Project Guide

Photonic Lantern Information Determination

Contents

Part I

Paper draft

1 SLM Reconstruction

This section contains information about the models and notebooks used to reconstruct SLM data.

1.1 Purpose

The purpose of this project is to find an neural network architecture able to reconstruct SLM data.

1.2 Data

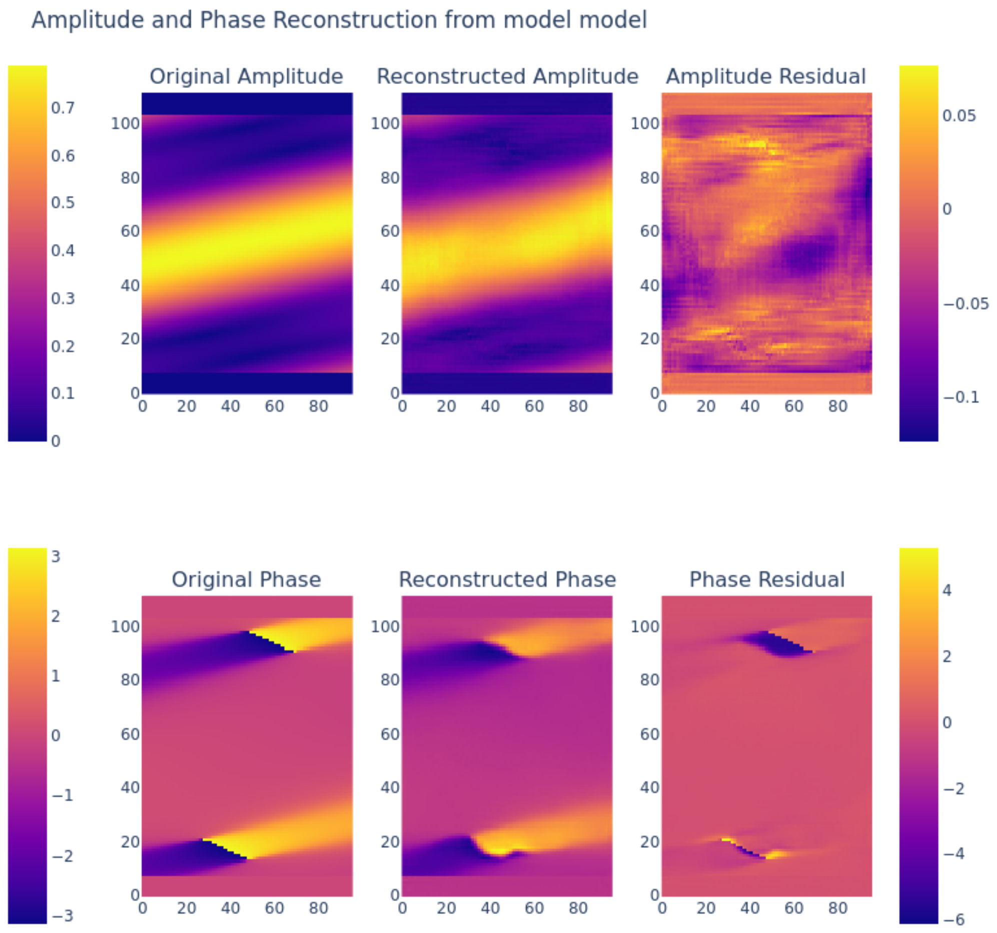
The data consists of 90000 complex fields generated with an SLM in the LAB, each datapoint is a 96x96x2 matrix containing information from data and phase of the complex field.

The corresponding 90000 output fluxes of the photonic lantern have the shape of a 19 element vector.

The data is from Barnaby's morgana folders that start with the prefix `slmcube_20230625_complsines-01sp_07_PSFWFs_file`

1.3 Results

The best results obtained are with a flux autoencoder and a convolutional neural network to decode the flux into amplitude and phase.



1.4 Code

- The paths related to the data of this project can be found in constants.py
- The model architectures and training hyperparameter set up can be found in

configurations.py and to instantiate them there are a set of functions in modeling_utils.py

- To train a fully connected neural network use AmplitudePhaseReconstruction-FullyConnected.ipynb.
- To train a convolutional neural network use AmplitudePhaseReconstruction-Convolutional.ipynb.
- To train a flux autoencoder neural network use AutoEncoderTraining.ipynb
- To train a autoencoder+convolutional SLM reconstructor use EncoderConvolutionalTraining.ipynb

1.5 Detailed results

To see all results check PART I from `all.pdf` .

2 PSF Reconstruction

This section contains information about the models and notebooks used to reconstruc PSFs data.

2.1 Purpose

The purpose of this project is to check if there are models that can reconstruct PSF data generated from a simulation using PL fluxes.

2.2 Data

The data related to this project was generated using HCIpy. The main functions used for this are `generate_psf_complex_fields` and `compute_output_fluxes_from_complex_field` that can be found in data_utils.py.

These PSFs are generated from atmospheric aberrations, NOT zernike coefficients.

There are 90000 datapoints in this project. The paths to them are found in psf_constants.py.

2.3 Code

- To generate the data see the notebook PSFGeneration.ipynb.
- To process the data (eg. flatten, compute PSF intensities, crop PSFs) see the notebook PSFProcessing.ipynb.
- To train models check the folder Training.

2.4 Results

To see all results check PART II from `all.pdf` .

3 Photonic Lantern Information Determination

This section contains information about the euclidean distances comparison between the different spaces along the optical pathway.

3.1 Purpose

The purpose of this project is to determine the relationship between the different spaces along the optical pathway to check of the information is conserved. If two

PSFs are similar or have a small euclidean distance between them, how similar will the output fluxes be?

To do this we select pairs of points from our datasets and measure the euclidean distances between them plotting the relationships in a scatter plot.

3.2 The data

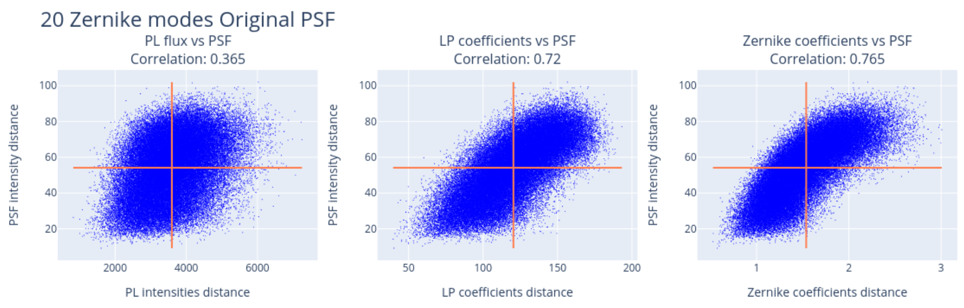
There are 4 different types of datasets:

- Zernike coefficients
- PSF
- LP coefficients
- Photonic Lantern Output fluxes

The paths to the datasets can be found in the file psf_constants.py.

In this case the PSFs are generated from Zernike coefficients instead of atmospheric aberrations so the function to generate them is `generate_zernike_psf_complex_fields` which can be found in the file data_utils.py.

3.3 Results



These plots are done as follows:

1. Chose a set of pairs of points. One point is a set of Zernike Coefficients, the PSF intensity they define, the LP coefficients from the overlap integral of the PSF with the LP modes and the output fluxes obtained after multiplying the transfer matrix with the LP coefficients.

2. Compute the euclidean distances between:

   - Zernike Coefficients of each point
   - Flattened PSF intensity matrix
   - Flattened complex LP coefficients
   - Output flux intensities of the PSF

3. The coordinates of each point in the above plot represent the euclidean distance in two different optical pathway spaces. (In the first on the left the y coordinate represent the distance between a pair of points in the PSF intensities space and the x coordinate the distance between the same pair of points in the Photonic Lantern fluxes space).

The results tell us that similarity between PSF imply similarity between PL fluxes but not viceversa.

3.4 Code

- To measure the euclidean distances use the notebook PLInformationDetermination.ipynb
- To plot the cloud of euclidean distances use the notebook LowOrderZernikePLInformationPlots.ipynb for zernike generated datasets or PLInformationPlots.ipynb

3.5 Detailed results

For a detailed report on the datasets, results and plots see Part III from `all.pdf` .

4 Clustering comparison

In this section a series of clustering algorithm are compared.

4.1 Purpose

The purpose of this project is to check the performance of different clustering algorithms when applied to our different datasets.

To do this, we create ground truth clusters of zernike mode coefficients (for example for 2 zernike modes we define two value ranges of [-1, -0.8] and [0.8, 1] which will create 4 different clear clusters) and from them we create the PSFs, LP coefficients and PL output fluxes.

We evaluate the following clustering algorithms:

- K-Means

- DBSCAN
- HDBSCAN
- Agglomerative

4.2 The data

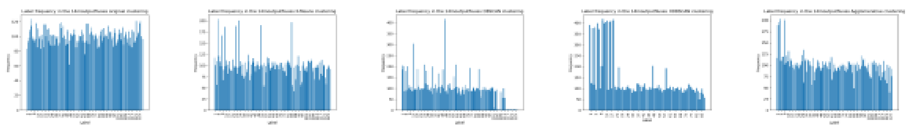
There are 4 different types of datasets:

- Zernike coefficients
- PSF
- LP coefficients
- Photonic Lantern Output fluxes

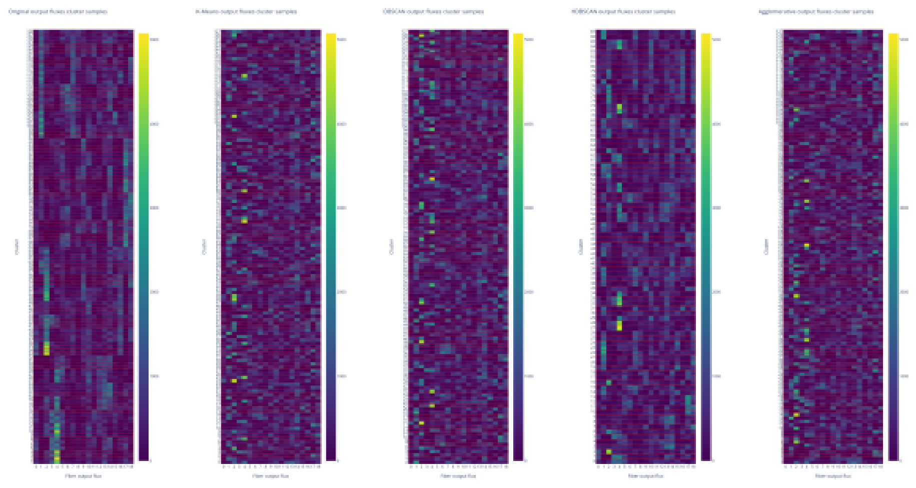
The paths to the datasets can be found in the file minidataset_constants.py.

We have used 2, 5, 9 and 14 zernike modes to create 4 datasets of 5000 datapoints each.

4.3 Results



(a) Original cluster densities (b) K-means cluster densities (c) DBSCAN cluster densities (d) HDBSCAN cluster densities (e) Agglomerative cluster densities



(f) Original cluster samples (g) K-means cluster samples (h) DBSCAN cluster samples (i) HDBSCAN cluster samples (j) Agglomerative cluster samples

Figure 215: Comparison between clustering Output fluxes algorithms

	Original	**K-Means**	**DBSCAN**	**HDBSCAN**	**Agglomerative**
Original	\	0.969	0.939	0.925	0.982
K-Means		\	0.929	0.923	0.970
DBSCAN			\	0.922	0.938
HDBSCAN				\	0.925

Table 91: Normalized Mutual Information between original Output fluxes clusters

Figure 1: Summary of clustering 20 zernike modes dataset

The best clustering algorithm is the Agglomerative followed by K-Means, although K-Means takes a lot of time.

4.4 Code

- The code to generate the datasets can be found in the notebook MinidatasetZernikePSFGeneration.ipynb
- The code to process the datasest can be found in the notebook MinidatasetProcessing.ipynb
- The code to reduce dimensionality of the data with UMAPs and PCAs can be found in the notebook MinidatasetDimensionalityReduction.ipynb
- The code to cluster the minidatasets is divided in a notebook per set of zernike modes, for example to see the clustering for the 14 zernike mode related datasets see notebook Minidataset14modesClustering.ipynb

4.5 Detailed results

For a detailed report on the datasets, results and plots see Part IV to VII from `all.pdf` .

5 Adjusted mutual information analysis

This section summarizes the adjusted mutual information project

5.1 Purpose

The purpose is the same than for the Photonic Lanter Information Determination with euclidean distances, but this time instead of looking at pairs of points we look at clusters of points to see how their relationship evolve along the optical pathway.

To do this we create datasets generated from 2, 5, 9, 14, 20, 27, 35 and 44 zernike modes.

When the datasets are created we use K-Means clustering to group points and measure the conservation of energy using the adjusted mutual information score.

After this, we also check how the adjusted mutual information changes with datasets of different sizes generated with 9 zernike modes.

5.2 The data

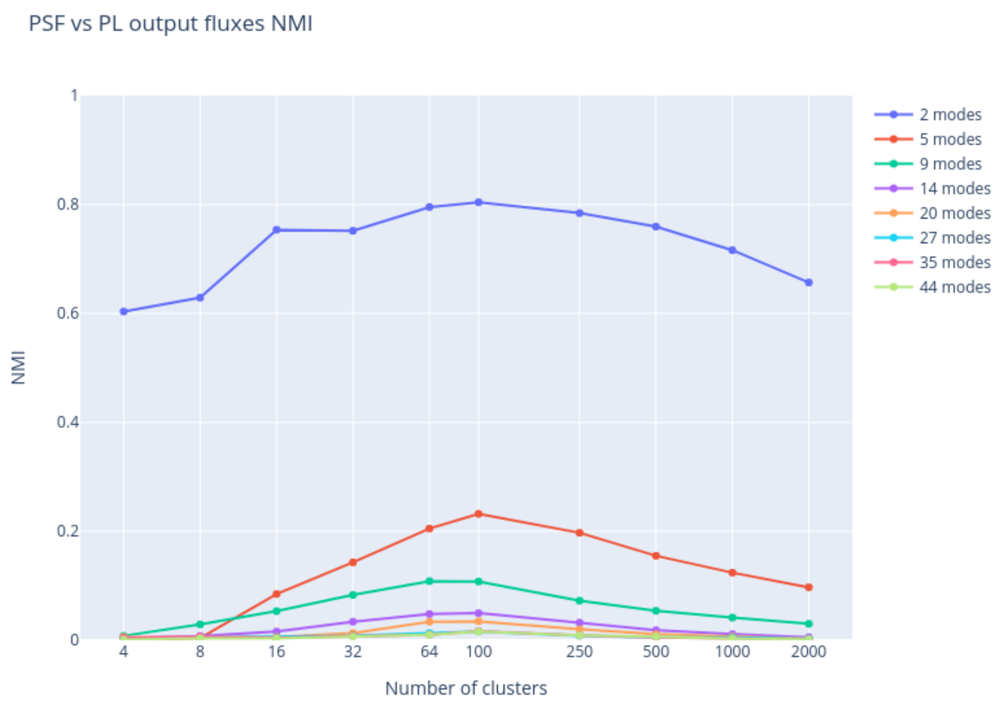
There are 4 different types of datasets:

- Zernike coefficients
- PSF
- LP coefficients
- Photonic Lantern Output fluxes

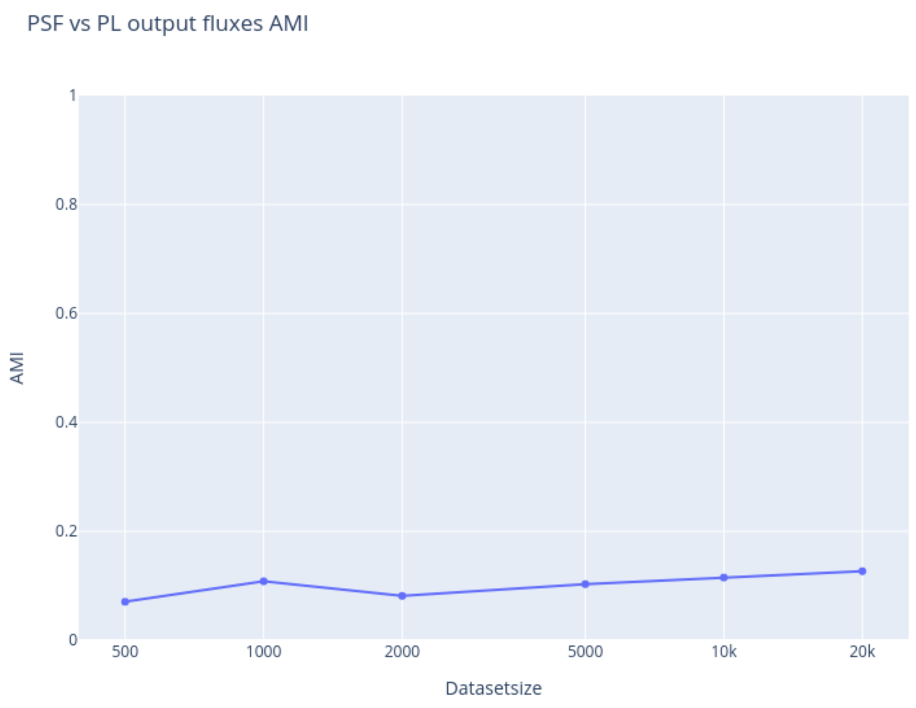
The paths to the datasets can be found in nmi_analysis_constants.py which contain the information for the 5000 datapoints datasets for 2, 5, 9, 14, 20, 27, 35 and 44 zernike modes.

The second set of paths to the datasets used to check how the AMI evolves with the number on points of the dataset can be found in ami_analysis_constants.py.

5.3 Results

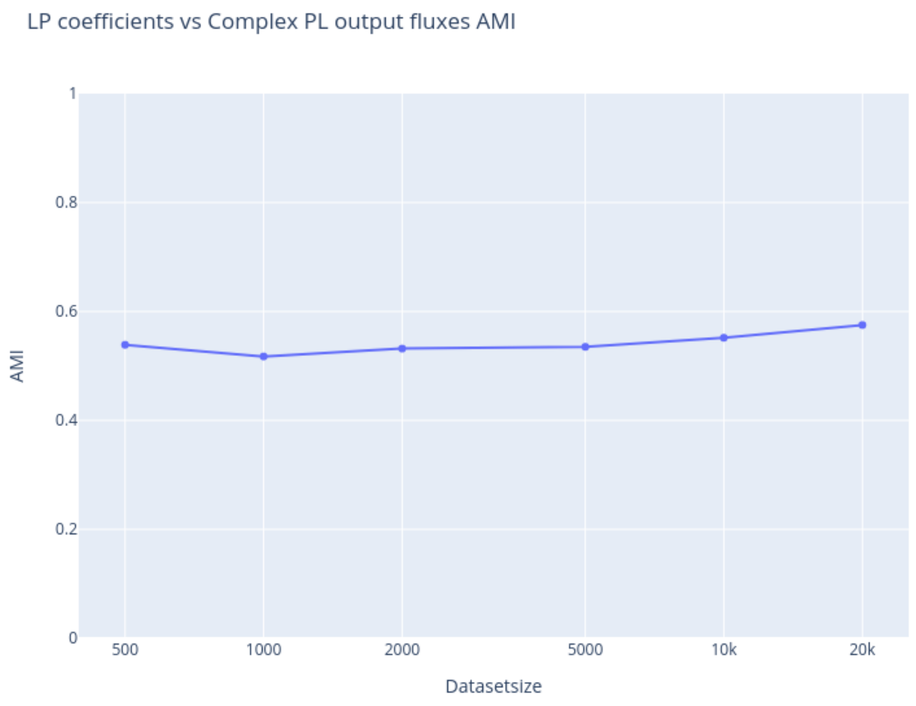


The conservation of information decreases over the number of Zernike modes, this could be improved by measuring the outputs of the photonic lantern in different wavelengths.



The AMI improves slightly as the number of datapoints increase, this could be because the clusters are more dense and the information locally is better conserved than globally.

These results are not what we expected so we looked at the AMI between LP complex coefficients and complex amplitudes of the photonic lantern:



The AMI between them should be very high but it is not so there is either a problem with the transfer matrix or the clustering (which could be losing information).

5.4 Code

Code for AMI analysis over number of Zernike modes

- The data generation for the AMI analysis over number of Zernike Modes can be found in NMIAnalysisDatasetZernikePSFGeneration-Copy1.ipynb
- The data processing for the AMI analysis over number of Zernike Modes can be found in NMIAnalysisdatasetProcessing.ipynb
- The dimensionality reduction of the data for the AMI analysis over number

of Zernike Modes can be found in NMIAnalysisDatasetDimensionalityReduction.ipynb

- The clustering of the data for the AMI analysis over number of Zernike Modes can be found in NMIAnalysisDatasetClustering.ipynb
- The plots and AMI analysis over number of Zernike Modes can be found in NMIAnalysisOverNClusters.ipynb

Code for AMI analysis over dataset size:

- The data generation for the AMI analysis over dataset size can be found in LastDanceDatasetZernikePSFGeneration.py
- The data processing for the AMI analysis over dataset size can be found in LastDancedatasetProcessing.py
- The dimensionality reduction of the data for the AMI analysis over dataset size can be found in LastDanceDatasetDimensionalityReduction.py
- The clustering of the data for the AMI analysis over dataset size can be found in LastDanceDatasetClustering.py
- The plots and AMI analysis over number of Zernike Modes can be found in LastDanceAMIAnalysisOverNClusters.ipynb

5.5 Detailed results

For a detailed report on the datasets, results and plots see Part VIII from `all.pdf`
.

6 Adjusted mutual information analysis

This section summarizes the adjusted mutual information project for a dataset generated using an arbitrary transfer matrix for a 19 fibre Photonic Lantern.

6.1 Purpose

The purpose is to determine wether if the problem in the section above is from the RSoft transfer matrix or the clustering algorithm information loss.

To do this we generate a transfer matrix with a condition number of 1 and again check the AMI between the different spaces for a dataset of 10000 points created with 9 Zernike modes.

When the datasets are created we use K-Means clustering to group points and measure AMI

6.2 The data

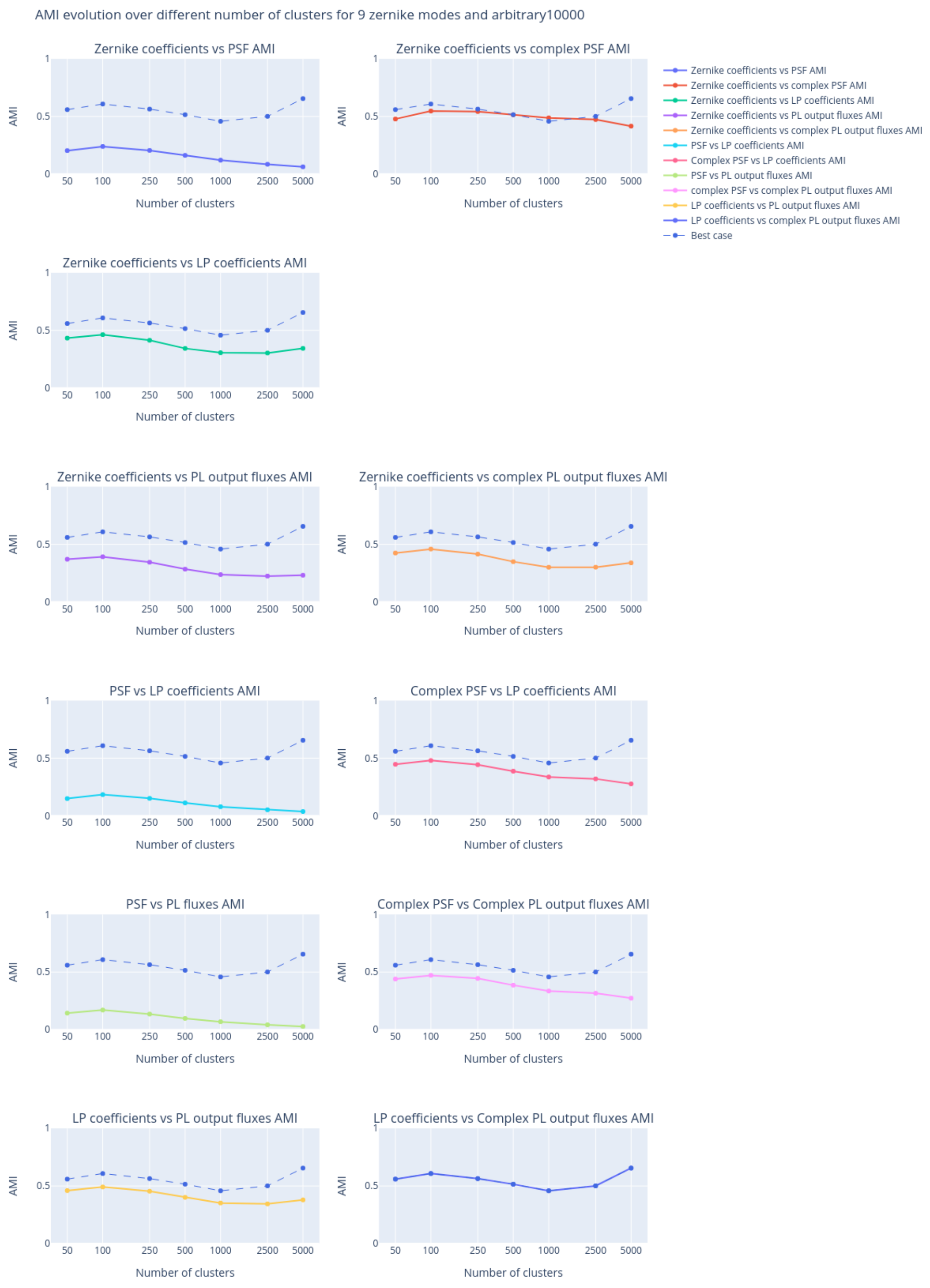
There are 6 different types of datasets:

- Zernike coefficients
- PSF intensities
- PSF complex fields
- LP coefficients
- Photonic Lantern Output fluxes

- Photonic Lanter Output complex amplitudes

The paths to the datasets used to check how the AMI evolves with the number on points of the dataset can be found in ami_analysis_constants.py.

6.3 Results


AMI evolution over different number of clusters for 9 zernike modes and arbitrary10000
Zernike coefficients vs PSF AMI
Zernike coefficients vs complex PSF AMI
Zernike coefficients vs LP coefficients AMI
Zernike coefficients vs PL output fluxes AMI
Zernike coefficients vs complex PL output fluxes AMI
PSF vs LP coefficients AMI
Complex PSF vs LP coefficients AMI
PSF vs PL fluxes AMI
Complex PSF vs Complex PL output fluxes AMI
LP coefficients vs PL output fluxes AMI
LP coefficients vs Complex PL output fluxes AMI
AMI
Number of clusters
0
0.5
1
50
100
250
500
1000
2500
5000
Zernike coefficients vs PSF AMI
Zernike coefficients vs complex PSF AMI
Zernike coefficients vs LP coefficients AMI
Zernike coefficients vs PL output fluxes AMI
Zernike coefficients vs complex PL output fluxes AMI
PSF vs LP coefficients AMI
Complex PSF vs LP coefficients AMI
PSF vs PL output fluxes AMI
complex PSF vs complex PL output fluxes AMI
LP coefficients vs PL output fluxes AMI
LP coefficients vs complex PL output fluxes AMI
Best case

The AMI between LP coefficients and PL complex amplitudes is still not high after using an arbitrary transfer matrix. The problem seems to be in the clustering process where information is loss, or maybe there is something that we are still missing in the model.

6.4 Code

Code for AMI analysis over dataset size:

- The data generation for the AMI analysis over dataset size can be found in TrueLastLastDanceDatasetZernikePSFGeneration.py
- The data processing for the AMI analysis over dataset size can be found in TrueLastLastDancedatasetProcessing.py
- The dimensionality reduction of the data for the AMI analysis over dataset size can be found in TrueLastLastDanceDatasetDimensionalityReduction.py
- The clustering of the data for the AMI analysis over dataset size can be found in TrueLastLastDanceDatasetClustering.py
- The plots and AMI analysis over number of Zernike Modes can be found in LastDanceAMIAnalysisOverNClusters.ipynb