MMAE 500 Final Project:
Dynamic Mode Decomposition Analysis of Poisson Equation with Neumann Boundary Conditions

MMAE 500 Spring 2020

Benjamin Thrun

Table of Contents

1 Abstract

The purpose behind this paper is to look into the powerful method known as *Dynamic Mode Decomposition* (DMD). DMD is used in a wide variety of data applications, with most notably being the field of fluid mechanics. Using a problem taken from MMAE 517: Computational Fluid Dynamics, a solver was created for the Poisson equation with Neumann boundary conditions which changes with each iteration. Taking the data from the iterations and using them in the DMD method demonstrated the ability of DMD to not only properly reconstruct data, but to also look into future state prediction. The results found in this procedure showed a low error between the true data and the reconstructed as well as a low error between the true data and future state prediction.

2 Introduction

The problem explored from Dr. Cassel's MMAE 517 course uses ADI along with Thomas Algorithm to solve a tridiagonal system of equations along each line [1]. The Poisson equation is:

$$\frac{\partial \phi^2}{\partial x^2} + \frac{\partial \phi^2}{\partial y^2} = \cos(m\pi x)\cos(n\pi y)$$

over the interval $0 \leq$ x $\leq$1 , $0 \leq$ y ≤ 1. The constants m and n are odd integers. For the Neumann Boundary conditions

$$\frac{\partial \phi}{\partial x} = 0 \quad at \quad x = 0,\ x = 1$$

and

$$\frac{\partial \phi}{\partial y} = 0 \quad at \quad y = 0,\ y = 1$$

Solving the Poisson equation with Neumann boundary conditions, the solution converges over an amount of iterations until a satisfactory convergence criterion is achieved [1]. The motivation behind Dynamic Mode Decomposition (DMD) is to reduce the system from higher dimensional data, and to reduce the set of modes. DMD is also able to provide insight for how the modes evolve in time. Analysing the problem outlined with DMD techniques will result in demonstrating the powerful characteristics of DMD.

3 Theory

In order to better understand the Dynamic Mode Decomposition method, some theory must be covered. To accomplish this both the variables and the specific steps involved in DMD must be explained and shown in order to have an in-depth understanding of the method that will be used.

3.1 Dynamic Mode Decomposition Variables

The variables that will be discussed and used throughout the DMD procedure are listed below. These variables will be used to interpret the data set. The purpose of this table is to provide a reference to refer back to if there is doubt on the meaning of a specific variable.

Symbol	Definition
A	Matrix representation of discrete-time linear dynamics
$\tilde{A}$	Reduced dynamics on POD subspace
b	Vecotr of DMD mode amplitudes
m	Number of data snapshots
r	Rank
t	Time
Δ t	Time Step
U	Left singular vectors (POD modes) of X
V	Right singular vectors of X
W	Eigenvectors of $\tilde{A}$
X	Data matrix, X $\in^{n\times(m-1)}$
X'	Shifted Data matrix, X $\in^{n\times(m-1)}$
Λ	Diagonal matrix of DMD eigenvalues (discrete-time)
λ	DMD eigenvalue
Σ	Matrix of singular values
Φ	Matrix of DMD modes, $\Phi \triangleq X'V\Sigma^{-1}W$
ϕ	DMD mode

Table 1: Pressure and Density Values from Interpolation

3.2 Dynamic Mode Decomposition Procedure

To have a better understating of the aims and objectives of the Dynamic Mode Decomposition process, the theory involved must be covered in order to have a more complete view of the bigger picture. Some DMD algorithms rely on uniform sampling of data in time, however the method that will be utilized here works with irregular sampled data and with concatenated data from numerical solutions. The method of DMD which will be used is taken from Brunton & Kutz [2]. There are four steps used in the DMD procedure which illustrate the deconstruction and reconstruction of the data into meaningful information, which can then be used to interpret large data sets. In order to begin DMD, two arrays, X and X' must be defined from a matrix A which contains all of the data. X will contain the data at a time step (t_m) and X' will contain the data at a leading time step (t_{m+1})

$$X = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x(t_1) & x(t_2) & \dots & x(t_{m-1}) \\ \vdots & \vdots & & \vdots \end{bmatrix} \tag{1}$$

$$X' = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x(t_2) & x(t_3) & \dots & x(t_m) \\ \vdots & \vdots & & \vdots \end{bmatrix} \tag{2}$$

The DMD algorithm taken from Brunton Kutz [2], proceeds as follows;

Step 1. Take the SVD of X

$$X \approx U\Sigma V^* \tag{3}$$

Where * denotes the conjugate transpose, $U = \in^{n\times r}$, $\Sigma = \in^{r\times r}$, and $V = \in^{m\times r}$. The rank of the SVD reduced approximation of X is r.

Step 2. The matrix A may be obtained by using the pseudo-inverse of X

$$A = X'V\Sigma^{-1}U^* \tag{4}$$

In practice, it is more efficient computationally to compute $\tilde{A}$, the r $\times$ r projection of the full matrix A onto POD modes:

$$\tilde{A} = U^*AU = U^*X'V\Sigma^{-1} \tag{5}$$

The main take away from this is that the reduced matrix $\tilde{A}$ has the same nonzero eigenvalues as the full matrix A. As such, there is no need to work with the high-dimensional A matrix.

Step 3. Compute the eigendecomposition of $\tilde{A}$:

$$\tilde{A}W = W\Lambda \tag{6}$$

Where columns of W are eigenvectors and Λ is a diagonal matrix containing the corresponding eigenvalues λ_k.

Step 4. The high-dimensional DMD modes Φ are reconstructed using the eigenvectors W of the reduced system and the time-shifted snapshot of matrix X' according to:

$$\Phi = X'V\Sigma^{-1}W \tag{7}$$

With the low-rank approximations of both the eigenvalues and the eigenvectors in hand, the projected future solution can be constructed for all time in the future. By first rewriting for convenience $\omega_k = \frac{ln(\lambda_k)}{\Delta t}$, the approximate solution at all future times is given by:

$$x(t) \approx \sum_{k=1}^{r} \phi_k exp(\omega_k t) b_k = \Phi exp(\Omega t) b \tag{8}$$

Where b_k is the initial amplitude of each mode, Φ is the matrix whose columns are the DMD eigenvectors ϕ_k , and $\Omega = diag(\omega)$ is a diagonal matrix whose entries are the eigenvalues ω_k . The eigenvectors ϕ_k are the same size as the state x, and b is a vector of the coefficients b_k .

In order to compute the initial coefficient values b_k . If we consider the initial snapshot x_1 at time $t_1 = 0$, then (8) gives $x_1 = \Phi$b. The matrix of eigenvectors Φ is generically not a square matrix so that the initial conditions

$$b = \Phi^{\dagger} x_1 \tag{9}$$

4 Method and Results

Now that the theory has been properly covered and explained from Brunton & Kutz [2]. The method used to find valuable results from the problem shown in the introduction will be explained. The Poisson equation solver had an output of a solution matrix after each iteration. Each iteration matrix was stored and then concatenated into a matrix A. Prior to running the DMD algorithm, a simple SVD was run on the matrix A, in order to find the eigen values which demonstrate the amount of energy in each mode.

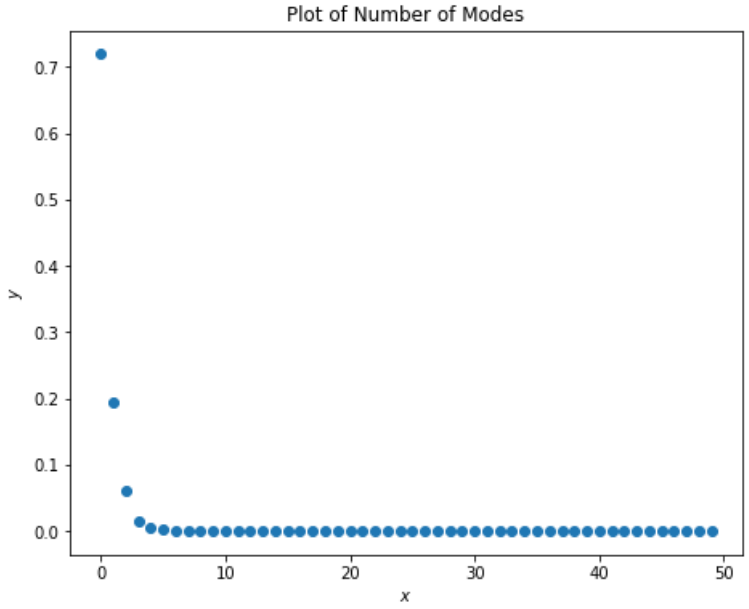

Figure 1: Number of modes for matrix A

Looking at the above figure, the observation can be made that the first four eigen values hold the most amount of the energy of the system. The observation can be made that this is a low rank structure due to the distribution of the modes. Therefore, when pursuing the reconstructed data later on, the rank that will be used, will be truncated, where the rank is 4 rather than the full 29 modes that can be observed in figure (1). Upon running the DMD algorithm laid out in section 3, the left singular vectors or U modes were plotted against the Φ matrix which comes from the DMD algorithm.

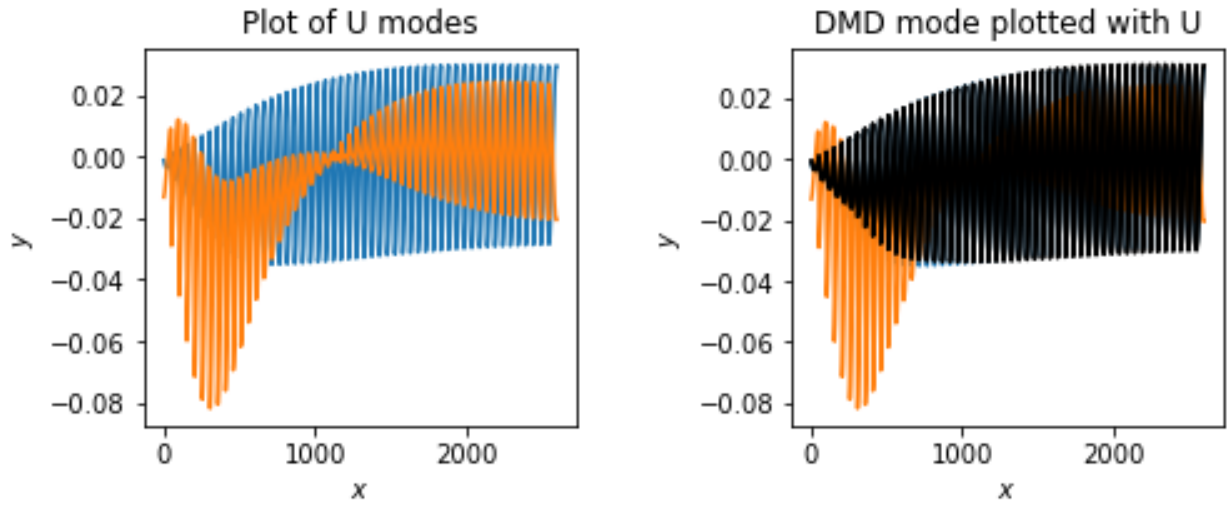

Figure 2: U Mode compared with DMD

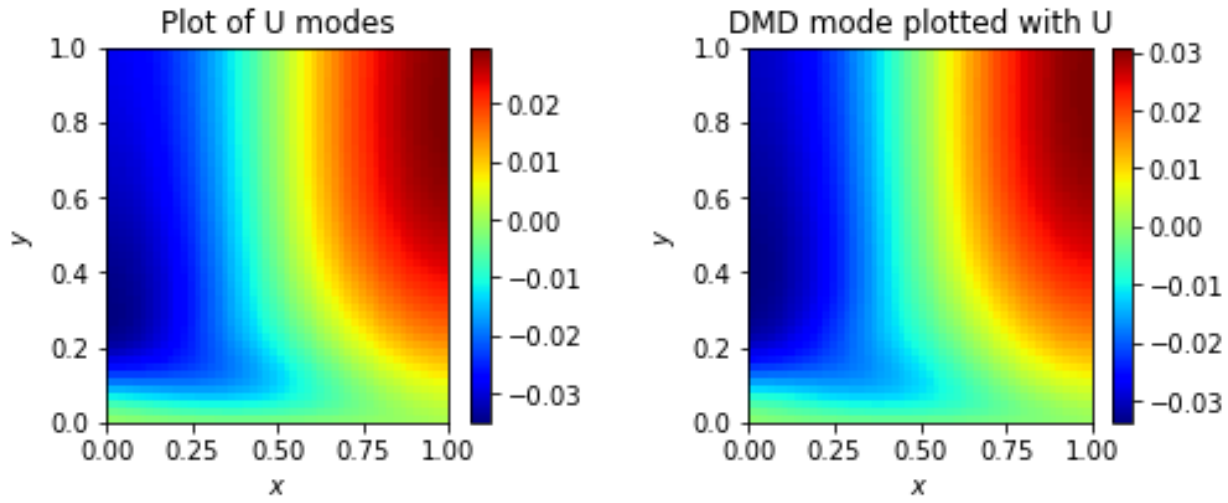

Figure 3: U Mode compared with DMD

The observation can be made that the Φ matrix, or eigenvectors of A perfectly line up with left singular vectors. Looking at the contour plots the relationship between Φ matrix and the left singular vectors is very clear as it is hard to distinguish between the two.

4.1 Analysis of True and Reconstructed Data

The desire to reconstruct data was explored, going back to the theory and using the DMD method to reconstruct the data. Two conditions were explored and plotted: the true data from the original set, and the reconstructed data from the DMD method. The error between the these two sets of data was also computed, such that the error is defined as the norm of the difference between the true and reconstructed data. The "iterations" or time steps that were looked at in this case were for 1, 5, 10, 20, and 40 iterations.

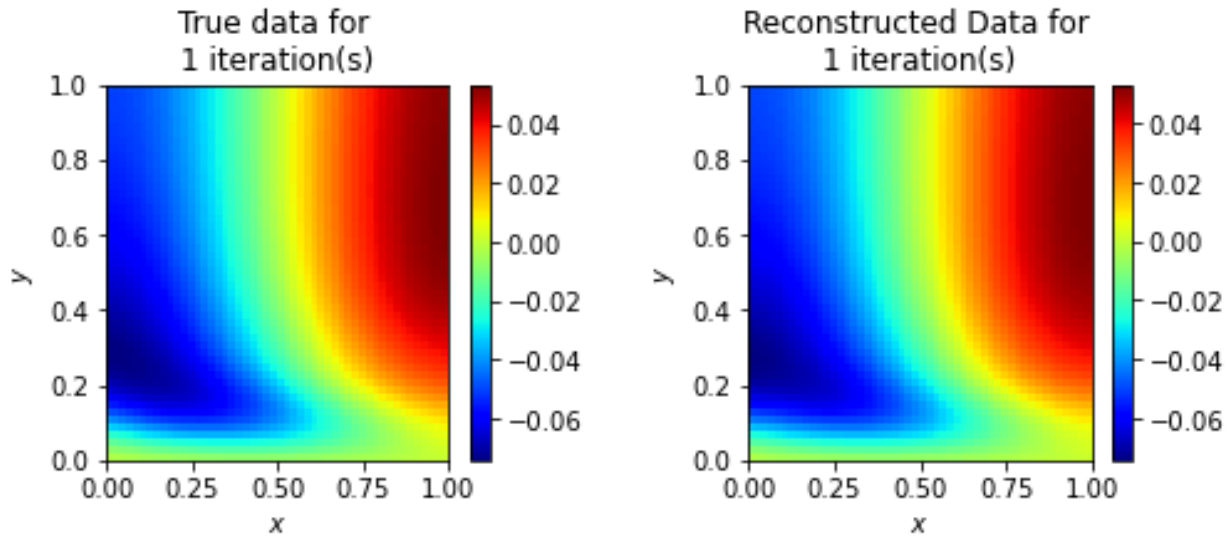

Figure 4: True, and reconstructed plots for the data after 1 iteration.

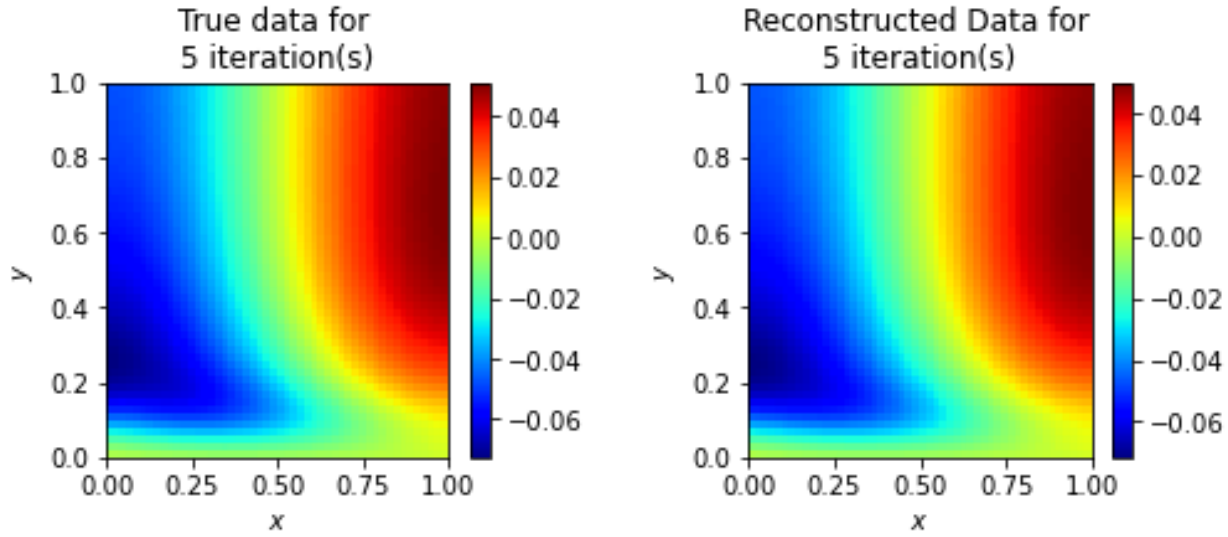

Figure 5: True, and reconstructed plots for the data after 5 iterations.

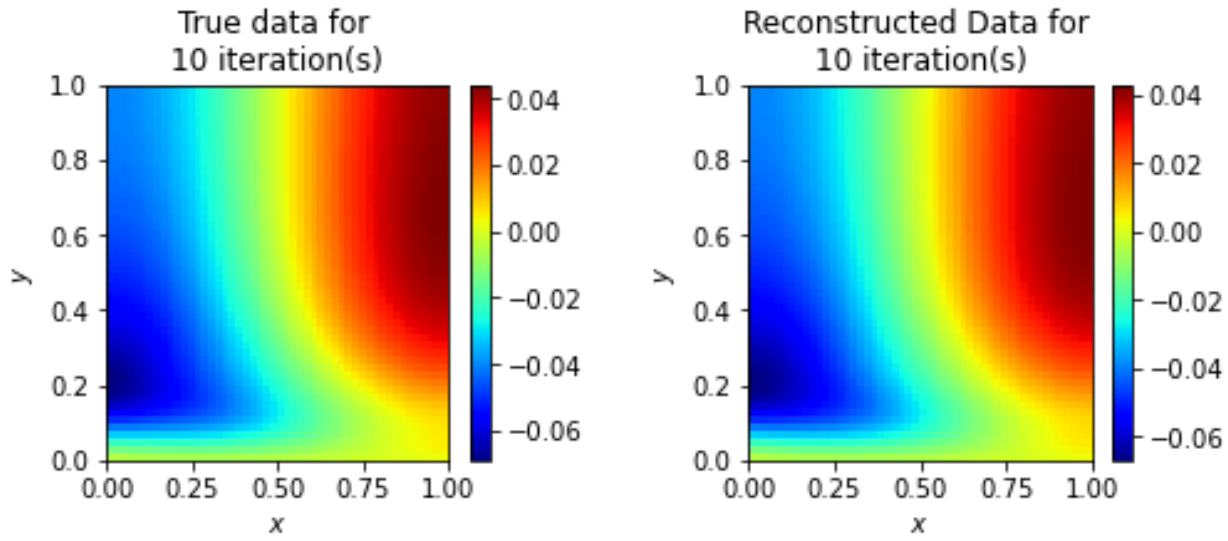

Figure 6: True, and reconstructed plots for the data after 10 iterations.

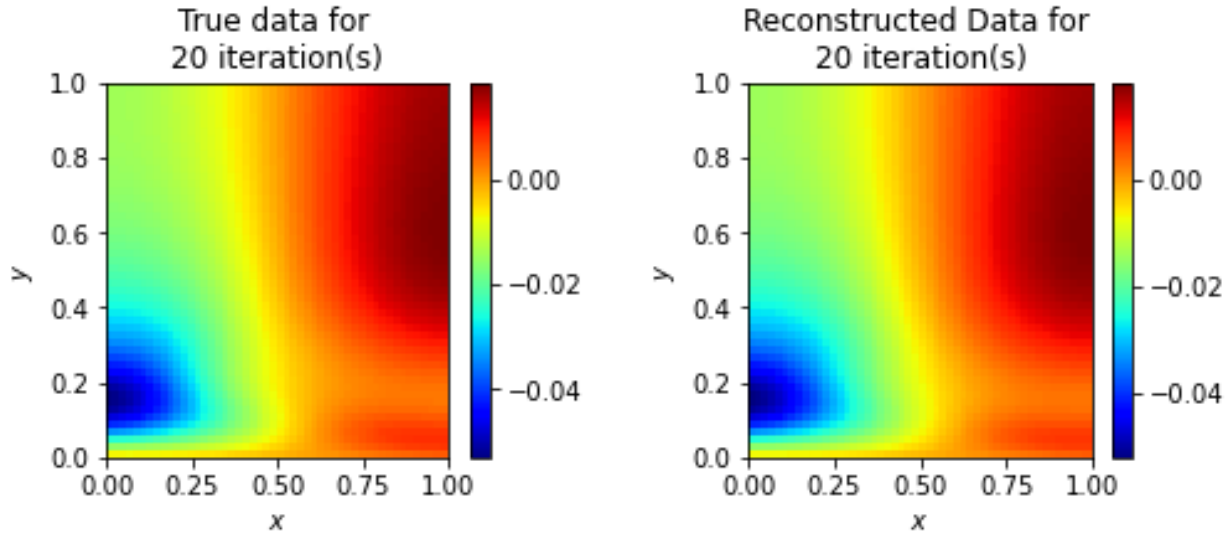

Figure 7: True, and reconstructed plots for the data after 20 iterations.

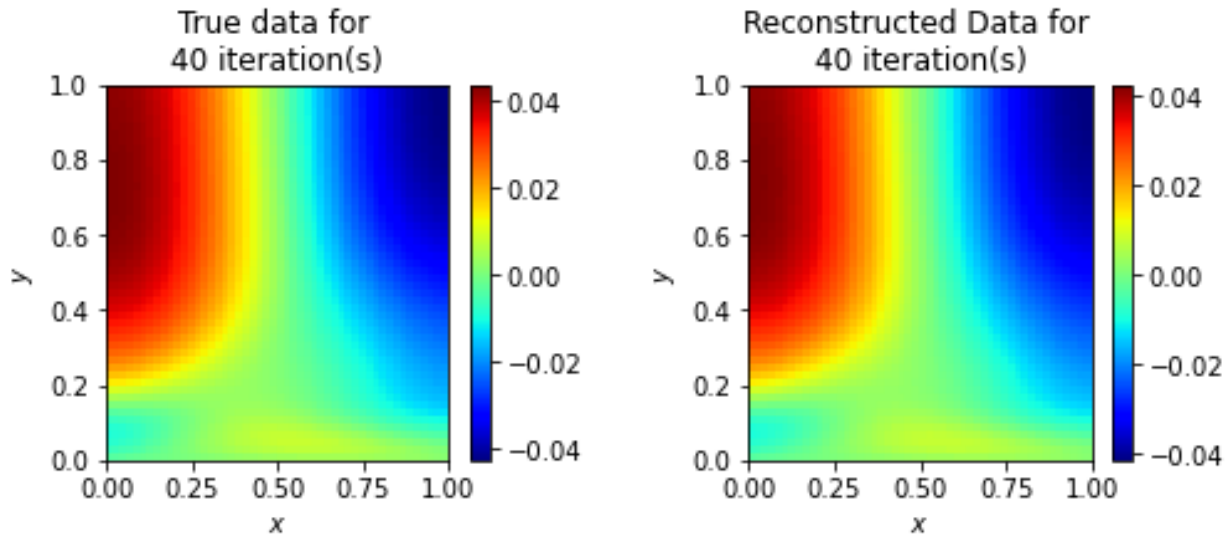

Figure 8: True, and reconstructed plots for the data after 40 iterations.

Iteration	Error
1	0.031047112235968104
5	0.05710641531045861
10	0.04205909929896853
20	0.02297443736261352
40	0.03608389113248515

Table 2: Error between true and reconstructed data

In a quick assessment looking at the data, it's obvious that the error is minimal from an eye test of the contour plots and looking at Table 2 confirms how small the respective error numbers are.

4.2 Future State Prediction Data Compared with Reconstructed Data

Another key feature of DMD is to look at future state prediction. The future state was plotted by taking two times the time vector which allows the DMD algorithm to run twice the iteration into the future. Another way of looking at this is that in essence of what we see at iteration number 10 for the predicted value will be the reconstructed iteration number 20. This is the same for predicted iteration 20 will be reconstructed 40 of the data set and so forth. A sample of this output can be demonstrated by the case for 10 iterations.

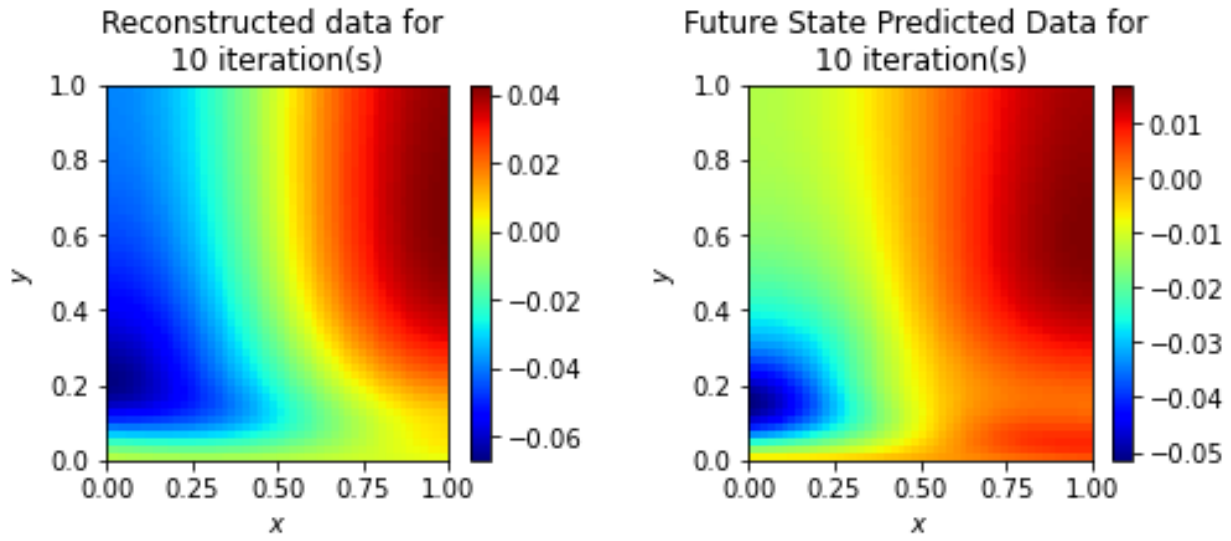

Figure 9: Sample of reconstructed and future state plots for the data after 10 iterations.

Looking at this plots, it is clear that the data shown for the reconstructed data is the same as the data for Figure 6. When looking at the future state prediction however, the data looks more like Figure 7. This result confirms the logic behind twice time vector correlating to twice the iteration. In order to have a better understanding of this, plots of this for iterations 10, 20, 40 are shown. In order to do so, the reconstructed and future iterations had to be run for iterations 5, 10, 20, 40; since future state 5 is the same as reconstructed state 10. An error table was also produced in order to have a numerical representation of the difference between the true and reconstructed as well as the reconstructed and future state predicted.

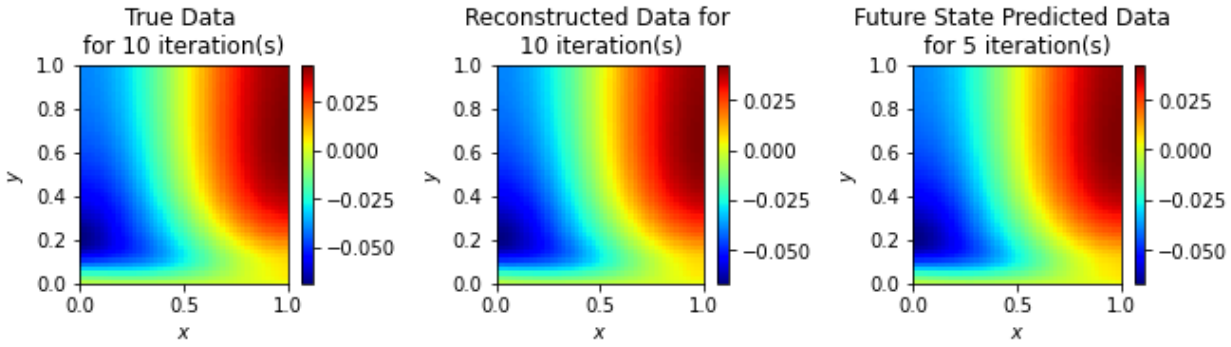

Figure 10: Reconstructed and future state plots for the data after 10 iteration equivalent.

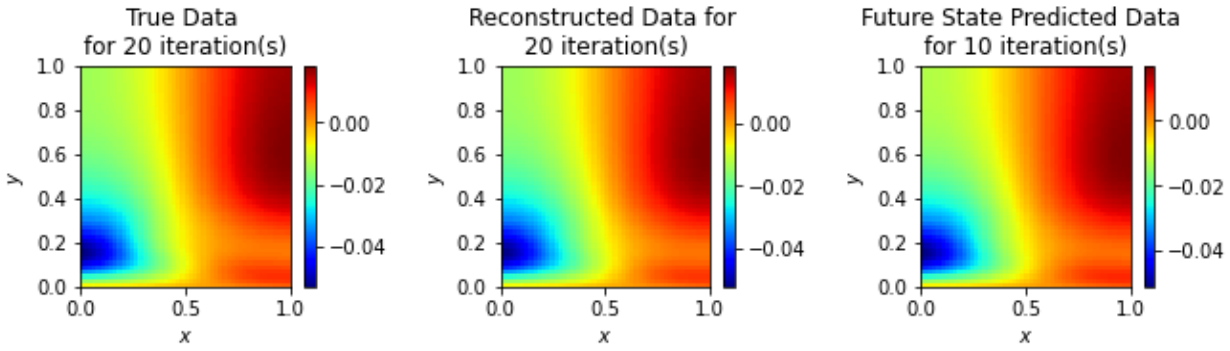

Figure 11: Reconstructed and future state plots for the data after 20 iteration equivalent.

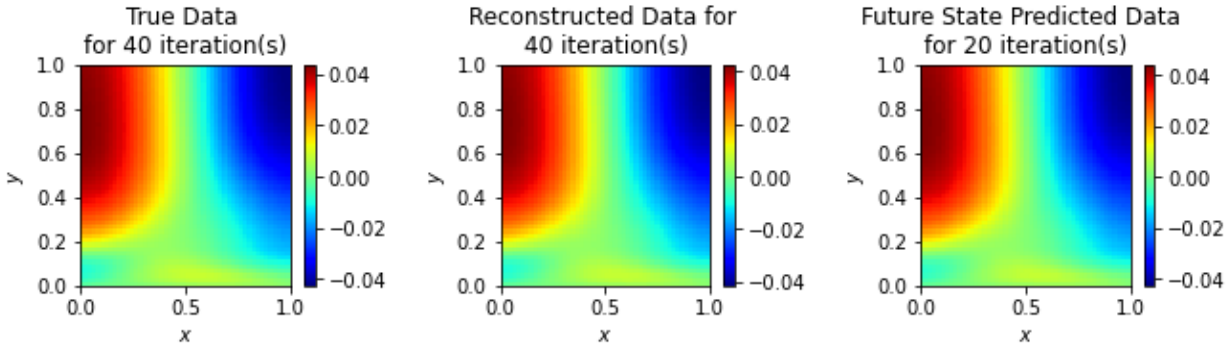

Figure 12: Reconstructed and future state plots for the data after 40 iteration equivalent.

Iteration True Data Equivalent #	Error Between True and Future State	Error Between Reconstructed and Future State
10	0.05442988136248694	0.013338363178793625
20	0.04624517847979632	0.04211421946344624
40	0.017004053498778537	0.04587783401078722

Table 3: Error between reconstructed and future state prediction

The future state prediction results were quite surprising in that they appeared to be a good representation of the data and seemed fairly accurate in comparison to the reconstructed data. Some limitations need to be recognized that if the time was put too far into the future rather than double it, than the predicted data may not appear as nicely as it did in this case.

5 Discussion of Results

Analyzing the results from the DMD method for the results from the *Analysis of True an Reconstructed Data* section, it is abundantly clear that the reconstructed data is very reflective of the true data when comparing the current iterations. The error which is the norm of the difference between the true and reconstructed data, demonstrated that there is very small numerical difference

between the data sets. The error is useful as otherwise the difference would be hard to observe. The results for this portion are very satisfactory as they exemplify the theory learned.

Looking at the results from the *Future State Prediction Data Compared with Reconstructed Data* section, the future state prediction method was done by taking twice the time vector and then comparing with the reconstructed data of the actual iteration, such that if twice the time vector for 20 iterations was used then in reality it is 40 iterations for the future state prediction. The results of the contour plots and the error table demonstrated this logic. The error was relatively low for both the true and reconstructed data which increases confidence in this approximation for future time steps.

One reason behind why the DMD is so successful is that the eigenvalues are negative which allows equation (8) to work properly, because if an exponential is raised to a positive power than the solution would blow up and grow exponentially however since the eigen values are negative for Ω this is helpful.

6 Conclusion

Dynamic mode decomposition is a very powerful method used in many applications in the scientific community for higher order sets of data. The results here demonstrated the effectiveness of DMD for Poisson's equations coupled with Neumann boundary conditions. Both the reconstructed state's comparison with the true, and the future state's comparison with the reconstructed were accurate and had a minimal error between them respectively. DMD is a tool, and like any tool it is only as good as the application it is being used for. DMD's strengths were highlighted in this problem as DMD is quite good at reducing higher order systems and looking at future state predictions of them especially for a numerical solution with snapshots in time such as the iterations used.

Looking forward, some interesting things that may be nice to look at would be how far into the future does this approximation hold up for and when does the DMD break down in respect to not being able to accurately predict the data.

References

[1] Cassel, Kevin. (2020). *MMAE 517: Computational Fluid Mechanics, Problem Set 2.* Retrieved from Blackboard.com

[2] Steven L. Brunton and J. Nathan Kutz. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control (1st. ed.).* Cambridge University Press, USA.