

بسم تعالی



آزمایشگاه علوم اعصاب

گزارش اصلی FMRI

امیرحسین زاهدی ۱۷۰۵۹۹۱۰

بهار ۱۴۰۳

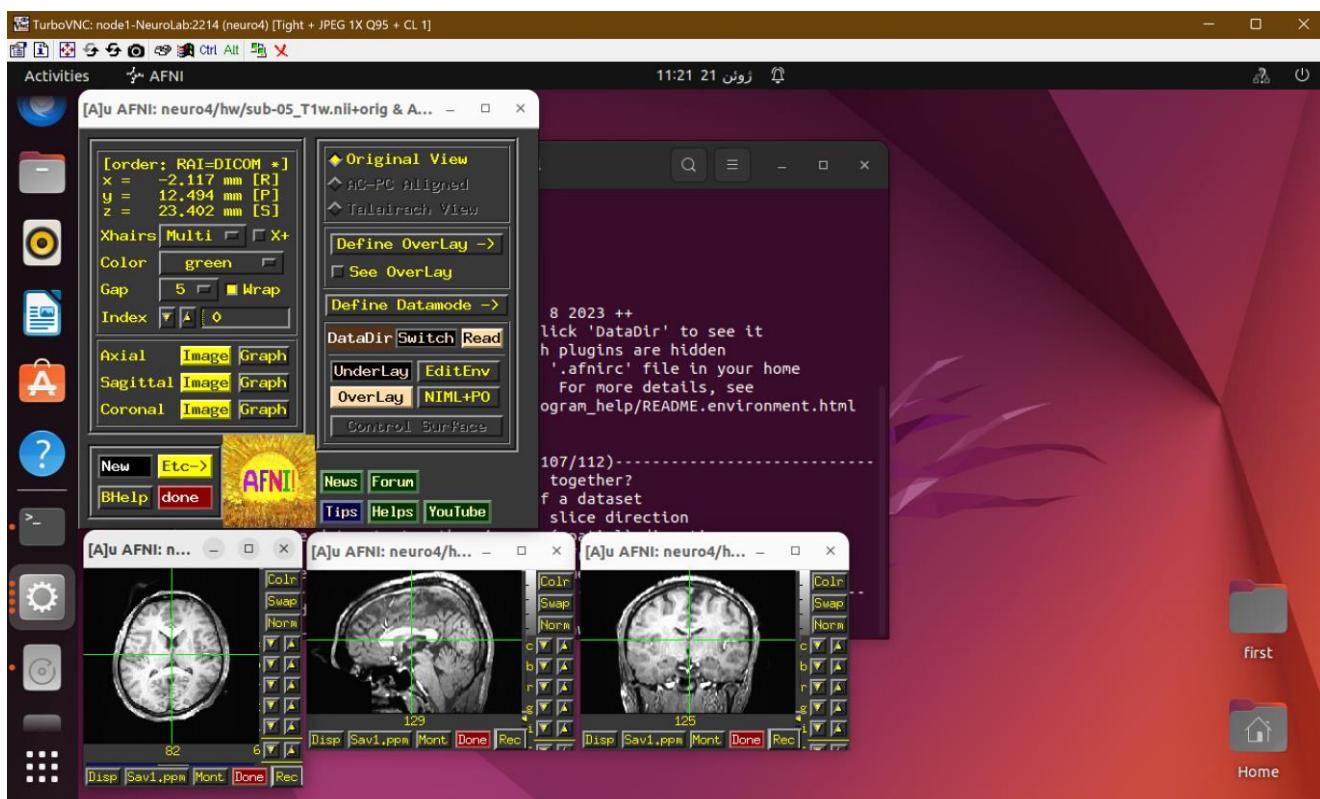
تعهدات: من برای این تمرین به همچنین گنجینه از مدل‌های زبانی بزرگ استفاده نکنم.
من در این تمرین، امتحانی صحیح فرد دیگری استفاده نکنم و مترین خود را در اختیار فرد دیگری نمی‌گذاشتم.

امیری راهنمایی

(تمامی کدهای استفاده شده به ترتیب در فایل تکست پیوست شده اند.)

۱. داده ساختاری:

الف) ابتدا با استفاده از مسیر گفته شده وارد سرور شده و AFNI را ران می‌کنیم. داده ساختاری را به صورت معمولی لود کرده و مشاهده می‌کنیم.



سپس با استفاده از دستور **3dSkullStrip -input sub-05_T1w.nii -prefix SkullStrip_output.nii** که آرگومان ورودی آن اسم تصویر نیفتی است و آرگومان خروجی آن اسم فایل خروجی است که می‌خواهیم تصویر بدون جمجمه را ذخیره کنیم. تصویر بدون جمجمه و سپس تصویر آن به صورت اورلی بر روی ام ار آی اصلی به شرح زیر هستند.



ب) با استفاده از دستور [3dinfo SkullStrip_output.nii](#) که خروجی حاصل از بخش قبل را به آن می‌دهیم، اطلاعات موجود در داده را مشاهده می‌کنیم. اطلاعات به صورت زیر هستند. با استفاده از تعداد وکسل‌ها در هر بعد و ابعاد هر وکسل می‌توان طول و عرض و ارتفاع آن را محاسبه کرد و تخمینی از حجم اخذ شده توسط دستگاه داشته باشیم.

```

2024
Template Space: ORIG
Dataset Type: Anat Bucket (-abuc)
Byte Order: LSB_FIRST {assumed} [this CPU native = LSB_FIRST]
Storage Mode: NIFTI
Storage Space: 20,971,520 (21 million) bytes
Geometry String: "MATRIX(0.9375,0,0,-120.2418,0,-0.9375,0,134.369,0,0,1,-58.5978
5):256,256,160"
Data Axes Tilt: Plumb
Data Axes Orientation:
    first (x) = Right-to-Left
    second (y) = Posterior-to-Anterior
    third (z) = Inferior-to-Superior [-orient RPI]
R-to-L extent: -120.242 [R] -to- 118.821 [L] -step- 0.938 mm [256 voxels]
A-to-P extent: -104.694 [A] -to- 134.369 [P] -step- 0.938 mm [256 voxels]
I-to-S extent: -58.598 [I] -to- 100.402 [S] -step- 1.000 mm [160 voxels]
Number of values stored at each pixel = 1
-- At sub-brick #0 '?' datum type is short: 0 to 3587

----- HISTORY -----
[neuro4@node1-NeuroLab: Fri Jun 21 11:54:33 2024] {AFNI_23.1.05:linux_ubuntu_16_
64} 3dskullStrip -input sub-05_T1w.nii -prefix SkullStrip_output.nii

```

ابعاد را ابتدا محاسبه کرده و سپس حجم را بدست می‌آوریم. $X = 256 * 0.938\text{mm}$ و $Y = 256 * 0.938\text{mm}$ و

$$L = XYZ = 9225.833 \text{ cm}^3$$

ج) همانطور که در الگ نیز دیدیم، مغز بدون جمجمه را بر روی تصویر اصلی انداخته و مقادیر مربوط به قشر سفید و قشر خاکستری و ونتریکل ها را مشاهده می کنیم. به صورت کلی بازه قشر سفید بزرگتر از ۲۵۰۰ است و بازه ونتریکل ها هم کوچکتر از ۱۵۰۰. البته این مقادیر دقیق نیستند و تخمینی هستند به همین دلیل با بررسی نقاط بیشتر و با اندکی آزمون و خطای توافق بازه قشر خاکستری را بهتر و دقیق تر یافت تا سپس با استفاده از آن بازه ماسک مورد نظر را تولید کنیم.

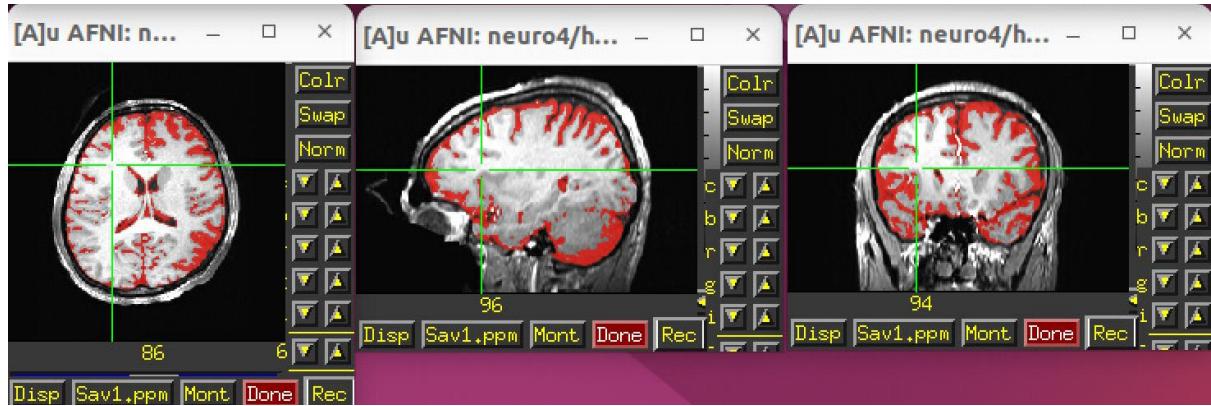
به ترتیب قشر خاکستری، قشر سفید و ونتریکل ها را می بینیم.



حال با استفاده از دستور

3dcalc -a SkullStrip_output.nii -expr 'within(a,1500,2500)' -prefix GrayMask_output.nii

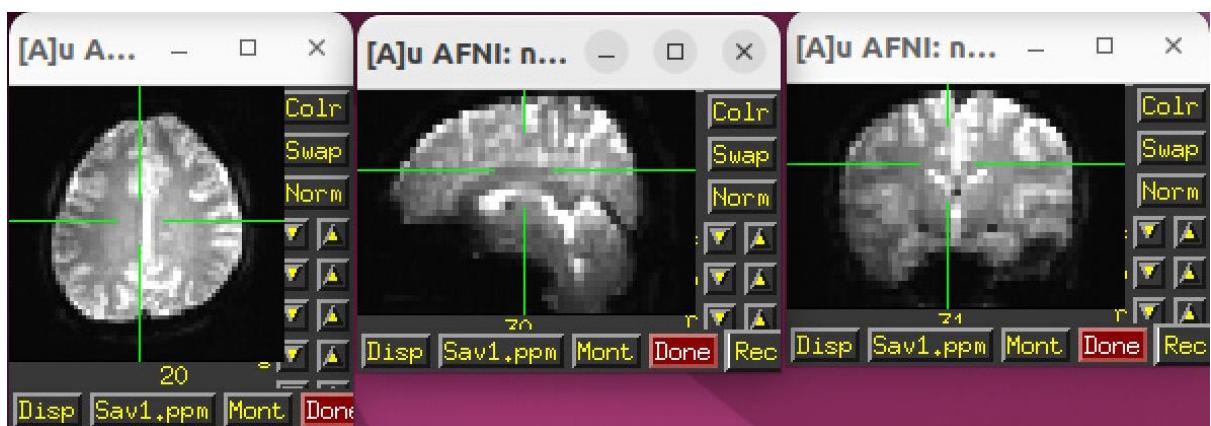
که بازه را تعیین کرده ایم، ماسک جداگانه قشر خاکستری را تولید کرده و در نهایت بر روی تصویر اصلی نمایش می دهیم.



مشاهده می کنیم که بخش خاکستری تا حد خوبی مشخص شده است.

۲. پیش پردازش تصاویر کارکردی:

الف) تصویر کارکردی داده شده به صورت نیفتش ذخیره شده است و نیازی به کانورت کردن ندارد لذا مرحله اول را رد می کنیم.



مرحله دوم که حذف والیوم های اولیه است تا دستگاه به ثبات اولیه برسد نیز نیاز به انجام نیست.

(slice timing correction) در بحث اصلاح زمان بندی اسالیس ها، چون اسالیس ها در زمان های متفاوتی ضبط می شوند و ترتیب متفاوتی نیز طبق فایل داده شده باید اعمال شود، ابتدا زمان ضبط هر اسالیس را بررسی می کنیم تا با استفاده از زمان ها و ترتیب اسالیس ها، دستور کارکشن زمان را اجرا کرده و ترتیب و زمان اسالیس ها را اصلاح کنیم.

```

Identifier Code: NII_4q3CiN_WJL7gFnpPFxRqAQ Creation Date: Fri Jun 21 16:41:56
2024
Template Space: ORIG
Dataset Type: Echo Planar (-epan)
Byte Order: LSB_FIRST {assumed} [this CPU native = LSB_FIRST]
Storage Mode: NIFTI
Storage Space: 44,564,480 (45 million) bytes
Geometry String: "MATRIX(2.994637,0.118959,-0.178862,-98.01894,0.123175,-2.99603
1,0.123839,108.8736,0.130286,0.098221,3.99408,-40.86719):64,64,32"
Data Axes Tilt: Oblique (3.426 deg. from plumb)
Data Axes Approximate Orientation:
    first (x) = Right-to-Left
    second (y) = Posterior-to-Anterior
    third (z) = Inferior-to-Superior [-orient RPI]
R-to-L extent: -98.019 [R] -to- 90.981 [L] -step- 3.000 mm [ 64 voxels]
A-to-P extent: -80.126 [A] -to- 108.874 [P] -step- 3.000 mm [ 64 voxels]
I-to-S extent: -40.867 [I] -to- 83.133 [S] -step- 4.000 mm [ 32 voxels]
Number of time steps = 170 Time step = 2.00000s Origin = 0.00000s
-- At sub-brick #0 '??' datum type is short [*1.22466]
-- At sub-brick #1 '??' datum type is short [*1.22466]
-- At sub-brick #2 '??' datum type is short [*1.22466]
** For info on all 170 sub-bricks, use '3dinfo -verb' **

```

مشاهده می کنیم که هر استپ ۲ ثانیه به طول می انجامد و حاوی ۳۲ اسلایس است. پس تفاوت زمانی بین هر اسلایس برابر ۶۲.۵ میلی ثانیه است. پس اینگونه می توانیم زمان اسلایس های داده شده طبق تکست را بدست آوریم.

پس از آن با استفاده از ترمینال تکستی حاوی زمان های اسلایس ها تولید می کنیم و با استفاده از دستور زیر، اصلاح زمانی را اعمال می کنیم.

`3dTshift -tpattern @slice_times.txt -prefix FT_run1_tshift
AVO_25_M21_062811_Vis01_170r.nii3`

(motion correction) برای اصلاح حرکات های ناخواسته و جبران آرتیفیکت ها سعی می کنیم نویز حاصل از موشن را به وسیله همسان سازی نسبت به مرجع، اصلاح کنیم. با دستور

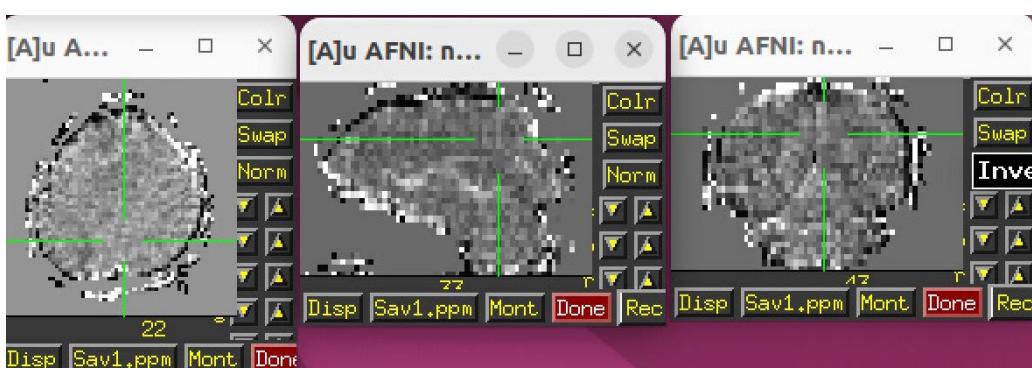
`3dvolreg -verbose -zpad 1 -base FT_run1_tshift+orig[2] -1Dfile dfile.FT1.1D -prefix
FT_run1.volreg -cubic -1Dmatrix_save mat.FT1.vr.aff12.1D FT_run1_tshift+orig`

اصلاح موشن را انجام می دهیم. البته در خروجی مشهود نیست.

(global intensity normalization) برای نرمالیزه کرده ابتدا میانگین اسلایس ها را حساب کرده و سپس هر کدام را نرمالایز می کنیم. با استفاده از دو دستور زیر.

`3dTstat -mean -prefix meanfunc1.nii.gz FT_run1.volreg+orig`

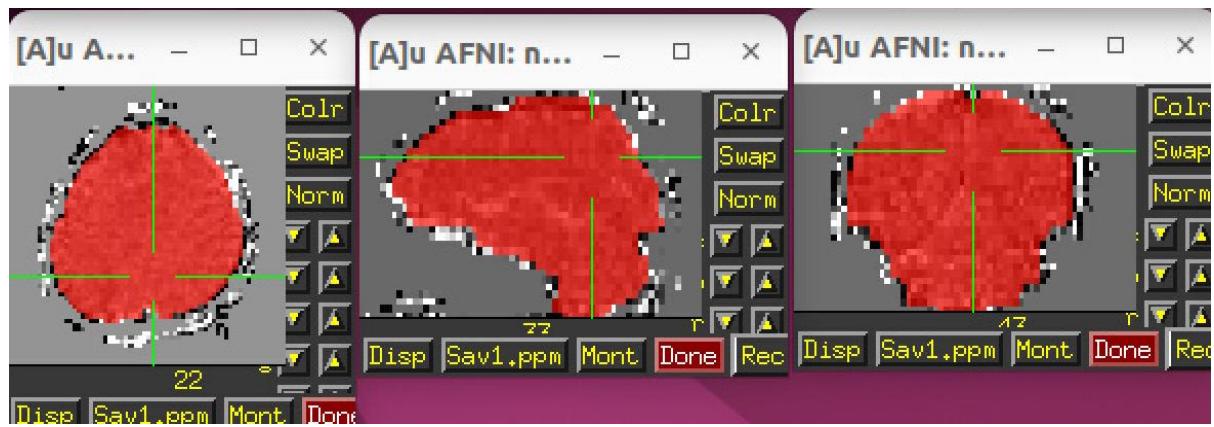
`3dcalc -a FT_run1.volreg+orig -b meanfunc1.nii.gz -expr '(a-b)/(abs(b)+1)*astep(b,200)*100' -prefix FT_run1.scale`



(special and temporal filtering) برای این بخش ابتدا با استفاده از دستور:

```
3dAutomask -clfrac 0.4 -prefix clean_base.nii.gz meanfunc1.nii.gz
```

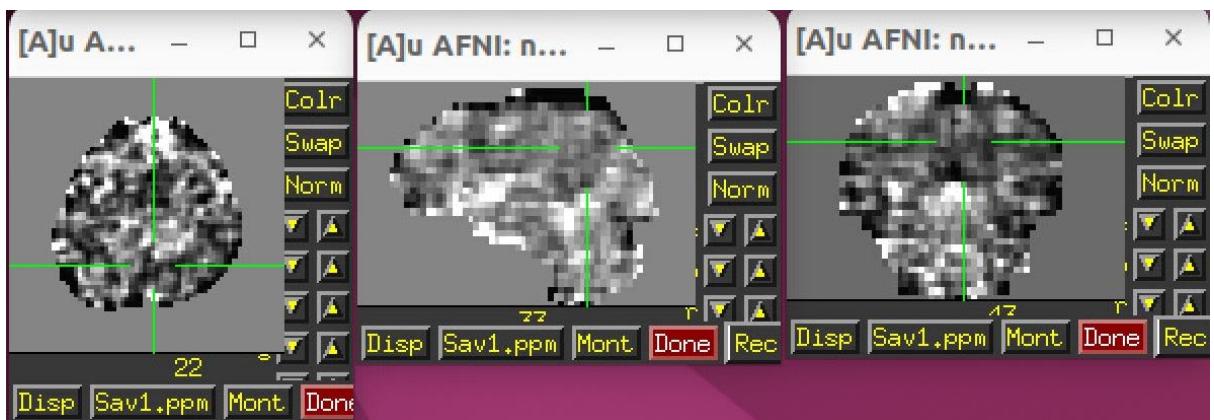
ماسکی می سازیم تا نواحی غیر مغزی را از مغزی جدا کند. پارامترهای آن ورودی و خروجی و مقدار آستانه شدت برای شناسایی نواحی است. بدین صورت نواحی نسبت به آستانه سنجیده شده و برچسب مغز یا غیر مغز می گیرند.



در مرحله بعد فیلتر بند پس را برای حذف نویز های بالا یا خیلی پایین انجام می دهیم. دستور چنین است:

```
3dBandpass -input FT_run1.scale+orig -mask clean_base.nii.gz -blur 5 -band 0.01 0.1 -prefix FT_run1.scale.smooth
```

به جز آرگومان های ورودی و خروجی ماسک طراحی شده در بخش قبل را دریافت می کند تا بر روی نواحی مغزی فیلتر اعمال شود. همچنین یک آرگومان گوسی مربوط به میزان اسموٹ کردن تصویر دارد.



بخش نرمالیزیشن و انطباق با داده ساختاری را که بخشی از پری پراسس است را در سوال سوم انجام می دهیم.

ب) نمی توان گفت که لزوماً کدام بهتر است و به داده برمیگردد اما اگر زمان اول اصلاح شود همه اسلایس ها با ترتیب مناسب قرار گرفته و هم زمان می شوند. اگر پس از آن اصلاح حرکت سر را اصلاح کنیم، مرجع را نسبت به ترتیب اصلاح شده انتخاب می کنیم و اینگونه اصلاح نویز حرکت سر پیوسته تراست و بهتر انجام می شود. همچنین رفع نویز موشن زمان اسلایس ها را نیز در نظر میگیرد به همین دلیل روش بهتر این است. از طرفی اگر قبل از اصلاح زمان موشن اصلاح شود، به دلیل تفاوت ترتیبی که در نظر گرفته نشده و بعداً اعمال می شود، نویز به خوبی حذف نشده و حتی افزوده میشود.

۳. منطبق کردن تصاویر کارکردی و ساختاری:

در بخش اول داده ساختاری بدون جمجمه را بدست آوردیم. در بخش دوم نیز تصویر میانگین از داده های کارکردی و سپس ماسک با استفاده از میانگین بر روی داده کارکردی بدست آمد. ابتدا داده کارکردی را با داده ساختاری هم راستا می کنیم. این کار با دستور زیر انجام می شود.

```
align_epi_anat.py -anat2epi -anat SkullStrip_output.nii -anat_has_skull no -suffix _al_junk -epi clean_base.nii.gz -epi_base 0 -epi_strip 3dAutomask -cost nmi -giant_move -check_flip -volreg off -tshift off
```

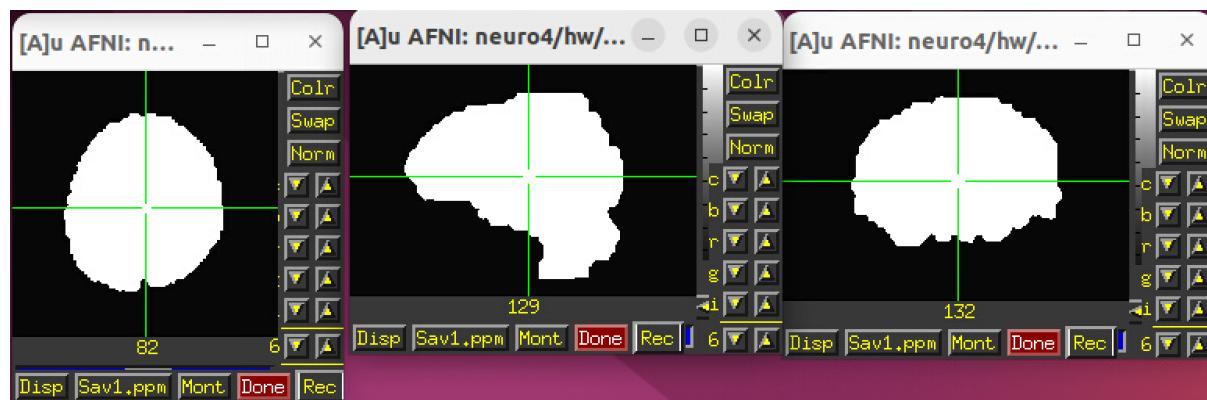
این کد در رودی داده ساختاری بدون جمجمه، بدون جمجمه بودن یا نبودن، پسوند اضافه شده به خروجی، داده کارکردی ورودی، اسلاس مرجع داده کارکردی را دریافت می کند. همچنین ورودی های دیگری اعم از ماسک جدا کننده مغز و نوع تابع هزینه و ... را می دهد.تابع هزینه MNI برای بهینه سازی فرایند هم راستا سازی استفاده می شود.

در مرحله بعد از تکه کدی استفاده می کنیم که ماتریس تبدیل را معکوس و ذخیره می کند:

```
cat_matvec SkullStrip_output_al_junk_mat.aff12.1D -I > func2mri_warp.1D
```

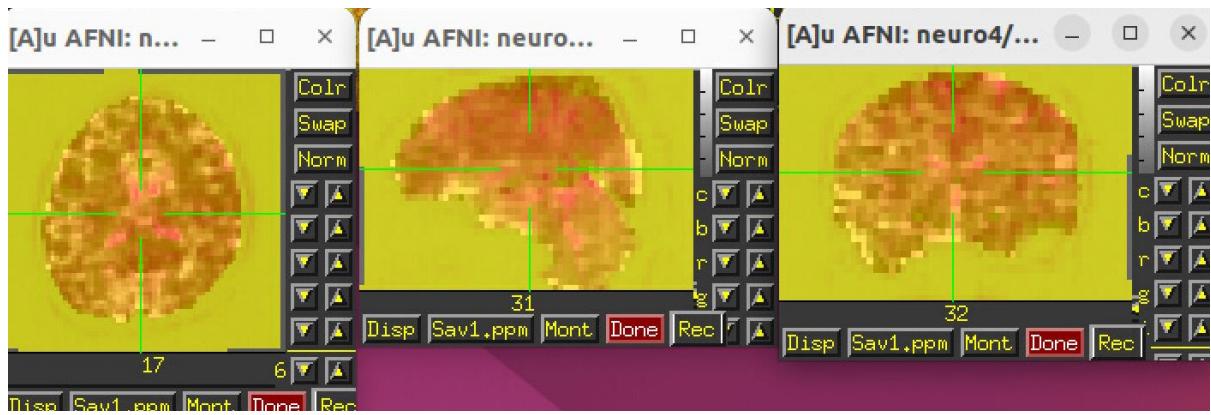
اینبار از دستوری استفاده می کنیم که تصویر کارکردی را با ساختاری هم راستا می کند که این عمل با استفاده از بردار تبدیل معکوس بدست آمده در بخش قبل انجام می گردد.

```
3dAllineate -base SkullStrip_output.nii -input clean_base.nii.gz -1Dmatrix_apply func2mri_warp.1D -prefix epimask2anat.nii.gz
```



با استفاده از دستور زیر یک تبدیل خطی و یک تبدیل غیر خطی را به ورودی اعمال می کنیم که ورودی تصویر هم راستا شده با استراکچرال است. در آخر خروجی بدست آمده در بخش بعدی استفاده می گردد.

```
3dNwarpApply -nwarp anatQQ.FT.aff12.1D anatQQ.FT_WARP.nii -source epimask2anat.nii.gz -master MNI152_2009_template_SSW.nii.gz -dxyz 3 -prefix epimask2mni.nii.gz
```

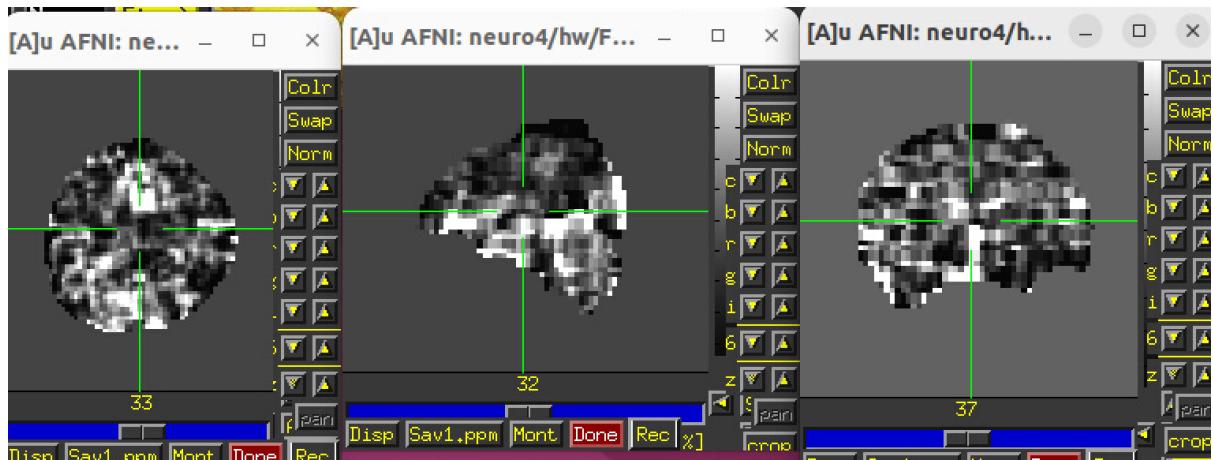


(این بخش جزو سوال ۲ است ولی چون مقدمات آن در سوال ۳ بدست می آمد در این بخش آورده شده است.):

در نهایت برای نرمال سازی تصویر بدست آمده از بخش های پیش پردازش قبلی، با استفاده از ماسک تبدیل بدست آمده و تصویر فیلتر شده از بخش دوم و ضرب این دو، خروجی نرمالیزه شده را بدست می آوریم. طبق دستور زیر: (البته قبل از اعمال دستور یک ریسمپلینگ برای تطبیق ابعاد انجام شده است.

```
3dcalc -a FT_run1.scale.smooth+tlrc -b epimask2mni.nii.gz-expr 'a*b' -prefix  
FT_run1.scale.smooth.masked
```

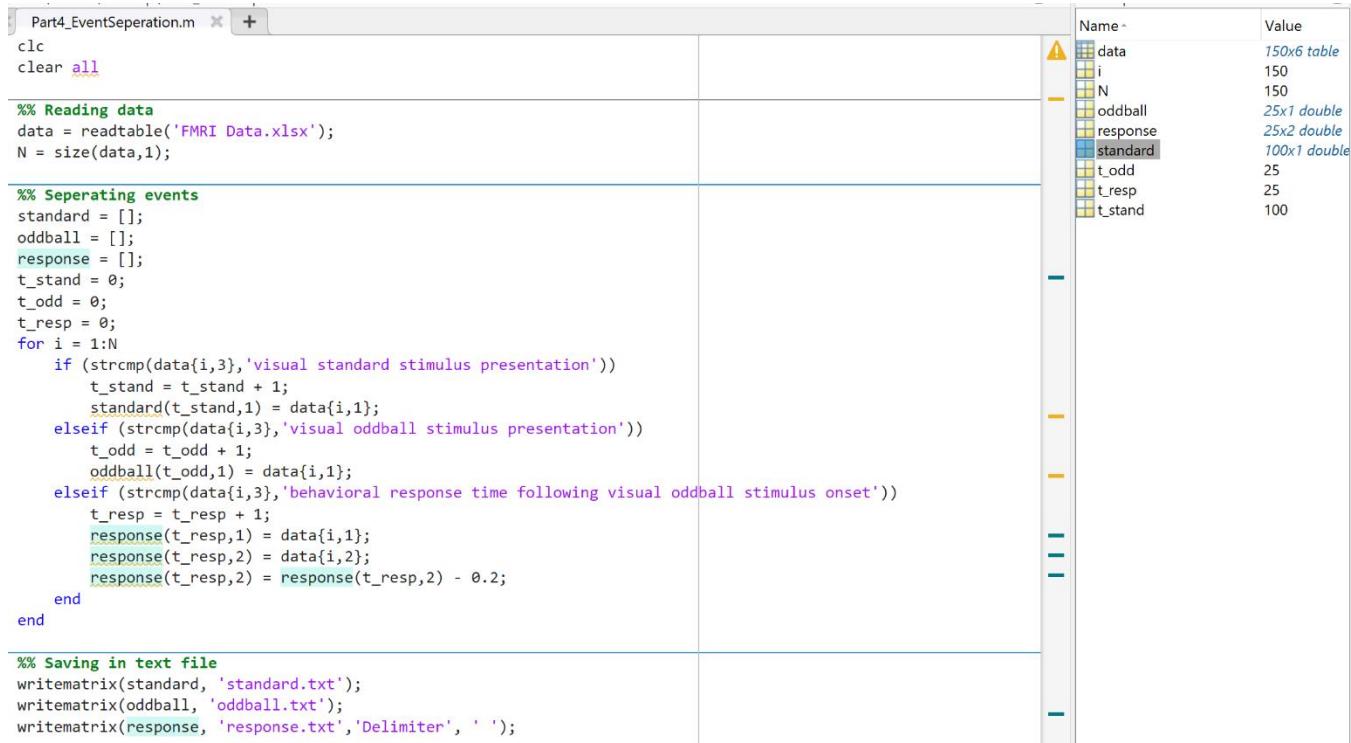
این دستور ورودی و ماسک را می گیرد و با توصیف ریاضی داده شده که در اینجا ضرب است، داده خروجی را بدست آورده و ذخیره می کند. خروجی نهایی می شود:



نمی دانم در چه حد تصویر خروجی صحیح است اما مراحل مجددا چک شده و گویا دوستان دیگر هم به همین نتیجه رسیده اند.

۴. ساختن رگرسور ها:

با توجه به تسک گفته شده، اطلاعاتی حاوی ایونت های دیداری استاندارد و آدبال داریم که فرد با دیدن درخشش نا معمول باید دکمه می زدہ است. برای ساخت رگرسور برای هر کدام از ایونت ها به علاوه ایونت فشردن دکمه لازم است که از فایل داده های کلی، زمان های مربوط به هر ایونت را استخراج کنیم (البته سه ردیف اول نیازمند اصلاح بودند). برای اینکار فایل تکست درون سرور را به اکسل داخل ویندوز منتقل کرده و برنامه متلبی نوشتم که زمان ایونت ها را به صورت جداگانه ذخیره کند.



The screenshot shows a MATLAB workspace window with two panes. The left pane displays a script named 'Part4_EventSeparation.m' containing MATLAB code for reading data from an Excel file, separating events into standard, oddball, and response categories, and saving them to text files. The right pane shows the workspace variables and their values:

Name	Value
data	150x6 table
i	150
N	150
oddball	25x1 double
response	25x2 double
standard	100x1 double
t_stand	100
t_odd	25
t_resp	25

در بخش اول کد، با توجه به ایونت ثبت شده جداسازی را انجام می دهیم. برای بخش کلید زدن، زمان واکنش را با استفاده از میزان ری اکشن تایم منهای ۲۰۰ میلی ثانیه بدست می آوریم و آن را نیز ذخیره می کنیم.

سپس سه فایل تکست ساخته می شود که این سه فایل تکست را در سرور نیز متقابلاً تولید می کنیم.

با استفاده از سه تکست بدست آمده و دستور `3dDeconvolve` می توانیم یک مدل خطی کلی را با استفاده از سه رگرسور گفته شده که با استفاده از این سه تکست بدست می آیند را اجرا کنیم. دستور کلی را در بخش ۵ می آورم ولی دستوری که متناسب با سوال ۴ باشد به شکل زیر است:

```
-num_stimts 3 \
-stim_times 1 standard.txt 'BLOCK(0.2,1)' \
-stim_label 1 standard \
-stim_times 2 oddball.txt 'BLOCK(0.2,1)' \
-stim_label 2 oddball \
```

```
-stim_times_AM1 3 response.txt 'dmBLOCK(1)' \
```

```
-stim_label 3 response \
```

این آرگومان ها که ورودی های تابع گفته شده هستند، ورودی اول تعداد رگرسور است، ورودی های بعدی سه رگرسور هستند که با تکست های بدست آمده تولید شده اند، که تکست متناظر هر سه رگرسور و لیبل متناظر آن ها مشخص شده است. بخش آخر روبروی نام فایل، نشان دهنده میزان به طول انجامیدن ایونت است که برای ایونت های استاندارد و آدبال ۲۰۰ میلی ثانیه است اما چون برای سومین رگرسور که فشردن کلید هست، زمان به صورت دقیق مشخص نیست از دستور زمان متغیر (1) استفاده می گردد. (1) BLOCK نشان دهنده نوع محرک است.

در این بخش رگرسور ها ساخته شده اند، برای تکمیل این بخش نیاز به انجام سوال پنجم است تا تحلیل GLM با استفاده از این رگرسور ها انجام شود.

۵. آنالیز :GLM

با استفاده از دستور زیر این آنالیز را انجام می دهیم.

```
3dDeconvolve -input FT_run1.scale.smooth+orig \
```

```
-polort 0 \
```

```
-num_stimts 3 \
```

```
-stim_times 1 standard.txt 'BLOCK(0.2,1)' \ \
```

```
-stim_label 1 standard \ \
```

```
-stim_times 2 oddball.txt 'BLOCK(0.2,1)' \ \
```

```
-stim_label 2 oddball \ \
```

```
-stim_times_AM1 3 response.txt 'dmBLOCK(1)' \ \
```

```
-stim_label 3 response \ \
```

```
-jobs 2 \ \
```

```
-gltsym 'SYM: oddball -standard' \ \
```

```
-glt_label 1 oddball-standard \ \
```

```
-gltsym 'SYM: 0.5*oddball +0.5*standard +0.5*response' \ \
```

```
-glt_label 2 mean.oddballstandard \ \
```

```
-fout -tout -x1D X.xmat.1D -xjpeg X.jpg \ \
```

```
-errts errts.FT1 \ \
```

```
-bucket stats.FT1 \ \
```

به صورت کلی GLM ساخت ترکیب خطی ای از رگرسورها با تخمین ضرایبشان است به صورتی که با استفاده از کمترین ارور مربع، شبیه ترین حالت را به داده دریافتی داشته باشد.

در این دستور داده ورودی فانکشنال را به همراه سه تکست حاوی زمان ایونت‌ها که رگرسورها را می‌سازند، به همراه لیبلشان را ورودی می‌دهیم. (البته قبل از آن میانگین را نیز \cdot می‌کنیم). همچنین دو پردازش از نوع میانگین فعالیت دو محرك و پاسخ و اختلاف فعالیت این دو نیز تعریف می‌کنیم. علاوه بر موارد گفته شده، f-test و t-test را نیز در این تابع اضافه می‌کنیم تا محاسبه شوند. در نهایت ماتریس طراحی هم به صورت معمولی هم به صورت تصویری به همراه مقادیر باقی مانده مدل ساخته شده نسبت به داده اصلی و خروجی نهایی مدل که شامل ضرایب رگرسیون است، بدست می‌آیند. ترتیب موارد بیان شده همانند ترتیب آرگومان‌های تابع بودند.

تابع را استفاده کرده و خروجی‌ها را ذخیره می‌کنیم تا در بخش بعدی از آن‌ها استفاده کنیم.

سوالاتی که پرسیده شده اند کمی ابهام دارند ولی به صورت کلی حالا که بر روی داده کارکردی پیش‌پردازش لازم به صورت کامل انجام شده است و دیتای خوبی برای بررسی نحوه عملکرد نواحی مختلف مغز در حین برخورد با محرك‌های بیرونی را داریم چرا که نه. می‌آییم و با توجه به زمان‌های تحریک و مدلی که برای فعالیت مغز در نظر می‌گیریم به ازای زمان هر کدام از آن‌ها، سیگنال‌هایی را تولید می‌کنیم که معتقد‌هستیم خروجی بدست آمده حاصلی از ترکیب خطی این سیگنال‌ها هستند. سپس به دنبال ضرایب آن‌ها هستیم تا بتوانیم به بهترین شکل مدل را بازسازی کنیم. پس وقتی که داده ما پیش‌پردازش شده است و آماده تحلیل سطح بالاتری است، رگرسیون حالت کلی که GLM باشد می‌تواند در این مرحله استفاده شود.

نمیدانم منظورتان از نسخه پیشنهادی چیست اما در ادامه از آماره‌های بدست آمده از تحلیل GLM می‌توانیم به بخش‌هایی از مغز که در طی فرایند آدبال، استاندارد معمولی و یا ری اکشن زدن دکمه فعالیت بیشتری انجام داده اند یا به عبارتی ضریب رگرسور متناظر بزرگتری دارند بررسیم که این خود نتیجه‌ای ارزشمند و قابل استفاده است. خروجی‌های بدست آمده از تحلیل گفته شده جز خود آماره‌ها، باقی مانده‌ها هم هستند که نشان می‌دهند چه میزان رگرسیون با استفاده از رگرسور‌های گفته شده نا موفق بوده اند.



۶. انطباق فعالیتها تصاویر ساختاری و نمایش آنها:

ابتدا با دستور زیر و با استفاده از ماتریس تبدیلی که برای تبدیل فانکشنال به استراکچرال بدست آوردیم، ضرایب را بر تصویر ساختار منطبق می کنیم.

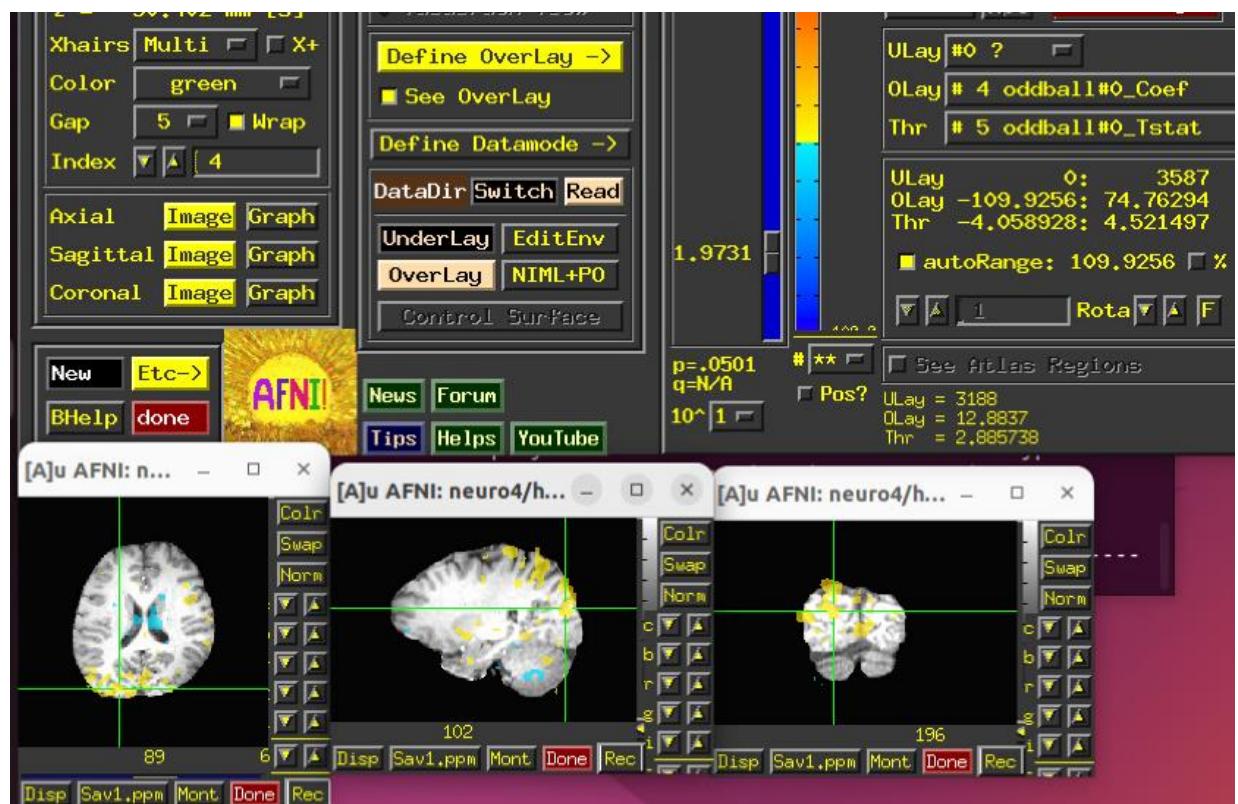
```
3dAllineate -base SkullStrip_output.nii -input stats.FT1+tlrc -1Dmatrix_apply  
func2mri_warp.1D -prefix stats.FT1.aligned.nii.gz
```

تصویر ساختاری را به عنوان پایه می دهیم. ورودی را که خروجی ضرایب باشد نیز می دهیم و ماتریس تبدیل را اعمال می کنیم. در نهایت خروجی را برای هرسه رگسسور و اختلاف آدال از استاندارد بر روی مغز به صورت اورلی رسم می کنیم و نقطه ای را به عنوان نقطه فعال نشان می دهیم. لازم به ذکر است که p -value را برابر 0.05 قرار می دهیم تا اطلاعات سیگنیفیکنت نمایش داده شوند.

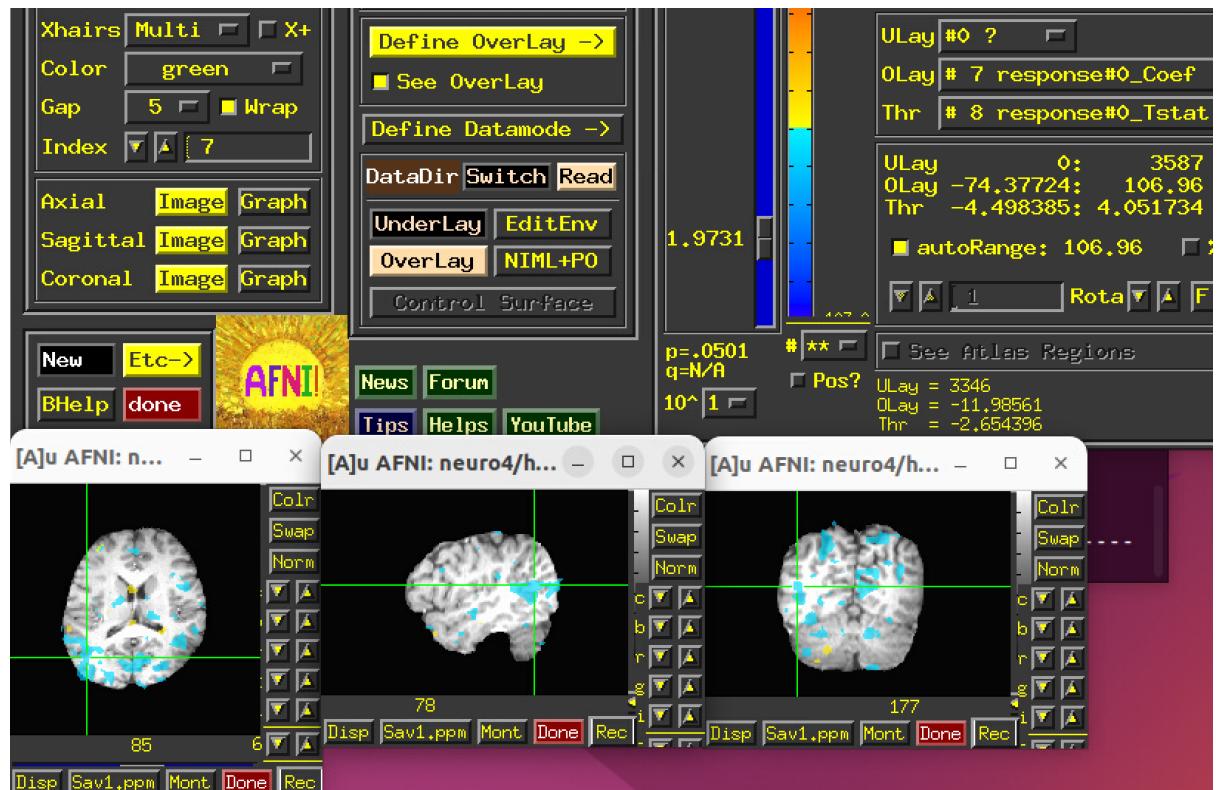
ابتدا استاندارد:



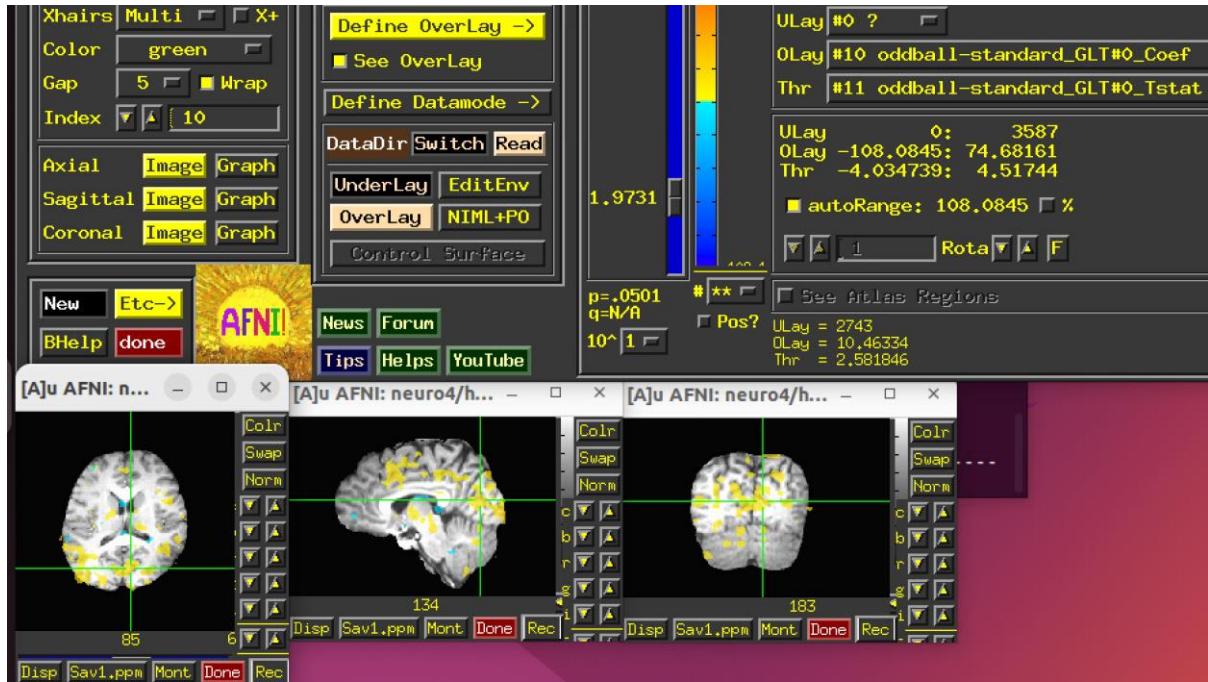
آداب:



پاسخ:



آدبال منهای استاندارد:



تا حد خوبی می توان فعالیت بخش اوكسیپیتال که مربوط به پردازش بینایی است را در همه شرایط مشاهده کرد و همچنین نواحی آهیانه ای و مرکزی تر کورتکس که مسئول تصمیم گیری و حرکات ماهیچه ها هست را به خصوص در آدبال و ریسپانس مشاهده کرد. فعالیت ها تقریبا همانطور که انتظار داریم هستند زیرا که تسک ما از نوع بینایی است و پاسخ آن نیز با کلیک کردن انجام می گردد، به همین دلیل فعالیت ناحیه هایی از مغز مرتبط با بینایی و فعالیت های حرکتی توقع می روند.

۷. تصحیح نقشه های فعالیت و مشاهده نتایج با کنترل خطای مثبت کاذب:

باقی مانده ها را مانند ضرایب همانطور که در بخش قبل مفصل توضیح داده شد، منطبق با داده ساختاری می کنیم.

```
3dAllineate -base SkullStrip_output.nii -input errts.FT1+tlrc -1Dmatrix_apply  
func2mri_warp.1D -prefix errts.FT1_aligned.nii.gz
```

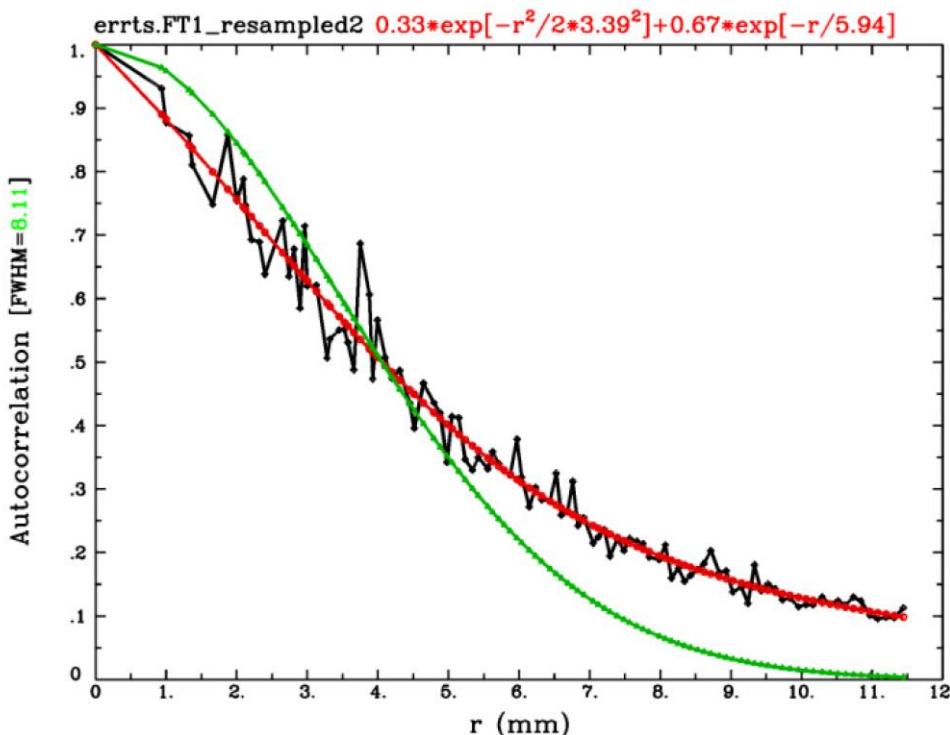
پس از آن با استفاده از ماسک قشر خاکستری دستور زیر را اجرا می کنیم. البته ابتدا ریسمپلینگ می کنیم تا ابعاد بخوانند.

```
3dFWHMx -acf -mask GrayMask_output.nii -input errts.FT1_resampled2+orig
```

این دستور پارامتر های همبستگی مکانی را محاسبه می کند و با توجه به آنکه قشر خاکستری را به عنوان ماسک می دهیم،

تحلیل را صرفا بر روی قشر خاکستری انجام می دهد. پارامترها:

1	0	0	0	0
2	0.331835	3.39421	5.94047	8.11322



حال اندازه خوشه ها را بر اساس آستانه احتمالی های ۰.۰۵ و ۰.۰۱ محاسبه می کنیم. این محاسبه همانند بخش قبل بر روی قشر خاکستری انجام می گیرد و از پارامتر های بدست آمده استفاده می گردد. NN ۱ نشان دهنده نزدیک ترین همسایگی برای خوشه بندی است. فرایند گفته شده با دستور زیر انجام می شود:

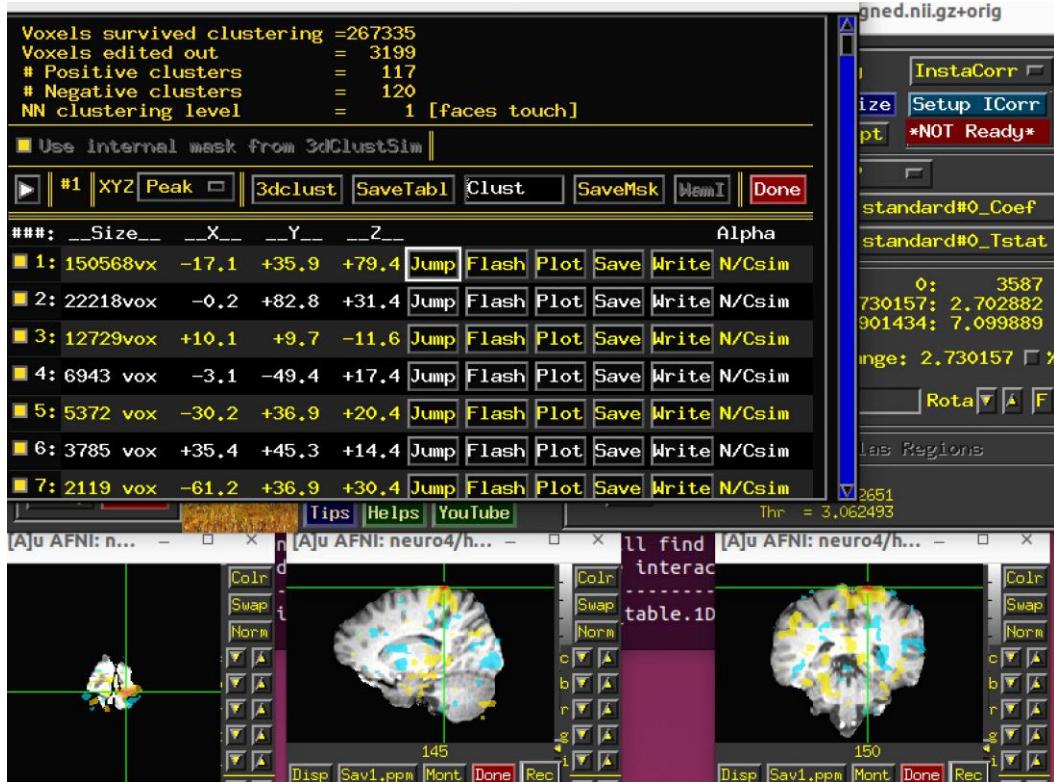
```
3dClustSim -mask GrayMask_output.nii -acf 0.331835 3.39421 5.94047 -NN 1 -pthr 0.01  
0.05 -prefix clustsim_output
```

جدول اندازه کلاستر های بدست آمده با NN=1 در فایل تکست ClustSym پیوست شده است.

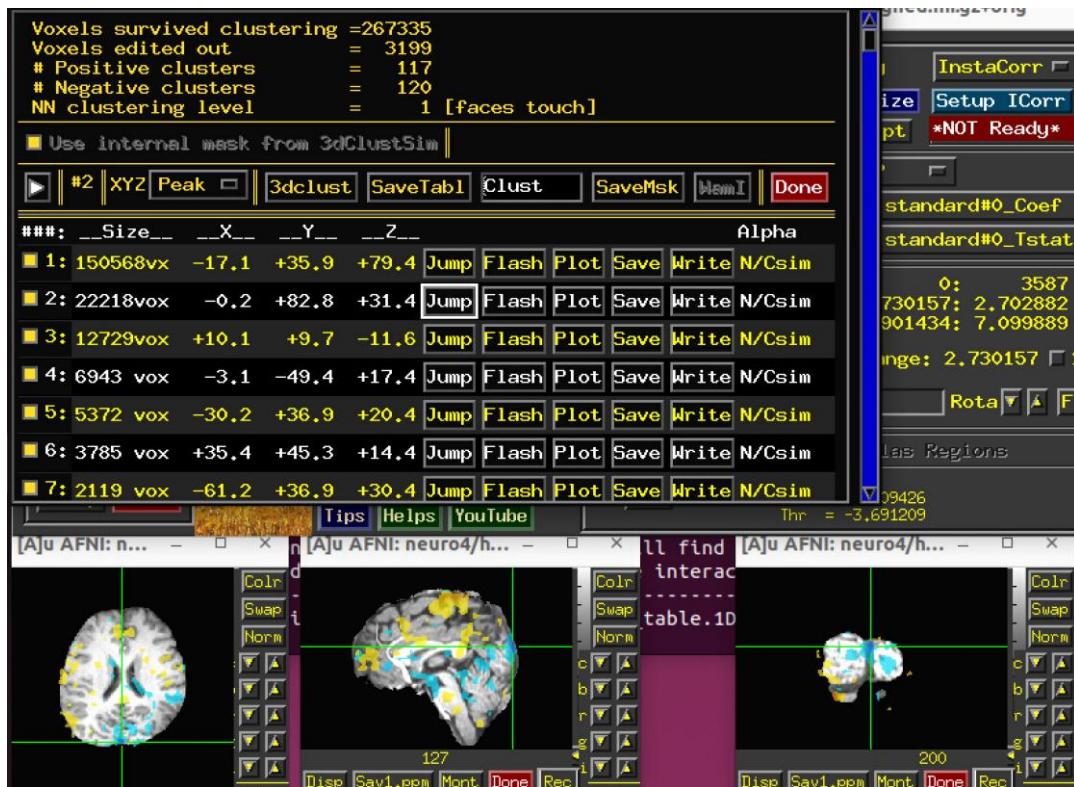
به صورت کلی کاهش FPR اندازه خوشه های یافت شده بیشتر می شوند زیرا که خوشه های کوچکتر که حاصل نویز هستند در نظر گرفته نمی شوند، این در نظر نگرفتن ناشی از آن است که معنادار بودن خوشه با کاهش FPR مهمتر شده.

نقشه فعالیت های هر ۴ بتا مپ با p -value برابر ۰.۰۵ را برای سه بزرگترین کلاستر می کشیم. جدول کلاستر های هر بتا مپ در فایل تکست متناظر پیوست شده است.

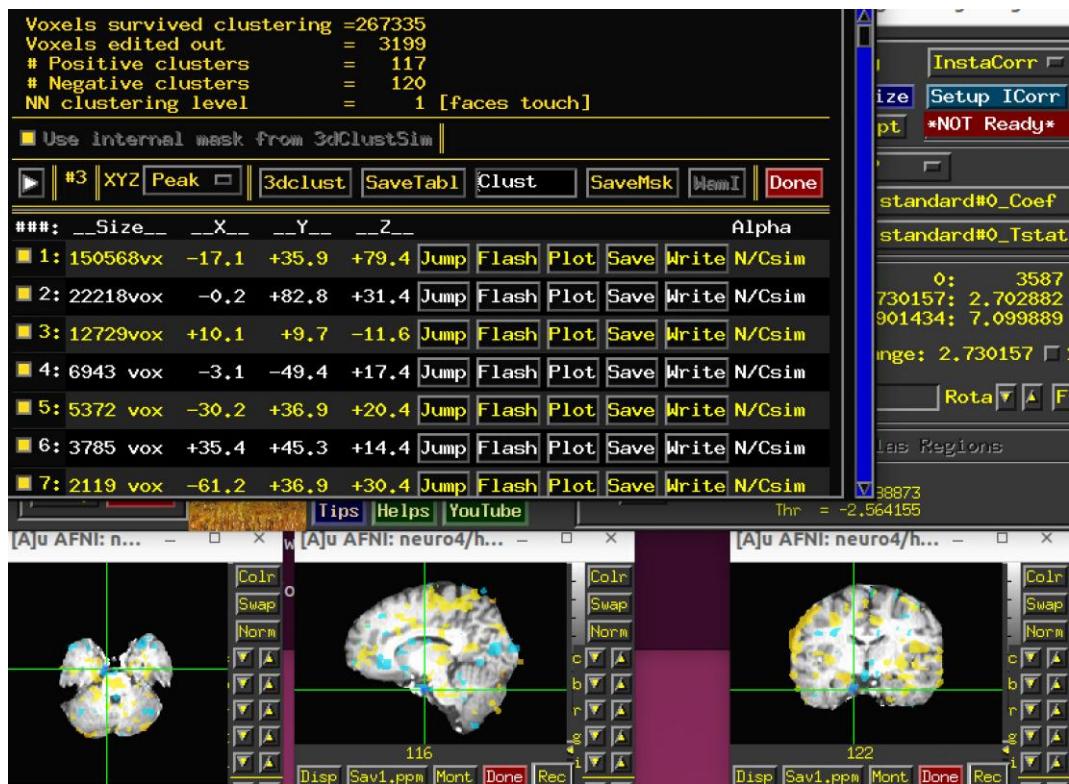
استاندارد - خوشه اول:



استاندارد - خوشه دوم:



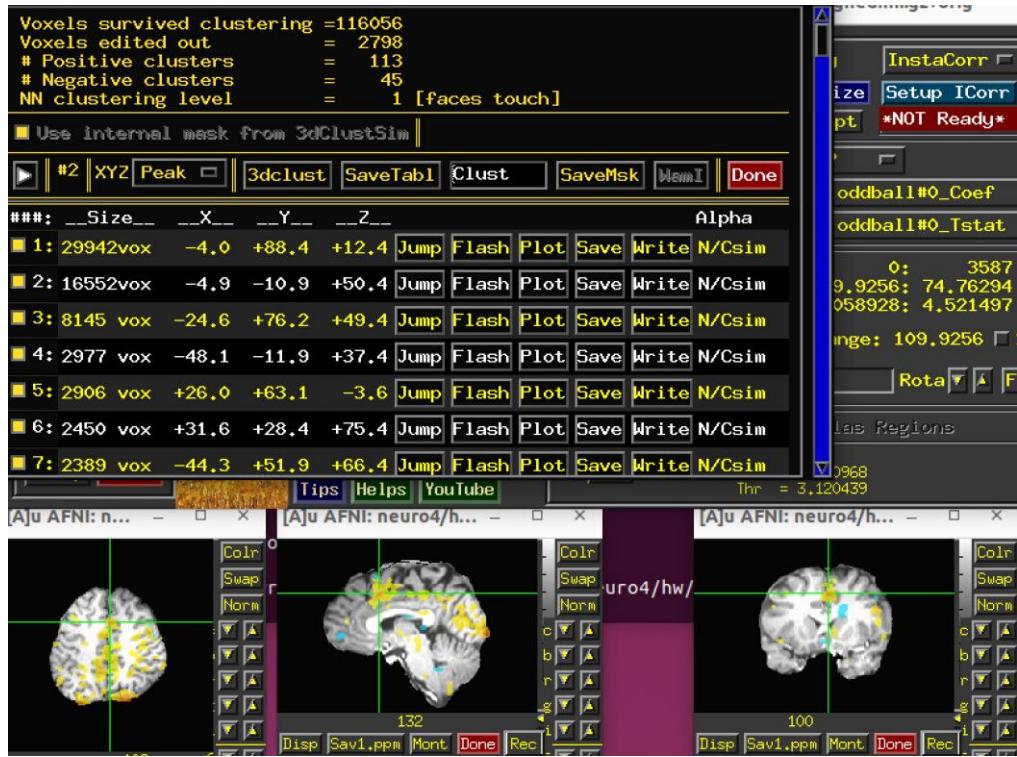
استاندارد - خوش سوم:



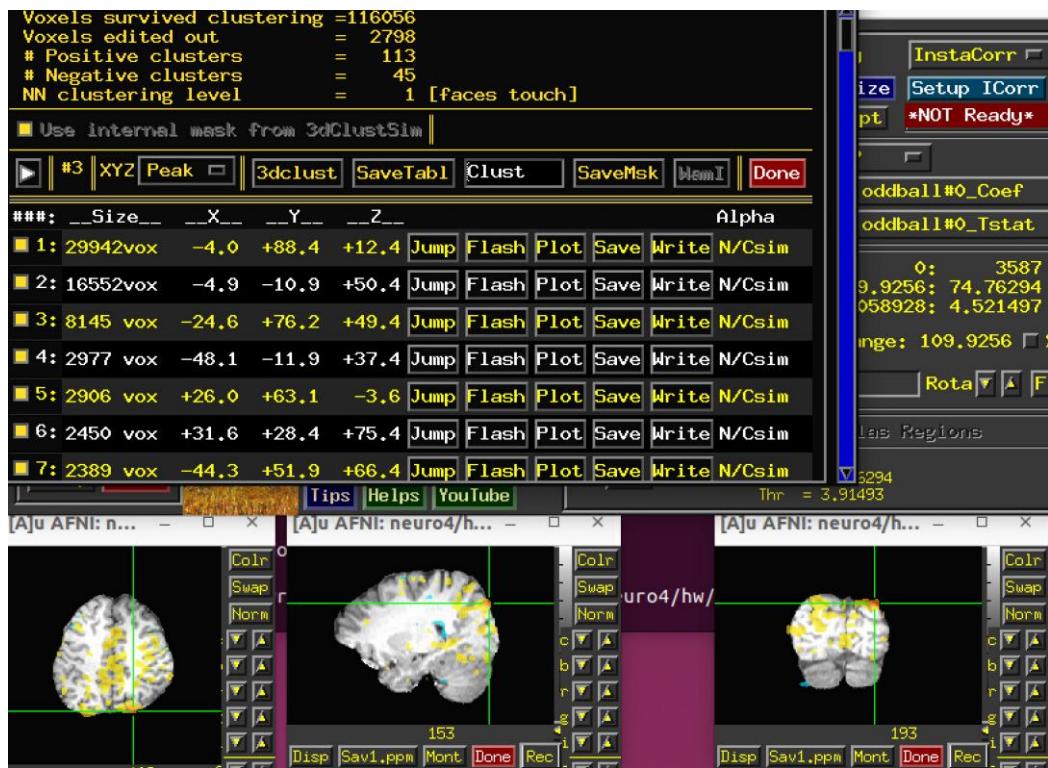
آدبال - خوش اول:



آدبار - خوش ۴ نوبت:



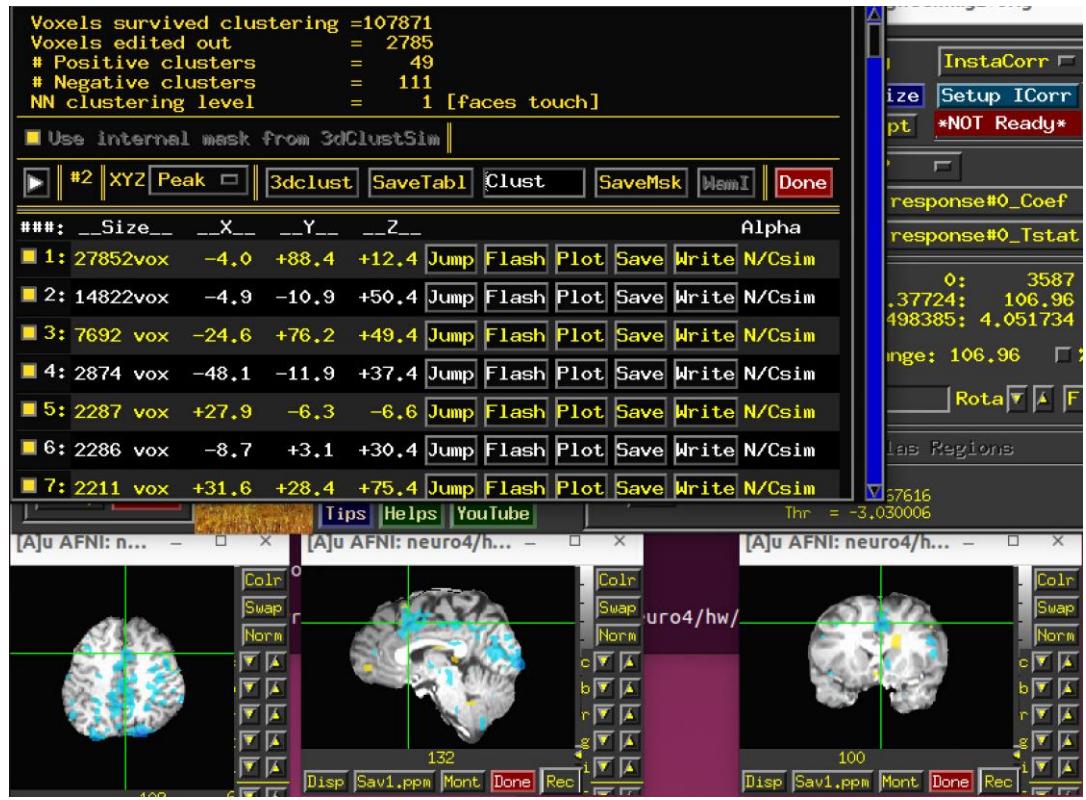
آدبار - خوش ۴ سوم:



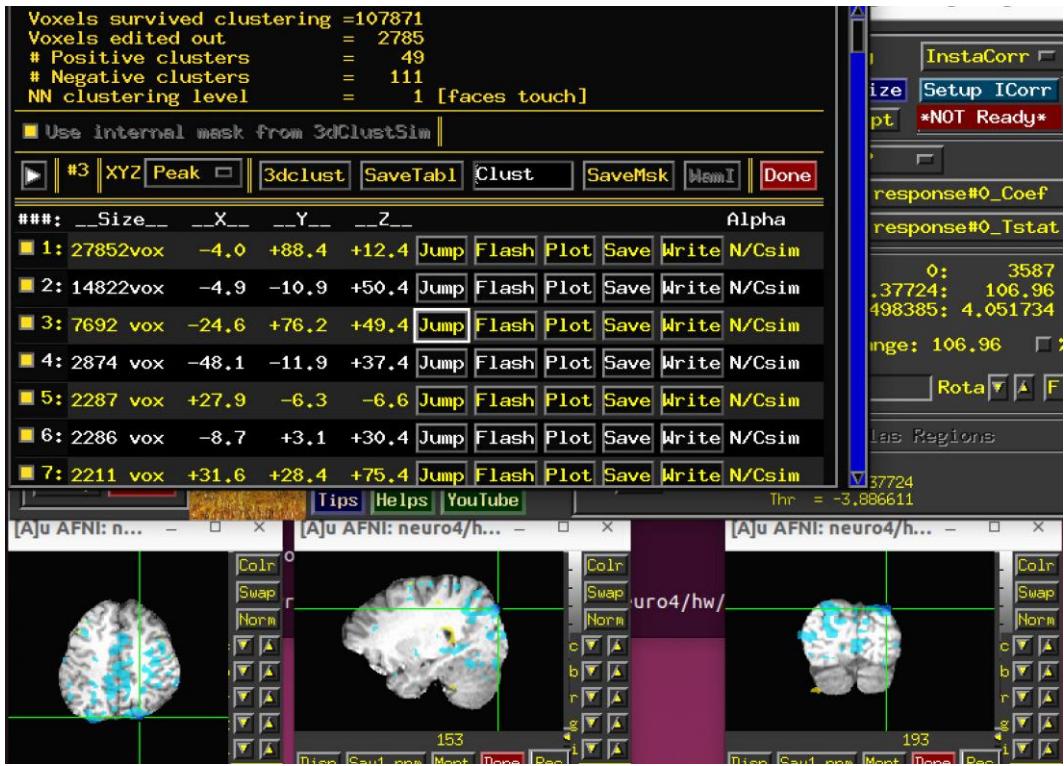
ریسپانس - خوش ۴ اول:



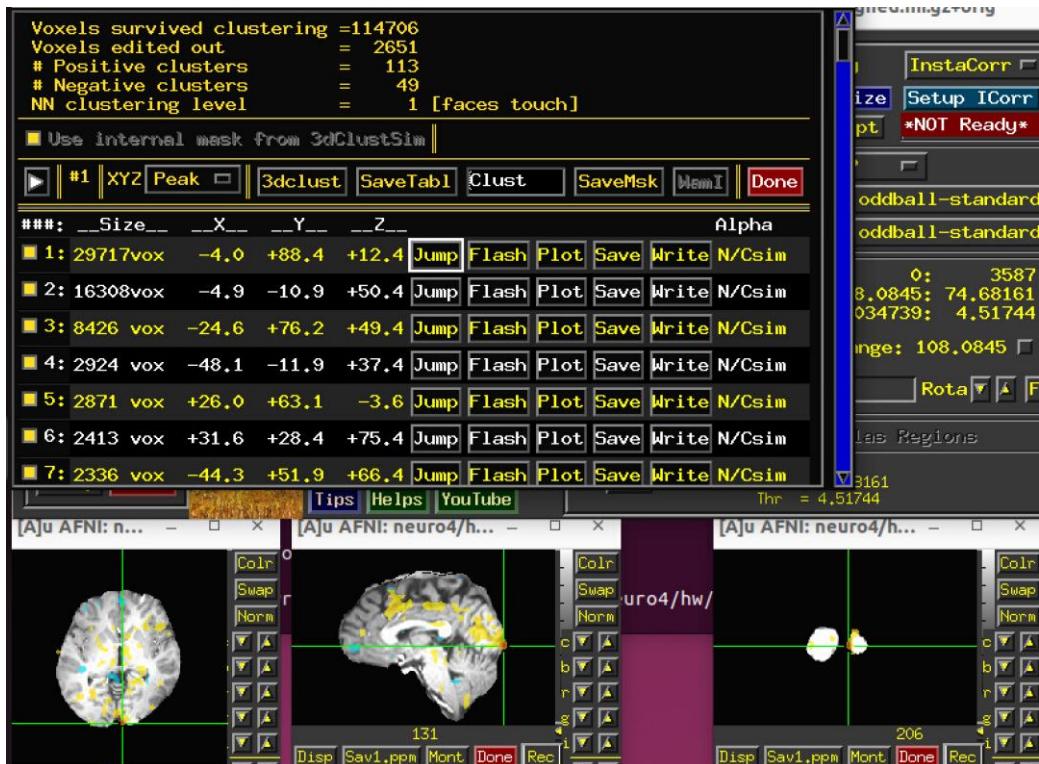
ریسپانس - خوش ۵ دوم:



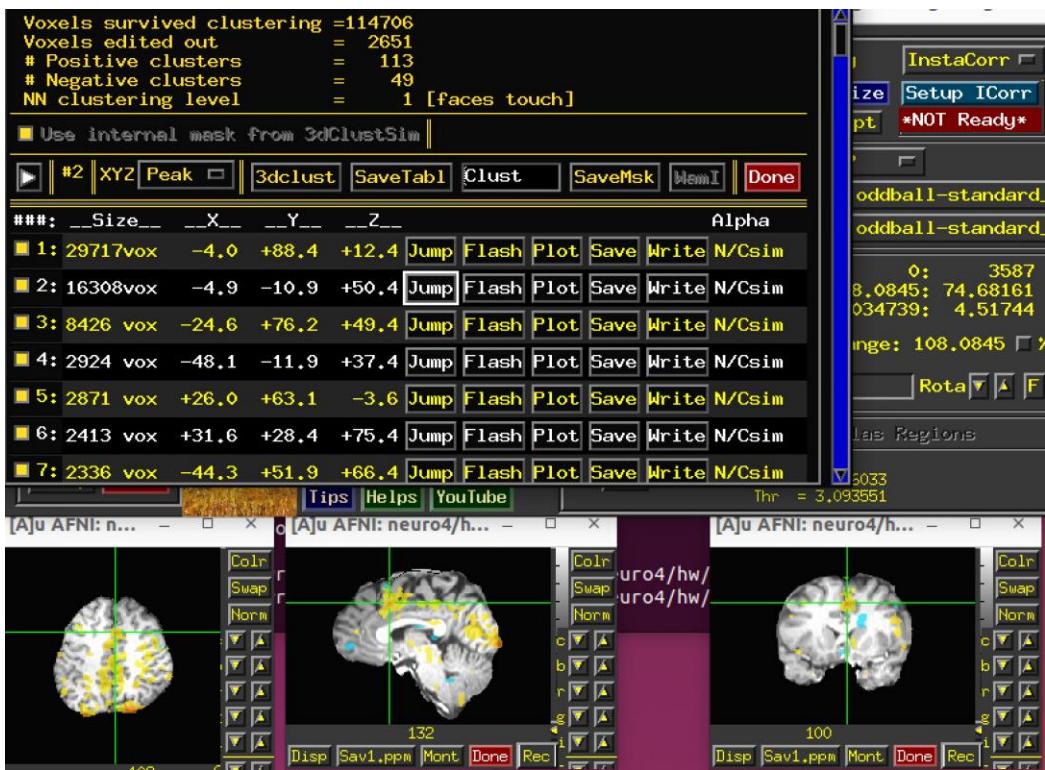
ریسپانس - خوش4 سوم:



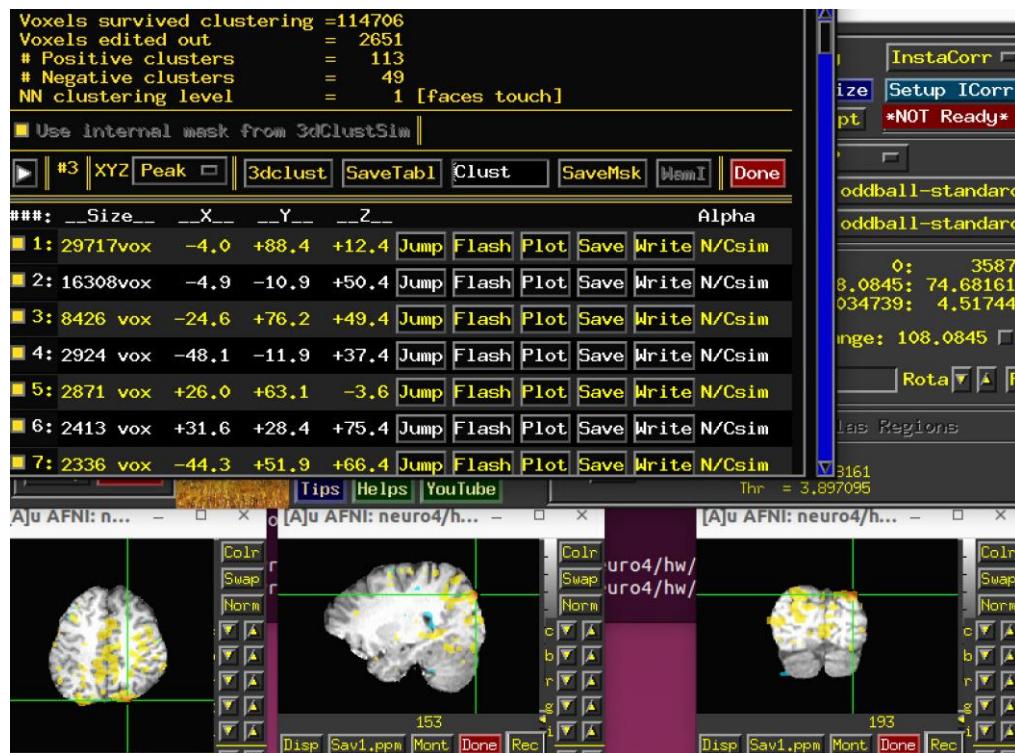
آدبال منهای استاندارد - خوشه اول:



آدبال منهای استاندارد - خوش ۴:



آدبال منهای استاندارد - خوش ۳:



منابع و مشورت ها:

مشورت با آقایان خسروی پور، شفیعی زادگان و نوردی و خانم ها دهقان و طاهری

کد ها و محتوای داخل سایت درس

هلپ داخل ترمینال برای افندی

فیلم های ضبط شده از کلاس

[Welcome to AFNI's documentation! — AFNI, SUMA and FATCAT: v24.1.22 \(nih.gov\)](#)

[AFNI program -help files \(nih.gov\)](#)