Optimisation Basics 3 — Practice Questions

CM52054: Foundational Machine Learning
Practice set with fully worked answers

Part A — Core Concepts

1) **Normal equation derivation.**
Given the optimisation problem:

$$w^* = \arg\min_w \|Xw - y\|^2,$$

derive the closed-form solution using calculus.

Answer:

$$\nabla_w \|Xw - y\|^2 = 2X^\top Xw - 2X^\top y.$$

Setting the gradient to zero:

$$2X^\top Xw^* - 2X^\top y = 0 \Rightarrow w^* = (X^\top X)^{-1} X^\top y.$$

2) **Gradient descent update rule.**
Write the update rule for gradient descent applied to the same objective.

Answer:

$$w_{t+1} = w_t - \alpha \nabla f(w_t) = w_t - \alpha(2X^\top X w_t - 2X^\top y).$$

3) **Effect of learning rate.**
State what happens if the learning rate α is:

a) too small;

b) too large.

Answer:

a) If α is too small, convergence is slow.

b) If α is too large, optimisation may oscillate or diverge.

4) **Interpretation of the Hessian.**
What does the Hessian represent in optimisation?

Answer: The Hessian encodes curvature of the objective function. Large eigenvalues indicate steep curvature requiring small steps; small eigenvalues indicate flatter regions where larger steps are possible.

Part B — Newton's Method

5) **Deriving Newton's method.**
Using the second-order Taylor expansion of $g(w)$ around w_t, derive the Newton update rule.

Answer: Using Taylor expansion:

$$g(w) \approx g(w_t) + \nabla g(w_t)(w - w_t) + \frac{1}{2}(w - w_t)^\top \nabla^2 g(w_t)(w - w_t).$$

Differentiating and setting to zero:

$$\nabla g(w_t) + \nabla^2 g(w_t)(w^* - w_t) = 0,$$

so

$$w^* = w_t - [\nabla^2 g(w_t)]^{-1} \nabla g(w_t).$$

6) **Why one-step convergence for linear regression?**

Answer: For linear regression,

$$g(w) = \|Xw - y\|^2, \quad \nabla g(w) = 2X^\top Xw - 2X^\top y, \quad \nabla^2 g(w) = 2X^\top X.$$

Since the Hessian is constant (independent of w),

$$w_{t+1} = w_t - (2X^\top X)^{-1}(2X^\top Xw_t - 2X^\top y) = (X^\top X)^{-1}X^\top y,$$

which is already the optimum, so convergence happens in one step.

7) **Why Newton's method is rarely used in large-scale ML.**
List three reasons.

Answer:

- Requires computing and inverting the Hessian (computationally expensive).
- Requires second-order derivatives, which may not be available or smooth.
- Only guarantees local convergence and is sensitive to poor initialisation.

8) **Gradient descent vs Newton: anisotropic contours.**
Explain why gradient descent struggles under anisotropic level sets and how Newton's method resolves this.

Answer: Gradient descent follows the steepest descent direction, causing zig-zag behaviour in elongated (anisotropic) contours. Newton's method rescales the gradient using the inverse Hessian, effectively normalising curvature and directing steps toward the optimum more efficiently.

Part C — More Challenging Questions

9) **Newton's method with damping.**
In practice, a modified version of Newton's method is often used:

$$w_{t+1} = w_t - \alpha_t \big[\nabla^2 g(w_t)\big]^{-1} \nabla g(w_t),$$

where $\alpha_t \in (0, 1]$ is a damping factor. Explain why using $\alpha_t = 1$ may fail even if the Hessian is invertible.

Answer:

Using $\alpha_t = 1$ may fail because the Newton step assumes the second-order Taylor approximation is accurate locally. If w_t is far from the optimum or if the Hessian is not strictly positive definite, the update may move uphill and cause divergence.

10) **Visual interpretation of anisotropy.**
Based on the contour plots shown in the slides 21-23, explain:

   a) Why gradient descent produces a zig-zag trajectory in anisotropic landscapes.

   b) Why Newton's method produces a more direct trajectory.

   c) How the Hessian relates to the geometry of these contours.

Answer:

   a) Gradient descent always moves in the direction of steepest descent, which is nearly orthogonal to the shortest path to the minimum in elongated level sets.

   b) Newton's method rescales gradients using the inverse Hessian, effectively rotating and scaling the step to point along the principal axes of curvature.

   c) The Hessian defines the curvature and orientation of the ellipse-shaped level sets: its eigenvectors give principal directions, and its eigenvalues determine stretching.

11) **Computational trade-offs.**
Compare gradient descent and Newton's method in terms of:

   a) computational cost per iteration,

   b) number of iterations needed for convergence,

   c) storage requirements.

Answer:

   a) Gradient descent requires only first-order derivatives and is cheap per iteration; Newton's method requires computing and inverting the Hessian, which is expensive.

   b) Newton's method converges much faster (often quadratically), sometimes in a single iteration, while gradient descent typically converges linearly.

   c) Gradient descent stores only the gradient and weights, while Newton's method must store and manipulate the full Hessian matrix.