Assignment 3
ECE 657A Winter 2022
Group 1

Jehanzeb Mirza
21115879

Mehmet Ali Pelit
21113622

2024-04-08

Introduction

Our objective for this assignment is to gain experience using Neural Networks and Fuzz Inference Systems. We have 6 problems in this assignment specified below.

1. Feedforward Back-Propogation Neural Network
2. Multilayer Perceptron
3. Membership functions and Linguistic Hedge
4. Fuzzy Measures
5. Showing t-norms
6. Interpreting Fuzzy Membership

Our invesitgation makes use of code written in Python and the source files, generated figures, and datasets are provided as supplementary files in addition to this report. We extensively use Numpy [1] and Pandas libraries for data management and Sci-kit learn's [2] implementations of algorithms. We use Pytorch for machine learning.

1 Feedforward Back-Propogation Neural Network

In this section we attempt to build a simple single hidden layer neural network.

We attempt to model the following two functions

$$f_1(x) = x * sin(6\pi x) * exp(-x^2) \qquad x \in [-1, 1] \qquad (1)$$

$$f_2(x) = exp(-x^2) * arctan(x) * sin(4\pi x) \qquad x \in [-2, 2] \qquad (2)$$

Plots of these functions can be seen below.

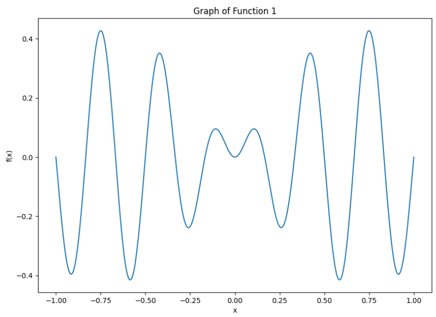


Figure 1: Function 1

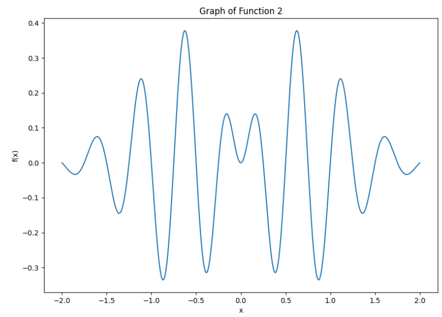


Figure 2: Function 2

For each function we linearly sample 1000 points in the functions domain and then we take 70% of those points as training samples, and 30% as testing samples.

1.1 Fixed Hidden Nodes - Variable Training Samples

For the first part, we investigate what happens when we adjust the number of samples used for training the neural network while leaving the architecture fixed.

We plot the training loss vs number of epocs (1000 epocs for speed and convergence reasons). Below for 20 samples and 100 samples.

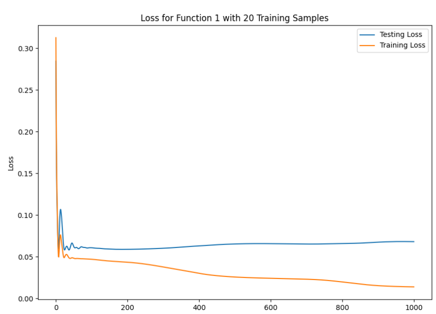


Figure 3: Function 1 Losses with 20 samples

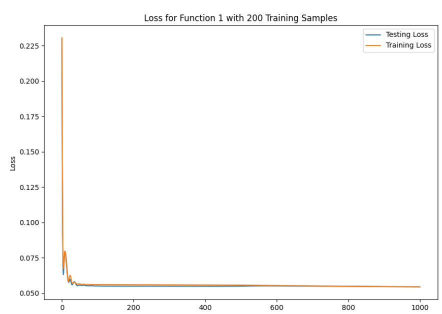


Figure 4: Function 1 Losses with 200 samples

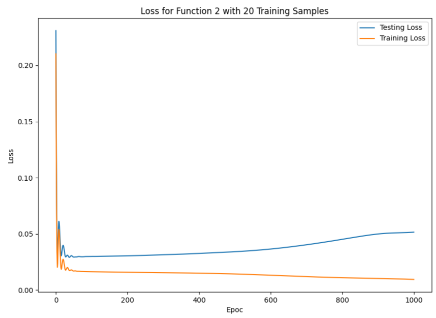


Figure 5: Function 2 Losses with 20 samples

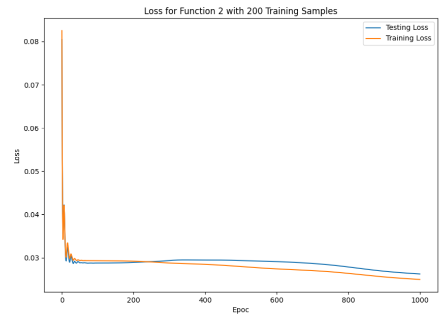


Figure 6: Function 2 Losses with 200 samples

The fixed architecture used is 1 input layer, a 3 node hidden layer, and 1 output layer. We used a sigmoid function / logistic function for the activation function of our neural network.

What we do here is that we train the network with only a random subsample of our original training samples. Our sample sizes is 20, 40, 60, 80, 100, 120, 140, 180, and 200 samples.

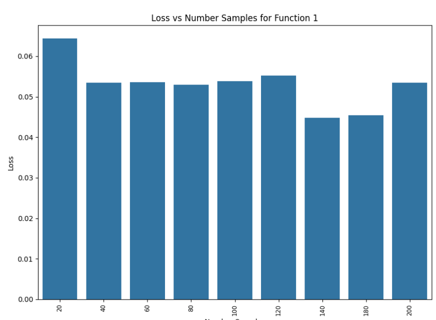


Figure 7: Function 1 Losses vs number of samples

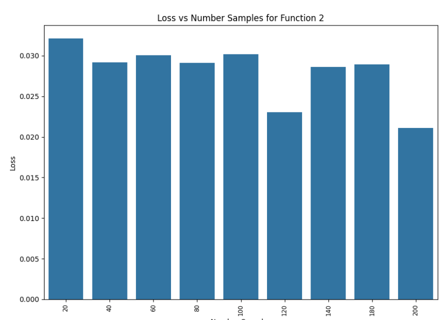


Figure 8: Function 2 Losses vs number of samples

We can see that it seems like for function 1 140 samples works best while for function 2 200 samples works well.

1.2 Variable Hidden Nodes - Fixed Training Samples

In this second part, we investigate what happens when we adjust the number of nodes used in the hidden layer for the neural networks while keeping the number of training samples fixed (100 training samples).

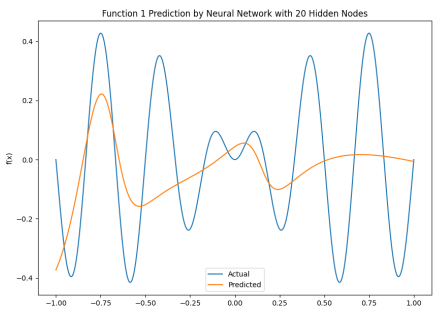


Figure 9: Function 1 Prediction with 20 nodes

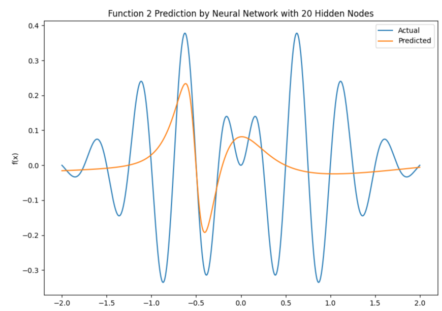


Figure 10: Function 2 Prediction with 20 nodes

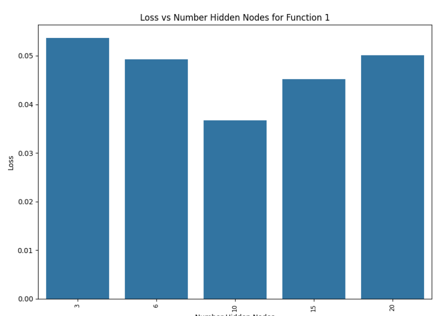


Figure 11: Function 1 Losses vs number of nodes

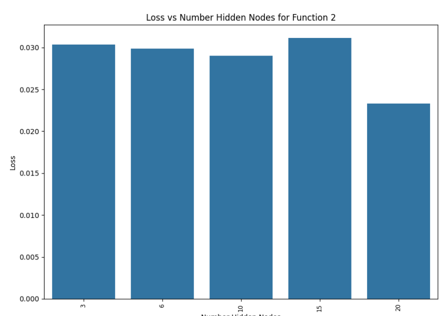


Figure 12: Function 2 Losses vs number of nodes

What we can see is that it seems for function 1 10 nodes works best and for function 2 20 nodes works best. However we can see that the neural network in general does a fairly terrible job of predicting the function.

1.3 Deductions

In general our NN does not work well for predicting this data. Our belief is that the number of parameters is far too low to model such complex functions and we need many more layers and hidden nodes in order to accuracy model it.

More nodes works better for this and more data does as well in a sort of general trend.

2 Multilayer Perceptron

In this section we train a multilayer perceptron to obtain the output of the following function

$$f(x_1, x_2) = sin(2\pi x_1) * cos(0.5\pi x_2) * exp(-x_1^2) \quad (3)$$

$$(x1, x2) \in [-1, 1] \times [-4, 4] \quad (4)$$

2.1 Fixed Hidden Nodes

In this section we fix the number of hidden neurons to 4 (double the number of input nodes) and analysze the performance of the neural network. The network still has one hidden layer just as in part 1.

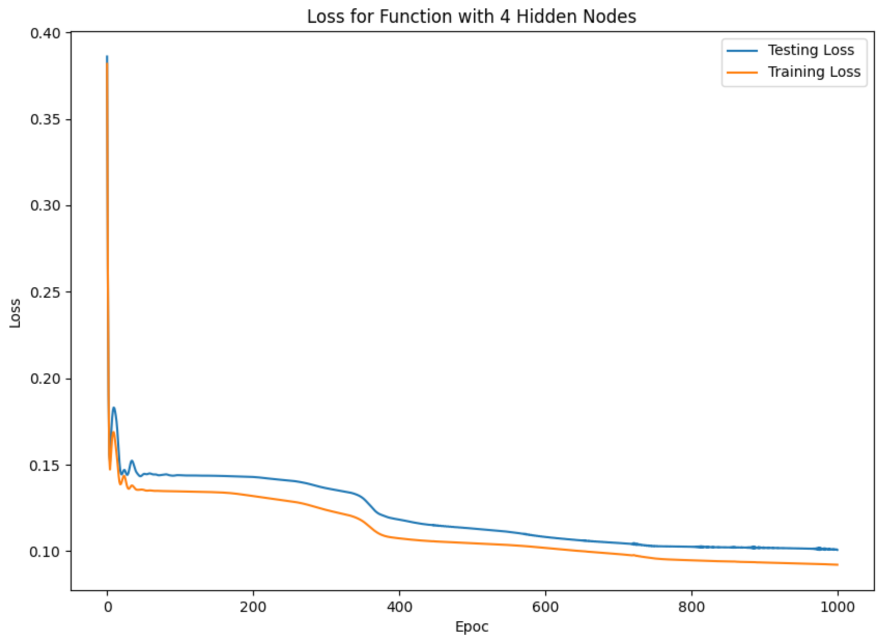


Figure 13: Losses with 4 nodes

Our loss goes down fairly quickly but it bottoms out at a relatively high amount of 0.1 so it probably doesn't work well.

2.2 More and fewer Hidden nodes

In this section we expand on the analysis above and instead the following number of hidden nodes: (2, 6, 8, 12, 20). We do this to try to find the best number of hidden neurons.

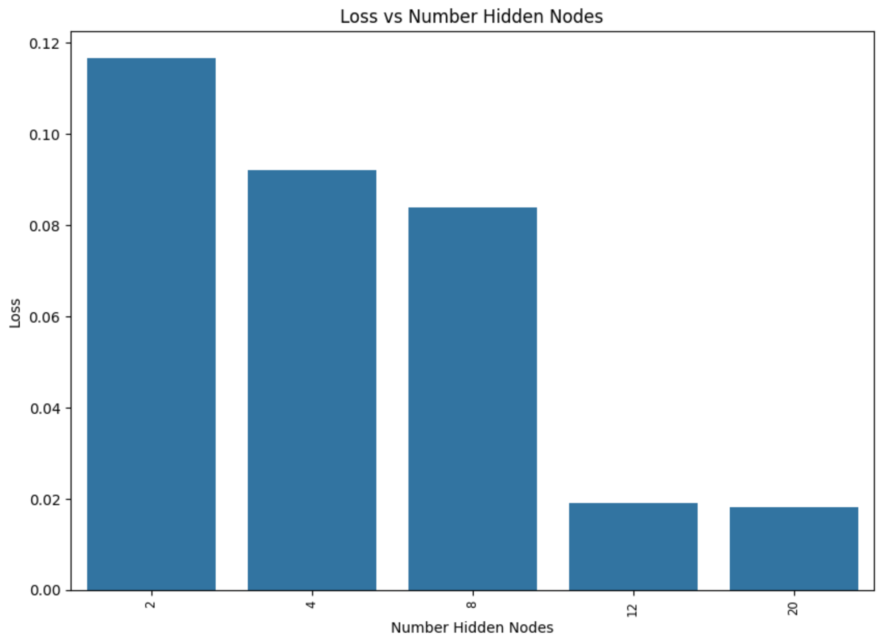


Figure 14: Losses vs number of nodes

We can see that the multilayer perceptron is best with the largest number of nodes (12 and 20). We can conclude that the network does not do a great job of prediciting the function.

3 Membership Functions and Linguistic Treatment

3.1 Linguistic Hedges

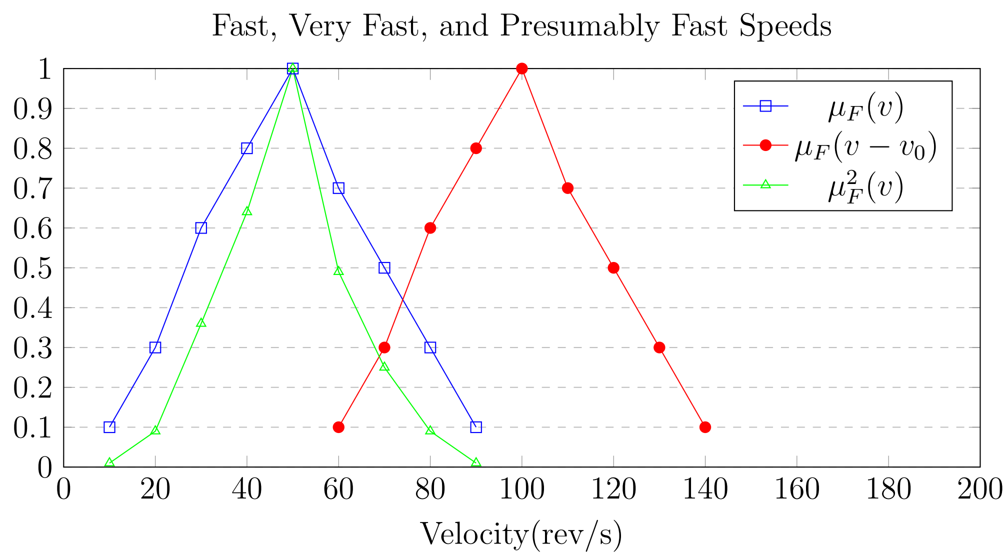
- Substracting v_0 from the velocity argument shift the membership values horizantally. Since $v_0 > 0$, it can be said that the velocity values will increase. The membership function will shift to the right. This is an appropriate approach for *very fast*.
- Squaring the membership function results in contraction. Higher degree is achieved, i.e., the certainity is increased. Values counted as far to

fast now have low degree, and values close to *fast* have high degree. This is an appropriate approach for *presumably fast*.

3.2 Speed

$$\text{Fuzzy Set of Very Fast} = \{\frac{0.1}{60}, \frac{0.3}{70}, \frac{0.6}{80}, \frac{0.8}{90}, \frac{1.0}{100}, \frac{0.7}{110}, \frac{0.5}{120}, \frac{0.3}{130}, \frac{0.1}{140}\}$$

$$\text{Fuzzy Set of Presumably Fast} = \{\frac{0.01}{10}, \frac{0.09}{20}, \frac{0.36}{30}, \frac{0.64}{40}, \frac{1.0}{50}, \frac{0.49}{60}, \frac{0.25}{70}, \frac{0.09}{80}, \frac{0.01}{90}\}$$



3.3 Power

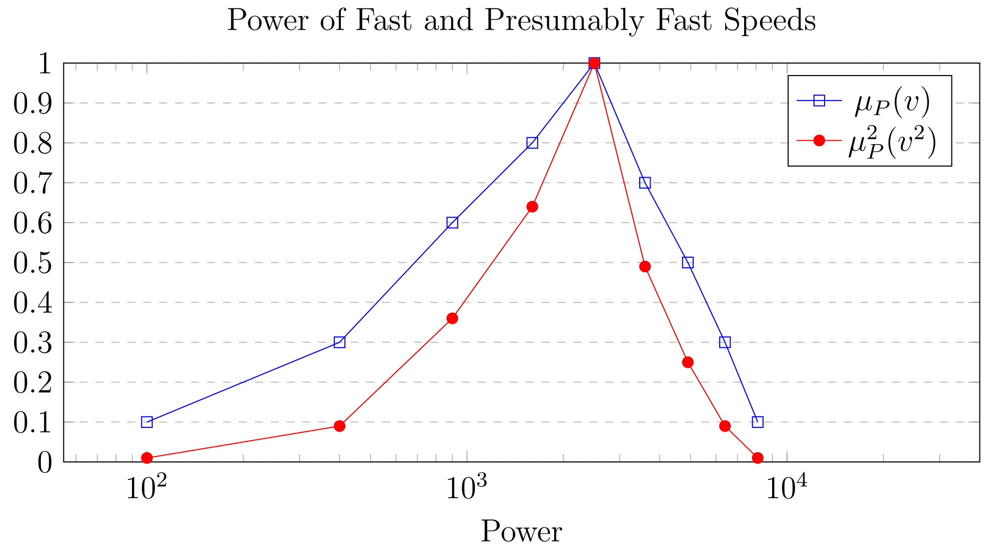
$$p = v^2$$

By Extension Principle:

$$V \rightarrow V^2 = \{0, 100, 400, ..., 36100, 40000\}$$

$$\text{Fuzzy Set of Fast Speed Power} = \{\frac{0.1}{100}, \frac{0.3}{400}, \frac{0.6}{900}, \frac{0.8}{1600}, \frac{1.0}{2500}, \frac{0.7}{3600}, \frac{0.5}{4900}, \frac{0.3}{6400}, \frac{0.1}{8100}\}$$

$$\text{Fuzzy Set of Presumably Fast Power} = \{\frac{0.01}{100}, \frac{0.09}{400}, \frac{0.36}{900}, \frac{0.64}{1600}, \frac{1.0}{2500}, \frac{0.49}{3600}, \frac{0.25}{4900}, \frac{0.09}{6400}, \frac{0.01}{8100}\}$$

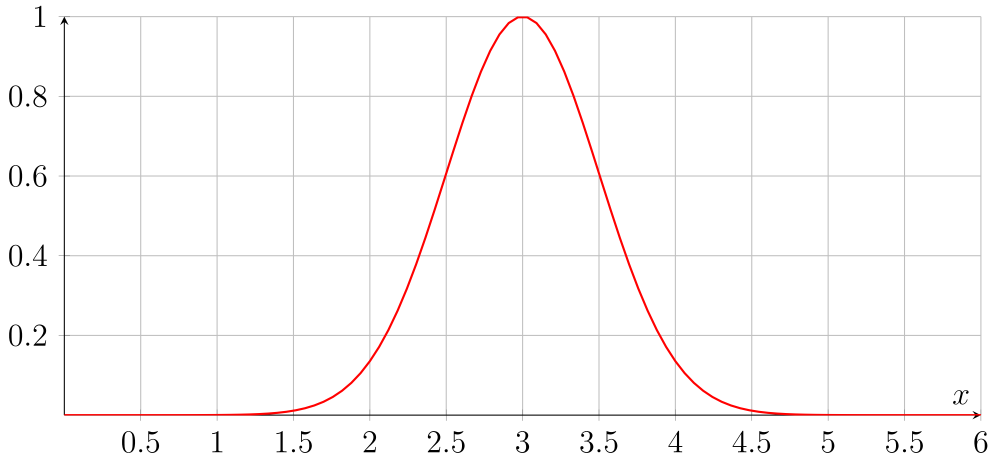


The power calculation of *presumably fast* has less fuzzy output than *fast*'s power calculation.

4 Degree of Fuzziness

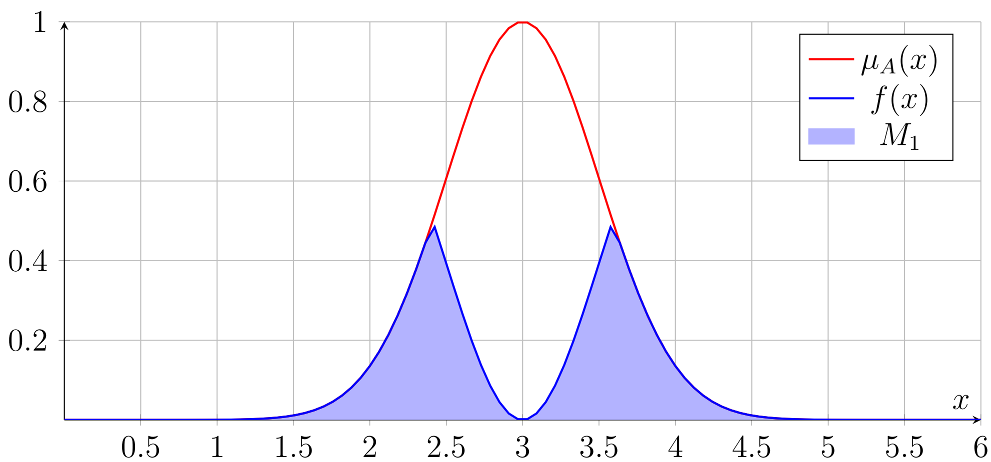
Membership function is $\mu_A(x) = e^{-\lambda(x-a)^n}$ for $\lambda = 2$, $n = 2$, and $a = 3$ for the support set $S = [0, 6]$.

Plot of $\mu_A(x)$



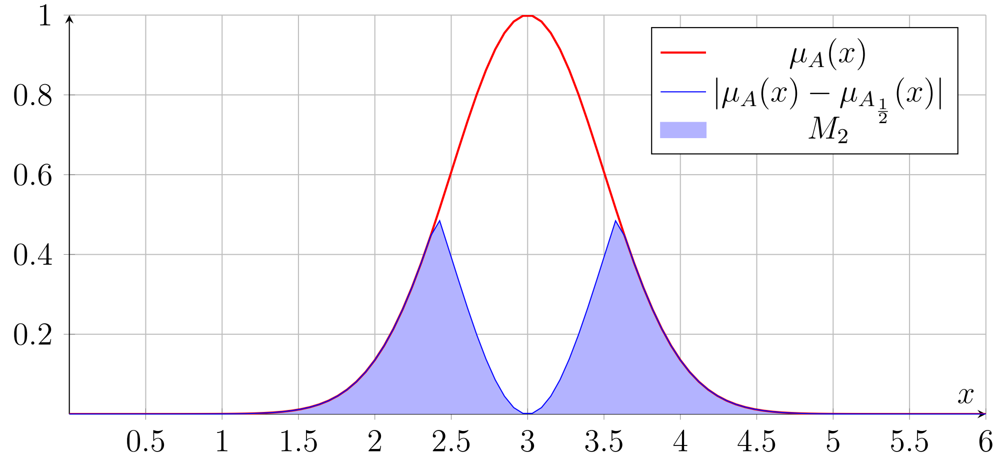
$$M_0 = \int_S \mu_A(x)\,dx \approx 1.253$$

Plot of $\mu_A(x)$ and $f(x)$



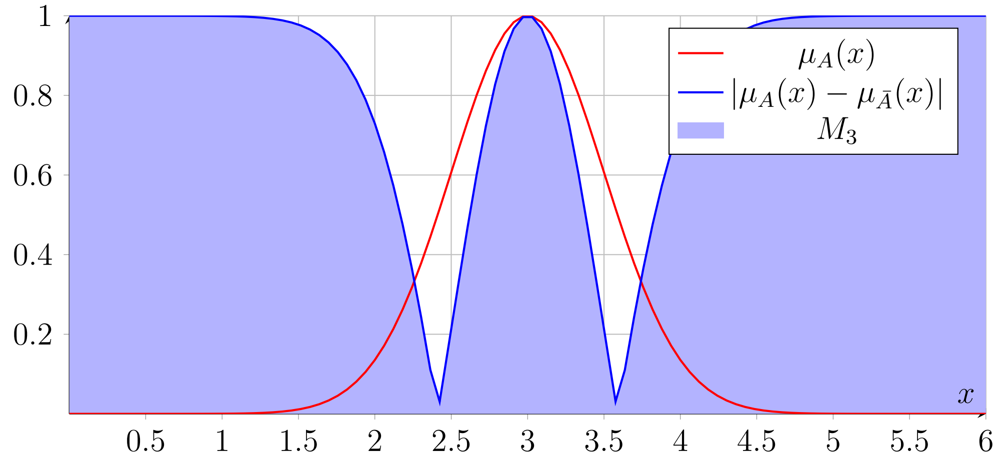
$$M_1 = \int_S f(x)\,dx \approx 0.523 \text{ where } f(x) = \begin{cases} \mu_A(x) & \text{if } \mu_A(x) \leq 0.5 \\ 1 - \mu_A(x) & \text{if } \mu_A(x) > 0.5 \end{cases}$$

Plot of $\mu_A(x)$ and $|\mu_A(x) - \mu_{A_{\frac{1}{2}}}(x)|$



$$M_2 = \int_S |\mu_A(x) - \mu_{A_{\frac{1}{2}}}(x)|\, dx \approx 0.523$$

Plot of $\mu_A(x)$ and $|\mu_A(x) - \mu_{\bar{A}}(x)|$



$$M_3 = \int_S |\mu_A(x) - \mu_{\bar{A}}(x)|\, dx \approx 4.954$$

4.1

$$M_1 = \int_S f(x)\, dx \quad \text{where} \quad f(x) = \begin{cases} \mu_A(x) & \text{for } \mu_A(x) \le 0.5 \\ 1-\mu_A(x) & \text{for } \mu_A(x) > 0.5 \end{cases}$$

Assume $\mu_A(x_1) = \mu_A(x_2) = 0.5$ where $x_1, x_2 \in S$ and $x_2 > x_1$

$$M_1 = \int_0^{x_1} \mu_A(x)\, dx + \int_{x_1}^{x_2} 1-\mu_A(x)\, dx + \int_{x_2}^{6} \mu_A(x)\, dx$$

$$= 2\left[\int_0^{x_1} \mu_A(x)\, dx + \int_{x_1}^{3} 1-\mu_A(x)\, dx \right]$$

$$M_2 = \int_S |\mu_A(x) - \mu_{A1/2}(x)|\, dx \quad \text{where} \quad \mu_{A1/2}(x) = \begin{cases} 1 & \text{for } \mu_A(x) \ge 0.5 \\ 0 & \text{for } \mu_A(x) < 0.5 \end{cases}$$

Again, same assumption : $\mu_A(x_1) = \mu(x_2) = 0.5$

$$M_2 = \int_0^{x_1} |\mu_A(x)|\, dx + \int_{x_1}^{x_2} |\mu_A(x) - 1|\, dx + \int_{x_2}^{6} |\mu_A(x)|\, dx$$

$$= 2\left[\int_0^{x_1} |\mu_A(x)|\, dx + \int_{x_1}^{3} |\mu_A(x) - 1|\, dx \right]$$

$$= 2\left[\int_0^{x_1} \mu_A(x)\, dx + \int_{x_1}^{3} 1-\mu_A(x)\, dx \right]$$

$$M_3 = \int_S |\mu_A(x) - \mu_{\bar{A}}(x)| \, dx$$

$$M_3 = \int_0^6 |\mu_A(x) - (1 - \mu_A(x))| \, dx$$

$$= \int_0^6 |2\mu_A(x) - 1| \, dx \qquad |2\mu_A(x) - 1| = \begin{cases} 2\mu_A(x) - 1, & \mu_A(x) \geq 0.5 \\ 1 - 2\mu_A(x), & \mu_A(x) < 0.5 \end{cases}$$

$$= \int_0^{x_1} 1 - 2\mu_A(x) \, dx + \int_{x_1}^{x_2} 2\mu_A(x) - 1 \, dx + \int_{x_2}^{6} 1 - 2\mu_A(x) \, dx$$

$$= 2 \left[\int_0^{x_1} 1 - 2\mu_A(x) \, dx + \int_{x_1}^{3} 2\mu_A(x) - 1 \, dx \right]$$

Further of calculations were not conducted because of their complexity. The relation between M_1, M_2 and M_3 is as below.

$M_1 = M_2 = \frac{1}{2}(S * 1 - M_3)$

4.2

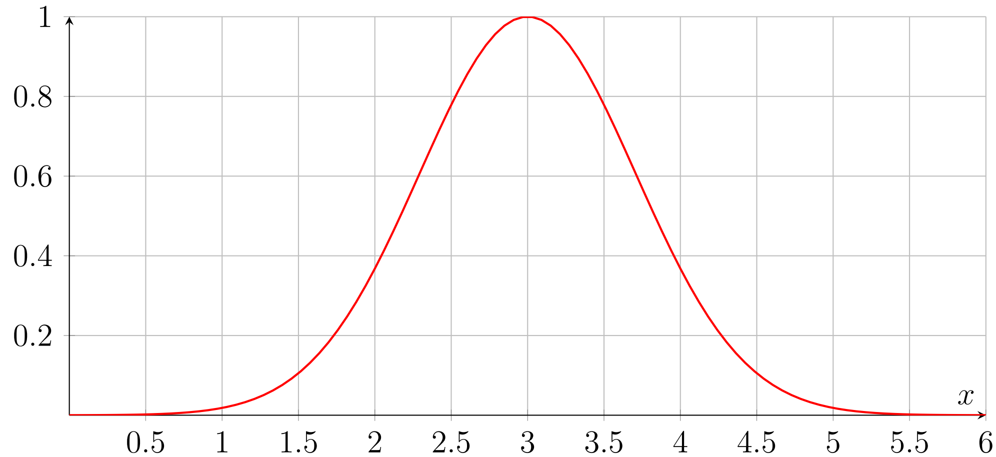
- M_1 shows the closeness to grade 0.5.
- M_2 shows the distance from 1/2-cut.
- M_3 shows the inverse of distance from complement.

Both of the measures M_1 and M_2 define the closeness to the most fuzzy grade. On the other hand, M_3 gives the closeness to certainity.

4.3

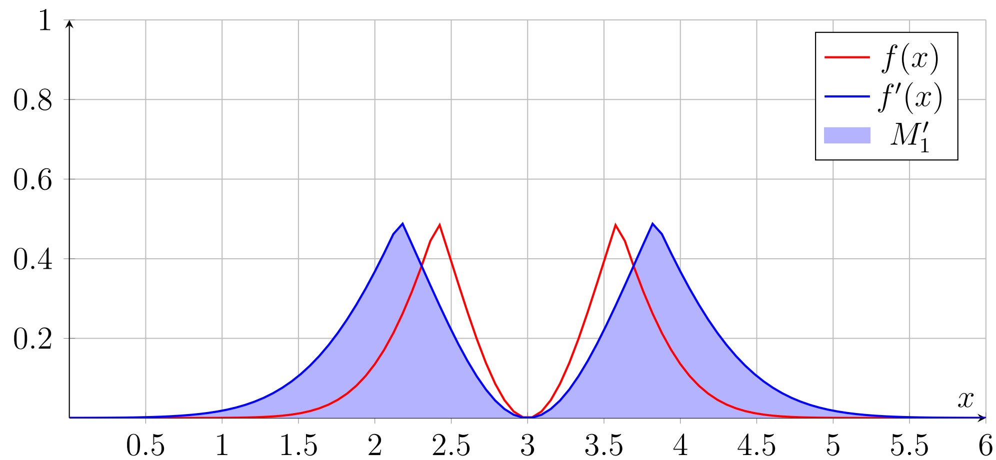
New membership function is $\mu'_A(x) = e^{-\lambda(x-a)^n}$ for $\lambda = 1$, $n = 2$, and $a = 3$ for the support set $S = [0, 6]$.

Plot of $\mu'_A(x)$



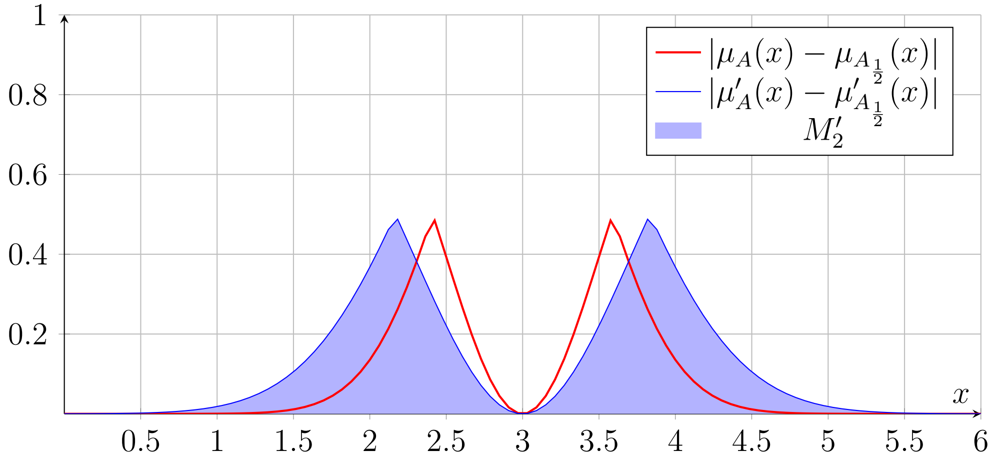
$$M'_0 = \int_S \mu'_A(x)\,dx \approx 1.772$$

Plot of $f(x)$ and $f'(x)$



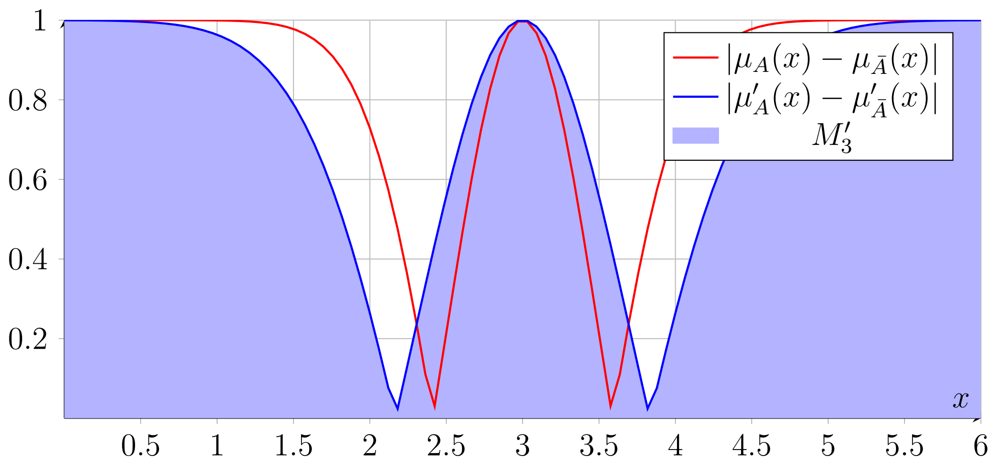
$$M'_1 = \int_S f'(x)\,dx \approx 0.740 \text{ where } f'(x) = \begin{cases} \mu'_A(x) & \text{if } \mu'_A(x) \leq 0.5 \\ 1 - \mu'_A(x) & \text{if } \mu'_A(x) > 0.5 \end{cases}$$

Plot of $|\mu_A(x) - \mu_{A_{\frac{1}{2}}}(x)|$ and $|\mu'_A(x) - \mu'_{A_{\frac{1}{2}}}(x)|$



$$M'_2 = \int_S |\mu'_A(x) - \mu'_{A_{\frac{1}{2}}}(x)|\, dx \approx 0.740$$

Plot of $|\mu_A(x) - \mu_{\bar{A}}(x)|$ and $|\mu'_A(x) - \mu'_{\bar{A}}(x)|$



$$M'_3 = \int_S |\mu'_A(x) - \mu'_{\bar{A}}(x)|\, dx \approx 4.520$$

	M_0	M_1	M_2	M_3
$\lambda = 2$	1.253	0.523	0.523	4.954
$\lambda = 1$	1.772	0.740	0.740	4.520

Table 1

If we compare the values of measures, we can see that the new membership function with $\lambda = 1$ has more uncertainity ($M_1 < M_1'$). This is expected because the variance increases by decreasing λ, hence the fuzzy part of the function gets wider.

5 Bounded Product

5.1 T-Norm

$$T(x,y) = \max[0, x+y-1]$$

① Commutativity

$$T(y,x) = \max[0, y+x-1] = \max[0, x+y-1] = T(x,y)$$

② Associativity

$$T(x, T(y,z)) = \max[0, x + T(y,z) - 1]$$

$$= \max[0, x + \max[0, y+z-1] - 1]$$

$$T(T(x,y), z) = \max[0, T(x,y) + z - 1]$$

$$= \max[0, \max[0, x+y-1] + z - 1]$$

Assume $y+z-1>0$ and $x+y-1>0$

$$T(x, T(y,z)) = \max[0, x+y+z-2]$$
$$T(T(x,y), z)) = \max[0, x+y+z-2]$$
$$T(x, T(y,z)) = T(T(x,y), z)$$

Regerdless of the sign of $(x+y-1)$ and $(y+z-1)$, the associativity will hold because of the linearity inside max function.

3 Non-decreasing

Since both x and y have non negative coefficients, increasing x or y or both can't decrease t-norm.

4 Boundery conditions

$$T(x,1) = \max[0, x]$$
$$= x, \text{ assuming } x \in [0,1]$$

$$T(x,0) = \max[0, x-1]$$
$$= 0, \text{ assuming } x-1 \in [-1,0]$$

$\therefore$ $T(x,y) = \max[0, x+y-1]$ satisfies all the properties of t-norm.

5.2 T-Conorm (S-Norm)

T-conorm (S-norm)

From De morgan's Law:

$$
\begin{aligned}
S(x,y) &= 1 - T(1-x, 1-y) \\
&= 1 - \max[0, (1-x)+(1-y)-1] \\
&= 1 - \max[0, 1-x-y] \\
&= \min[1-0, 1-(1-x-y)] \\
&= \min[1, x+y]
\end{aligned}
$$

6 Application of Fuzziness

6.1 Fuzziness and Fuzzy modifier

- a: shifts the membership function horizontally
- λ: changes the variance of the function
- n: changes the steepness of the function

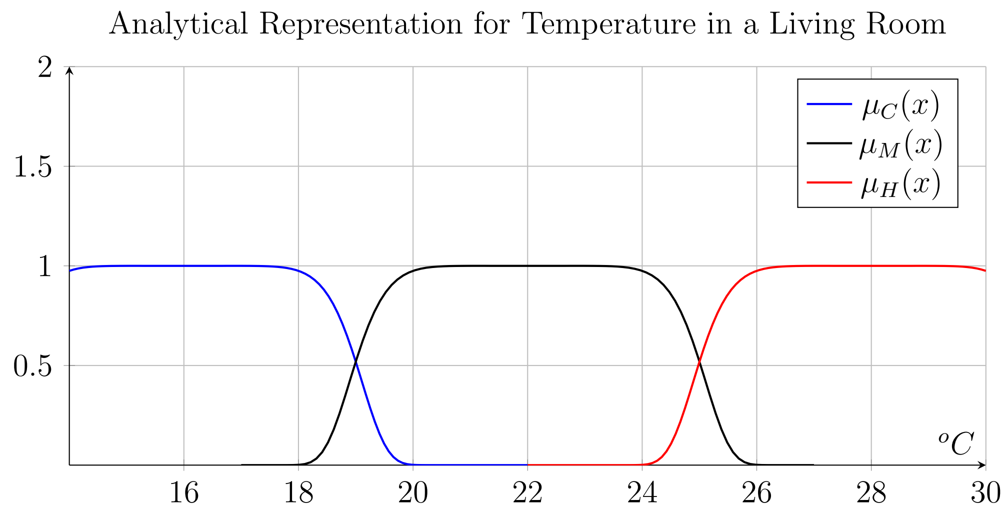
	Fuzziness	**Fuzzy modifier**
a	no change	no change
λ	inverse relation	small λ: 'somewhat' large λ: 'very'
n	inverse relation	small n: 'somewhat' large n: 'very'

Table 2

With different parameter a, different values in the support set S can be provoked by shifting the membership function over S.

6.2 Temperature Application

- The support set S: [14,30] ^oC
- Fuzzy set C for cold, and membership function is $\mu_C(x)$
- Fuzzy set M for comfortable/mild, and membership function is $\mu_M(x)$
- Fuzzy set H for hot, and membership function is $\mu_H(x)$



$$\mu(x) = e^{-\lambda|x-a|^n}$$

	λ	a	n
$\mu_C(x)$	0.0001	16	8
$\mu_M(x)$	0.0001	22	8
$\mu_H(x)$	0.0001	28	8

Table 3

Conclusion

In conculusion we were successfully able to meet our objectives for this lab and developed experience using neural networks and using fuzzy logic.

References

[1] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.