Question 1.1

A)

I used the RoBERTa model for this assignment. I selected it over the the GPT2-small model, because during training parts of the sentence are randomly masked and the model has to predict the missing word. This is different from the GPT2-small method of training where training is autoregressive, meaning words of the sentence are input one after the other during training. This means that the RoBERTa model should be better at understanding the meaning of a sentence regardless of whether words are left out or the flow of the sentence is interrupted. Since we are not dealing with clean data i.e. a written story, but instead translated speech-to-text data that was then cleaned for this project, there are some issue with sentence construction and flow of the story. Excerpts from the two stories below highlight this. The intended uses of RoBERTa are for decision making and question response, while GPT2-small is for text generation. I felt that the use cases for RoBERTa were more similar to that of the human brain, which is critical since we will be comparing it to neural data. Additionally I was not a fan of the fact the GPT2-small model output required a seed to standardize. This is done in the extract_embed_pretrain.py script with helper functions from mdl_funcs.py.

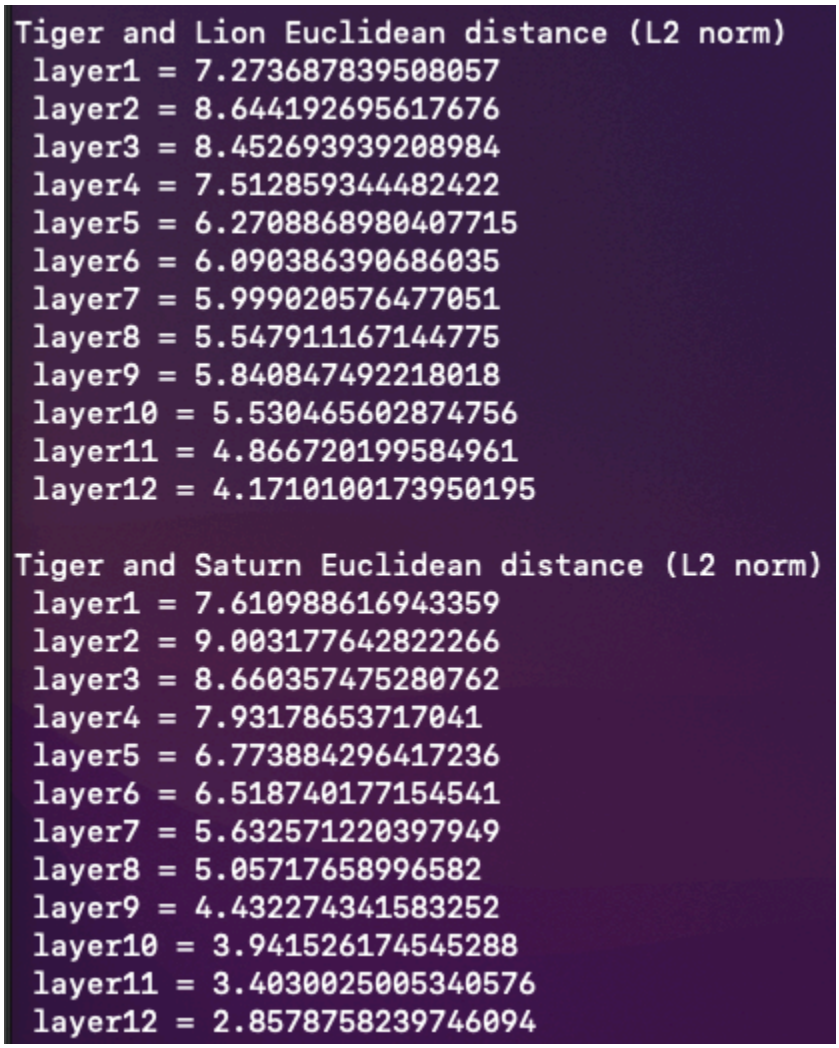
```
adollshouse
alright thank you very much alright this is my story i was dating this girl
like like two ago been dating her for like eight and she to meet my she was
really giving me like how come i haven met your and you know i always just
i never you know i never brought anybody home to meet my i was like thirty
old never met brought anybody home i never i ever like meeting people in
general like i meet her i always when you had to meet like you know go out
with

adventuresinsayingyes
first taking care of other when i was actually still just a child myself so
that meant people were paying me basically to play and that was really
awesome but it also meant that i was getting a ton of experience and that
by the time i was sixteen i probably could make macaroni and cheese with my
shut or change a diaper with one hand tied behind my back which i only
probably ever would have to do with those very naughty but their mother
knew
```

B)

L2 distance for tiger and lion and tiger and saturn for all 12 twelve layers of RoBERTa is shown below. Model embeddings indicate how a word activates the weights of the neural network as it passes through different layers, so the embedding conveys meaning of the word in a high dimensional space. The distance values of the words indicate how similar the words are as they move through the neural network. They start off as more dissimilar in the earlier layers and become more similar closer to the output layers. I would have expected tiger and lion to be more similar than tiger and saturn in the final layer. However this could be because I did not provide a

context sentence and the neural network is attributing similarity to the letters of the word instead of inherent meaning.

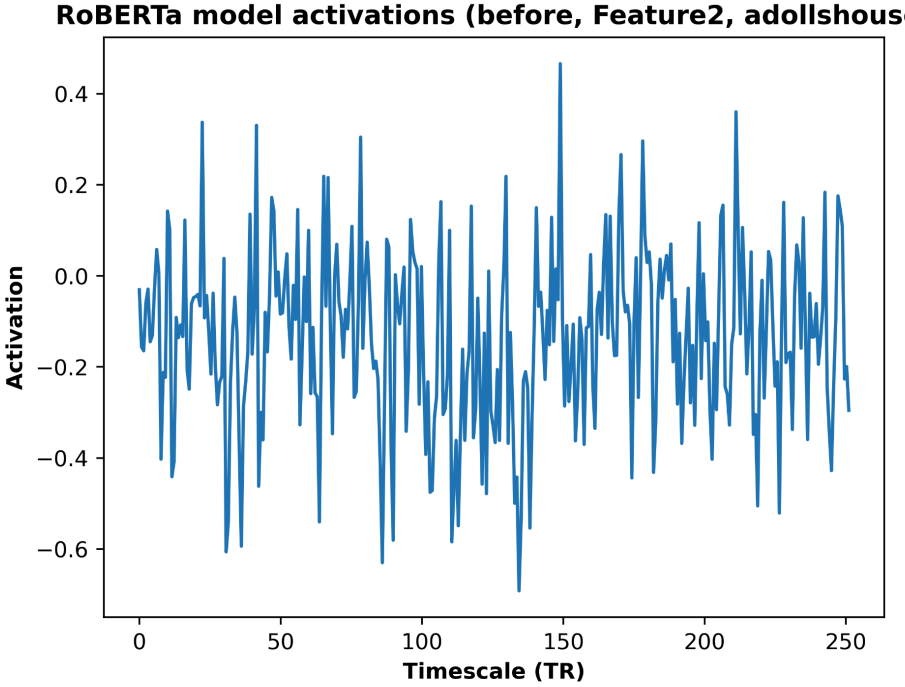


C)

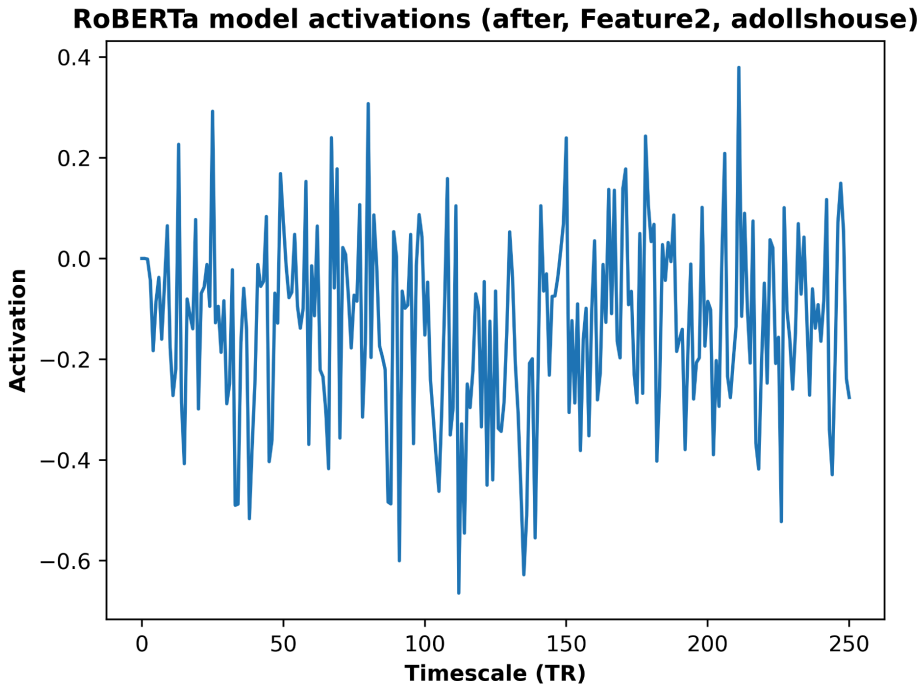
I used the pretrain model for RoBERTa with the RobertaTokenizerFast function so I could see how words are tokenized (token_split function). I then averaged together the embeddings from words that were split during tokenization. I opted to truncate the embeddings because I got an error saying the model_max_length = 512 and the story length exceeded that. Online stated that we could change the mode_max_length if we trained the network from scratch but we were instructed to use pretrain so I truncated instead. Truncation is done by removing a token from the longest tokens until the sequence is under the model_max_length, in this way meaning from all the words in the story are preserved. I made a flag in the function mdl_embed that takes a sentence for context and provides the embedding for a word on the sentence. However, that seems to be giving me errors when using the test words only. So I got the embeddings for the test words tiger, lion and saturn without context.

Question 1.2

Plot of RoBERTa model final layer embedding before it was resampled so that dimension 2 matched that of the time it took to read the stories in the fMRI task.



Plot of the model embedding after it was resampled using Lanczos sampling and shifted back 4 lags so that the time delay matches that produced as a result of the HRF lag in fMRI data. I selected shift=4 because that was recommended in the FIR model section of the Speech tutorial from the Huth lab.



Original RoBERTa model embedding shape, dimension 1 = 3 layers, dimension 2 = word tokens, dimension 3 = width of each layer.

```
torch.Size([3, 328, 768]) = adollshouse
torch.Size([3, 394, 768]) = adventuresinsayingyes
torch.Size([3, 333, 768]) = afatherscover
torch.Size([3, 288, 768]) = againstthewind
torch.Size([3, 384, 768]) = alternateithicatom
torch.Size([3, 356, 768]) = avatar
torch.Size([3, 389, 768]) = backsideofthestorm
torch.Size([3, 380, 768]) = becomingindian
torch.Size([3, 400, 768]) = wheretheressmoke
```

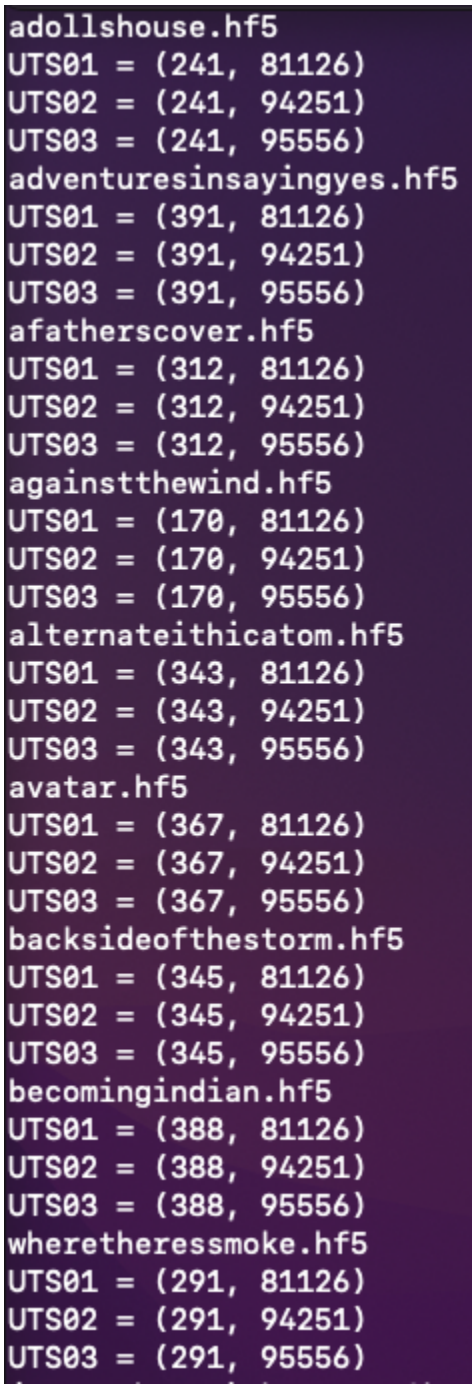
Time, in repetition time (TR) of fMRI task, taken from TextGrids file xmax for each story and divided by TR=2 (found in the json file)

```
(pytorch_env) kanestor@kanestors-MacBook-Pro hw1 % python align_neural_data.py
[251, 401, 322, 180, 353, 377, 355, 398, 300]
```

Model embedding shape after shift=4, resampling (Lanczos sampling) to match fMRI delay and task time (TR). Dimension 2 is still the word tokens but will be considered as time in TR to match the fMRI data. Data normalized using `stats.zscore`, `np.mean` = -1.179519813678785e-18 and `np.std` = 1.0.

```
(3, 251, 768) = adollshouse
(3, 401, 768) = adventuresinsayingyes
(3, 322, 768) = afatherscover
(3, 180, 768) = againstthewind
(3, 353, 768) = alternateithicatom
(3, 377, 768) = avatar
(3, 355, 768) = backsideofthestorm
(3, 398, 768) = becomingindian
(3, 300, 768) = wheretheressmoke
```

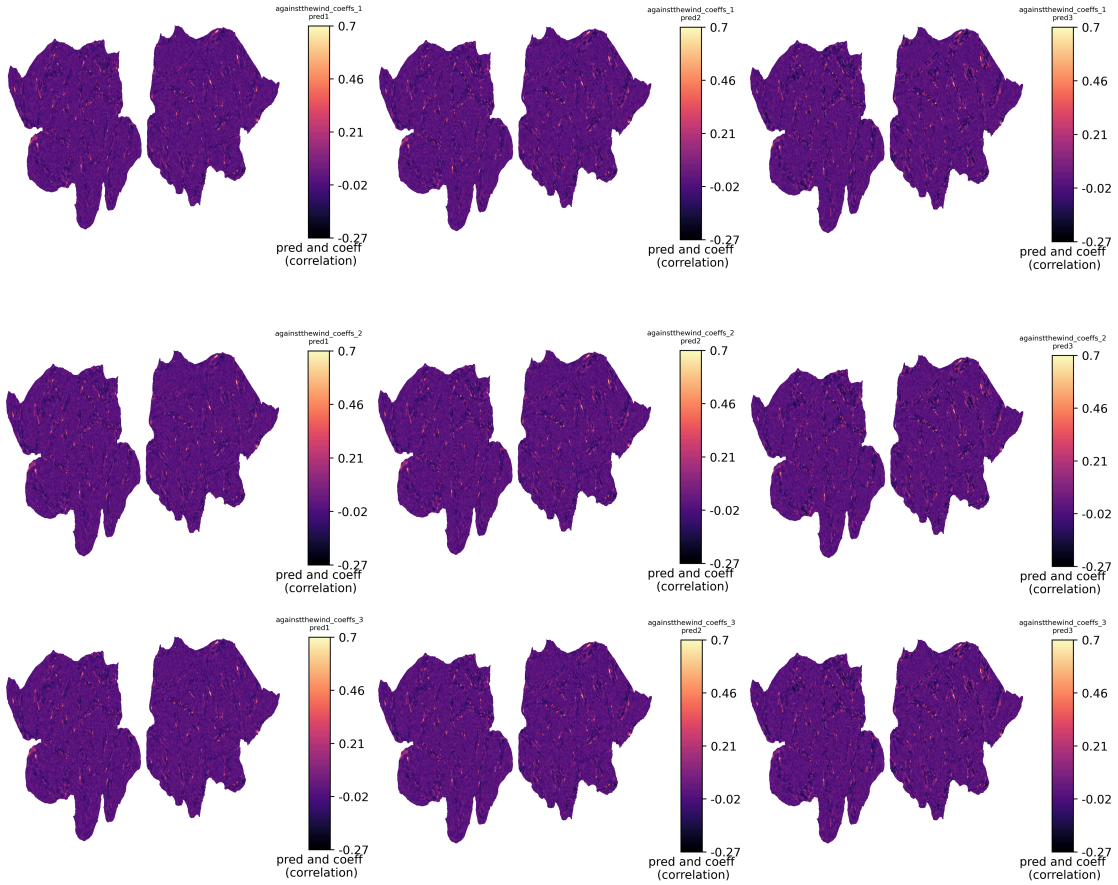
fMRI data shape before resampling to match model embeddings. Data originally preprocessed to trim 10 TR before and after for noise and dimension 2 indicates masked pycortex 'thick' mask for only cortical voxels. Checked using np.mean (-4.221528264873832e-17) and np.std (0.9999999999999999), fMRI data was already normalized, so no need to do this again. Dimension 1 = time in TR, Dimension 2 = number of cortical voxels.



fMRI data after resample to match fMRI timescale in dimension 1 = subjects, dimension 2 = fMRI timescale in TR and dimension 3 = resampled cortical voxels to match the width of RoBERTa model layers.

```
(3, 251, 768), fmri after  = adollshouse
(3, 401, 768), fmri after  = adventuresinsayingyes
(3, 322, 768), fmri after  = afatherscover
(3, 180, 768), fmri after  = againstthewind
(3, 353, 768), fmri after  = alternateithicatom
(3, 377, 768), fmri after  = avatar
(3, 355, 768), fmri after  = backsideofthestorm
(3, 398, 768), fmri after  = becomingindian
(3, 300, 768), fmri after  = whereetheressmoke
```

Question 1.3

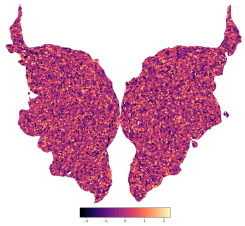


The plots above show the correlation between the three subjects in the test set story **wherethereissmoke** model embeddings, used to predict fMRI matrices and the regularization coefficients for each story used during training. Shown above are the correlations for the test set predictions and **againstthewind** coefficients. The images for the other pairwise correlations can be found in the zip file submission. In the above image each row has the same coefficients, while each column has the same prediction correlation. Overall there are more positive correlations with predictions made from subject UTS01 (pred1) e.g. for each row the first image is lighter in comparison to the other images from that row. When examining the image from the first column (all the first images from each row i.e. pred1), it seems that pred1 has more positive correlations with regularization coefficients from the first subject aka the same subject. It seems that UTS01 is not only the most accurate predictor for fMRI matrices for stories/coefficients produced from that subject but from coefficients produced from other subjects i.e. UTS01 is a generalizable predictor.

To make these images I used the nilearn module with the code below, due to issues with pycortex I elaborate on in the paragraphs below. The same minimum and maximum values were used for all the figures (vmin=-0.27, vmax=0.7), and this was based on the initial auto

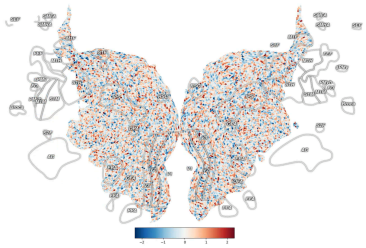
colorbar for all plots and I selected the minimum and maximum values. The colormap used in magma, because although this is a one directional colorbar and we are plotting negative and positive values for the correlation, the extreme colours are black and yellow with purple and red in the middle. I believed these colours provided enough juxtaposition to each other to be easily identifiable as negative and positive values in the image.

```
# plot 2d cortex flat map - in nilearn, save fig
plotting.plot_surf(hcp.mesh.flat, hcp.cortex_data(np.ravel(cor)),
colorbar=True, cmap='magma', vmin=-0.27, vmax=0.7) #
plt.xlabel('pred and coeff\n (correlation)', fontsize=10)
plt.title(f'{m_save_name}\npred{i+1}', fontsize=6)
plt.savefig(f'{output_figs_dir}{m_save_name}_pred{i+1}.png', dpi=500)
# plt.show()
plt.close()
```



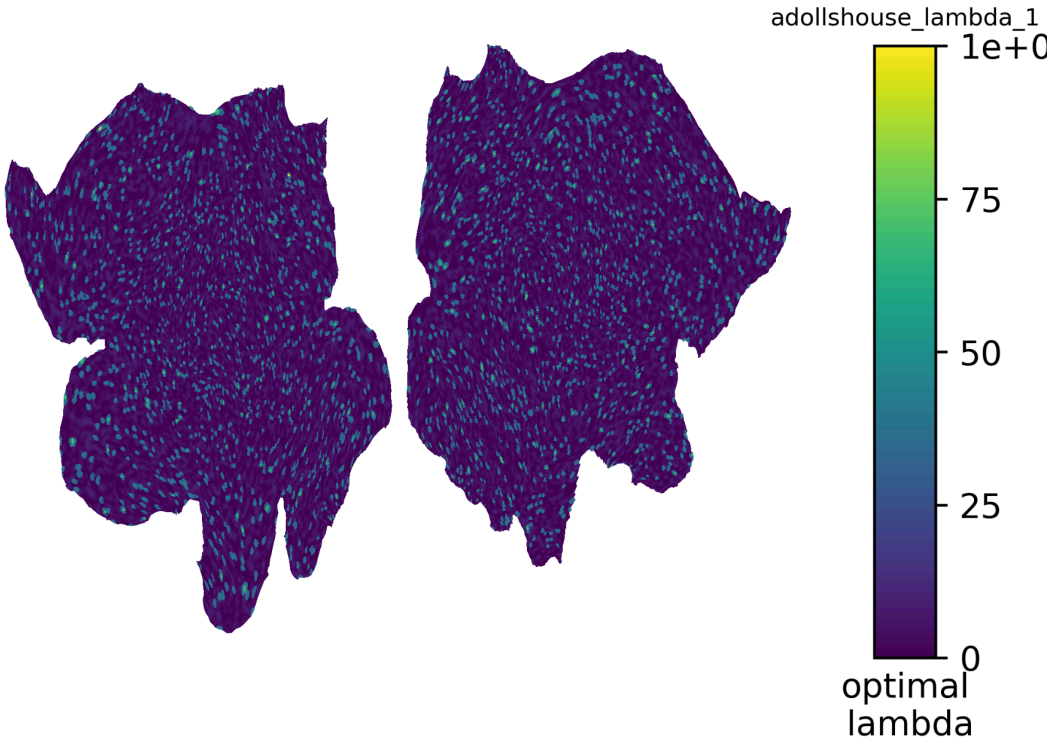
I had a number of issues using the pycortex module, as I noted over email with Prof. Wehbe. I was not able to get the module working on my Mac, the module downloaded but would error out when imported. I was finally able to get pycortex downloaded and imported on my Ubuntu Linux machine. I was able to use the code block below to create the 2d brain flat map shown directly above, based on random input data. When I tried to use cortex.Volume command to input my own data I then got errors again.

```
# plot random flat map in pycortex
np.random.seed(11)
volume = cortex.Volume.random(subject='S1', xfmname='retinotopy',
cmap='magma')
cortex.quickflat.make_figure(volume, with_rois=False)
# plt.savefig(f'{output_figs_dir}pycortex_random_flatmap.png')
plt.tight_layout()
# plt.show()
plt.close()
```

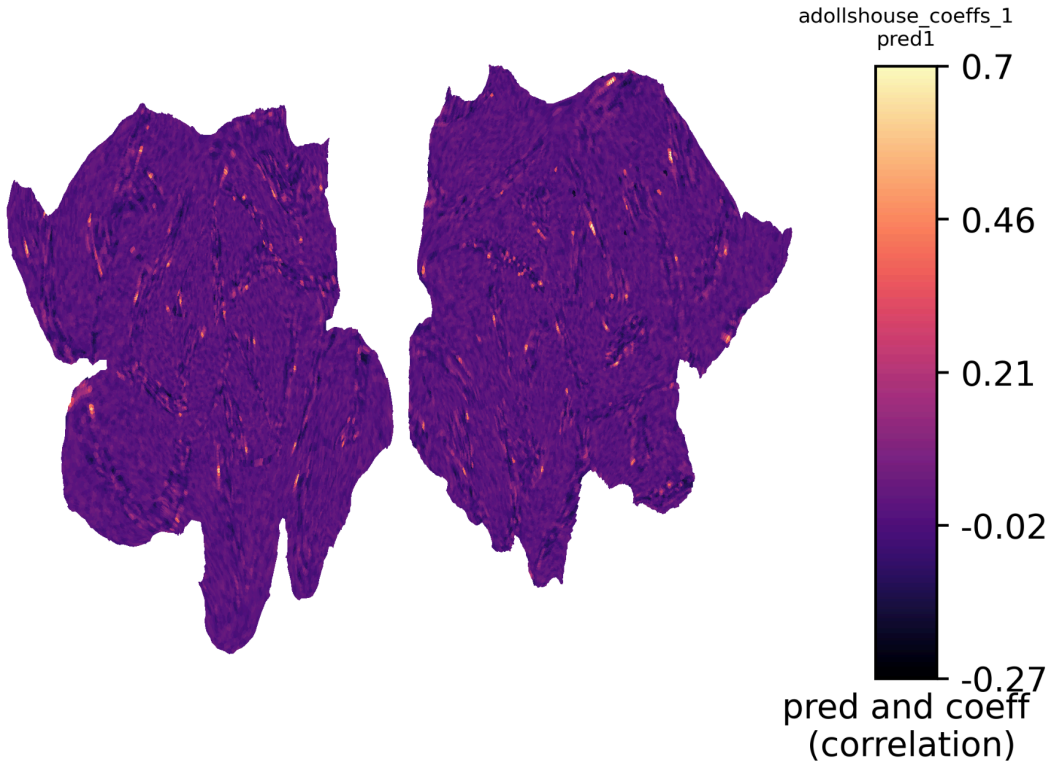


I was initially able to use the code below to input my own data (image above) but when I went to input the correlation data it stopped working and I got errors again. For these reasons I used the **nilearn** module to create the correlation brain flatmap and optimal regularization parameter flatmap.

```
# worked initially in pycortex but now errors out
cortex.quickflat.make_figure(predict_fmri[0])
plt.show()
```



The image above shows the optimal regularization parameter or lambda value output from the ridge regression training (**SimpleCVRidgeDiffLambda**), for subjects UTS01 and the story **adollshouse**. The interpretation from this image is that lambda values closer to 0 are more prevalent and optimal for ridge regression training. The colormap used is **viridis** because there is good contrast between the minimum value colour of dark purple and the maximum value colour of yellow.



The image above shows the correlation between fMRI prediction matrix and the regularization coefficients from the same subject UTS01. This image should indicate performance of the prediction values compared to the training coefficients (adollshouse), when tested on a different story (wherethereissmoke). The image is comprised mostly of positive correlation i.e. lighter colours, with a few darker spots indicating negative correlation. This is in alignment with the pattern of the optimal lambda parameter image where the image is mostly comprised of colours indicating optimal lambda is closer to 0, with a few lighter spots indicating instances where larger lambda values were selected. These larger lambda values could be considered nonoptimal and could be the cause for the negative correlation values on the prediction and coefficient correlation image.

Question 1.4

For this question I used the subject UTS01 RoBERTa model embedding from the final model layer to signify the words of the story avatar (reminder these model embeddings are truncated). We were asked to compare the averaged (across time i.e. axis=0) ridge regression coefficient output vector (signifying single voxel values) to the words of the story in order to identify the words that best match the coefficient vector. To do this I used np.dot to compare the coefficient vector to all the word tokens from the final layer of RoBERTa for subject UTS01. I then got the index of the best match tokens, but since they are truncated this does not match the words from clean_stories. So I then ran the words of avatar through RoBERTa again individually, averaged truncated tokens from the final layer, then used np.corr.coef to determine which words were the best correlated with the top three truncated embeddings used in the encoding model.

These words are related in commonality in the semantic sense of the words and not the literal sense of the words. If they were to be used in a sentence such as the following the words would be highly related to each other: “The **perpetrator continued** their activities until their **identity** was uncovered”. Since RoBERTa was trained to detect semantic reasoning in a sentence regardless of whether the words are listed chronologically it makes sense that these seemingly unrelated words can be related to each other given the proper context. The averaged ridge regression coefficient vector seems to represent the semantic relationing of words from the training set instead of their direct word meaning or distance of word spelling to each other.

```
idx of top 3 best match words (coeffs, embed) = [878, 929, 994]

top 3 best match words (coeffs, embed) = ['continued', 'identity', 'perpetrator']
```

Question 1.5

The plot below shows the representational dissimilarity analysis (RSA) value comparing the fMRI data matrix of wherethereissmoke story to the RoBERTa model layer 11, 12, and 13 embeddings and the predicted fMRI output from the ridge regression model (not trained on wherethereissmoke as that was the test set). Intuitively the model layers closer to the input layer are most like the original fMRI input data i.e mdl_layer_11, whereas the final hidden layer (mdl_layer_13) is least like the original fMRI because it is now representative of the semantic context of the word. Also as expected the original fMRI data has a high RSA value of 0.94 in comparison to the fMRI prediction of wherethereissmoke from the ridge regression model. There is no colorbar for the barplot, I used the colour cyan as it pops as a colour and is different from what I used in previous plots.