

CMPUT 267: Machine Learning I · Fall 2026

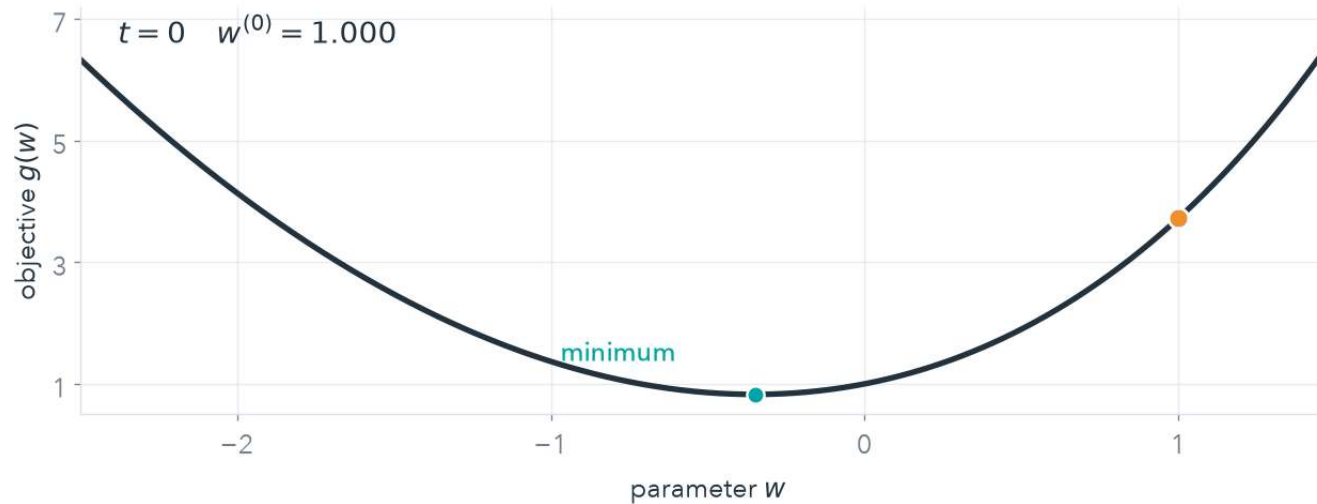
Gradient Descent Review and Newton's Method

Xi Ye

2026-10-13

Recall: Gradient Descent

At the current parameters, use the gradient to choose a downhill direction.



$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta^{(t)} \nabla g(\mathbf{w}^{(t)})$$

Batch, Mini-Batch, and Stochastic GD

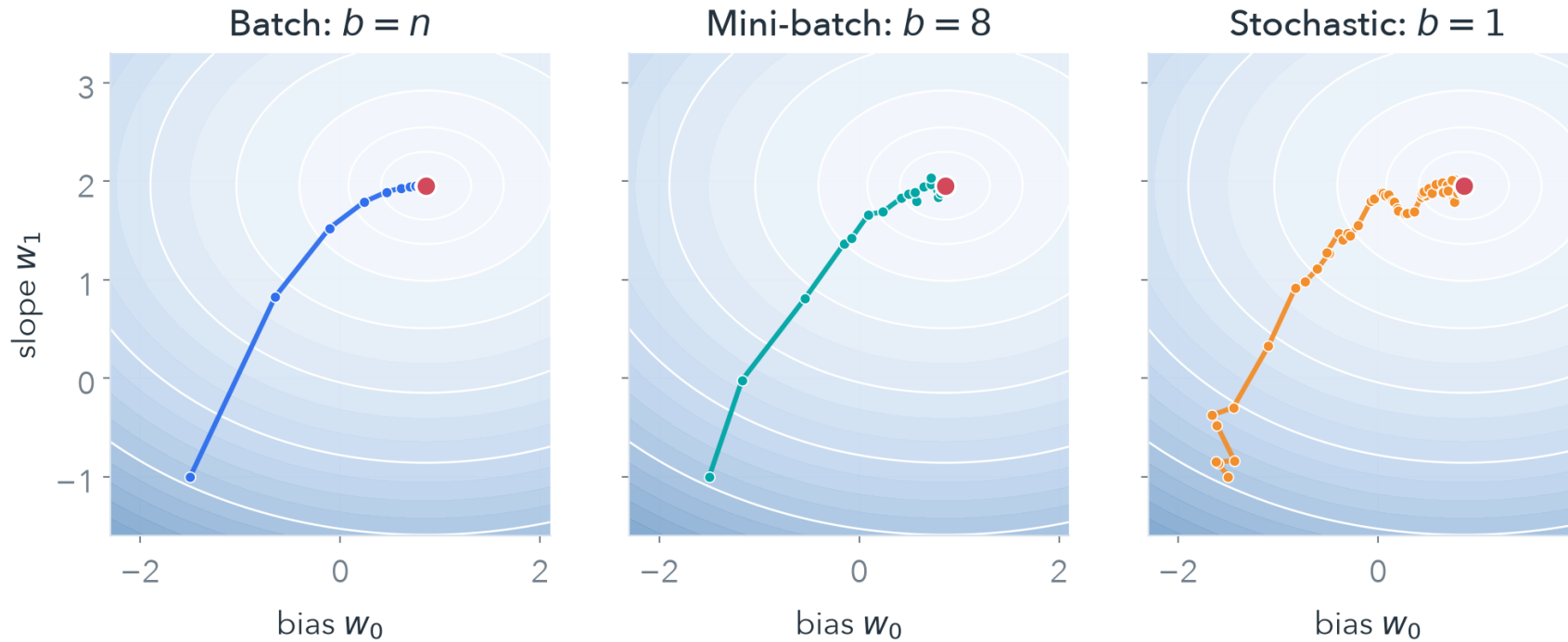
Let B_t be the examples used at update t , with batch size $b = |B_t|$.

$$\nabla \hat{L}_{B_t}(\mathbf{w}^{(t)}) = \frac{1}{b} \sum_{i \in B_t} \nabla \ell_i(\mathbf{w}^{(t)}), \quad \mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta^{(t)} \nabla \hat{L}_{B_t}(\mathbf{w}^{(t)}).$$

Method	Examples in one update	Batch size	Updates per epoch
Batch GD	The full training set	$b = n$	1
Mini-batch GD	A subset of the training set	$1 < b < n$	About n/b
Stochastic GD	One training example	$b = 1$	n

i The update rule stays the same. The batch size changes the gradient estimate and how often the parameters update.

Batch Size Changes the Search



Batch GD follows a stable full-data direction. Smaller batches make cheaper, more frequent updates with greater variation.

Recall: Polynomial Regression

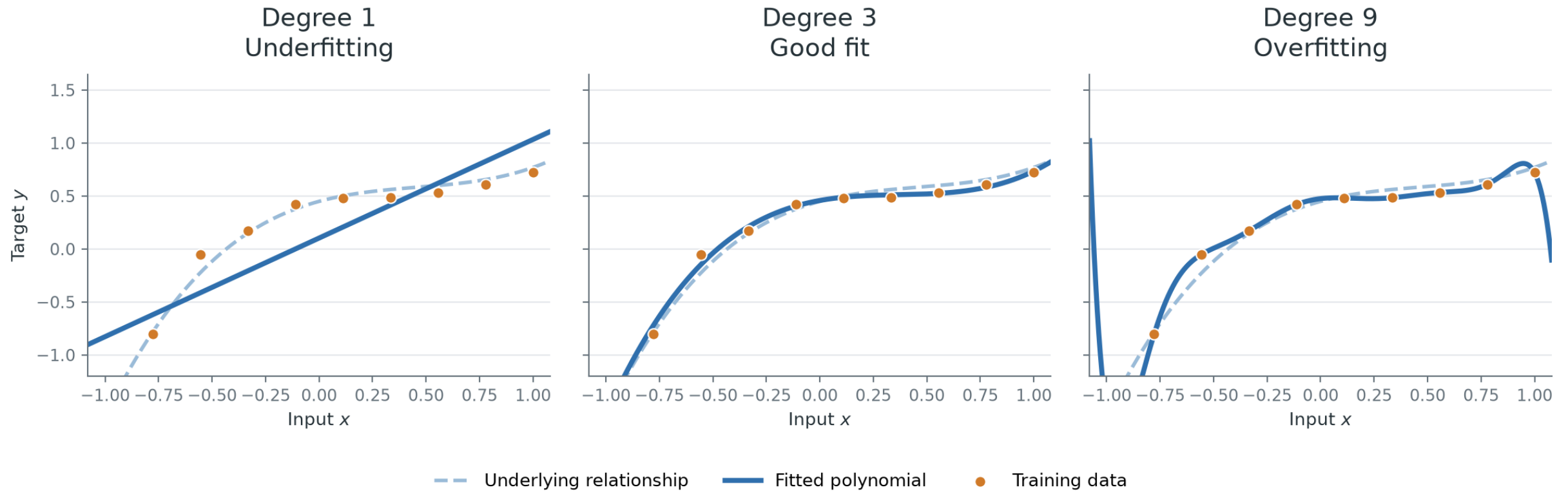
A polynomial feature map adds powers and interactions to the input x .

$$\phi_2(x) = \begin{pmatrix} 1 \\ x \\ x^2 \end{pmatrix}, \quad f_{\mathbf{w}}(x) = \mathbf{w}^\top \phi_2(x) = w_0 + w_1x + w_2x^2.$$

$$\hat{L}_p(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n (\phi_p(\mathbf{x}_i)^\top \mathbf{w} - y_i)^2.$$

- i The predictor is non-linear as a function of $\mathbf{x}$, but is still linear as a function of w . After feature mapping, this is still a **linear regression problem**.

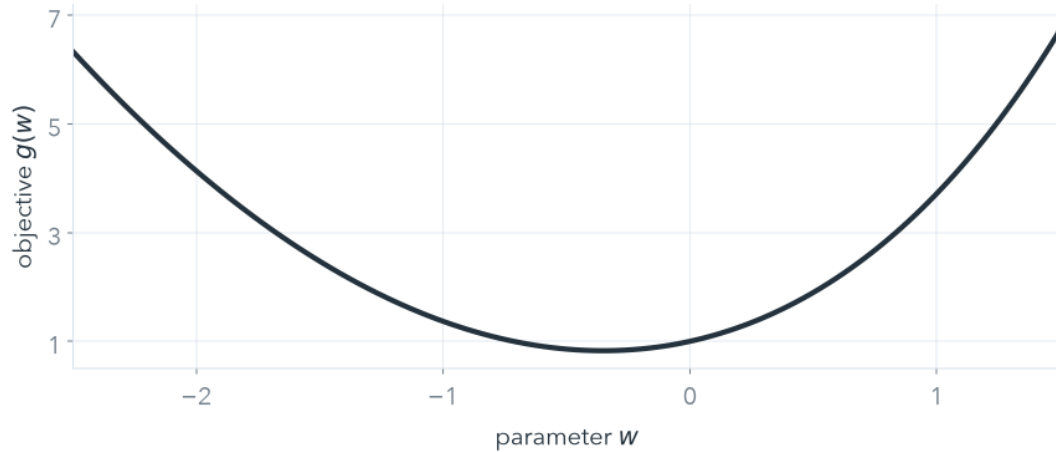
Underfitting, Good Fit, and Overfitting



⚠ Higher degree fits data better but could result overfitting (not generalizing).

Newton's Method

A Convex Objective Without a Closed-Form Minimizer



Recall the example:

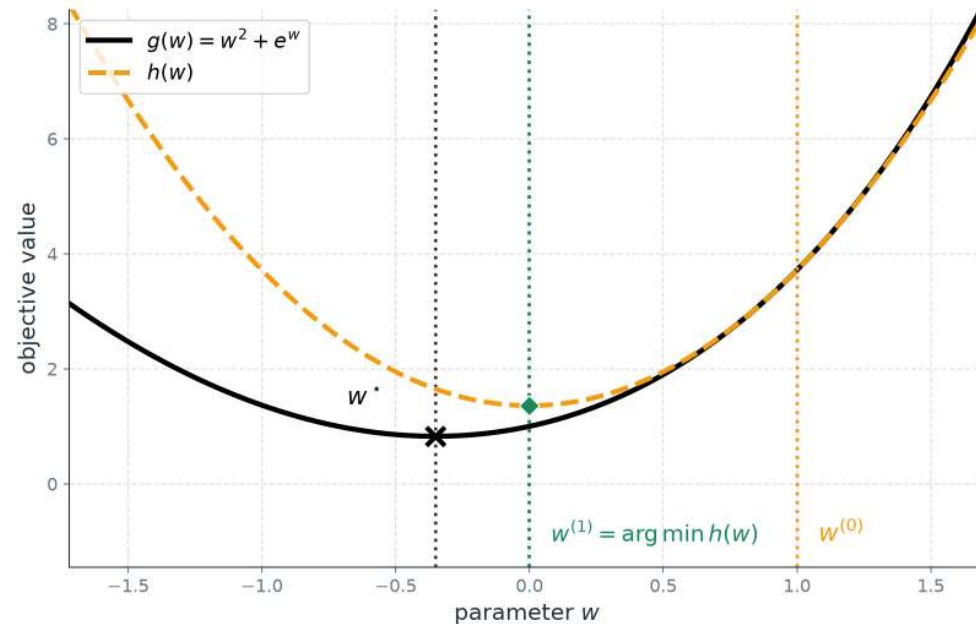
$$g(w) = w^2 + e^w$$

$$g'(w) = 2w + e^w$$



The unique minimizer solves $2w + e^w = 0$, which has no solution using only elementary functions (no closed form solution).

Solve by Approximation



Near a chosen point $w^{(0)}$, find a simpler function h such that

$$h(w) \approx g(w) \quad \text{when } w \text{ is near } w^{(0)}.$$

Choose h so that its minimizer has a closed-form expression.

$$w^{(1)} = \arg \min_w h(w) \quad \text{approximates} \quad w^* = \arg \min_w g(w).$$

i A second-order Taylor expansion gives us such a function $h(w)$.

Taylor Expansion Refresher

A Taylor expansion approximates a smooth function near a chosen point $w^{(0)}$ using a polynomial. Let $\Delta = w - w^{(0)}$.

$$\begin{aligned} T_n(w) &= \sum_{k=0}^n \frac{g^{(k)}(w^{(0)})}{k!} (w - w^{(0)})^k \\ &= g(w^{(0)}) + g'(w^{(0)})\Delta + \frac{g''(w^{(0)})}{2!} \Delta^2 + \dots + \frac{g^{(n)}(w^{(0)})}{n!} \Delta^n \end{aligned}$$

For degree two

$$T_2(w) = \underbrace{g(w^{(0)})}_{\text{value}} + \underbrace{g'(w^{(0)})\Delta}_{\text{slope}} + \underbrace{\frac{1}{2}g''(w^{(0)})\Delta^2}_{\text{second derivative}}.$$

Taylor Expansion Around a Point

Choose an expansion point $w^{(0)}$ and write $\Delta = w - w^{(0)}$.

$$g(w) \approx g(w^{(0)}) + g'(w^{(0)})\Delta + \frac{1}{2!}g''(w^{(0)})\Delta^2$$

Expand to:

$$\begin{aligned} h(w) = & g(w^{(0)}) + g'(w^{(0)})(w - w^{(0)}) \\ & + \frac{1}{2}g''(w^{(0)})(w - w^{(0)})^2. \end{aligned}$$

Minimizing the Quadratic Approximation

Differentiate $h(w)$ with respect to w :

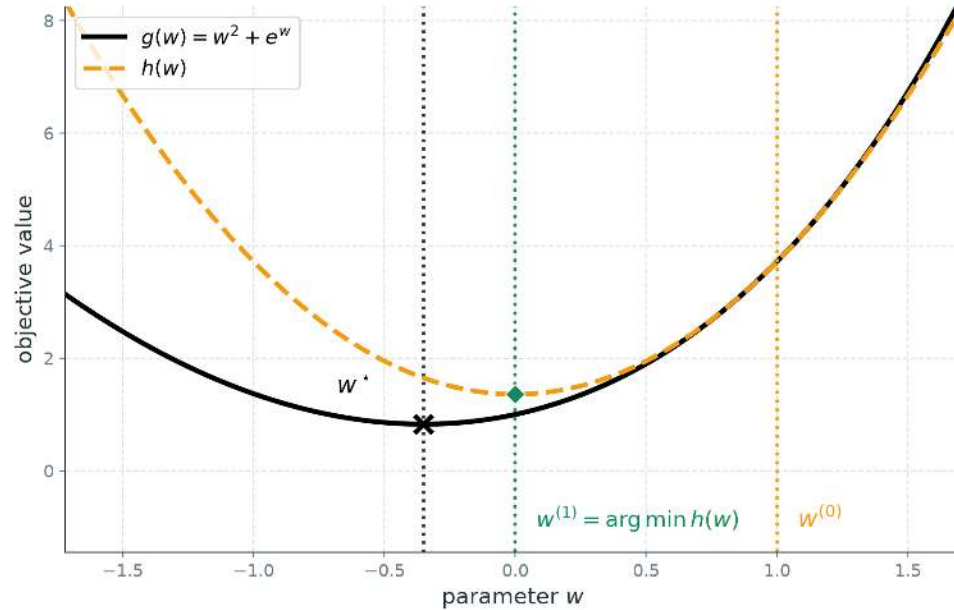
$$h'(w) = g'(w^{(0)}) + g''(w^{(0)})(w - w^{(0)}).$$

Set this derivative to zero at the minimizer of the quadratic:

$$0 = g'(w^{(0)}) + g''(w^{(0)})(w^{(1)} - w^{(0)}).$$

$$w^{(1)} = w^{(0)} - \frac{g'(w^{(0)})}{g''(w^{(0)})}$$

Example: Newton Step



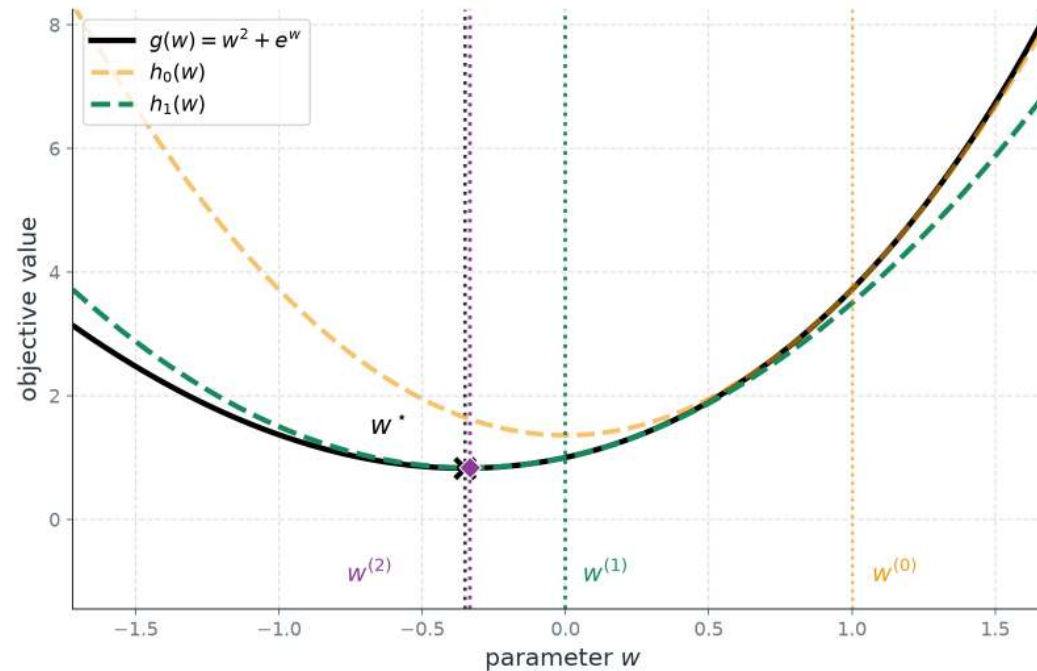
For $g(w) = w^2 + e^w$, start at $w^{(0)} = 1$.

$$\begin{aligned} w^{(1)} &= w^{(0)} - \frac{g'(w^{(0)})}{g''(w^{(0)})} \\ &= 1 - \frac{g'(1)}{g''(1)} \\ &= 1 - \frac{2 + e}{2 + e} = 0. \end{aligned}$$

One quadratic approximation moves the search from 1 to 0.

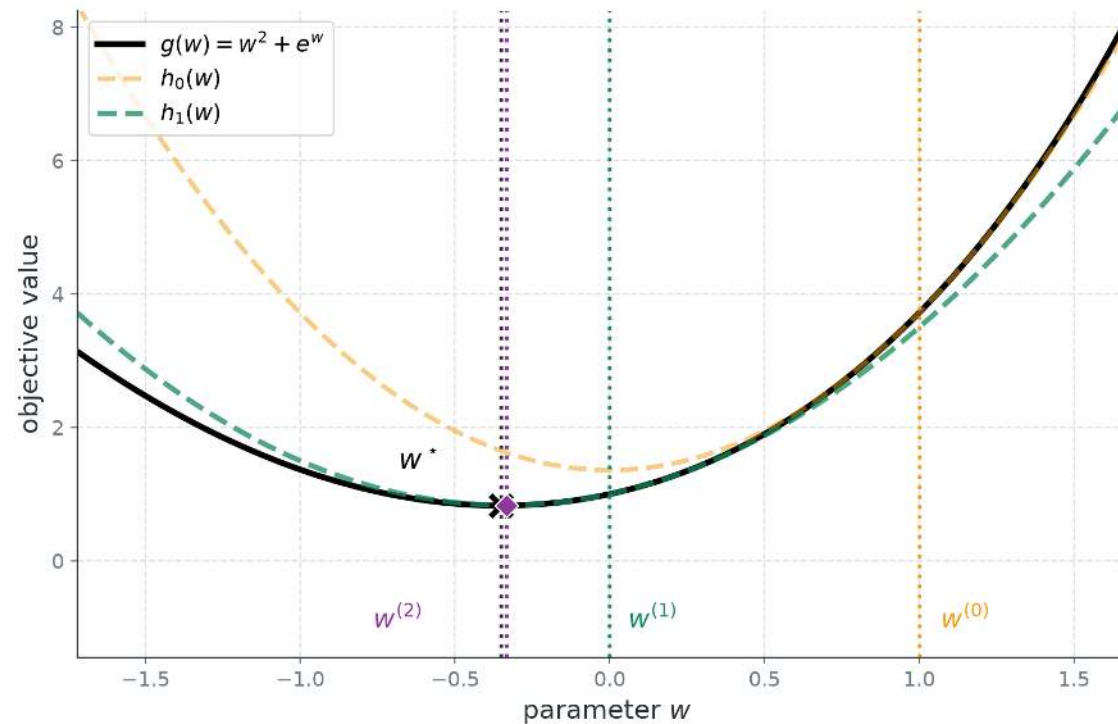
Rebuild the Approximation

At $w^{(1)} = 0$, construct a new Taylor quadratic $h_1(w)$.



$$w^{(2)} = 0 - \frac{g'(0)}{g''(0)} = -\frac{1}{3}.$$

Newton Iterates



$$w^{(0)} = 1, \quad w^{(1)} = 0, \quad w^{(2)} = -\frac{1}{3}, \quad w^* \approx -0.3517.$$



Each iteration matches a new quadratic at the latest point. Near this minimizer, the approximations become increasingly accurate.

Newton's Method in One Dimension

Input: a twice differentiable objective g , an initial point $w^{(0)}$, and a tolerance ϵ

1. Construct the Taylor quadratic $h_t(w)$ around $w^{(t)}$.

2. Minimize h_t in closed form:

$$w^{(t+1)} = \arg \min_w h_t(w) = w^{(t)} - \frac{g'(w^{(t)})}{g''(w^{(t)})}.$$

3. Stop when $|g(w^{(t+1)}) - g(w^{(t)})| < \epsilon$.

 The basic step requires $g''(w^{(t)}) \neq 0$. A positive second derivative makes the quadratic approximation locally convex.

First-Order and Second-Order Updates

Gradient descent

$$w^{(t+1)} = w^{(t)} - \eta^{(t)} g'(w^{(t)})$$

Uses the **first derivative** and a chosen learning rate.

Newton's method

$$w^{(t+1)} = w^{(t)} - \frac{g'(w^{(t)})}{g''(w^{(t)})}$$

Uses the **first and second derivatives**. The second derivative determines the step scale.

 Gradient descent is a first-order method. Newton's method is a second-order method.

The Second Derivative Controls the Step

$$\text{Newton step} = -\frac{g'(w^{(t)})}{g''(w^{(t)})}.$$

Large positive second derivative

The objective bends sharply. Dividing by a large $g''(w^{(t)})$ produces a smaller step.

Small positive second derivative

The objective is flatter. Dividing by a small $g''(w^{(t)})$ produces a larger step.

 The quadratic model supplies a local scale for the update.

Second Derivatives in Several Dimensions

For $\mathbf{w} \in \mathbb{R}^d$, the **Hessian** collects every second partial derivative.

$$H_g(\mathbf{w}) = \begin{pmatrix} \frac{\partial^2 g}{\partial w_1^2} & \cdots & \frac{\partial^2 g}{\partial w_1 \partial w_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 g}{\partial w_d \partial w_1} & \cdots & \frac{\partial^2 g}{\partial w_d^2} \end{pmatrix}.$$

- i Diagonal entries are second partial derivatives with respect to one coordinate. Off-diagonal entries are mixed second partial derivatives involving two coordinates.

Multivariable Newton Update

The local quadratic approximation is

$$q_t(\mathbf{w}) = g(\mathbf{w}^{(t)}) + \nabla g(\mathbf{w}^{(t)})^\top (\mathbf{w} - \mathbf{w}^{(t)}) + \frac{1}{2} (\mathbf{w} - \mathbf{w}^{(t)})^\top H_g(\mathbf{w}^{(t)}) (\mathbf{w} - \mathbf{w}^{(t)}).$$

$$\boxed{\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - H_g(\mathbf{w}^{(t)})^{-1} \nabla g(\mathbf{w}^{(t)})}$$

i In an implementation, solve $H_g(\mathbf{w}^{(t)})\mathbf{s}^{(t)} = \nabla g(\mathbf{w}^{(t)})$ and set $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \mathbf{s}^{(t)}$.
Avoid forming the inverse explicitly.

Cost of Second-Order Information

For d parameters:

Quantity	Size	Typical dense cost
Gradient $\nabla g(\mathbf{w})$	d entries	Problem dependent
Hessian $H_g(\mathbf{w})$	$d \times d$ entries	$O(d^2)$ storage
Newton system $H\mathbf{s} = \nabla g$	d unknowns	$O(d^3)$ solve

 Newton steps can be powerful when d is moderate and the Hessian is well behaved. The storage and solve become expensive for large models.

Wrap-up

- Batch, mini-batch, and stochastic gradient descent use different batch sizes in the same update rule.
- Polynomial degree can produce underfitting, a good fit, or overfitting.
- Newton's method comes from minimizing a second-order Taylor approximation.
- In several dimensions, the Hessian provides second-order information and also creates the main computational cost.

A Second Newton Example

Consider the objective

$$g(w) = w^4 + w^2 - w.$$

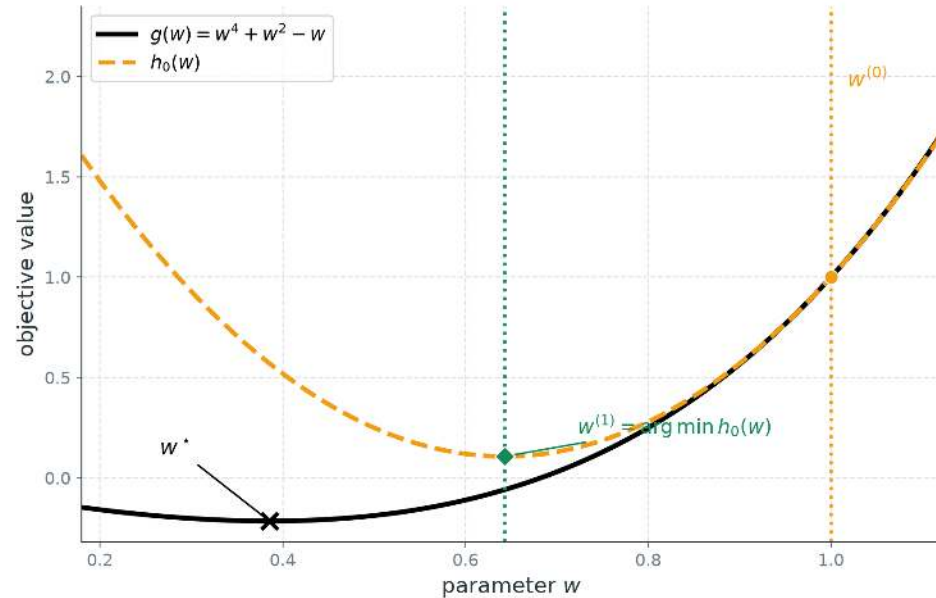
Its first two derivatives are

$$g'(w) = 4w^3 + 2w - 1, \quad g''(w) = 12w^2 + 2 > 0.$$

$$w^{(t+1)} = w^{(t)} - \frac{4(w^{(t)})^3 + 2w^{(t)} - 1}{12(w^{(t)})^2 + 2}.$$

i The objective is strictly convex, so it has one minimizer. Start at $w^{(0)} = 1$.

Newton Example: Step 1



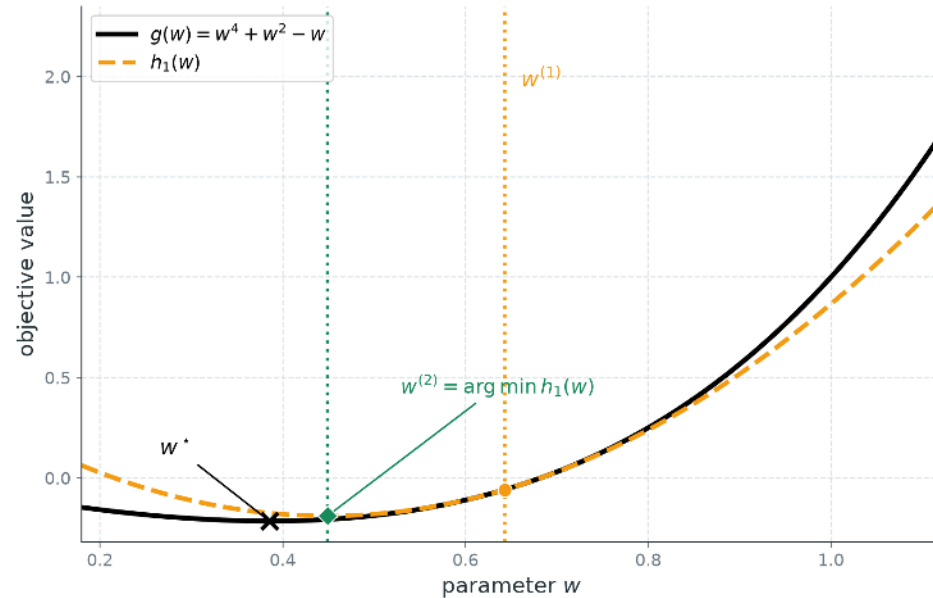
At $w^{(0)} = 1$,

$$g'(1) = 5, \quad g''(1) = 14.$$

$$\begin{aligned} w^{(1)} &= w^{(0)} - \frac{g'(w^{(0)})}{g''(w^{(0)})} \\ &= 1 - \frac{5}{14} = \frac{9}{14} \approx 0.6429. \end{aligned}$$

i The objective decreases from $g(w^{(0)}) = 1.0000$ to $g(w^{(1)}) \approx -0.0588$.

Newton Example: Step 2



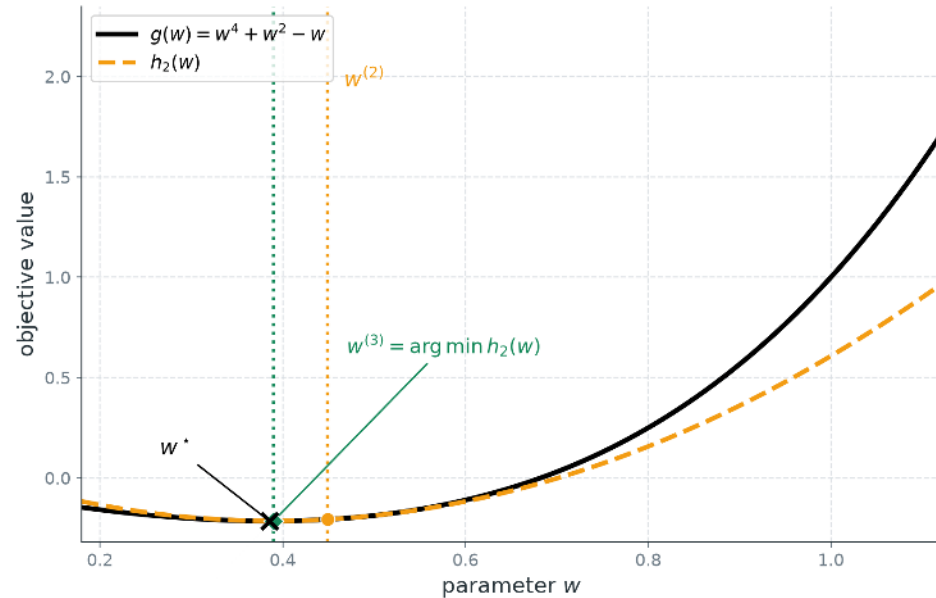
At $w^{(1)} \approx 0.6429$,

$$g'(w^{(1)}) \approx 1.3484, \quad g''(w^{(1)}) \approx 6.9592.$$

$$\begin{aligned} w^{(2)} &= w^{(1)} - \frac{g'(w^{(1)})}{g''(w^{(1)})} \\ &\approx 0.6429 - \frac{1.3484}{6.9592} \approx 0.4491. \end{aligned}$$

i The objective decreases from $g(w^{(1)}) \approx -0.0588$ to $g(w^{(2)}) \approx -0.2067$.

Newton Example: Step 3



At $w^{(2)} \approx 0.4491$,

$$g'(w^{(2)}) \approx 0.2605, \quad g''(w^{(2)}) \approx 4.4203.$$

$$\begin{aligned} w^{(3)} &= w^{(2)} - \frac{g'(w^{(2)})}{g''(w^{(2)})} \\ &\approx 0.4491 - \frac{0.2605}{4.4203} \approx 0.3902. \end{aligned}$$

i After three steps, $w^{(3)} \approx 0.3902$ is close to the minimizer $w^* \approx 0.3855$.