گزارش پروژه سوم

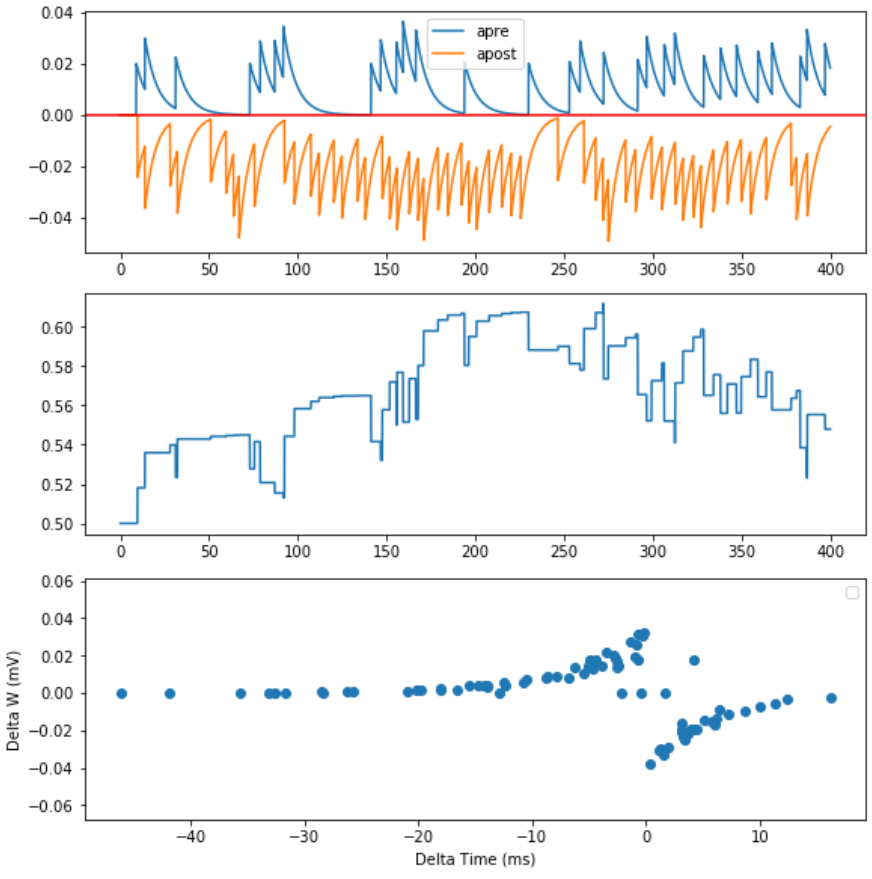
مجتبی کنعانی – ۹۶۲۲۲۰۷۱

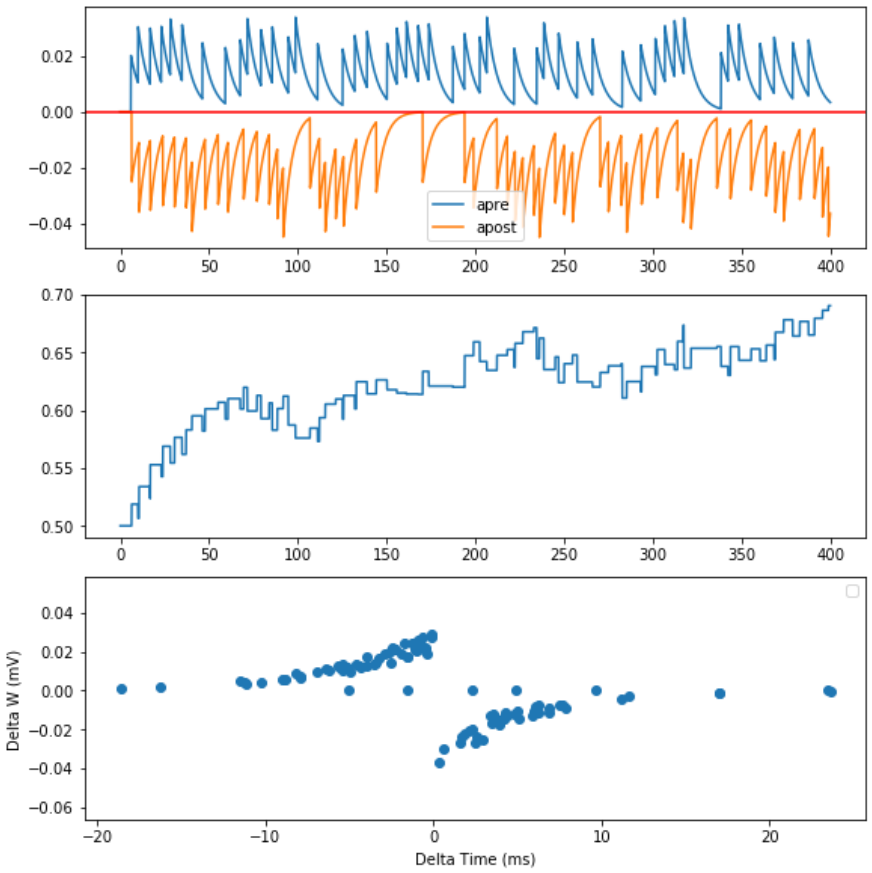
در این پروژه قصد داریم مدل یادگیری stdp را پیاده سازی کنیم.

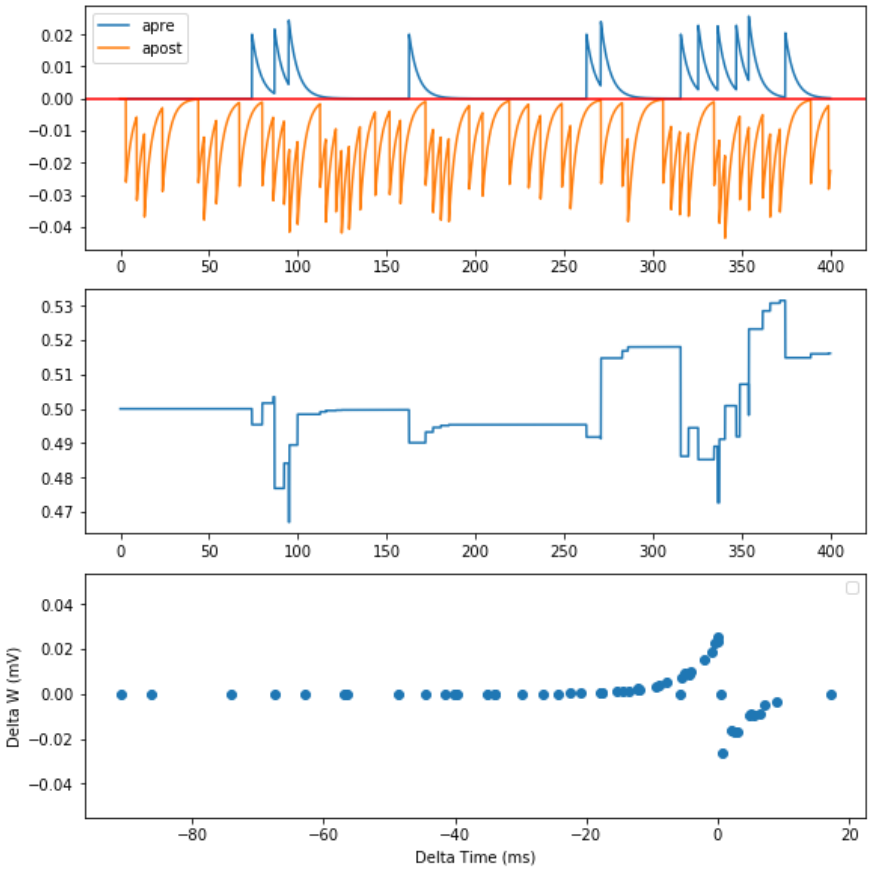
برای سوال اول از دو نورون استفاده میکنیم که به هر دو نورون جریان ورودی پالسی وارد میکنیم تا به پتانسیلشان به حد آستانه برسد و سپس بر اساس زمان spike زدن نورون ها وزن اتصال بین آن ها را تنظیم میکنیم

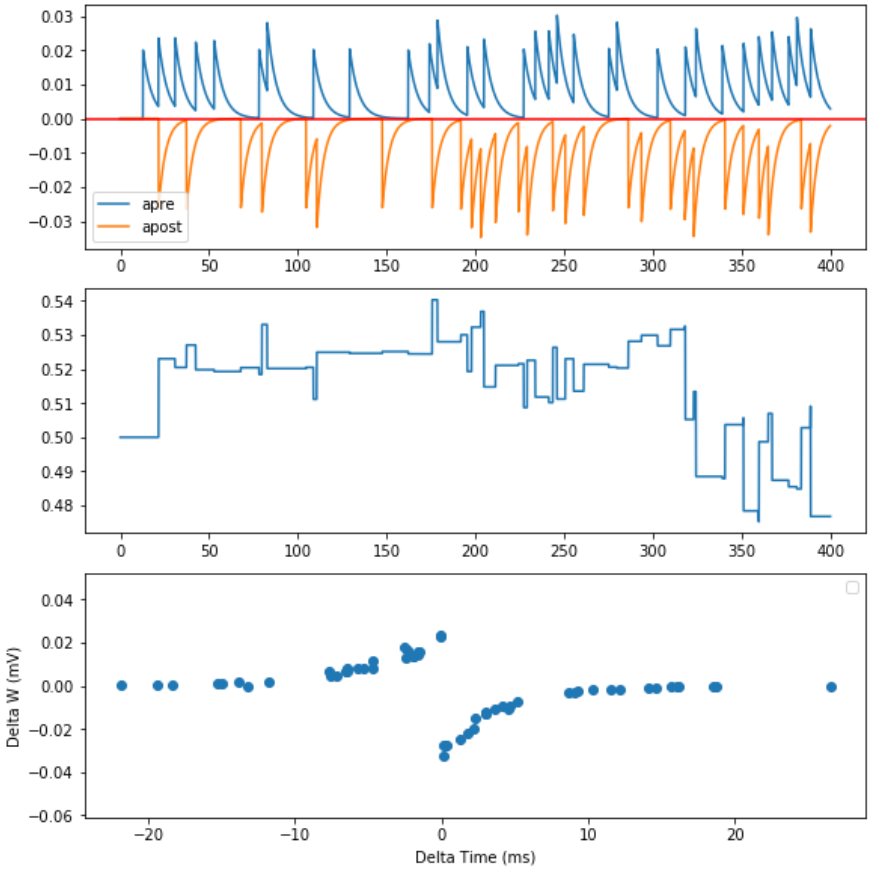
برای اینکار از مدل نورونی LIF استفاده کردیم و پتانسیل نورون pre و post در بخش اول نمودار ها قابل مشاهده است که در آن پتانسیل نورون pre با رنگ آبی در بالا و پتانسیل نورون post با رنگ نارنجی و برعکس در پایین قرار دارد و این نمایش به این دلیل است که در یادگیری stdp در زمان spike زدن نورون pre باید به اندازه پتانسیل نورون post ضربدر یک ضریب تابت -A مقدار w را کاهش دهیم، همینطور در زمان spike زدن نورون post به اندازه پتانسیل نورون pre ضربدر ضریب تابت +A مقدار w را افزایش دهیم که از رابطه زیر پیروی میکند.

در بخش دوم نمودار نیز تغیرات وزن را مشاهده میکنید و در بخش سوم نمودار نیز تغییرات w نسبت به فاصله بین اسپایک نورون pre و post را مشاهده میکنید. در زیر چند نمونه از این آزمایش هارا مشاهده میکنید.




apre
apost
Delta W (mV)
Delta Time (ms)


apre
apost
Delta W (mV)
Delta Time (ms)



متاسفانه موفق به پیاده سازی درست تمرین شماره ۲ و ۳ نشدم، توضیح مختصری از کد تمرین دوم در زیر موجود است.

کد شامل ۳ تابع است:

: Make_poisson_pattern_current

ورودی ها:

N : تعداد نورون های pre

Patterns : الگو های موجود برای این نورون ها

Duration : زمان شبیه سازی

Delta_time : کوچک ترین بازه زمانی

Possibility : احتمال وجود pattern در زمان اسپایک های نورون های pre

از خروجی این تابع در SpikeGeneratorGroup استفاده میشود و قصد داریم زمان تمام اسپیگ های نورون های pre در طول آزمایش را در این تابع مشخص کنیم. برای اینکار ابتدا بزرگترین عدد موجود در الگو ها را پیدا میکنیم و آن را L مینامیم. سپس کل بازه مورد نظرمان را با گام هایی به طول L پیشروی میکنیم و در هر گام با توجه به مقدار possibility تصمیم میگیریم که آیا باید در این گام از الگو ها استفاده بکنیم یا از نویز ، اگر قرار شد از یک الگو استفاده کنیم باز هم بین الگو هایی که در patterns وجود داد یکی را به صورت تصادفی انتخاب میکنیم و زمان اسپایک ها را با توجه به آن محاسبه و ذخیره میکنیم و اگر قرار شد از نویز استفاده کنیم

مقداری random برای هر یک از نورون ها در بازه مربوطه مشخص میکنیم به صورتی که وارد گام بعدی نشود.

از ٤٠٠ ثانیه آخر برای مرحله test استفاده میکنیم ، در ثانیه 300 – duration الگوی اول را وارد میکنیم، در ثانیه 200 – duration الگوی دوم را وارد میکنیم و در ثانیه 100 – duration نویز وارد میکنیم تا در آخر بتوانیم از روی این سه مرحله درستی یادگیری را بررسی کنیم.

:pulse_input_current_generator

ورودی ها:

Max_I : مقدار جریان در زمان پالس زدن

Duration : زمان شبیه سازی

Delta_time : کوچک ترین بازه زمانی

Possibility : احتمال وجود جریان پالسی

این تابع به ازای تمام لحظات شبیه سازی یکی از دو مقدار ۰ و max_i را به عنوان جریان ورودی در آن لحظه انتخاب میکند که احتمال انتخاب max_i برابر possibility و احتمال انتخاب ۰ برابر possibility-۱ است

از این تابع برای تولید جریان ورودی در نورون های post استفاده میشود و در آن max_i برابر ۰ در نظر گرفته میشود یعنی عملا به ازای تمام لحظات جریان ۰ وارد نورون های post میشود و یعنی تنها عامل افزاینده پتانسیل در این نورون ها اسپایک زدن نورون های pre میباشد.

Simulate

ورودی ها:

Coef : در این شبیه سازی وزن ها بین ۰ تا ۱ قرار دارند و چون مقدار کوچکی است، در زمان اسپایک زدن نورون های pre ضریب ثابت coef ابتدا در w ضرب میشود و سپس مقدار آن به پتانسیل نورون post اضافه میشود.

Duration : زمان شبیه سازی

Apre : ضریب +A در فرمول stdp

I_1 , I_2,P_1,P_2 : مقادیری که برای دو نورون post در تابع pulse_input_current_generator استفاده میشود و همانطور که گفته شد همه برابر ۰ در نظر گرفته میشوند.

P_pattern : مقدار possibility در تابع Make_poisson_pattern_current

Tau_pre : ثابت زمانی نورون های pre synaptic

اندیس ۱ و ۲ اشاره به نورون post synaptic اول و دوم دارد.

U_rest_post : پتانسل rest نورون مربوطه

Tau_post : ثابت زمانی نورون مربوطه

Threshold_post : پتاسیل آستانه نورون مربوطه

Resistance_post : مقاومت نورون مربوطه – از آن جا که مقدار جریان ورودی در این دو نورون برابر ۰ است و مقدار مقاومت در آن ضرب میشود این مقدار اهمیتی ندارد

در این تابع پس از فراخوانی توابع بالایی و ابتدایی از خروجی تابع اول یک spikegeneratorgroup برای نورون های pre میسازیم و نورون های post را نیز با یک مدل lif و به کمک مقادیر ورودی میسازیم و این نورون ها را به یکدیگر متصل میکنیم و stdp را در بخش synapses پیاده سازی میکنیم.

نمودار ها شامل ۸ بخش است:

۱ : مقدار پتانسیل نورون pre و post برای یکی از سیناپس های دلخواه.

۲: مقدار پتاسیل نورون اول در طول کل زمان آموزش و تست

۳: مقدار پتاسیل نورون دوم در طول کل زمان آموزش و تست

٤: مقدار پتاسیل نورون اول در زمان تست

٥: مقدار پتاسیل نورون دوم در زمان تست

٦: تغییرات وزن ها در زمان آزمایش

۷: زمان اسپایک نورون های post

۸: زمان اسپایک نورون های pre

در این نمودار ها خطوط عمودی قرمز نشاندهنده حضور الگوی ۱، خطوط آبی نشان دهنده حضور الگوی ۲ و آخرین خطی که به رنگ نارنجی علامت گذاری شده است نشاندهنده حضور نویز است، نویز در زمان آموزش نیز وجود دارد اما در آن جا با خطوط رنگی مشخص نشده است و هر جایی که آبی یا قرمز نباشد نویز قرار دارد. انتظار داریم در نمودار های ٤ و ٥ مقدار پتانسیل در حضور الگوی یک و دو خیلی بیشتر از حضور نویز باشد اما چنین نیست و تقریبا در یک سطح قرار دارند.

برای کد شماره ۳ نیز از همین شیوه استفاده کردم با این تفاوت که نورون سومی در لایه آخر وجود دارد که وظیفه inhibit کردن را به عهده دارد و delay ها به شکلی که در صورت سوال مطرح شده تنظیم شده اند اما در این سوال هم موفق به گرفتن نتیجه درست نشدم.