Apêndice II - Código em "R"

```
# DESEMPENHO ACADÊMICO FUZZY #
# Carregando pacotes do R
require(xlsx)
require(readxl)
require(FuzzyToolkitUoN) # pacote fuzzy da Universidade de Nottingham

# Selecionando o diretorio de trabalho
path="C:\\Users\\Luiz Gaviao\\1ELFA\\Dados"
setwd(path)

# Importando base de dados
dados = read_excel("Dados Fuzzy.xlsx", col_names = FALSE, sheet = 1)

# Parâmetros dos boxplots das disciplinas
# Whisker tradicional de Tukey (coef=1.5)

D1 = boxplot.stats(dados[,1], coef = 1.5)$stats
D2 = boxplot.stats(dados[,2], coef = 1.5)$stats
D3 = boxplot.stats(dados[,3], coef = 1.5)$stats
D4 = boxplot.stats(dados[,4], coef = 1.5)$stats
D5 = boxplot.stats(dados[,5], coef = 1.5)$stats
D6 = boxplot.stats(dados[,6], coef = 1.5)$stats
D7 = boxplot.stats(dados[,7], coef = 1.5)$stats
D8 = boxplot.stats(dados[,8], coef = 1.5)$stats

# Gráficos das disciplinas
disciplinas = cbind(dados[,1],dados[,2],dados[,3],dados[,4],
+dados[,5],dados[,6],dados[,7],dados[,8])
boxplot(disciplinas, col = "lightgray", main="Base de Dados (8 disciplinas)")

# Carregando pacote da Universidade de Nottingham
require(FuzzyToolkitUoN)

Nota = newFIS('Nota') # criando FIS

### Criando funções de pertinência das variáveis (disciplinas)

### Disciplina 1

D1 # identificando os parâmetros das funções de pertinência da Disciplina 1
Nota = addVar(Nota, "input", "Disc.1", 0:10)
MF1.1 = triMF("FCN", 0:10, c(0,0,4.5,1))
MF2.1 = triMF("MAbM", 0:10, c(0,4.5,6,1))
MF3.1 = triMF("PAbM", 0:10, c(4.5,6,6.65,1))
MF4.1 = triMF("Md", 0:10, c(6,6.65,7.3,1))
MF5.1 = triMF("PAcM", 0:10, c(6.65,7.3,8.8,1))
MF6.1 = triMF("MAcM", 0:10, c(7.3,8.8,10,1))
MF7.1 = triMF("FCP", 0:10, c(8.8,10,10,1))
Nota = addMF(Nota, "input", 1, MF1.1)
Nota = addMF(Nota, "input", 1, MF2.1)
Nota = addMF(Nota, "input", 1, MF3.1)
Nota = addMF(Nota, "input", 1, MF4.1)
Nota = addMF(Nota, "input", 1, MF5.1)
Nota = addMF(Nota, "input", 1, MF6.1)
Nota = addMF(Nota, "input", 1, MF7.1)

D2 # identificando os parâmetros das funções de pertinência da Disciplina 2
Nota = addVar(Nota, "input", "Disc.2", 0:10)
MF1.2 = triMF("FCN", 0:10, c(0,0,0.8,1))
MF2.2 = triMF("MAbM", 0:10, c(0,0.8,4.9,1))
MF3.2 = triMF("PAbM", 0:10, c(0.8,4.9,6.45,1))
MF4.2 = triMF("Md", 0:10, c(4.9,6.45,7.7,1))
MF5.2 = triMF("PAcM", 0:10, c(6.45,7.7,9.8,1))
MF6.2 = triMF("MAcM", 0:10, c(7.7,9.8,10,1))
MF7.2 = triMF("FCP", 0:10, c(9.8,10,10,1))
Nota = addMF(Nota, "input", 2, MF1.2)
Nota = addMF(Nota, "input", 2, MF2.2)
Nota = addMF(Nota, "input", 2, MF3.2)
Nota = addMF(Nota, "input", 2, MF4.2)
Nota = addMF(Nota, "input", 2, MF5.2)
Nota = addMF(Nota, "input", 2, MF6.2)
Nota = addMF(Nota, "input", 2, MF7.2)

D3 # identificando os parâmetros das funções de pertinência da Disciplina 3
Nota = addVar(Nota, "input", "Disc.3", 0:10)
MF1.3 = triMF("FCN", 0:10, c(0,0,0.3,1))
MF2.3 = triMF("MAbM", 0:10, c(0,0.3,1.8,1))
MF3.3 = triMF("PAbM", 0:10, c(0.3,1.8,6.1,1))
MF4.3 = triMF("Md", 0:10, c(1.8,6.1,8.2,1))
MF5.3 = triMF("PAcM", 0:10, c(6.1,8.2,10,1))
MF6.3 = triMF("MAcM", 0:10, c(8.2,10,10,1))
# Disciplina SEM outliers positivos (notas altas em geral)
Nota = addMF(Nota, "input", 3, MF1.3)
Nota = addMF(Nota, "input", 3, MF2.3)
Nota = addMF(Nota, "input", 3, MF3.3)
Nota = addMF(Nota, "input", 3, MF4.3)
Nota = addMF(Nota, "input", 3, MF5.3)
Nota = addMF(Nota, "input", 3, MF6.3)

D4 # identificando os parâmetros das funções de pertinência da Disciplina 4
Nota = addVar(Nota, "input", "Disc.4", 0:10)
MF1.4 = triMF("FCN", 0:10, c(0,0,0.5,1))
MF2.4 = triMF("MAbM", 0:10, c(0,0.5,4,1))
MF3.4 = triMF("PAbM", 0:10, c(0.5,4,6.3,1))
MF4.4 = triMF("Md", 0:10, c(4,6.3,7.7,1))
MF5.4 = triMF("PAcM", 0:10, c(6.3,7.7,10,1))
MF6.4 = triMF("MAcM", 0:10, c(7.7,10,10,1))
# Disciplina SEM outliers positivos (notas altas em geral)
Nota = addMF(Nota, "input", 4, MF1.4)
Nota = addMF(Nota, "input", 4, MF2.4)
Nota = addMF(Nota, "input", 4, MF3.4)
Nota = addMF(Nota, "input", 4, MF4.4)
Nota = addMF(Nota, "input", 4, MF5.4)
Nota = addMF(Nota, "input", 4, MF6.4)

D5 # identificando os parâmetros das funções de pertinência da Disciplina 5
Nota = addVar(Nota, "input", "Disc.5", 0:10)
MF1.5 = triMF("FCN", 0:10, c(0,0,6.5,1))
MF2.5 = triMF("MAbM", 0:10, c(0,6.5,7.8,1))
MF3.5 = triMF("PAbM", 0:10, c(6.5,7.8,8.8,1))
MF4.5 = triMF("Md", 0:10, c(7.8,8.8,9.3,1))
MF5.5 = triMF("PAcM", 0:10, c(8.8,9.3,10,1))
MF6.5 = triMF("MAcM", 0:10, c(9.3,10,10,1))
# Disciplina SEM outliers positivos (notas altas em geral)
Nota = addMF(Nota, "input", 5, MF1.5)
Nota = addMF(Nota, "input", 5, MF2.5)
Nota = addMF(Nota, "input", 5, MF3.5)
Nota = addMF(Nota, "input", 5, MF4.5)
Nota = addMF(Nota, "input", 5, MF5.5)
Nota = addMF(Nota, "input", 5, MF6.5)

D6 # identificando os parâmetros das funções de pertinência da Disciplina 6
Nota = addVar(Nota, "input", "Disc.6", 0:10)
MF1.6 = triMF("FCN", 0:10, c(0,0,4.9,1))
MF2.6 = triMF("MAbM", 0:10, c(0,4.9,7,1))
MF3.6 = triMF("PAbM", 0:10, c(4.9,7,7.95,1))
MF4.6 = triMF("Md", 0:10, c(7,7.95,8.6,1))
MF5.6 = triMF("PAcM", 0:10, c(7.95,8.6,9.9,1))
MF6.6 = triMF("MAcM", 0:10, c(8.6,9.9,10,1))
MF7.6 = triMF("FCP", 0:10, c(9.9,10,10,1))
Nota = addMF(Nota, "input", 6, MF1.6)
Nota = addMF(Nota, "input", 6, MF2.6)
Nota = addMF(Nota, "input", 6, MF3.6)
Nota = addMF(Nota, "input", 6, MF4.6)
Nota = addMF(Nota, "input", 6, MF5.6)
Nota = addMF(Nota, "input", 6, MF6.6)
Nota = addMF(Nota, "input", 6, MF7.6)

D7 # identificando os parâmetros das funções de pertinência da Disciplina 7
Nota = addVar(Nota, "input", "Disc.7", 0:10)
MF1.7 = triMF("FCN", 0:10, c(0,0,0.5,1))
MF2.7 = triMF("MAbM", 0:10, c(0,0.5,3,1))
MF3.7 = triMF("PAbM", 0:10, c(0.5,3,6,1))
MF4.7 = triMF("Md", 0:10, c(3,6,7,1))
MF5.7 = triMF("PAcM", 0:10, c(6,7,9,1))
MF6.7 = triMF("MAcM", 0:10, c(7,9,10,1))
MF7.7 = triMF("FCP", 0:10, c(9,10,10,1))
Nota = addMF(Nota, "input", 7, MF1.7)
Nota = addMF(Nota, "input", 7, MF2.7)
Nota = addMF(Nota, "input", 7, MF3.7)
Nota = addMF(Nota, "input", 7, MF4.7)
```

```
Nota = addMF(Nota, "input", 7, MF5.7)
Nota = addMF(Nota, "input", 7, MF6.7)
Nota = addMF(Nota, "input", 7, MF7.7)

D8 # identificando os parâmetros das funções de pertinência da Disciplina 8
Nota = addVar(Nota, "input", "Disc.8", 0:10)
MF1.8 = triMF("FCN", 0:10, c(0,0,0.3,1))
MF2.8 = triMF("MAbM", 0:10, c(0,0.3,2.2,1))
MF3.8 = triMF("PAbM", 0:10, c(0.3,2.2,5.05,1))
MF4.8 = triMF("Md", 0:10, c(2.2,5.05,6.8,1))
MF5.8 = triMF("PAcM", 0:10, c(5.05,6.8,9,1))
MF6.8 = triMF("MAcM", 0:10, c(6.8,9,10,1))
MF7.8 = triMF("FCP", 0:10, c(9,10,10,1))
Nota = addMF(Nota, "input", 8, MF1.8)
Nota = addMF(Nota, "input", 8, MF2.8)
Nota = addMF(Nota, "input", 8, MF3.8)
Nota = addMF(Nota, "input", 8, MF4.8)
Nota = addMF(Nota, "input", 8, MF5.8)
Nota = addMF(Nota, "input", 8, MF6.8)
Nota = addMF(Nota, "input", 8, MF7.8)

# Resultados com parâmetros distribuídos por equidade
Nota = addVar(Nota, "output", "Result", 0:10)
MF1.9 = triMF("I", 0:10, c(0,0,1.67,1))
MF2.9 = triMF("Ru", 0:10, c(0,1.67,3.34,1))
MF3.9 = triMF("Rg", 0:10, c(1.67,3.34,5,1))
MF4.9 = triMF("B", 0:10, c(3.34,5,6.67,1))
MF5.9 = triMF("MB", 0:10, c(5,6.67,8.33,1))
MF6.9 = triMF("O", 0:10, c(6.67,8.33,10,1))
MF7.9 = triMF("E", 0:10, c(8.33,10,10,1))
Nota = addMF(Nota, "output", 1, MF1.9)
Nota = addMF(Nota, "output", 1, MF2.9)
Nota = addMF(Nota, "output", 1, MF3.9)
Nota = addMF(Nota, "output", 1, MF4.9)
Nota = addMF(Nota, "output", 1, MF5.9)
Nota = addMF(Nota, "output", 1, MF6.9)
Nota = addMF(Nota, "output", 1, MF7.9)

### Criando regras de inferência com base no método Combs

Nota = addRule(Nota, c(1,0,0,0,0,0,0,0,1,1,2))
Nota = addRule(Nota, c(0,1,0,0,0,0,0,0,1,1,2))
Nota = addRule(Nota, c(0,0,1,0,0,0,0,0,1,1,2))
Nota = addRule(Nota, c(0,0,0,1,0,0,0,0,1,1,2))
Nota = addRule(Nota, c(0,0,0,0,1,0,0,0,1,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,1,0,0,1,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,1,0,1,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,1,1,1,2))

Nota = addRule(Nota, c(2,0,0,0,0,0,0,0,2,1,2))
Nota = addRule(Nota, c(0,2,0,0,0,0,0,0,2,1,2))
Nota = addRule(Nota, c(0,0,2,0,0,0,0,0,2,1,2))
Nota = addRule(Nota, c(0,0,0,2,0,0,0,0,2,1,2))
Nota = addRule(Nota, c(0,0,0,0,2,0,0,0,2,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,2,0,0,2,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,2,0,2,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,2,2,1,2))

Nota = addRule(Nota, c(3,0,0,0,0,0,0,0,3,1,2))
Nota = addRule(Nota, c(0,3,0,0,0,0,0,0,3,1,2))
Nota = addRule(Nota, c(0,0,3,0,0,0,0,0,3,1,2))
Nota = addRule(Nota, c(0,0,0,3,0,0,0,0,3,1,2))
Nota = addRule(Nota, c(0,0,0,0,3,0,0,0,3,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,3,0,0,3,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,3,0,3,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,3,3,1,2))

Nota = addRule(Nota, c(4,0,0,0,0,0,0,0,4,1,2))
Nota = addRule(Nota, c(0,4,0,0,0,0,0,0,4,1,2))
Nota = addRule(Nota, c(0,0,4,0,0,0,0,0,4,1,2))
Nota = addRule(Nota, c(0,0,0,4,0,0,0,0,4,1,2))
Nota = addRule(Nota, c(0,0,0,0,4,0,0,0,4,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,4,0,0,4,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,4,0,4,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,4,4,1,2))

Nota = addRule(Nota, c(5,0,0,0,0,0,0,0,5,1,2))
Nota = addRule(Nota, c(0,5,0,0,0,0,0,0,5,1,2))
Nota = addRule(Nota, c(0,0,5,0,0,0,0,0,5,1,2))
Nota = addRule(Nota, c(0,0,0,5,0,0,0,0,5,1,2))
Nota = addRule(Nota, c(0,0,0,0,5,0,0,0,5,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,5,0,0,5,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,5,0,5,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,5,5,1,2))

Nota = addRule(Nota, c(6,0,0,0,0,0,0,0,6,1,2))
Nota = addRule(Nota, c(0,6,0,0,0,0,0,0,6,1,2))
Nota = addRule(Nota, c(0,0,6,0,0,0,0,0,6,1,2))
Nota = addRule(Nota, c(0,0,0,6,0,0,0,0,6,1,2))
Nota = addRule(Nota, c(0,0,0,0,6,0,0,0,6,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,6,0,0,6,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,6,0,6,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,6,6,1,2))

Nota = addRule(Nota, c(7,0,0,0,0,0,0,0,7,1,2))
Nota = addRule(Nota, c(0,7,0,0,0,0,0,0,7,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,7,0,0,7,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,7,0,7,1,2))
Nota = addRule(Nota, c(0,0,0,0,0,0,0,0,7,7,1,2))
# Inputs 3, 4 e 5 não possuem conj. fuzzy "7", pois as notas são altas

# Cálculo dos Resultados Fuzzy
Result.f = matrix(nrow=54, ncol=1)
Result.m = matrix(nrow=54, ncol=1)

i = 1
repeat {
   Aluno = as.numeric(dados[i,]) # t1 agora é um vetor de dados
   fuzzy = evalFIS(Aluno, Nota, numPoints = 101)
   mean = mean(Aluno)
   Result.f[i,] = as.numeric(fuzzy)
   Result.m[i,] = mean

   i = i+1
   if (i > 54) break # número limite de alunos
}
Rank.fuzzy = round(rank(-Result.f),0)
Rank.media = round(rank(-Result.m),0)
Resultados = cbind (Result.f,Rank.fuzzy,Result.m,Rank.media)
colnames(Resultados) = c("Result Fuzzy","Rank","Result Médias","Rank")
Resultados

# Transporte dos resultados para planilha MS Excel
write.xlsx(Resultados,'C:\\Users\\Luiz Gaviao\\1ELFA\\Resultados.xlsx')
```