University of Thessaly

Neuro-Fuzzy Computing

ECE447

3rd Problem Set

Alexandra Gianni   Nikos Stylianou
ID: 3382   ID: 2917

March 1, 2024

Problem 1

In this problem, we want to design a RBF classifier by hand. The input data are shown below. The network should produce the following output:

$$out_{RBF} = \begin{cases} > 0, & \text{when input vector is in the designated area} \\ < 0, & \text{otherwise} \end{cases}$$

The designated area is consisted of two circles with centers at $(-1, 1.5)$, $(2, 2)$ and radius of $\rho_1 = 0.5$, $\rho_2 = 0.25$ respectively, like figure 1.
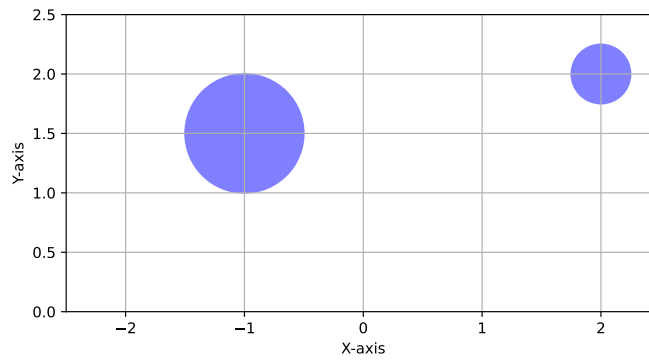

Figure 1

Following the standard procedure for an RBF network, the network will have two layers. On the RBF layer, there will be two neurons, each corresponding to one of the designated regions. As far as the output layer is concerned, there will be only a single neuron responsible for the classification output.

Moving on to some more critical parameters of the network. The centers of the RBF will be located at the same centers as the given circles, meaning at $(-1, 1.5)$ and $(-2, 2)$. The spread (σ) needs to be chosen such that the area under the Gaussian function covers the shaded regions appropriately.

In the absence of the exact radii of the circles, a common approach is to use the distance between the centers of the two circles to estimate the spread. The spread should typically be around the radius of the circles or slightly larger to ensure that the function covers the entire area of interest.

The distance between the two centers can be calculated using the following equation:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} = \sqrt{(-3)^2 + (-0.5)^2} = 3.0414$$

Since we don't want the spread to be too large to avoid overlap, we can take a fraction of this distance for each spread. If the circles are not overlapping in the classification task, we can choose the spread to be half of the distance, ensuring that each neuron's function covers its respective circle adequately. Thus, the calculated spread is 1.52.

Moving on to the weights of the network, we will set the weight from each RBF neuron to the output neuron to 1, assuming no overlap between the shaded regions and no need to differentiate the influence of each RBF neuron. The weights are set to 1 to ensure that each RBF neuron's activation contributes positively and equally to the output.

Lastly, the bias of the output neuron should be set to -1, so that when neither of the RBF neurons is activated, the output neuron's activation is negative. When one or both RBF neurons are activated, the sum will be greater than 0, and the output neuron will produce a positive result.

Problem 2

We are asked to write a Python program that implements steepest descent algorithm for the 1-S^1-1 RBF network. The input function that we want to approximate is

$$g(p) = 1 + \sin\left(p\pi/8\right), \quad \text{for } p \in [-4, 4]$$

We select 30 data randomly from that interval and all parameters are initialized as small numbers using `numpy.random.randn` function. It returns a number at the exact specification as needed.

Figure 2: Input function in the specified area and the randomly assigned train data points.

For the randomly assigned data points, we added a custom seed number using in order for the results to be comparable but still use randomness. The number of centers define the total number of hidden layers in the network. So, in order to train it, thus impacting its accuracy.

Every arithmetic operation is done in matrix mode using `np.array` object type for better scalability and automation. For learning rate values α we used $[0.001, 0.005, 0.01]$ and for center the values $[4, 8, 12, 20]$ as suggested. As for the iterations number, we opted for $20 \cdot 10^3$ as we didn't want any possible bottleneck in the epoch number.

In figures 3, 4, 5 and 6 we can see the output of the neural network for each α and number of center (*hidden layers*).

Firstly, let's consider the impact of the number of centers. As we increase hat number from 4 to 8, 12, and then 20, the output dynamics undergo a notable transformation. With a smaller number of centers, the network struggles to capture intricate patterns in the data, leading to underfitting. This can be seen particularly well in figure 3, where number of centers is 4. The network fails to predict the input signal across all data points. Conversely, increasing the number of centers enables the network to model more complex relationships, reducing the underfitting and improving generalization. Again, these observations are very clear from the first change in the number of centers, from 4 to 8. The latter network has a smaller error across all data points when compared to the input signal. This increase of quality is not observed only in this increase of center numbers, but to all increases.

Next, let's delve into the impact of the learning rate on the RBF layer's output. A higher learning rate accelerates the convergence of the training process, enabling the model to reach a satisfactory solution faster.
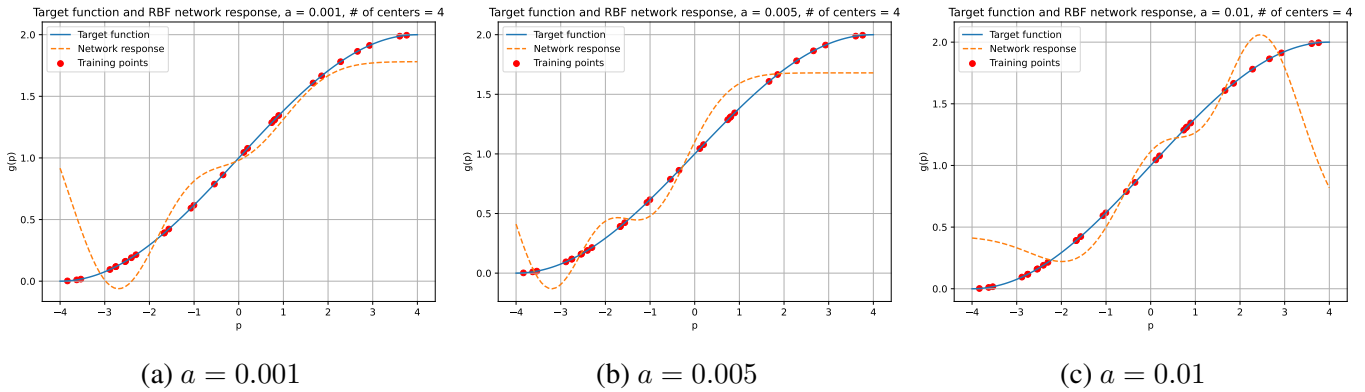
(a) $a = 0.001$ (b) $a = 0.005$ (c) $a = 0.01$

Figure 3: Input function approximation with number of centers $= 4$.

(a) $a = 0.001$ (b) $a = 0.005$ (c) $a = 0.01$
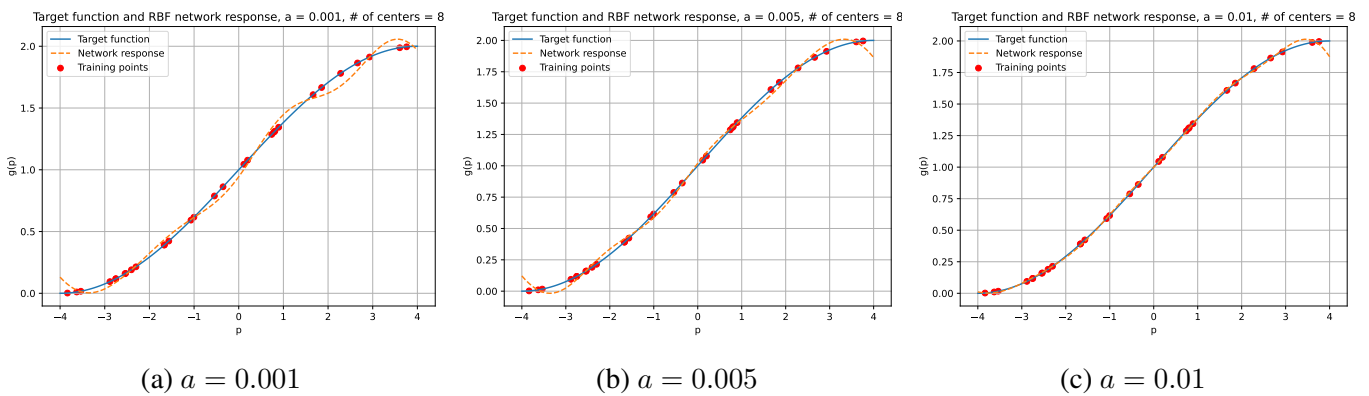
Figure 4: Input function approximation with number of centers $= 8$.

(a) $a = 0.001$ (b) $a = 0.005$ (c) $a = 0.01$
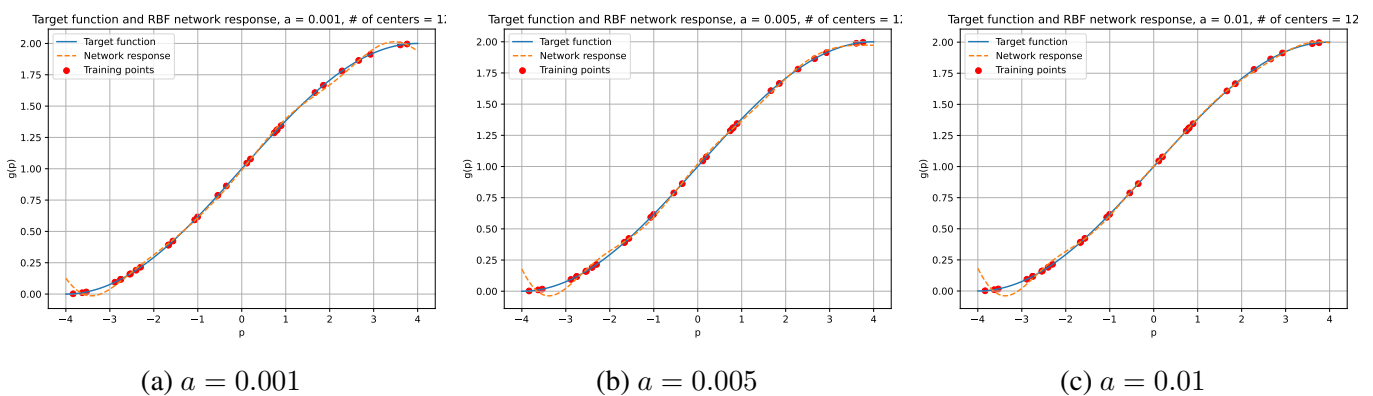
Figure 5: Input function approximation with number of centers $= 12$.

(a) $a = 0.001$ (b) $a = 0.005$ (c) $a = 0.01$
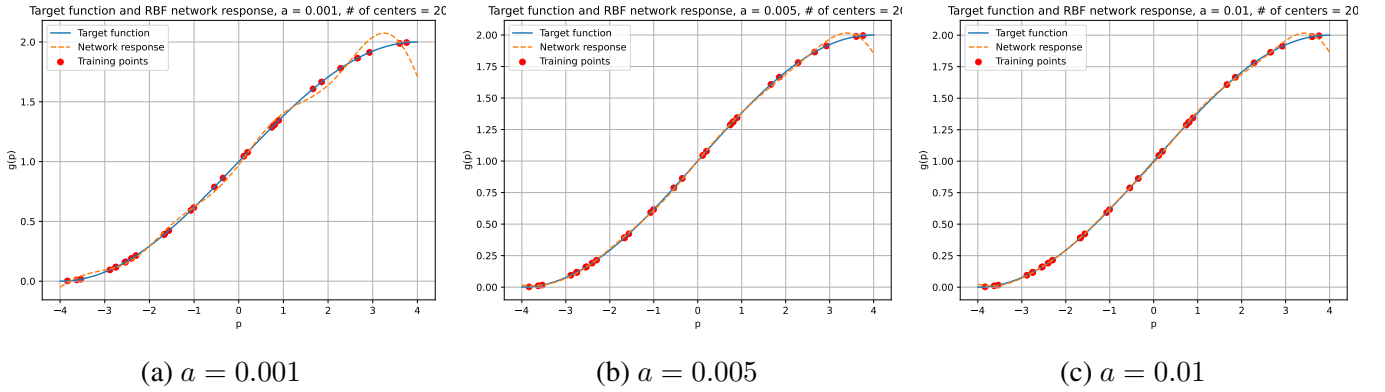
Figure 6: Input function approximation with number of centers $= 20$.

However, a learning rate that is too high might leads to oscillations or divergence, hindering the convergence process. From our calculations, when α is increased above 0.05, the system is diverging and by a large factor. On the other hand, a lower learning rate ensures more cautious updates to the model parameters, reducing the risk of divergence but potentially prolonging the convergence process. Thus, the output dynamics with different learning rates exhibit variations in convergence speed and possibly final performance, with an optimal learning rate striking a balance between convergence efficiency and stability.

Problem 3

LVQ (Learning Vector Quantization) is a type of artificial neural network algorithm used for supervised learning. It belongs to the category of competitive learning algorithms and is particularly effective for classification tasks.
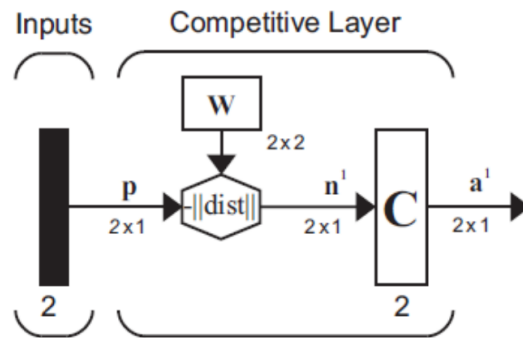

Figure 7: Given neural network.

The following $-||w_i - p||$ applies to every n_i^1. Also, we can also state that $a^1 = compet\,(n^1)$, where $compet$ is competitive learning layer.

During training, the distance between a and w is calculated using $dist = norm\,(p - w_{1,2})$ and judging by whose norm is grater, the winning neutron gets updated. Also, the presentation order of the vectors during training is: $p_1, p_2, p_3, p_2, p_3, p_1$.

All initial values are presented below:

$$p_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad p_2 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}, \quad p_3 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}, \quad w_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad w_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

In order to reduce unnecessary computations, we implemented a convergence control that stops training when weights have very small differences with each other.

After letting the network converge, we have as final weights the following:

Training converged at epoch 10.

$$w_1 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}, \quad w_2 = \begin{bmatrix} 0.2 \\ 1.4 \end{bmatrix}$$

Finally, in figure 8 we can see the weights over epochs during training. The code for this is in file `prob3.py`.
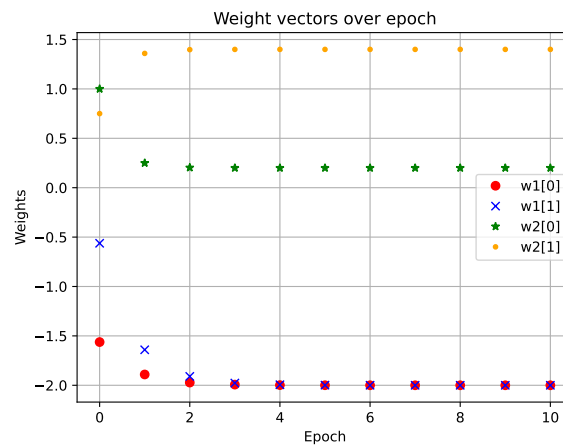

Figure 8: Weights over epoch during training.

Problem 4

Problem 4 depends on Problem 3's neural network, but it is executed using different initial values.
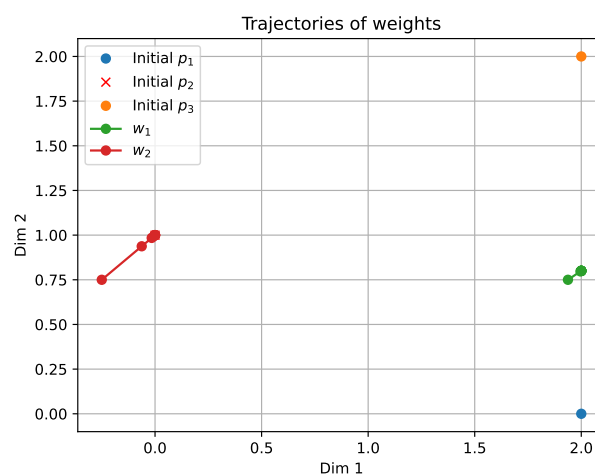

Figure 9: Trajectory of weights during training.

In figure 9, the weight trajectory can be seen. Based on the final weight positions after training, we can infer the clustering of the three vectors as follows:

Weight W_1 has moved towards and is located at the coordinates $(2.0, 0.8)$, which is close to both p_1 $(2, 0)$ and p_3 $(2, 2)$. Given that these points are close in the input space and W_1 is approximately at the mean of the vertical component of p_1 and p_3, it suggests that W_1 would be the winning neuron for inputs similar to p_1 and p_3 in a larger number of iterations. This means that both p_1 and p_3 would be clustered together.

Weight W_2 has moved to a position very close to $p2$ $(0, 1)$, which is at the coordinates $(-9.536\,743\,16e - 7$, $9.999\,990\,46e - 1) \approx (0, 1)$. This indicates that W_2 has adjusted to represent input p_2.

Therefore, if the network continues to be trained for a larger number of iterations, the three vectors are likely to be clustered into two groups:

- Cluster 1: Inputs p_1 and p_3, represented by W_1.

- Cluster 2: Input p_2, represented by W_2.

Problem 5

We are asked to generate an Auto Regressive (*AR*) model and then create and RNN to predict it. Specifically, we chose to implement an LSTM as it has more parameters than GRU and is more adaptable to new information.

Samples must be generated from an AR model of the following form:

$$X_t = 0.5X_{t-1} - 0.1Xt - 2 + 0.2X_{t-3} + U_t$$

where U_t is independent-identically distributed Uniform in the interval $(-0.25, 0.25)$.

At first, we create the samples using `generate_AR_data()` and the dataset for the neural network to train using `create_dataset()`.

After creating all of the necessary data, we create a new model according to some predefined training sample sizes which are: $[100, 200, 500, 1000, 2000]$. Calculation of valid loss and cost square error function is what follows the previous steps.

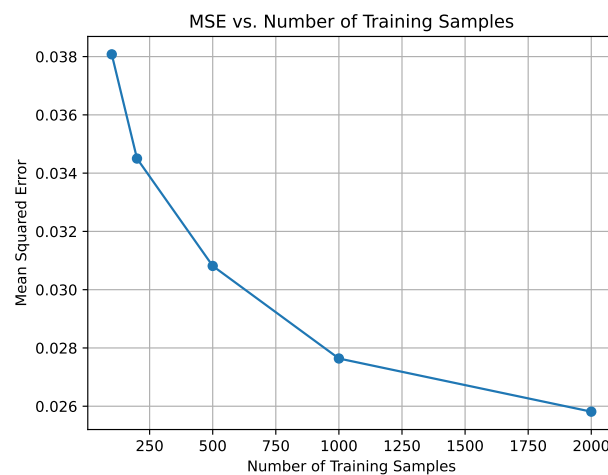At the end, all cost functions are shown in figure 10.

Figure 10: Averaged cost square error function over number of training samples

The observed decrease in MSE with an increase in the number of training samples is indicative of the LSTM's capacity to learn and model the dependencies in the time series data. LSTMs are designed to capture long-term relationships, which is particularly advantageous for AR models where the future value is a function of past values.

The continuous improvement in prediction accuracy with more data suggests that the LSTM has not reached its capacity. Typically, there is a concern that a model may overfit to the training data if given too many examples. However, in this case, overfitting does not appear to be an issue. This could be due to the stochastic nature of the AR process and the noise term, which require the model to generalize well to perform accurately. It's also worth noting the learning dynamics of LSTM networks. During the initial phase of training, the network rapidly learns the gross features of the time series. As training progresses, the rate of improvement in MSE slows down, suggesting that the model begins to refine its understanding of finer details within the data. This refinement process requires more nuanced examples, which can only be provided by larger datasets.

Problem 6

This problem depends on Problem 5, as it uses its technique to train a neural network. The only thing changing here is the `generate_AR_data()` function, that now creates a time series that is described using the following equation:

$$X_t = U_t + a_1 U_{t-1} + a_2 U_{t-2} + a_3 U_{t-3} + a_4 U_{t-4} + a_5 U_{t-5} + a_6 U_{t-6}$$

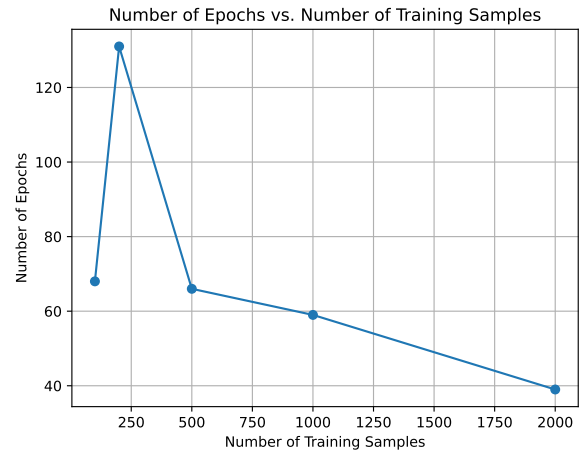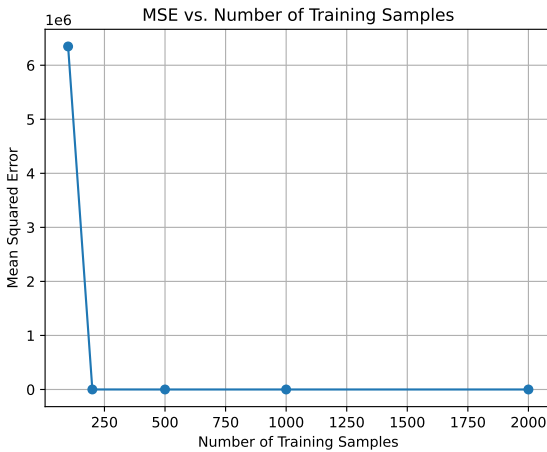with $a_1 = a_2 = 5$ and $a_3 = a_4 = a_5 = a_6 = -0.5$.

Figure 11: Averaged cost square error function over number of training samples

Figure 12: Number of epochs required for the NN to converge

Figures 11 & 12 are generated with data taken from neural network's train procedure.

Both of these graphs together indicate a relationship between the amount of data the model is trained on and the model's performance and training characteristics. More data generally seems to lead to better performance (*as seen by lower MSE*) and potentially more efficient training (*fewer epochs needed as training samples increase*). The outlier or spike in the number of epochs at 200 samples happened because NN's input was fairly small in size. If we look at even smaller values, we can see that the number of epochs is smaller than the one on 200 samples but if we see the MSE number, we observe that network does anything that converging.

The U_t being uniformly distributed means that it has a constant probability over the interval $(0, 0.5)$, and being independent and identically distributed means that each U_t is drawn from the same distribution and is independent of others. The coefficients used for data generation suggest that U_{t-1} and U_{t-2} have a significant positive influence on the current value, while U_{t-3} to U_{t-6} have a small negative influence.

Overall, these graphs suggest that the LSTM model benefits from more data which is consistent with the behavior of most machine learning models, as LSTM are used for exactly this reason.

Problem 7

Modeling expressions as fuzzy subsets in the context of fuzzy logic allows us to handle the uncertainty and imprecision inherent in certain descriptions like "large", "very small", or "medium-weight". In fuzzy logic, each element of the subset has a degree of membership ranging from 0 to 1, where 0 means "not a member at all" and 1 means "fully a member".

Below are examples of how you might model the given expressions as fuzzy subsets using MATLAB code snippets, assuming some reasonable definitions for each term.

LARGE INTEGERS

For "Large integers", we need a membership function that gradually increases as the integers become larger. A simple way to model this is to use a function that increases the membership score as the number exceeds a certain threshold. However, defining "large" is subjective and can vary depending on the context. For simplicity, let's assume integers greater than 100 are considered large, but the transition starts at 50, becoming more "large" as the number increases.
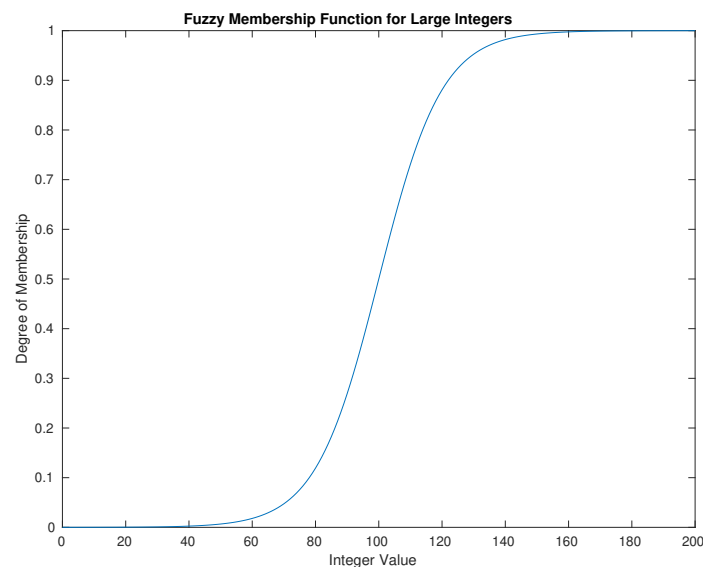

Figure 13: Fuzzy membership function for Large Integers

we are using the sigmoid function for a smooth transition.

VERY SMALL NUMBERS

For "very small numbers", we can consider numbers close to zero as having a higher degree of membership. "Very small numbers" can include both positive numbers close to zero and negative numbers. For this, a membership function that assigns higher scores to numbers closer to zero can be used. We'll focus on positive numbers for simplicity, but this can be adjusted to include negatives.
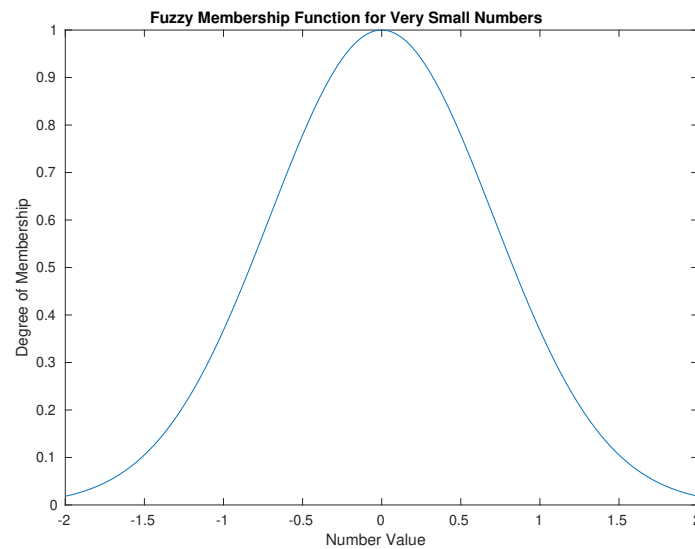
Figure 14: Fuzzy membership function for Very Small Numbers

we are using the Gaussian function centered at 0.

MEDIUM-WEIGHT MEN
Assuming the average weight range for medium-weight men is between 70kg and 90kg, with the peak at 75kg, we used another Gaussian function centered around 75kg, considering an average weight range. This function assigns a higher degree of membership to weights close to 75kg, with the membership degree decreasing for weights further away from this center. This approach models the fuzzy concept of "medium weight".
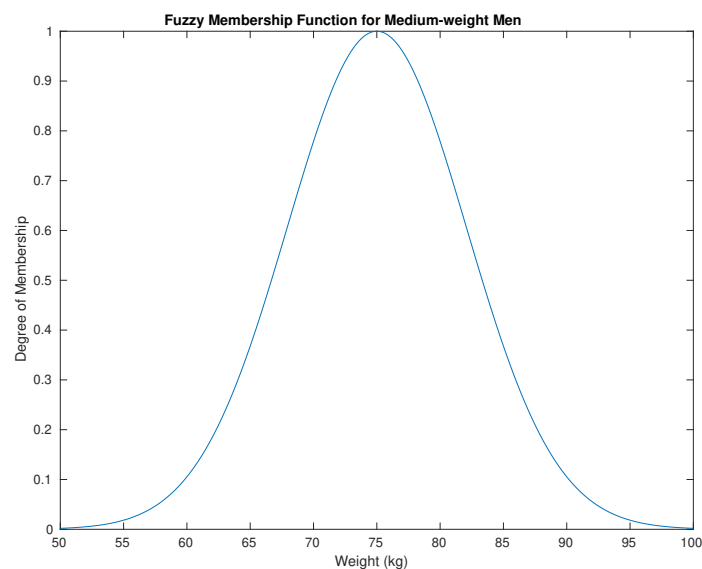

Figure 15: Fuzzy membership function for Medium_weighted Men

NUMBERS APPROXIMATELY BETWEEN 10-20
For numbers approximately between 10 and 20, a trapezoidal membership function is employed to model

numbers in this range, providing a clear illustration of numbers with a high degree of membership strictly within the 10 to 20 range, and a linear decrease to 0 as numbers diverge from this range. This function effectively captures the fuzzy boundaries of being "approximately between" two values.
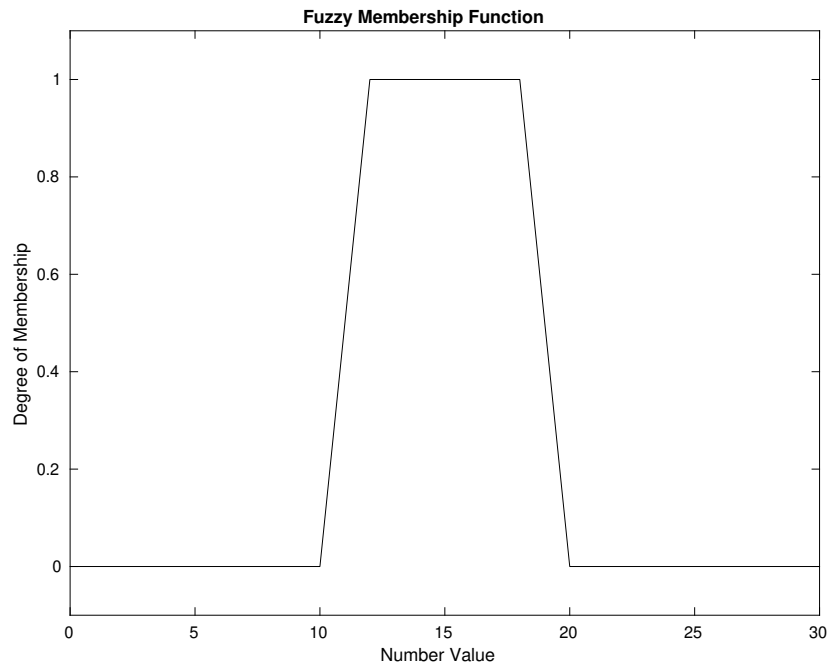

Figure 16: Fuzzy membership function for Numbers approximately between 10 and 20

To summarize, modeling expressions as fuzzy subsets involves assigning a degree of membership to each element relative to a particular set based on its characteristics. In fuzzy logic, the choice of membership function (such as sigmoid, Gaussian, or trapezoidal) depends on the specific context and the nature of the fuzzy set being modeled. These functions help in translating vague or imprecise terms into a quantitative framework that can be analyzed and manipulated mathematically.

Problem 8

The term "ordinary subset of level α in fuzzy set theory refers to a non-fuzzy subset that is created from a fuzzy subset by taking into consideration only those elements whose degrees of membership are at least α. In essence, it's a threshold-based selection from a fuzzy set in which each element is only included if its membership score is greater than or equal to α.
Conversely, this idea is extended to pairs of components for a fuzzy relation by the "ordinary relation of level α". A set of ordered pairs with corresponding degrees of membership that indicate the strength of the relationship between the elements form a fuzzy relation. Thus, the set of all pairs whose membership in the fuzzy relation is at least α is the ordinary relation of level α.

Analytically, the ordinary relation of level α for a fuzzy relation is defined as the set of all pairs (x, y) for which the membership function $\mu_{\tilde{R}}(x, y)$ is greater than or equal to α. Alternatively, it is a crisp set that is derived from the fuzzy relation by incorporating all element pairs with a degree of membership greater than the specified level α

For a fuzzy relation with a membership function $\mu_{\tilde{R}}(x, y) = 1 - \frac{1}{1+x^2+y^2}$, the ordinary relation of level 0.3 is the set:

$$R_{0.3} = (x, y) | \mu_{\tilde{R}}(x, y) \geq 0.3 \tag{1}$$

This means you are looking for all the pairs (x, y) where the membership value is at least 0.3.

To determine analytically the ordinary relation of level 0.3, we will solve the inequality

$$\mu_{\tilde{R}}(x, y) \geq 0.3$$
$$1 - \frac{1}{1 + x^2 + y^2} \geq 0.3$$
$$\frac{1}{1 + x^2 + y^2} \leq 0.7$$
$$1 + x^2 + y^2 \geq \frac{1}{0.7}$$
$$x^2 + y^2 \geq \frac{1}{0.7} - 1$$
$$x^2 + y^2 \geq \frac{3}{7}$$

From basic math we know that $x^2 + y^2$ is the equation of a circle. This represents the region outside a circle centered at the origin with a radius of $\sqrt{\frac{3}{7}}$. This means the ordinary relation of level 0.3 includes all (x, y) pairs whose distance from the origin is at least $\sqrt{\frac{3}{7}}$.

To visualize this, we will plot a circle centered at the origin with a radius of $\sqrt{\frac{3}{7}}$. The area outside this circle represents the ordinary relation of level 0.3.
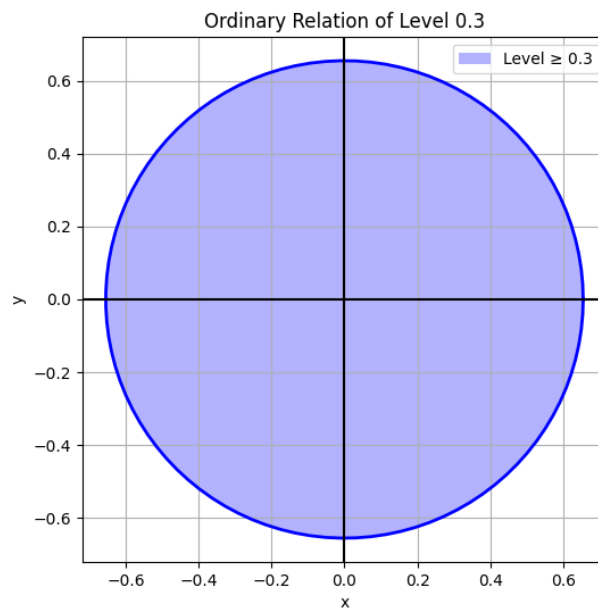

Figure 17: Ordinary relation of Level 0.3

The plot above represents the ordinary relation of level 0.3 for the given fuzzy relation. The blue shaded area, including the boundary, otside the circle indicates the region where the membership function is greater than or equal to 0.3. This area corresponds to all points (x, y) whose distance from the origin is at least $\sqrt{\frac{3}{7}}$, as derived analytically. The boundary circle itself represents the exact threshold where the membership value 0.3 and the area outside this circle encompasses all points with a membership value greater than 0.3., fulfilling the condition for the ordinary relation of level 0.3.

The area enclosed by the circle is: Area = 1.346396851533

Problem 9

To calculate the index of fuzziness v of a fuzzy subset A defined on a fuzzy subset A defined on a reference set $E = [0, \alpha] \subseteq \mathbb{R}$ with a membership function, we can use the following general formula for the index of fuzziness:

$$\nu_L(\tilde{A}) = \frac{1}{|E|} \sum_{x \in E} \min(\mu_{\tilde{A}}(x), 1 - \mu_{\tilde{A}}(x)) \tag{2}$$

In a continuous domain, this becomes:

$$\nu_L(\tilde{A}) = \frac{1}{\text{measure}(E)} \int_E \min(\mu_{\tilde{A}}(x), 1 - \mu_{\tilde{A}}(x)) \, dx \tag{3}$$

Question A

Given the specific membership function $\mu_{\tilde{A}}(x) = \frac{x^2}{\alpha^2}$, for $x \in [0, \alpha]$ and assuming $E = [0, \alpha]$ the linear index of fuzziness can be calculated directly because the measure of E is . Therefore, the formula for the linear index of fuzziness over the interval $[0,]$ is:

$$\nu_L(\tilde{A}) = \frac{1}{\alpha} \int_0^\alpha \min\left(\frac{x^2}{\alpha^2}, 1 - \frac{x^2}{\alpha^2}\right) dx \tag{4}$$

However, since $\frac{x^2}{\alpha^2}$ increases monotically from 0 to 1 as x goes from 0 to α. and the function $\min\left(\frac{x^2}{\alpha^2}, 1 - \frac{x^2}{\alpha^2}\right)$ is symmetric around $x = \frac{\alpha}{\sqrt{2}}$, where $\frac{x^2}{\alpha^2} = 1 - \frac{x^2}{\alpha^2}$, the integral simplifies to 2 times the integral from 0 to $\frac{\alpha}{\sqrt{2}}$:

$$\nu_L(\tilde{A}) = \frac{2}{\alpha} \int_0^{\frac{\alpha}{\sqrt{2}}} \frac{x^2}{\alpha^2} \, dx \tag{5}$$

$$\nu_L(\tilde{A}) = \frac{2}{\alpha^3} \int_0^{\frac{\alpha}{\sqrt{2}}} x^2 \, dx \tag{6}$$

$$\nu_L(\tilde{A}) = \frac{2}{\alpha^3} \left[\frac{x^3}{3}\right]_0^{\frac{\alpha}{\sqrt{2}}} \tag{7}$$

$$\nu_L(\tilde{A}) = \frac{2}{\alpha^3} \cdot \frac{\left(\frac{\alpha}{\sqrt{2}}\right)^3}{3} \tag{8}$$

$$\nu_L(\tilde{A}) = \frac{2}{\alpha^3} \cdot \frac{\alpha^3}{3\sqrt{8}} \tag{9}$$

$$\nu_L(\tilde{A}) = \frac{\sqrt{2}}{3} \tag{10}$$

This result represents the linear index of fuzziness of A which is independent of α due to the normalization over the interval $[0, \alpha]$.

Question B
Given the membership function

$$\mu_{\tilde{A}}(x) = \begin{cases} \dfrac{4x^2}{\alpha^2} & \text{if } 0 \leq x \leq \dfrac{\alpha}{2}, \\ \dfrac{4(x-\alpha)^2}{\alpha^2} & \text{if } \dfrac{\alpha}{2} < x \leq \alpha. \end{cases} \tag{11}$$

The calculation for the index of fuzziness $v(\tilde{A})$ involves evaluating two integrals:

$$\nu(\tilde{A}) = \frac{1}{\alpha} \left(\int_0^{\frac{\alpha}{2}} \left| \frac{4x^2}{\alpha^2} - 0.5 \right| dx + \int_{\frac{\alpha}{2}}^{\alpha} \left| \frac{4(x-\alpha)^2}{\alpha^2} - 0.5 \right| dx \right) \tag{12}$$

After simplifying, this expression remains in terms of integrals of absolute values:

$$\frac{1}{\alpha} \left(\int_0^{\frac{\alpha}{2}} \left| 0.5 - \frac{4x^2}{\alpha^2} \right| dx + \int_{\frac{\alpha}{2}}^{\alpha} \left| 3.5 - \frac{8x}{\alpha} + \frac{4x^2}{\alpha^2} \right| dx \right) \tag{13}$$

To proceed, we need to evaluate these integrals explicitly. However, the complexity of absolute value functions requires careful piecewise integration. For an explicit solution, these integrals would be evaluated considering the behavior of the functions within the given intervals. Given the complexity of directly integrating these absolute value functions symbolically, a numerical approach or further simplification by analyzing the sign changes in the functions over the intervals may be needed for a complete solution.

Let's attempt a further simplification by evaluating the expressions within the absolute values to determine the points of sign change and then integrate accordingly.

The critical points for the functions within the absolute values are:

- For $\dfrac{4x^2}{\alpha^2} - 0.5$:

  The critical points are at $x = \pm 0.354\alpha$, but given the restriction $0 \leq x \leq \dfrac{\alpha}{2}$, only $\mathbf{x = 0.354\alpha}$ is acceptable.

- For $\dfrac{4(x-\alpha^2}{\alpha^2} - 0.5$:

  The critical points are at $x = 0.646\alpha$ and $x = 1.354\alpha$, but given the restriction $\dfrac{\alpha}{2} \leq x \leq \alpha$, only $\mathbf{x = 0.646\alpha}$ is acceptable.

Since we're interested in the intervals $0 \leq x \leq \dfrac{\alpha}{2}$ and $\dfrac{\alpha}{2} < x \leq \alpha$, we recognize that the function changes behavior at these critical points.

Given the complexity and to provide a precise answer, let's refine the approach by directly integrating the functions over their specified intervals, acknowledging that the absolute value's impact was incorrectly applied. We should calculate the fuzziness index by integrating the membership function's deviation from 0.5 where applicable, considering the piecewise nature of the function:

- $\int_0^{\frac{\alpha}{2}} \left| \frac{4x^2}{\alpha^2} - 0.5 \right| dx$

- $\int_{\frac{\alpha}{2}}^{\alpha} \left| \frac{4(x-\alpha)^2}{\alpha^2} - 0.5 \right| dx$

Problem 10

The min-max composition of fuzzy relations is a method for combining two fuzzy relations to produce a new fuzzy relation. The process is somewhat analogous to matrix multiplication in linear algebra but uses the "min" and "max" operations instead of multiplication and addition.

Given two fuzzy relations $R1$ and $R2$, with membership functions $\mu_{\tilde{R1}}(x, y)$ and $\mu_{\tilde{R2}}(y, z)$ respectively, the min-max composition $R = R1 \circ R2$ has a membership function defined for each pair (x, z) by:

$$\mu_R(x, z) = \max_y \min(\mu_{R1}(x, y), \mu_{R2}(y, z) \tag{14}$$

In order to gain a deeper understanding, , we will analyse the expression

- $\min(\mu_{R1}(x, y), \mu_{R2}(y, z))$: For a fixed x and z, and for each intermediate value of y, we must calculate the minimum membership value between $\mu_{\tilde{R1}}(x, y)$ and $\mu_{\tilde{R2}}(y, z)$. This minimum value represents the strongest constraint on the relationship between x and z via the intermediary y.

- $\max_y$: Select the maximum value out of all these minimum values as the membership degree for the pair (x, z) in the relation R.

Now, let's analyze the exponential terms within the membership functions. We have:

$$\mu_{\tilde{R1}}(x, y) = e^{-k(x-y)^2}, \quad \text{for } k \geq 1 \tag{15}$$

$$\mu_{\tilde{R2}}(y, z) = e^{-k(y-z)^2}, \quad \text{for } k \geq 1 \tag{16}$$

The exponential function $e^{-k(w)^2}$ is a Gaussian function that has its maximum value when $w = 0$ and decreases as w moves away from 0 in either direction. This means that for a fixed x and z, the membership values $\mu_{\tilde{R1}}(x, y)$ and $\mu_{\tilde{R2}}(y, z)$ will be higher when y is close to x and z respectively, because the terms $(x - y)^2$ and $(y - z)^2$ will be smaller, making the exponent closer to 0.

When we take the minimum of these two membership values, we are essentially looking for the point where the overlap between the two functions is the greatest, since the minimum value is dominated by the smaller of the two membership degrees.

In other words, since the membership functions involve an exponential function of a squared term, the minima will occur where the arguments of the exponential functions are closest to each other, that is, when $(x - y)^2$ and $(y - z)^2$ are minimal.

Graphically, we can represent the membership functions μ_{R1} and μ_{R2} as surfaces over $x - y$ and $y - z$ planes, respectively. The composition μ_R would be a new surface over the $x - z$ plane, which is derived from the "peaks" of the minimum values between the two original surfaces for every x and z. We will assume a value for k to proceed with the calculation. Let's set $k = 1$ for simplicity. We will first define the membership functions and then compute the max-min composition for a range of x and z values.
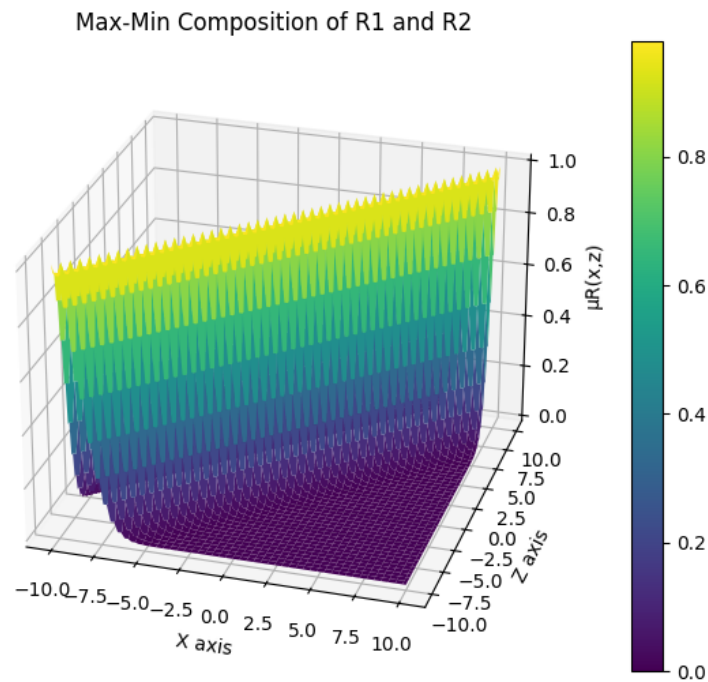
Figure 18: Max-Min Composition of Fuzzy Relations R1 and R2

The 3D plot above shows the Max-Min composition of the fuzzy relations R_1 and R_2 over the range of x and z values. The surface represents the degree of membership in the composite relation for each pair (x, z), calculated as the maximum of the minimum values of $\mu_{\tilde{R1}}(x, y)$ and $\mu_{\tilde{R2}}(y, z)$ across all posible values of y. The height of the surface at any point indicates the strength of the relation between x and z in the composite relation R.