به نام خدا

تمرین عملی نظریه بازی

پردیس زهرایی

۹۹۱۰۹۷۷۷

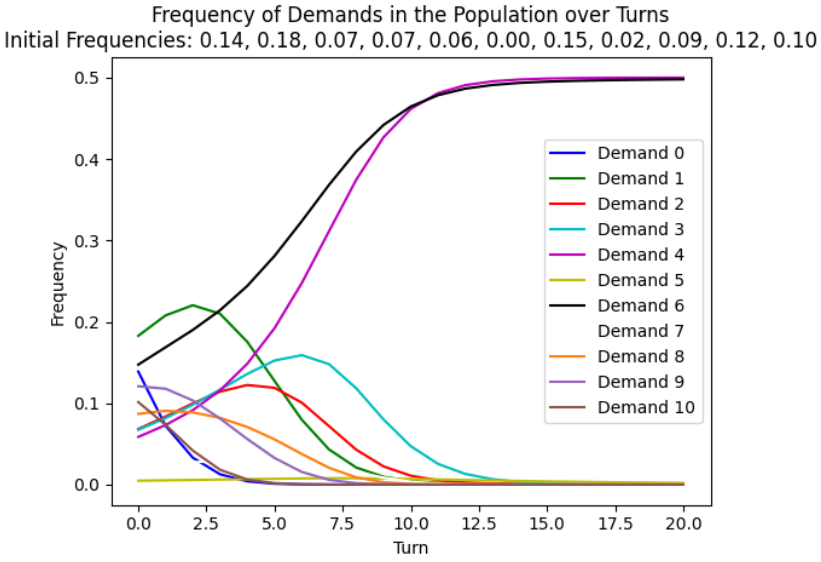
طبق کد پایتون قرار داده شده اگر یک شبیه ساز برای بازی تقسیم کیک با ۱۰ برش بنویسیم، سپس با تست کردن ورودی های مختلف (در اینجا ۱۰ نمونه آورده ولی بیشتر هم می تواند باشد) مشاهده شده که حالات پایدار متفاوت هستند.

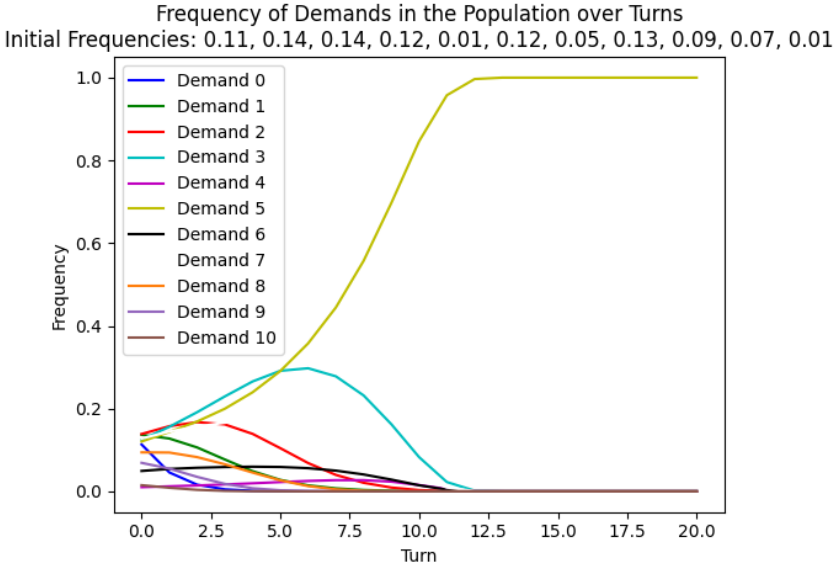
قطعه کد به این صورت است:

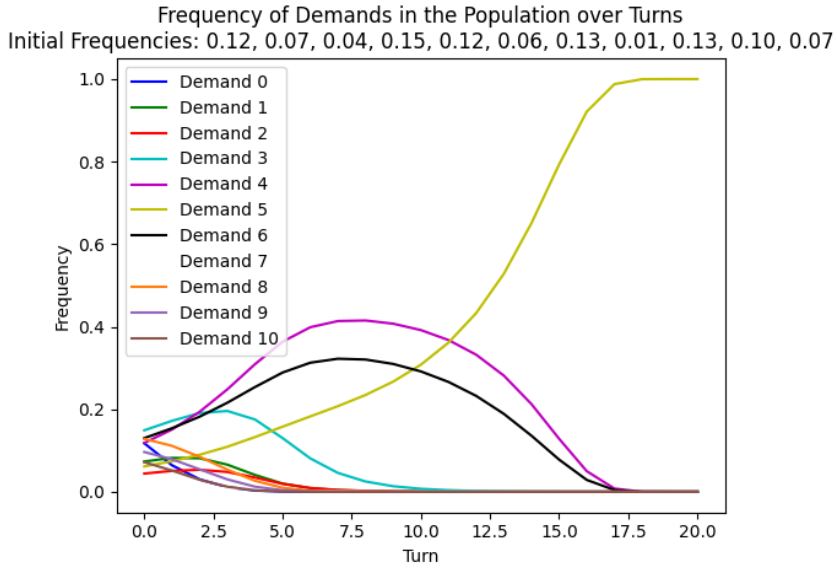
```
import numpy as np
import matplotlib.pyplot as plt
def replicator_dynamics(frequencies):
    """ Runs a round of the game where people get their
    demand if the sum of two players is less than or equal
    to 10. Otherwise they'll receive zero. Then each player
    chooses one person from the population at random and will
    adopt their strategy if they had a greater payoff than them.
    """
    # Calculate the payoffs
    payoffs = np.zeros((11, 11))
    for i in range(11):
        for j in range(11):
            if i + j <= 10:
                payoffs[i][j] = i
            else:
                payoffs[i][j] += 0

    # Calculate the fitnesses
    fitnesses = payoffs / np.sum(payoffs)

    # Calculate the new frequencies
    new_frequencies = np.zeros(11)
    for i1 in range(11):
        for j1 in range(11):
            for i2 in range(11):
                for j2 in range(11):
                    if payoffs[i1][j1] >= payoffs[i2][j2]:
                        new_frequencies[i1] += frequencies[i1] *
frequencies[j1] * frequencies[i2] * frequencies[j2]
                    else:
                        new_frequencies[i2] += frequencies[i1] *
frequencies[j1] * frequencies[i2] * frequencies[j2]

    return new_frequencies / np.sum(new_frequencies)
```

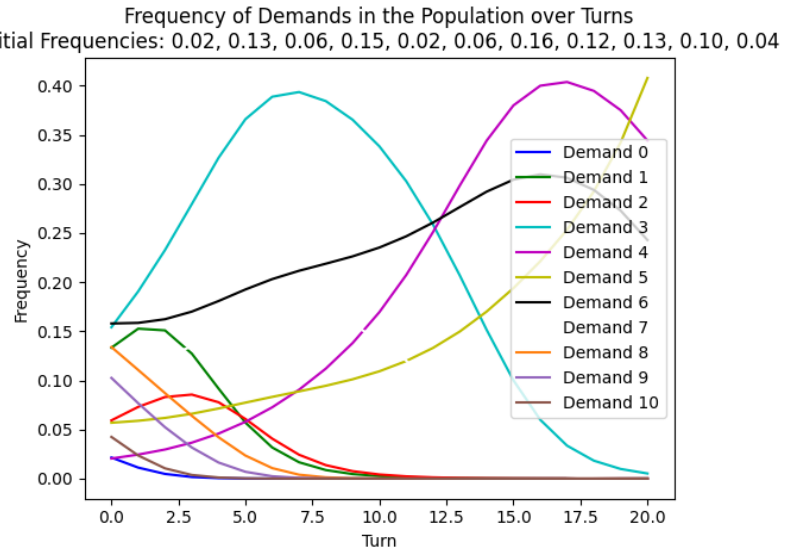
```
def plot_strategy_frequencies(frequencies_over_turns,
initial_frequencies,x):
    strategies = np.arange(11)
    turns = len(frequencies_over_turns)
    # Assign different color to each strategy
    colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'w',
'tab:orange', 'tab:purple', 'tab:brown']
    strategies_over_turns = np.array(frequencies_over_turns).T

    for i in range(len(strategies_over_turns)):
        plt.plot(np.arange(turns), strategies_over_turns[i],
color=colors[i], label=f"Demand {i}")

    plt.xlabel('Turn')
    plt.ylabel('Frequency')
    initial_frequencies_str = ', '.join([f'{freq:.2f}' for freq
in initial_frequencies])
    plt.title(f'Frequency of Demands in the Population over
Turns\nInitial Frequencies: {initial_frequencies_str}')
    plt.legend()
    plt.savefig(f'plot_{x + 1}.png')
    plt.show()

# Define the initial frequency distribution by randomly
assigning
# a frequency to each strategy
for x in range(10):
    print("round: ", x + 1)
    initial_frequencies = np.random.random(11)
    initial_frequencies /= np.sum(initial_frequencies)

    # Simulate the replicator dynamics over 10 turns
    turns = 20
    frequencies_over_turns = [initial_frequencies]

    for turn in range(turns):
        new_frequencies =
replicator_dynamics(frequencies_over_turns[-1])
        frequencies_over_turns.append(new_frequencies)

    plot_strategy_frequencies(frequencies_over_turns,initial_fre
quencies,x)
```

و با حالات اولیه متفاوت به این شکل در می آید:

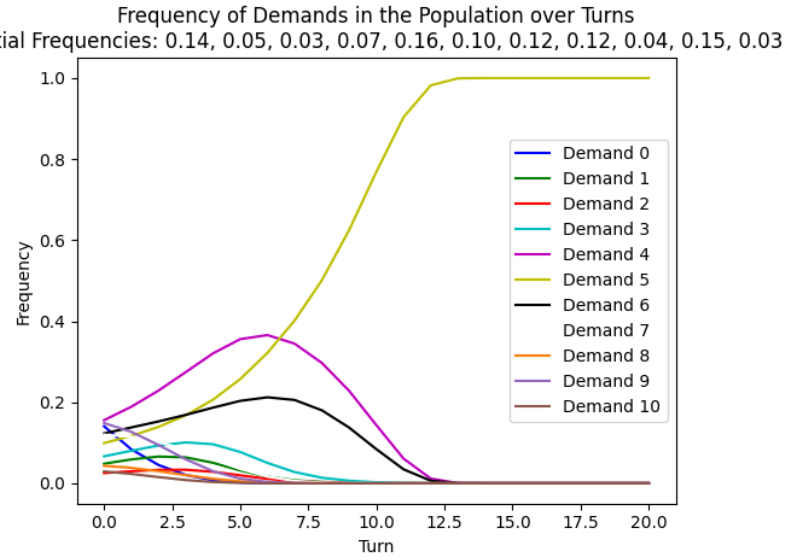

Frequency of Demands in the Population over Turns
Initial Frequencies: 0.14, 0.18, 0.07, 0.07, 0.06, 0.00, 0.15, 0.02, 0.09, 0.12, 0.10
Frequency
Turn
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10

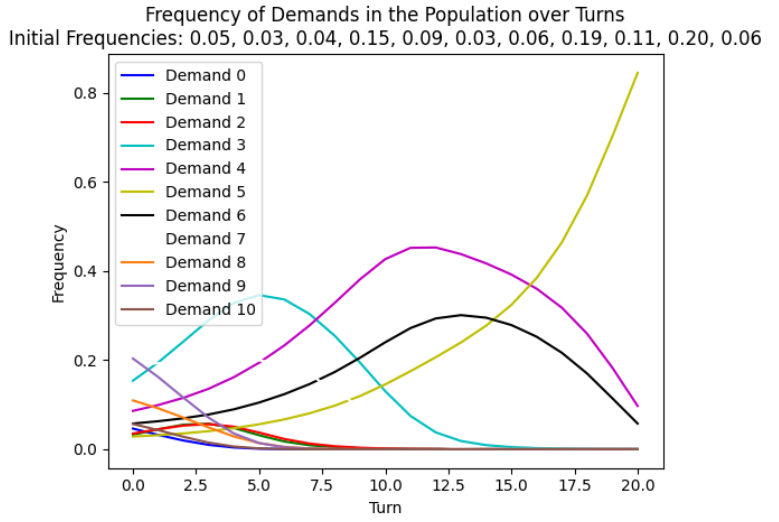

Frequency of Demands in the Population over Turns
Initial Frequencies: 0.11, 0.14, 0.14, 0.12, 0.01, 0.12, 0.05, 0.13, 0.09, 0.07, 0.01
Frequency
Turn
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10

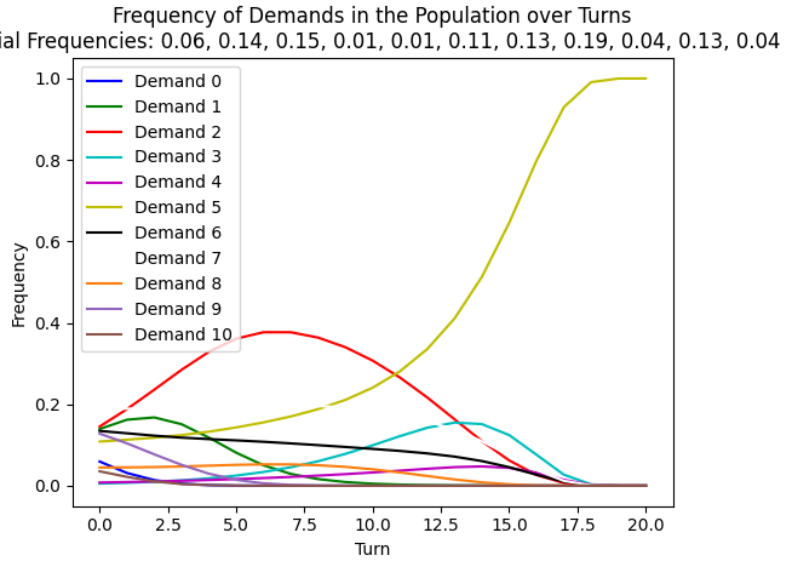

Frequency of Demands in the Population over Turns
Initial Frequencies: 0.12, 0.07, 0.04, 0.15, 0.12, 0.06, 0.13, 0.01, 0.13, 0.10, 0.07
Frequency
Turn
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10

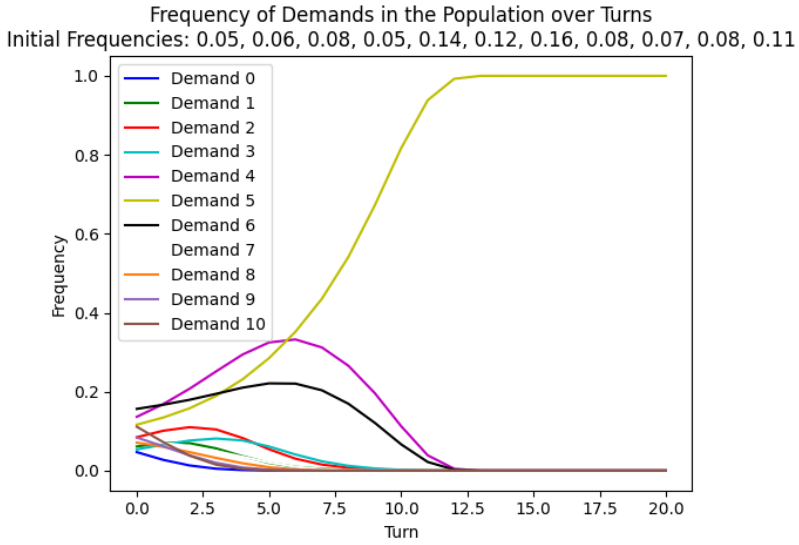

Frequency of Demands in the Population over Turns
Initial Frequencies: 0.02, 0.13, 0.06, 0.15, 0.02, 0.06, 0.16, 0.12, 0.13, 0.10, 0.04
Frequency
Turn
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10

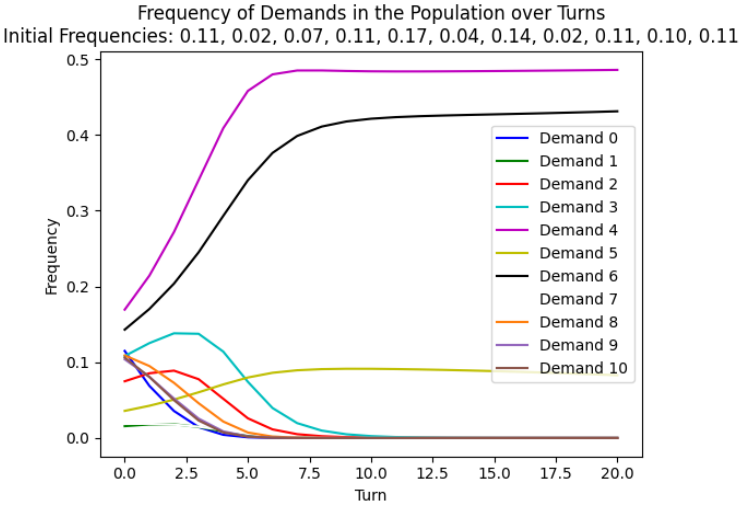
round: 4

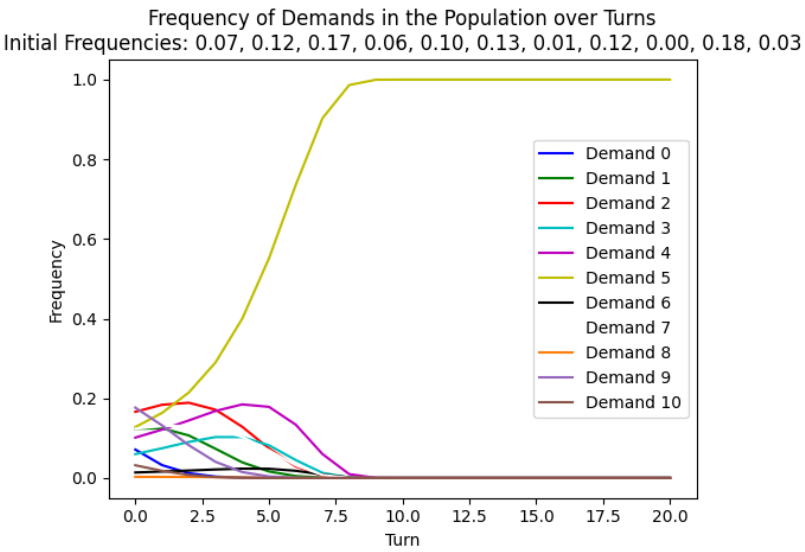

Frequency of Demands in the Population over Turns
Initial Frequencies: 0.14, 0.05, 0.03, 0.07, 0.16, 0.10, 0.12, 0.12, 0.04, 0.15, 0.03
Frequency
Turn
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10


Frequency of Demands in the Population over Turns
Initial Frequencies: 0.05, 0.03, 0.04, 0.15, 0.09, 0.03, 0.06, 0.19, 0.11, 0.20, 0.06
Frequency
Turn
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10


Frequency of Demands in the Population over Turns
Initial Frequencies: 0.06, 0.14, 0.15, 0.01, 0.01, 0.11, 0.13, 0.19, 0.04, 0.13, 0.04
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10
Frequency
Turn


Frequency of Demands in the Population over Turns
Initial Frequencies: 0.05, 0.06, 0.08, 0.05, 0.14, 0.12, 0.16, 0.08, 0.07, 0.08, 0.11
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10
Frequency
Turn


Frequency of Demands in the Population over Turns
Initial Frequencies: 0.11, 0.02, 0.07, 0.11, 0.17, 0.04, 0.14, 0.02, 0.11, 0.10, 0.11
Demand 0
Demand 1
Demand 2
Demand 3
Demand 4
Demand 5
Demand 6
Demand 7
Demand 8
Demand 9
Demand 10
Frequency
Turn



که همانطور که مشخص است نمودارهای کاملا متفاوتی تولید شده اند که هرکدام وابسته به توزیع اولیه هستند.

در نتیجه می توان با توجه به نمودارهای بالا و تفاوت آنها در حالت اولیه گفت، حالت های پایدار که جمعیت می تواند به آن برسد به توزیع فراوانی اولیه تقاضاها بستگی دارد. زمانی که فراوانی تقاضاها در جمعیت به طور قابل توجهی از یک دور turn به دور دیگر تغییر نمی کند، به حالت پایدار می رسد.

به عنوان مثال، اگر همه بازیکنان در ابتدا ۱۰ درخواست کنند، آنگاه همه بازیکنان در نوبت اول ۱۰ امتیاز دریافت خواهند کرد. در نوبت‌های بعدی، بازیکنان به انتخاب تصادفی بازیکنان دیگر ادامه می‌دهند و اگر سود بیشتری داشتند، استراتژی خود را اتخاذ می‌کنند. از آنجایی که همه بازیکنان بازدهی یکسانی دارند، فراوانی تقاضاها در جمعیت تغییر قابل توجهی نخواهد داشت و جمعیت به وضعیت پایداری stable می‌رسد که همه بازیکنان ۱۰ درخواست می‌کنند.

ولی اگر فرکانس های اولیه به طور مساوی بین تمام خواسته ها توزیع شود، در نوبت اول برخی از بازیکنان یک بازده دریافت می کنند در حالی که برخی دیگر صفر دریافت می کنند. در نوبت‌های بعدی،استراتژی بازیکنانی که پاداش دریافت کرده‌اند، احتمالاً توسط بازیکنان دیگر انتخاب می‌شوند. این باعث می شود که تعداد تقاضاهایی که صفر دریافت کرده اند کاهش می یابد. با گذشت زمان، این فرآیند منجر به یک وضعیت پایدار می شود که در آن همه بازیکنان ۵ را درخواست می کنند.

به طور خلاصه، بسته به توزیع فرکانس اولیه تقاضاها، به حالت های پایدار مختلف برسند.