Solutions to Review Questions

(Some of the following solutions was AI generated, so read them carefully and let me know if you have any questions.)

1. **Two steps of the Bisection Algorithm on** $f(x) = x^2 - 2$ over $[0, 2]$:

   - Step 1: $f(0) = -2$, $f(2) = 2$. Midpoint $c = 1$, $f(1) = -1$. New interval: $[1, 2]$.
   - Step 2: $f(1) = -1$, $f(2) = 2$. Midpoint $c = 1.5$, $f(1.5) = 0.25$. New interval: $[1, 1.5]$.

   Final midpoint: $x = 1.25$ (after two steps).

2. **Two steps of Newton's Method for** $f(x) = x^2 - 2$ with $x_0 = 1$:

   - Step 1: $f(1) = -1$, $f'(1) = 2$, $x_1 = 1 - \frac{-1}{2} = 1.5$
   - Step 2: $f(1.5) = 0.25$, $f'(1.5) = 3$, $x_2 = 1.5 - \frac{0.25}{3} = 1.4167$

3. **Voronoi Cell:** A region consisting of all points closer to one given center than any other. In clustering, Voronoi cells represent the influence area of each cluster center.

4. **Neural Gas Algorithm:**

   Explain the role of ϵ and λ: The parameter ϵ is the learning rate, and controls the update for the winning center. The parameter λ controls how many centers get substantial updates (large λ means many centers are updated).

5. **Minimizing** $F(c) = \frac{1}{p}\sum_{i=1}^{p}(x_i - c)^2$: Take derivative:

$$F'(c) = \frac{1}{p}\sum_{i=1}^{p} 2(c - x_i) = \frac{2}{p}\sum(c - x_i) = 0 \Rightarrow c = \frac{1}{p}\sum x_i$$

6. **k-means 1 iteration with centers as first 2 columns:**

$$X = \begin{bmatrix} -1 & 1 & 1 & -2 & -1 \\ 1 & 0 & 2 & 1 & -1 \end{bmatrix}$$

   Initial centers: $C_1 = (-1, 1)$, $C_2 = (1, 0)$

   SOLUTION: Takes a bit of computation. First sort the data by its distance to the centers (for brevity, it's OK to compute the squared distances):

Point	Dist 1	Dist 2	Winner
$(-1, 1)$	0	5	1
$(1, 0)$	5	0	2
$(1, 2)$	5	4	2
$(-2, 1)$	1	10	1
$(-1, -1)$	4	5	1

Therefore, points $1, 4, 5$ belong to cluster 1. The new cluster center is the average of these: $(1/3)(-4, 1)$

And points 2, 4 belong to cluster 2. The new cluster center is the average: $(1, 1)$.

The distortion error prior to moving the centers is the sum of the distances.

$$0 + 1 + 4 + 0 + 4 = 9$$

After update, it's take a bit to compute using a calculator, but we should get approx 5.3.

7. **Neural Gas update ($\epsilon = \lambda = 1$):**

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad C = \begin{bmatrix} -1 & 1 & 2 \\ 1 & 0 & 3 \end{bmatrix}$$

Compute distances, rank centers, update each with $\Delta C_i = \epsilon \cdot e^{-\text{rank}(i)/\lambda}(\mathbf{x} - C_i)$

Distances (in order) are 5, 4, 2, so the rank order is $2, 1, 0$ (these are the s_i). Computations (these are column vectors):

$$(-1, 1) = (-1, 1) + \mathrm{e}^{-2}(2, 1)$$
$$(1, 0) = (1, 0) + \mathrm{e}^{-1}(0, 2)$$
$$(2, 3) = (2, 3) + 1 \cdot (-1, -1)$$

8. **Partial Derivatives of Error:**

$$y = w_1 x_1 + w_2 x_2 + b \Rightarrow E = (t - y)^2$$

$$\frac{\partial E}{\partial w_1} = -2(t - y)x_1, \quad \frac{\partial E}{\partial w_2} = -2(t - y)x_2, \quad \frac{\partial E}{\partial b} = -2(t - y)$$

9. **Classify critical points:** Use gradient and Hessian of $f(x, y) = x^2 + xy + y^2 + y$.

$$\nabla f = (2x + y, x + 2y + 1), \quad H = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

The Calc 3 version: $f_{xx}f_{yy} - f_{xy}^2 = 4 - 1 = 3 > 0$. Now, $f_{xx} = 2 > 0$, so we have a local minimum.

The eigenvalue version: The characteristic equation for the Hessian matrix is

$$\lambda^2 - 4\lambda + 3 = 0 \quad \Rightarrow \quad \lambda = 1, 3.$$

Both eigenvalues are positive, so the critical point is a local minimum.

10. **Newton's method one step for where $\nabla f = 0$**

If we're looking for where $\mathbf{g}(\mathbf{x}) = 0$, then the method is:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - Jg(\mathbf{x})^{-1}\mathbf{g}(\mathbf{x})$$

If $\mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x})$, this formula looks like:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - Hf(\mathbf{x})^{-1}\nabla f(\mathbf{x})$$

With that, we first compute $\nabla f(1, 1)$ and $Hf(1, 1)$:

$$\nabla f = \left. (2x + y, x + 2y + 1) \right|_{(1,1)} = (3, 4)$$

The Hessian matrix is the same as the previous problem,

$$Hf = \left[\begin{array}{cc} 2 & 1 \\ 1 & 2 \end{array} \right] \quad \Rightarrow \quad Hf^{-1} = \frac{1}{3}\left[\begin{array}{cc} 2 & -1 \\ -1 & 2 \end{array} \right] \quad \Rightarrow \quad Hf^{-1}\nabla f = \frac{1}{3}\left[\begin{array}{cc} 2 & -1 \\ -1 & 2 \end{array} \right]\left[\begin{array}{c} 3 \\ 4 \end{array} \right] = \frac{1}{3}\left[\begin{array}{c} 2 \\ 5 \end{array} \right]$$

Now Newton's method with step size $1/10$:

$$\left[\begin{array}{c} 1 \\ 1 \end{array} \right] = \left[\begin{array}{c} 1 \\ 1 \end{array} \right] - \frac{1}{10} \cdot \frac{1}{3}\left[\begin{array}{c} 2 \\ 5 \end{array} \right] = \left[\begin{array}{c} 28/30 \\ 25/30 \end{array} \right] = \left[\begin{array}{r} 14/30 \\ 5//6 \end{array} \right]$$

11. **Line of best fit:**

(The data was given on pg 116 of the optimization notes, but left off of the question).

Here's the data:

$$(-1,0), \quad (1,1), \quad (2,3), \quad (3,2)$$

(a) The error function is:

$$E(m,b) = (0-m+b)^2 + (1-m-b)^2 + (3-2m-b)^2 + (2-3m-b)^2$$

(b) Rather than using a simplified form from the last answer, it might be easier to do it this way:

$$\frac{\partial E}{\partial m} = -2\sum x_i(y_i - (mx_i+b)), \qquad \frac{\partial E}{\partial b} = -2\sum(y_i - (mx_i+b))$$

Evaluating at $m=0$, $b=-1$, we compute errors for each point. Here's a table of values:

x	y	$mx+b$	$y_i-(mx_i+b)$
-1	0	-1	1
1	1	-1	2
2	3	-1	4
3	2	-1	3

Now compute the gradients:

$$\begin{aligned}
\frac{\partial E}{\partial m} &= -2[(-1)(1)+(1)(2)+(2)(4)+(3)(3)] \\
&= -2[-1+2+8+9] = -2(18) = -36 \\
\frac{\partial E}{\partial b} &= -2[1+2+4+3] = -2(10) = -20
\end{aligned}$$

Apply the gradient descent update:

$$m_{\text{new}} = m - \alpha\cdot\frac{\partial E}{\partial m} = 0 - 0.1(-36) = 3.6$$

$$b_{\text{new}} = b - \alpha\cdot\frac{\partial E}{\partial b} = -1 - 0.1(-20) = 1$$

(c) Using the third point $(2,3)$, from the previous table, we see that $mx_3+b=-1$ and the error $(y_3-(mx_3+b))=4$. Now the gradient is approximated using only this point:

$$\begin{aligned}
\frac{\partial E}{\partial m} &\approx -2x_3(y_3-(mx_3+b)) = -2(2)(4) = -16 \\
\frac{\partial E}{\partial b} &= -2(y_3-(mx_3+b)) = -2(4) = -8
\end{aligned}$$

Update the parameters:

$$\begin{aligned}
m_{\text{new}} &= 0 - 0.1(-16) = 1.6 \\
b_{\text{new}} &= -1 - 0.1(-8) = -1 + 0.8 = -0.2
\end{aligned}$$

This was just a "toy example" to show the computations- we wouldn't expect our gradient to be well approximated by a single point in this example.

12. **Affine to linear:** Augment $\mathbf{x}$ with 1: $\hat{\mathbf{x}} = \begin{bmatrix} x \\ 1 \end{bmatrix}$, then $\hat{A} = [A \mid \mathbf{b}]$.

13. **Widrow-Hoff Update Rule:**
$$W \leftarrow W + \alpha(T - Y)X^T$$

14. **Classification range:** Better to use what is sometimes referred to as "one hot encoding":
$$\{[1,0,0,0,0]^T, ..., [0,0,0,0,1]^T\}$$
(These are the columns of the 5×5 identity matrix).

15. **Direction of fastest decrease**

    We use two "facts":

    (a) The directional derivative in the direction of unit vector $\mathbf{u}$:
    $$D_u f(\mathbf{x}) = \nabla f(\mathbf{x}) \cdot \mathbf{u}$$

    (b) $\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \|\mathbf{y}\| \cos(\theta)$

    Put these together to get the minimum of the directional derivative:
    $$D_u f(\mathbf{x}) = \|\nabla f(\mathbf{x})\| \cos(\theta)$$
    The minimum is attained when $\cos(\theta) = -1$, or when $\theta = \pi$, which means we point in the negative direction of the gradient.

16. **Tangent line to** $f(t) = \begin{bmatrix} 3t-1 \\ t^2 \end{bmatrix}$ at $t = 1$:
$$f(1) = (2,1), f'(t) = (3,2t), f'(1) = (3,2) \Rightarrow \text{line: } \mathbf{L}(t) = (2,1) + (3,2)(t-1)$$

17. **Linearization of** $f(x,y) = x^2 + y^2 - 3xy + 2$ at $(1,0)$:
$$\nabla f = (2x - 3y, 2y - 3x) \Rightarrow \nabla f(1,0) = (2,-3)$$
$$f(1,0) = 3 \Rightarrow L(x,y) = 3 + 2(x-1) - 3(y-0)$$

18. **k-means algorithm:**

    (a) Initialize k centers.

    (b) Assign each point to nearest center.

    (c) Update centers as mean of assigned points.

    (d) Repeat until convergence.

19. **Neural Gas algorithm:**

    - Select a data point $\mathbf{x}$ at random.
    - Sort the set of centers by distance to $\mathbf{x}$, so that the first center is the closest (the winner). Let s_{i_k} be the number of centers closer to $\mathbf{x}$ than the current one. That is, the vector $\mathbf{s}$ will be the integers from 0 to $k-1$, and it tells you the order of the centers.
    - Update rule:
    $$\mathbf{c}^{(i_k)} = \mathbf{c}^{(i_k)} + \epsilon \exp\left(\frac{-s_{i_k}}{\lambda}\right)(\mathbf{x} - \mathbf{c}^{(i_k)})$$
    - In the connection (adjacency) matrix M, set the edge between the closest and next closest centers equal to 1 ($M_{i_1,i_2} = 1$). Also set the time for that connection equal to zero ($T_{i_1,i_2} = 0$).

- Age all connections by 1 ($T = T + 1$)
- Remove all old connections.
- Repeat.

20. **DBSCAN:** Three sets: Core points, Border points, Noise points.

    Be able to define each: A point is a core point if there are at least MinPts number of points (including itself) within a distance of ϵ. A point is a border point if there are less than MinPts within ϵ, but a core point is within ϵ. If not a core or border point, then it is noise.

21. **DBSCAN parameters:** ϵ (radius), MinPts (minimum number of neighbors).

22. **Widrow-Hoff update (1 point):**

    First, check sizes. We're mapping $\mathbb{R}^2 \to \mathbb{R}$, so the weight matrix W is 1×2, and b is a scalar.

    With
    $$X = \begin{bmatrix} 2 \\ -1 \end{bmatrix}, \qquad T = [1], \qquad W = \begin{bmatrix} 1 & 1 \end{bmatrix}, \qquad b = 1$$

    Then
    $$Y = WX + b = (1)(2) + (1)(-1) + 1 = 2 \quad \text{and} \quad T - Y = 1 - 2 = -1$$

    Now update:
    $$W_{\text{new}} = W + \alpha(t - y)\mathbf{x}^T = \begin{bmatrix} 1 & 1 \end{bmatrix} + 0.1(-1)\begin{bmatrix} 2 & -1 \end{bmatrix} = \begin{bmatrix} 0.8 & 1.1 \end{bmatrix},$$
    $$b_{\text{new}} = b + \alpha(t - y) = 1 - \frac{1}{10} = 0.9$$