

به نام خدا



دانشگاه صنعتی شریف

دانشکده مهندسی برق

علوم اعصاب یادگیری، شناخت و حافظه

پروژه‌ی پایانی درس: بررسی

استاد: دکتر کربلایی آقاجان

امیرحسین برقراری شماره دانشجویی

مازیار شمسی‌پور ۹۸۱۰۱۸۴۴

۲۷ تیر ۱۴۰۰

مقدمه

توضیحات کلی در مورد کدها و پروژه و مقاله.

۱ آشنایی با مقاله پژوهش

۱.۱ هدف پژوهش

هدف این مقاله مدل سازی receptive field نوروهای پیچیده غش بصری^۱ می باشد. در واقع با اعمال تحریک های تصادفی spatiotemporal از توزیع $p(s)$ و مشاهده پاسخی نوروها می خواهیم راستایی را بدست بیاوریم که در آن $p(s|r)$ تفاوت معناداری با توزیع اولیه تحریک ها داشته باشد. تفاوت این مقاله با پژوهش های قبل خود آن است که به دلیل رفتار غیر خطی نوروهای پیچیده موجود در Visual cortex نمی توانیم از آنالیزهای خطی spike-triggered average که پیش از این کارگشا بودند استفاده کنیم. این مقاله روش spike-triggered correlation analyses را پیشنهاد می دهد که بر پایه Wiener Kernel طراحی شده است.

نهایتا با این پژوهش با بررسی هایی که انجام می دهد ادعا می کند که می توان با این روش پایه ای برای تحریک ها ارائه داد که تعداد کمی از آن ها مشخص کننده ویژگی های مرتبط و تعداد زیادی مربوط به ویژگی های پوچ می باشند.

۲.۱ نوروهای «پیچیده»

این مفهوم اولین بار در مقاله نوبلیست Hubel and Wiesel مطرح شده است. نوروهای ساده نوروهایی هستند که رابطه تحریک-پاسخ آنها نسبت به زمان خطی می باشد. این موضوع باعث به وجود آمدن مناطق ON-OFF در Receptive Field نورو مورد نظر می شود. به عکس دست نوشته موجود در مقاله مذکور برای جزئیات بیشتر توجه کنید.



شکل ۱: مناطق ON-OFF حوزه دریافتی نوروهای ساده^۲

همچنین در بخش Material and Methods آمده است که اگر نسبت هارمونیک اول به مقدار DC حوزه دریافتی بزرگتر از یک باشد نورو را ساده می نامیم. روشن است که نوروهایی که از تعاریف بالا پیروی نکنند ساده نبوده و پیچیده اند.

^۱ Visual cortex

^۲ EVOLUTION OF IDEAS ON THE PRIMARY VISUAL CORTEX, 1955-1978. BY DAVID H. HUBEL

STC analysis ۳.۱

می‌دانیم که اگر یک ویژگی در تحریک باعث تغییر احتمال اسپایک زدن شود، می‌دانیم که اگر تحریک‌ها را به فضای آن spike-triggered منتقل کنیم باید تفاوت قابل توجهی بین مقادیر $p(s|r)$ و $p(s)$ باشد.^۳

روش‌های مبتنی بر correlation با هدف بررسی تغییر این توزیع احتمالات با توجه به تغییرات گشتاور مرتبه‌ی دوم آنها (واریانس) طراحی می‌شوند.

در این بین روش PCA به این خاطر که پایه‌ای در اختیار ما قرار می‌دهد که راستاهای بیشترین تا کمترین واریانس می‌باشند، بسیار مناسب کار ما خواهد بود و می‌تواند ویژگی‌هایی را که در آن واریانس تحریک‌ها بسیار بالا می‌باشد را در اختیار ما بگذارد.

روش spike-triggered analysis مبتنی بر موارد فوق مراحل زیر را انجام می‌دهد:

۱. ابتدا با فرض اینکه حافظه‌ی نورون‌ها بیشتر از ۱۶ فریم یا ۲۶۸ میلی‌ثانیه نخواهد بود؛ پترن‌های تحریک را متشکل از ۱۶ نوار رنگی در ۱۶ فریم و در فضای ۲۵۶ بعدی تعریف می‌کنیم.

۲. ماتریس spike-triggered correlation را به صورت زیر تعریف می‌کنیم:

$$C = \frac{1}{N} \sum_{i=1}^N S(i)^T S(i)$$

که در آن $S(i)$ بردار ۲۵۶ بعدی در i امین باریست که نورون در طول آزمایش اسپایک زده است و N تعداد کل اسپایک‌ها در طول آزمایش است.

۳. بردارویژه‌ها و مقدارویژه‌های این ماتریس محاسبه شده و با توجه به اندازه‌ی مقدارویژه‌ها رتبه‌بندی می‌شوند.

با توجه به PCA می‌دانیم که حاصل راستاهایی خواهد بود که در آن واریانس تحریک‌هایی که موجب اسپایک می‌شوند از زیاد به کم مرتب شده‌اند.

۴.۱ اعتبارسنجی نتایج

برای اعتبارسنجی مشاهدات روش spike-triggered correlation روشی که مقاله استفاده می‌کند این است که دنباله‌ای از اسپایک‌های تصادفی با تعداد اسپایک‌های مساوی با N تولید می‌کند و ماتریس correlation تحریک‌هایی که موجب این اسپایک‌های فرضی تصادفی شدند را ایجاد می‌کنیم. هدف این است که control correlation matrix را محاسبه کنیم که به این معناست

^۳ $p(\text{response} | \text{stimulus})$

که روشی که در قبل پیش گرفتیم را برای داده‌هایی که می‌دانیم مستقل از اسپایک زدن هستند (تمام داده‌ها) نیز اعمال کنیم. با توجه به حجم بالای داده‌ها N مساوی تعداد اسپایک‌ها را در نظر می‌گیریم و ۵ بار فرایند تولید $\text{control correlation matrix}$ را انجام می‌دهیم و میانگین مقدارویژه‌ها را محاسبه می‌کنیم. با توجه به متن مقاله حاصل این میانگین‌گیری $\pm 5/2 SD$ بازه‌ی اطمینان ما با $p < 10^{-4}$ خواهد بود. و بنابراین اگر مقدارویژه‌ای خارج از این بازه‌ی اطمینان باشد؛ آن مقدارویژه و بردارویژه دارای تفاوت آشکار خواهند بود. (در واقع آن بردار ویژه راستایست که تحریک‌های موجب اسپایک در آن واریانس و در نتیجه توزیع احتمال کاملاً متفاوتی با توزیع اولیه دارند)

۵.۱ نتایج پژوهش

همانطور که در بخش‌های قبل اشاره کردیم هدف پیدا کردن eigenvalue هایی بود که خارج از بازه‌ی اطمینان باشند که آن‌ها به عنوان ویژگی‌های اصلی موجود در receptive-field گزارش شود. نتایج مقاله نشان می‌دهد که برای تعداد زیادی از سلول‌های پیچیده دو بردارویژه پیدا شده‌است که این بردار ویژه‌ها ناحیه‌ی ON-OFF مشخص و مجزا از همی دارند و همچنین correlation بسیار پایینی دارند. همچنین می‌بینیم که این راستاها به خوبی سبب تحریک نورون‌ها می‌شوند با این حال برای eigenvalue هایی که در بازه‌ی اطمینان مذکور قرار دارند تاثیر بسیار کمتری در تحریک نورون‌ها مشاهده شده است.

۲ آشنایی با دیتاست

۱.۲ بخش اول

با استفاده از تابع تعبیه شده `fget_hdr.m` یکی از فایل‌های `sa0` را باز می‌کنیم و به محتویات آن توجه می‌کنیم.

hdr.DataInfo		hdr.FileInfo	
Field	Value	Field	Value
ID	'a01bmsq1D'	Type	'DAN_SPK'
DataFrom	'000412.a01bmsq1D.ra1'	Version	'1.0'
Channel	0	Fname	'000412.a01bmsq1D.s...
SampleRate	10000	Creator	'f21mlv.vi'
Gain	10000	Time	'2000,4,12,20,10,16'
DAQMode	0		
DAQResolution	0		
DataType	0		
DataUnit	0		
TimeMode	0		
TimeOffset	0		
ThreshPeakHigh	6542		
ThreshPeakLow	53250		
ThreshValleyHigh	-66541		
ThreshValleyLow	-28176		
ThreshWidthMax	-19		
ThreshWidthMin	506		

شکل ۲: محتویات فایل `hdr`

می‌توانیم ببینیم که در این فایل اطلاعاتی اطلاعاتی مربوط به خود فایل از جمله زمان ایجاد دیتاست و سازنده‌ی آن و همچنین اطلاعاتی در مورد داده‌ها از جمله نرخ نمونه‌گیری وجود دارد. همچنین در فایل `log` نیز اطلاعاتی مربوط به نوری که از آن نمونه‌گیری می‌شود وجود دارد.

نوع اطلاعات	برخی از اطلاعات موجود
اطلاعات مربوط به فایل	زمان ایجاد فایل، سازنده‌ی فایل، ورژن فایل
اطلاعات مربوط به تحریک	نوع تحریک، نرخ بروزسانی مانیتور، نرخ تغییر فریم‌ها
اطلاعات مربوط به <code>tuning curve</code>	زاویه‌ی ترجیحی نورون، طول و عرض ترجیحی نورون
اطلاعات مربوط به نمونه‌گیری	نرخ نمونه‌گیری، نوع کارگذاری الکترودها، تعداد چنل‌ها

و اطلاعات دیگری از این دست می‌باشد.

۲.۲ بخش دوم

تابع `Func_ReadData` را به صورتی که خواسته شده پیاده‌سازی می‌کنیم. ایده‌ی کلی در پیاده‌سازی این تابع این بود که ابتدا با استفاده از تابع `dir` فایل‌های موجود در دایرکتوری نورون مورد نظر را می‌خوانیم و فایل‌های `sa0` را پیدا می‌کنیم و اطلاعات آن را با استفاده از `fget_spk.m` می‌خوانیم.

همچنین با توجه به اینکه در بخش بعد به Spike-count rate احتیاج پیدا می‌کنیم یکی از خروجی‌های Optional را برابر با آن تعریف می‌کنیم، برای محاسبه‌ی این مقدار نیز تنها کافیست توجه کنیم که:

$$\langle r \rangle = \left\langle \frac{\#spikes}{time} \right\rangle_{trials}$$

نهایتاً تابع مورد نظر به صورت زیر خواهد بود:

۱.۲.۲ قطعه کد: تابع Func_ReadData

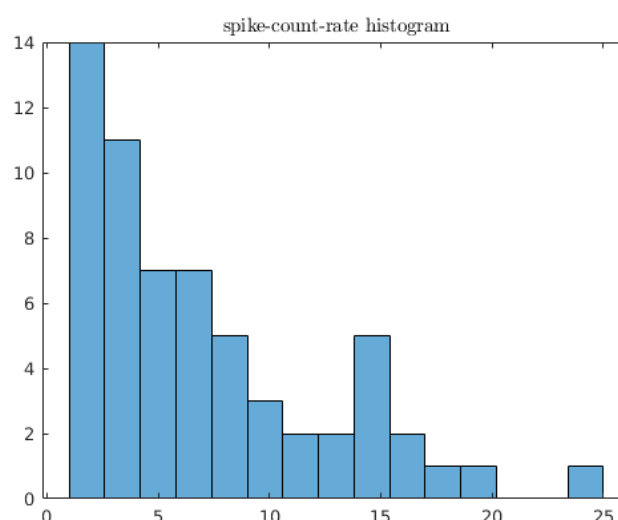
```

1 % Recomend: select Data/ and MatlabFucntions/
2 % directories and add them to matlab PATH
3 function [Output, spike_count_rate] = Func_ReadData(NeuronCode)
4     Path = [pwd, '/Data', '/Spike_and_Log_Files/', NeuronCode];
5     Listing = dir(Path);
6     Index_Counter = 0;
7     events = {};
8     hdrs = {};
9     frame_rate = 59.72; %Hz
10    frame_number = 32767;
11    spike_count = [];
12    for i = 1:length(Listing)
13        if (contains(Listing(i).name, 'msq1d.sa0', 'IgnoreCase', true)
14            && ...
15            ~contains(Listing(i).name, 'sa0.sub', 'IgnoreCase', true)
16            && ...
17            ~contains(Listing(i).name, 'sa0.', 'IgnoreCase', true))
18            Index_Counter = Index_Counter + 1;
19            [events{Index_Counter}, hdrs{Index_Counter}] = fget_spk(
20                Listing(i).name, 'get');
21            spike_count(Index_Counter) = length(events{Index_Counter});
22        end
23    end
24    Output = struct('events', events, 'hdr', hdrs);
25    spike_count_rate = mean(spike_count) * frame_rate/frame_number;
26 end

```

۳.۲ بخش سوم

در این بخش ابتدا با استفاده از تابع `dir` در پوشه‌ی مربوط به اطلاعات اسپایک‌ها نام تمام نورون‌ها را استخراج می‌کنیم. سپس با پیمایش بر روی آن‌ها و با استفاده از تابعی که در بخش قبل نوشتیم نرخ اسپایک هر نورون را نگهداری می‌کنیم. نهایتاً هیستوگرام نرخ اسپایک‌ها به صورت زیر خواهد بود:



شکل ۳: هیستوگرام نرخ اسپایک‌های نورون‌ها

سپس به سادگی می‌توانیم نورون‌هایی که نرخ اسپایک کمتر از ۲ دارند را با استفاده از قطعه کد زیر از لیست اولیه‌ای که برای کد نورون‌ها تهیه کردیم حذف کنیم.

۱.۳.۲ قطعه کد: حذف نورون‌ها با نرخ اسپایک پایین

```

1 index_counter = 0;
2 for i = 1:61
3     if SCRA(i) < 2
4         index_counter = index_counter + 1;
5         index(index_counter) = i;
6     end
7 end
8 fprintf('Excluded Neurons:\n')
9 disp(neuron_codes(index))
10 neuron_codes(index) = [];

```

خروجی این قطعه کد شماره‌ی نورون‌های حذف شده را نمایش می‌دهد که به صورت زیر است:

Excluded Neurons :

"000413.b03"
 "000413.b04"
 "000413.b05"
 "000418.a01"
 "000420.b02"
 "000524.c01"
 "000907.f07"

۴.۲ بخش چهارم

در این قسمت تابع `Func_StimuliExtraction` به گونه‌ای پیاده می‌کنیم که با دریافت دنباله‌ای از زمان اسپایک‌ها تحریک‌هایی با ابعاد 16×16 که سبب این اسپایک‌ها شدند را خروجی بدهد. برای اینکار باید توجه کنیم که با توجه به متن مقاله نرخ تغییر فریم‌های 60 Hz و به طور دقیق با توجه به فایل‌های لاگ 59.7213 Hz می‌باشد. همچنین از آنجایی که نرخ نمونه برداری 1 ms می‌باشد می‌توان دید که با استفاده از کد زیر می‌تواند اندیس‌هایی از فریم‌ها که در آن‌ها اسپایک رخ داده است را ببینیم. (از آنجایی که تمایل داریم ابعاد ماتریس 16×16 باشد اندیس‌های کمتر از ۱۶ را نادیده در نظر می‌گیریم)

۱.۴.۲ قطعه کد: یافتن اندیس‌های زمان اسپایک و نادیده گرفتن اندیس‌های کمتر ۱۵

```
1 indices = ceil((events * frame_rate) / 10000);
2 indices = indices(indices>15);
```

سپس می‌توانیم ماتریس‌های 16×16 که سبب اسپایک شدند را تشکیل دهیم. برای راحت‌تر شدن کار در مراحل بعد اولاً ورودی دوم ذکر شده در صورت سوال یعنی `msqlD` را اختیاری در نظر گرفتیم که مقدار دیفالت آن همان فایلی خواهد بود که در پوشه‌ی `Data` قرار دارد. همچنین در مراحل بعد از آنجایی که می‌خواهیم دنباله‌ای تصادفی از اسپایک‌ها تولید کنیم راحت‌تر خواهد بود که فقط اندیس‌ها را به عنوان ورودی به این تابع بدهیم به این منظور یک ورودی `option` تعریف کردیم که اگر مقدار آن `'random'` باشد مقدار `events` در ورودی را به عنوان اندیس‌ها در نظر می‌گیرد و در غیر این صورت همان منطق قبلی را اجرا می‌کند.

۳ بررسی با روش کلاسیک STA

در کد نهایی که برای این بخش ارائه شده است بر روی تمام ۵۴ نورونی که در بخش قبل حذف نشده‌اند عملیات‌ها صورت می‌گیرد. و حاصل تنها ذخیره می‌شود و تصاویر نمودارها export می‌شوند. (بگم دست نزنه یا نگم؟)

۱.۳ بخش اول

برای محاسبه‌ی این بخش تابع Func_FindSTA پیاده سازی شده است که نحوه‌ی عملکرد آن به صورت زیر است.

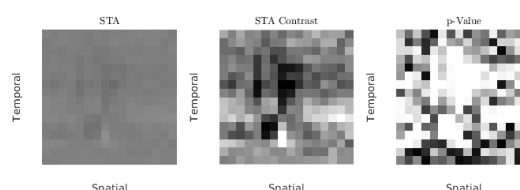
۱.۱.۳ قطعه کد: تابع محاسبه‌ی STA

```

1 function [sta, spike_triggered] = Func_FindSTA(neuron)
2     outs = neuron.outs;
3     spike_triggered = [];
4     for i = 1:length(outs)
5         spike_triggered_trial = Func_StimuliExtraction(outs(i).events
6             );
7         spike_triggered = cat(3, spike_triggered,
8             spike_triggered_trial);
9     end
10    N = length(spike_triggered);
11    sta = (sum(spike_triggered, 3) / N);
12 end

```

که می‌توان دید بنابر تعریف STA پیاده‌سازی شده است. این تابع تمام مقادیر spike_triggered را نیز خروجی می‌دهد چرا که در مراحل بعد به آن نیاز خواهیم داشت. برای همان مثالی که در صورت پروژه آمده است خروجی زیر را می‌گیریم:



شکل ۴: خروجی برای نورون 000601.c05

۲.۳ بخش دوم

خروجی این بخش در بخش اول نمایش داده شد با این حال ذکر می‌کنیم که برای محاسبه p -value به دست آمده از t -test از تابع آماده `ttest` در متلب استفاده کردیم.

۱.۲.۳ قطعه کد: محاسبه t -test

```
1 [~, p] = ttest(permute(spike_triggered,[3 1 2]));
```

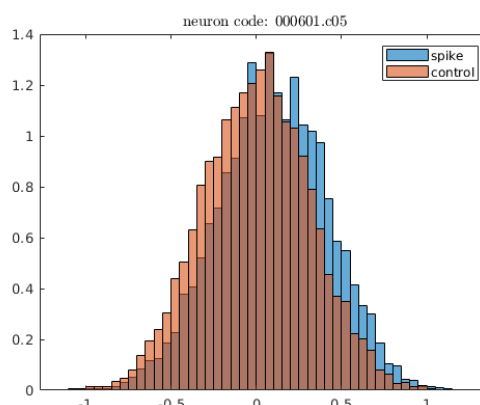
۳.۳ بخش سوم

برای محاسبه $correlation$ از آنجایی که یکی از ماتریس‌هایی که در اختیار ماست ۳ بعدی است محاسبه $correlation$ را به صورت دستی انجام دادیم و تابع `Func_Correlation` را پیاده‌سازی کردیم، که اساس کار آن ضرب درایه به درایه موجود در متلب می‌باشد. همچنین برای تولید دنباله‌ای تصادفی از اسپایک‌ها یا همان ماتریس کنترل به صورت زیر عمل کردیم و در اینجا اضافه کردن پارامتر ورودی اضافه به تابع کارمان را ساده‌تر می‌کند. و تنها کافیت دنباله‌ای تصادفی از شماره‌ی فریم‌ها ایجاد کنیم.

۱.۳.۳ قطعه کد: ایجاد دنباله‌ی تصادفی از تحریک‌ها

```
1 spike_sample = randi([16 32767], 1, N);
2 stim_control = Func_StimuliExtraction(spike_sample, 'random');
```

خروجی هریک از ماتریس‌های کنترل و `spike-triggered` در فضای جدید برای این نوروں به صورت زیر است.



شکل ۵: هیستوگرام $correlation$ ماتریس‌های کنترل و `spike-triggered` با STA

۴.۳ بخش چهارم

برای محاسبه p -value در اینجا نیز از تابع آماده $ttest2$ موجود در متلب استفاده می‌کنیم. خروجی آن به صورت زیر است.

$p\text{-value} = 0.000000$, null-hypothesis rejected, means are different.

نتیجه‌ی آزمون t -test این است که توزیع $p(s|r)$ و $p(s)$ در فضای مربوط به STA دارای میانگین‌های متفاوتی می‌باشند. با این حال با توجه به شکل ۶ برایمان روشن است که این دو توزیع علارغم اینکه میانگین‌های متفاوتی دارند اما به دلیل اینکه سطح مشترک زیر نمودار آنها بسیار زیاد است نمی‌توان تفاوت معناداری بین آنها قائل شد.

به عبارت دیگر نتیجه‌ی آزمون می‌گوید که توزیع $p(s|r)$ و $p(s)$ تفاوت میانگین قابل توجهی دارند که این برخلاف انتظار ما نمی‌باشد چرا که $p(s|r)$ از تصویر بردارهای منجر به اسپایک بر روی STA حاصل می‌شود که واضح است که باید میانگین مثبتی داشته باشد و از طرفی روشن است که با انتخاب تحریک‌های رندم و تصویر کردن روی STA میانگینی نزدیک به صفر خواهیم داشت. با این حال همانطور که از شکل مشخص است تفاوت معنادار در میانگین نمی‌تواند تفاوت معنادار در توزیع‌ها را نشان بدهد و همانطور که از متن مقاله به یاد داریم در آنالیزهایی که بر پایه‌ی correlation طراحی می‌شوند تغییر در توزیع‌ها را با تغییر در گشتاور درجه‌ی دوم آنها یا همان واریانس می‌سنجیم.

۵.۳ بخش پنجم

به طور کلی میدانیم که خطای تخمین ما از رابطه‌ی زیر محاسبه می‌شود:

$$P(\text{Error}) = P(\text{Error}|s = 1)P(r = 1) + P(\text{Error}|r = 0)P(r = 0)$$

از طرفی می‌دانیم که با توجه به اینکه تحریک‌های اولیه کاملاً تصادفی بودند $P(s = 1)$ همان نرخ اسپایک زدن هر نورون در همین آزمایش می‌باشد. ما قبلاً نرخ spike-count rate را محاسبه کرده بودیم و از توضیحات درس به یاد داریم که:

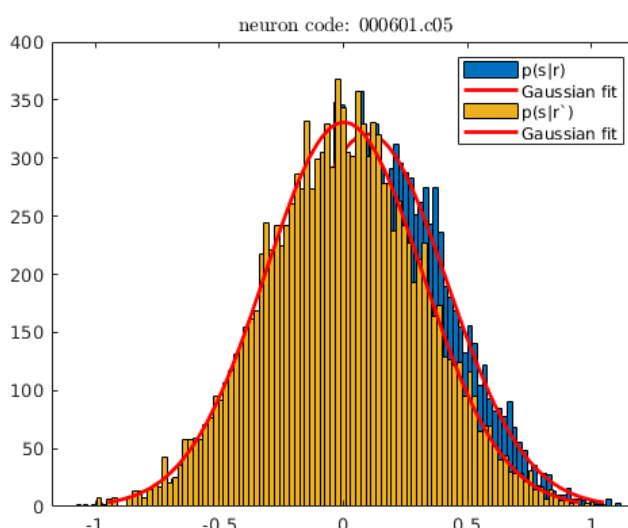
$$P(r = 1) = \frac{\langle \text{count} \rangle}{\Delta t}$$

که در اینجا Δt همان نرخ تغییر فریم‌ها می‌باشد. حال با توجه به نوع تخمینمان که تنها به یک پارامتر θ وابسته می‌باشد مقدار خطا را باز نویسی می‌کنیم:

$$P(\text{Error}) = P(X > \theta | r = 0)P(r = 0) + P(X < \theta | r = 1)P(r = 1)$$

توزیع $P(X|r=1)$ همان توزیع تصویر spike-triggered ها بر روی STA می باشد که طبق فرض مسئله آن را نرمال در نظر می گیریم. از طرف دیگر با توجه به sparse بودن نورون ها و احتمال پایین اسپایک زدن آنها (برای اکثر نورون ها با توجه به شکل ۳ در حدود 10^{-2} می باشند) می توانیم توزیعی که برای تصویر control محاسبه کردیم را توزیع $P(X|r=0)$ در نظر بگیریم و بنابراین می توانیم $P(\text{Error})$ را محاسبه کنیم و مشتق آن نسبت θ را برابر با صفر قرار دهیم تا بهترین ترشهود را محاسبه کنیم.

برای محاسبه این مقادیر در متلب ابتدا یک فیت گاوسی بر روی هیستوگرام ها ایجاد می کنیم و سپس μ و σ را محاسبه می کنیم و آن ها را به همراه احتمال اسپایک زدن نورون به تابع نوشته شده ی Func_Accuracy می دهیم که ترشهود و دقت تخمین ما را بیان می کند.

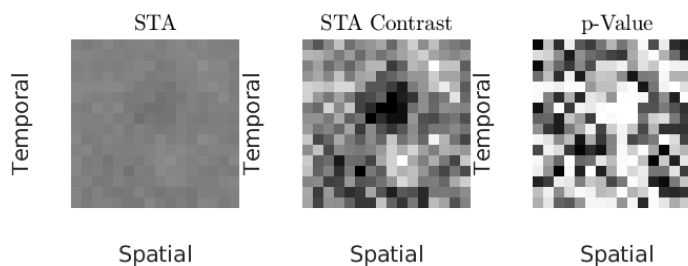
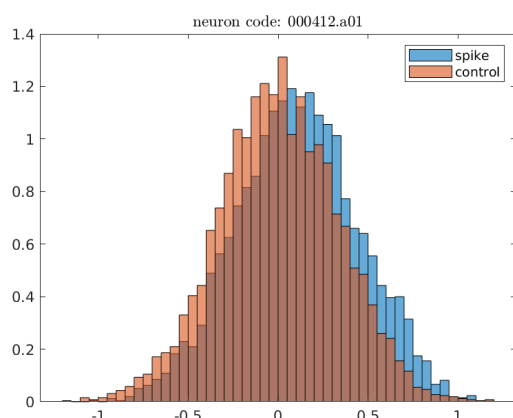


شکل ۶: هیستوگرام ها به همراه فیت گاوسی آن ها

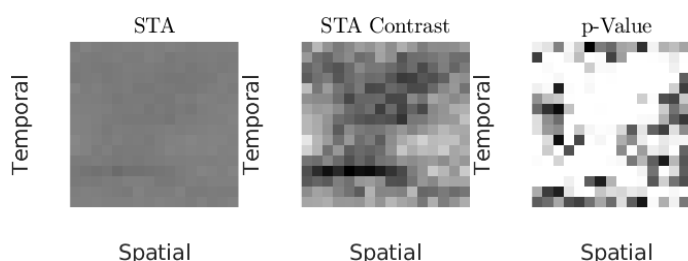
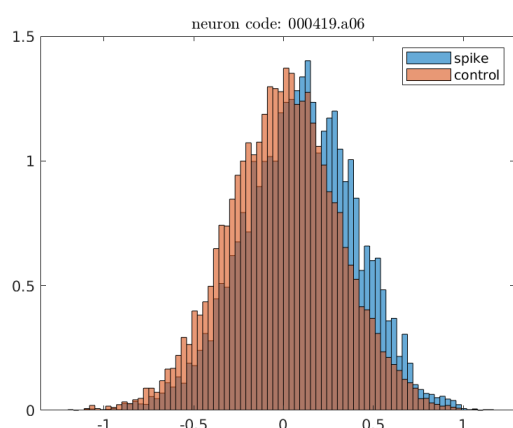
همچنین توجه می کنیم که برای محاسبه ی Func_Accuracy از سایت های محاسبه کننده استفاده کردیم و خروجی آنها را در تابع متلب قرار دادیم.

۶.۳ بخش ششم

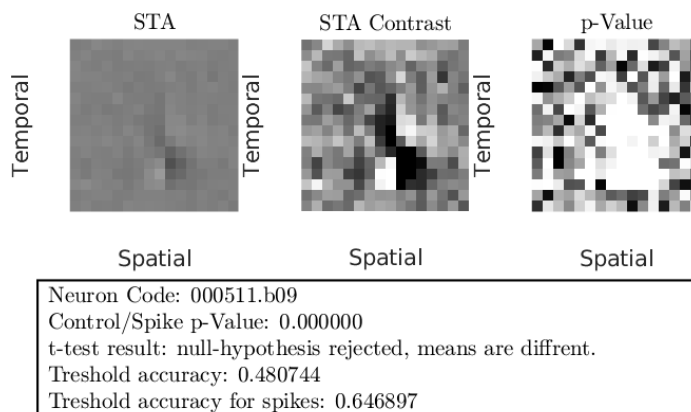
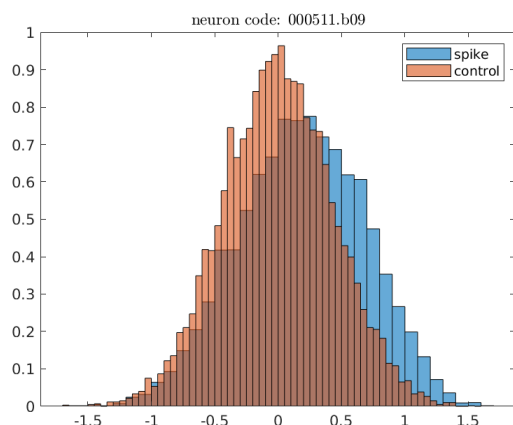
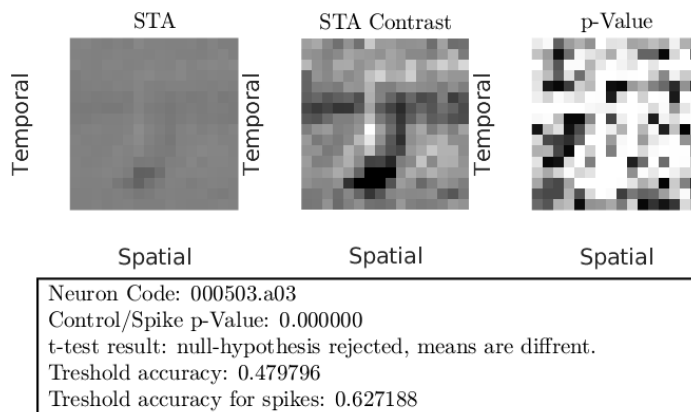
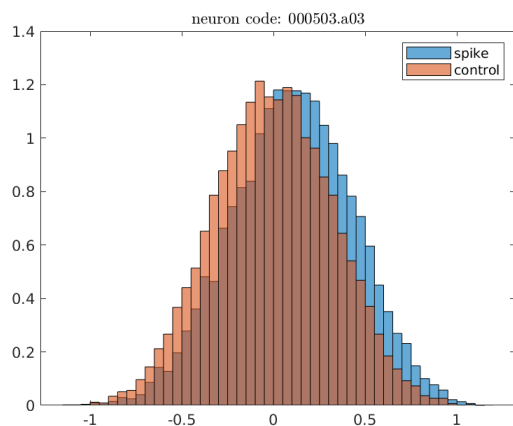
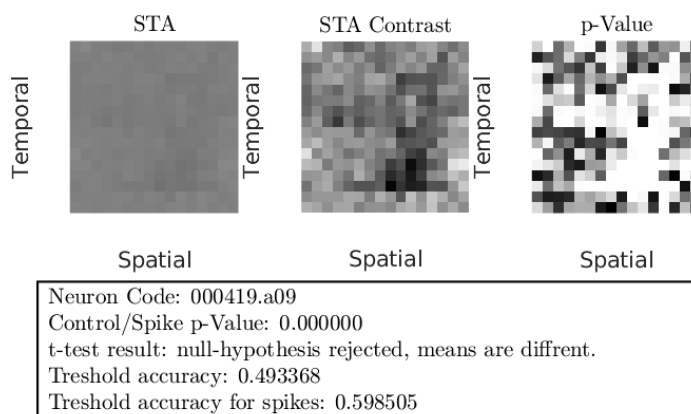
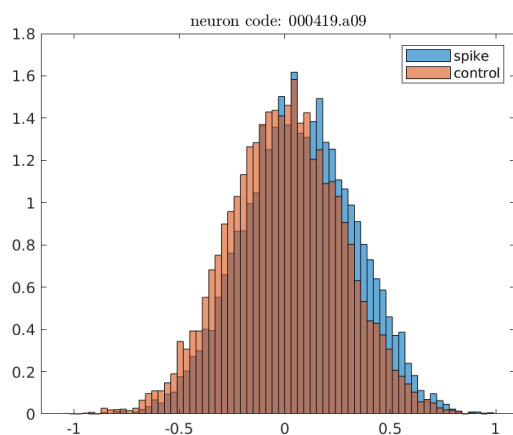
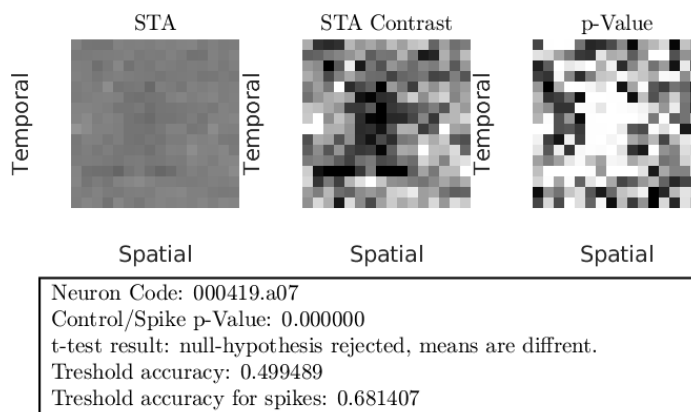
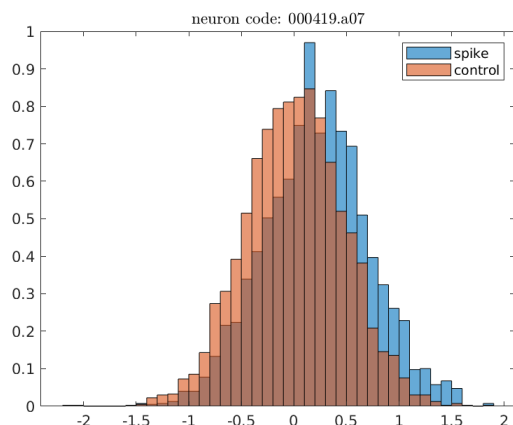
با اجرای مراحل بالا برای تمامی ۵۴ نورون باقی مانده به خروجی‌های زیر دست پیدا می‌کنیم لازم است توجه کنیم که داده‌های کنترل به صورت رندم تولید می‌شوند و بنابراین با اجرای این کدها ممکن است نتایج اندکی متفاوت بگیریم هر چند در کلیت پاسخ تأثیری نخواهند داشت.

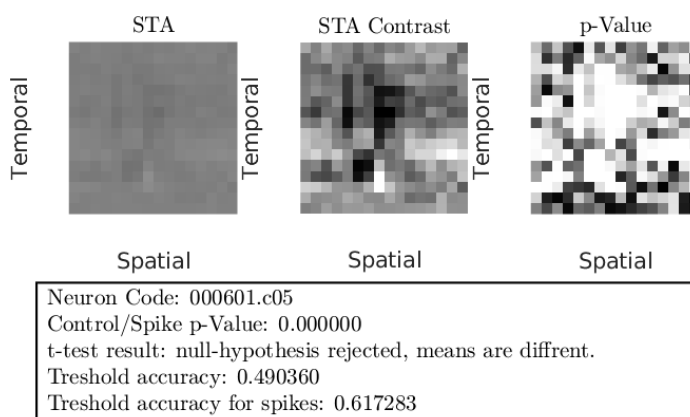
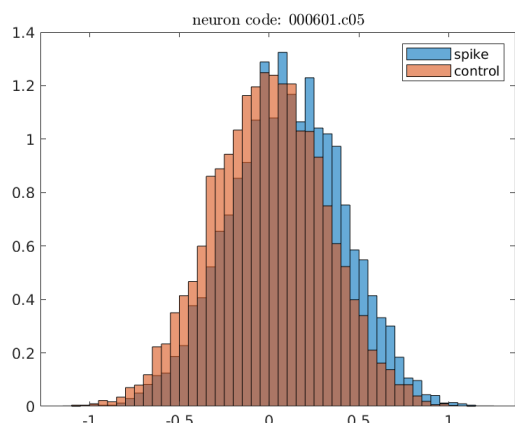
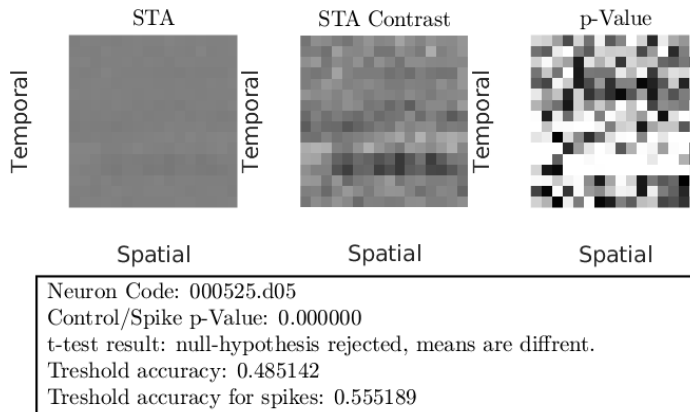
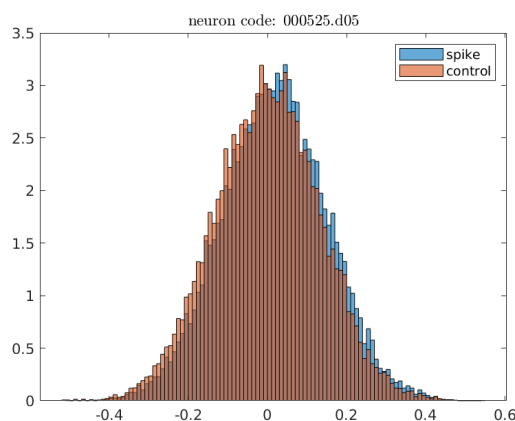
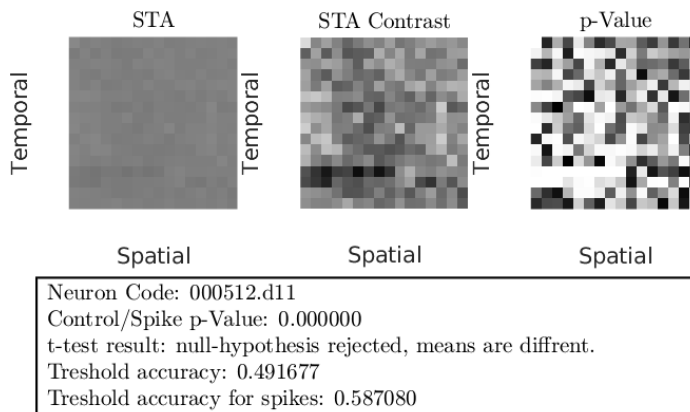
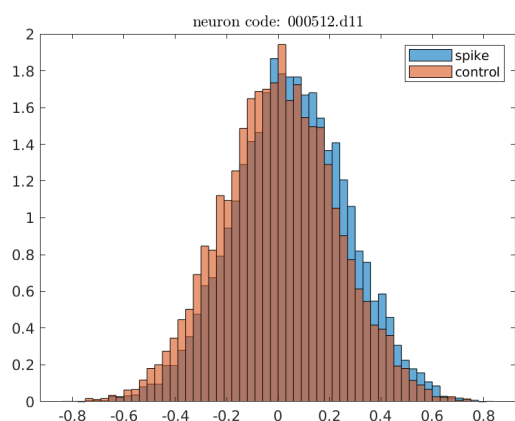
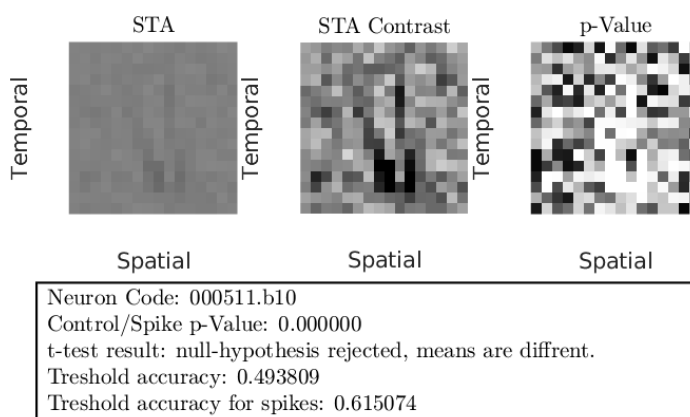
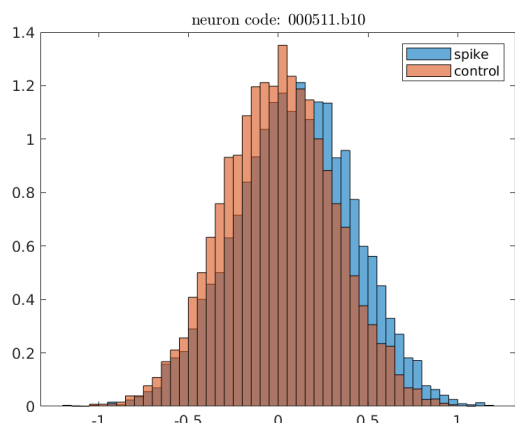


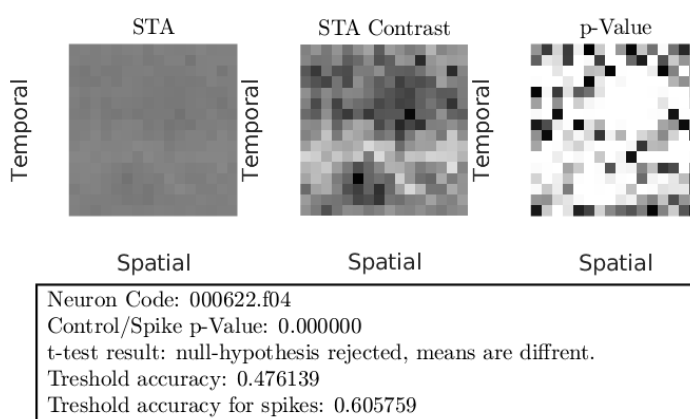
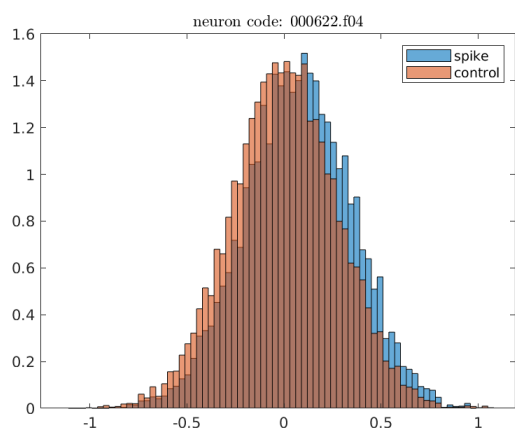
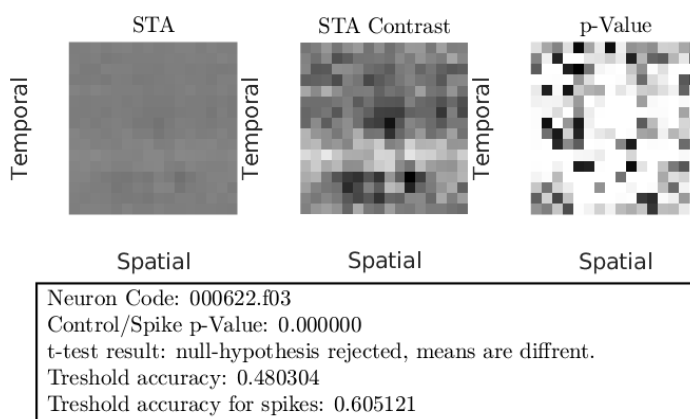
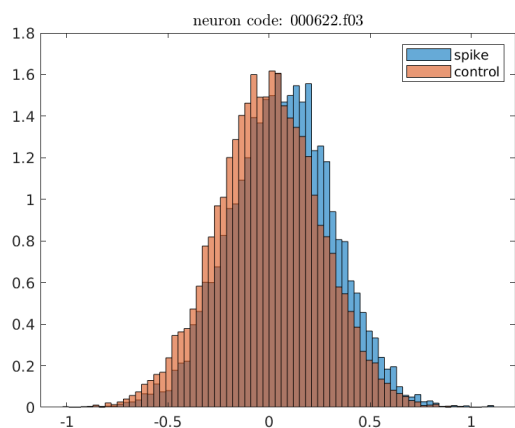
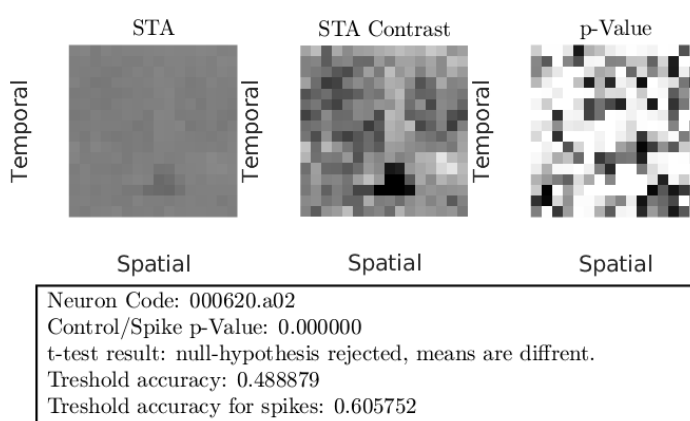
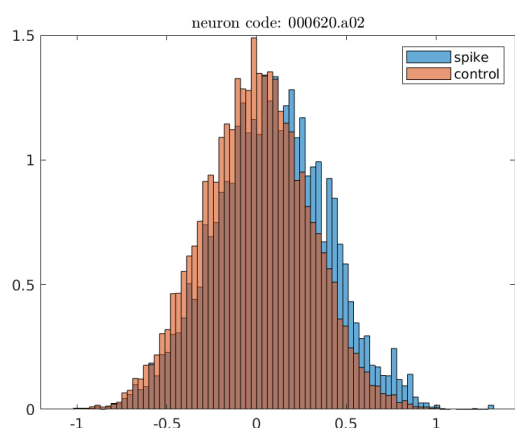
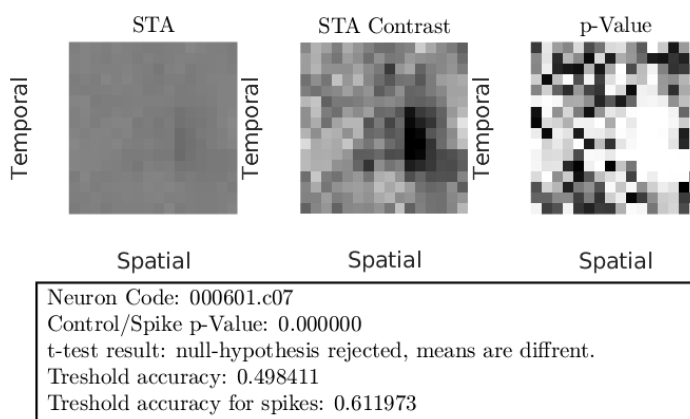
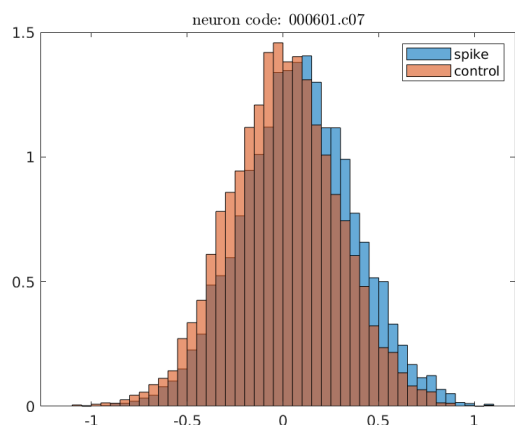
Neuron Code: 000412.a01
Control/Spike p-Value: 0.000000
t-test result: null-hypothesis rejected, means are different.
Threshold accuracy: 0.499714
Threshold accuracy for spikes: 0.626165

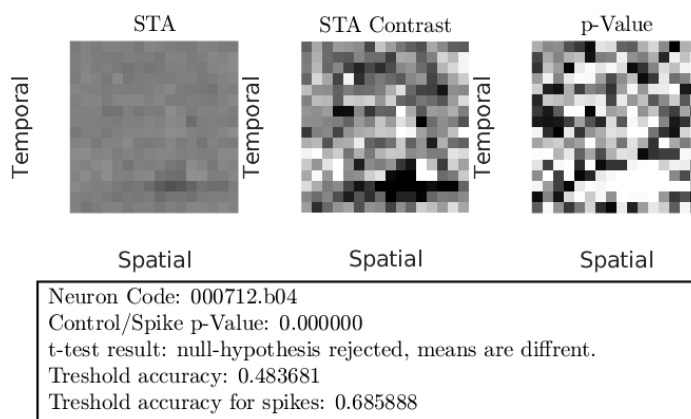
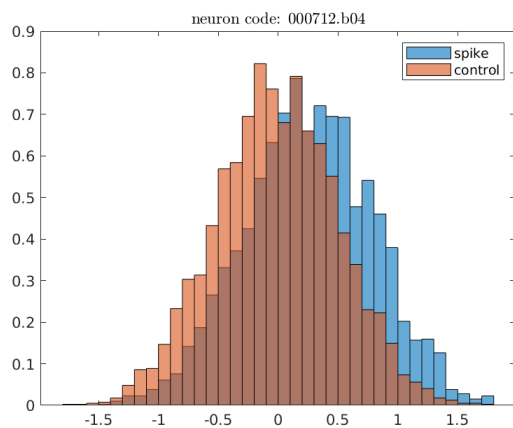
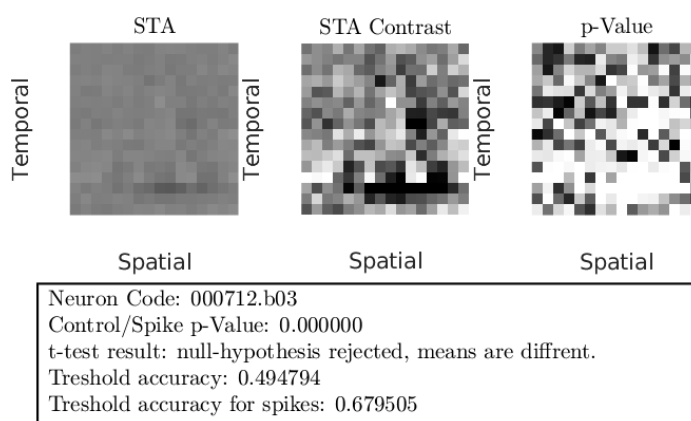
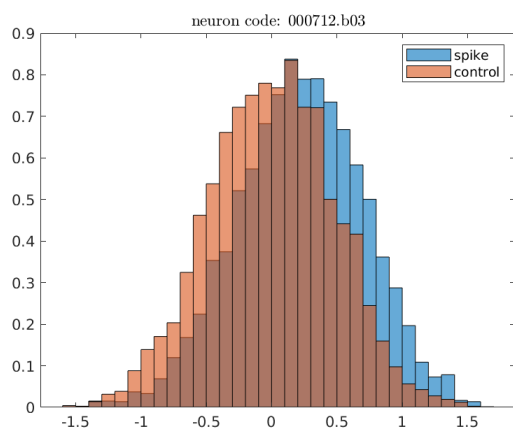
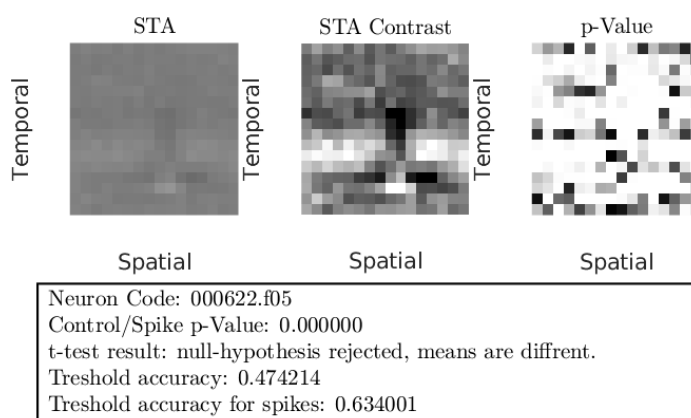
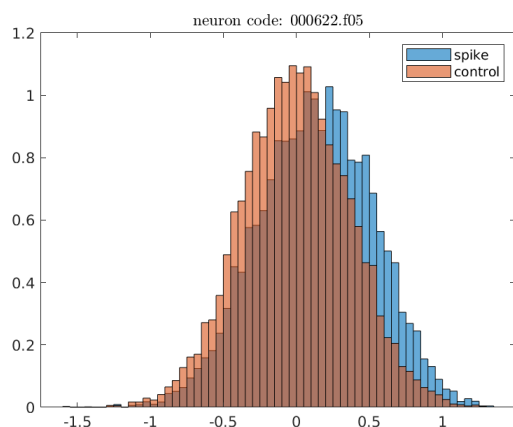


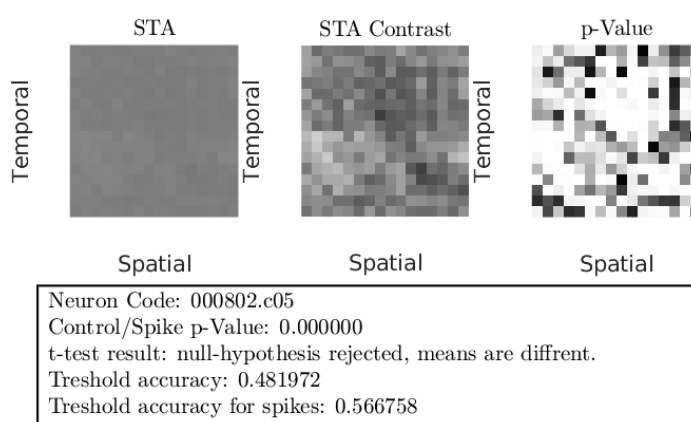
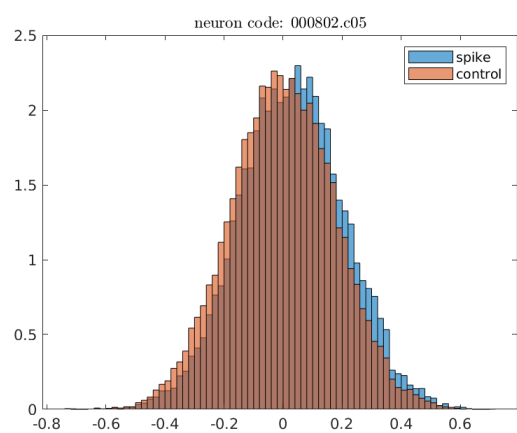
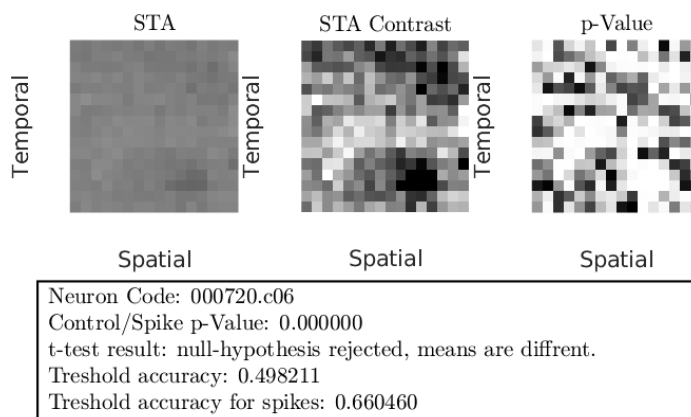
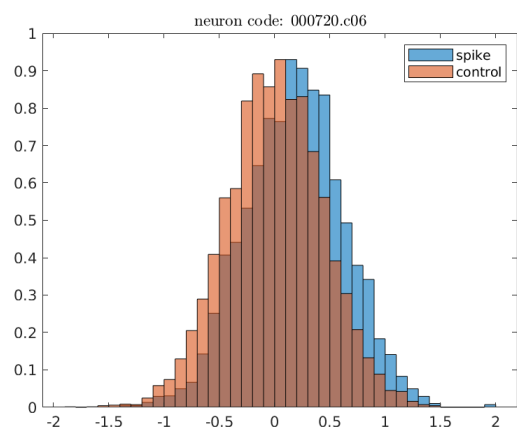
Neuron Code: 000419.a06
Control/Spike p-Value: 0.000000
t-test result: null-hypothesis rejected, means are different.
Threshold accuracy: 0.482390
Threshold accuracy for spikes: 0.629744

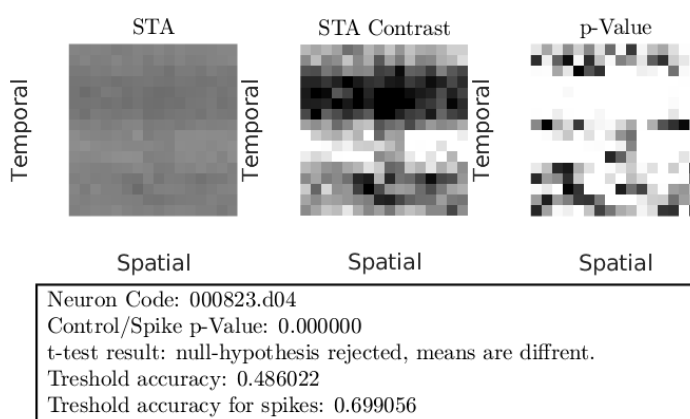
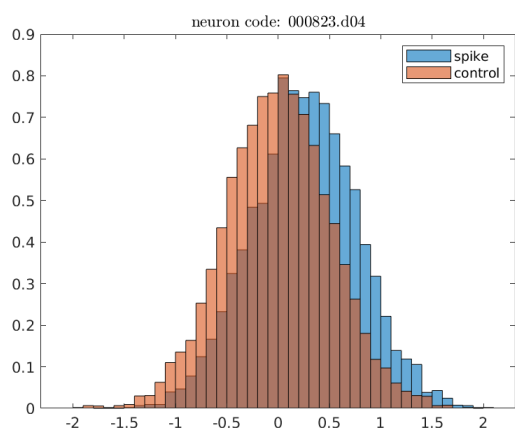
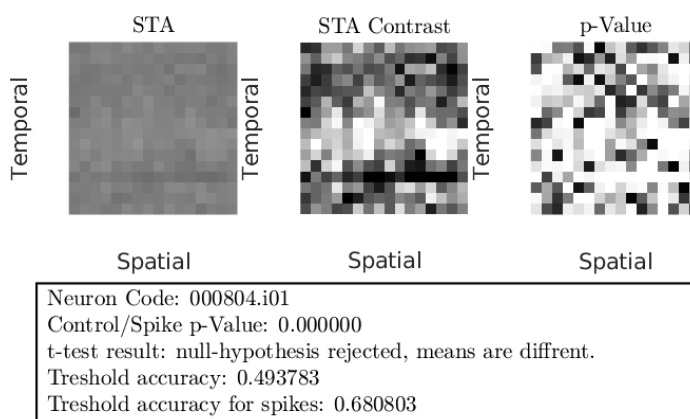
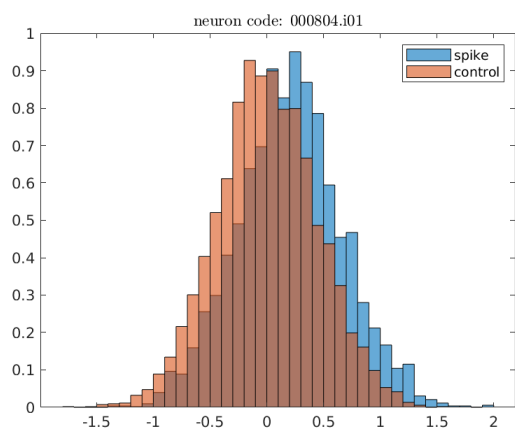
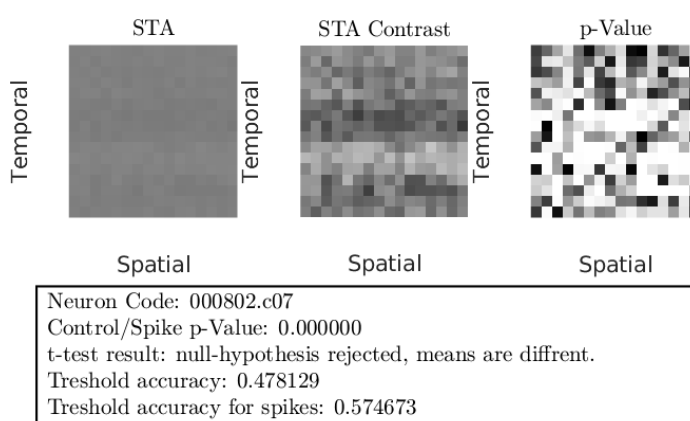
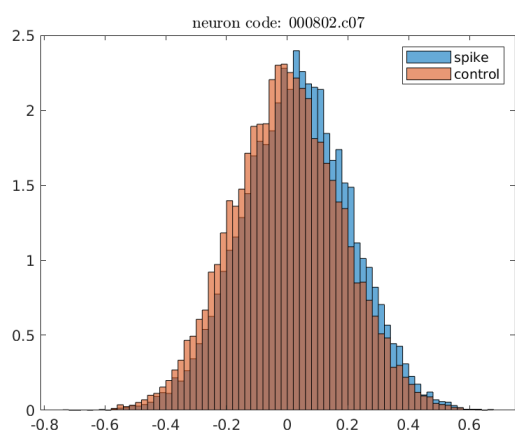
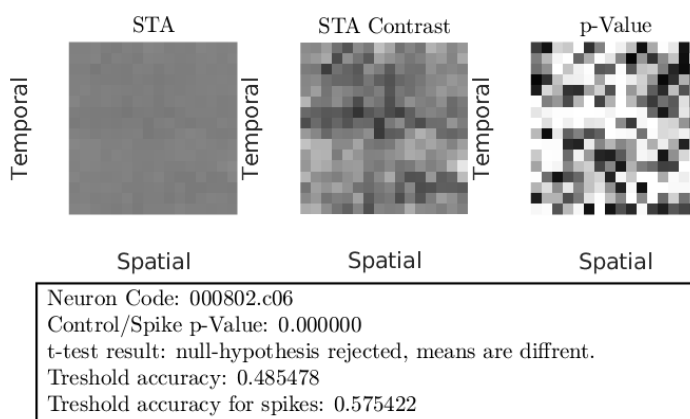
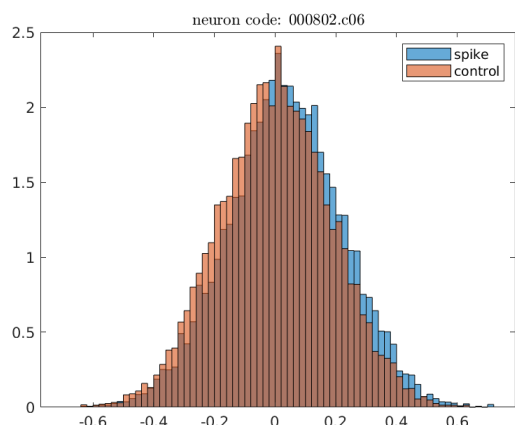


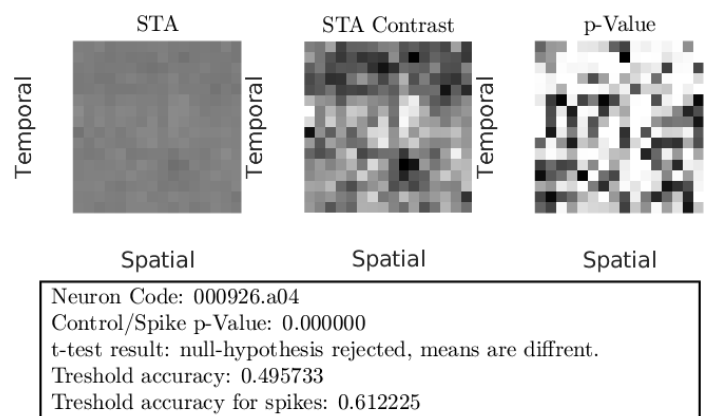
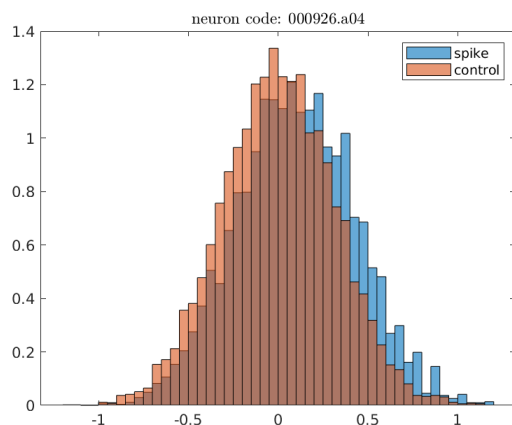
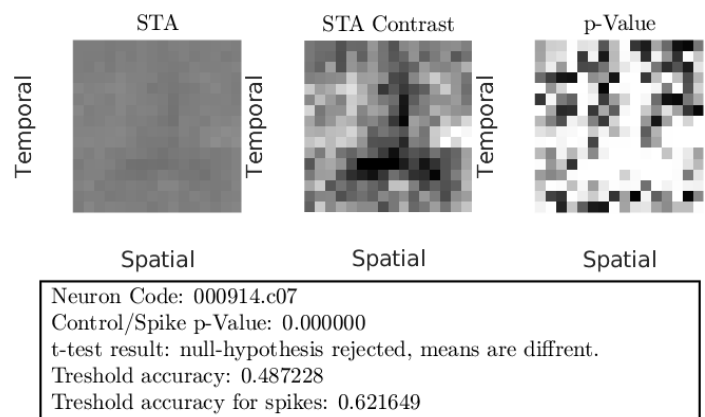
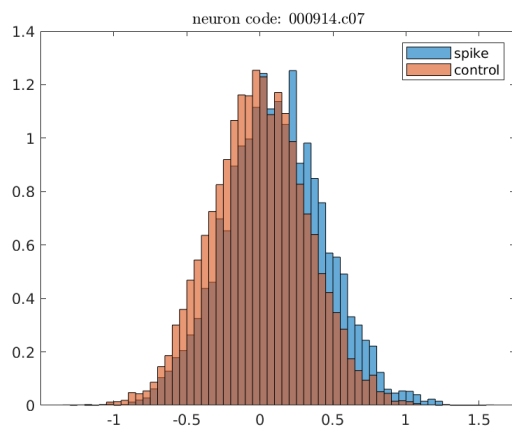
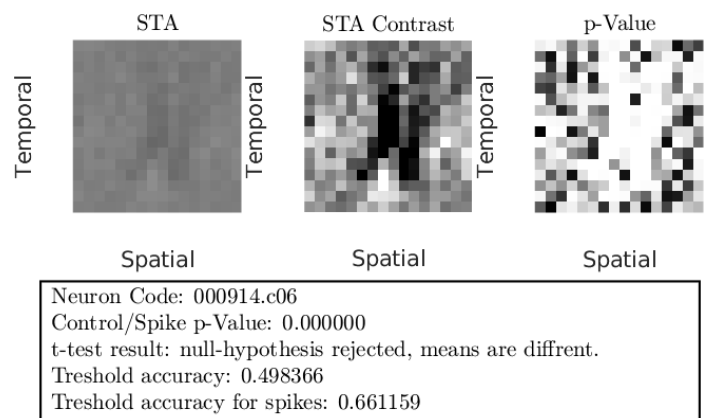
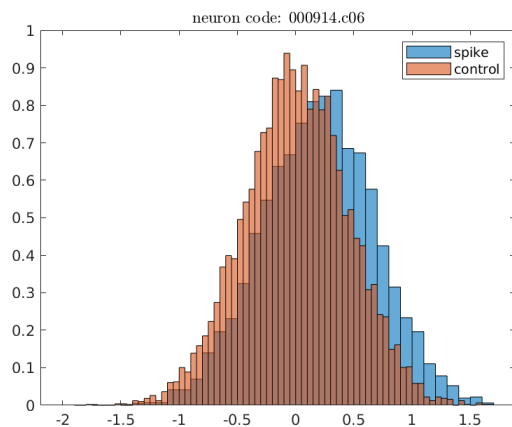
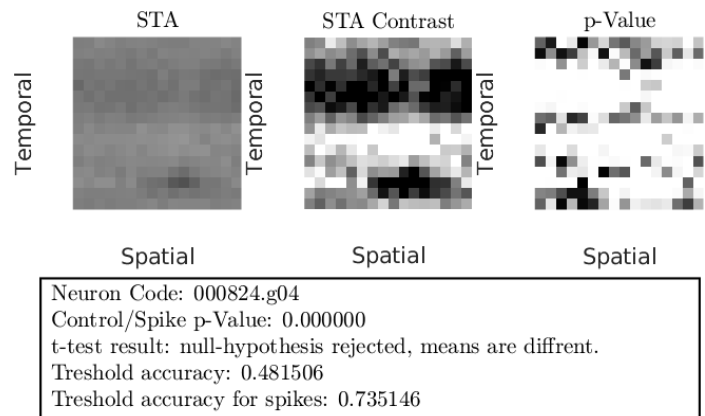
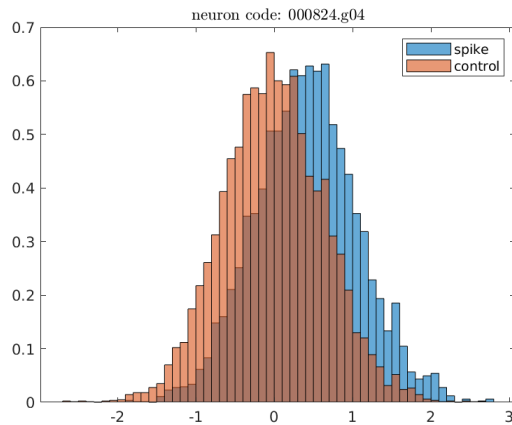


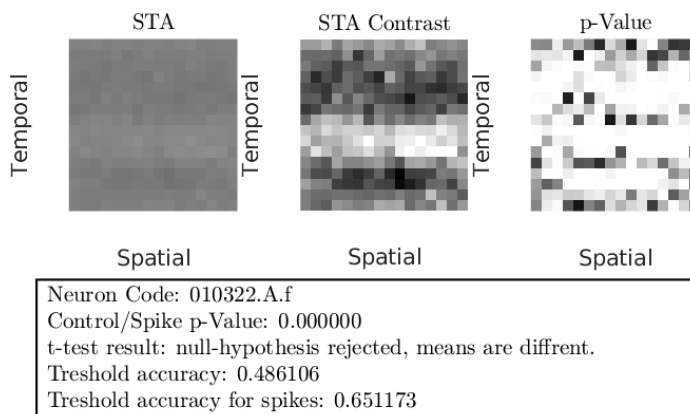
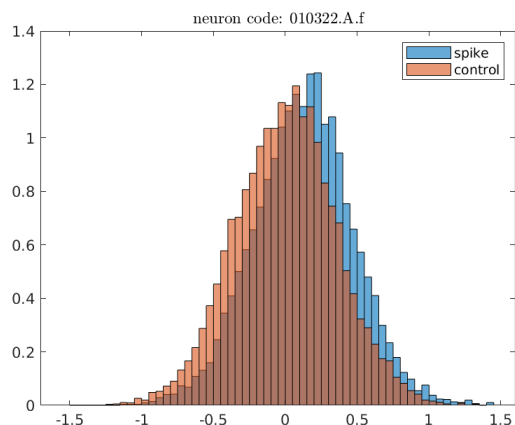
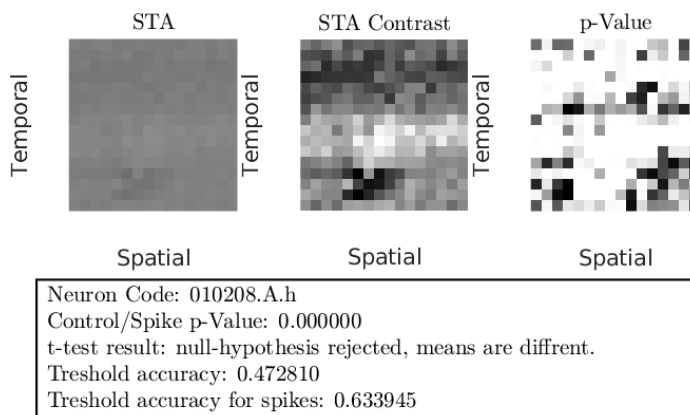
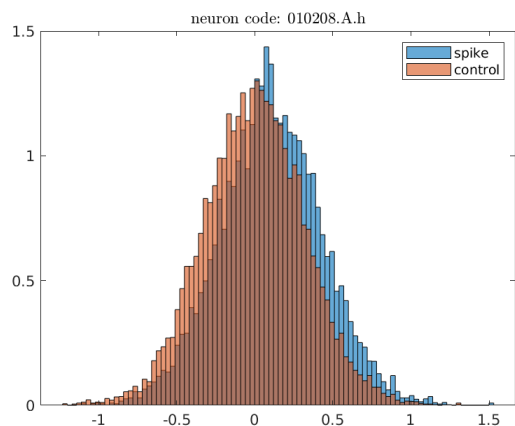
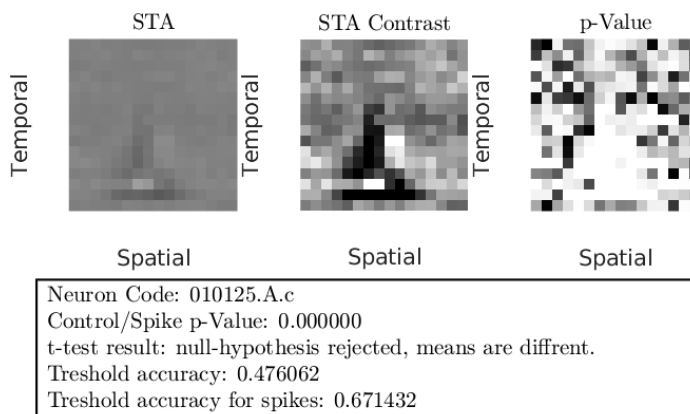
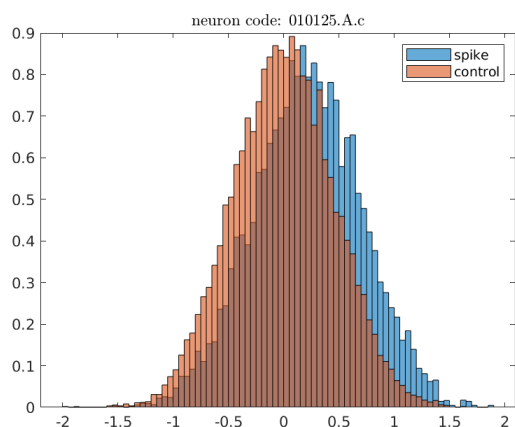
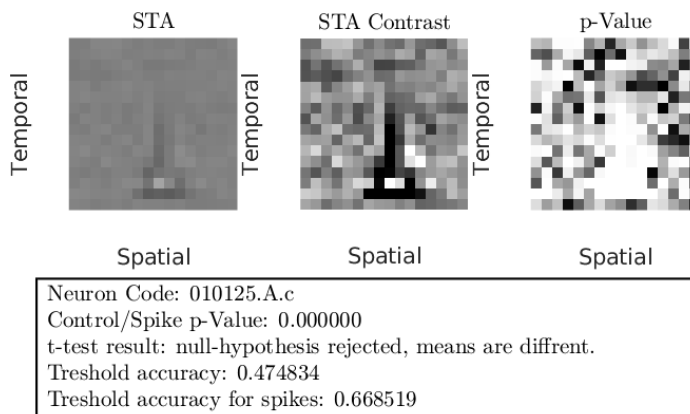
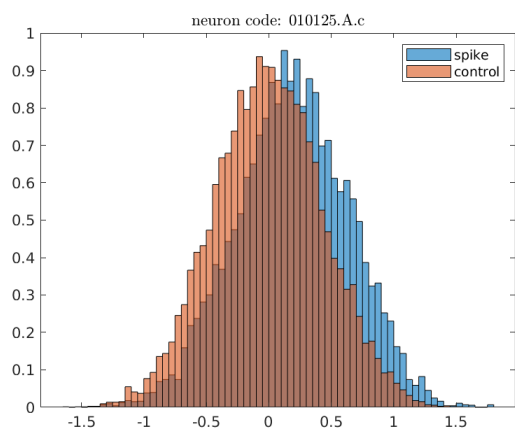


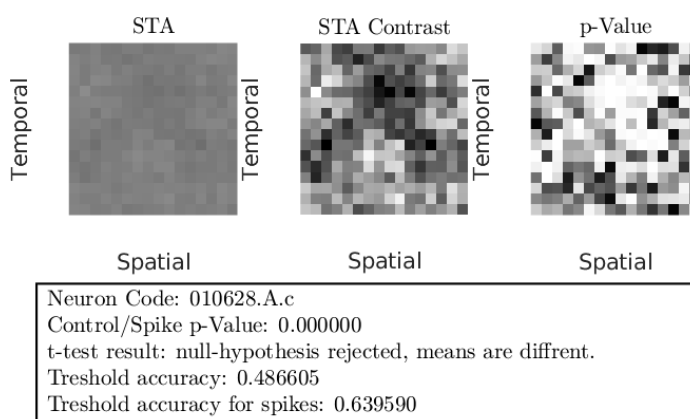
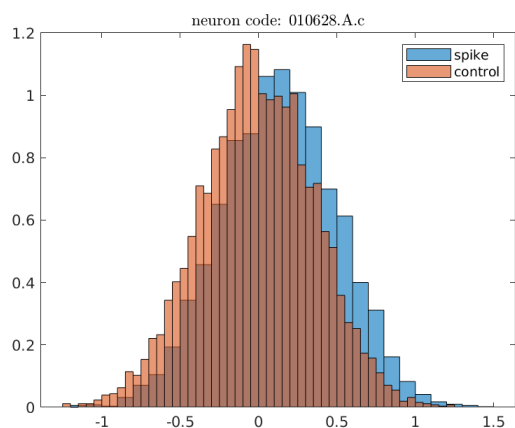
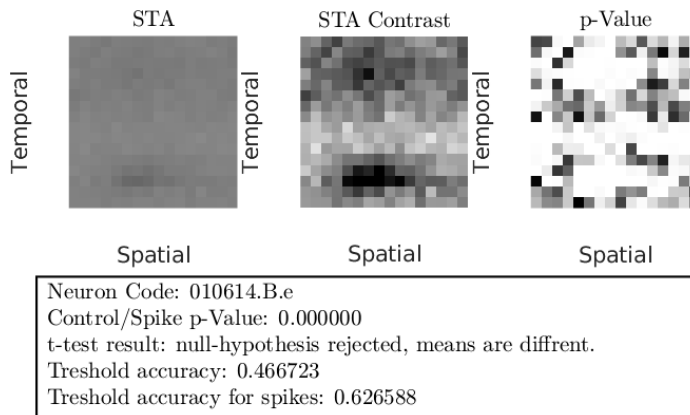
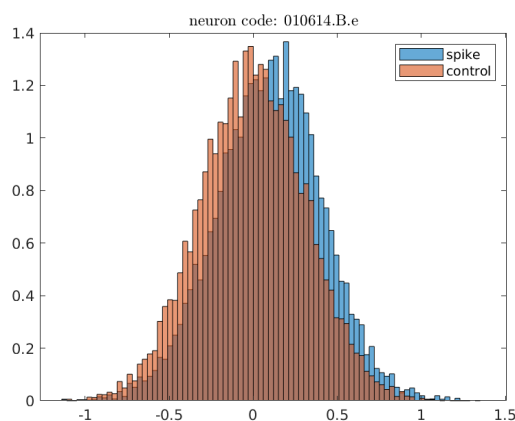
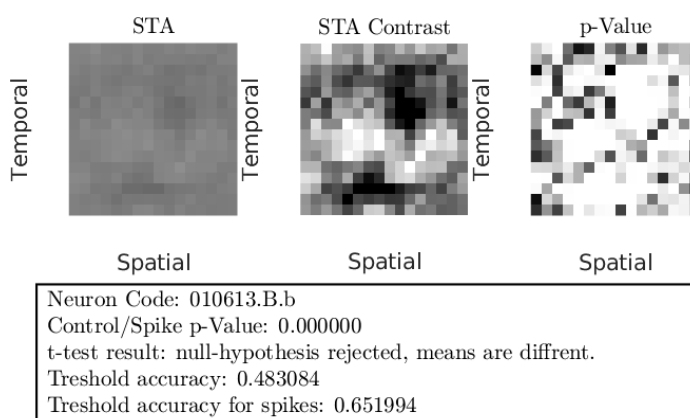
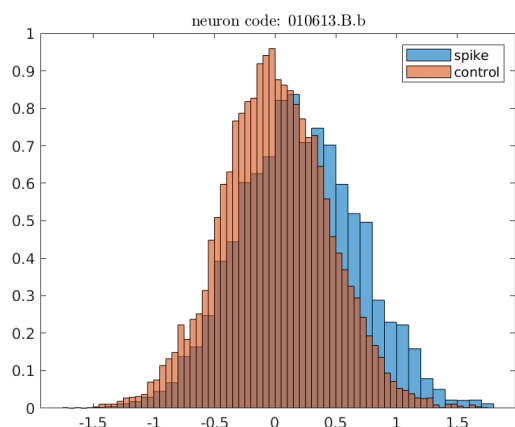
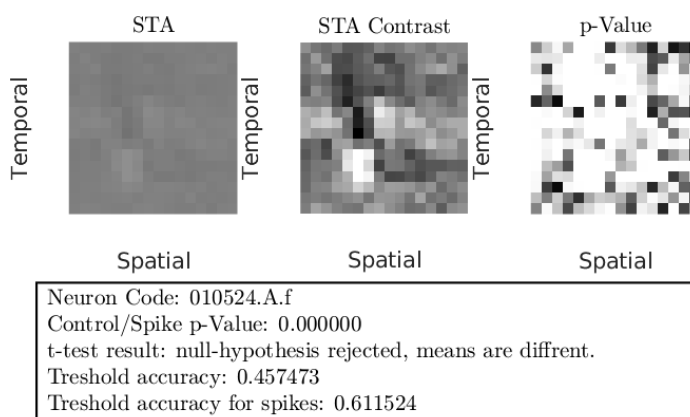
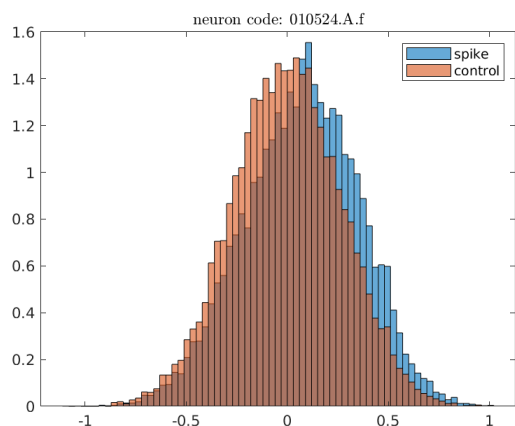


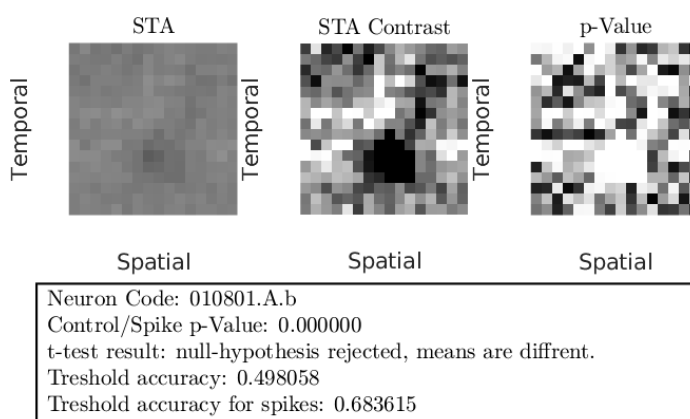
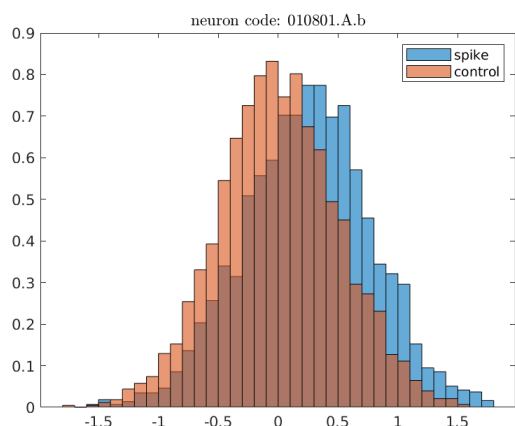
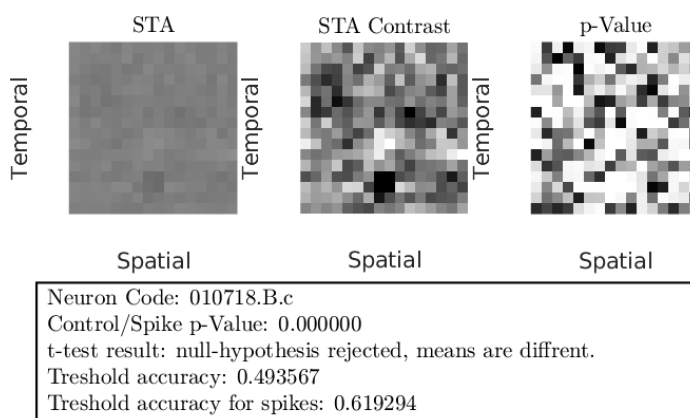
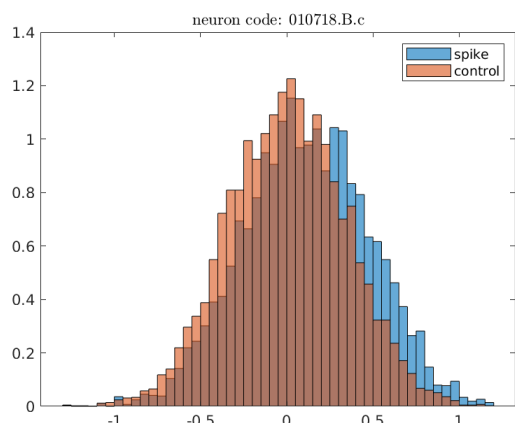
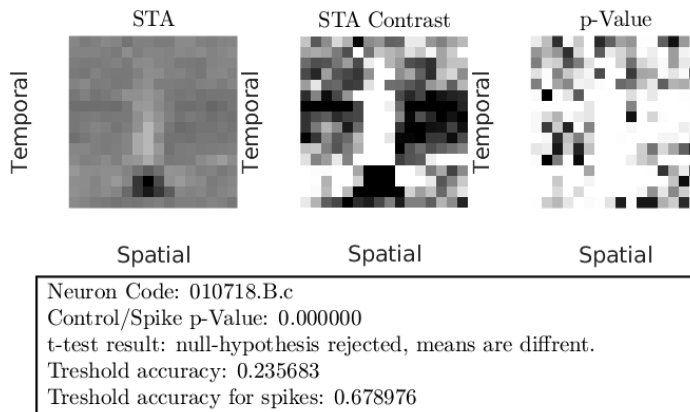
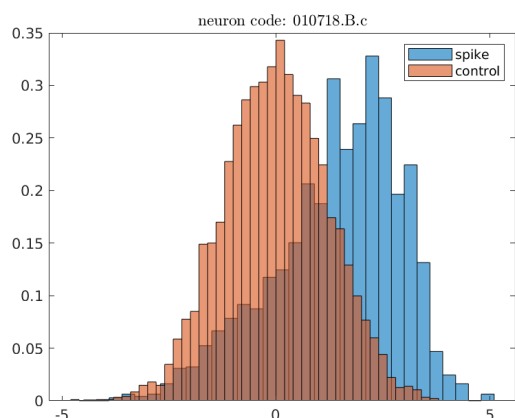
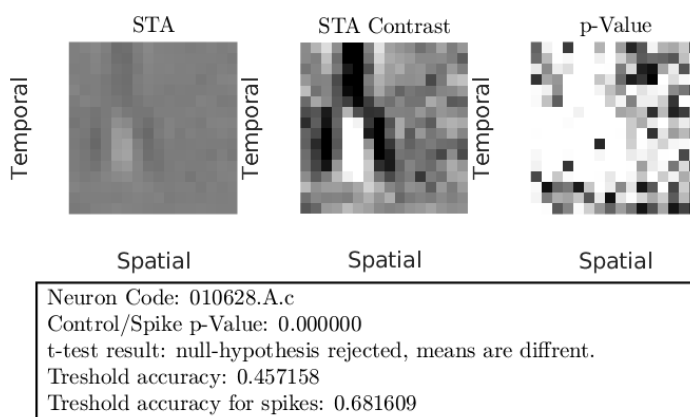
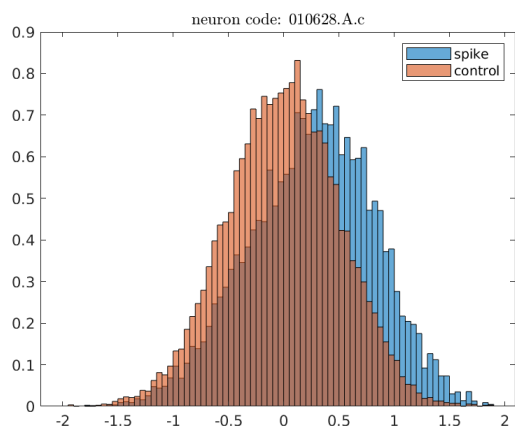


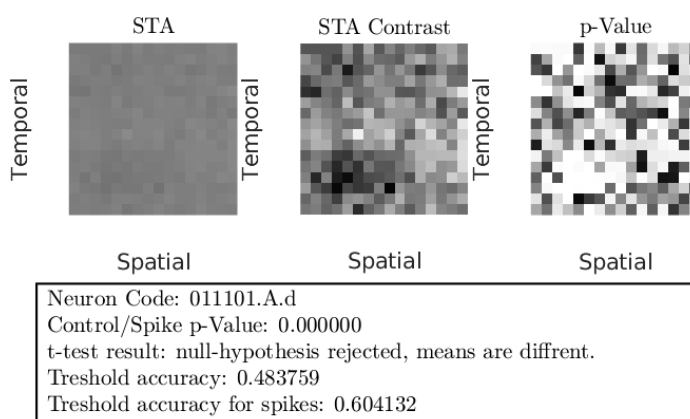
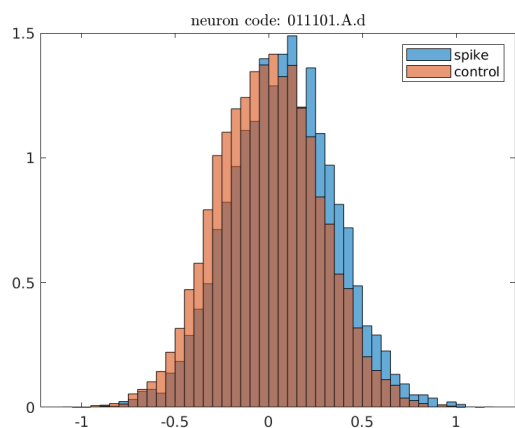
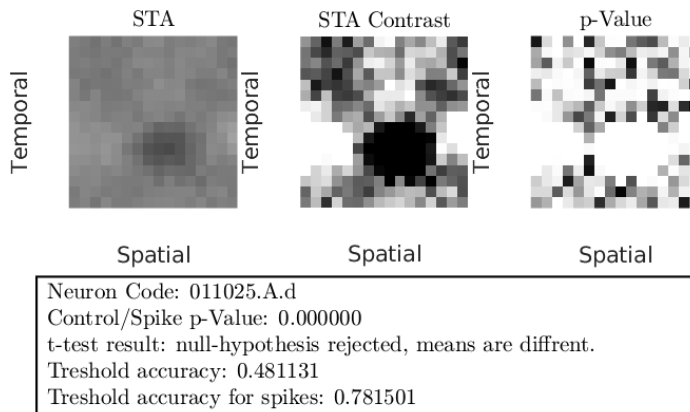
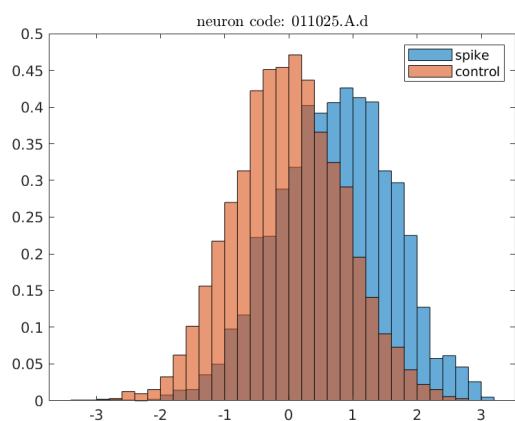
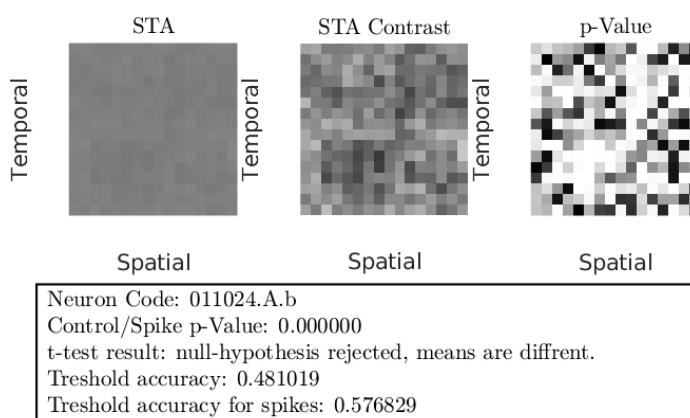
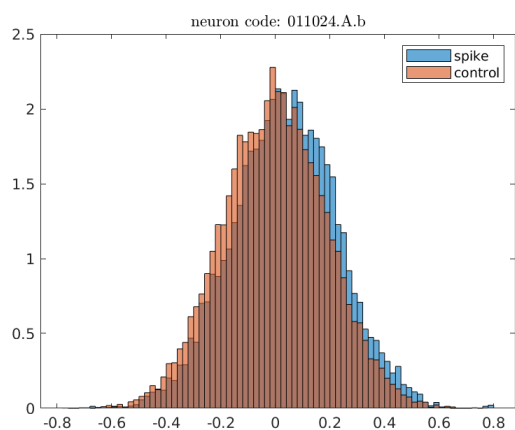
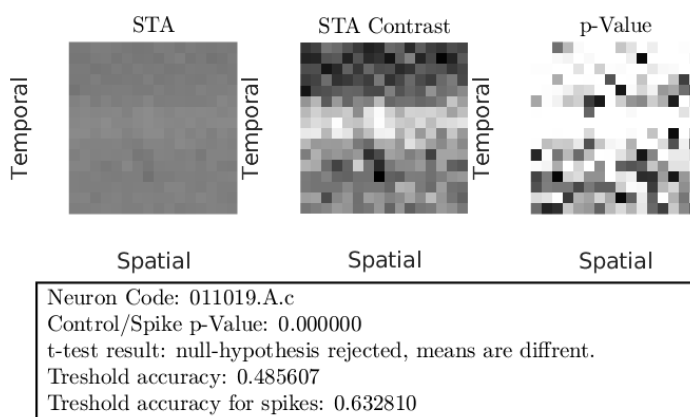
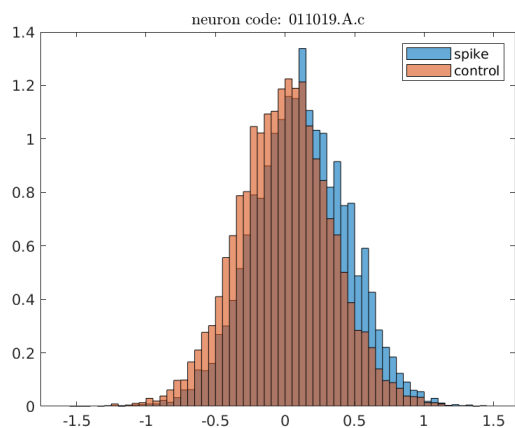


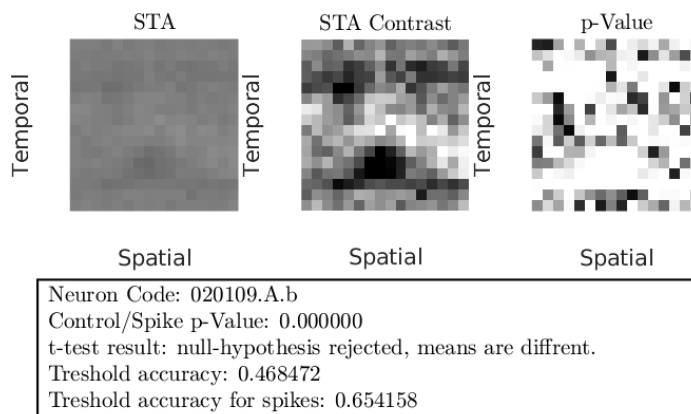
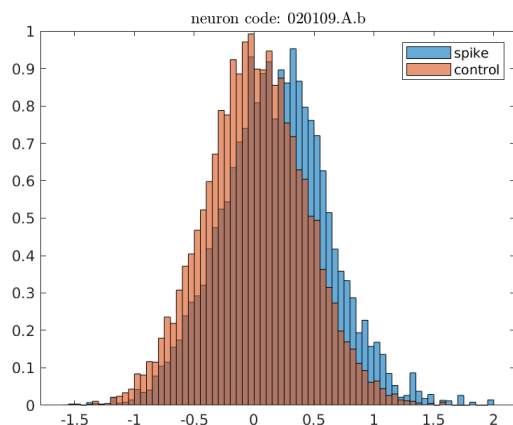
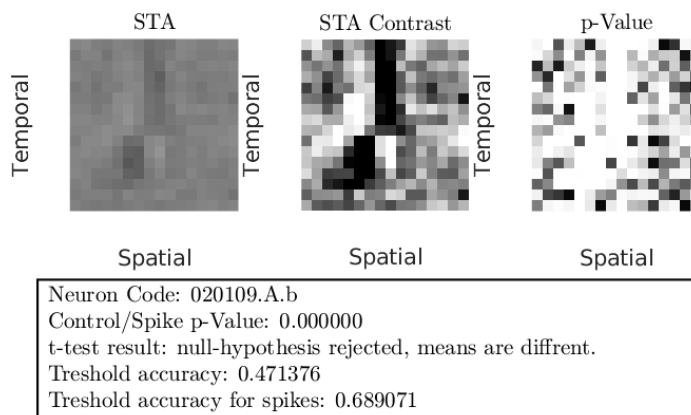
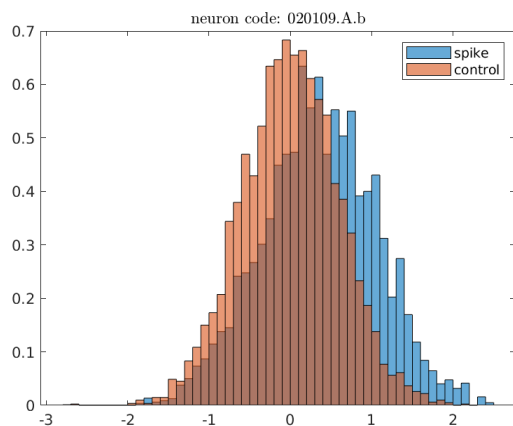
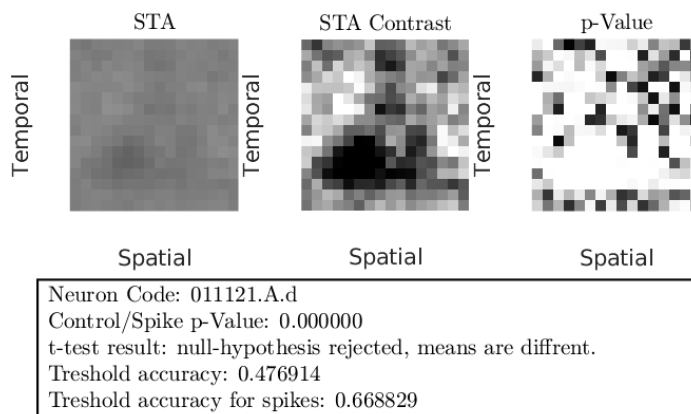
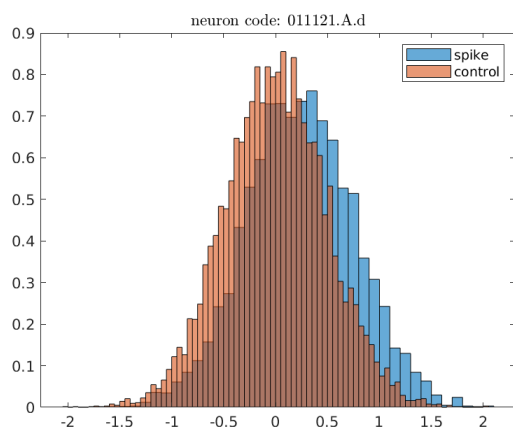
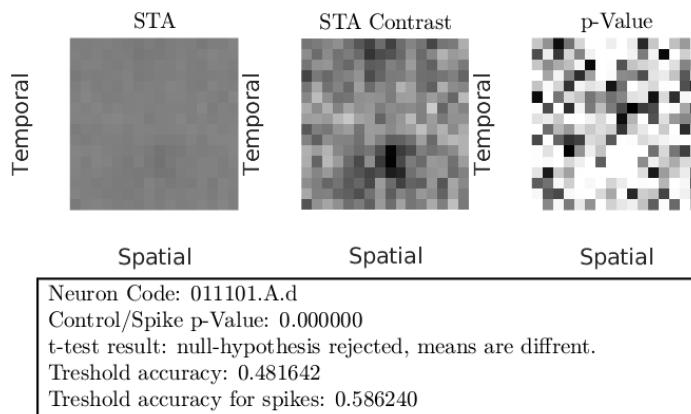
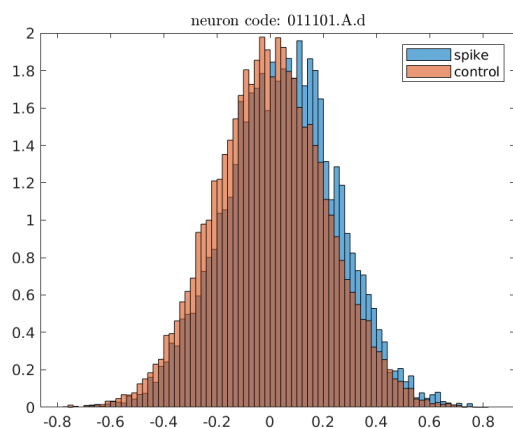


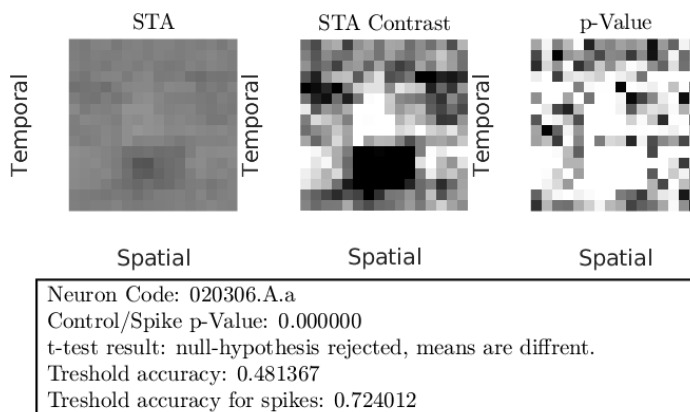
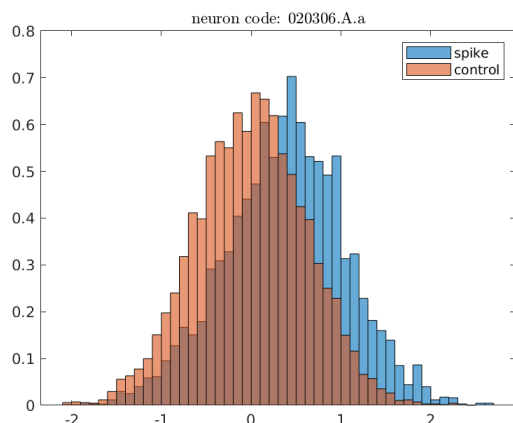
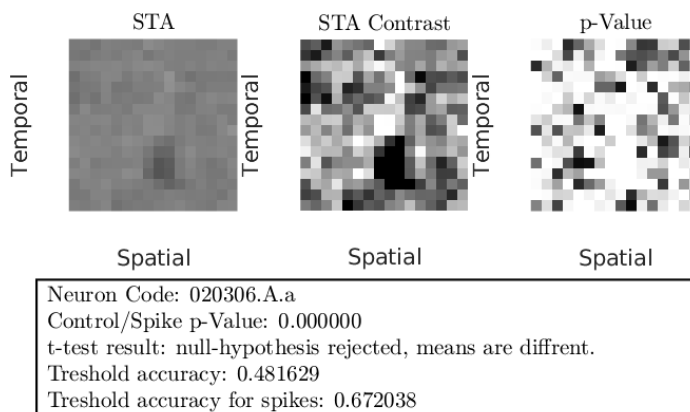
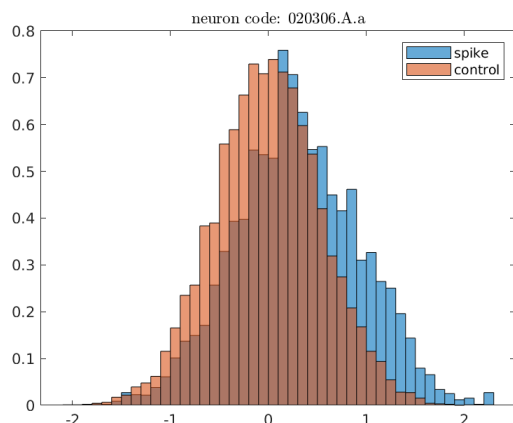
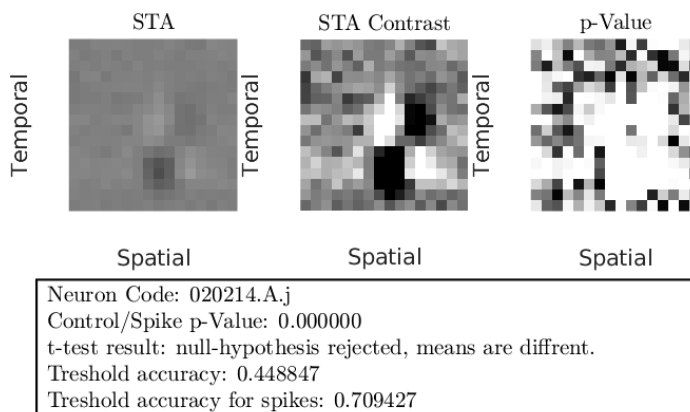
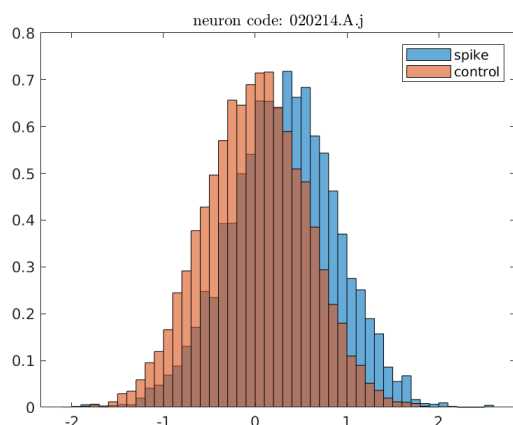
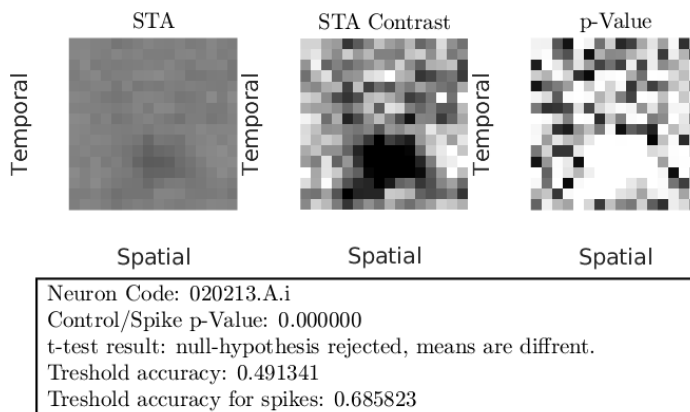
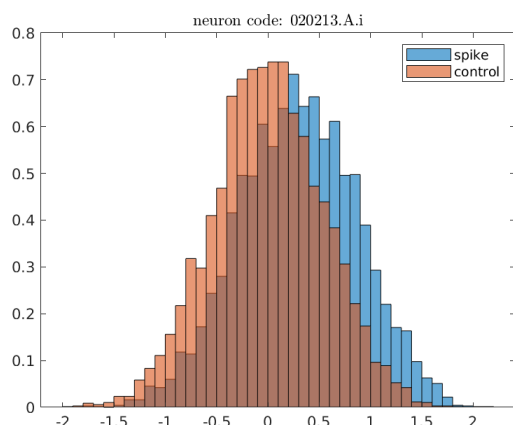


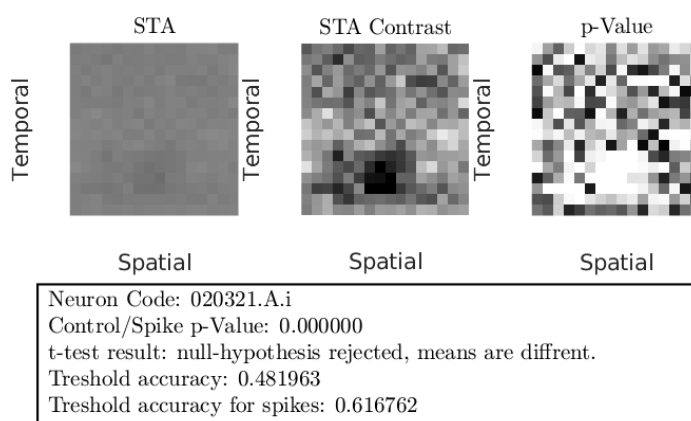
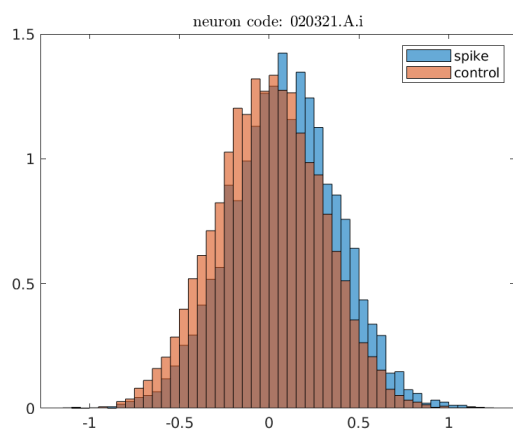
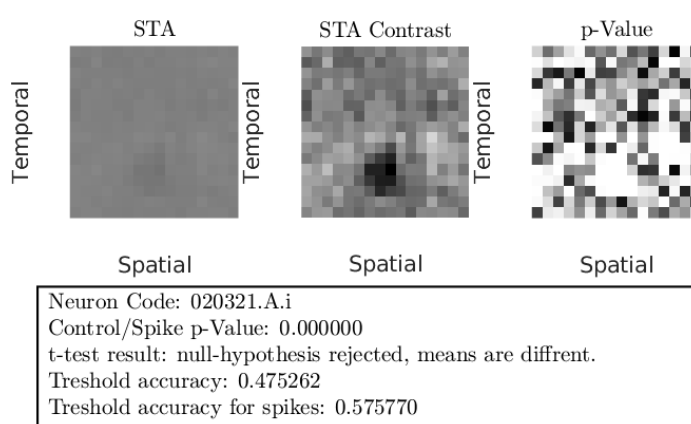
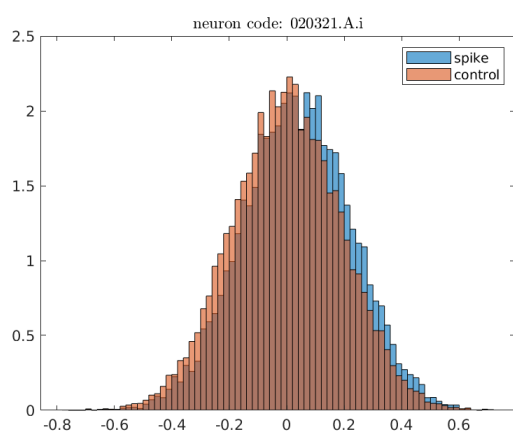
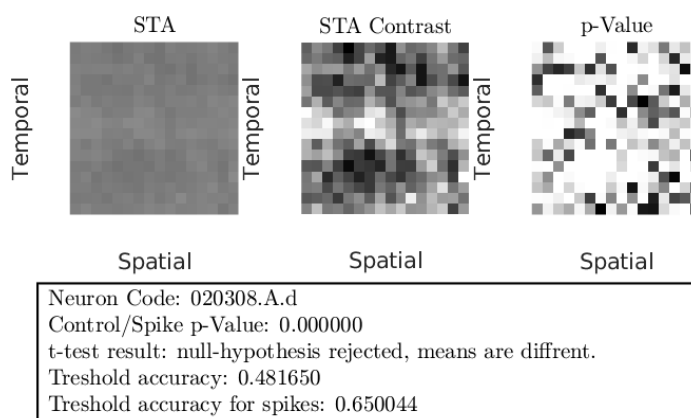
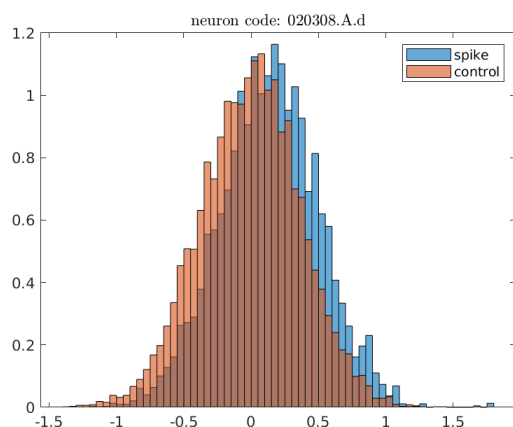






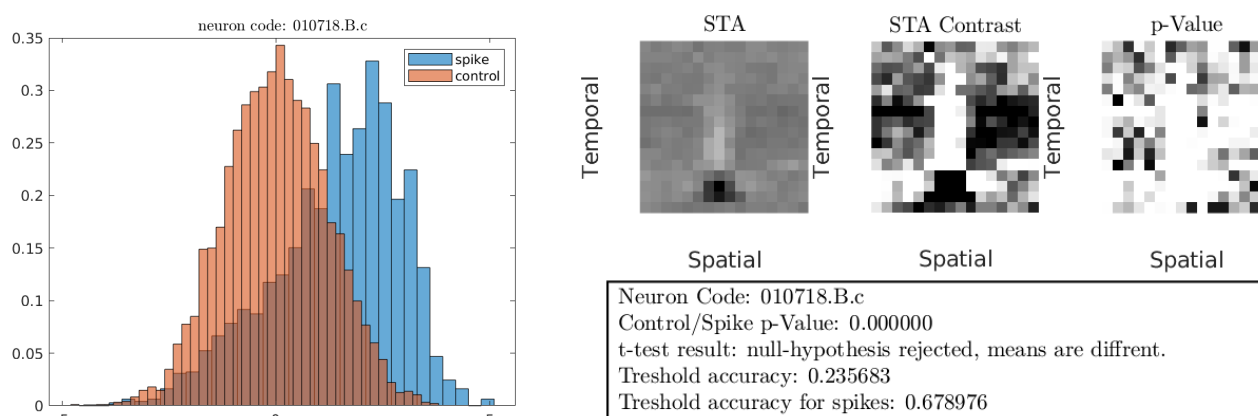




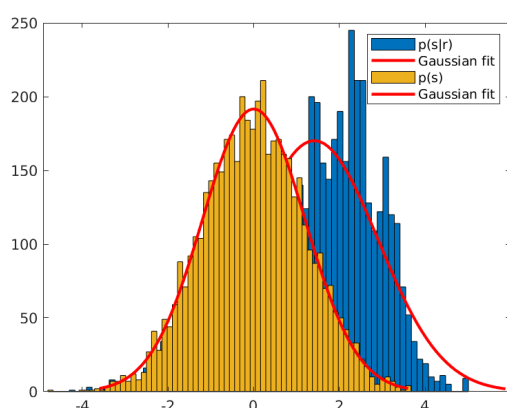


۷.۳ بخش هفتم

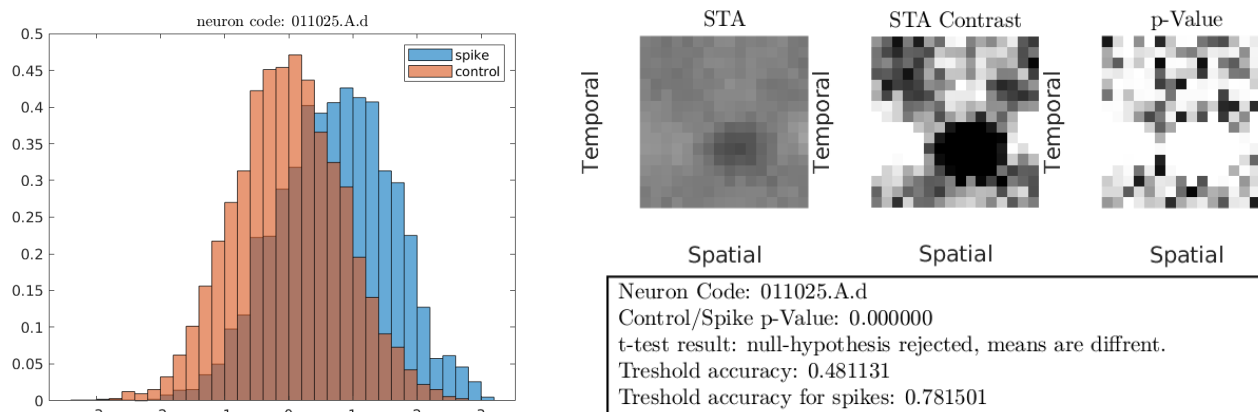
با اینکه میانگین توزیع‌ها با توجه به تست‌های آماری با هم تفاوت دارند از آنجایی که تقریباً تمامی نمودارهای رسم شده نشان می‌دهند که توزیع تصویر تحریک‌ها بر راستای STA برای کلیه تحریک‌ها و تحریک‌هایی که موجب اسپایک شدند تفاوت چشمگیری با هم ندارند و به درستی نمی‌توان ترشهودی پیدا کرد که با انتخاب آن داده‌ها را با دقت خوبی بتوان دسته‌بندی کرد. از طرف دیگر مولفه‌ی STA در بیشتر نورون‌ها الگوی معناداری ندارد و بیشتر شبیه نویز می‌باشد. به عبارتی تا حد خیلی زیادی می‌توان به اعای مقاله مبنی بر پیچیدگی نورون‌هایی که با آنها سروکار داریم استناد کرد. با این وجود می‌توان دید که برخی از نورون‌های جدولی که در بالا ارائه کردیم نه تنها پترنی با الگوی معنادار دارند بلکه نمودار توزیع آن‌ها به علت تفاوت میانگین که باهم دارند را به خوبی می‌توان تفکیک کرد که نسبت به پیچیده بودن رفتار این دسته از نورون‌ها می‌توان تردید داشت. از بین می‌توان نورون با کد 010718.B.c را مثال زد به نمودار و نتایج به دست آمده از آن توجه می‌کنیم.



می‌توانیم ببینیم که دو توزیع آن نسبتاً جدایی پذیر به نظر می‌آیند و همچنین الگو معناداری در تصویر با کانتراست بالای STA وجود دارد. البته باید توجه کنیم علت اینکه ترشهود مناسب برای این نورون انتخاب نشده این است که توزیع spike در آن گاوسی به نظر نمی‌رسد و فیت گاوسی ما اطلاعات نادرستی از توزیع به ما می‌دهد. به این فیت و داده‌های حذف شده توسط آن توجه کنید.



در بین نورون‌های بررسی شده اما نورون‌های دیگری هم وجود داشتند که می‌شد الگوی معناداری در STA آنها مشاهده کرد و از این بین می‌شود به نورون 011025.A.d اشاره کرد که توزیع نمودارهای آن هم جدا از هم به نظر می‌رسد.



همچنین ترشهولدی که برای این نورون پیدا کردیم درصد قابل توجهی از فریم‌های منجر به اسپایک را شناسایی کرده است. نهایتاً می‌توان گفت به طور کلی ادعای مقاله درباره‌ی پیچیده بودن این نورون‌ها درست به نظر می‌رسد با این حال در مورد تعدادی از آنها باید احتیاط بیشتری به خرج بدهیم.

۴ طرح سوال دلخواه

۱.۴ صورت مسئله‌ی طرح شده

با خواندن مقاله متوجه می‌شویم که هدف آن یافتن راستاهاییست که در آن داده‌های موجب اسپایک، واریانس به طور قابل توجهی زیادتری نسبت به کل داده‌ها داشته باشد و در نتیجه توزیع داده‌های موجب اسپایک در آن راستا با توزیع کل داده‌ها متفاوت باشد. در طول انجام این پروژه به این موضوع اشاره نشد.

در اینجا می‌خواهیم این موضوع را مشاهده کنیم یعنی اولاً ببینیم که در راستاهای پیشنهادی مقالی واریانس داده‌ها به طرز قابل توجهی بیشتر از کل داده‌هاست و ببینیم که در روش کلاسیک STA این موضوع برقرار نیست. ثانیاً با بررسی بردارویژه‌های مرتب شده نوع تغییر p-value در تست واریانس را ببینیم.

۲.۴ تست مورد استفاده

برای بررسی موضوعات گفته شده به یک تست آماری نیاز داریم که بزرگ‌تر بودن واریانس یک نمونه نسبت به یک نمونه‌ی دیگر را بررسی کند. با جست‌وجو برای این هدف به تست f-test دست پیدا کردیم که تابع آماده آن موجود است به طور کلی f-test برای دو جمعیت استفاده می‌شود تا ببینیم آیا این دو جمعیت واریانس یکسانی دارند یا نه. ما در اینجا می‌خواهیم از آزمون یک-طرفه استفاده کنیم و تنها بزرگ‌تر بودن واریانس جمعیت اول (توزیع تحریک‌ها به شرط اسپایک) را بررسی کنیم بنابراین:

- $H_0 : \sigma_1^2 = \sigma_2^2$
- $H_a : \sigma_1^2 > \sigma_2^2$