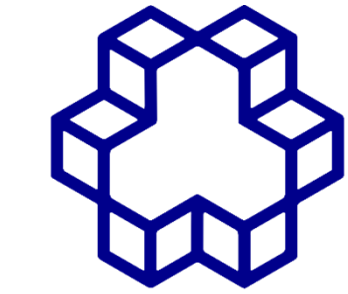
بسمه تعالی



دانشگاه صنعتی خواجه نصیرالدین طوسی

نام و شماره دانشجویی :

علی عبدی

9728463

عنوان :

مینی پروژه سوم

تشریح مساله و کدها

سوال اول

- ابتدا با غیرفازی ساز میانگین شروع میکنیم و کد کران مرتبه اول با توابع تعلق مثلثی بصورت زیر است:

```
import time
start_time = time.time()
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import warnings
warnings.filterwarnings('ignore')

alpha = -1
beta = 1
h = 0.05
N = 41

x1 = np.arange(alpha, beta, 0.01)
x2 = np.arange(alpha, beta, 0.01)
x1, x2 = np.meshgrid(x1, x2)

g_bar = np.zeros((N*N, 1))
e_i1 = np.zeros((N, 1))
e_i2 = np.zeros((N, 1))

num = 0
den = 0
k = 0

def trimf(x, abc):
    return np.fmax(np.fmin((x-abc[0])/(abc[1]-abc[0]), (abc[2]-x)/(abc[2]-
abc[1])), 0)

for i1 in range(1,N):
    for i2 in range(1,N):
        e_i1[i1-1,0] = -1 + h*(i1-1)
        e_i2[i2-1,0] = -1 + h*(i2-1)
        if i1==1:
            mu_A_x1 = trimf(x1, [-1,-1,-1+h])
```

```
        elif i1==N:
            mu_A_x1 = trimf(x1,[1-h, 1, 1])
        else:
            mu_A_x1 = trimf(x1,[-1+h*(i1-2), -1+h*(i1-1), -1+h*(i1)])

        if i2==1:
            mu_A_x2 = trimf(x2, [-1,-1,-1+h])
        elif i2==N:
            mu_A_x2 = trimf(x2,[1-h, 1, 1])
        else:
            mu_A_x2 = trimf(x2,[-1+h*(i2-2), -1+h*(i2-1), -1+h*(i2)])

        g_bar[k,0]= 1/(1+e_i1[i1, 0]**2+e_i2[i2, 0]**2)
        num = num + g_bar[k,0]*mu_A_x1*mu_A_x2
        den=den+mu_A_x1*mu_A_x2
        k=k+1

f_x = num/den
g_x = 1/(3+x1+x2)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

## For Error
# ax = fig.add_subplot(111, projection='3d')

E = g_x - f_x
surf = ax.plot_surface(x1, x2, E, cmap=cm.coolwarm,
                       linewidth=0, antialiased=False)
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$Error$')
ax.set_title('$Error$')
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.savefig('fuzzy1.svg')
plt.show()

# Print time of execution
print("--- %s seconds ---" % (time.time() - start_time))
```

- حال با کران مرتبه دوم کدش را مینویسیم:

```
import time
start_time = time.time()
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import warnings
warnings.filterwarnings('ignore')

alpha = -1
beta = 1
h = 0.25
N = 9

x1 = np.arange(alpha, beta, 0.01)
x2 = np.arange(alpha, beta, 0.01)
x1, x2 = np.meshgrid(x1, x2)

g_bar = np.zeros((N*N, 1))
e_i1 = np.zeros((N, 1))
e_i2 = np.zeros((N, 1))

num = 0
den = 0
k = 0

def trimf(x, abc):
    return np.fmax(np.fmin((x-abc[0])/(abc[1]-abc[0]), (abc[2]-x)/(abc[2]-
abc[1])), 0)

for i1 in range(1,N):
    for i2 in range(1,N):
        e_i1[i1-1,0] = -1 + h*(i1-1)
        e_i2[i2-1,0] = -1 + h*(i2-1)
        if i1==1:
            mu_A_x1 = trimf(x1, [-1,-1,-1+h])
        elif i1==N:
            mu_A_x1 = trimf(x1,[1-h, 1, 1])
        else:
            mu_A_x1 = trimf(x1,[-1+h*(i1-2), -1+h*(i1-1), -1+h*(i1)])

        if i2==1:
```

```
            mu_A_x2 = trimf(x2, [-1,-1,-1+h])
        elif i2==N:
            mu_A_x2 = trimf(x2,[1-h, 1, 1])
        else:
            mu_A_x2 = trimf(x2,[-1+h*(i2-2), -1+h*(i2-1), -1+h*(i2)])

        g_bar[k,0]= 1/(1+e_i1[i1, 0]**2+e_i2[i2, 0]**2)
        num = num + g_bar[k,0]*mu_A_x1*mu_A_x2
        den=den+mu_A_x1*mu_A_x2
        k=k+1

f_x = num/den
g_x = 1/(3+x1+x2)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

## For Error
# ax = fig.add_subplot(111, projection='3d')

E = g_x - f_x
surf = ax.plot_surface(x1, x2, E, cmap=cm.coolwarm,
                       linewidth=0, antialiased=False)
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$Error$')
ax.set_title('$Error$')
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.savefig('fuzzy3.svg')
plt.show()

# Print time of execution
print("--- %s seconds ---" % (time.time() - start_time))
```

- حال با غیرفازی ساز ماکزیمم میخواهیم سیستم فازی بسازیم و ابتدا کد کران مرتبه اول را مینویسیم:

```
import time
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import warnings
```

```
warnings.filterwarnings('ignore')

start_time = time.time()

alpha = -1
beta = 1
h = 0.05
N = 41

x1 = np.arange(alpha, beta, 0.01)
x2 = np.arange(alpha, beta, 0.01)
x1, x2 = np.meshgrid(x1, x2)

g_bar = np.zeros((N*N, 1))
e_i1 = np.zeros((N, 1))
e_i2 = np.zeros((N, 1))

num = 0
den = 0
k = 0

def trimf(x, abc):
    return np.fmax(np.fmin((x-abc[0])/(abc[1]-abc[0]), (abc[2]-x)/(abc[2]-
abc[1])), 0)

for i1 in range(1,N):
    for i2 in range(1,N):
        e_i1[i1-1,0] = -1 + h*(i1-1)
        e_i2[i2-1,0] = -1 + h*(i2-1)
        if i1==1:
            mu_A_x1 = trimf(x1, [-1,-1,-1+h])
        elif i1==N:
            mu_A_x1 = trimf(x1,[1-h, 1, 1])
        else:
            mu_A_x1 = trimf(x1,[-1+h*(i1-2), -1+h*(i1-1), -1+h*(i1)])

        if i2==1:
            mu_A_x2 = trimf(x2, [-1,-1,-1+h])
        elif i2==N:
            mu_A_x2 = trimf(x2,[1-h, 1, 1])
        else:
            mu_A_x2 = trimf(x2,[-1+h*(i2-2), -1+h*(i2-1), -1+h*(i2)])

        g_bar[k,0]= 1/(1+e_i1[i1, 0]**2+e_i2[i2, 0]**2)
        num = num + g_bar[k,0]*mu_A_x1*mu_A_x2
```

```
        den=den+mu_A_x1*mu_A_x2
        k=k+1

f_x = num/den
g_x = 1/(3+x1+x2)

# Calculate maximum-based fuzzy error
E_max = np.fmax(0, g_x - f_x)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

surf = ax.plot_surface(x1, x2, E_max, cmap=cm.coolwarm,
                       linewidth=0, antialiased=False)
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$Error_{\max}$')
ax.set_title('$Error_{\max}$')
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.savefig('fuzzy2.svg')
plt.show()

# Print time of execution
print("--- %s seconds ---" % (time.time() - start_time))
```

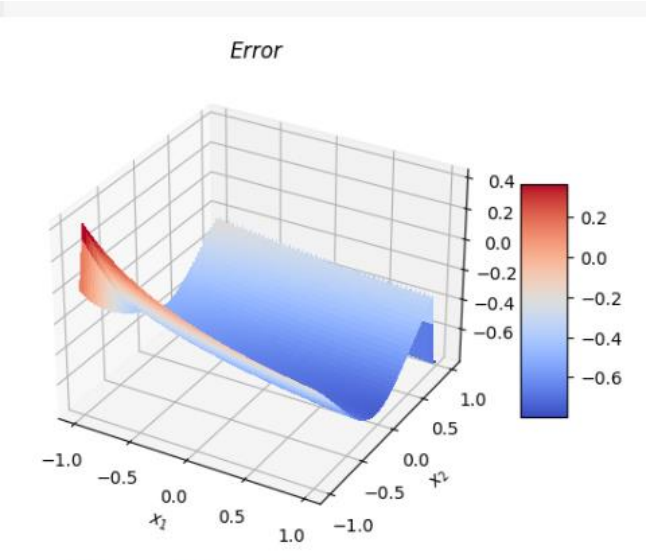
- حال کران مرتبه دوم:

```
import time
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import warnings
warnings.filterwarnings('ignore')

start_time = time.time()

alpha = -1
```

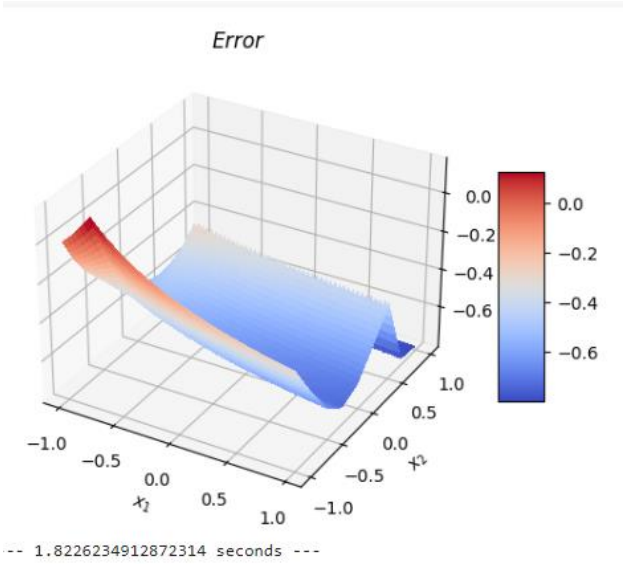
```
beta = 1
h = 0.25
N = 9

x1 = np.arange(alpha, beta, 0.01)
x2 = np.arange(alpha, beta, 0.01)
x1, x2 = np.meshgrid(x1, x2)

g_bar = np.zeros((N*N, 1))
e_i1 = np.zeros((N, 1))
e_i2 = np.zeros((N, 1))

num = 0
den = 0
k = 0

def trimf(x, abc):
    return np.fmax(np.fmin((x-abc[0])/(abc[1]-abc[0]), (abc[2]-x)/(abc[2]-
abc[1])), 0)

def maxf(x, abc):
    return np.fmax(np.fmax((x-abc[0])/(abc[1]-abc[0]), (abc[2]-x)/(abc[2]-
abc[1])), 0)

for i1 in range(1,N):
    for i2 in range(1,N):
        e_i1[i1-1,0] = -1 + h*(i1-1)
        e_i2[i2-1,0] = -1 + h*(i2-1)
        if i1==1:
            mu_A_x1 = trimf(x1, [-1,-1,-1+h])
        elif i1==N:
            mu_A_x1 = trimf(x1,[1-h, 1, 1])
        else:
            mu_A_x1 = trimf(x1,[-1+h*(i1-2), -1+h*(i1-1), -1+h*(i1)])

        if i2==1:
            mu_A_x2 = trimf(x2, [-1,-1,-1+h])
        elif i2==N:
            mu_A_x2 = trimf(x2,[1-h, 1, 1])
        else:
            mu_A_x2 = trimf(x2,[-1+h*(i2-2), -1+h*(i2-1), -1+h*(i2)])

        g_bar[k,0]= 1/(1+e_i1[i1, 0]**2+e_i2[i2, 0]**2)
        num = num + g_bar[k,0]*mu_A_x1*mu_A_x2
        den=den+mu_A_x1*mu_A_x2
```

```
        k=k+1

f_x = num/den
g_x = 1/(3+x1+x2)

# Calculate minimum-based fuzzy error
E_min = np.minimum(0, g_x - f_x)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

surf = ax.plot_surface(x1, x2, E_min, cmap=cm.coolwarm,
                       linewidth=0, antialiased=False)
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$Error_{\min}$')
ax.set_title('$Error_{\min}$')
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.savefig('non_fuzzy.svg')
plt.show()

# Print time of execution
print("--- %s seconds ---" % (time.time() - start_time))
```

- حالا نمودارهای خطاها را میکشیم به ترتیب:
- با غیرفازی ساز میانگین مراکز(سمت چپ کران مرتبه اول و سمت راست کران مرتبه دوم)

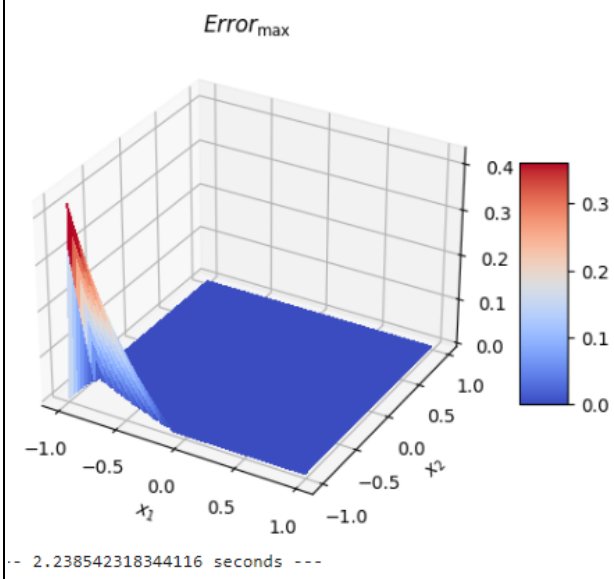


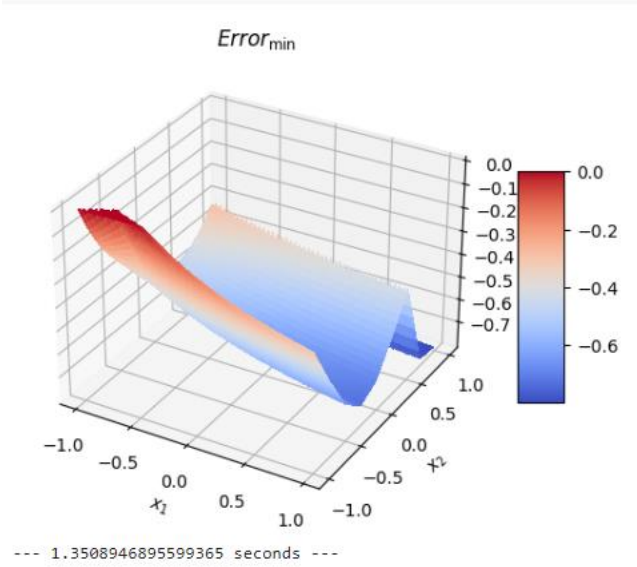
--- 2.1141653060913086 seconds ---



-- 1.8226234912872314 seconds ---

- با غیرفازی ساز ماکزیمم(سمت چپ کران مرتبه اول و سمت راست کران مرتبه دوم)





- کران مرتبه دوم در هر دو غیرفازی ساز فاصله مراکز را زیاد میکند درعین حال دقت مناسبی را با قواعد کمتر میدهد و در شکل های بالا گویا خطای غیرفازی ساز ماکزیمم در کران مرتبه دو کمی بیشتر است اما در کران مرتبه اول کمتر است نسبت به غیرفازی ساز میانگین مراکز .