Practice Problems for Pulse-Check 2

March 23, 2026

1. A set $C \subseteq \mathbb{R}^n$ is **convex** if:

   (A) It contains the origin

   (B) It is bounded

   (C) It contains all integer points between any two members

   (D) For any $x, y \in C$ and $\theta \in [0, 1]$, the point $\theta x + (1 - \theta)y \in C$

2. Which of the following sets is **not** convex?

   (A) $\{x \in \mathbb{R}^2 : x_1^2 + x_2^2 \geq 1\}$ (exterior of a circle)

   (B) $\{x : Ax \leq b\}$ (a polyhedron)

   (C) $\{x : \|x\|_2 \leq 5\}$ (a ball)

   (D) $\{x : 0 \leq x \leq 1\}$ (a box)

3. A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if:

   (A) f is differentiable everywhere

   (B) f has a unique minimum

   (C) $f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y)$ for all x, y and $\theta \in [0, 1]$

   (D) $\nabla f(x) = 0$ at exactly one point

4. Which of the following is convex?

   (A) $f(x) = \sin(x)$

   (B) $f(x) = \max(x^2, 3x + 1)$

   (C) $f(x) = -x^2$

   (D) $f(x) = x^3$

5. For a twice-differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, f is convex if and only if:

(A) The Hessian $\nabla^2 f(x)$ is positive semidefinite for all x

(B) $\nabla f(x) = 0$ for some x

(C) The Hessian $\nabla^2 f(x) = 0$ for all x

(D) $\nabla f(x) > 0$ for all x

6. Consider $f(x) = 5x^2$. Its gradient is $\nabla f(x) = 10x$, so the Lipschitz constant of the gradient is $L = 10$. The largest safe step size for gradient descent is:

(A) $\alpha = 10$

(B) $\alpha = 5$

(C) $\alpha = 1/5$

(D) $\alpha = 1/10$

7. The fundamental property of convex optimization states that:

(A) Convex problems are always unbounded

(B) Gradient descent always converges in one step

(C) Any local minimum of a convex function over a convex set is also a global minimum

(D) Convex functions have no minima

8. For a convex and differentiable function f, the first-order optimality condition for the unconstrained problem $\min_x f(x)$ is:

(A) $\nabla f(x^*) = 0$ (necessary and sufficient)

(B) $\nabla^2 f(x^*) = 0$

(C) $f(x^*) = 0$

(D) $\nabla f(x^*) = 0$ (necessary but not sufficient)

9. For a **non-convex** function $f(x_1, x_2) = x_1^4 - x_1^2 + x_2^2$, the point $(0, 0)$ satisfies $\nabla f = 0$. This point is:

(A) A global minimum

(B) A saddle point

(C) A local minimum

(D) A global maximum

10. The gradient descent convergence rate for smooth convex functions is:

(A) $O(1/k^2)$

(B) $O(e^{-k})$

(C) $O(1/\sqrt{k})$

(D) $O(1/k)$

11. The gradient $\nabla f(x)$ points in the direction of:

    (A) Steepest descent

    (B) The nearest minimum

    (C) Steepest ascent

    (D) The nearest saddle point

12. A linear function $f(x) = c^\top x$ is:

    (A) Convex but not concave

    (B) Neither convex nor concave

    (C) Concave but not convex

    (D) Both convex and concave

13. In PyTorch, what does `loss.backward()` compute?

    (A) The gradients of the loss with respect to all leaf tensors with `requires_grad=True`

    (B) The value of the loss

    (C) The Hessian of the loss

    (D) The forward pass

14. In PyTorch, calling `.backward()` twice on different losses **without** calling `x.grad.zero_()` in between will:

    (A) Overwrite the gradient with the new value

    (B) Raise an error

    (C) Accumulate (add) the new gradient to the existing one

    (D) Set the gradient to zero

15. Why is the `torch.no_grad()` context manager needed when updating parameters in gradient descent?

    (A) To enable GPU computation

    (B) To save memory

    (C) To speed up computation

(D) To prevent PyTorch from tracking the update step as part of the computation graph

16. In backpropagation, each node in the computational graph needs two things to compute its contribution to the gradient:

(A) The forward value and the Hessian

(B) The local derivative and the upstream gradient (from the next node)

(C) The learning rate and the loss value

(D) The input data and the labels

17. In ordinary least squares, $\hat{\beta} = (A^\top A)^{-1} A^\top y$. This can blow up when:

(A) A has too many rows

(B) y is too large

(C) A is square

(D) Columns of A are nearly collinear, making $A^\top A$ ill-conditioned

18. Ridge regression solves $\min_x \|Ax - b\|^2 + \lambda \|x\|^2$. The role of λ is:

(A) To put a floor under every eigenvalue of $A^\top A$, preventing numerical blow-up

(B) To make the problem infeasible

(C) To remove constraints

(D) To maximize the residual

19. In ridge regression, choosing λ too large will:

(A) Cause numerical instability

(B) Give the same result as ordinary least squares

(C) Shrink β toward zero, underfitting the data

(D) Make the problem non-convex

20. In projected gradient descent for $\min_{x \in C} f(x)$, each iteration does:

(A) $x_{k+1} = x_k - \alpha \nabla f(x_k)$ (ignore constraints)

(B) $x_{k+1} = x_k - \alpha \Pi_C(\nabla f(x_k))$ (project the gradient, then step)

(C) $x_{k+1} = \Pi_C(x_k) - \alpha \nabla f(x_k)$

(D) $x_{k+1} = \Pi_C(x_k - \alpha \nabla f(x_k))$ (gradient step, then project onto C)

21. The projection of $z = (7, 9)$ onto the box $[0, 5]^2$ is:

    (A) $(7, 9)$
    (B) $(0, 0)$
    (C) $(5, 9)$
    (D) $(5, 5)$

22. The projection of $z = (6, 8)$ onto the unit ball $\{x : \|x\| \leq 1\}$ is:

    (A) $(6, 8)$
    (B) $(0.6, 0.8)$
    (C) $(1, 1)$
    (D) $(3, 4)$

23. Projected gradient descent works well when the projection Π_C is cheap. For which feasible set is projection **expensive**?

    (A) A box: $l \leq x \leq u$
    (B) A ball: $\|x\| \leq r$
    (C) The nonnegative orthant: $x \geq 0$
    (D) A general polyhedron: $Ax \leq b$, $Cx = d$, $x \geq 0$

24. The penalty method converts $\min_x f(x)$ s.t. $g(x) \leq 0$ into:

    (A) $\min_x f(x) + \rho \cdot [\max(0, g(x))]^2$
    (B) $\min_x f(x) + \rho \cdot g(x)$
    (C) $\min_x f(x) - \rho \cdot g(x)$
    (D) $\min_x f(x)$ (ignoring the constraint)

25. The main drawback of the penalty method with large ρ is:

    (A) The problem becomes infeasible
    (B) The penalty term disappears
    (C) The constraint is never approximately satisfied
    (D) The objective becomes ill-conditioned, causing gradient descent to slow down or produce numerical errors

26. The Lagrangian for $\min_x f(x)$ subject to $g(x) \leq 0$ is:

(A) $L(x, \lambda) = f(x) - \lambda g(x)$, $\lambda \geq 0$

(B) $L(x, \lambda) = \lambda f(x) + g(x)$

(C) $L(x, \lambda) = f(x) + \lambda g(x)$, $\lambda \geq 0$

(D) $L(x, \lambda) = f(x) + \lambda g(x)$, λ unconstrained

27. In the Lagrangian framework, if $g(x) > 0$ (constraint violated) and we maximize over $\lambda \geq 0$:

(A) The maximum is $f(x)$

(B) The maximum is $+\infty$

(C) The maximum is 0

(D) The maximum is $f(x) + g(x)$

28. In primal-dual gradient descent, the dual variable λ for an inequality constraint $g(x) \leq 0$ is updated by:

(A) $\lambda_{k+1} = \lambda_k - \alpha \cdot g(x_k)$

(B) $\lambda_{k+1} = \lambda_k + \alpha \cdot g(x_k)$ (no projection)

(C) $\lambda_{k+1} = \max(0, \lambda_k + \alpha \cdot g(x_k))$

(D) $\lambda_{k+1} = 0$

29. For an **equality** constraint $h(x) = 0$, the corresponding multiplier ν is:

(A) Constrained to $\nu \geq 0$

(B) Always zero

(C) Constrained to $\nu \leq 0$

(D) Unconstrained (can be any real number)

30. Compared to the penalty method, the Lagrangian method:

(A) Learns the right multiplier λ^* automatically, avoiding the ill-conditioning of large ρ

(B) Requires sending $\rho \to \infty$ for exact solutions

(C) Cannot handle equality constraints

(D) Always converges faster

31. The four KKT conditions for a convex problem are: stationarity, primal feasibility, dual feasibility, and complementary slackness. Complementary slackness requires:

(A) All constraints are tight at the optimum

(B) All multipliers are positive

(C) $\lambda_i^* \cdot g_i(x^*) = 0$ for each inequality constraint

(D) The gradient is zero everywhere

32. Consider $\min_x (x-5)^2$ s.t. $x \leq 3$. The optimal solution is $x^* = 3$, $\lambda^* = 4$. Verify stationarity: $\nabla_x L = 2(x-5) + \lambda = 2(3-5) + 4 =$?

(A) 0 (stationarity holds)

(B) -4

(C) 4

(D) 8

33. In the problem above, suppose the constraint were $x \leq 10$ instead. Since the unconstrained optimum $x = 5$ is feasible, the optimal multiplier would be:

(A) $\lambda^* = 4$

(B) $\lambda^* = 0$ (constraint is not active, complementary slackness)

(C) $\lambda^* = 10$

(D) $\lambda^* = -4$

34. A symmetric matrix A is positive semidefinite (PSD) if:

(A) All entries of A are nonnegative

(B) $\det(A) > 0$

(C) A is invertible

(D) $x^\top A x \geq 0$ for all $x \in \mathbb{R}^n$

35. Which of the following is **not** an equivalent characterization of $A \succeq 0$?

(A) All eigenvalues of A are nonnegative

(B) All diagonal entries of A are nonnegative

(C) $A = W^\top W$ for some matrix W

(D) $x^\top A x \geq 0$ for all x

36. In the Max-Cut problem, we assign labels $y_i \in \{-1, +1\}$ to vertices. The expression $\frac{1-y_i y_j}{2}$ equals:

(A) 1 if $y_i = y_j$, 0 otherwise

(B) $y_i + y_j$

(C) 0 if $y_i = y_j$, 1 if $y_i \neq y_j$

(D) $|y_i - y_j|$

37. The SDP relaxation for Max-Cut relaxes $y_i \in \{-1, +1\}$ (1-D vectors) to:

(A) $y_i \in [-1, 1]$ (continuous relaxation)

(B) $y_i \in \{0, 1\}$ (binary relaxation)

(C) $y_i \geq 0$ (nonneg relaxation)

(D) Unit vectors $v_i \in \mathbb{R}^n$ with $\|v_i\| = 1$ (vector relaxation)

38. The Goemans-Williamson rounding algorithm works by:

(A) Picking a random hyperplane and partitioning vertices by which side their SDP vector falls on

(B) Rounding each v_i to the nearest integer

(C) Sorting vertices by their first coordinate

(D) Greedily assigning vertices to maximize cut

39. The Goemans-Williamson algorithm achieves an approximation ratio of:

(A) 1/2

(B) 1 (exact)

(C) ≈ 0.878 (i.e., $\mathbb{E}[\text{cut}] \geq 0.878 \cdot \text{OPT}$)

(D) 2 (within a factor of 2)

40. The SDP relaxation value OPT_v relates to the true Max-Cut value OPT by:

(A) $\text{OPT}_v = \text{OPT}$ always

(B) $\text{OPT}_v \leq \text{OPT}$

(C) No relation

(D) $\text{OPT}_v \geq \text{OPT}$ (it is a relaxation of a maximization, so it can only overestimate)

41. Under the Unique Games Conjecture, the Goemans-Williamson 0.878 ratio for Max-Cut is:

(A) The best possible in polynomial time

(B) Easily improvable to 0.95

(C) Only valid for bipartite graphs

(D) Worse than a random partition

42. The set of all $n \times n$ PSD matrices $\mathcal{S}_n^+$ forms:

(A) A discrete set

(B) A polyhedron

(C) A convex cone (closed under nonnegative linear combinations)

(D) An empty set

43. An SDP can be viewed as a linear program (LP) where the variable is a **matrix** Y and the constraint $Y \succeq 0$ replaces:

(A) $Y \geq 0$ (entry-wise nonnegativity)

(B) $\det(Y) \geq 0$

(C) $\text{trace}(Y) = 1$

(D) An infinite set of linear constraints $v^\top Y v \geq 0$ for all $v \in \mathbb{R}^n$

44. The Rastrigin function $f(x) = x^2 - 10\cos(2\pi x) + 10$ is an example of a function where gradient descent fails because:

(A) The function is not differentiable

(B) The function is unbounded

(C) The function has many local minima and GD gets stuck in one

(D) The gradient is always zero

45. In simulated annealing, when a neighbor x' has cost $f(x') > f(x)$ (worse), it is accepted with probability:

(A) 0 (never)

(B) Δ/T

(C) 1 (always)

(D) $e^{-\Delta/T}$ where $\Delta = f(x') - f(x)$ and T is the temperature

46. As the temperature $T \to 0$ in simulated annealing, the algorithm behaves like:

(A) Greedy descent (rejects almost all worsening moves)

(B) A random walk (accepts everything)

(C) Brute-force enumeration

(D) Gradient descent

47. As the temperature $T \to \infty$, SA behaves like:

    (A) Greedy descent

    (B) Branch-and-bound

    (C) It terminates immediately

    (D) A random walk (accepts almost everything)

48. In SA with $T = 10$ and $\Delta = 7$, the acceptance probability is $e^{-7/10} \approx 0.497$. If the temperature drops to $T = 2$ with the same $\Delta = 7$, the acceptance probability becomes:

    (A) 0.497 (unchanged)

    (B) $e^{-7/2} \approx 0.030$

    (C) $e^{-2/7} \approx 0.751$

    (D) 1.0

49. The geometric cooling schedule $T(k) = T_0 \cdot \alpha^k$ (with $\alpha \approx 0.995$) is preferred over the logarithmic schedule $T(k) = c/\ln(1+k)$ because:

    (A) Geometric cooling has a convergence proof; logarithmic does not

    (B) Logarithmic cooling is faster

    (C) They are identical

    (D) Logarithmic cooling has a convergence proof but cools so slowly it is impractical; geometric cooling works well despite no formal guarantee

50. SA returns the **best solution found across all iterations**, not the final solution. Why?

    (A) The final solution is always optimal

    (B) The final solution is always infeasible

    (C) The current solution may have accepted uphill (worsening) moves and may not be the best visited

    (D) The algorithm does not track the current solution

51. For TSP, the 2-opt neighborhood operator works by:

    (A) Reversing a segment of the tour between two positions

    (B) Swapping two random cities in the tour

    (C) Removing two cities and reinserting them elsewhere

    (D) Adding a new city to the tour

52. In a genetic algorithm, **elitism** means:

    (A) All individuals are equally likely to be parents

    (B) Only the worst individuals are kept

    (C) Mutation rate is set to 100%

    (D) The best few individuals survive unchanged to the next generation

53. Naive crossover fails on permutation-based problems (like TSP) because:

    (A) Cutting and pasting halves of two permutations creates duplicates and missing elements

    (B) Permutations cannot be represented as arrays

    (C) Crossover always produces the same child

    (D) Permutations are continuous variables

54. Order Crossover (OX) for permutations works by:

    (A) Averaging two parent tours

    (B) Randomly shuffling both parents

    (C) Taking the better parent unchanged

    (D) Copying a segment from Parent 1, then filling remaining positions with cities from Parent 2 in order, skipping duplicates

55. When is simulated annealing preferred over branch-and-bound for TSP?

    (A) When n is small and a proof of optimality is needed

    (B) When the problem is convex

    (C) When n is large (thousands of cities) and a good solution in bounded time is acceptable

    (D) When Gurobi is available

56. In the hybrid approach "SA warm-starts Gurobi," the SA solution is used as:

    (A) The LP relaxation

    (B) A cutting plane

    (C) A replacement for the objective function

    (D) An initial incumbent for branch-and-bound, enabling more aggressive pruning

57. In a genetic algorithm, tournament selection works by:

(A) Selecting the globally best individual every time

(B) Selecting parents uniformly at random

(C) Picking k random individuals and selecting the best among them as a parent

(D) Selecting the two oldest individuals

58. Which statement about metaheuristics (SA, GA) vs. gradient descent is correct?

(A) GD works on discrete problems; metaheuristics work only on continuous problems

(B) Metaheuristics always find the global optimum; GD never does

(C) Metaheuristics require gradients; GD does not

(D) GD is preferred for high-dimensional differentiable problems (e.g., neural networks); metaheuristics are for problems without usable gradients or with combinatorial structure

Solutions:

1. (D)
2. (A)
3. (C)
4. (B)
5. (A)
6. (D)
7. (C)
8. (A)
9. (B)
10. (D)
11. (C)
12. (D)
13. (A)
14. (C)
15. (D)
16. (B)
17. (D)
18. (A)
19. (C)
20. (D)
21. (D)
22. (B)
23. (D)
24. (A)
25. (D)

26. (C)
27. (B)
28. (C)
29. (D)
30. (A)
31. (C)
32. (A)
33. (B)
34. (D)
35. (B)
36. (C)
37. (D)
38. (A)
39. (C)
40. (D)
41. (A)
42. (C)
43. (D)
44. (C)
45. (D)
46. (A)
47. (D)
48. (B)
49. (D)
50. (C)
51. (A)

52. (D)
53. (A)
54. (D)
55. (C)
56. (D)
57. (C)
58. (D)