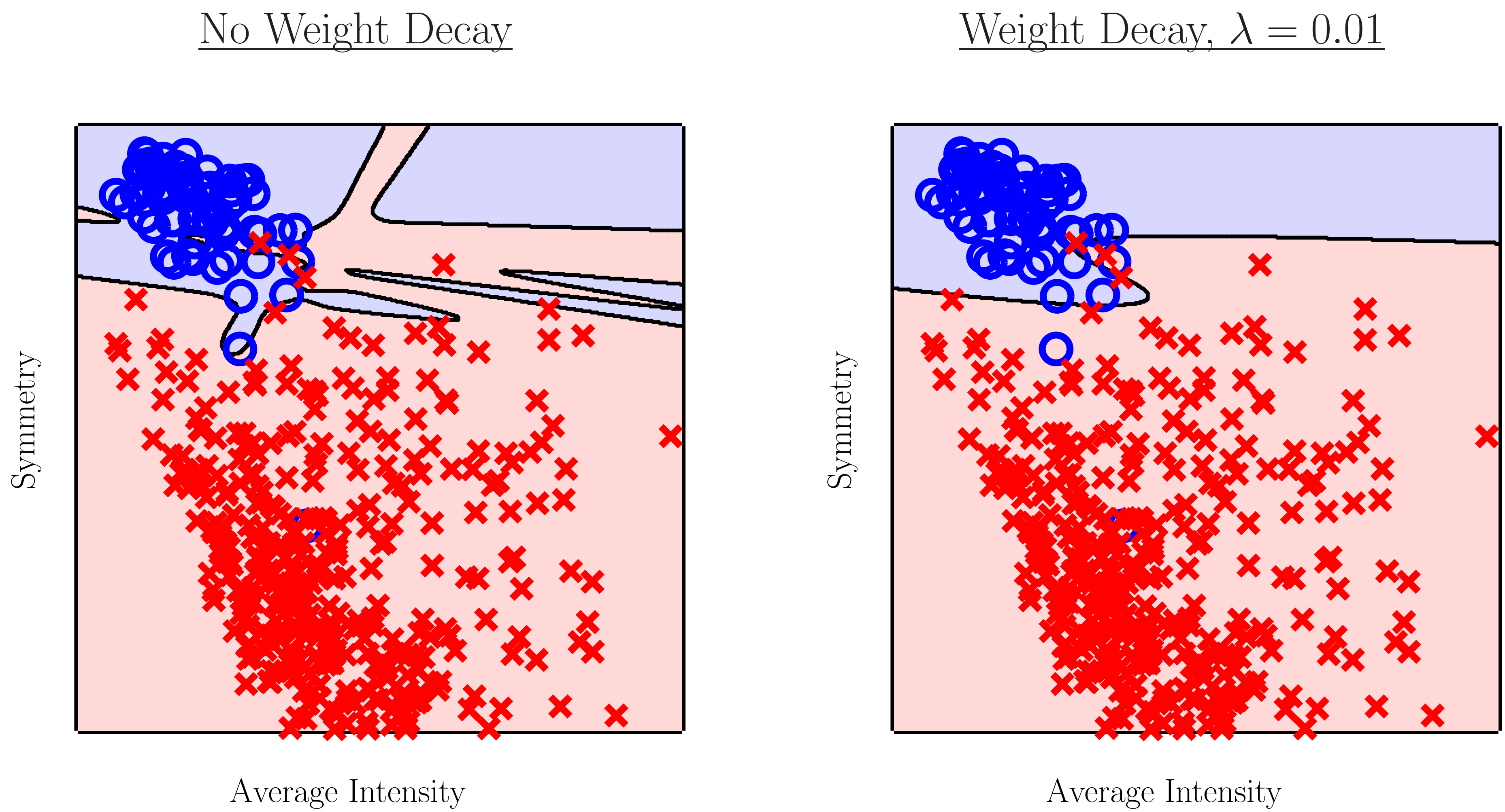
$$\boldsymbol{\delta}^{(\ell)} = \theta'(\mathbf{s}^{(\ell)}) \otimes \left[\mathrm{W}^{(\ell+1)}\boldsymbol{\delta}^{(\ell+1)}\right]_1^{d^{(\ell)}}$$

Regularization – Weight Decay

$$E_{\text{aug}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(h(\mathbf{x}_n;\mathbf{w}) - y_n)^2 + \frac{\lambda}{N}\sum_{\ell,i,j}(w_{ij}^{(\ell)})^2$$

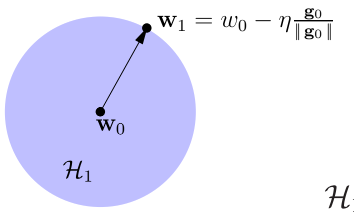
$$\frac{\partial E_{\text{aug}}(\mathbf{w})}{\partial \mathrm{W}^{(\ell)}} = \underbrace{\frac{\partial E_{\text{in}}(\mathbf{w})}{\partial \mathrm{W}^{(\ell)}}}_{\uparrow \atop \text{backpropagation}} + \frac{2\lambda}{N}\mathrm{W}^{(\ell)}$$

Weight Decay with Digits Data

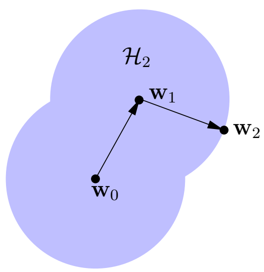


Early Stopping

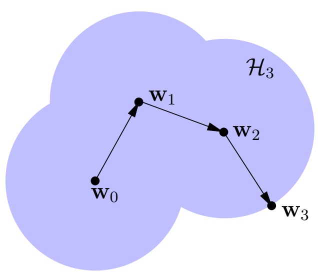
Gradient Descent



$$\mathcal{H}_1 = \{\mathbf{w} : \ \|\mathbf{w} - \mathbf{w}_0\| \le \eta\}$$



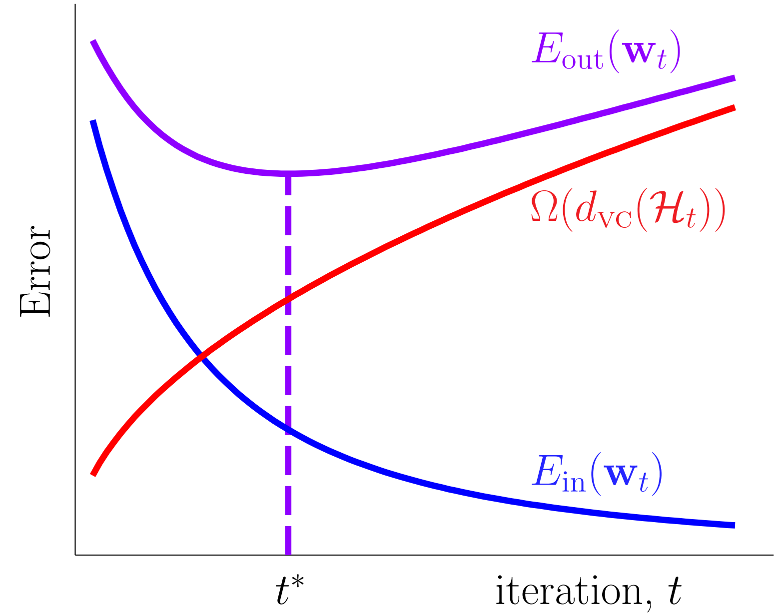
$$\mathcal{H}_2 = \mathcal{H}_1 \cup \{\mathbf{w} : \ \|\mathbf{w} - \mathbf{w}_1\| \le \eta\}$$

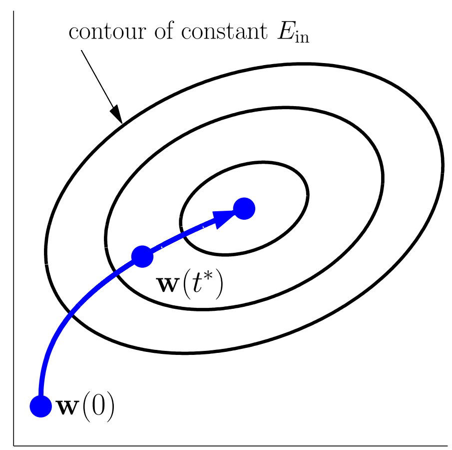


$$\mathcal{H}_3 = \mathcal{H}_2 \cup \{\mathbf{w} : \ \|\mathbf{w} - \mathbf{w}_2\| \le \eta\}$$

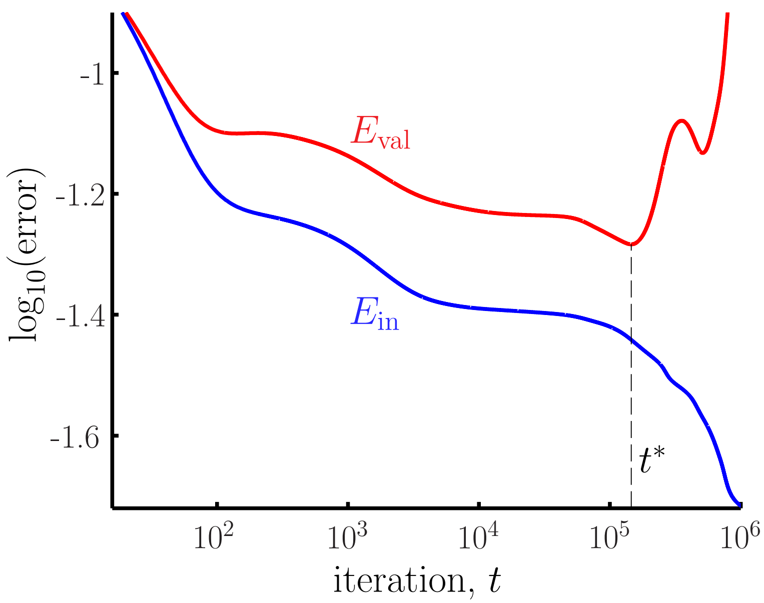
Each iteration explores a larger $\mathcal{H}$

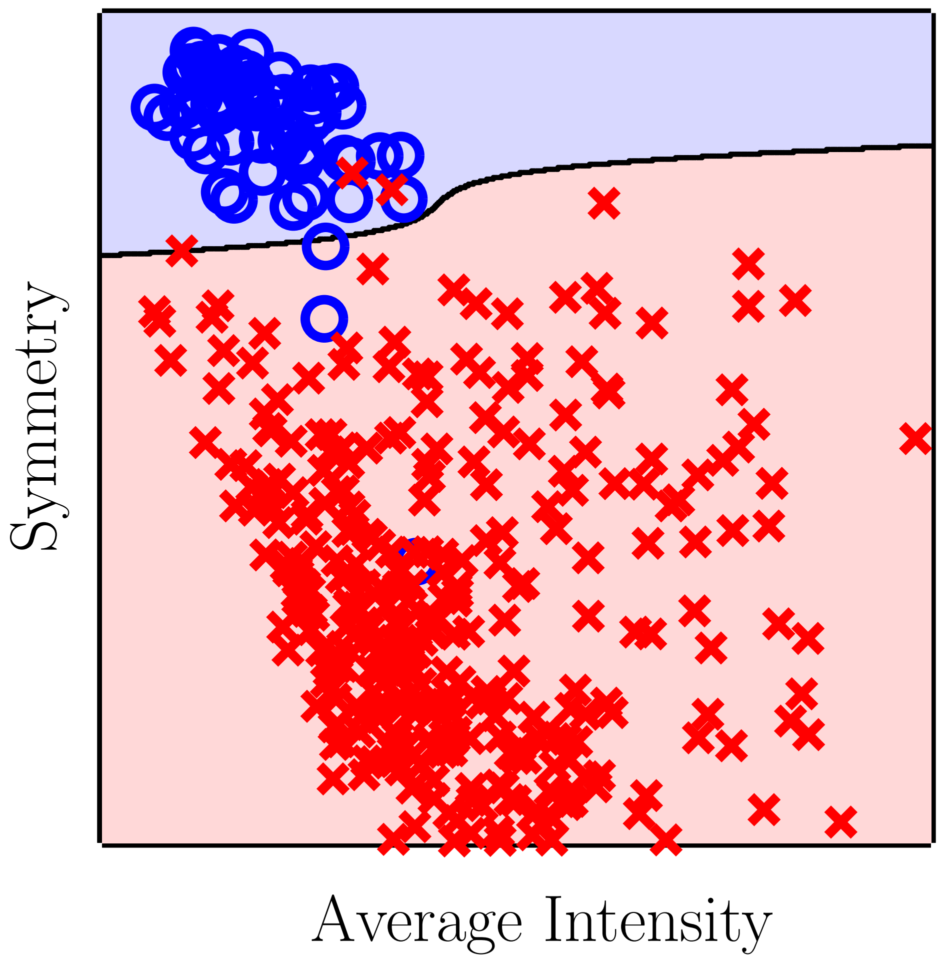
$$\mathcal{H}_1 \subset \mathcal{H}_2 \subset \mathcal{H}_3 \subset \mathcal{H}_4 \subset \cdots$$





Early Stopping on Digits Data

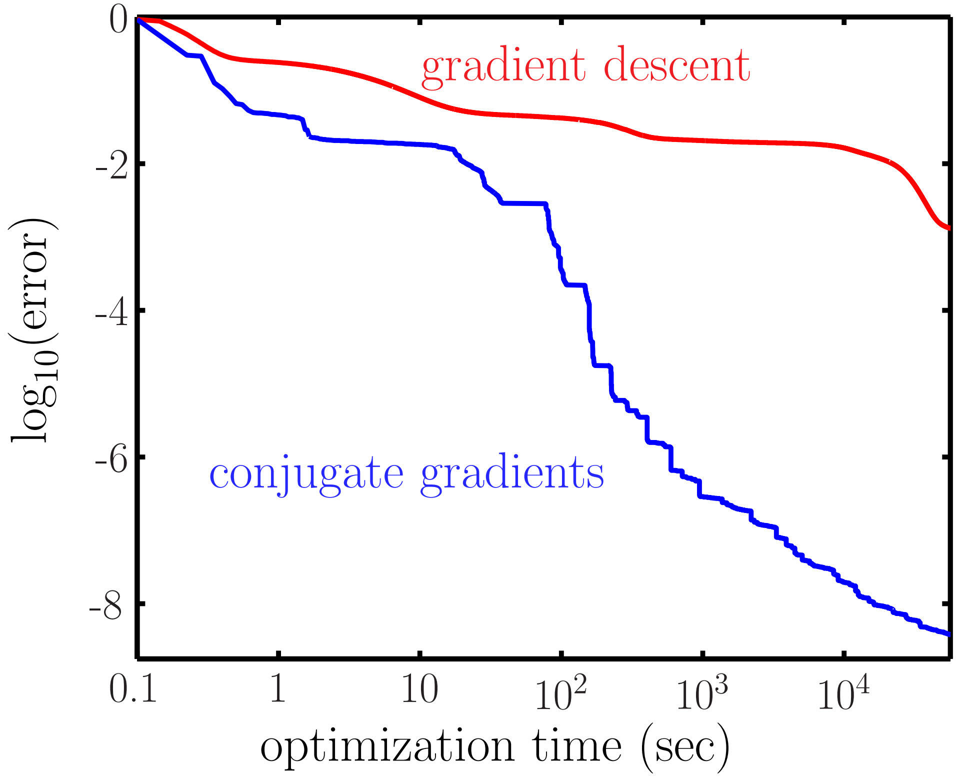




Use a validation set to determine t^*
Output $\mathbf{w}^*$, do not retrain with all the data till t^*.

Minimizing E_{in}

1. Use regression for classification
2. Use better algorithms than gradient descent



Variable Learning Rate Gradient Descent

Initialize $\mathbf{w}(0)$, and η_0 at $t = 0$. Set $\alpha > 1$ and $\beta < 1$.

while stopping criterion has not been met **do**

  Let $\mathbf{g}(t) = \nabla E_{\text{in}}(\mathbf{w}(t))$, and set $\mathbf{v}(t) = -\mathbf{g}(t)$.

  if $E_{\text{in}}(\mathbf{w}(t) + \eta_t \mathbf{v}(t)) < E_{\text{in}}(\mathbf{w}(t))$ **then**

    accept: $\mathbf{w}(t+1) = \mathbf{w}(t) + \eta_t \mathbf{v}(t)$;

    increment η: $\eta_{t+1} = \alpha \eta_t$. $\alpha \in [1.05, 1.1]$

  else

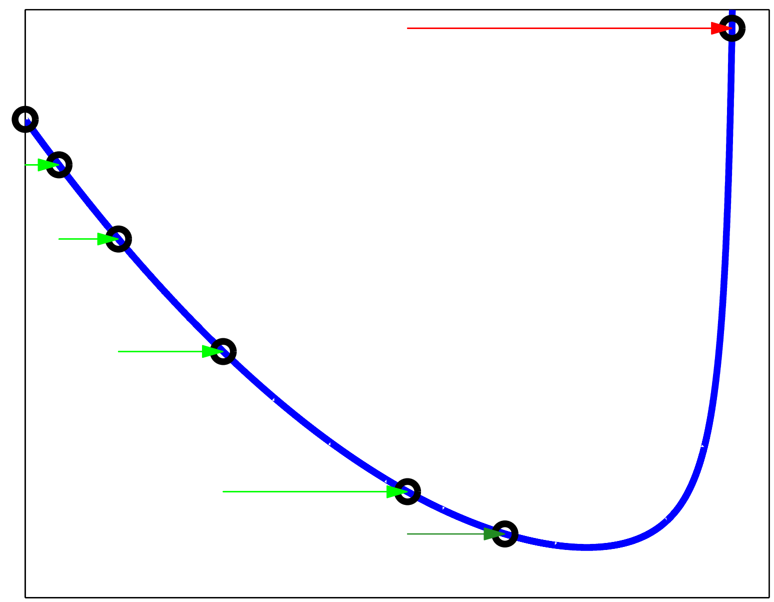
    reject: $\mathbf{w}(t+1) = \mathbf{w}(t)$;

    decrease η: $\eta_{t+1} = \beta \eta_t$. $\beta \in [0.7, 0.8]$

  end if

  Iterate to the next step, $t \leftarrow t + 1$.

end while



Steepest Descent - Line Search ⟶

Steepest Descent - Line Search

1: Initialize $w(0)$ and set $t = 0$;

2: **while** stopping criterion has not been met **do**

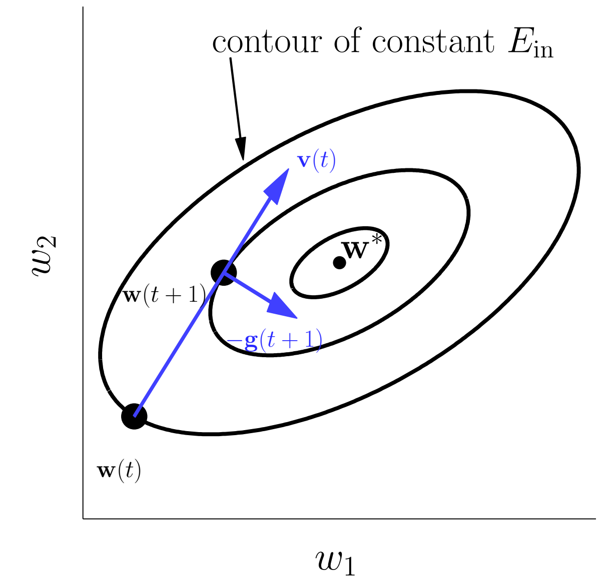
3: Let $\mathbf{g}(t) = \nabla E_{\text{in}}(\mathbf{w}(t))$, and set $\mathbf{v}(t) = -\mathbf{g}(t)$.

4: Let $\eta^* = \text{argmin}_\eta\, E_{\text{in}}(\mathbf{w}(t) + \eta\mathbf{v}(t))$.

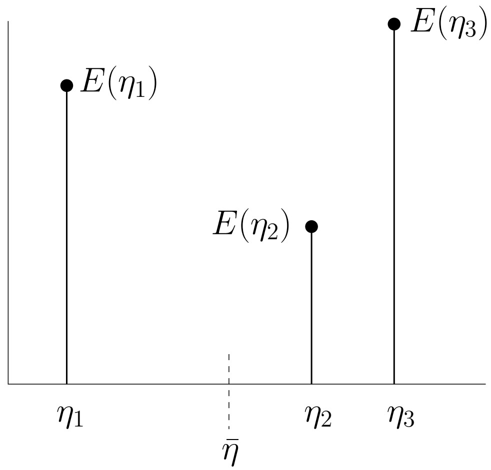
5: $\mathbf{w}(t+1) = \mathbf{w}(t) + \eta^*\mathbf{v}(t)$.

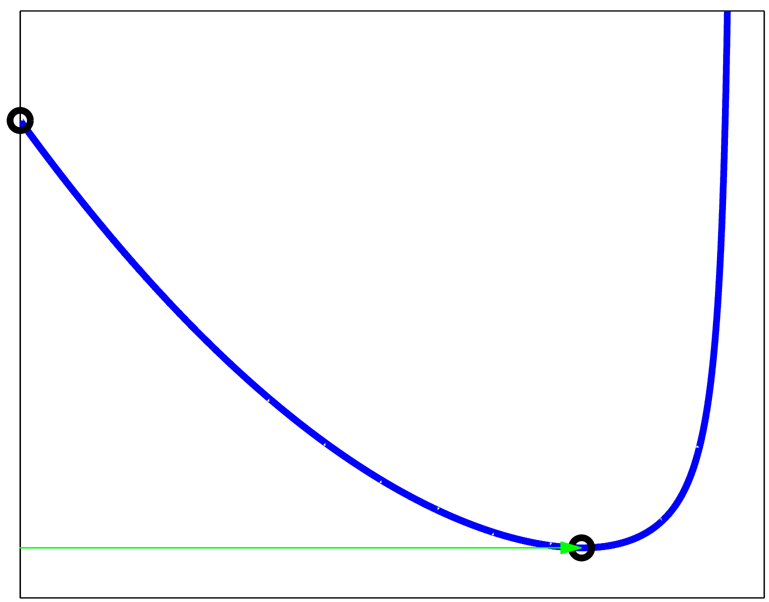
6: Iterate to the next step, $t \leftarrow t + 1$.

7: **end while**

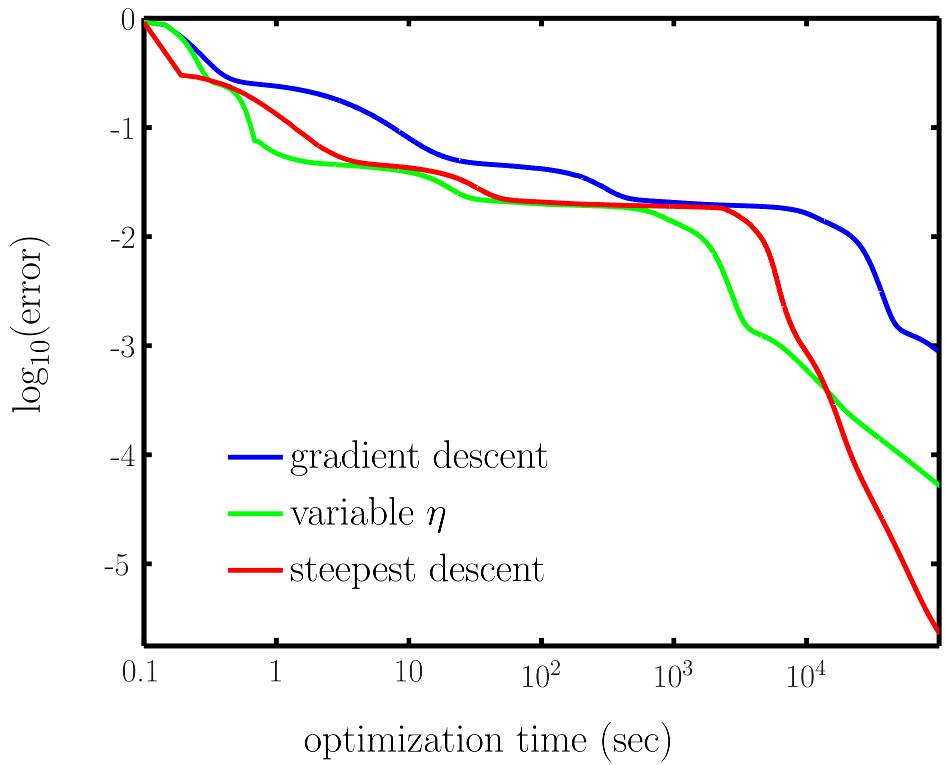


How to accomplish the line search (step 4)?
Simple bisection (binary search) suffices in practice





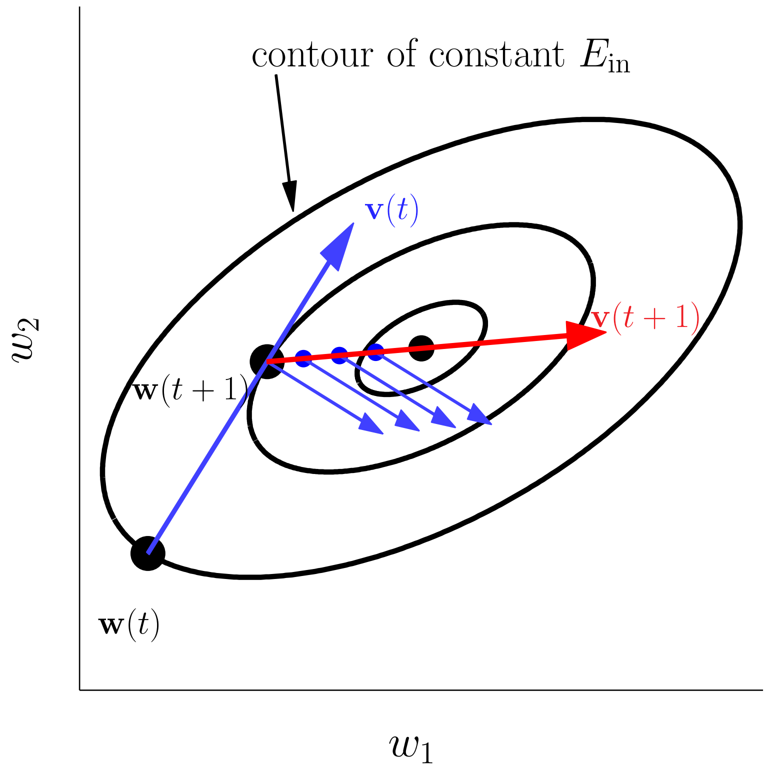
Comparison of Optimization Heuristics

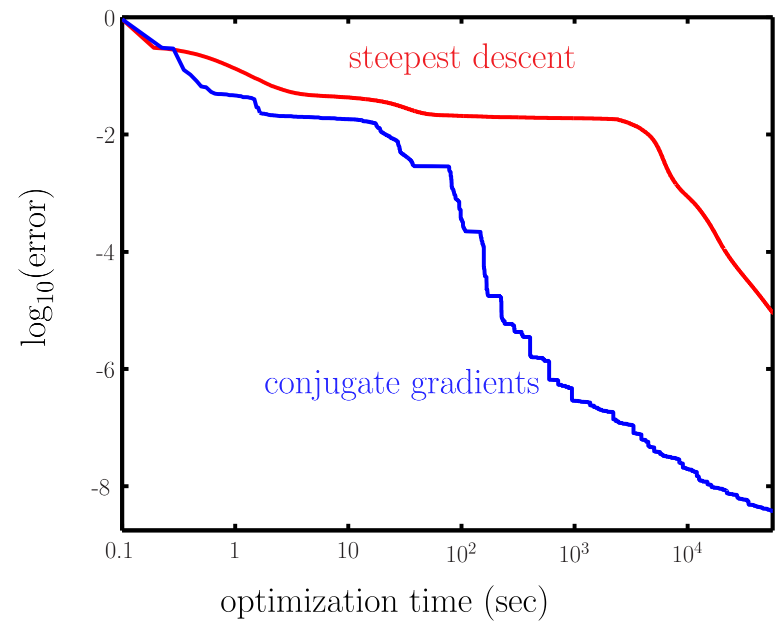


	Optimization Time		
Method	10 sec	1,000 sec	50,000 sec
Gradient Descent	0.122	0.0214	0.0113
Stochastic Gradient Descent	**0.0203**	**0.000447**	$\mathbf{1.6310 \times 10^{-5}}$
Variable Learning Rate	0.0432	0.0180	0.000197
Steepest Descent	0.0497	0.0194	0.000140

Conjugate Gradients

1. Line search just like steepest descent.
2. Choose a better direction than $-\mathbf{g}$





Method	Optimization Time 10 sec	1,000 sec	50,000 sec
Stochastic Gradient Descent	0.0203	0.000447	1.6310×10^{-5}
Steepest Descent	0.0497	0.0194	0.000140
Conjugate Gradients	**0.0200**	$\mathbf{1.13 \times 10^{-6}}$	$\mathbf{2.73 \times 10^{-9}}$

There are better algorithms (eg. Levenberg-Marquardt), but we will stop here ☹