Question 1 (Computational complexity.)
What is the computational complexity of the following pseudo code? Express your answer as a function of $N$ using $\Theta(\cdot)$ notation. Make sure to justify your answer.

```plaintext
a = 0 ## initialize the variable a with 0
for i = 0 to N-1 { ## loop over i
    for j = i+1 to N { ## loop over j
        a = a + my_function(i,j) ## increment by function (runs in constant time)
    } ## end loop over j
} ## end loop over i
```

Solution: The function gets called with pairs:
(i = 0, j = 1), (i = 0, j = 2), \ldots, (i = 0, j = N), $N$ pairs
(i = 1, j = 2), \ldots, (i = 1, j = N), $N-1$ pairs
\ldots
(i = N - 2, j = N - 1), (i = N - 2, j = N), 2 pairs
(i = N - 1, j = N). 1 pair
The number of pairs is $N + (N - 1) + \ldots + 2 + 1 = \frac{N(N+1)}{2}$. Because each function call takes constant time, the computational complexity has the same rate of growth as $\frac{N(N+1)}{2}$. This is $\Theta(N^2)$, because the rate of growth of a polynomial is a monomial whose exponent is the polynomial’s greatest exponent.
**Question 2 (Curve fitting.)**
The following sequence of measurements is generated by a model of the form

\[ y = f(x) + N(0, 0.01), \]

where \( f(x) \) is a polynomial of unknown order, and the noise is modelled by a Gaussian distribution with zero mean and variance 0.01. The inputs \( x \) and outputs \( y \) are each of length 8, and are given by the following:

<table>
<thead>
<tr>
<th>( x )</th>
<th>0.94</th>
<th>1.35</th>
<th>1.54</th>
<th>2.48</th>
<th>2.60</th>
<th>3.84</th>
<th>3.95</th>
<th>4.35</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y )</td>
<td>5.75</td>
<td>9.52</td>
<td>12.01</td>
<td>30.60</td>
<td>33.93</td>
<td>85.77</td>
<td>91.82</td>
<td>118.11</td>
</tr>
</tbody>
</table>

We have used a curve fitting numerical package to compute the coefficients using the following polynomial approximation,

\[ y(x) = \sum_{i=0}^{\text{model order}} a_i x^i, \]

that best fit the data. The coefficients estimated for polynomials of model order 2 through 5 are listed below.

<table>
<thead>
<tr>
<th>Model order</th>
<th>( a_5 )</th>
<th>( a_4 )</th>
<th>( a_3 )</th>
<th>( a_2 )</th>
<th>( a_1 )</th>
<th>( a_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9.016</td>
<td>-15.533</td>
<td>13.511</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1.052</td>
<td>0.782</td>
<td>3.633</td>
<td>0.720</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0.043</td>
<td>0.592</td>
<td>2.490</td>
<td>1.053</td>
<td>2.030</td>
</tr>
<tr>
<td>5</td>
<td>-0.035</td>
<td>0.500</td>
<td>-1.663</td>
<td>7.640</td>
<td>-4.328</td>
<td>4.082</td>
</tr>
</tbody>
</table>

These polynomial models were used to compute the estimated \( y \) for our 8 inputs. In the table below, each row corresponds to an input \( x \) (its value appears in the first column), and the latter columns correspond to the fitting errors for our polynomials.

<table>
<thead>
<tr>
<th>( x )</th>
<th>order-2</th>
<th>order-3</th>
<th>order-4</th>
<th>order-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.94</td>
<td>1.127</td>
<td>-0.053</td>
<td>-0.007</td>
<td>-0.004</td>
</tr>
<tr>
<td>1.35</td>
<td>-0.563</td>
<td>0.093</td>
<td>0.046</td>
<td>0.033</td>
</tr>
<tr>
<td>1.54</td>
<td>-1.036</td>
<td>-0.001</td>
<td>-0.048</td>
<td>-0.037</td>
</tr>
<tr>
<td>2.48</td>
<td>-0.122</td>
<td>0.009</td>
<td>0.050</td>
<td>0.058</td>
</tr>
<tr>
<td>2.60</td>
<td>0.049</td>
<td>-0.088</td>
<td>-0.042</td>
<td>-0.051</td>
</tr>
<tr>
<td>3.84</td>
<td>1.125</td>
<td>0.053</td>
<td>0.011</td>
<td>0.004</td>
</tr>
<tr>
<td>3.95</td>
<td>0.801</td>
<td>0.029</td>
<td>-0.011</td>
<td>-0.002</td>
</tr>
<tr>
<td>4.35</td>
<td>-1.381</td>
<td>-0.043</td>
<td>0.001</td>
<td>-0.001</td>
</tr>
</tbody>
</table>

Which polynomial is the most suitable model? Make sure to justify your answer.

**Solution:** The main issue in this question is to realize that *some model orders may involve over-fitting, while others involve under-fitting*. How can we identify which ones are which? A reasonable starting point is to realize that the higher order models are suspect of possible over-fitting, while the lower order ones under-fitting.

The low-order models with under-fitting are relatively easy to see here. For the order-2 model, typical fitting errors have magnitude (absolute error) on the order of 1. In contrast, higher order models have errors on the order of 0.1 and often less. Not only that, but the variance of the error in the process that generates \( y \) from \( x \) is 0.01, meaning that the standard deviation is 0.1. Therefore, the order-2 model is under-fitting; the larger fitting errors are likely due to a lower-order model, and these are not actual noise. Many students understood this part of the question intuitively.

What about the higher-order models? Because (again) the actual error has standard deviation 0.1, anything below that level is over kill. Some students didn’t see this part of the question. The order-3 model provides errors on the order of 0.1 (this is reasonable), but
higher order models feature somewhat lower errors. Therefore, the order-3 model is reasonable, while the order-4 and 5 models suffer from over-fitting. We conclude that the order-3 model seems to be the correct one.

One can also answer this question “from first principles” using the minimum description length (MDL) principle, but this is not truly needed to solve the question.
**Question 3** (Bayesian classification.)

Based on two classes, red and blue, a variable \( x \) is generated in different ways. The probabilities of the two classes satisfy \( \Pr(\text{blue}) = 0.7 \) and \( \Pr(\text{red}) = 0.3 \). The distributions of \( x \) for each class are Gaussian densities, \( f_{\text{blue}} = \mathcal{N}(0, 1) \) and \( f_{\text{red}} = \mathcal{N}(0, 4) \), where the Gaussian means of both components are zero, and the variances of the components corresponding to blue and red classes are 1 and 4, respectively. Recall that a Gaussian random variable \( x \) with mean \( \mu \) and variance \( \sigma^2 \) has probability density function (pdf) given by

\[
f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.
\]

(a) Derive a Bayesian classifier by computing the posterior probability, \( \Pr(\text{red}|x) \).

**Solution:**

\[
\Pr(\text{red}|x) = \frac{f(\text{red}, x)}{f(x)} \frac{f(\text{red}, x)}{f(\text{red}, x) + f(\text{blue}, x)} = \frac{\Pr(\text{red})f(x|\text{red})}{\Pr(\text{red})f(x|\text{red}) + \Pr(\text{blue})f(x|\text{blue})}
\]

\[
= \frac{0.3 \cdot \frac{1}{\sqrt{2\pi 4}} e^{-\frac{x^2}{8}}}{0.3 \cdot \frac{1}{\sqrt{2\pi 4}} e^{-\frac{x^2}{8}} + 0.7 \cdot \frac{1}{\sqrt{2\pi 1}} e^{-\frac{x^2}{2}}}
\]

\[
= \frac{0.3 e^{-\frac{x^2}{8}}}{0.3 e^{-\frac{x^2}{8}} + 0.7 e^{-\frac{x^2}{2}}}
\]

(b) If one must decide between red and blue given \( x \), where is/are the decision boundary/boundaries? (In other words, for what value of \( x \) do we have \( \Pr(\text{red}|x) = 0.5 \)? You need not provide a numerical value; an expression will be fine.)

**Solution:** In order for the probability to be half, the two terms in the denominator of the expression for the posterior, \( \Pr(\text{red}|x) \), should be identical,

\[
\frac{0.3}{2} e^{-\frac{x^2}{8}} = 0.7e^{-\frac{x^2}{2}}.
\]

Simplifying this expression yields \( e^{-\frac{x^2}{8}} e^{\frac{x^2}{2}} = e^{\frac{3x^2}{8}} = \frac{2.07}{0.3} = \frac{14}{3} \), meaning that \( \frac{3x^2}{8} = \ln(14/3) \), which can be simplified further.
**Question 4** (Dynamic programming.)

In the figure below, the value on each edge represents the length of the route between two nodes; note that you are only allowed to move toward the right. Your goal is to determine the shortest route from $x_0$ to $x_6$. To do so, you will use dynamic programming (DP).

(a) The graph is arranged in layers. A DP-based approach for solving this problem is to process layer by layer, starting from the right-most layer (containing $x_6$), continuing to the second right-most layer (connected to $x_6$ with edges of length 6 and 7), and so on. Explain in words how you will process each node in each layer.

(b) Use your algorithm to determine the shortest route from $x_0$ to $x_6$. Show your answer by sketching the path on the figure. (You may find it helpful to jot down next to each node the length of the shortest route linking it to $x_6$.)

**Solution:** The DP-based approach is to compute in each layer the lengths of the shortest routes from all the nodes in that layer to $x_6$. For node $x^*$, denote the length of its shortest route by $\Psi(x^*)$. There are either one or two edges from $x^*$ to the right. If there is one such edge, then denoting the node we reach by moving right as $r(x^*)$, $\Psi(x^*)$ is the sum of $\Psi(r(x^*))$ and the length of the edge linking $x^*$ to $r(x^*)$, which can be expressed as

$$\Psi(x^*) = \Psi(r(x^*)) + \text{length}(x^*, r(x^*))$$

If there are two edges, denote the nodes reached by moving toward the upper right and lower right by $ur(x^*)$ and $lr(x^*)$, respectively. The lengths of the two possible routes are
\( \Psi(\text{ur}(x^*)) + \text{length}(x^*, \text{ur}(x^*)) \) and \( \Psi(\text{lr}(x^*)) + \text{length}(x^*, \text{lr}(x^*)) \). We choose \( \Psi(x^*) \) as the minimum of these two,

\[
\Psi(x^*) = \min\{ \Psi(\text{ur}(x^*)) + \text{length}(x^*, \text{ur}(x^*)), \Psi(\text{lr}(x^*)) + \text{length}(x^*, \text{lr}(x^*)) \}.
\]

Finally, let us not ignore the basis case; in the right-most layer, \( \Psi(x_6) = 0 \).

Running the algorithm on this graph:

- Right-most layer: \( \Psi(x_6) = 0 \) is the basis case.
- Second right-most: there is only one edge linking each of the nodes to \( x_6 \), and their \( \Psi \) costs are 7 (top) and 6 (bottom).
- We now process three nodes. The top one has cost \( 9+7=16 \), because it only has one edge moving to the right. Similarly, the bottom one has \( 5+6=11 \). The middle one has cost \( \min\{8+7, 7+6\} = 13 \).
- We continue processing the layers, moving to the left, until we reach \( x_0 \).
Question 5 (Homework 5.)  
We discussed in class how to use denoisers iteratively within the approximate message passing (AMP) algorithmic framework. In class, we derived a denoiser for a signal \( x \in \mathbb{R}^N \) that takes values \(+1\) and \(-1\).  
This question considers a Rademacher signal. Each \( x_n \), entry \( n \in \{1, \ldots, N\} \) of the vector \( x \), takes the values \(-1\) or \(+1\), each with probability \( \epsilon \), else \( x_n \) is zero with probability \( 1 - 2\epsilon \). (The \( N \) symbols are independent and identically distributed, where \( \Pr(X_n = -1) = \Pr(X_n = +1) = \epsilon \) and \( \Pr(X_n = 0) = 1 - 2\epsilon \). You may assume that \( 0 < \epsilon < 0.5 \).)  
Our denoiser estimates \( x \) from the pseudo data, \( v = x + \mathcal{N}(0, \sigma^2) \), where the conditional probability density function (pdf) for \( V_n \) conditioned on \( X_n \) obeys 

\[
    f(V_n|X_n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n - x_n)^2}{2\sigma^2}}.
\]

We compute the conditional expectation, 

\[
    E[X_n|V_n] = (-1) \Pr(X_n = -1|V_n) + (0) \Pr(X_n = 0|V_n) + (1) \Pr(X_n = 1|V_n)
\]

The conditional probabilities, \( \Pr(X_n = 1|V_n) \) and \( \Pr(X_n = -1|V_n) \), follow Bayes’ rule, 

\[
    \Pr(X_n = 1|V_n) = \frac{f(X_n = 1, V_n)}{f(V_n)} = \frac{f(X_n = 1, V_n)}{f(X_n = -1, V_n) + f(X_n = 0, V_n) + f(X_n = 1, V_n)},
\]

and individual terms are expanded as \( f(X_n = 1, V_n) = \Pr(X_n = 1)f(V_n|X_n = 1) \), and so on.  

Below you will compute \( \Pr(X_n = -1|V_n) \) as a function of the parameters \( \epsilon \) and \( \sigma \).  

(a) Provide expressions for \( f(V_n|X_n = -1) \), \( f(V_n|X_n = 0) \), and \( f(V_n|X_n = +1) \).  

**Solution:** The three expressions are 

\[
    f(V_n|X_n = -1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n + 1)^2}{2\sigma^2}},
\]

\[
    f(V_n|X_n = 0) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n)^2}{2\sigma^2}},
\]

\[
    f(V_n|X_n = +1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n - 1)^2}{2\sigma^2}}.
\]

(b) Provide expressions for \( f(X_n = -1, V_n) \), \( f(X_n = 0, V_n) \), and \( f(X_n = +1, V_n) \).  

**Solution:** The three expressions are 

\[
    f(X_n = -1, V_n) = \Pr(X_n = -1)f(V + n|X_n = -1) = \frac{\epsilon}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n + 1)^2}{2\sigma^2}},
\]

\[
    f(X_n = 0, V_n) = \Pr(X_n = 0)f(V + n|X_n = 0) = \frac{1 - 2\epsilon}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n)^2}{2\sigma^2}},
\]

\[
    f(X_n = +1, V_n) = \Pr(X_n = +1)f(V + n|X_n = +1) = \frac{\epsilon}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v_n - 1)^2}{2\sigma^2}}.
\]

(c) Using equation (1) as a guideline, provide an expression for \( \Pr(X_n = -1|V_n) \).  

**Solution:** 

\[
    \Pr(X_n = -1|V_n) = \frac{f(X_n = -1, V_n)}{f(V_n)} = \frac{f(X_n = -1, V_n)}{f(X_n = -1, V_n) + f(X_n = 0, V_n) + f(X_n = 1, V_n)}.
\]