# گزارش پروژه ۲ علوم اعصاب

## علی صالح ۹۷۲۲۲۰۵۳


در این پروژه ما از کتابخانه ای استفاده نکرده و از مدل LIF تمرین قبلی و یک مدل جدید به نام NeuronsGroup استفاده می‌کنیم.

- مدل LIF

  این مدل همانند سری قبل یک سری پارامتر های مخصوص مدل Leaky integrate and fire به همراه یک تابع جریان ورودی دریافت می‌کند و در یک مدت زمان مشخص شده رفتار نورون را به ما نشان می‌دهد.
  تغییری که در کد این مدل نسبت به پروژه‌ی قبلی انجام دادیم این است که بعد از start کردن یک نورون این نورون تا انتهای تایم جلو نمی‌رود.
  از generator استفاده شده و با هر بار next کردن یک آبجکت از این مدل یک واحد زمانی نورون جلو می‌رود.

- مدل NeuronsGroup

  در این مدل ما یک لیست از نورون ها (آبجکت LIF) به همراه یک لیست از کانکشن ها داریم و همچنین پارامتر هایی مثل مقدار تاثیر نورون های Excitatory و Inhibitory با واحد ولت و همچنین مقدار تاخیر این تاثیر به واحد ثانیه را هم ورودی می‌گیرد.
  لیست کانکشن ها هم به این شکل است که خودش شامل n لیست است و لیست i ام شامل اندیس نورون هایی است که نورون i با آنها ارتباط دارد و بعد از هر spike به اندازه ی مقدار تاثیر ورودی با تاخیر که آن هم ورودی داده شده روی آنها تاثیر می‌گذارد.دو متود u_t_plot و raster_plot به ترتیب نمودار u_t ۵ ۵ نورون رندوم از این مجموعه نورون و نمودار اسپایک به زمان کل نورون ها را می‌کشند.

کد گروه نورونی:

```
class NeuronsGroup:

  def __init__(self, neurons, connections, excw=10, inhw=-10,
    exc_delay=1, inh_delay=1, iteration_count=1000):
    self.neurons = neurons
    self.neroun_action = []
    for i in neurons:
      self.neroun_action.append(i.start())

    self.connections = connections
    self.excw = excw
    self.inhw = inhw
    self.iteration_count = iteration_count
    self.spikes = []
    self.exc_spikes_time = []
    self.exc_spikes = []
    self.inh_spikes_time = []
    self.inh_spikes = []
    self.exc_delay = exc_delay
    self.inh_delay = inh_delay
    self.spikes_effect = []

  def start(self):
    self.spikes_effect = [[0] * len(self.neurons) for _ in range(self.iteration_count)]
    for t in range(self.iteration_count):
      flag = False

      for i in range(len(self.neroun_action)):

        action_info = next(self.neroun_action[i])
        if action_info['is_spike']:

          for j in self.connections[i]:
            if self.neurons[i].neuron_type == 'Excitatory':
              self.exc_spikes.append(i + 1)
              self.exc_spikes_time.append(t)
              if t+self.exc_delay < self.iteration_count:
                self.spikes_effect[t + self.exc_delay][j] += self.excw

            if self.neurons[i].neuron_type == 'Inhibitory':
              self.inh_spikes.append(i + 1)
              self.inh_spikes_time.append(t)
              if t+self.inh_delay < self.iteration_count:
                self.spikes_effect[t+self.inh_delay][j] += self.inhw
      for i in range(len(self.neurons)):
        self.neurons[i].u += self.spikes_effect[t][i]
```

```
  def neurons_u_plot(self, neurons_count=5):
    legend = []
    for i in range(min(neurons_count, len(self.neurons))):
      plt.plot(list(map(lambda j: j * self.neurons[i].delta_t_ms,
range(len(self.neurons[i].u_t)))), self.neurons[i].u_t)
      legend.append('neuron ' + str(i+1))
    plt.legend(legend)

  def raster_plot(self):
    plt.scatter(self.exc_spikes_time, self.exc_spikes, color='blue', s=10)
    plt.scatter(self.inh_spikes_time, self.inh_spikes, color='red', s=10)
    plt.legend(['Excitatory', 'Inhibitory'])
```

## آزمایش های neurons group:

### آزمایش اول:

در اولین آزمایش یک جمعیت نورونی شمل دو نورون میسازیم
پارامتر های تک نورون ها به طور دیفالت مقدار های زیر هستند:

```
total_time_ms=100, delta_t_ms=0.1, treshold=-45, u_rest=-70, start_u=-80, u_reset=-65, u_spike=5,
R=10, tau=8
```

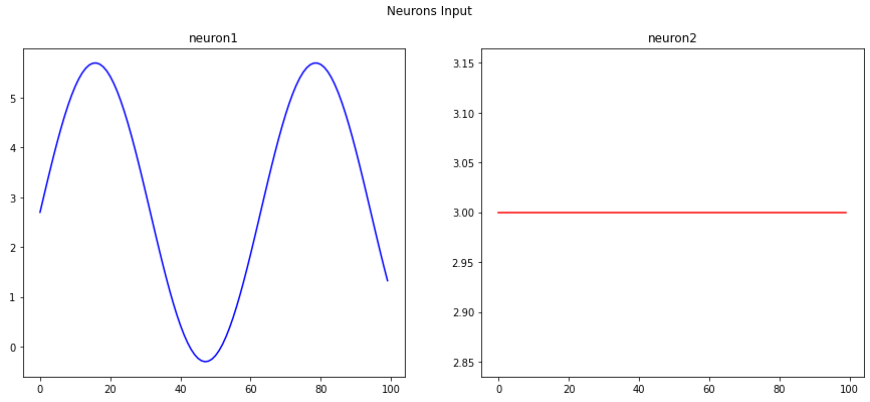
نوع این دو نورون Excitatory است و هر دو با هم ارتباط دارند.
پارامتر های دیگر neurons group:

```
excw=10, inhw=-10, exc_delay=1, inh_delay=1
```

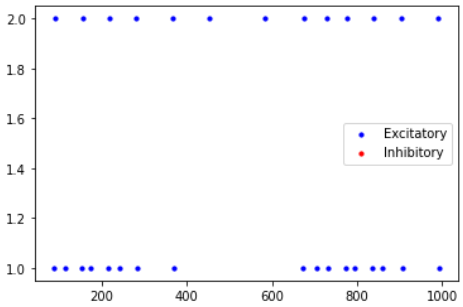
برای اینکه بیشتر مکانیزم لیست کانکشن ها را متوجه شوید به طور مثال در این رابطه لیست کانکشن ها به شکل زیر است:

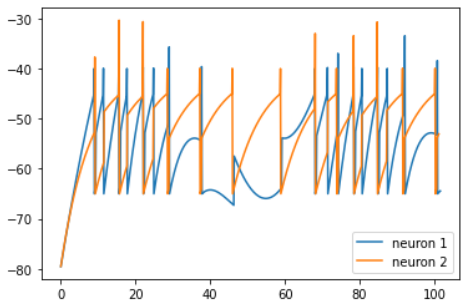
```
[[0] , [1]]
```

جریان ورودی این دو نورون به ترتیب سینوسی و ثابت است:

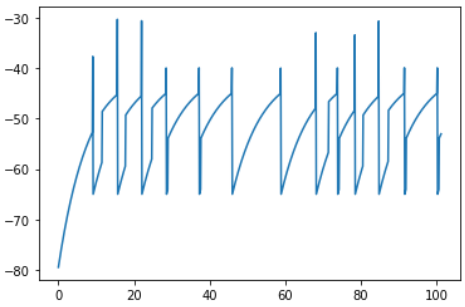
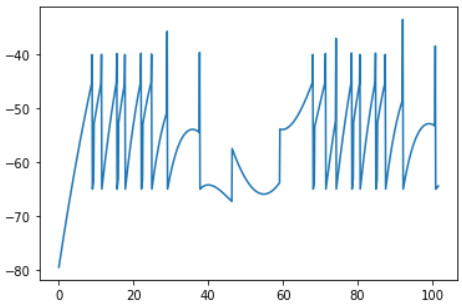



مدل را start می‌کنیم.
نمودار u-t و raster دو نورون را بررسی می‌کنیم.



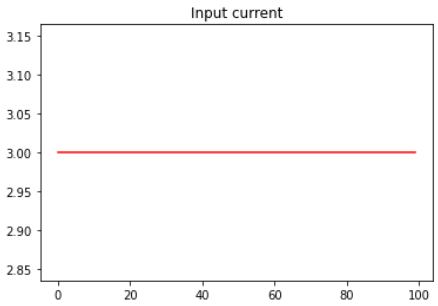
نمودار دو نورون جدا جدا:

تاثیر گذاری اسپایک های نورون های روی یکدیگر کاملا مشخص است.
با هر اسپایک در یکی از نورون ها پتانسیل در نورون دیگر به اندازه excw که ۱۰ ولت است زیاد می‌شود.

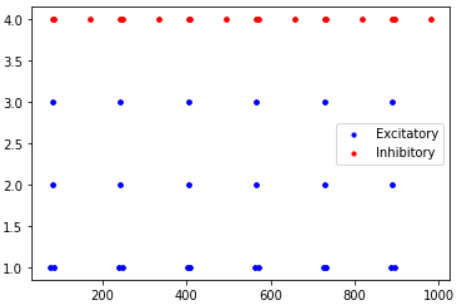
## آزمایش دوم:

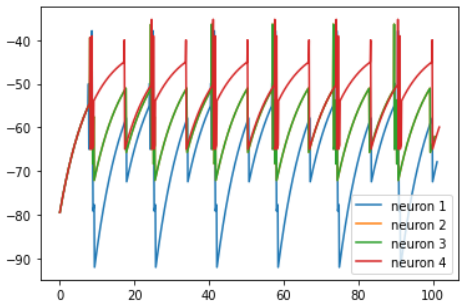
در این آزمایش جمعیت نورونی ما شامل ۴ نورون است که ۳ تای آنها تحریکی و یکی از آنها مهاری است. و جریان همه ی نورون ها جریان ثابت است.




پارامتر های تک نورون ها و جمعیت نورونی همان پارامتر های آزمایش قبلی است به جز treshold نورون اول که به جای -۴۵ مقدار -۵۵ دارد.

مدل را start می‌کنیم.
نمودار u-t و raster دو نورون را بررسی می‌کنیم.

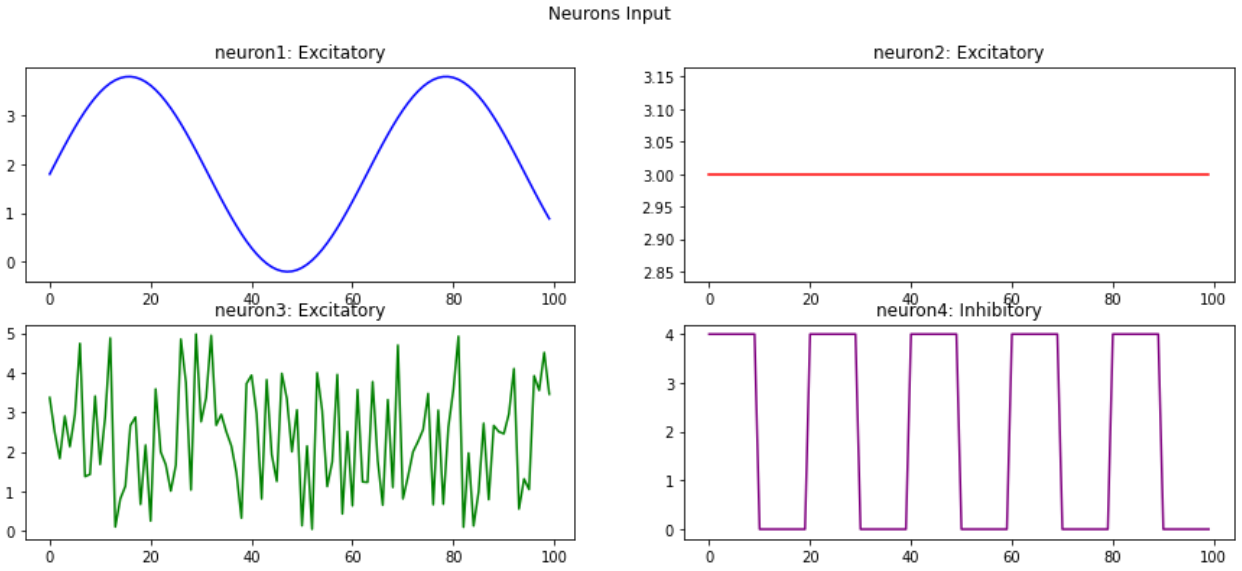




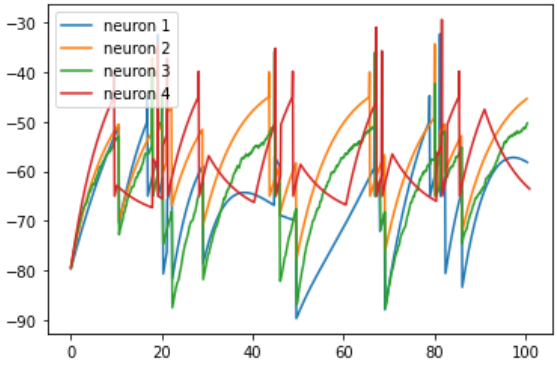
**آزمایش سوم:**

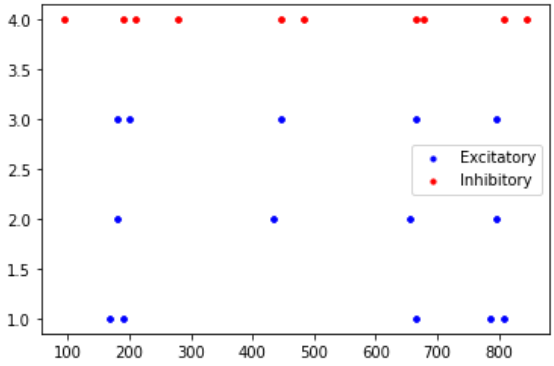
چهار نورون با پارامتر های قبلی و با جریان ورودی های زیر در نظر می‌گیریم.

نورون **step input** مهاری است و مابقی تحریکی




نتایج:



**تابع create neurons group:**

این تابع ورودی از ما تعداد نورون ها و تعداد نورون های مهاری و تعداد نورون های تحریکی و احتمال کانکشن مهاری و تحریکی و همچنین جریان ورودی را از ما می‌گیرد و یک **neuron group** مطابق ورودی می‌سازد.
احتمال کانکشن ها به این شکل عمل میکند که یک عدد بین صفر و یک را می‌گیرد و در تعداد نورون های مهاری یا تحریکی بسته به نوع احتمال ضرب می‌کند و به اندازه ی عدد حاصل ایندکس های رندوم میسازیم و هر نورون را به آنها وصل می‌کنیم.
یعنی اگر احتمال نورون مهاری ۰.۱ باشد نورون های مهاری به ۰.۱ نورون های دیگر وصل هستند و اسپایکشان روی آنها تاثیر می‌گذارد.
همچنین می‌توان یک با **kwargs** پارامتر های دیگر **neurons group** و **Lif** را وارد کنیم.

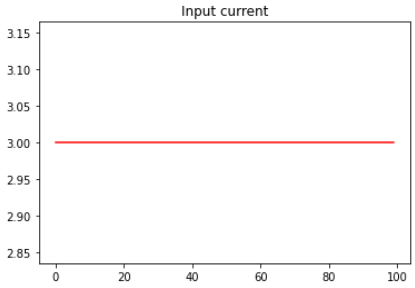
با این تابع قسمت اول پروژه را پیاده سازی می‌کنیم.

```
def create_neuron_group(neurons_count, exc_count, inh_count, exc_prob, inh_prob, I, **kwargs):
  neurons = []
  connections = []
  exc_neuron_conn_count = int(neurons_count * exc_prob)
  inh_neuron_conn_count = int(neurons_count * inh_prob)

  for i in range(exc_count):
    args = {}
    if 'Excitatory' in kwargs.keys():
      for arg in kwargs['Excitatory']:
        args[arg] = kwargs[arg][i]
    neuron = LIF(I[i], neuron_type='Excitatory', **args)
    neurons.append(neuron)
    connections.append(random.sample(range(neurons_count), exc_neuron_conn_count))
  for i in range(inh_count):
    args = {}
    if 'Inhibitory' in kwargs.keys():
      for arg in kwargs['Inhibitory']:
        args[arg] = kwargs[arg][i]
    neuron = LIF(I[i], neuron_type='Inhibitory', **args)
    neurons.append(neuron)
    connections.append(random.sample(range(neurons_count), inh_neuron_conn_count))

  neurons_group = NeuronsGroup(neurons, connections, **kwargs)
  return neurons_group
```

## ۸۰ نورون تحریکی و ۲۰ نورون مهاری

**آزمایش اول:**

در  آزمایش اول به عنوان ورودی به نورون ها جریان ثابت ۳ آمپر می‌دهیم.

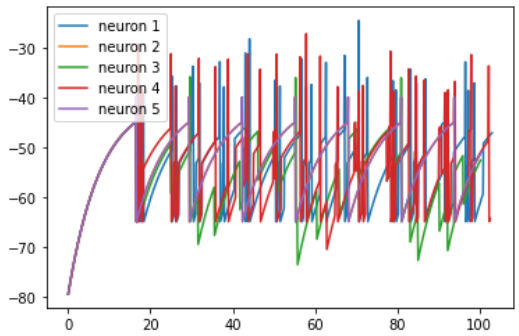


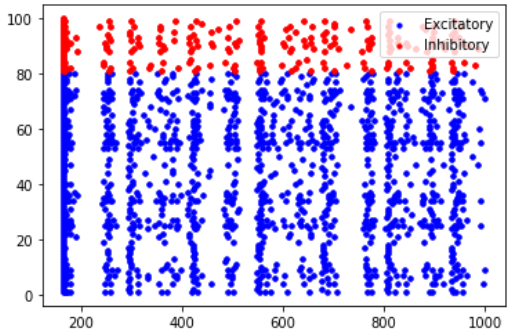

احتمال کانکشن نورون های تحریکی را ۰.۰۲ و احتمال کانکشن نورون های مهاری را ۰.۰۳ قرار می‌دهیم.

پارامتر های تک نورون ها و گروه نورنی:

```
total_time_ms=100, delta_t_ms=0.1, treshold=-45, u_rest=-70, start_u=-80, u_reset=-65, u_spike=5,
R=10, tau=8, excw=10, inhw=-10, exc_delay=1, inh_delay=1
```

نتایج:



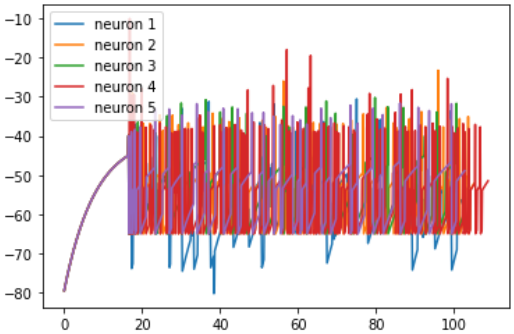




حالا با همان

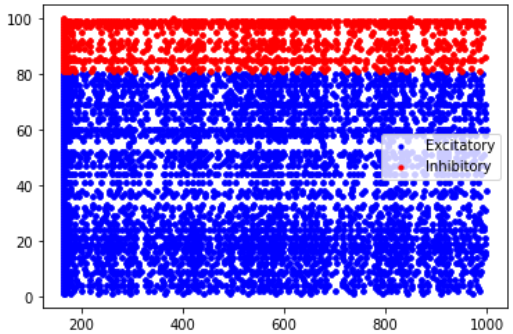
جریان ورودی و پارامتر های سابق احتمال کانکشن نورون های تحریکی را زیاد می‌کنیم.

(Excitatory probability = 0.03, Inhibitory probability = 0.03)

نتایج:




واضح است که با افزایش احتمال

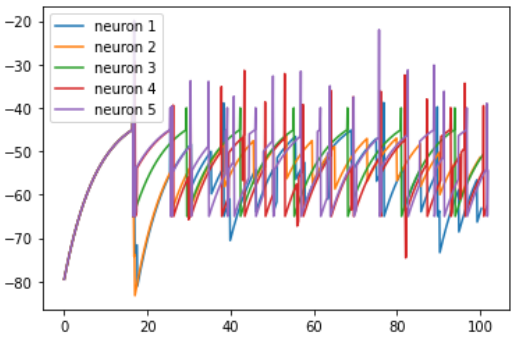


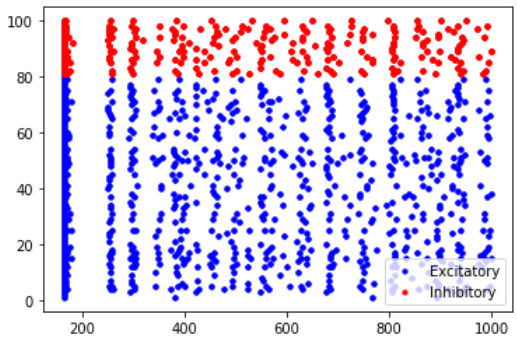
نورون های تحریکی تعداد اسپایک ها بیشتر شده است.

حالا احتمال کانکشن نورون های مهاری را زیاد می‌کنیم.

(Excitatory probability = 0.02, Inhibitory probability = 0.04)

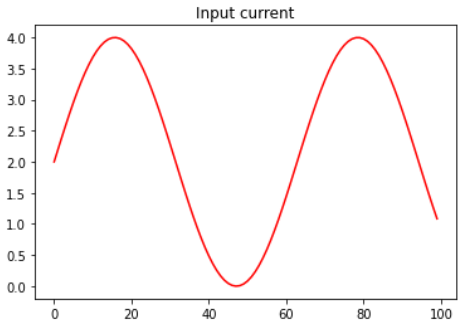
نتایج:



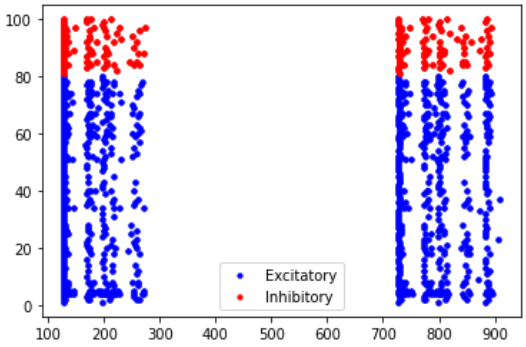
**آزمایش دوم:**

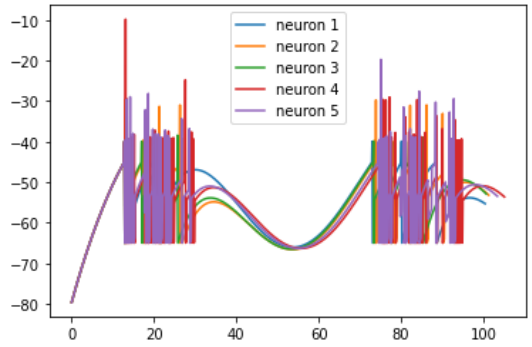
با همان پارامتر های قبلی جریان سینوسی به نورون ها می‌دهیم.




(Excitatory probability = 0.02, Inhibitory probability = 0.03)

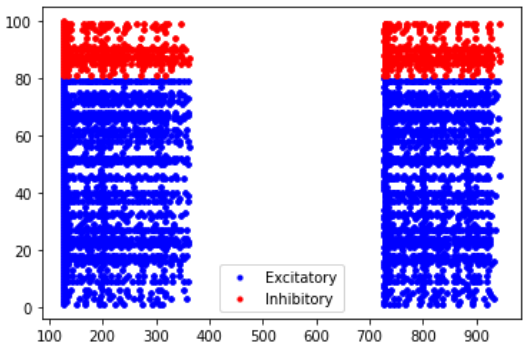
نتایج:

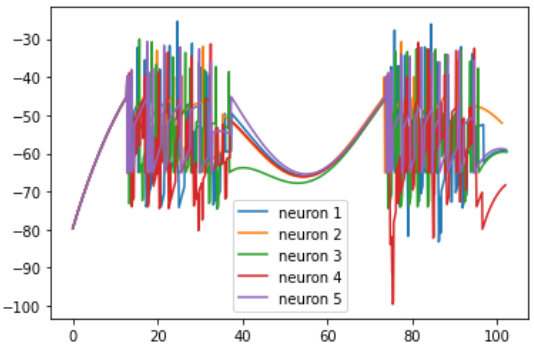



(Excitatory probability = 0.03, Inhibitory probability = 0.03)

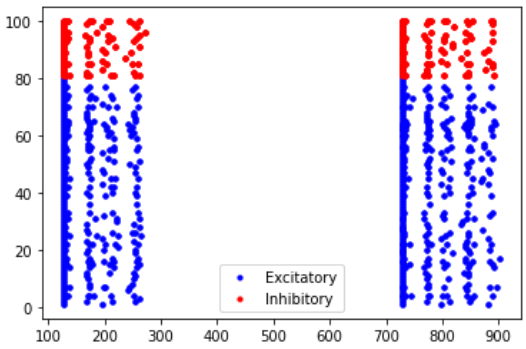
نتایج:

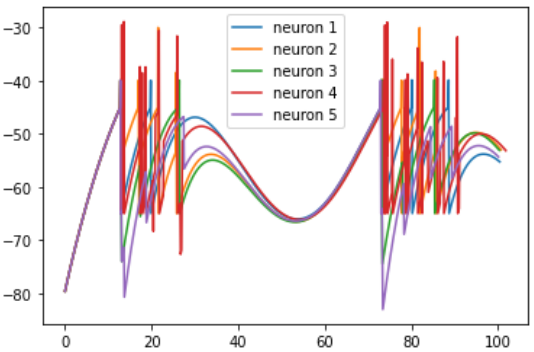



(Excitatory probability = 0.02, Inhibitory probability = 0.04)

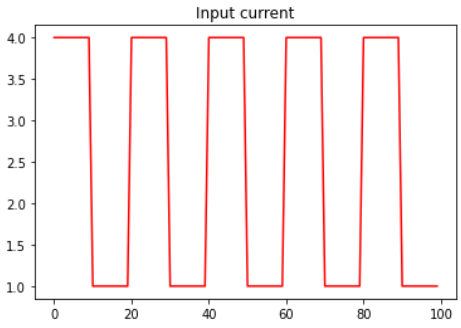
نتایج:



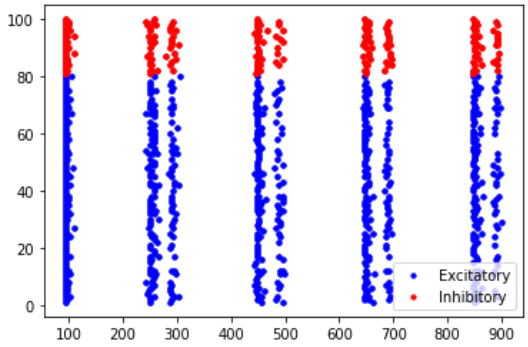
**آزمایش سوم:**

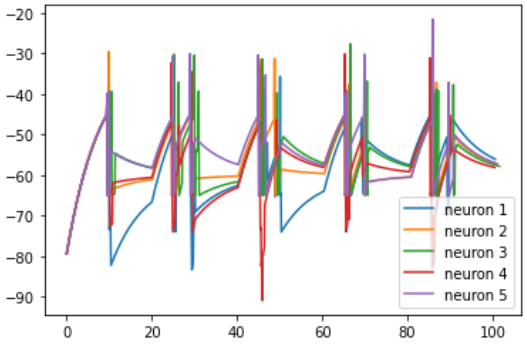
این بار جریان step input می‌دهیم:




(Excitatory probability = 0.02, Inhibitory probability = 0.03)

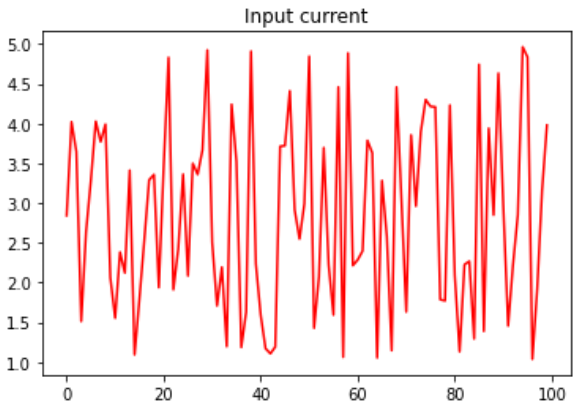
نتایج:



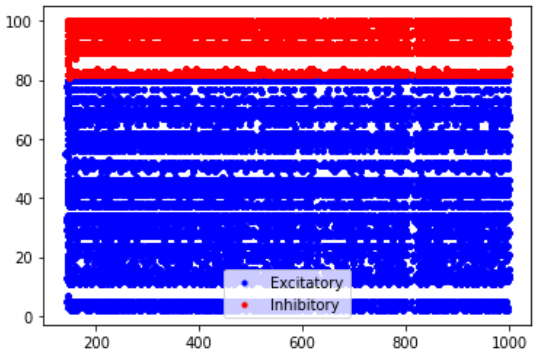
**آزمایش چهارم:**

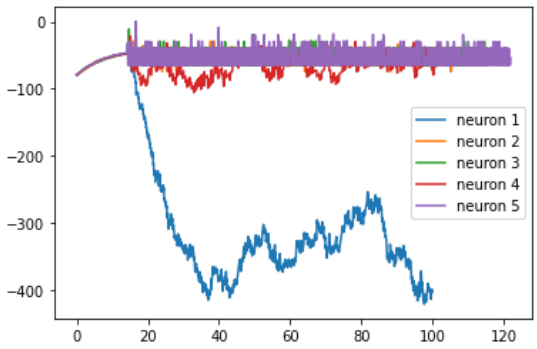
این بار جریان ورودی رندوم می‌دهیم:




(Excitatory probability = 0.05, Inhibitory probability = 0.1)

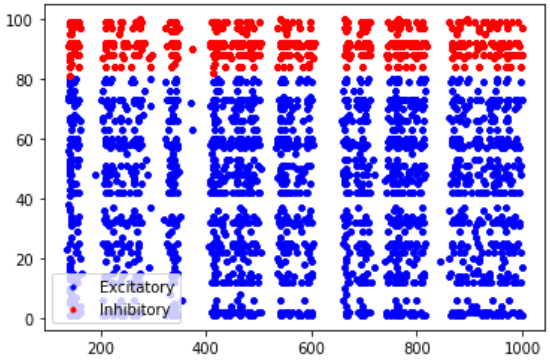
نتایج:

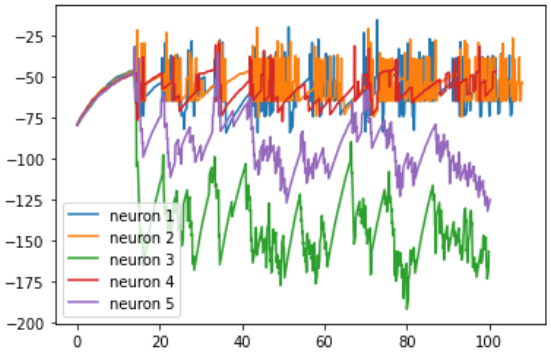



(Excitatory probability = 0.05, Inhibitory probability = 0.2)

نتایج:



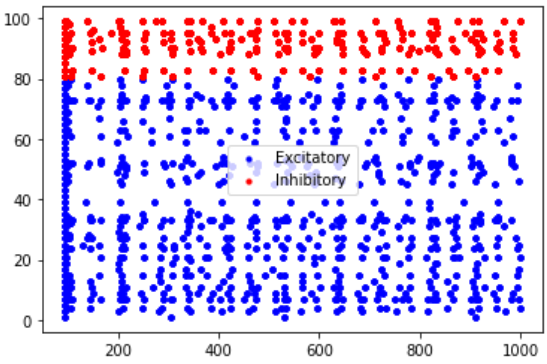



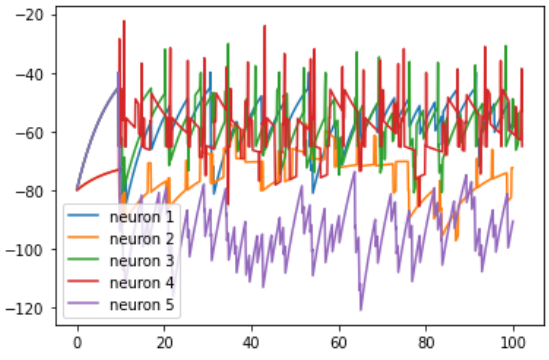
**آزمایش پنجم:**

این بار جریان ورودی ثابت را فقط به نصف نورون ها می‌دهیم و به بقیه نورون ها جریان صفر می‌دهیم.

(Excitatory probability = 0.05, Inhibitory probability = 0.2)

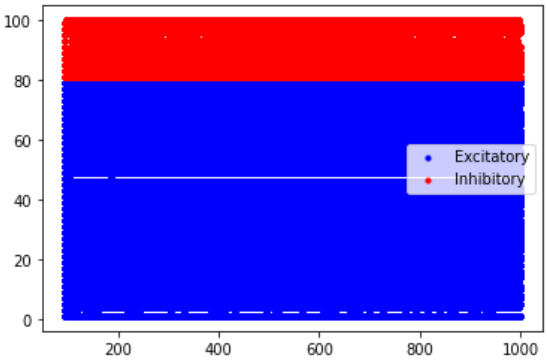
نتایج:

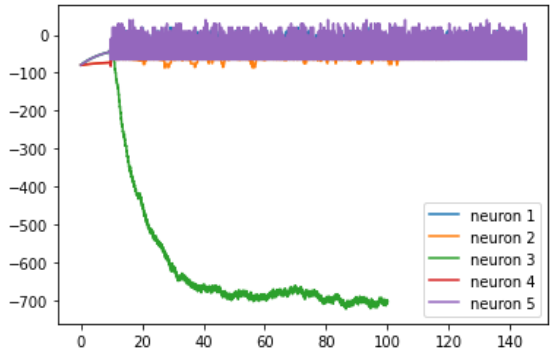



(Excitatory probability = 0.1, Inhibitory probability = 0.2)

نتایج:



## تاثیر گذاری گروه های نورونی روی همدیگر

کد neuron group را عوض می‌کنیم تا بتوانیم جمعیت های نورونی را به هم وصل کنیم.
یک متود connect در کلاس neuron group تعریف می‌کنیم که باعث می‌شود جمعیت نورونی ای که به آن وصل می‌کنیم به عنوان ورودی به این متود می دهیم.

```
def connect(self, neuron_group):
  self.connected_neuron_groups.append(neuron_group)
```

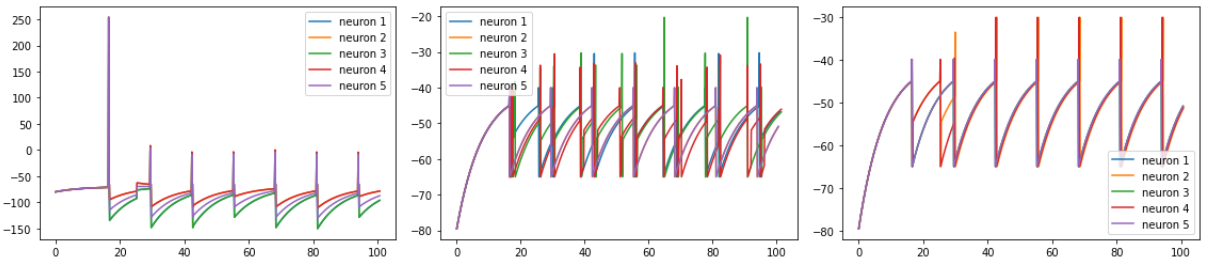
کد start را هم با جنریتور پایتون عوض می‌کنیم که هر n جمعیت نورونی ای که داریم با هم پیش بروند.

```
if spikes_count >= self.spikes_threshold:
  for j in self.connected_neuron_groups:
    for neuron in j.neurons:
      neuron.u += self.connected_neuron_groups_effect
```

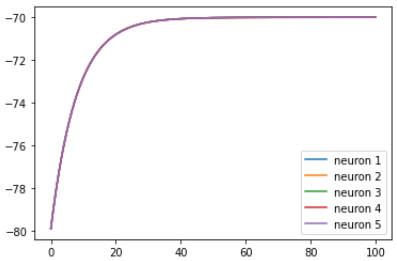
**نحوه ی تاثیرگذاری جمعیت های نورنی روی هم:**
دو پارامتر spikes_threshold و connected_neuron_groups_effect تعریف می‌کنیم.
هر زمانی که تعداد اسپایک های جمعیت نورونی ما از spikes_threshold بیشتر شد پتانسیل کل نورون های همه ی جمعیت هایی که به جمعیت نورونی ما وصل اند را به اندازه connected_neuron_groups_effect اضافه می‌کنیم.

همانطور که در پروژه گفته شده بود ما دو جمعیت نورونی تحریکی و یک جمعیت نورونی مهاری می‌سازیم.
تعداد نورون های هر سه جمعیت ۱۰۰ نورون است.
به دو جمعیت نورونی تحریکی جریان ثابت ۳ آمپر وصل می‌کنیم. و به جمعیت مهاری جریان صفر وصل می‌کنیم.
جمعیت نورونی مهاری را به دو جمعیت تحریکی وصل می‌کنیم.
مقدار spikes_threshold را 20 قرار می‌دهیم و مقدار connected_neuron_groups_effect را 2 آمپر قرار می‌دهیم.
نتایج:

جمعیت نورونی تحریکی ۱: جمعیت نورونی تحریکی ۲: جمعیت نورونی مهاری:



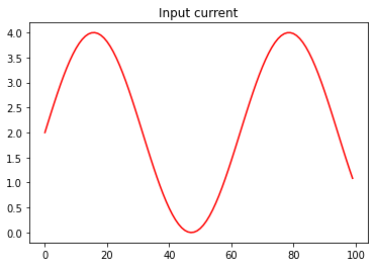

اگر جمعیت نورونی مهاری رو به دو جمعیت نورونی تحریکی وصل نمی‌کردیم به نتیجه زیر می‌رسیدیم:



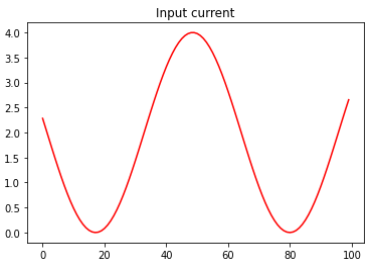

واضح است که دو جمعیت تحریکی روی جمعیت سوم تاثیر گذاشتند و باعث افزایش پتانسیل آن شده اند.

حالا به دو جمعیت تحریکی جریان سینوسی متفاوت می‌دهیم.
جریان ورودی سه جمعیت نورونی:

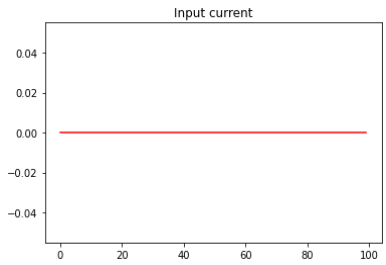
جمعیت نورونی تحریکی ۱:



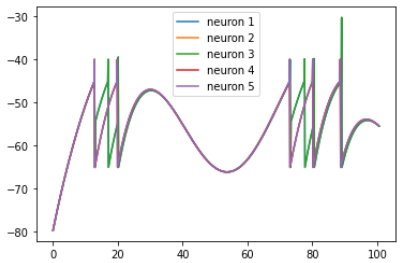

جمعیت نورونی تحریکی ۲:



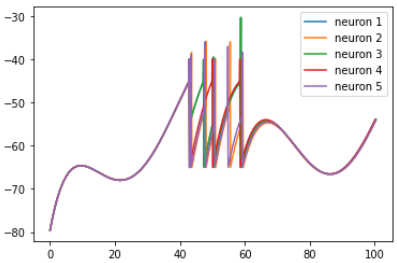

جمعیت نورونی مهاری:




جمعیت نورونی تحریکی ۱:




جمعیت نورونی تحریکی ۲:




جمعیت نورونی مهاری: