

Курсов проект по
Размити множества

Тема: Размити числа
/принцип на разширението/

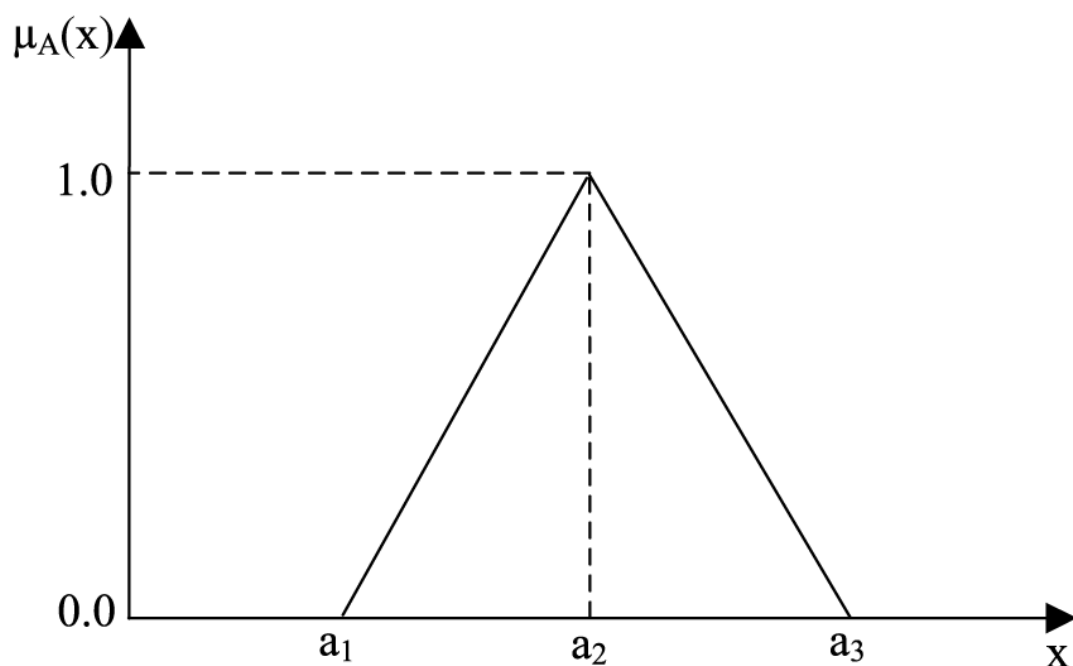
Валентина Динкова, ф.н. 71112, 2 група

27 януари 2010 г.

1 Размити числа

Размитите числа са изпъкнали, нормализирани размити множества, чието функцията на принадлежност е дефинирана в R и е частично непрекъсната. Размитите числа представят интервал от реални числа, чиито граници са размити. Най-популярната им форма е триъгълната. Триъгълните размити числа се представят чрез 3 точки:

$$\tilde{A} = (a_1, a_2, a_3)$$



$$\mu_A(x) = \begin{cases} 0, & x < a_1 \\ \frac{x-a_1}{a_2-a_1}, & a_1 \leq x \leq a_2 \\ \frac{a_3-x}{a_3-a_2}, & a_2 \leq x \leq a_3 \\ 0, & x > a_3 \end{cases}$$

1.1 Аритметични операции - принцип на разширението

Нека имаме две размити числа $\tilde{A} = \varphi(a_1, b_1, c_1)$ и $\tilde{B} = \varphi(a_2, b_2, c_2)$.

Тогава:

$$\tilde{A} + \tilde{B} = \varphi(a_1 + a_2, b_1 + b_2, c_1 + c_2)$$

$$\tilde{A} - \tilde{B} = \varphi(a_1 - a_2, b_1 - b_2, c_1 - c_2)$$

$$\tilde{A} \cdot \tilde{B} = \varphi(\min(a_1 a_2, a_1 c_2, c_1 a_2, c_1 c_2), b_1 b_2, \max(a_1 a_2, a_1 c_2, c_1 a_2, c_1 c_2))$$

$$\tilde{A} / \tilde{B} = \varphi(\min(a_1/a_2, a_1/c_2, c_1/a_2, c_1/c_2), b_1/b_2, \max(a_1/a_2, a_1/c_2, c_1/a_2, c_1/c_2))$$

Ето и алгоритъма реализиран чрез Python:

```
#!/usr/bin/python
import pylab
```

```
class FuzzyNumber:
```

```
    def __init__(self, left, peak, right):
        self.peak = peak
        self.left = left
        self.right = right
        self.all=[self.left, self.peak, self.right]
```

```
    def __add__(self, other):
        a = self.left + other.left
        b = self.peak + other.peak
        c = self.right + other.right
        return FuzzyNumber(a, b, c)
```

```
    def __sub__(self, other):
        a = self.left - other.left
        b = self.peak - other.peak
        c = self.peak - other.peak
        return FuzzyNumber(a, b, c)
```

```
    def __mul__(self, other):
        a = min(self.left * other.left, self.left * other.right,
                self.right * other.left, self.right * other.right)
```

```

    b = self.peak * other.peak
    c = max(self.left * other.left, self.left * other.right,
            self.right * other.left, self.right * other.right)
    return FuzzyNumber(a, b, c)

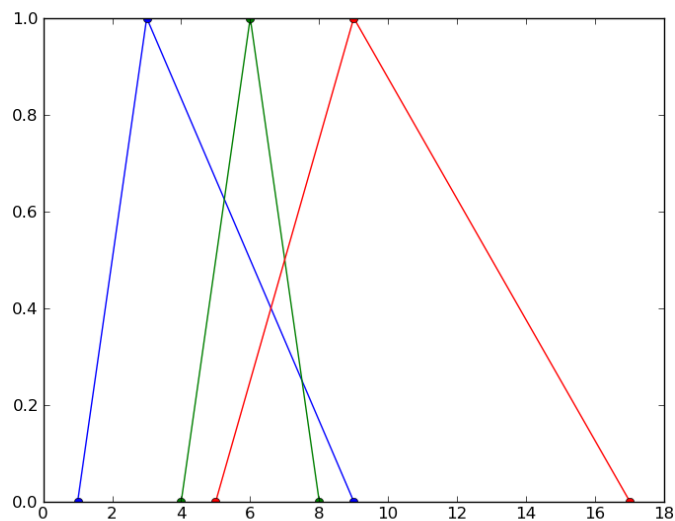
def __div__(self, other):
    if (other.left != 0 and other.peak != 0 and other.right != 0):
        a = min(self.left / other.left, self.left / other.right,
                self.right / other.left, self.right / other.right)
        b = self.peak / other.peak
        c = max(self.left / other.left, self.left / other.right,
                self.right / other.left, self.right / other.right)
        return FuzzyNumber(a, b, c)

def tuple(self):
    return tuple(self.all)

def __str__(self):
    return 'FuzzyNumber(' + str(self.left) + ', ' + str(self.peak) + ', ' + str(self.right) + ')'

```

На пример резултатът от събирането на размитите числа $a = \text{FuzzyNumber}(1,3,9)$ и $b = \text{FuzzyNumber}(4,6,8)$ е $\text{FuzzyNumber}(5, 9, 17)$. Изчертава се следната графика:



В синьо и зелено са съответно числата a и b , а в червено е резултатът от тяхното събиране. Изчертаването на графиката е реализирано с помощта на библиотеката `matplotlib` по следният начин:

```
#plotting

def add_number(r, fig, ax):
    n = r.tuple()
    DATA = ((0, n[0]),
             (1, n[1]),
             (0, n[2]))

    (y,x) = zip(*DATA)

    ax.plot(x, y, marker='o')
    for i in xrange(len(DATA)):
        (x,y) = DATA[i]

if(__name__=='__main__'):
    fig = pylab.figure()
    ax = fig.add_subplot(111)
    for n in [a,b, a+b]: #, a-b, a*b, a/b]:
        add_number(n,fig,ax)
    print (a + b)
    pylab.show()
```