

UNIVERSIDADE DE SÃO PAULO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
ESCOLA DE ENGENHARIA DE SÃO CARLOS

SEL5755 - Sistemas Fuzzy

Prof Dr. Ivan Nunes da Silva

EPC 3

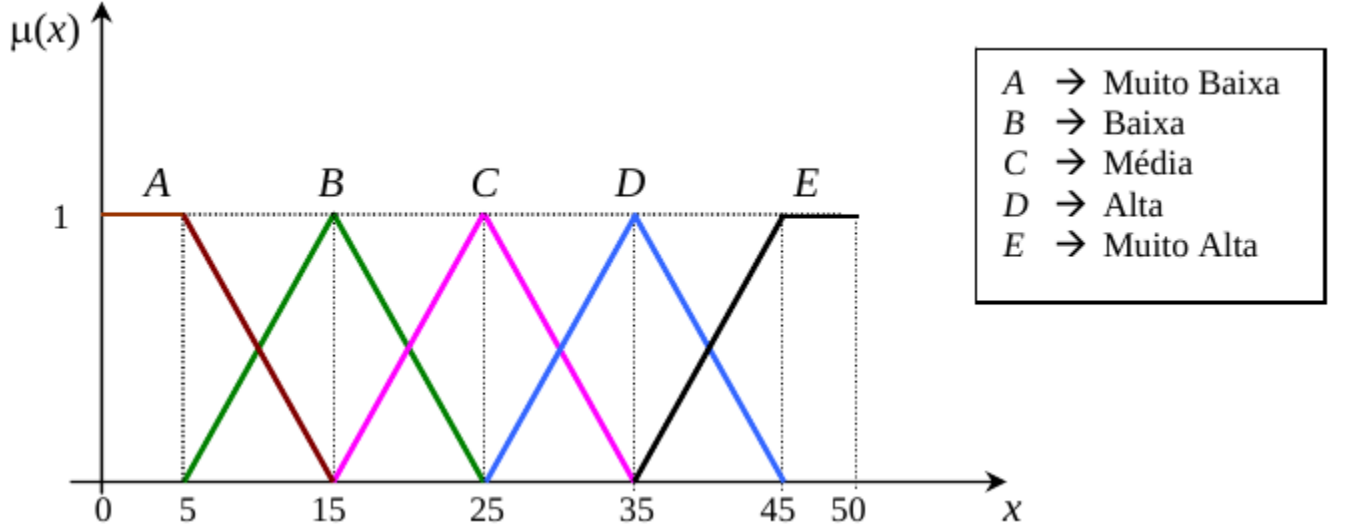
Alunos:

Isabela R. do Prado ROSSALES
6445435

Jonas Rossi DOURADO
6445442

São Carlos,
10 de setembro de 2012

- 1 Considere a função de pertinência abaixo a qual está descrevendo 5 conjuntos *fuzzy* que representam a temperatura de um processo industrial.



- a) Encontre as expressões analíticas referentes a cada um dos conjuntos *fuzzy*.

$$\mu_A(x) = \begin{cases} 1 & \text{se } 0 \leq x \leq 5 \\ -0,1x + 1,5, & \text{se } 5 < x \leq 15 \\ 0, & \text{caso contrário} \end{cases}$$

$$\mu_B(x) = \begin{cases} 0,1x - 0,5, & \text{se } 5 \leq x \leq 15 \\ -0,1x + 2,5, & \text{se } 15 < x \leq 25 \\ 0, & \text{caso contrário} \end{cases}$$

$$\mu_C(x) = \begin{cases} 0,1x - 1,5, & \text{se } 15 \leq x \leq 25 \\ -0,1x + 3,5, & \text{se } 25 < x \leq 35 \\ 0, & \text{caso contrário} \end{cases}$$

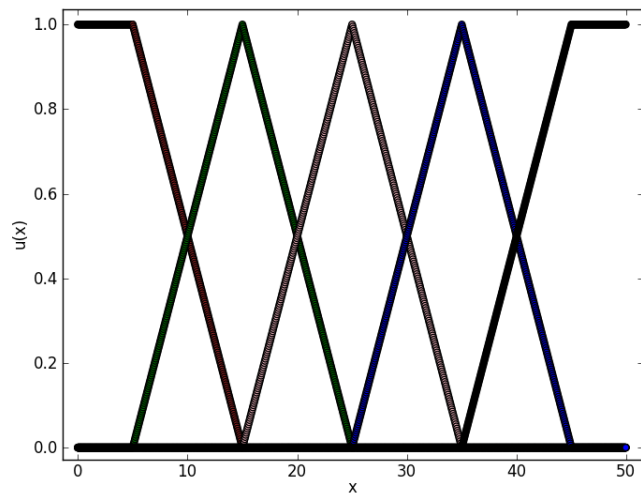
$$\mu_D(x) = \begin{cases} 0,1x - 2,5, & \text{se } 25 \leq x \leq 35 \\ -0,1x + 4,5, & \text{se } 35 < x \leq 45 \\ 0, & \text{caso contrário} \end{cases}$$

$$\mu_E(x) = \begin{cases} 0,1x - 3,5, & \text{se } 35 \leq x \leq 45 \\ 1 & \text{se } 45 < x \leq 50 \\ 0, & \text{caso contrário} \end{cases}$$

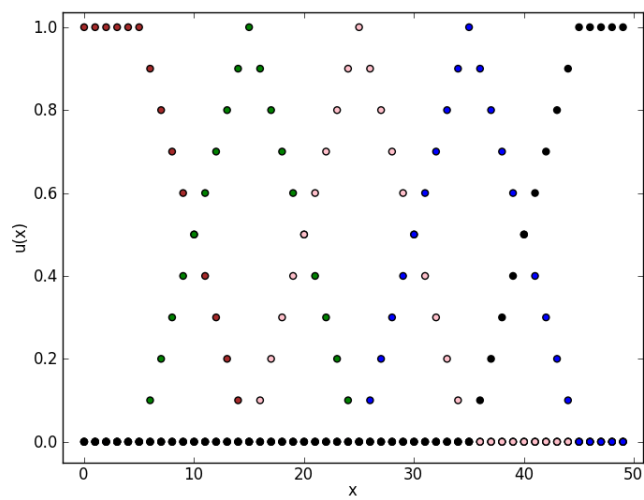
- b) Elabore os procedimentos computacionais que permitam mapear os conjuntos *fuzzy* acima, utilizando para tanto 1000 pontos de discretização. Sugestão: Utilize Arrays.
- Para checar o nível de generalização do seu sistema, verifique se haverá necessidade de modificações acentuadas quando passarmos a trabalhar com 500 ou 2000 pontos de discretização.
- c) Elabore os procedimentos computacionais que, dado um valor qualquer de x , permitam indicar quais dos conjuntos *fuzzy* acima estarão ativos, ou sejam, aqueles que possuem $\mu(x) \neq 0$.

- d) Elabore os procedimentos computacionais que, dado um conjunto *fuzzy* que está ativo, retorne o respectivo valor do grau de pertinência em relação a um valor de temperatura x pertencente ao universo de discurso.
 - e) Elabore os procedimentos computacionais que dado um conjunto *fuzzy*, bem como um valor de grau de pertinência, retorne então o respectivo conjunto crisp representando o α – corte efetuado.
- 2 Baseado nos procedimentos computacionais realizados acima faça:
- a) Imprima numa mesma folha os gráficos (conforme a figura anterior) dos cinco conjuntos *fuzzy* quando utilizamos 50 e 1000 pontos de discretização para representá-los, explicando ainda a importância de se especificar corretamente este parâmetro.
É importante não aumentar excessivamente o número de pontos de discretização a fim de evitar excesso de uso dos recursos de armazenamento disponíveis. Entretanto, caso o número de pontos de discretização seja baixo, pode haver um aumento imprecisão dos valores obtidos.
 - b) Imprima o conjunto *fuzzy* resultante da União dos cinco conjuntos *fuzzy* definidos acima, utilizando para tanto 1000 pontos de discretização e o operador Máximo.
 - c) Imprima o conjunto *fuzzy* resultante da Interseção dos cinco conjuntos *fuzzy* definidos acima, utilizando para tanto 500 pontos de discretização e o operador Mínimo.
 - d) Imprima o conjunto *fuzzy* resultante da operação de Complemento efetuado sobre o conjunto *fuzzy* C.
- 3 Baseado nos procedimentos computacionais realizados no primeiro e segundo exercício, considerando-se ainda apenas os conjuntos *fuzzy* ativos para uma determinada temperatura, faça os seguintes gráficos:
- a) Imprima o conjunto *fuzzy* resultante da União dos conjuntos *fuzzy* ativos em $x = 16,75$.
 - b) Imprima o conjunto *fuzzy* resultante da União dos conjuntos *fuzzy* ativos em $x = 37,29$.
 - c) Imprima o conjunto *fuzzy* resultante da Interseção dos conjuntos *fuzzy* ativos em $x = 20$.
 - d) Imprima o conjunto *fuzzy* resultante da Interseção dos conjuntos *fuzzy* ativos em $x = 40$.
- 4 Refaça o exercício anterior adotando os operadores Soma Algébrica (União) e Produto Algébrico (Interseção).
- 5 Baseado nos procedimentos computacionais anteriores, imprima então os gráficos resultantes das seguintes operações:
- a) $A \cup B \cup C$
 - b) $B \cap (C \cup D)$
 - c) $(A \cap B) \cup (B \cap C)$
 - d) $\overline{A} \cup (B \cap C) \cup \overline{D}$

Ex 2a.

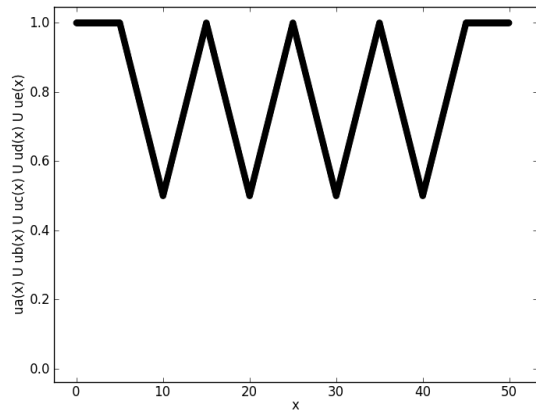


(a) 1000 pontos de discretização

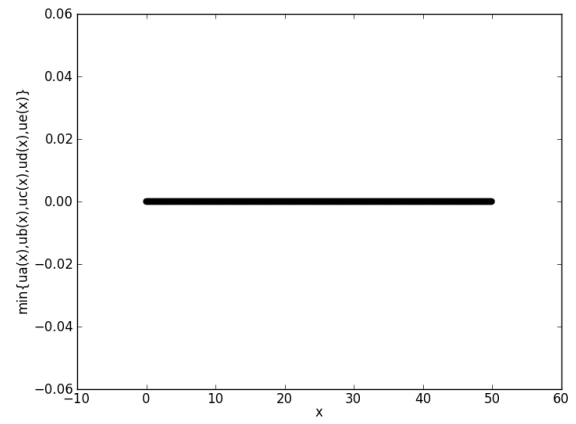


(b) 1000 pontos de discretização

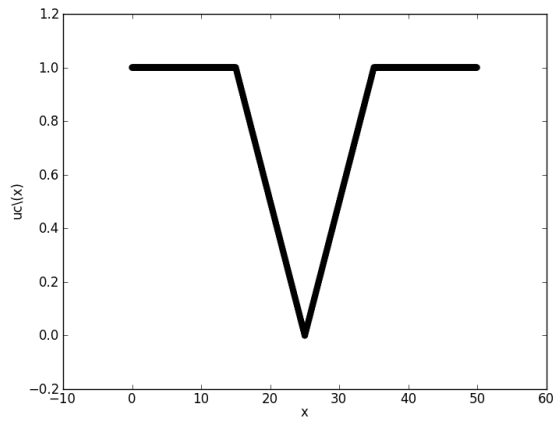
Ex 2 b,c,d



(c) 2b

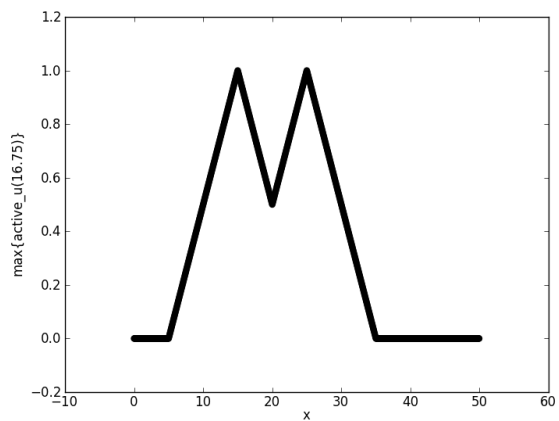


(d) 2c

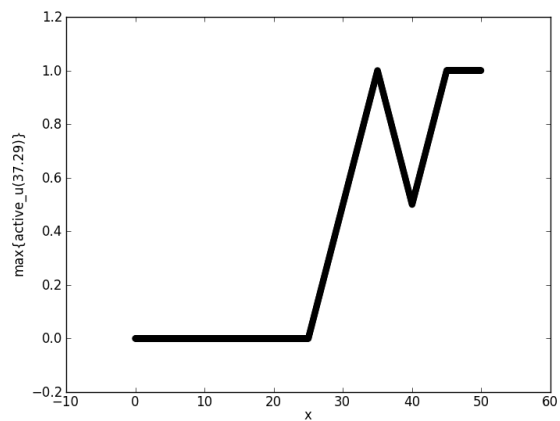


(e) 2d

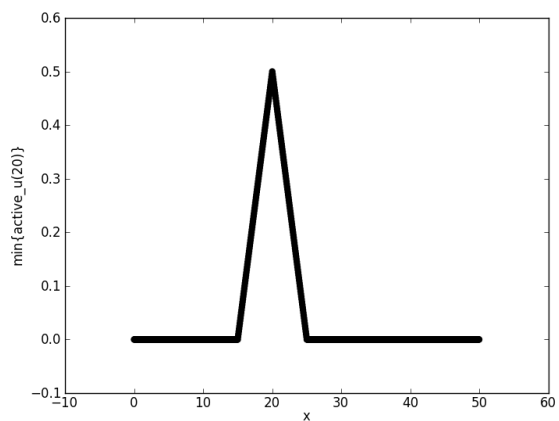
Ex 3



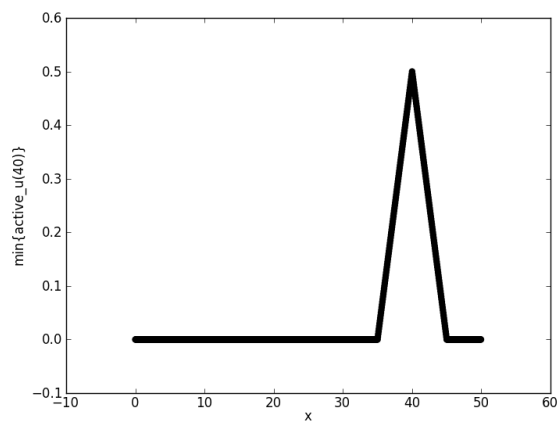
(f) 3a



(g) 3b

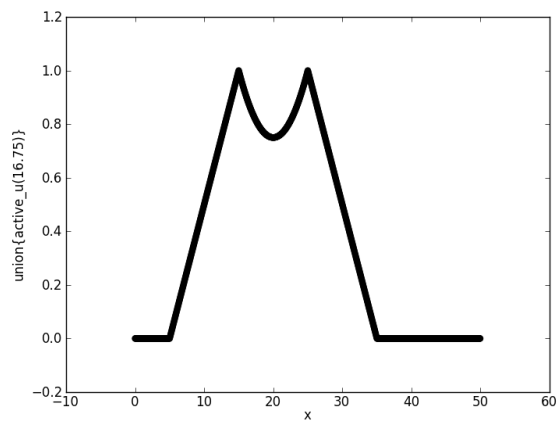


(h) 3c

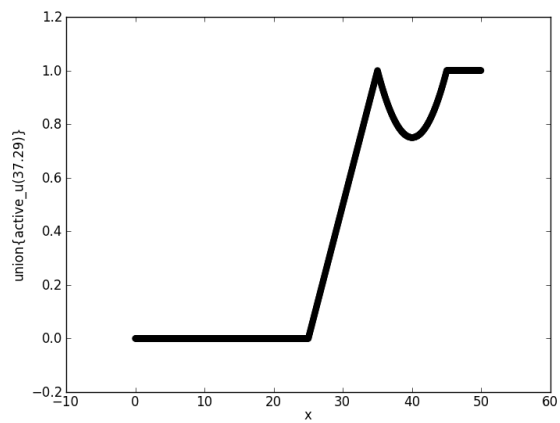


(i) 3d

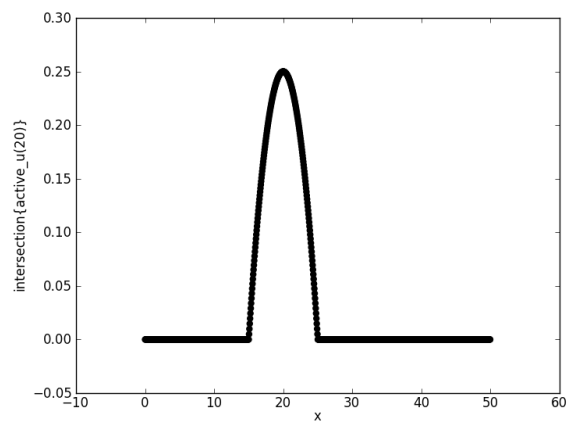
Ex 4



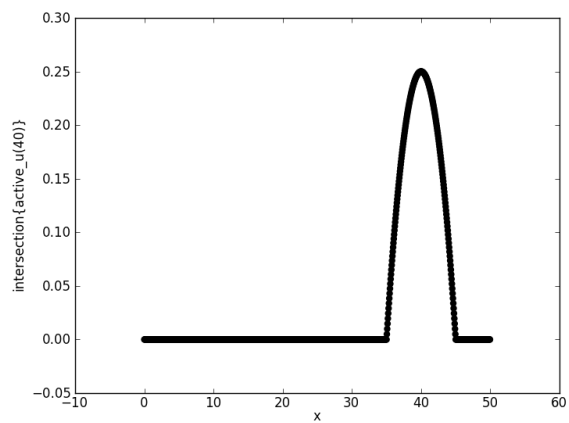
(j) 4a



(k) 4b

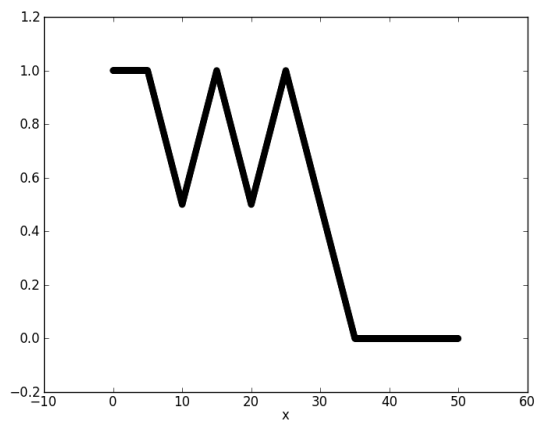


(l) 4c

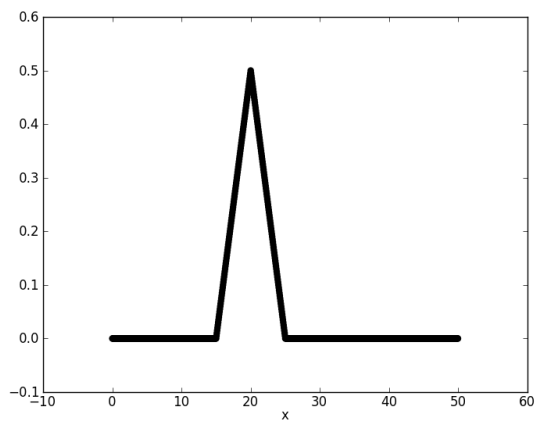


(m) 4d

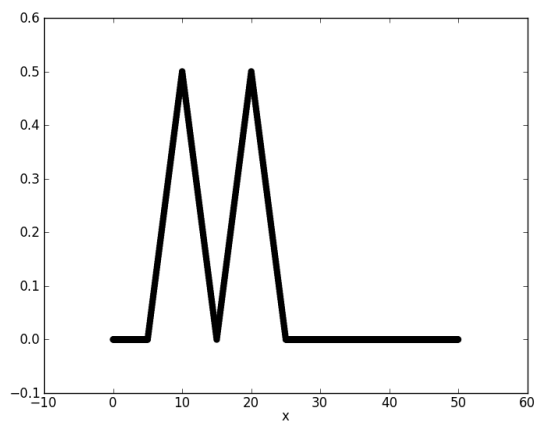
Ex 5



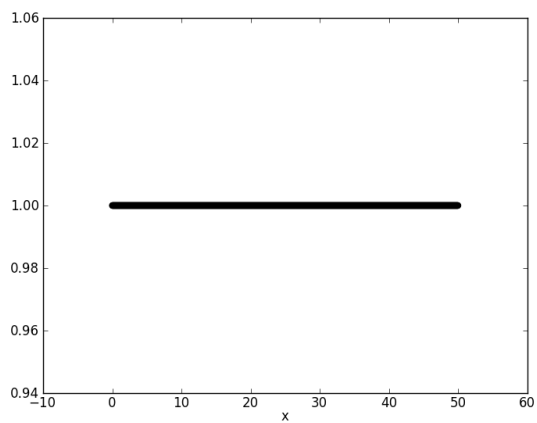
(n) 5a



(o) 5b



(p) 5c



(q) 5d


```

import matplotlib
import matplotlib.pyplot
import numpy

N_POINTS = 1000

# Exercise 1b
def ua(x):
    if 0<=x and x <= 5:
        return 1
    elif 5<x and x<=15:
        return -0.1*x+1.5
    return 0

# Exercise 1b
def ub(x):
    if 5<=x and x<=15:
        return 0.1*x-0.5
    elif 15<x and x<=25:
        return -0.1*x+2.5
    return 0

# Exercise 1b
def uc(x):
    if 15<=x and x<=25:
        return 0.1*x-1.5
    elif 25<x and x<=35:
        return -0.1*x+3.5
    return 0

# Exercise 1b
def ud(x):
    if 25<=x and x<=35:
        return 0.1*x-2.5
    elif 35<x and x<=45:
        return -0.1*x+4.5
    return 0

# Exercise 1b
def ue(x):
    if 35<=x and x<=45:
        return 0.1*x-3.5
    elif 45<x and x<=50:
        return 1
    return 0

# Exercise 1b
def plot_u():
    xa = numpy.arange(0,50,50.0/N_POINTS)
    ya = map(ua, xa)
    xb = numpy.arange(0,50,50.0/N_POINTS)

```

```

yb = map(ub, xb)
xc = numpy.arange(0,50,50.0/N_POINTS)
yc = map(uc, xc)
xd = numpy.arange(0,50,50.0/N_POINTS)
yd = map(ud, xd)
xe = numpy.arange(0,50,50.0/N_POINTS)
ye = map(ue, xe)
matplotlib.pyplot.scatter(xa, ya, c = "brown", marker = 'o')
matplotlib.pyplot.scatter(xb, yb, c = "green", marker = 'o')
matplotlib.pyplot.scatter(xc, yc, c = "pink", marker = 'o')
matplotlib.pyplot.scatter(xd, yd, c = "blue", marker = 'o')
matplotlib.pyplot.scatter(xe, ye, c = "black", marker = 'o')
matplotlib.pyplot.xlabel('x')
matplotlib.pyplot.ylabel('u(x)')
matplotlib.pyplot.show()

# Exercise 1c
def active_u(x):
    active = []
    if ua(x) > 0.0:
        print 'A',
        active.append('A')
    if ub(x) > 0.0:
        print 'B',
        active.append('B')
    if uc(x) > 0.0:
        print 'C',
        active.append('C')
    if ud(x) > 0.0:
        print 'D',
        active.append('D')
    if ue(x) > 0.0:
        print 'E',
        active.append('E')
    return active

# Exercise 1d
def u(active, x):
    if active == 'A':
        return ua(x)
    elif active == 'B':
        return ub(x)
    elif active == 'C':
        return uc(x)
    elif active == 'D':
        return ud(x)
    elif active == 'E':
        return ue(x)

# Exercise 1e
def crisp(active, value):

```

```

x = numpy.arange(0,50,50.0/N_POINTS)

if active == 'A':
    return filter (lambda a: ua(a) >= value, x)
elif active == 'B':
    return filter (lambda a: ub(a) >= value, x)
elif active == 'C':
    return filter (lambda a: uc(a) >= value, x)
elif active == 'D':
    return filter (lambda a: ud(a) >= value, x)
elif active == 'E':
    return filter (lambda a: ue(a) >= value, x)

def ex2b():
x = numpy.arange(0,50,50.0/N_POINTS)
ya = map(ua, x)
yb = map(ub, x)
yc = map(uc, x)
yd = map(ud, x)
ye = map(ue, x)

matplotlib.pyplot.scatter(x, map(lambda a,b,c,d,e:
    max([a,b,c,d,e]), ya, yb, yc, yd, ye), c = "black", marker = 'o')
matplotlib.pyplot.xlabel('x')
matplotlib.pyplot.ylabel('ua(x)∪ub(x)∪uc(x)∪ud(x)∪ue(x)')
matplotlib.pyplot.show()

def ex2c():
x = numpy.arange(0,50,50.0/N_POINTS)
ya = map(ua, x)
yb = map(ub, x)
yc = map(uc, x)
yd = map(ud, x)
ye = map(ue, x)

matplotlib.pyplot.scatter(x, map(lambda a,b,c,d,e:
    min([a,b,c,d,e]), ya, yb, yc, yd, ye), c = "black", marker = 'o')
matplotlib.pyplot.xlabel('x')
matplotlib.pyplot.ylabel('min{ua(x),ub(x),uc(x),ud(x),ue(x)}')
matplotlib.pyplot.show()

def ex2d():
x = numpy.arange(0,50,50.0/N_POINTS)
yc = map(lambda a: 1-uc(a), x)

matplotlib.pyplot.scatter(x, yc, c = "black", marker = 'o')
matplotlib.pyplot.xlabel('x')
matplotlib.pyplot.ylabel('uc\ (x)')
matplotlib.pyplot.show()

def ex3a():

```

```

x = numpy.arange(0,50,50.0/N_POINTS)
union = [0]*N_POINTS

for i in active_u(16.75):
    union = map(lambda q,w: max([q,w]), union,
                map(lambda a: u(i, a), x))

matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
matplotlib.pyplot.xlabel('x')
matplotlib.pyplot.ylabel('max{active_u(16.75)}')
matplotlib.pyplot.show()

def ex3b():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [0]*N_POINTS

    for i in active_u(37.29):
        union = map(lambda q,w: max([q,w]), union,
                    map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('max{active_u(37.29)}')
    matplotlib.pyplot.show()

def ex3c():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [1]*N_POINTS

    for i in active_u(20):
        union = map(lambda q,w: min([q,w]), union,
                    map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('min{active_u(20)}')
    matplotlib.pyplot.show()

def ex3d():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [1]*N_POINTS

    for i in active_u(40):
        union = map(lambda q,w: min([q,w]), union,
                    map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('min{active_u(40)}')
    matplotlib.pyplot.show()

```

```

def ex4a():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [0]*N_POINTS

    for i in active_u(16.75):
        union = map(lambda q,w: q+w-q*w, union,
                    map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('union{active_u(16.75)}')
    matplotlib.pyplot.show()

def ex4b():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [0]*N_POINTS

    for i in active_u(37.29):
        union = map(lambda q,w: q+w-q*w, union, map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('union{active_u(37.29)}')
    matplotlib.pyplot.show()

def ex4c():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [1]*N_POINTS

    for i in active_u(20):
        union = map(lambda q,w: q*w, union, map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('intersection{active_u(20)}')
    matplotlib.pyplot.show()

def ex4d():
    x = numpy.arange(0,50,50.0/N_POINTS)
    union = [1]*N_POINTS

    for i in active_u(40):
        union = map(lambda q,w: q*w, union, map(lambda a: u(i, a), x))

    matplotlib.pyplot.scatter(x, union, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.ylabel('intersection{active_u(40)}')
    matplotlib.pyplot.show()

def ex5a():
    x = numpy.arange(0,50,50.0/N_POINTS)

```

```

res = map(lambda q,w,e: max([q,w,e]), map(ua, x),map(ub, x),map(uc, x))

matplotlib.pyplot.scatter(x, res, c = "black", marker = 'o')
matplotlib.pyplot.xlabel('x')
matplotlib.pyplot.show()

def ex5b():
    x = numpy.arange(0,50,50.0/N_POINTS)

    res = map(lambda q,w,e: min([q, max([w,e])]),
              map(ub, x),map(uc, x),map(ud, x))

    matplotlib.pyplot.scatter(x, res, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.show()

def ex5c():
    x = numpy.arange(0,50,50.0/N_POINTS)

    res = map(lambda q,w,e: max([min([q,w]), min([w,e])]),
              map(ua, x),map(ub, x),map(uc, x))

    matplotlib.pyplot.scatter(x, res, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.show()

def ex5d():
    x = numpy.arange(0,50,50.0/N_POINTS)

    res = map(lambda q,w,e,r: max([1-q, min([w,e]), 1-r]),
              map(ua, x),map(ub, x),map(uc, x), map(ud,x))

    matplotlib.pyplot.scatter(x, res, c = "black", marker = 'o')
    matplotlib.pyplot.xlabel('x')
    matplotlib.pyplot.show()

if __name__ == "__main__":
    ex5d()

```