Investigating the optimal model order of an MVAR model to describe cortical connections in the brain

Authors: Parker Fortier, Matt Henningsen

Abstract

The human brain is a network of many interconnected neurons, and its function depends on interactions between its different areas. The brain's activity can be measured using electrocorticography, which measures and records the electrical activity taking place within the brain by placing electrodes directly onto a subject's brain. A multivariate autoregressive (MVAR) model is used in this study to describe cortical interactions while subjects are sitting and awake in a resting state. Even though the brain is not fully causal, a MVAR model applies here because it takes a weighted sum of the past few outputs to predict the current output. However, the main investigation of this study is the optimal model order of the MVAR model, which is the number of previous timepoints used when predicting the output. The optimal model order is determined in this study by minimizing the mean squared error computed from the 10-fold cross validation between the predicted output and the actual data. This is a suitable method to try to have the model order capture only the entire causal influence of the brain's connections. The predicted output is computed by applying ridge regression between the previous 'p' outputs of all the channels to predict the next output of each channel. The method of regularization is also analyzed in this study. These different methods of normalizing the regularization parameter were investigated for the selection of the optimal model order: i) Scaling by the norm of the output vector; ii) Scaling by the length of the output vector; iii) Scaling by the trace of the feature matrix; iv) Scaling by the largest singular value of the feature matrix. Overall, we are not able to draw a comprehensive conclusion on an optimal model order across all subjects or channels due to the randomness of the brain, but we find some interesting differences across different regularization techniques

INTRODUCTION

Multivariate Autoregressive Model (MVAR) 1.1

A multivariate autoregressive model can be used to describe a random time-varying process whose consecutive measurements convey information about the system. More specifically, it models the current output as the weighted linear sum of its previous outputs [1]. The formulation of an autoregressive model is shown in Equation 1.

$$Y(t) = \sum_{k=1}^{p} \mathbf{A}(k) Y(t-k) + U(t)$$

Equation 1: Autoregressive Model Formulation [2]

Equation 1 is a representation of how output Y(t) is a sum of the previous 'p' outputs from 'k' timepoints weighted by coefficients A(k), plus some white noise, U(t). In our work, we are the most interested in the model order, 'p' shown in Equation 1. The model order is the number of timepoints back that the model will be using to predict the next output. The goal is that the model order captures only the entire causal influence of the previous channels.

Multivariate autoregressive models have been widely used to study cortical connective properties using intracranial data (see Winterhalder et al., Banks et al. [3] and [4]). While it seems to be a fair method in modeling cortical connections in the brain, MVAR models assume that the data has both a constant mean and is stationary. These assumptions are sometimes infringed when looking at the entirety of the data, certainly when the subject makes a change in his actions, thoughts, or behavior [5]. The brain is not fully causal, but we can identify the correlations between past outputs in the brain with the current output and extrapolate that to be a cortical interaction. Granger Causality, a method similar to multivariate autoregressive modeling, has also been used to understand the organization of neural function using simulated data and on data accumulated from monkeys [6].

History and Importance 1.2

The goal in finding the optimal memory parameter is to ensure that we are not looking back too far to overfit the model to uncorrelated data, while ensuring the capture of the entire causal influence. Several different methods can be used in the selection of the optimal model order: Akaike's information criterion (AIC), Bayesian Information criterion (BIC), and cross validation. However, each of these methods are often inconsistent with each other when using them to

find the optimal model order [8]. In this study we use 10-fold cross validation to pick the optimal model order. To keep it tractable, we started with a limited number of model orders to test across: p = 4, 8, 12, 16, and 20. We later added p= 24 and 28 because we had enough computation power to increase the memory in our regression calculations and saw some subjects and channels had not yet converged to an optimal model order. There has been limited research on the optimal MVAR model order when describing cortical connections in the human brain on a large scale, and very few studies have been done that use a large number of subjects with greater than 20 electrodes attached to the brain during monitoring. In discussions with Dr. Barry Van Veen of the University of Wisconsin-Madison, we learned that his research into finding cortical connections using an MVAR model used a model order of 8. A study using a time-varying MVAR model on simulated data showed that the model order was best when it was large enough to capture the delay of the causal influence, but increasing the model order beyond that had very little effect on the connectivity performance of the model. However, in a this study by Pagnotta and Plomp, they were unable to see any sort of convergence on an optimal model order when looking at real electroencephalographic benchmark data, and their only conclusion was that it was better to pick model order too large rather than too small [7]. We aimed to perform our analysis on a far larger scale to see whether increasing the amount of channels and subjects would allow us to converge on a specific model order across subjects, or across channels within a given subject.

The human brain depends upon an intricate network of neurons and specialized functional areas that work together to make the human body work. It is of interest to see if we can pinpoint these relationships to identify the causal influences throughout the brain. There has been a significant amount of work done to characterize these interactions using the information obtained from the electrical signals emitted from the brain. One study has shown that the interactions seen were different in wakefulness than in sleep [5]. Another showed that cortical functional connectivity varied in different states of consciousness, specifically being asleep versus being under anesthesia [4].

The data to train and test the model was obtained from Dr. Matthew Banks at the University of Wisconsin-Madison, and he obtained the data from several neurosurgical patients who were diagnosed with medically refractory epilepsy. They were undergoing invasive electrocorticographic brain monitoring to identify areas of interest in preparation for a resection surgery. All subjects were undergoing electrophysiological monitoring out of medical necessity, and written informed consent was obtained from all subjects. The subjects were all epileptic, but were not on any antiepileptic drugs and were not cognitively impaired in any way that would affect this study. The data was obtained while patients were awake and in a resting state.

The data obtained from each subject was different in several ways. The subjects' data had between 20 and 200 hundred electrodes (channels) attached to their brains. The data was accumulated for approximately 240 seconds, and was sampled at 250 Hz. Figure 1 shows a sample section of data obtained from one channel from a single subject.

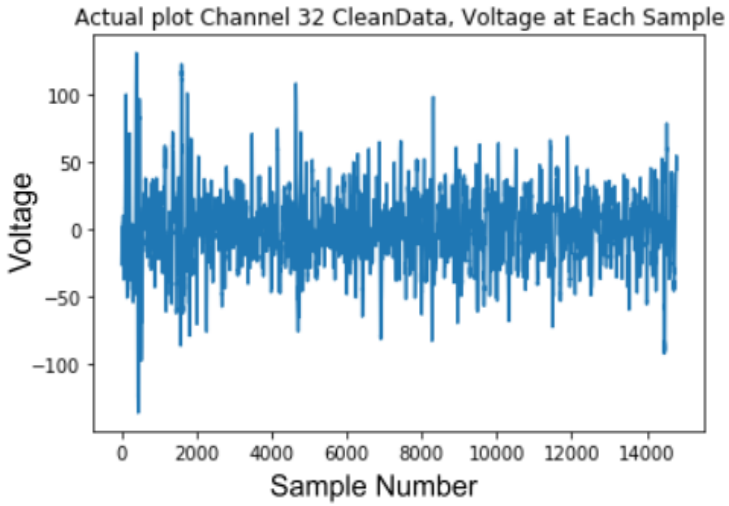


Figure 1: Sample Section of Data

Methods

Mathematical Formulation 2.1

In order to solve for the best weights A(k) in Equation 1, we used ordinary least squares and ridge regression where W = A(K) below. Both ordinary least squares and ridge regression are minimization problems that are seen in Equations 2 and 3 respectively.

$$argmin_w ||XW - Y||_2$$

Equation 2: Ordinary Least Squares

$$argmin_w ||XW - Y||_2 + \lambda||W||_2$$

Equation 3: Ridge Regression

In both of these equations Y is a matrix of the recorded data that we have, where each column is a different channel of the EEG and each row represents a datapoint. Y is time series data. X is the feature matrix that contains the past information that the multivariate autoregressive model uses to estimate Y.

A feature matrix with a memory parameter of one is created by taking Y, inserting a row of zeros at the first row index, and cutting off the last row of data. If the memory parameter is two, the feature matrix from a memory point of one is shifted down and concatenated next to the feature matrix of memory parameter one. This process is repeated for as many memory points as are needed in our feature matrix.

W is how much we need to weigh the past outputs of all of the channels to predict Y. One way to solve the minimization problems in Equations 2 and 3 is by finding a closed form solution using Equations 4 and 5.

$$W = (X^T X)^{-1} X^T Y$$

Equation 4: Closed form solution to ordinary least squares

$$W = (X^T X + \lambda I)^{-1} X^T Y$$

Equation 5: Closed form solution to ridge regression

The closed form solution can be less reliable than other methods of solving Equations 3 and 4 when the choice of lambda is too small because ill conditioned matrices cannot be inverted which makes Equation 4 unsolvable. The lambda in Equation 5 helps condition the matrix to make sure it is invertible; however, if the selection of lambda is too small, we will fail in finding the solution. If an iterative method such as gradient descent is used to solve Equations 2 and 3 they can still potentially find a solution even when using the closed form solution has an ill conditioned matrix.

Besides conditioning the matrix, a sufficiently large regularization parameter also helps to prevent overfitting because it will limit how large the weights are, so that we do not fit the noise into the model.

A large portion of searching for the optimal memory parameter included making sure that we had proper selections of a regularization parameter when solving the ridge regression problem. A good regularization parameter will help ensure that the problem is solvable, help limit the impact of overfitting our data, and will work well for different subjects across each of their channels.

The first option that we considered was a regularization parameter of zero. This is the ordinary least squares problem. While this regularization parameter will not do much to ensure that we can solve the least squares solution or help much to prevent overfitting, it is consistent across subjects and channels.

Having no regularization made the solution very prone to overfitting. We next decided to regularize based on the length of the data. It is important to at least scale the regularization parameter by the data length because if we do not, the ||XW - Y|| term in Equation 3 can become much greater than the lambda term if the data is long. When we do this, the lambda in Equation 3 accounts for the length of the data as well as a scaling factor. For example, one possible selection for the regularization parameter would be length*0.1. This method provides a consistent regularization parameter for all of the channels within a subject, but the final regularization parameter will be different across different subjects. While this is useful since we have data of varying lengths, it does not account for the channels having different magnitudes.

In order to account for both data length and different data sizes we wanted to use the two norm of the data to scale the bias term of the ridge regression formula, so the term becomes a fraction of the size of Y. This would change the problem to take the form of:

$$argmin_w \frac{||XW - Y||_2}{||Y||_2} + \lambda||W||_2$$

Equation 6: Scaling the error term of the ridge regression equation by the two norm of Y

This was not practical for us to do. In order to get a similar formulation we scaled lambda by the two norm of Y to get:

$$argmin_w ||Xw - y||_2 + ||y||_2\lambda||w||_2$$

Equation 7: Scale lambda term by the two norm of Y

This allows the regularization term to be scaled by both the length and magnitude of the data. Notice in Equation 7 that both w and y are now lowercase. This is because w and y are both now vectors as opposed to matrices as a result of doing this calculation one channel at a time. This means that within the same subject, the regularization parameter (two norm of y*lambda) is different for each channel. This took longer to solve, but it helped create more robust results.

During a meeting with Professor Barry Van Veen, University of Wisconsin Madison, on July 16 2021 he stated that in his current research the trace of the feature vector scaled by 0.001 and 0.0001 generally gives good results for feature vectors that have eight memory points. This results in a ridge regression problem as follows:

$$argmin_w ||Xw - y||_2 + trace(X^T X)\lambda||w||_2$$

Equation 8: Ridge regression, scaling the regularization term by the feature matrix trace

Through implementation we found that the trace of the feature matrix increases linearly as we increase the number of memory points that we are using. This is reasonable because the diagonal of the matrix is increasing in length, leading to a longer sum.

In an attempt to find a value that is related to the trace of a matrix, but doesn't increase in value as much as the trace, we decided to use the largest singular value of the feature matrix to scale the regularization term. This also increases as we increase the amount of memory points in the feature matrix. Because the regularization parameter was changing, it made it difficult to conclude anything about what model order was optimal. In order to alleviate this, we chose the largest singular value of the eight memory point feature matrix and scaled it to the same magnitude as the trace of the feature matrix scaled by 0.001 and 0.0001. This resulted in a regularization parameter that was consistent between channels of a subject, but different subject to subject.

The figures showing the mean squared error of the model using the trace of the feature matrix and largest singular value of the feature matrix to scale the regularization parameter are located in Appendix A. They were not used in the selection of the optimal model order because they did not scale the regularization term in a way that was interpretable across different model orders.

Coding Approach 2.2

We had two main methods of solving the various ridge regression problems. The first is the closed form solutions in Equations 4 and 5. This form can be solved directly in python using numpy. This method is also easily converted to be solved on a GPU by using pytorch to solve the closed form formula.

We largely chose not to use the closed form solution, opting to leverage Scikit Learn's Ridge function. Scikit Learn is an open source machine learning library for python [9]. While we were not able to get Scikit Learn's functions to work on a GPU, they offered a great benefit when running on a CPU. It could solve the problems in multiple ways depending on what worked best. If the feature matrix was sparse, it switched to a solver for faster results and if there was no way to get the closed form solution when the matrix was ill conditioned, then it used an iterative method such as gradient descent to find the solution to the minimization problem. In practice, the function largely found the solution via the closed form method; however, we thought that using the Scikit learn library was optimal to create a more robust code base.

The decision to use the Scikit learn library over solely using the closed form solution was solidified after we collected timing information on how quickly we could solve the problem on a CPU vs GPU, this information is shown in Figure 2.

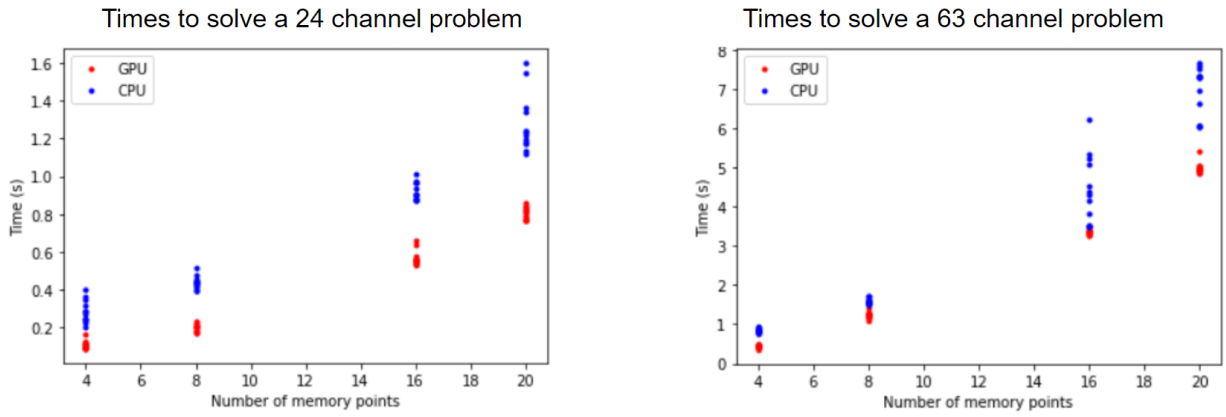


Figure 2: Comparing computation times in CPU vs. GPU

Although the GPU times are faster, it was more practical to parallelize our problem using CPUs because of the resources available to us, particularly HTCondor on the UW campus.

In order to assure that we were not simply fitting the noise, we used 10-fold cross validation when training and testing the coefficients to our model. We calculated the mean squared error for the hold out sets and then averaged them together to get our final results. In cases where we solved for all of the channels at once this resulted in the mean squared error across all channels within a given subject. When results were channel by channel, the channels were then averaged together to get a comprehensive mean squared error for that subject.

Discussion

Results 3.1

Simulation 3.1.1

Before we started testing on real data, we tested our model on simulated data. The simulated data was created using Equation 9 that Ding et al have used previously [6].

$$X_t = 0.9X_{t-1} - 0.5X_{t-2} + \epsilon_t$$
$$Y_t = 0.8Y_{t-1} - 0.5Y_{t-2} + 0.16X_{t-1} - 0.2X_{t-2} + \eta_t$$

Equation 9: Equations to create MVAR data, epsilon and eta are inputs that are white noise [6]

When we solved this using ridge regression, we scaled lambda by the length of the data, but did not do any of the other lambda calculations. This was because we are not using the simulation results to select the optimal memory parameter.

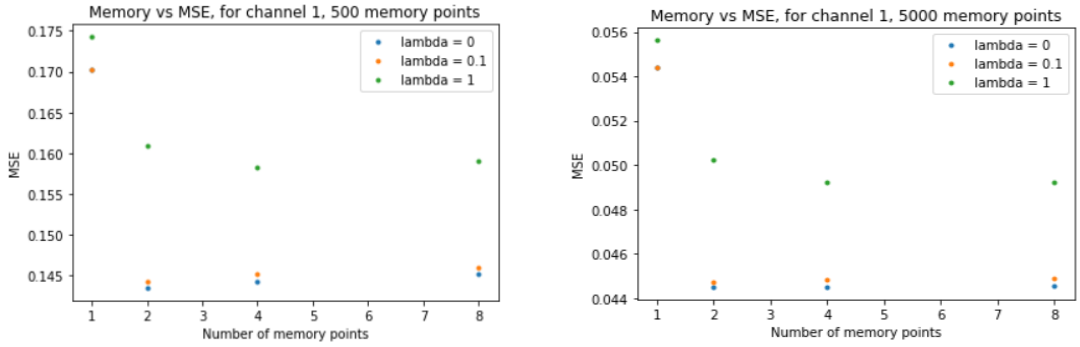


Figure 3: Results from solving with simulated data for 500 and 5000 data points respectively

The results show that picking an excessively large memory parameter will not get the best outcome due to overfitting. When lambda is small, a model order of two is optimal. This is what we expect because the model we used to create the data (Equation 9) is dependent on two memory points. It is also important to note that with the larger lambda, the optimal model order is four. This tells us that choosing too large a lambda may lead us to believe that the optimal memory is more memory points that what it is in reality.

After running the first simulation, we wanted to create more complicated data to test the robustness of our model. Our next data simulation incorporated Equation 10 obtained from Ding et al. [6] to introduce causal influence within the data with white noise added.

$$
\begin{aligned}
X_{1t} &= 0.95\sqrt{2}X_{1(t-1)} - 0.9025X_{1(t-2)} + \epsilon_{1t} \\
X_{2t} &= 0.5X_{1(t-2)} + \epsilon_{2t} \\
X_{3t} &= -0.4X_{1(t-3)} + \epsilon_{3t} \\
X_{4t} &= -0.5X_{1(t-2)} + 0.25\sqrt{2}X_{4(t-1)} + 0.25\sqrt{2}X_{5(t-1)} + \epsilon_{4t} \\
X_{5t} &= -0.25\sqrt{2}X_{4(t-1)} + 0.25\sqrt{2}X_{5(t-1)} + \epsilon_{5t},
\end{aligned}
$$

Equation 10: Complex Data Simulation Formulation [6]

Performing the regression calculations the same way with the new simulated data, Figure 4 shows the results from this regression analysis.

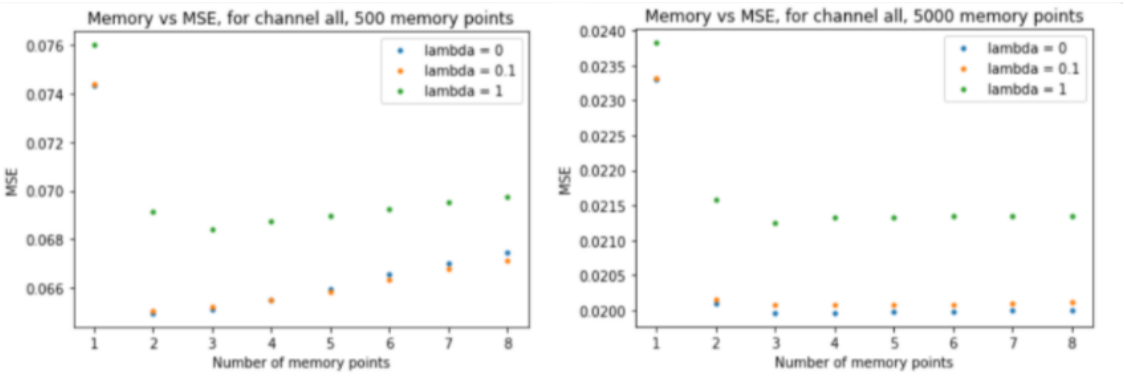


Figure 4: Complex data simulation results

The results in Figure 4 again show the same general shape. The most interesting thing that resulted from this simulated data analysis was the noticeable differences between lambdas. Equation 10 shows that the third channel of X is dependent on the output three timepoints previous. In the plot using 500 data points, the drop in error from p = 2 to p = 3 is evident with the large ridge regression parameter of 1, but there was an increase in error with the smaller regularization parameters. The larger regularization parameter makes the model less sensitive to overfitting, so it is reasonable that the larger regularization parameter is better for this data when you have a smaller number of time points. The results from the smaller regularization parameters of 0.1 and 0 already begin to exhibit overfitting from the added noise at p = 3, causing this simulation to mischaracterize the optimal model order to be two when the regularization parameter is too small. This highlighted the importance of choosing a suitable regularization parameter in our model order selection analysis.

Human Data Testing 3.1.2

After gaining insight from testing the models with known causal influence, we moved on to testing human data. Results were largely similar in shape to that of the simulation when we scaled the regularization parameter by the length of the data, shown in Figure 5, below.

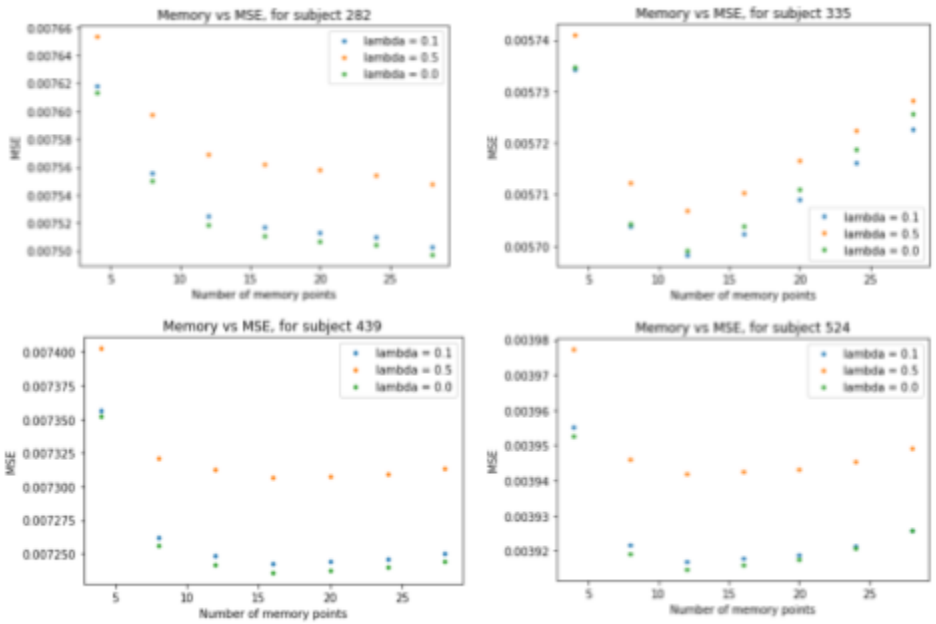


Figure 5: Results using data length to scale Lambda from human data

The graph for subject 282 does not have the bowl shape that is indicative of overfitting. This could be due to subject 282 having less channels, so 20 memory points is still a relatively small model order (20 memory points * 23 channels). The shapes seem to imply that each subject other than 282 has an optimal model order.

In order to better understand how good different mean squared errors are, the variance of the data from the four subjects is in Table 1 below.

Table 1: Average variance across all channels for each subject.

Subject number	Variance
282	676
335	1096
439	14401
524	35712

The variance for each subject is consistently multiple orders of magnitude larger than the mean squared error for the respective subject in Figure 4. This shows that the model accounts for the majority of the variance even in the cases with four memory points.

Adjusting the lambda by length of the data helps ensure that our lambda is robust to inputs of different lengths, but it can still be heavily impacted by data that is of different magnitudes. The results of scaling the lambda by the two norm of Y (Equation 7) yield similar outcomes to scaling lambda by data length.

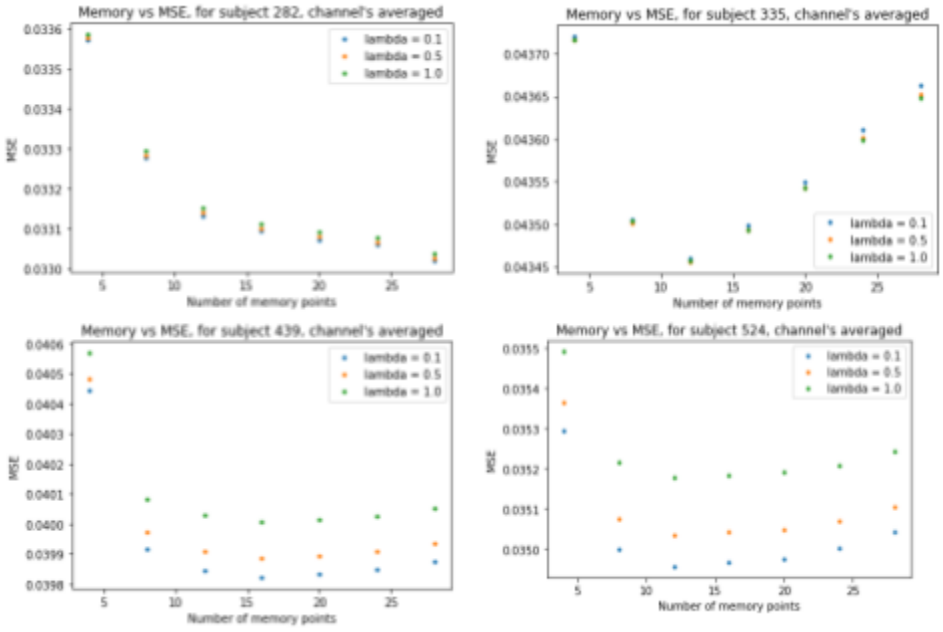


Figure 6: Results using the two norm of Y to scale Lambda from human data

The mean squared error values are much larger using the two norm of Y to scale the regularizer than when only data length is used to scale; however, they are still extremely small when compared to the variances in Table 1. The larger mean squared errors are likely a result of lambda being larger than it was when we were only accounting for data length. Generally the larger lambda will cause error to rise, but will help prevent overfitting and sensitivity to noise.

So far, the mean squared error has been all of the channels within a subject averaged together. When we look at the mean squared error for multiple channels within the same subject we got interesting results shown in Figure 7. These were inconsistent results that made it difficult to draw a definitive conclusion within a given subject.

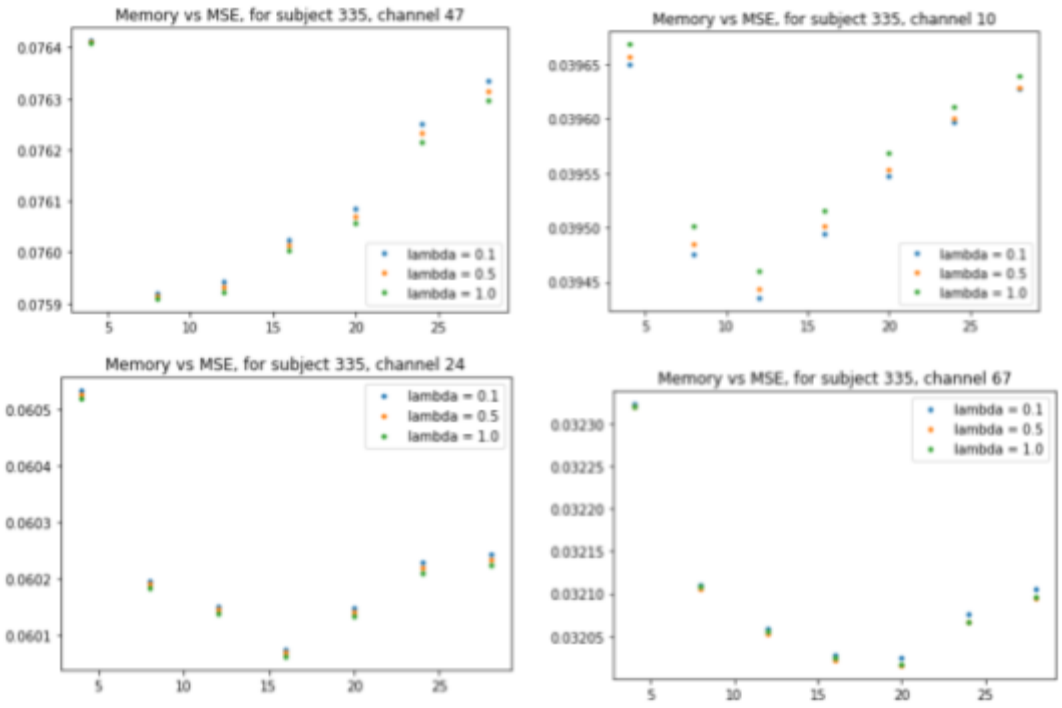


Figure 7: Results using the two norm of Y to scale Lambda from human data, looking at a single channel at a time.

The best memory parameter within a single subject seems to change depending on the particular channel that we are using. This increases the difficulty in selecting one optimal global memory parameter and suggests different areas of the brain are different enough that the best MVAR model to predict them will vary from location to location.

When looking at how the fitted data compared to the actual electrocorticographic brain data, we noticed a peculiar pattern. Shown in Figure 8 are two plots: the left contains a small subsection of real accumulated brain data from a given channel in orange, and the predicted output from our MVAR model computed with a model order of 12 in blue. The right shows the real data again in orange, and the blue plot shows the real data shifted ahead one time point. There are close similarities between our predicted data and the time shifted data which made us notice that the largest coefficient for each channel was the coefficient corresponding to the output in the same channel one time point previous. This is what we expect caused the mean squared error to be so low when comparing it with the variance of the data itself.

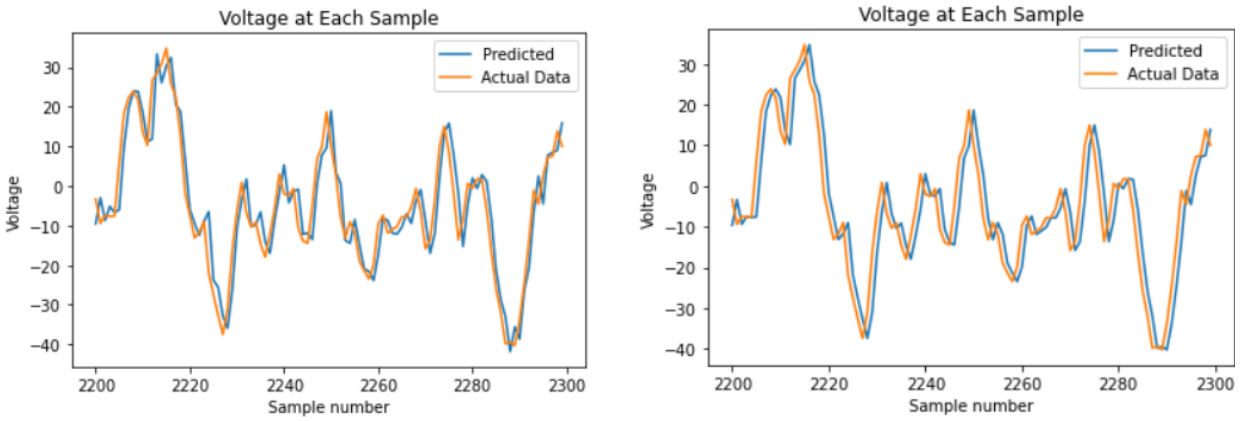


Figure 8: Comparing the predicted output with the actual output lagged one time point behind

After seeing this interesting similarity, we thought it would be an interesting idea to remove the current channel from the regression fit in the MVAR model. A visual representation of the data resulting from this is shown in Figure 9. The error is increased by two orders of magnitude when removing the current channel, and as is evident in Figure 9, the MVAR fitted data with the channel removed is worse.

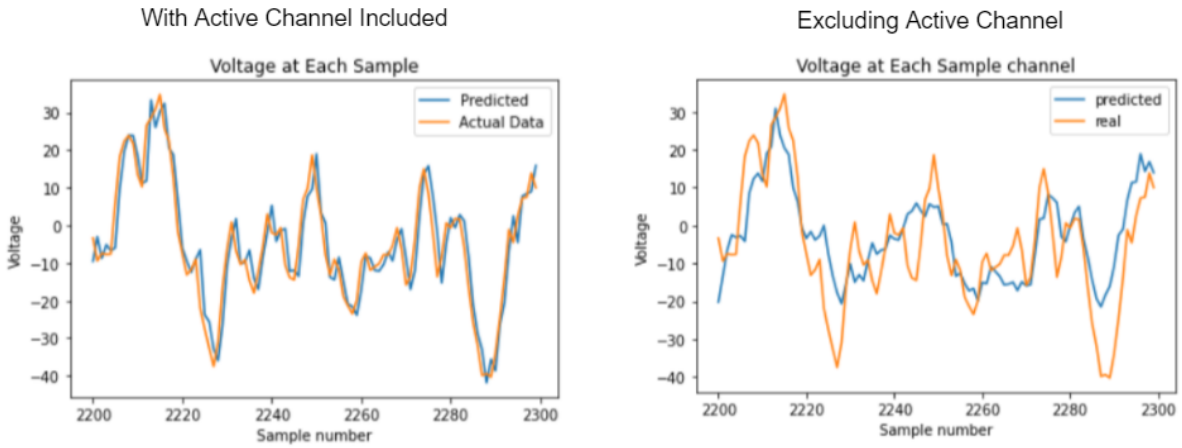


Figure 9: Comparison of fitted data with the active channel included vs. excluded

These results were not used in our analysis in the selection of the optimal model order because in an email (B. Van Veen, personal communication, July 26 ,2021) Professor Van Veen, UW-Madison stated, “The conventional way to think about it is that we trust a connection as being real only if there is no other way the information could have been accounted for. So channel B only connects to A [in] a meaningful way if B explains a part of channel A that cannot be accounted for via any other channel, including the past of A.” This showed that the past

information in the active channel could not be accounted for with only information given from other channels. The error results from the regression calculation with the active channel not included are in Appendix A.

Conclusion 3.2

In conclusion, it appears that each subject and channel is different, so there is not a general rule for what model order is optimal. The global model order that results in the lowest mean squared error subject-wide is not guaranteed to give the best result for each individual channel.

The best case scenario would be to run a parameter sweep that includes number of memory points as a parameter to get what should work best for a given situation. If this is not possible it's probably best to use more than 4 memory points because we saw the largest decrease in mean squared error consistently from 4 to 8 memory points. It is also best to keep the model order low when the optimal model order is unknown because it has the best tradeoff between risk of overfitting and keeping mean squared error low.

Acknowledgments

We would like to thank the following people for all their help in completing this study:

Christina Koch, HTCondor Support

Dr. Barry Van Veen, Guided us in all our work and pointed us in the right direction

Dr. Matthew Banks, Provided a great amount of data to use

Member Contributions

Both team members worked together to make the proposal, presentation, and report. Both were also responsible for understanding all theory and providing input into the project direction.

Parker Fortier: Main contributor to code that solved regression problems, result plotting, and author of the function list

Matt Henningsen: Main contributor to HTC submission code, to the comparisons with/without active channel in regression, and to all data visualization

References

[1] L. Harrison, W. D. Penny, and K. Friston, “Multivariate autoregressive modeling of fmri time series,” *NeuroImage*, vol. 19, no. 4, pp. 1477–1491, Aug. 2003.

[2] B Van Veen, M Banks, “Brain Networks” presented by B Van Veen in ECE697 class at UW-Madison, May 27, 2021.

[3] M. Winterhalder, B. Schelter, W. Hesse, K. Schwab, L. Leistritz, D. Klan, R. Bauer, J. Timmer, and H. Witte, “Comparison of linear signal processing techniques to infer directed interactions in multivariate neural systems,” *Signal Processing*, vol. 85, no. 11, pp. 2137–2160, Jul. 2005.

[4] M. I. Banks, B. M. Krause, C. M. Endemann, D. I. Campbell, C. K. Kovach, M. E. Dyken, H. Kawasaki, and K. V. Nourski, “Cortical functional connectivity indexes arousal state during sleep and anesthesia,” *NeuroImage*, vol. 211, p. 116627, Feb. 2020.

[5] J.-Y. Chang, A. Pigorini, M. Massimini, G. Tononi, L. Nobili, and B. D. Van Veen “Multivariate autoregressive models with exogenous inputs for intracerebral responses to direct electrical stimulation of the human brain,” *Frontiers in Human Neuroscience*, vol. 6, Nov. 2012.

[6] M. Ding, Y. Chen, and S. L. Bressler, “Granger causality: Basic theory and application to neuroscience,” *Handbook of Time Series Analysis*, pp. 437–460, Feb. 2008.

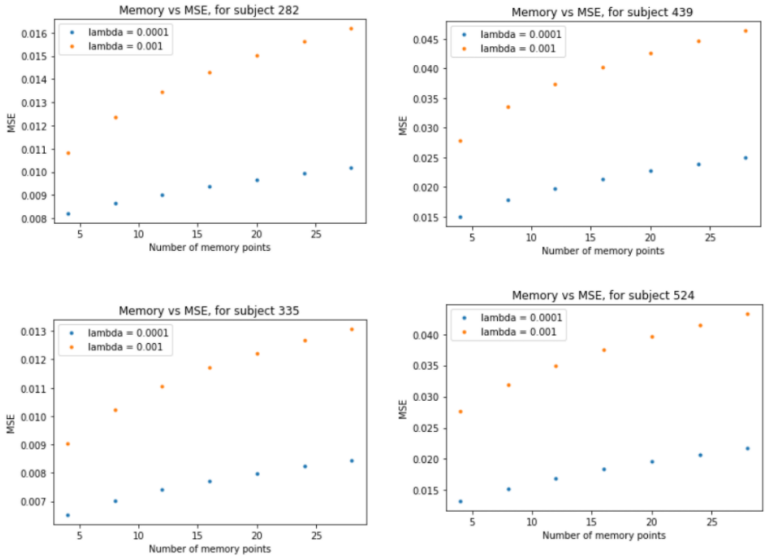
[7] C. Porcaro, F. Zappasodi, P. M. Rossini, and F. Tecchio, “Choice of multivariate autoregressive model order affecting real network functional connectivity estimate,” *Clinical Neurophysiology*, vol. 120, no. 2, pp. 436–448, 2009.

[8] Pagnotta MF, Plomp G (2018) Time-varying MVAR algorithms for directed connectivity analysis: Critical comparison in simulations and benchmark EEG data. PLoS ONE 13(6): e0198846. https://doi.org/10.1371/journal.pone.0198846

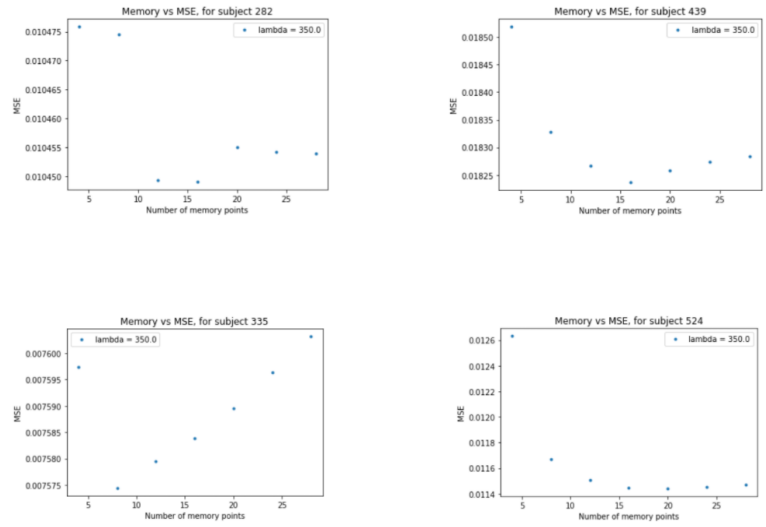
[9] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

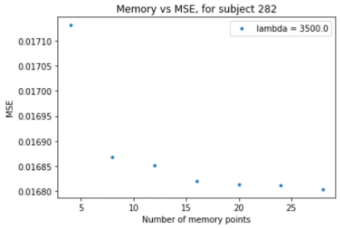
Appendix A

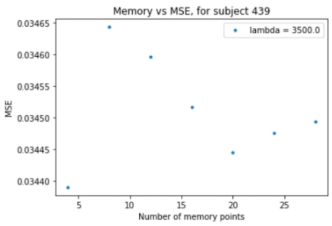
Trace of X'X used to scale lambda:

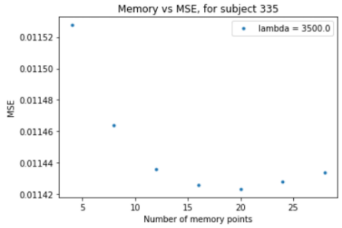


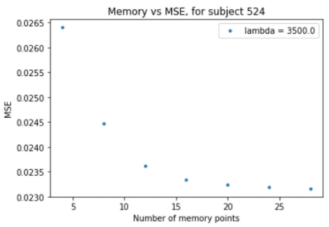
Largest singular value of the model order 8 feature matrix to scale lambda:











The two norm of y to scale lambda with the active channel removed from the feature matrix: