Classfying Handwritten Digits Through K-Means Clustering

David Wang  
McKelvey School of EngineeringWashington University in St. Louis  
St. Louis, Missouri, USA  
d.wang1@wustl.edu

Kathy Pan  
McKelvey School of EngineeringWashington University in St. Louis  
St. Louis, Missouri, USA  
kaiyun.p @wustl.edu

*Abstract*—In this case study we used the skills and methods we have learned in linear algebra and MATLAB to simulate dynamical systems and explore strategies to control them. This case study links topics of linear dynamical systems and control engineering and applies them to model the real-world issue of COVID-19 pandemic spread dynamics.

# Introduction (*Heading 1*)

This project utilizes concepts of linear dynamical systems to approach modelling the issue of pandemic spread, using the recent outbreak of the novel COVID-19 virus as data. As this project utilizes a myriad of linear algebraic techniques and topics learned in ESE 105, it enforces the learning done in class and allows a practical application of the subjects explored. Particularly, we seek to implement a classic pandemic spread model called the SIRD model. We then aim to fit it to local COVID-19 spread data. After such, we generalize the SIRD model to account for the individuals in lockdown, account for vaccinations, and to create a larger model which can highlight the number of people in each local region, and how people can travel between regions.

# Methods

As aforementioned in the introduction, our overall task comes fourfold. Firstly, we aim to implement a classic model describing pandemic spread, named the SIRD model. Each state in this model is represented by a four-dimensional vector, each element in which represents the fractional number of people in a population that are susceptible, infected, recovered, and deceased from the disease. After such implementation, we aim to fit this model to data from the ongoing COVID-19 pandemic. We aim to achieve this particularly within the context of our local regions: St. Louis, Jefferson City, and Springfield, MO. From such application, we can make inferences surrounding real-world COVID-19 dynamics based on our model. Thirdly, we seek to generalize the SIRD model to create a new, SLIRD model. This model instead has 5 states, with the addition of the ‘L’ state highlighting the proportion of the population in lockdown, with the implication that the population belonging to the other states (SIR) are currently not locked down. Finally, we seek to create an intervention strategy. We may choose to implement lockdown laws, vaccination deployments, or treatment options. We then seek to optimize this solution to better the modelled pandemic situation.

## SIRD Modelling

Firstly, we must create a SIRD model to represent current pandemic dynamics. As highlighted in Section 9.3 of Boyd and Vendenberghe’s Introduction to Applied Linear Algebra, the SIRD model takes the form of a linear dynamical system. Such a system possesses a state vector representing the state of the pandemic at each time step. To model the next time step, the transition matrix is used, which represents the development of the pandemic at any timestep .



Concretely, the state vector at time is defined , each element of which corresponds to a fractional quantity of people susceptible, infected, recovered (and hence immune), and dead. The transition vector , after the multiplication , yields another vector .

Following (1), it is evident that any progression in timestep can be modelled as a simple matrix multiplication between a state vector and a transition matrix. Hence, given any arbitrary state at any time step, as well as a transition matrix, the state space of any time after the initial one given can be calculated. Namely,

2

We modelled our SIRD based on the concept given by (2). Using this model, an example epidemic evolution trend can be seen in Fig. 1.

Chart

Description automatically generated with low confidence

Fig. 1. Evolution trend of a sample epidemic based on the linear SIRD model

Despite changing the coefficients in the transition matrix, the model tends to converge with the majority of the population recovered and a small fraction deceased, as seen in Fig. 1. Of course, given an arbitrarily high death rate, it is also possible for a disease to kill faster than it is infecting, with the model hence converging with some fraction of population still susceptible. However, since this requires an unrealistically high lethality rate, we did not explore this approach. Instead, we concluded that in the supermajority of cases, the model converges with only recovered and deceased.

## COVID-19 Data Regression

Since modelling the SIRD model, we were able to fetch real COVID-19 data from years 2020 to 2021 in the local regions of St. Louis, Jefferson City, and Springfield, MO. We seek to regress our SIRD model to such real world data, and obtain transition matrices which model the spread of COVID-19.

Chart, line chart

Description automatically generated

Fig. 2. COVID-19 Data in 3 regions since 2020-3-17

Fig. 2 highlights

2

With this, we have found 10 outliers in the test set of 200 images, as shown in Fig. 1. The outlier flags are shown in Fig. 2 in the form of a stem plot.

A picture containing qr code

Description automatically generated

Fig.1. Visual representation of outliers determined in test set

# Results and Discussion

As described in the methodology, we have tested a range of values to search for the number of centroids with highest accuracy. We have represented this finding visually through a line plot. As expected, the greater the number of clusters, the smaller the average distance between all data points and their centroids. This is simply due to the fact that more centroids will inherently better fit a test set, through a higher centroid count may not be optimal due to the overfitting problem. This notion is highlighted in Fig. 3, where a higher corresponds to a lower distance between data point and centroid, hence suggesting better fit.

However, a better fit to the training set does not necessarily correspond to greater accuracy in testing.

As shown in Fig. 4, when evaluated against the test set of 200 images, the model with performed the best, with an accuracy of 153 predictions. In choosing our value of choice, we have also added a penalty for each additional we used. When not considering the outlier classification, according to the scoring metric, score is awarded as according to (3):

Chart, line chart

Description automatically generated

Fig. 4. Accurate predictions in test set performed by each model of varying K values

3

Hence we have modified our program to use the model which yields the greatest score, by subtracting half of the from each model’s accuracy to yield each program’s score.

In this case, the optimal model is still at .

Chart, line chart

Description automatically generated

Fig. 5. Score yielded by each model

Now, we plot the K-Means cost function, as shown in (1), as a function of iterations. As expected the classic shape of the steep drop into a shallowing decline is seen in Fig. 6.

Chart

Description automatically generated

Fig. 7. Predicted labels and true labels performed by

Chart, histogram

Description automatically generated

Fig. 6. K-Means cost of the model as a function of iterations

To visualize the performance of our model, we have shown a representation of the labels as predicted by our model, and the correct labels in the test set in Fig. 7.

However, to evaluate the correctness of our model, we first had to assign each centroid to a digit which it represented. We did this by checking each centroid’s associated data points during training. Since the training set is labelled with the correct digits, we have taken each centroid’s digit representation to equal the mode of the data points’ digit representations. In other words, each centroid represents the digit that occurs most commonly in its associated data points. With this, we have also displayed the centroids of our model in Fig. 8.

Calendar

Description automatically generated with medium confidence

Fig. 8. Centroids and digit representations of the model

With these centroids, we were able to correctly evaluate the digits of approximately 75% of the data points we were presented, at least in the test set.

# Conclusion

From our experimentation, we have found a set of centroids which can correctly classify approximately 75% of an MNIST hand-written digits dataset. We have also derived a method to identify potential outliers based on the k-means architecture.

Despite our work, however, our k-means clustering algorithm does have its drawbacks – some inherent within the algorithm, and others in the way we have structured our model. Firstly, k-means clustering is very computationally expensive. There is no guarantee that the algorithm will converge in any given amount of time, with a very high time complexity of , where represents the number of data points in the input data set [1]. This time complexity is further compounded by our choice to evaluate the k-means algorithm for 10 values of , which does yield a better model, but takes a lot more time. Secondly, a few assumptions were made in our program, the main assumption being that each digit could be identified according to its cluster. For example, the differences between how some “1”s and “7”s are written are very miniscule, and the two digits could very much be clustered together with indistinguishable differences, based on their similarity in that they both have a vertical line in the center. Finally, the position of the handwritten digits also has a significant effect on the classification. A digit written that is offset by a few pixels can have significant implications on the classification, as the k-means algorithm takes each pixel as an independent dimension, and will hence read two offset images of the same digit to have little to no similarity. To seek improvement, we could use a smaller number of trials in to narrow down the optimal value, which trades optimality against time complexity. To solve the offsetting problem, we may perform some data manipulation to center each digit in the center.

##### References

1. M. K. Pakhira, "A Linear Time-Complexity k-Means Algorithm Using Cluster Shifting," 2014 International Conference on Computational Intelligence and Communication Networks, 2014, pp. 1047-1051, doi: 10.1109/CICN.2014.220.