# COL786 Research Project Report
# Modelling Visual Field of a mouse using statistical techniques on Neuronal Spiking response to natural movies

**Members:**
Sparsh Chaudhri 2019MT10765 Credit
Anannya Mathur 2019TT10953 Audit

## Background

Information in the brain is represented as action potentials (neuron spikes), which may be grouped into spike trains or even coordinated waves of brain activity. The Spike Response Model is a generalisation of the leaky integrate-and-fire model and gives a simple description of action potential generation in neurons, which are generated when the membrane voltage passes a threshold from below. We aim to use spike responses of visual areas of the brain to model visual fields. This can have profound applications to medical applications, such as in Parkinson's, where patients report a decline in visuospatial cognition, in patients with hallucinations, schizophrenia or coma to make meaningful conclusions from neuronal observations for patients, and more.

## Literature Review

1. The most prominent V1 to MT model was of steady-state responses to spatiotemporally homogeneous stimuli developed by Simoncelli and Heeger. It is a classical model employing biological relationships and electrical mechanics to study and model neuronal activity.
2. Spatiotemporal receptive field model was a three-dimensional model developed by Nishimoto and Gallant in 2011 using neuronal spiking responses to naturalistic movies.
3. Suppressive influences were studied by Cui and the team in 2013, with a detailed study of suppressive inputs. They showed that local excitatory and local+global inhibitory components of input to V1 neurons could predict MT responses.
4. Güçlü and Gerven used Nishimoto's fMRI dataset to model brain activity using RNNs on hemodynamic response in 2017.

## Initial topic and revised objectives:

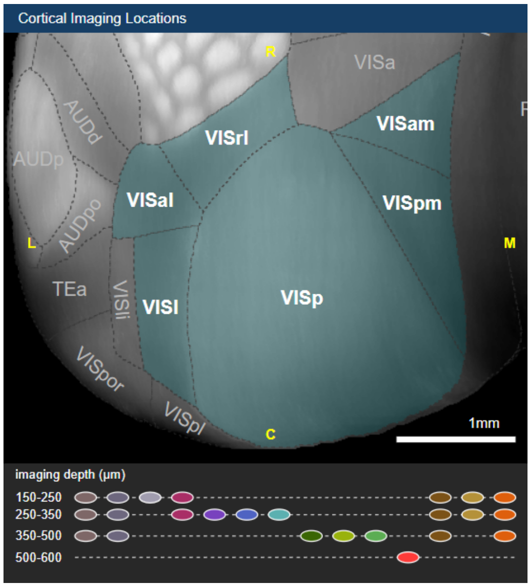
The initial project topic was "Optical flow field modelling through spike analysis using a machine learning framework " and the main objective was to use machine learning models of random forests, clustering and neural networks to model flow fields in an interpretable manner, to be consistent to its GLM model. The original dataset was the mt-2 dataset of Visual Cortex from CRCNS titled "Extracellular recordings from area MT of awake macaques in response to naturalistic movies." The problem that motivated us to deviate from the original topic was that for each neuron, only a subset of the total stimuli was available, which corresponded to the sections where the particular neuron had high activity. Thus stimuli did not match for neurons, and thus, we could not have any data to test our methods on.

We now work with a new dataset: Allen Brain Observatory: Visual Coding, the first standardised in vivo survey of physiological activity in the mouse visual cortex, featuring representations of visually evoked calcium responses from GCaMP6-expressing neurons in selected cortical layers, visual areas and Cre lines. It has neuronal responses to common stimuli, and instead of optic flow estimation, we try to approach its parent problem of image reconstruction. The main challenge is that the stimuli are very short, i.e. the data has neuronal responses to very few frames of stimuli (180 frames, 30 readings per frame).

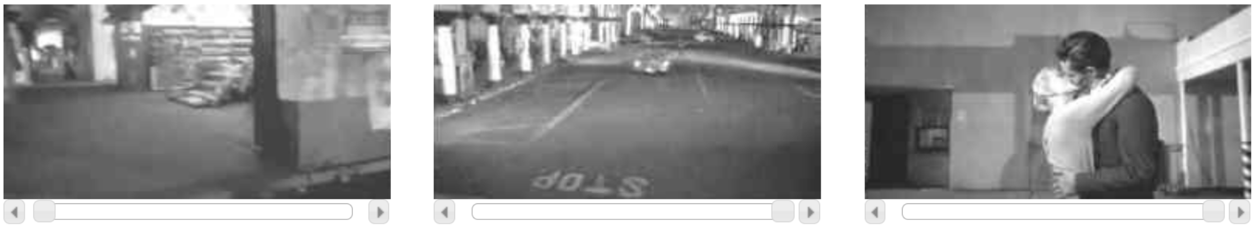
## Allen Dataset

The properties of the subset of data we use are as follows:

1) The gene type of mice of our data corresponds to the CRE_line Emx1-IRES-Cre
2) Visual stimuli: Two 30 second clips and one 120 second clip from the opening scene of Touch of Evil (Zugsmith & Welles, 1958) have been incorporated into a visual stimulus presentation series.
3) Stimuli shape: Each frame of stimuli has 30 spike counts associated with it, which gives shapes of the arrays as:
    a) stim_one.shape=(900,)
    b) stim_two.shape=(900,)
    c) stim_three.shape=(3600,)
4) Number of neurons common to the stimuli and CRE_line = 4811
5) Dimension of each frame = (76,152)
6) All visual areas selected.
7) Train:Test was 80:20 split. Test data was sliced off from the end of the combined stimuli array. As each frame had multiple occurrences in the stimuli, the test frame contained all instances of one stimuli frame.



**Maps of regions of the visual area of mouse brain**



1 second (Touch of Evil) 30 second (Touch of Evil) 120 second (Touch of Evil)

**Visual Stimuli**

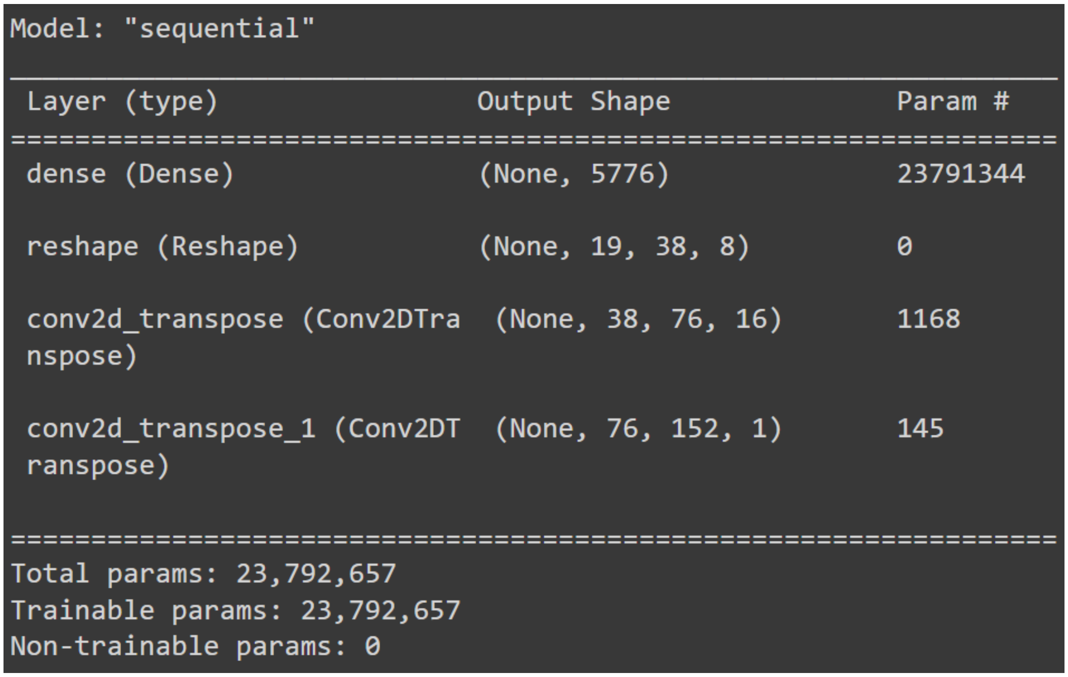
## Models

We made a total of five models to predict the visual field from neuronal spikes.

1. **Deconvolution:**

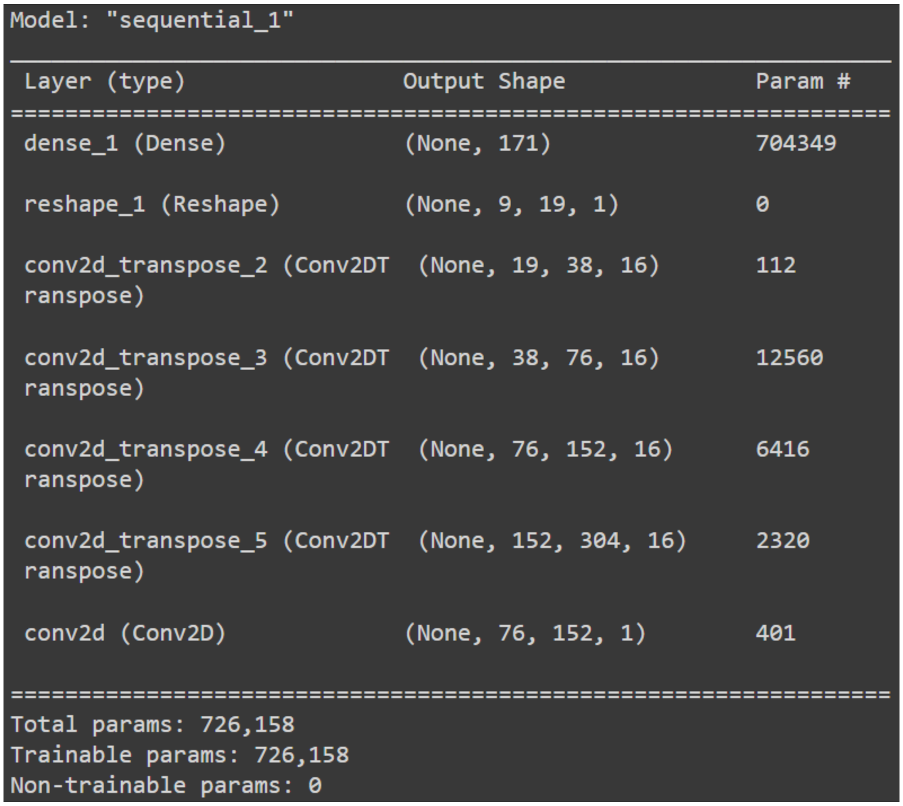
   Neural networks can be used for prediction and in our case, image generation. Using the inverse process of convolution, called deconvolution, we can upsample the input vector of neuronal data through filters to obtain the image. Two neural network architectures were constructed:

   a. **High parameter model:** This model had 5776 outputs from the first dense layer, which were deconvolved to produce the image. The architecture is given below:



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 5776)              23791344

 reshape (Reshape)           (None, 19, 38, 8)         0

 conv2d_transpose (Conv2DTra  (None, 38, 76, 16)       1168
 nspose)

 conv2d_transpose_1 (Conv2DT  (None, 76, 152, 1)       145
 ranspose)

=================================================================
Total params: 23,792,657
Trainable params: 23,792,657
Non-trainable params: 0
```

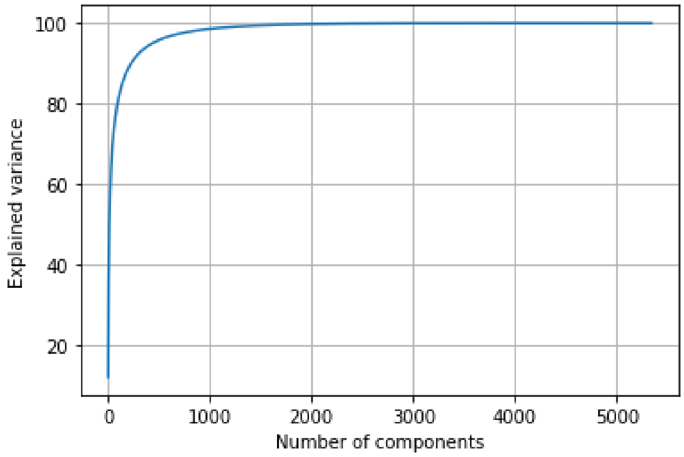
   b. **Low parameter model**: This model had 171 outputs from the first dense layer, which were deconvolved and then convolved again to produce the image. The architecture is given below:



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 dense_1 (Dense)             (None, 171)               704349    
                                                                 
 reshape_1 (Reshape)         (None, 9, 19, 1)          0         
                                                                 
 conv2d_transpose_2 (Conv2DT  (None, 19, 38, 16)       112       
 ranspose)                                                       
                                                                 
 conv2d_transpose_3 (Conv2DT  (None, 38, 76, 16)       12560     
 ranspose)                                                       
                                                                 
 conv2d_transpose_4 (Conv2DT  (None, 76, 152, 16)      6416      
 ranspose)                                                       
                                                                 
 conv2d_transpose_5 (Conv2DT  (None, 152, 304, 16)     2320      
 ranspose)                                                       
                                                                 
 conv2d (Conv2D)             (None, 76, 152, 1)        401       
                                                                 
=================================================================
Total params: 726,158
Trainable params: 726,158
Non-trainable params: 0
```

2. **PCA-GLM estimator:**

Principal Component Analysis (PCA) is a popular dimensionality reduction technique used in Machine Learning applications. PCA condenses information from a large set of variables into fewer variables by applying some sort of transformation onto them. The transformation is applied in such a way that linearly correlated variables get transformed into uncorrelated variables. The first few transformed features (termed as Principal Components) are rich in information, whereas the last features contain mostly noise with negligible information in them.



We used PCA using the following pipeline:

a. Use PCA to generate an orthonormal basis for stimuli.
b. As each stimulus has a different set of magnitudes for basis vectors, through its neural spike the activation of the basis vector in relation to that neuron can be estimated.
c. Make a matrix of the average magnitude of each basis vector (col) corresponding to the time series of each neuron (row). This is done by taking a sum of that particular basis vector's magnitude for a stimulus weighted by the neuronal spike for that stimulus. Each row is divided by the norm of that neuron activation, basically making it a projection.
d. For prediction, project the matrix onto the neuron data vector to get estimates of magnitudes of each basis vector.
e. Take a linear combination of the basis vectors with the magnitude to get a predicted image.

Pseudocode:

```
data <- input where each row is neural spike data vector for each neuron
stimuli <- input where each row is 1D reshaped pixel values for each 2D stimuli
pca_dims <- number of PCA components to break stimuli set into
pca <- object of class PCA(n_components=pca_dims)
pca.fit(stimuli)
stimuli_transformed <- pca.transform(stimuli)
eigenvectors <- pca.components_
n_avg <- np.dot(data, stimuli_transformed )
for i in range(n_avg.shape[0]):
  n_avg[i, :] /= np.sqrt(np.dot(data[i], data[i].T))
frame <- 50
d <- test_data[:, frame]
cs <- np.dot(d, n_avg)/np.sqrt(np.dot(d, d.T))
img <- np.dot(cs, eigenvectors)
img <- np.reshape(img, (76,152))
```

### 3. Factor analysis estimator:

Factor analysis is a statistical method used to describe variability among observed, correlated variables in terms of a potentially lower number of unobserved variables called factors. For example, it is possible that variations in six observed variables mainly reflect the variations in two unobserved (underlying) variables. Factor analysis searches for such joint variations in response to unobserved latent variables and gives factors that are linear combinations of original variables and capture most variance. Factor analysis gives a matrix called loading matrix in which each variable's relation to each factor is given on a magnitude of -1 to 1, with values close to 1 meaning it is included positively in the basis.

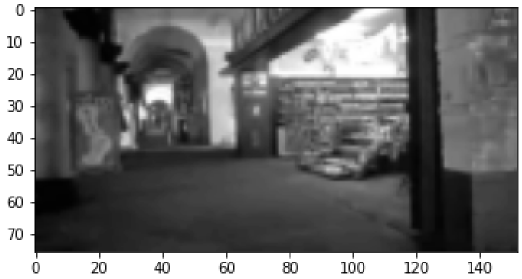
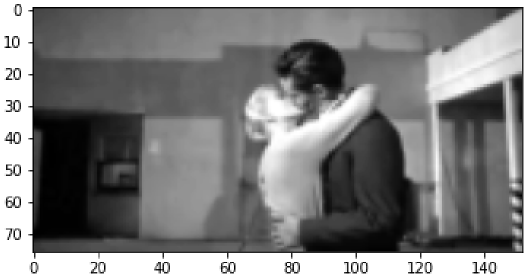
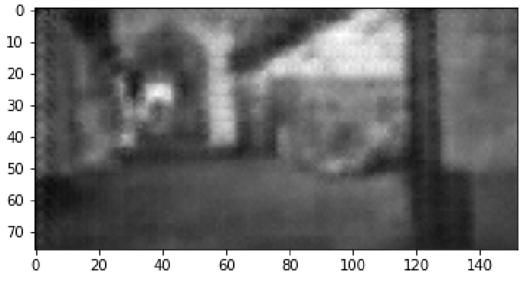
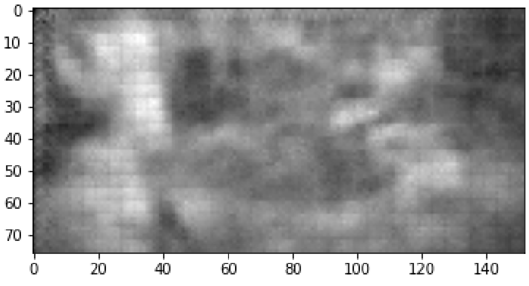
As factor analysis does not generate any eigenvectors, we use a cutoff and for each factor (column), we take only those pixels (rows) which have a magnitude higher than the cutoff in the loading matrix. For simplicity, we assign a value of 1 to those pixels and 0 to others. This way, we get a set of factor vectors that approximately form a basis for the stimuli. To this basis, we use the same approach as the PCA-GLM approach.

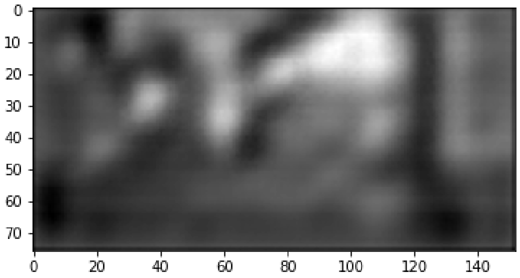
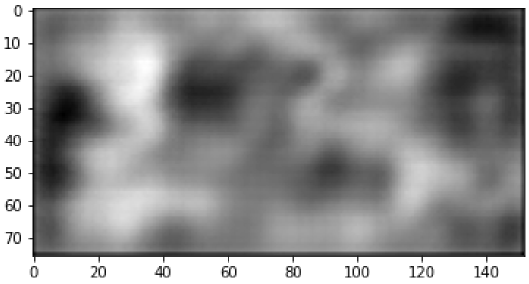
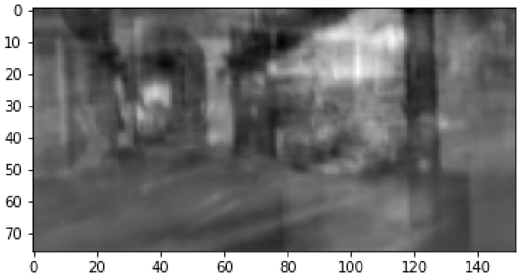
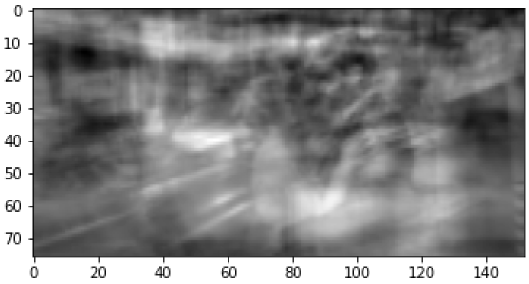
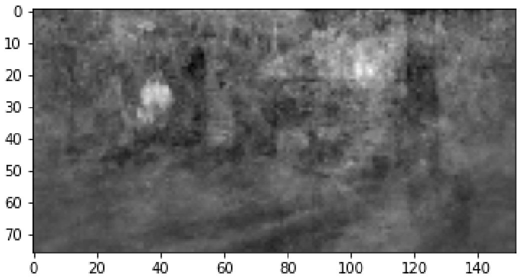
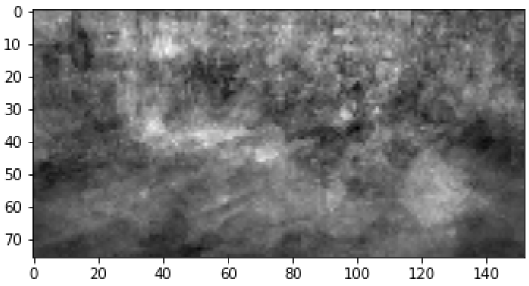
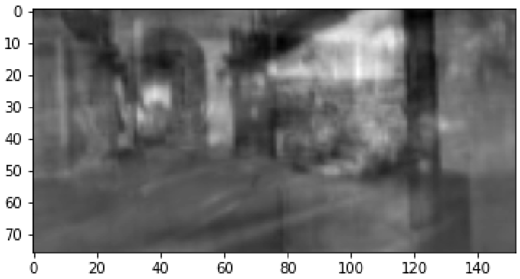
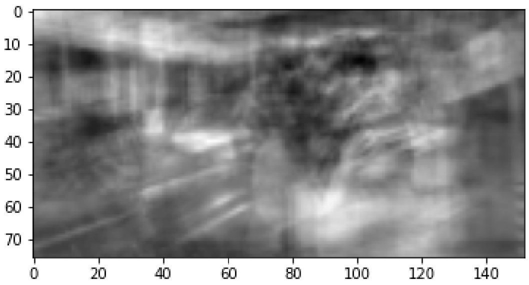
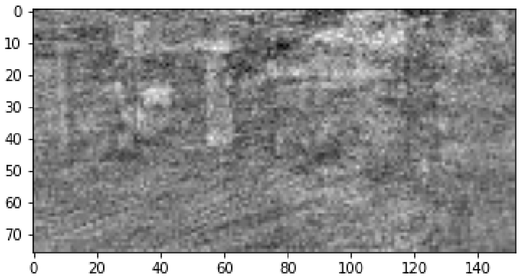
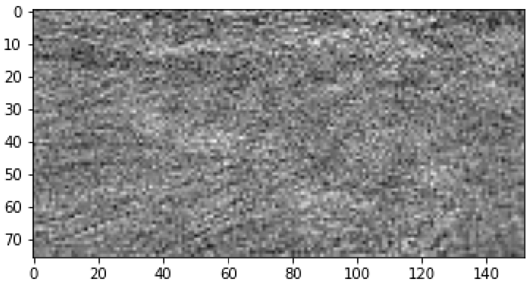
4. **Kernel PCA-GLM estimator:**
   This method is similar to PCA, but the originally linear operations of PCA are performed in a reproducing kernel Hilbert space. The rbf kernel was used by us. Only 500 components were used, and gamma was set to 15.

5. **Apriori PCA-GLM back estimator:**

   This method was similar to PCA, instead of using average magnitudes of the PCA components for each neuron, we can compute each neuron's average spiking in response to each PCA component (basis vector). Effectively for each basis vector, we collect neurons that correspond to it the most. With this knowledge, we can use neuron activations to predict whether that basis will be activated. This approach leads to direct activation of the basis, which can be more useful when the number of input neurons is low and the stimuli are more.

## Results

| | Train set | Test set |
|---|---|---|
| Label |  |  |
| Prediction<br>Method:<br>Deconvolution<br>(High Parameter) |  |  |

| | | |
|---|---|---|
| Prediction Method: Deconvolution (Low Parameter) |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |
| Prediction Method: PCA-GLM |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |
| Prediction Method: Factor Analysis |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |
| Prediction Method: Kernel PCA-GLM |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |
| Prediction Method: Apriori PCA-GLM back estimator |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |  0<br>10<br>20<br>30<br>40<br>50<br>60<br>70<br>0<br>20<br>40<br>60<br>80<br>100<br>120<br>140 |

### Test RMSE on image data scaled to (0,1)

| Deconv (High Parameter) | Deconv (Low Parameter) | PCA-GLM | Factor Analysis | Kernel PCA-GLM | Back estimator |
|---|---|---|---|---|---|
| 4.397272848 | 4.034334821 | 8.108326384 | 3.794570470 | 8.099627053 | 8.077937525 |

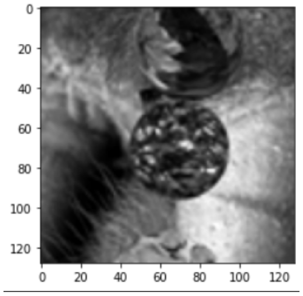
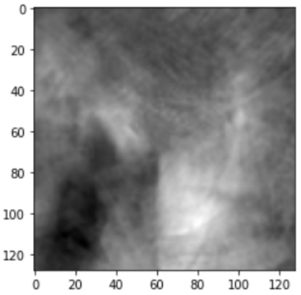
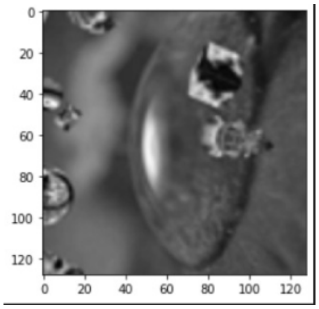
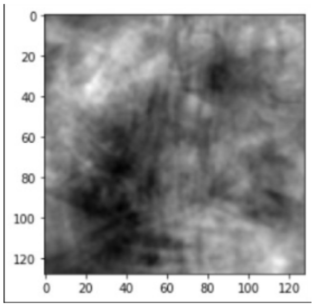
## A solution for the original topic:

One key problem with the original problem was the presence of different stimuli for each neuron. This might be the case, for example, while measuring neurons using electrodes for a patient in response to irreproducible stimuli. Here, we propose that PCA can still be used. If we take the magnitude of response for each neuron in response to stimuli, for the stimuli where the neuron does not have any recorded data, we will give it activation of zero, which will mean that it is inactive to that stimuli. One way is to try to estimate it by taking only those stimuli where it is active.

### PCA-GLM on the mt-2 dataset:

For every neuron, we had different frames of the movie stimuli that covered twice the receptive field of that neuron. To combat the limitations of the dataset, we performed PCA on the stimuli frames and collected mean spike counts for them. We then generated a GLM to establish a connection between the movie frames and spike counts.

**Results:**

<table>
<tr><th></th><th>Target Image</th><th>Reconstructed Image</th></tr>
<tr><td>1.</td><td> </td><td> </td></tr>
<tr><td>2.</td><td> </td><td> </td></tr>
</table>

## Conclusions

Firstly, we observe a stark difference between the quantitative RMSE and qualitative error of the estimators. PCA based methods easily outclass the rest when it comes to visual quality. This is probably due to **artefacts introduced by PCA methods**. **The back estimator does not generate any artefacts but instead introduces a lot of noise (which is absent from the PCA methods). In between the two methods, there is factor analysis, which has a mix of both noise and artefacts, which is why we suspect that it has a lower RMSE score.**

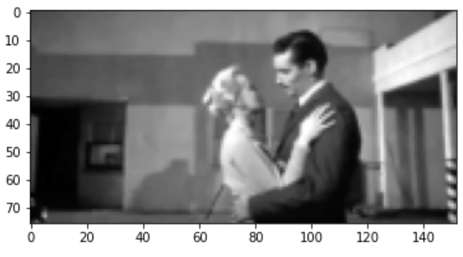
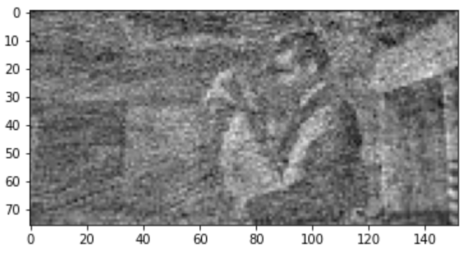
The visual performance offered by the PCA-GLM, and the KernelPCA-GLM estimator is better than neural networks for small datasets. It works best to predict images that have been seen even once in the stimuli set. The high parameter neural network fails due to overfitting, and the low parameter neural network fails due to not being able to capture enough information.

Factor analysis shows potential for improvement in two key areas, as currently, the test results are not legible:

1. Increasing the number of factors.
2. Estimation of basis from loading matrix can employ a more complex method to generate a better basis. The current method is way too simplistic.

Kernel PCA-GLM, found to be at least as great as PCA-GLM, produced good results. If parameter gamma and the number of components are tuned further, it might lead to better results.

A modified PCA and the back estimator can solve the problem of unrelated stimuli by relating them through PCA basis vectors. Tuning might be needed for the back estimator to work with the test dataset. A good image estimation of an example from the training set is shown below:

| Label | Reconstructed Image |
|---|---|
|  |  |

GLM cannot be used for optic flow estimation since flow vectors form a singular matrix, thus leaving scope for future work in this domain. Due to highly positive or negative or 0 values in the flow vectors matrix, the python libraries apparently create NaNs/infs while computing inverse, which leads to failure of GLM.

**To conclude, we would recommend further research into combining the PCA-GLM and the back estimator to produce meaningful results. Factor analysis may also be combined with them in an attempt to improve the models further. The combination approaches may utilise other machine learning frameworks, such as GANs through transfer learning.**

## References

1. Katherine R. Sherrill et al, Functional connections between optic flow areas and navigationally responsive brain regions during goal-directed navigation, NeuroImage 118 (2015) 386–396
2. Shinji Nishimoto and Jack L. Gallant, A Three-Dimensional Spatiotemporal Receptive Field Model Explains Responses of Area MT Neurons to Naturalistic Movies, The Journal of Neuroscience, October 12, 2011
3. Functional Parcellation of Mouse Visual Cortex Using Statistical Techniques Reveals Response-Dependent Clustering of Cortical Processing Areas
4. Jonathan W. Pillow et al, Spatio-temporal correlations and visual signalling in a complete neuronal population, Nature Vol 454| 21 August 2008
5. Thomas Naselarisa, Kendrick N. Kayb, Shinji Nishimotoa, and Jack L. Gallanta, Helen Wills Neuroscience Institute, University of California, Berkeley, CA 94720, USA Encoding and Decoding in fMRI