

Notes on Dynamic Mode Decomposition (with some code)

camillejr.github.io/science-docs

Preface

Dynamic Mode Decomposition (DMD) is a data-driven method of finding low-rank structures in high-dimensional data sets.

These notes are taken from two lectures on Dynamic Mode Decomposition: [1] and [2] by Prof. Nathan Kutz from the University of Washington.

This document is still in preparation.

Contents

| | | |
|-----|---|---|
| 1 | Description of the system | 1 |
| 2 | Linear dynamical systems | 1 |
| 3 | Dynamic Mode Decomposition theory | 2 |
| 3.1 | Exact DMD | 2 |
| 3.2 | Going low-rank | 2 |
| 3.3 | Going back to the original dimensions . . | 2 |
| 4 | A broader view to more possibilities | 3 |
| 4.1 | Optimized DMD | 3 |
| 4.2 | Robust DMD | 3 |
| 5 | Python example | 3 |
| A | Solution to linear dynamical systems | 3 |
| B | Moore-Penrose inverse | 3 |

1 Description of the system

We have a system described by a differential equation:

$$\frac{d\vec{x}}{dt} = f(\vec{x}, t, \mu) \quad (1)$$

The function $f(\vec{x}, t, \mu)$ is a way of *modeling* that system. We also have *measurements* of the system in different points in space at time k , in the form of a vector(s) \vec{y}_k :

$$\vec{y}_k = g(\vec{x}_k) \quad (2)$$

where \vec{x}_k is the quantity of interest that we are aiming at measuring. The fact that we might not be able to measure it directly is accounted for by some function $g()$ (although it might happen that $\vec{y}_k = \vec{x}_k$, meaning that we are able to measure \vec{x}_k directly).

Notice, that for measurements at many moments in time, we may stack all the collected vectors \vec{y}_i for different times i to create a matrix whose columns represent time snapshots and whose rows represent position in space.

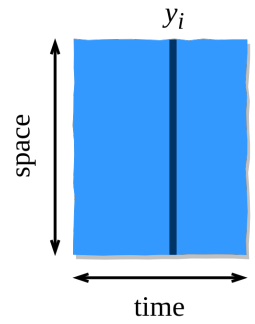


Figure 1: Data matrix with measurements of the system.

2 Linear dynamical systems

We are from now interested in systems where the governing equation from eq.(1) is not known (in other words, the function f is unknown) and we solely rely on measurements of the system which, in general, form a high-dimensional data set.

In the Dynamic Mode Decomposition we approximate that data set by a linear dynamical system of the form:

$$\frac{d\vec{x}}{dt} = A\vec{x} \quad (3)$$

This is in fact a very handy approximation since we are able to write down exact solutions to linear systems. Once we assume that the general solution is of the form:

$$\vec{x} = \vec{v}e^{\lambda t} \quad (4)$$

to obtain the parameters we effectively solve the eigenvalue problem:

$$A\vec{v} = \lambda\vec{v} \quad (5)$$

The exact solution to the linear system from eq.(3) is:

$$x = \sum_{j=1}^n b_j \phi_j e^{\lambda_j t} \quad (6)$$

For a reader who is now shaky about how this solution was derived, more can be found in appendix A.

3 Dynamic Mode Decomposition theory

3.1 Exact DMD

For the moment, we assume that we can measure the system directly, that is we measure $\vec{y}_i = \vec{x}_i$. Moreover, we assume that our data is collected in equal¹ time steps Δt . The measurements are combined inside a large matrix \mathbf{X} where each of its columns represents one time snapshot:

$$\mathbf{X} = [\vec{x}_1 \ \vec{x}_2 \ \vec{x}_3 \ \dots \ \vec{x}_m] \quad (7)$$

We split the large matrix \mathbf{X} into two matrices \mathbf{X}_1 and \mathbf{X}_2 such that:

$$\mathbf{X}_1 = [\vec{x}_1 \ \vec{x}_2 \ \vec{x}_3 \ \dots \ \vec{x}_{m-1}] \quad (8)$$

$$\mathbf{X}_2 = [\vec{x}_2 \ \vec{x}_3 \ \vec{x}_4 \ \dots \ \vec{x}_m] \quad (9)$$

If we now assume that a linear operator will map the first element of \mathbf{X}_1 with the first element of \mathbf{X}_2 , second with the second, third with the third, and so on, matrix \mathbf{X}_2 can be thought of as a matrix representing the *future state* of the matrix \mathbf{X}_1 . That linear operator is assumed to be a matrix \mathbf{A} .

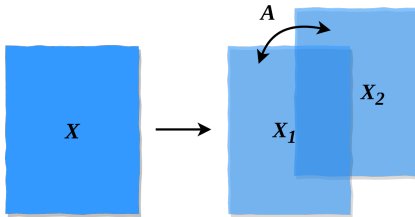


Figure 2: Splitting the data matrix into *past* and *future* matrices \mathbf{X}_1 and \mathbf{X}_2 , linked by the linear operator \mathbf{A} .

Note here, that for nonlinear systems, a matrix that transforms \vec{x}_1 to \vec{x}_2 is different from a matrix that transforms \vec{x}_2 to \vec{x}_3 and so on. DMD assumes, however, that there is one matrix \mathbf{A} that does all these transformations

¹Which is indeed a special case for real life measurements. Check section 4 for more information.

at once, with the least amount of error. It finds the *best-fit* linear dynamical system for the non-linear data set. In mathematical terms, we are looking for such \mathbf{A} that:

$$\mathbf{X}_2 = \mathbf{A}\mathbf{X}_1 \quad (10)$$

To solve such system we multiply both sides by the pseudoinverse of matrix \mathbf{X}_1 which we denote by \mathbf{X}_1^+ :

$$\mathbf{A} = \mathbf{X}_2\mathbf{X}_1^+ \quad (11)$$

The pseudoinverse described here, also known as the Moore-Penrose inverse², is computed using the least squares method. There is therefore certain information lost when going from eq.(10) to eq.(11).

Once we have solved for matrix \mathbf{A} , we can go back to eq.(5) and solve for eigenvalues and eigenvectors.

Up to this point, this is what the **exact DMD** computes.

3.2 Going low-rank

There is a problem that the eq.(11) may pose. Matrices \mathbf{X}_1^+ and \mathbf{X}_2 typically represent huge spatial dimensionality³ which in turn means that the matrix \mathbf{A} can become a square matrix of a massive size.

We are hence reluctant to perform the multiplication of matrices as is stated in eq.(11).

The hope comes from the *Singular Value Decomposition* (SVD). We believe that there are low-rank structures hidden in the data set and we are able to reduce the dimensionality of matrix \mathbf{A} without significant loss of information.

We perform the SVD on matrix \mathbf{X}_1 and later replace it with its lower-rank approximation:

$$\mathbf{X}_1 = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad (12)$$

$$\mathbf{X}_1 \approx \mathbf{X}_{1r} = \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^T \quad (13)$$

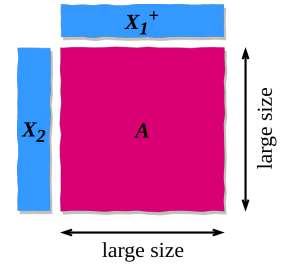


Figure 3: Building the linear operator \mathbf{A} .

3.3 Going back to the original dimensions

DMD modes are not orthogonal.

²Check appendix B for more information.

³This is often the case for data sets where we have few snapshots in time but a large number of spatial points where the measurements were taken.

4 A broader view to more possibilities

What can go wrong with our data sets?

4.1 Optimized DMD

- varying time steps
We mentioned earlier, that

4.2 Robust DMD

Sparse Identification

5 Python example

A Solution to linear dynamical systems

The general solution to the linear dynamical system of the form:

$$\frac{d\vec{x}}{dt} = A\vec{x} \quad (14)$$

is:

$$\vec{x} = \vec{v}e^{\lambda t} \quad (15)$$

Computing the time derivative of the eq. 15 we get:

$$\frac{d\vec{x}}{dt} = \vec{v}\lambda e^{\lambda t} \quad (16)$$

And substituting the eq. 15 to eq. 14 we get:

$$\frac{d\vec{x}}{dt} = A\vec{v}e^{\lambda t} \quad (17)$$

The nontrivial solution for the equality of these two above equations is obtained when:

$$A\vec{v} = \lambda\vec{v} \quad (18)$$

which is the statement of eigenvalue problem.

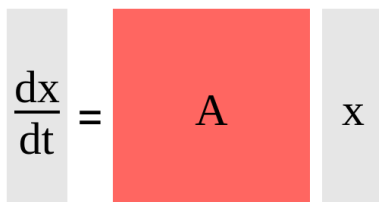
The diagram shows the equation $\frac{dx}{dt} = Ax$ where each term is enclosed in a rectangular box. The box for $\frac{dx}{dt}$ is light gray, the box for A is red, and the box for x is light gray. The equals sign is placed between the first and second boxes.

Figure 4: Linear dynamical system.

B Moore-Penrose inverse

References

- [1] N. Kutz, Dynamic Mode Decomposition (Theory) <https://youtu.be/bYfGVQ1Sg98>
- [2] N. Kutz, Dynamic Mode Decomposition Code <https://youtu.be/KAau5TBU0Sc>
- [3] G. Strang, "Introduction to Linear Algebra", 5th edition