Calculators and other accessories are not permitted.
Include **all intermediate calculations** necessary to justify your answer.

Problem 1 . . . . *(6%)*

Evaluate the following Haskell expressions:

(a) `[1,4..15]`

(b) `map (*2) [-2,0,2]`

(c) `zip [-1,0,1] [3,2,0]`

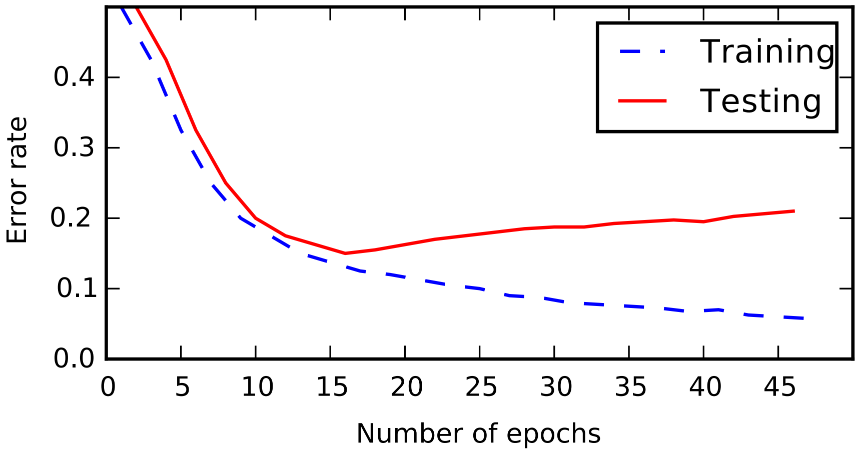

Figure 1: Plot of error rates for a neural network.

Problem 2 . . . . *(16%)*

Figure 1 shows the error rates for training and testing of some neural network. Explain (in some depth) how we read such a plot with reference to key concepts of neural networks and machine learning where appropriate. Your answer should cover, but not be limited by, the following questions:

1. What do we mean by an epoch?
2. What is the difference between training and testing error?
3. How many epochs should be used?
4. Why is one curve monotonically decreasing while the other decreases first and then starts to increase again?
5. Give possible causes for the increase in testing error when the number of epochs increases.

Problem 3 . . . . *(12%)*

This question concerns key features and challenges of functional programming.

(a) Explain what we mean by recursion.

(b) Why is recursion more important in functional programming than it is in common imperative languages?

(c) Show how to use recursion to calculate the sum of elements in a list $a = [a_1, a_2, \ldots, a_n]$.

(d) Consider a function `getChar` reading a character from the keyboard and returning it. Why is not `getChar` a *function* in the strictest sense? In other words, why does such a function pose particular challenges in many functional languages?

Problem 4 .... *(15%)*

This problem concerns the single neuron and its key properties.

```haskell
type Neuron = [Double]
type Vector = [Double]
recall :: Vector -> Neuron -> Double
train :: Double -> Vector -> Double -> Neuron -> Neuron
```

(a) Briefly explain what a neuron is.

(b) Give a definition of the `recall` function. (You may use Haskell syntax or mathematics as you prefer.)

(c) Give a definition of the `train` function. (You may use Haskell syntax or mathematics as you prefer.)

(d) The neuron is a *linear classifier*. Explain what this means and how it is a limitation in practice. Feel free to draw figures to support your explanation.

(e) Explain how we can use the neuron to construct non-linear classifiers.

Problem 5 .... *(12%)*

A simple algorithm for optimising 1D functions is implemented in the code below. Since the search space is 1D and equivalent to a line, the algorithm can search along the line until it finds an optimum. If the goal is to find a maximum (minimum), the algorithm moves up (down) the line until it cannot get any further, and the stopping point is the solution.

```haskell
f0 :: Double -> Double
f0 x = 4*x - 2*x^2

fopt :: (Double -> Double) -> Double -> Double -> (Double, Double, Int)
fopt f x0 delta | x0 - delta < l || x0 + delta > u = (x0, f x0, 0)
                | f (x0 - delta) > f x0            = (a, b, c + 1)
                | f (x0 + delta) > f x0            = (a', b', c' + 1)
                | otherwise                        = (x0, f x0, 0)
                  where l = -20
                        u = 20
                        (a,b,c)    = fopt f (x0 - delta) delta
                        (a',b',c') = fopt f (x0 + delta) delta
```

Consider the function `fopt` in the code above and the following example evaluation:

```
*Exam> fopt f0 5 0.01
(<A>,<B>,400)
```

(a) Calculate the theoretical return values <A> and <B> by hand using calculus.

(b) Is the algorithm a maximum-seeking algorithm, a minimum-seeking algorithm, or both?

(c) Explain the meaning of the last value, '400', in the returned tuple above. Also, explain the meaning of the numbers '5', '0.01' in the example function evaluation above, and how they affect the evaluation.

(d) Suppose `f0` was given as a polynomial of degree three or higher. Explain why this poses a problem to the algorithm.

Problem 6 .... *(5%)*

Model predictive control (MPC) can be used to solve dynamic optimisation problems, that is problems which have to be solved repeatedly at every time step.

(a) Why is MPC also called receding horizon control?

(b) How can a GA be used in MPC?

Problem 7 . . . . *(8%)*

In terms of search and optimisation algorithms such as the GA, what is meant by the following terms:

(a) elitism

(b) convergence rate

(c) exploration

(d) exploitation

Problem 8 . . . . *(10%)*

Suppose you want to implement a binary GA for learning character strings. Your set of characters (“alphabet”) contains the 26 English characters ’a’, ’b’, . . . , ’z’.

(a) How many bits are needed to encode all 26 characters?

(b) Give an example of an encoding scheme.

(c) Give an example of a suitable cost function that compares candidate strings with a target string. You do not have to use mathematical equation(s) unless you prefer to; a text explanation is sufficient.

(d) Illustrate your example cost function by comparing the following strings `s1` and `s2` and calculating the cost:

```
s1 = "getrat"
s2 = "target"
```

Problem 9 . . . . *(16%)*

Suppose you are going to design and train a neural network for letter recognition to enable digitalisation of text (OCR). You are given a data set including samples of 20 different fonts[1]. Each object includes the letter (26 possible values, A–Z) and 16 geometrical features.

Discuss how you would undertake the development of a suitable neural network, answering the following questions:

1. What is the input to the neural network?
2. What is the output from the neural network? Discuss different ways to encode the output and make a recommendation.
3. The data set includes 20 000 instances. Explain how you would split this into different subsets and how each subset would be used in the development process.
4. Give an overview of other key consideration to be made during the design of a neural network, and suggest a technique to resolve each one.

[1] *Letter Recognition Data Set* by David J. Slate, published by UCI Machine Learning Repository.