User Guide VQPCA Matlab

matteo.savarese

August 6, 2023

1 Introduction

This is the User Guide for the MATLAB implementation of the VQPCA algorithm [1] and some of its variants. This User Guide will describe how the main code works, and the main options and functions the user can specify or use to customize the analysis. Similar work has been implemented in Python, which is available at `https://github.com/burn-research/OpenMORe`. This work presents the same implementation of that work, with some new customized options for the VQPCA routine.

2 Main function

The main code for the VQPCA routine is `local_pca_new.m`. The arguments of the function are described below.

2.1 `local_pca_new.m`

```
function [idx, infos] = local_pca_new(X, k, stop_rule, inputs, opt)
```

Main function to perform VQPCA.

Inputs:
`X`: *ndarray*
raw data containing observations as rows and variables (features) as columns. Its size is then $n_{obs} \times n_{var}$

`k`: *int*
user selected number of clusters

`stop_rule`: *int*
stopping rule used for PCA. For better reference, check the function `pca_lt.m`. The different rules are listed in 2.1.1

`inputs`: *float*
Input for selected stopping rule. For better explanation, check the function `pca_lt.m`. The inputs corresponding to the different stopping rules are defined in 2.1.1

`opt`: *struct*
Structure variable containing the available options for VQPCA. The main options are described in Section 2.1.2

Outputs:
`idx`: *1d-array (int)*
Cluster labels' array. Size n_{obs}

`infos`: *struct*
dictionary containing several fields and information of the resulting VQPCA routine. The different fields are described in [ref.]

2.1.1 `stop_rule` and `inputs`

Extended documentation is available in the function `pca_lt.m`, which is the in-house code for PCA. The list of available options for PCA dimensionality selection is reported below.

`stop_rule = 1`
This stopping rule corresponds to the **global amount of variance** to retain. The VQPCA will cut the dimensionality in each cluster such that the selected amount of variance is preserved in each cluster. Therefore, each cluster can have a different number of dimensions q. The corresponding `inputs` must be a float between 0 and 1.

`stop_rule = 2`
This stopping rule corresponds to the **individual variance rule**. We can retain the components whose eigenvalues are greater than the average of the eigenvalues (Kaiser, 1960) or than 0.7 times the

average of the eigenvalues (Joliffe 1972). For a correlation matrix, this average equals 1. The corresponding `inputs` are 1 for Kaiser and 2 for Joliffe.

`stop_rule = 3`
This stopping rule corresponds to the **broken stick** model. No additional inputs are required. For the correct compilation of the code, select `inputs` = 1.

`stop_rule = 4`
This stopping rule corresponds to **imposing a fixed number of eigenvectors**. The user can select `inputs` = q, where q is an integer and should not be greater than n_{var}.

`stop_rule = 5`
This stopping rule corresponds to the **Test of significance of the larger eigenvalues**. The `inputs` correspond to the confidence level for the critical χ squared value: $\alpha \in [0, 1]$

2.1.2 `opt` customization

`opt.Init` = *string*
This field allows for selecting different types of initialization methods. Available initialization methods are:

- `"random"` (*default*): this initializes the clusters by selecting `k` random centroids from the observation and performing a first iteration of the code considering eigenvectors as **I** matrix.
- `"uniform1"`: this initializes the clusters by taking `k` samples uniformly from the dataset
- `"uniform2`: this initializes the `k` clusters by binning the data uniformly in `k` groups.
- `"best_db"`: this initializes the clusters by performing an initial number of random initializations, using different random centroids for every iteration. Then, for each iteration, the VQPCA index described in [ref.] is evaluated, and the *best* random initial solution is chosen.

`opt.Centering` = *int*
Available option for data pre-processing. Available options are:

- `opt.Centering` = 1 (*default*): data will be centered around their mean
- `opt.Centering` = 0: data will not be centered

Refer to the function `AuxiliaryFunctions/center.m`.

`opt.Scaling` = *string*
Available option for data scaling (prior VQPCA, not within the loop, scaling is not performed twice). Available options are:

- `"auto"` (*default*): normalizing data by the standard deviation
- `"pareto"`: *pareto* scaling, normalizing data by the square root of the standard deviation
- `"max"`: normalizing data by their maximum value
- `"vast"`: normalizing data using the *vast* criterion
- `"level"`: the *level* criterion is used
- `"no"`: no scaling is performed

Refer to the function `AuxiliaryFunctions/scale.m` for more info.

`opt.Algorithm` = *string*
Available option for the algorithm for the VQPCA routine, available options are:

- `"VQPCA"` (*default*): the standard VQPCA routine is chosen
- `"FPCA"`: the variation of VQPCA based on mixture fraction partition is used in this case. This is a semi-supervised routine, and the mixture fraction array must be provided as an option. Namely you need to specify `opt.f = f`, where f is the n_{obs} array of the mixture fraction, and the stoichiometric mixture fraction `opt.fs = fs`, where f_s is the stoichiometric mixture fraction value. Please, for reference, see the work of Zdibał et al. [2]

For `FPCA`, refer to the function `AuxiliaryFunctions/condition.m`

`opt.MaxIter` = *int*
Available option to select the maximum number of iterations. *default* value is 200.

`opt.EpsRecMin` = *float*
The threshold for reconstruction error variance convergence. *default* value is 1.0E-06.

`opt.CustomError` = *string*
Available options to select a custom reconstruction error (or VQPCA distortion function). Available options are:

- `"Squared"` *default*: squared reconstruction error is used
- `"SquareRoot"`: the L_2 norm of the projection distance is used
- `"CustomPower"`: a custom power for the reconstruction error is selected. The power must be specified by the user via `opt.Power` = p, where p is a float

Refer to the function `CustomPenalties/custom_rec_err.m` for more info.

References

[1] Nanda Kambhatla and Todd Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9:1493–1516, 10 1997.

[2] Kamila Zdybał, Giuseppe D'Alessio, Antonio Attili, Axel Coussement, James C. Sutherland, and Alessandro Parente. Local manifold learning and its link to domain-based physics knowledge. *Applications in Energy and Combustion Science*, 14:100131, 2023.