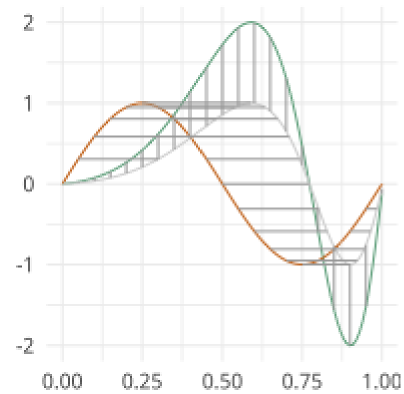
fdasrsf Documentation

Release 2.5.0

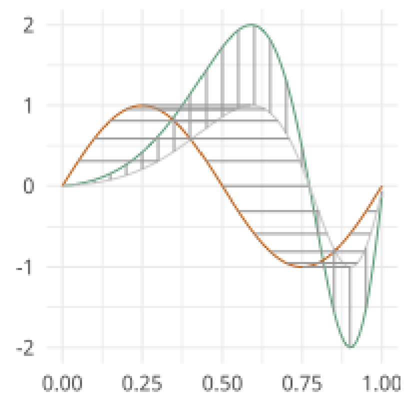
J. Derek Tucker

Nov 17, 2023

CONTENTS



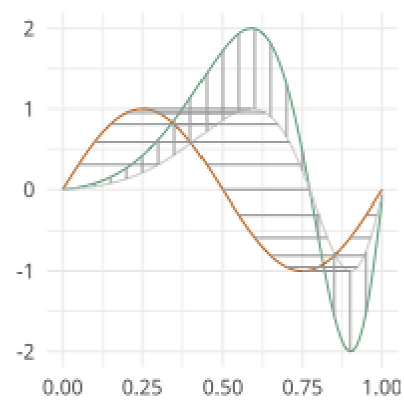
A python package for functional data analysis using the square root slope framework and curves using the square root velocity framework which performs pair-wise and group-wise alignment as well as modeling using functional component analysis and regression.



CHAPTER ONE

USER GUIDE

Contents:



1.1 Elastic Functional Alignment

Otherwise known as time warping in the literature is at the center of elastic functional data analysis. Here our goal is to separate out the horizontal and vertical variability of the functional data

[1]:
```
import fdasrsf as fs
import numpy as np
```

Load in our example data

[2]:
```
data = np.load('../../bin/simu_data.npz')
time = data['arr_1']
f = data['arr_0']
```

We will then construct the `fdawarp` object

[3]:
```
obj = fs.fdawarp(f,time)
```

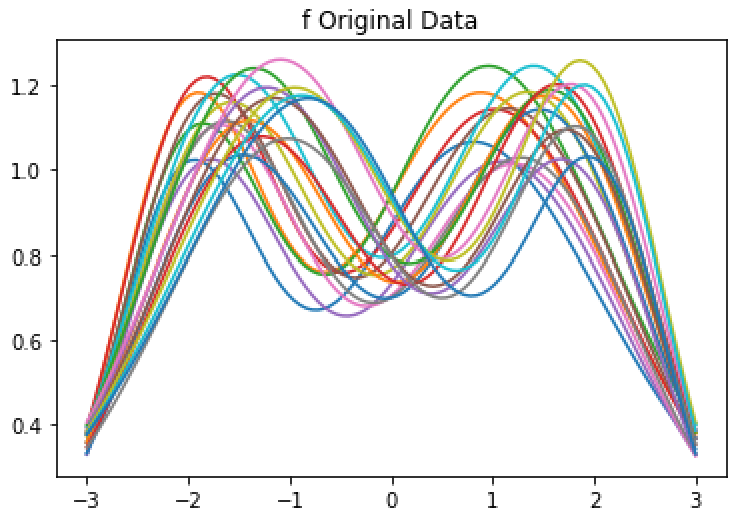
Next we will align the functions using the elastic framework

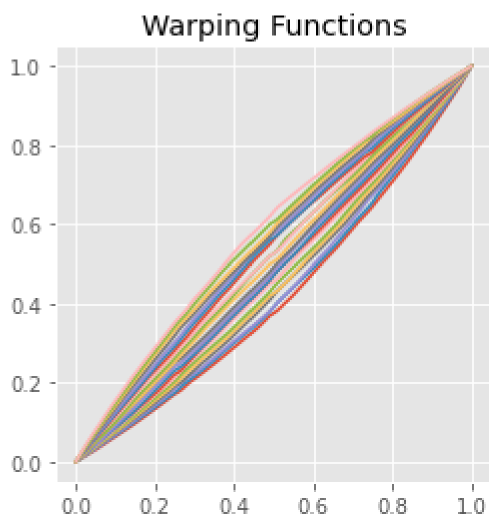
[4]:
```
obj.srsf_align(parallel=True)
```

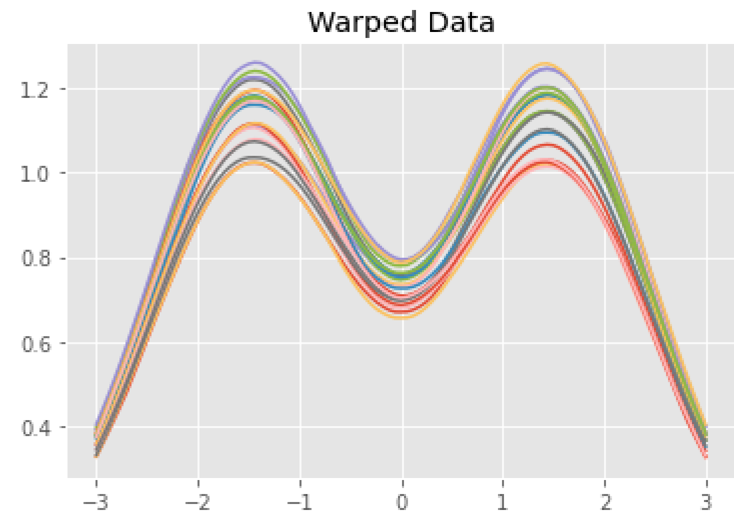
```
Initializing...
Compute Karcher Mean of 21 function in SRSF space...
updating step: r=1
updating step: r=2
```

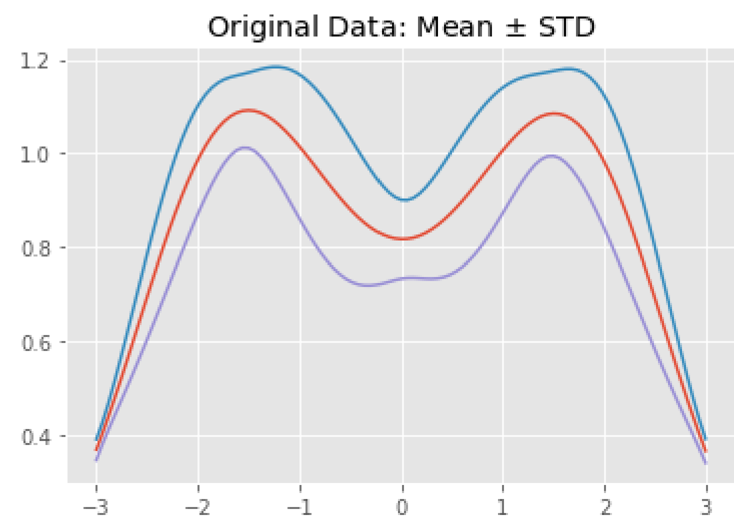
Display plots demonstrating the alignment

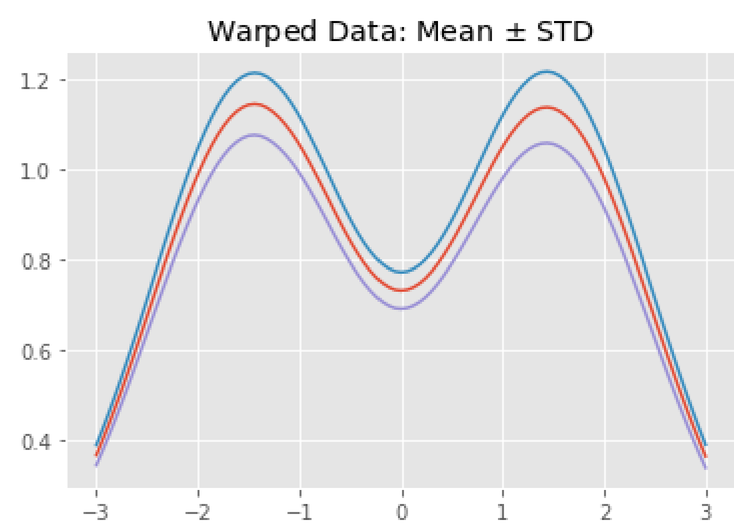
```
[5]: obj.plot()
```

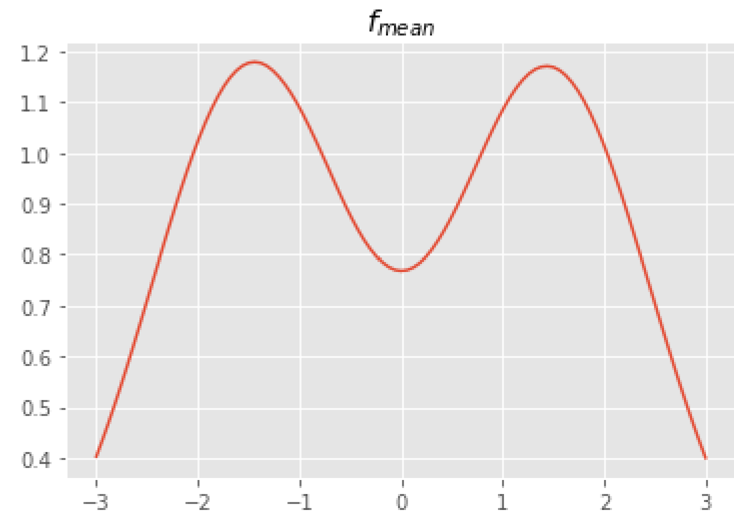


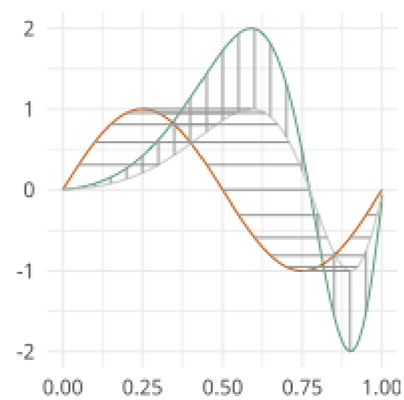



Warped Data
1.2
1.0
0.8
0.6
0.4
−3
−2
−1
0
1
2
3


Original Data: Mean ± STD
1.2
1.0
0.8
0.6
0.4
−3
−2
−1
0
1
2
3


Warped Data: Mean ± STD
1.2
1.0
0.8
0.6
0.4
−3
−2
−1
0
1
2
3





1.2 Elastic Functional Principal Component Analysis

After we have aligned our data we can compute functional principal component analysis (fPCA) on the aligned data, warping functions, and jointly

[1]:
```
import fdasrsf as fs
import numpy as np
```

We will load in our example data again and compute the alignment

[2]:
```
data = np.load('../../bin/simu_data.npz')
time = data['arr_1']
f = data['arr_0']
obj = fs.fdawarp(f,time)
obj.srsf_align(parallel=True)
```

```
Initializing...
Compute Karcher Mean of 21 function in SRSF space...
updating step: r=1
updating step: r=2
```

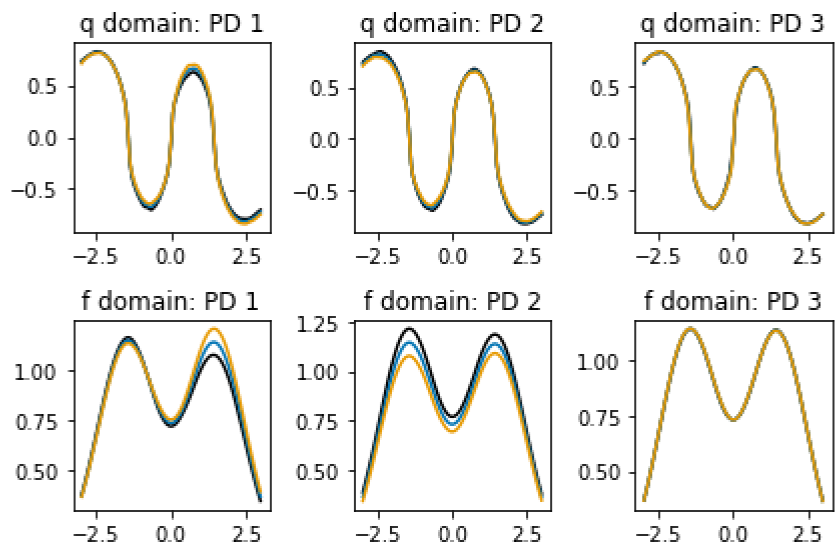
1.2.1 Vertical fPCA

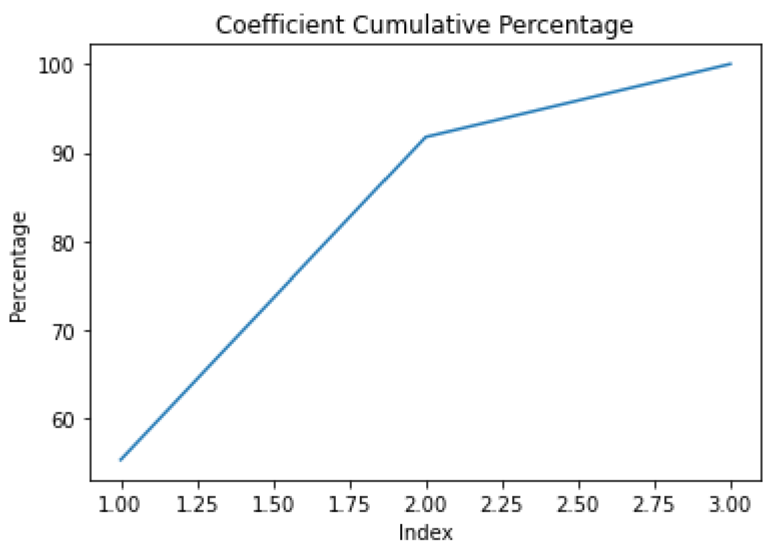
We will first compute fPCA on the aligned functions, by constructing the object and computing the PCA for the number of components, default=3)

```
[3]: vpca = fs.fdavpca(obj)
     vpca.calc_fpca(no=3)
```

We then can plot the principal directions

```
[4]: vpca.plot()
```





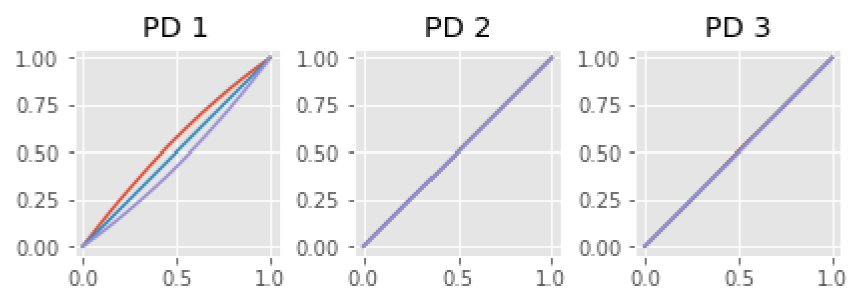
1.2.2 Horizontal fPCA

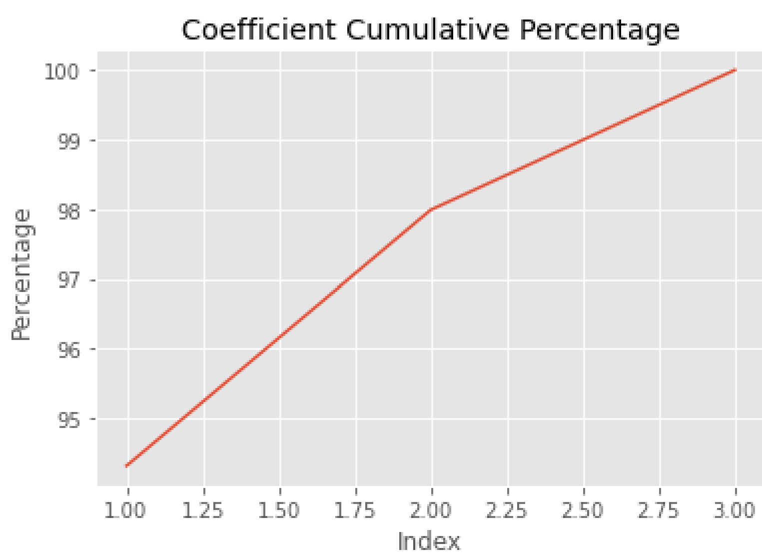
We can then compute PCA on the set of warping functions

[5]:
```
hpca = fs.fdahpca(obj)
hpca.calc_fpca(no=3)
```

We then can plot the principal directions

[6]:
```
hpca.plot()
```





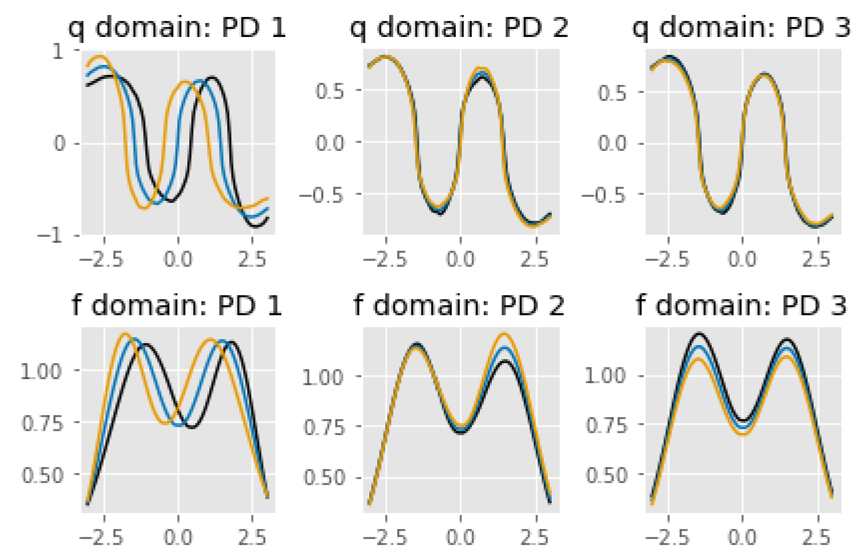
1.2.3 Joint fPCA

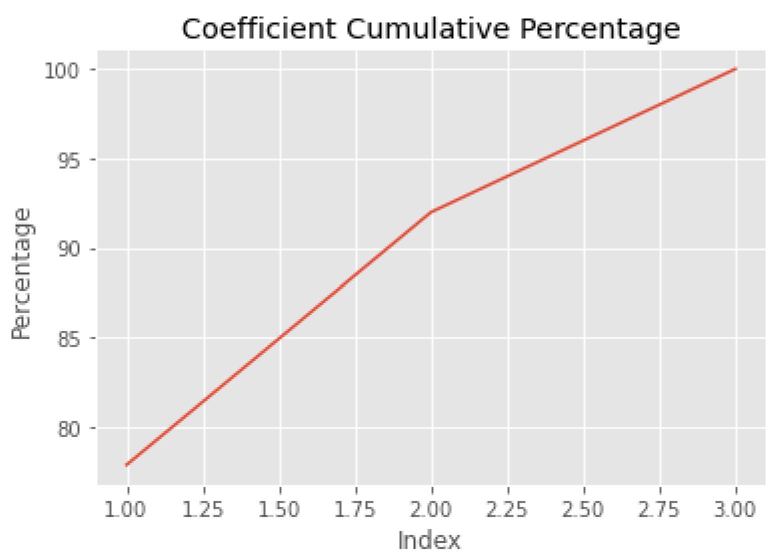
We can also compute the fPCA on jointly on the phase/amplitude space if we feel there is correlation between the variabilities

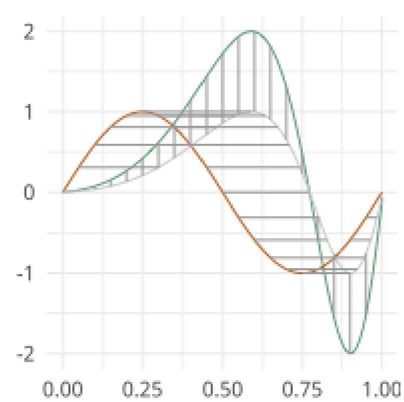
[7]:
```
jpca = fs.fdajpca(obj)
jpca.calc_fpca(no=3)
```

We then can plot the principal directions

[8]:
```
jpca.plot()
```


q domain: PD 1
q domain: PD 2
q domain: PD 3
f domain: PD 1
f domain: PD 2
f domain: PD 3


Coefficient Cumulative Percentage
Percentage
Index



1.3 Multivariate Functional Example

This notebook will show how to use the `fdasrsf` package to align and statistically analyze a set of multivariate functions using the SRVF framework

1.3.1 Load Packages

We will load the required packages and the example data set (MPEG7)

[1]:
```
import fdasrsf as fs
import matplotlib.pyplot as plt
import numpy as np
data = np.load('../bin/gait_data.npz',allow_pickle=True)
f = data['f']
g = data['g']
time = data['time']
```

Now we will construct a 2-D array of a set of 1-D functions from the gait data

[2]:
```
M,K = f.shape

beta = np.zeros((2,M,K))
beta[0,:,:] = f
beta[1,:,:] = g
```

1.3.2 Analyze

We now will construct a `fdacurve` object

[ ]:
```
obj = fs.fdacurve(beta,N=M)
```

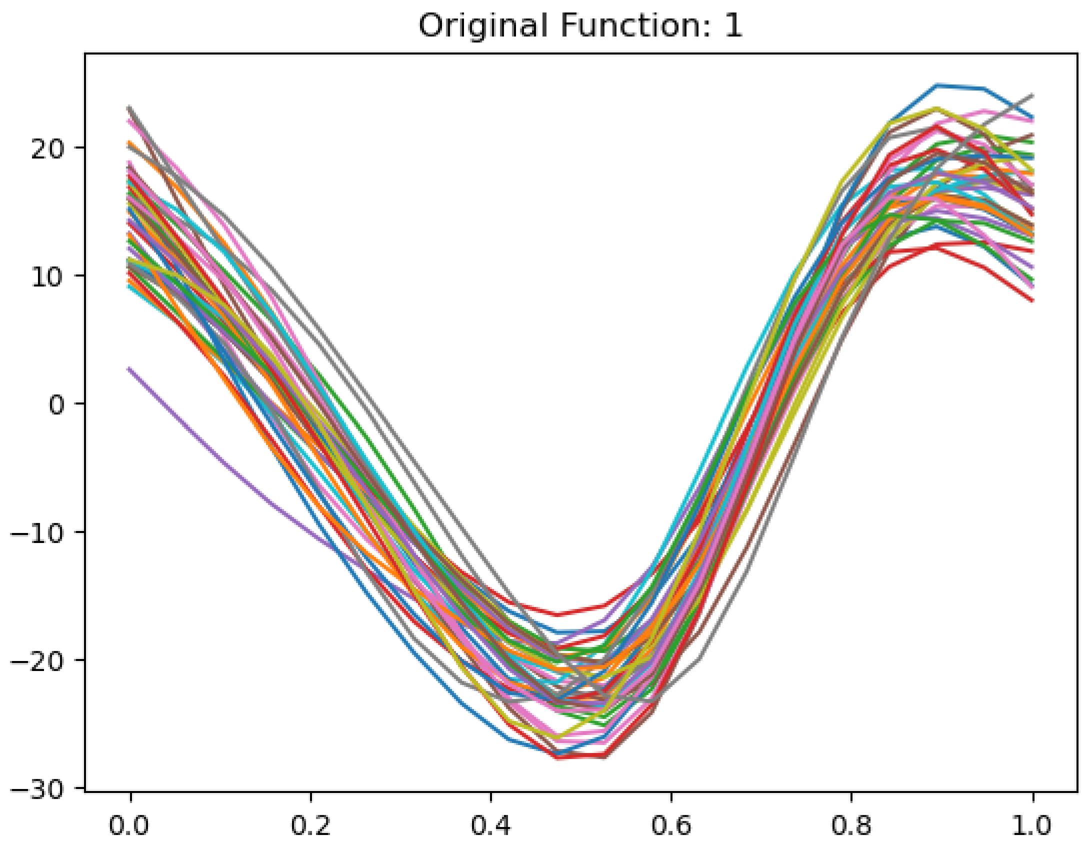
Next, find the Karcher mean and align the curves to the mean

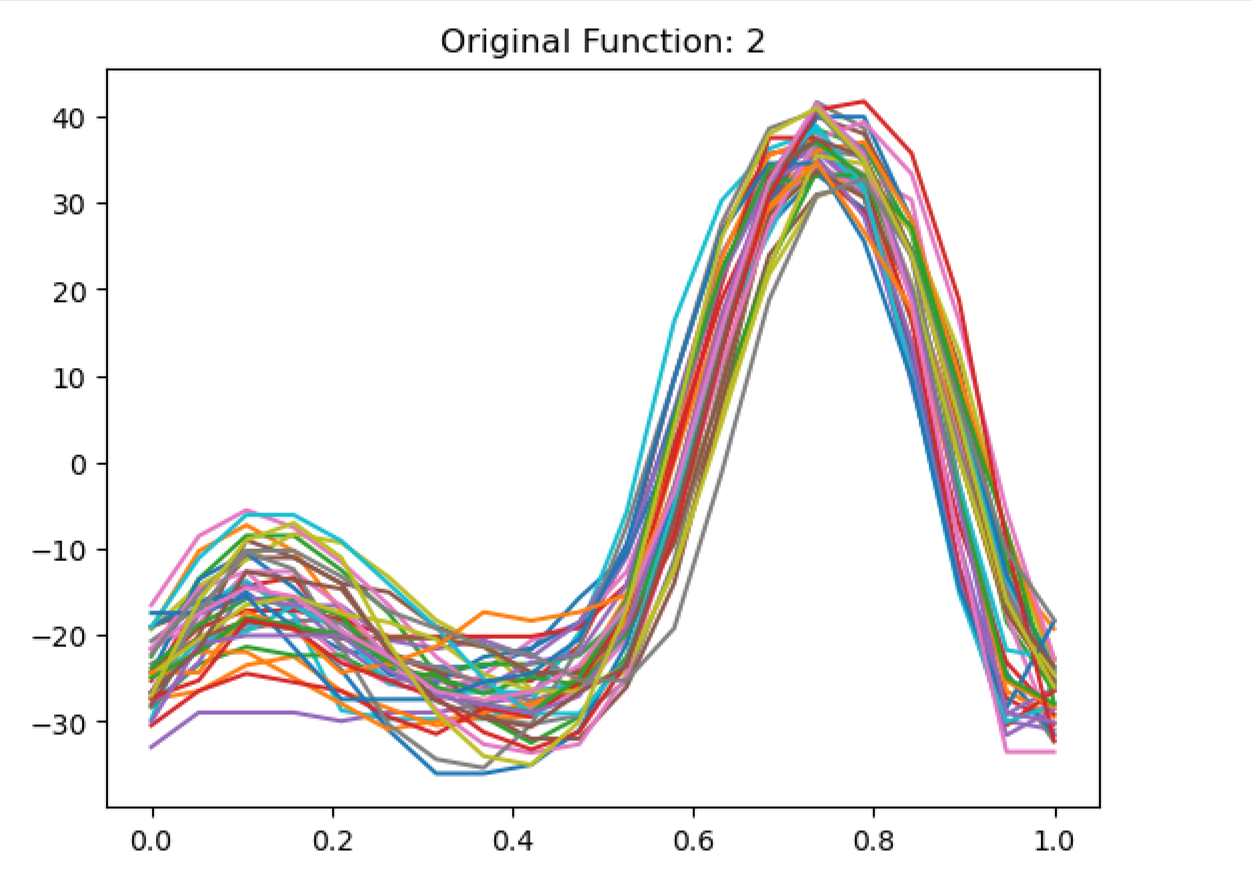
[ ]:
```
obj.karcher_mean(rotation=False)
obj.srvf_align(rotation=False)
```

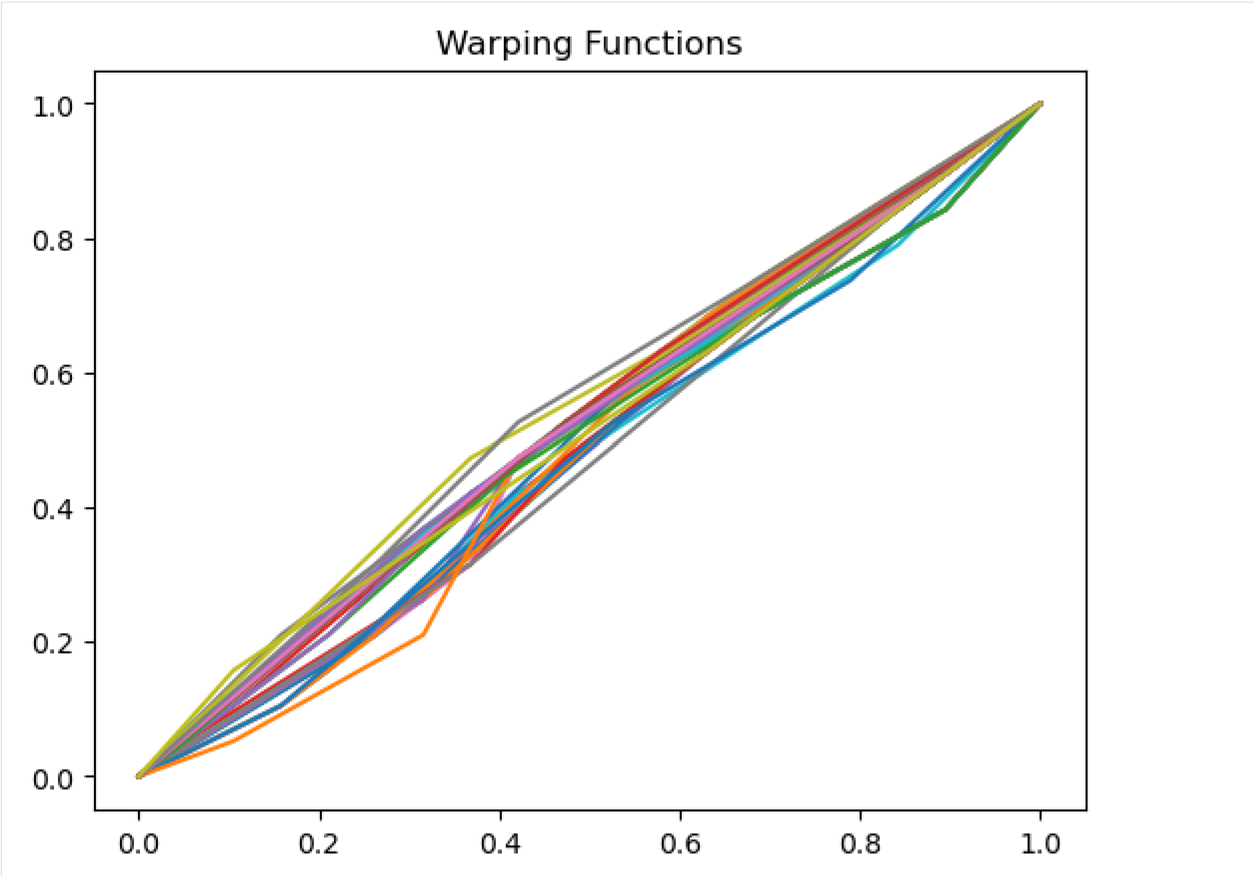
```
Computing Karcher Mean of 39 curves in SRVF space with lam=0
updating step: 1
updating step: 2
updating step: 3
updating step: 4
updating step: 5
updating step: 6
updating step: 7
updating step: 8
updating step: 9
updating step: 10
updating step: 11
```

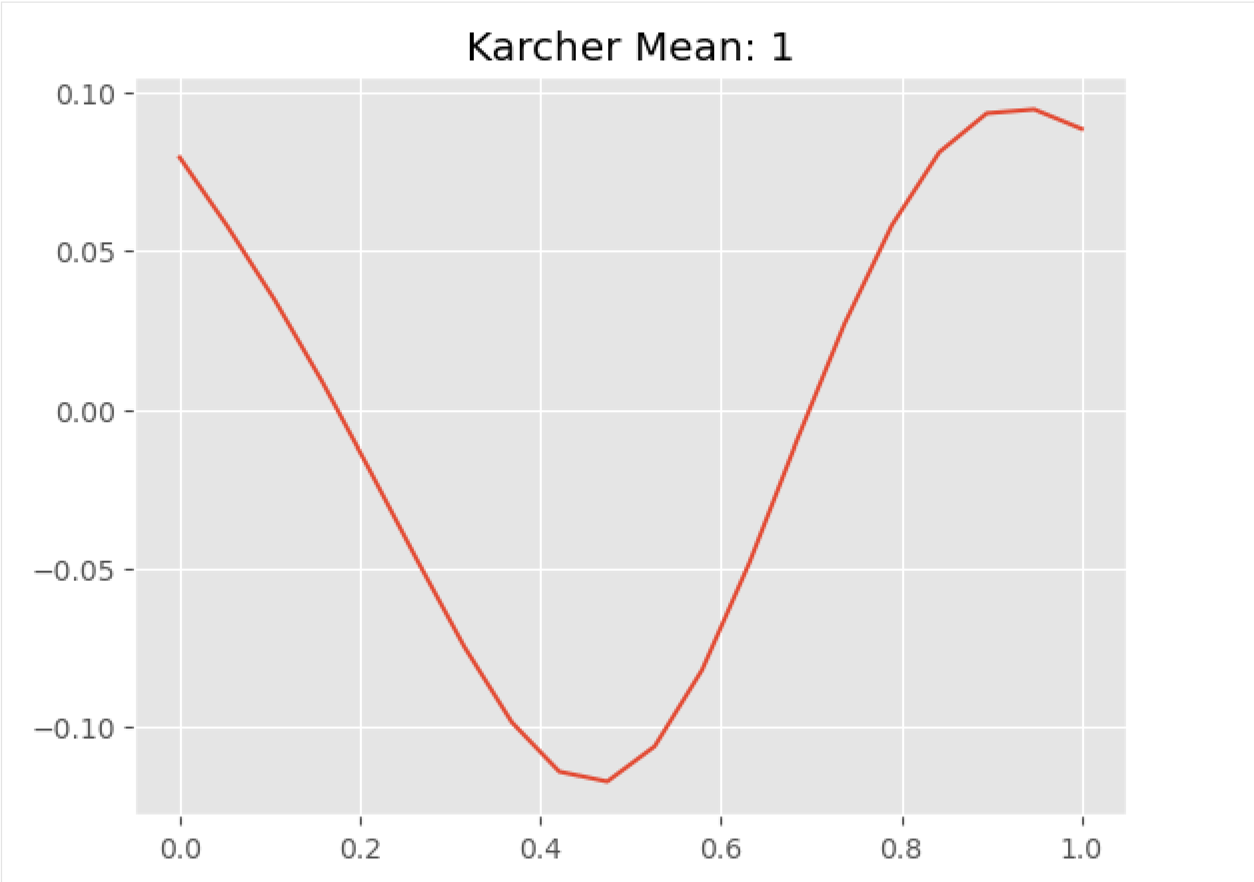
We will now plot the results

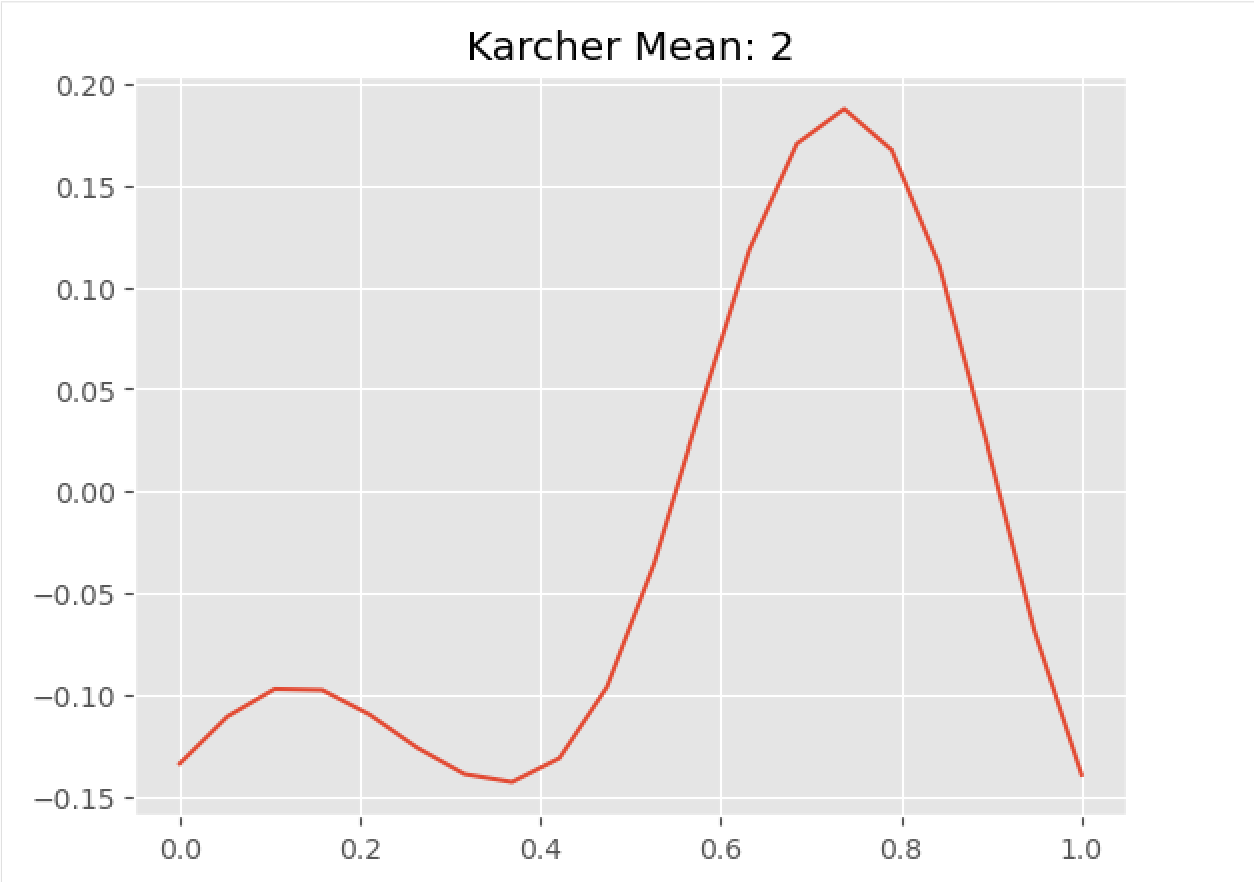
```
[ ]: obj.plot(multivariate=True)
```

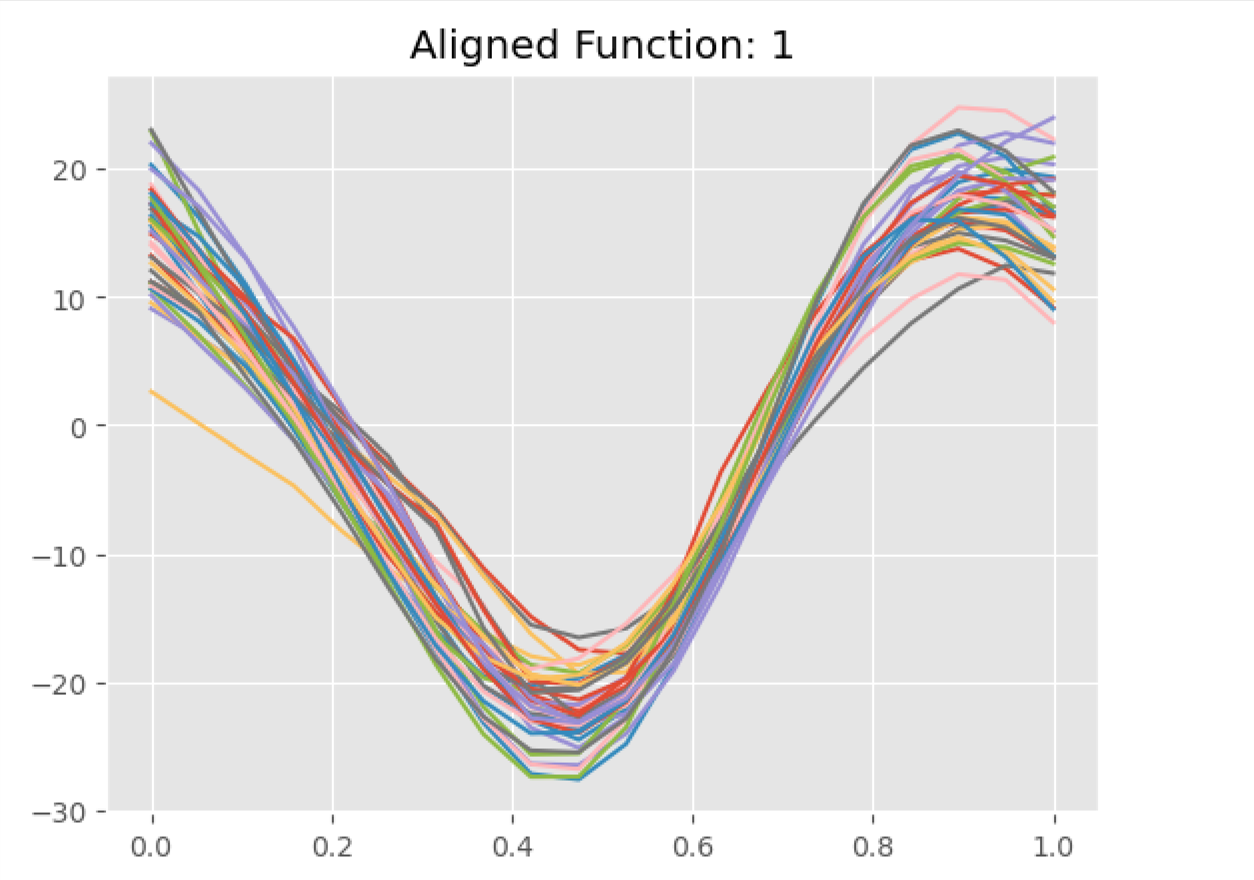


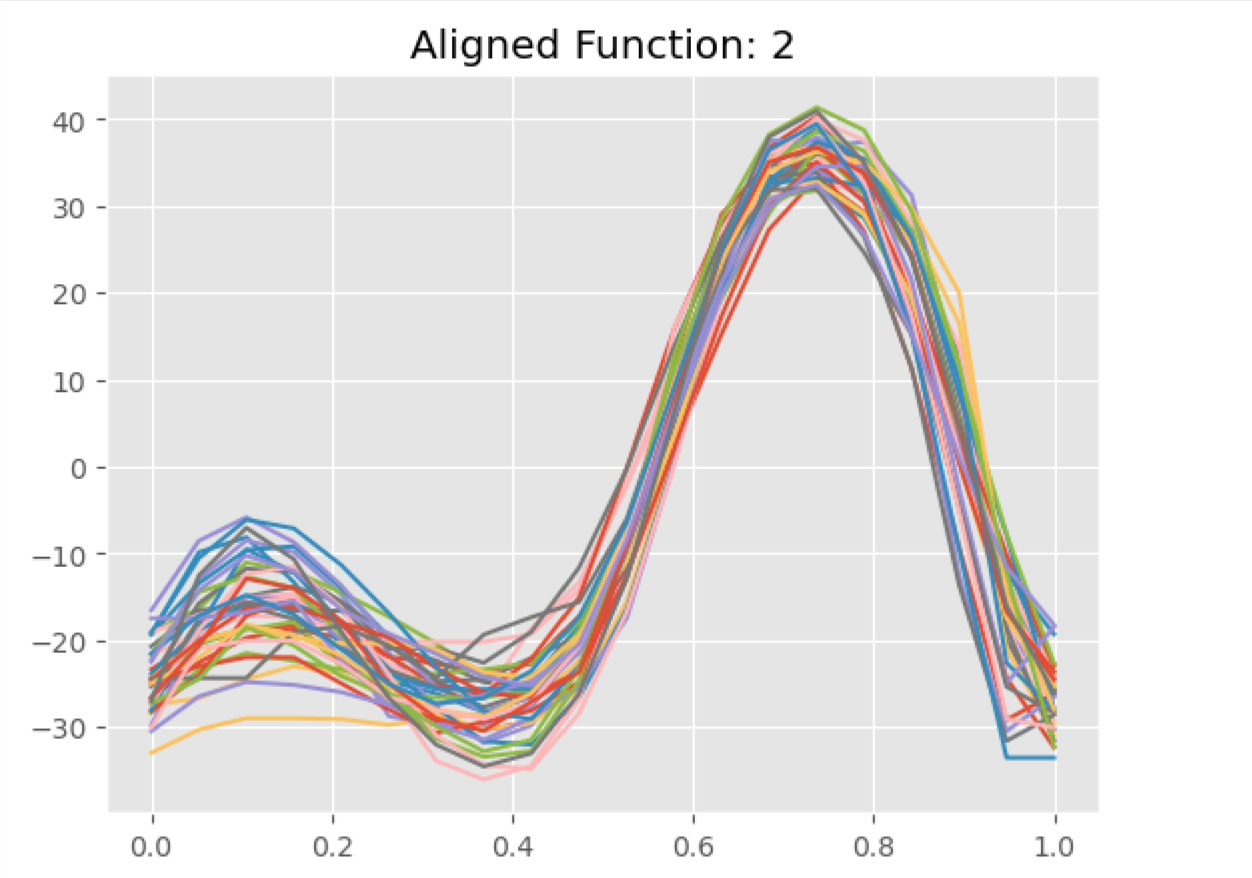

Original Function: 2
40
30
20
10
0
−10
−20
−30
0.0
0.2
0.4
0.6
0.8
1.0


Warping Functions
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

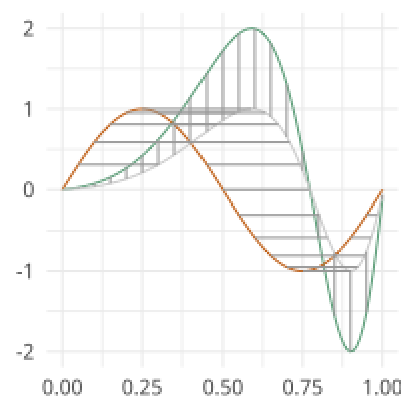

Karcher Mean: 1
0.10
0.05
0.00
−0.05
−0.10
0.0
0.2
0.4
0.6
0.8
1.0


Karcher Mean: 2
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
0.0
0.2
0.4
0.6
0.8
1.0


Aligned Function: 1
20
10
0
−10
−20
−30
0.0
0.2
0.4
0.6
0.8
1.0


Aligned Function: 2
40
30
20
10
0
−10
−20
−30
0.0
0.2
0.4
0.6
0.8
1.0

```
[ ]:
```


2
1
0
-1
-2
0.00
0.25
0.50
0.75
1.00

1.4 Elastic Curve Alignment

Otherwise known as time warping in the literature is at the center of elastic functional data analysis. Here our goal is to separate out the horizontal and vertical variability of the open/closed curves

[1]:
```
import fdasrsf as fs
import numpy as np
```

Load in our example data

[2]:
```
data = np.load('../../bin/MPEG7.npz',allow_pickle=True)
Xdata = data['Xdata']
curve = Xdata[0,1]
n,M = curve.shape
K = Xdata.shape[1]

beta = np.zeros((n,M,K))
for i in range(0,K):
    beta[:,:,i] = Xdata[0,i]
```

We will then construct the `fdacurve` object

[3]:
```
obj = fs.fdacurve(beta,N=M)
```

We then will compute karcher mean of the curves

[4]:
```
obj.karcher_mean()
```

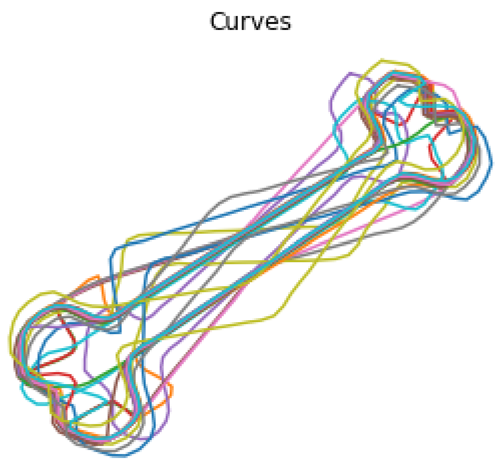
```
Computing Karcher Mean of 20 curves in SRVF space..
updating step: 1
updating step: 2
updating step: 3
updating step: 4
updating step: 5
updating step: 6
updating step: 7
```

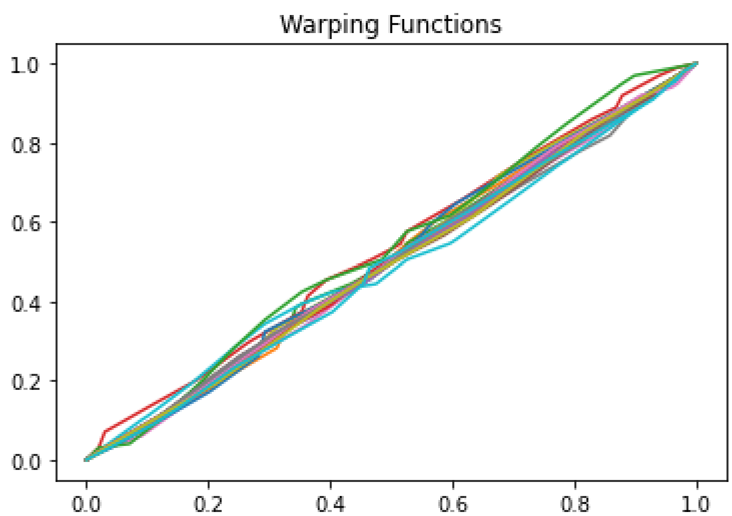
We then can align the curves to the karcher mean

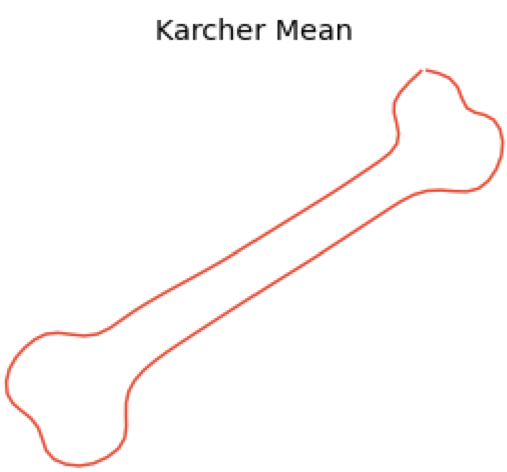
[5]:
```
obj.srvf_align(rotation=False)
```

Plot the results

[6]:
```
obj.plot()
```


Curves


Warping Functions
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Karcher Mean

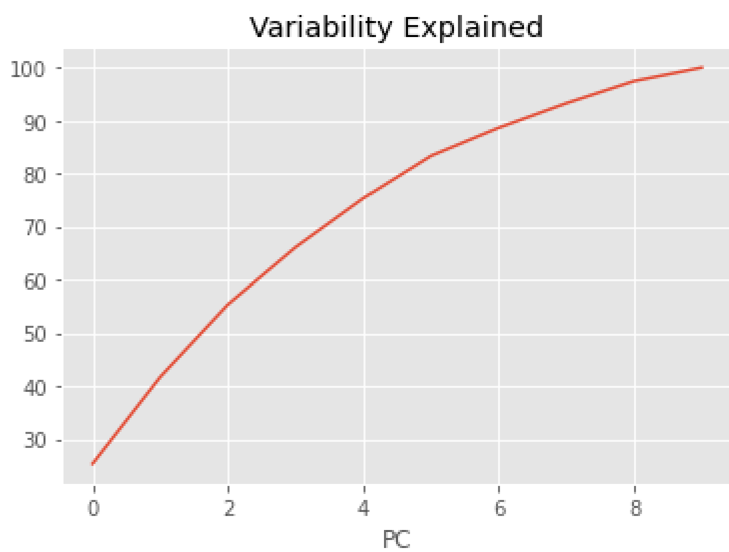
1.4.1 Shape PCA

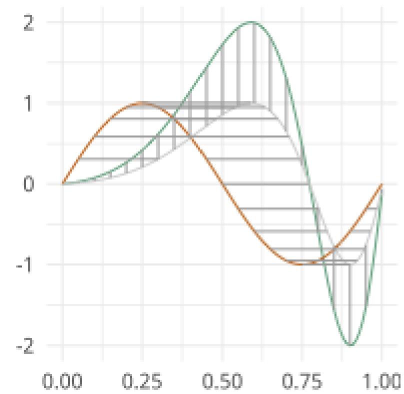
We then can compute the Karcher covariance and compute the shape pca

[7]:
```
obj.karcher_cov()
obj.shape_pca()
```

Plot the principal directions

[8]:
```
obj.plot_pca()
```

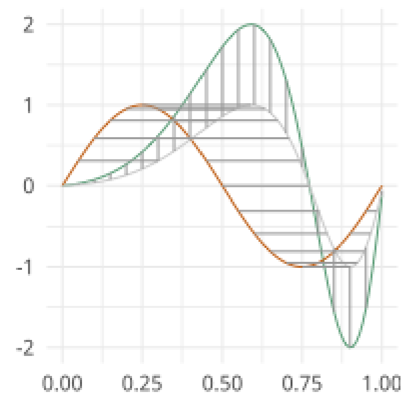



2
1
0
-1
-2
0.00
0.25
0.50
0.75
1.00

CHAPTER

TWO

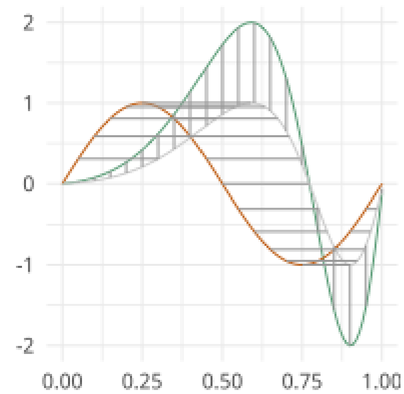
FUNCTIONAL ALIGNMENT



CHAPTER

THREE

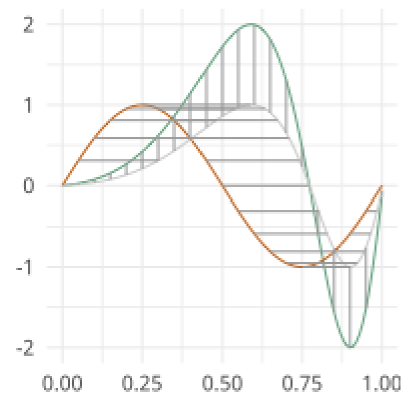
FUNCTIONAL PRINCIPAL COMPONENT ANALYSIS



CHAPTER

FOUR

ELASTIC FUNCTIONAL BOXPLOTS



CHAPTER

FIVE

FUNCTIONAL PRINCIPAL LEAST SQUARES

Partial Least Squares using SVD

moduleauthor:: J. Derek Tucker <jdtuck@sandia.gov>

`fPLS.pls_svd`(*time*, *qf*, *qg*, *no*, *alpha=0.0*)

This function computes the partial least squares using SVD

Parameters

- **time** – vector describing time samples
- **qf** – numpy ndarray of shape (M,N) of N functions with M samples
- **qg** – numpy ndarray of shape (M,N) of N functions with M samples
- **no** – number of components
- **alpha** – amount of smoothing (Default = 0.0 i.e., none)

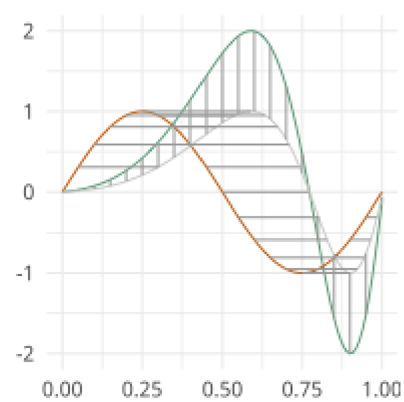
Return type
numpy ndarray

Return wqf
f weight function

Return wqg
g weight function

Return alpha
smoothing value

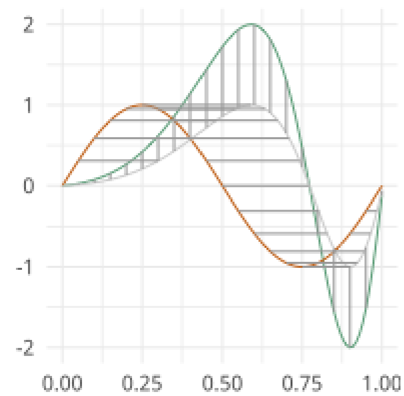
Return values
singular values



CHAPTER

SIX

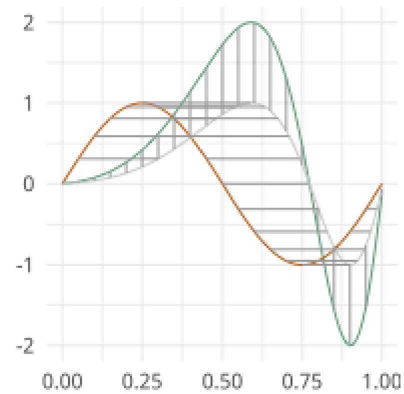
ELASTIC REGRESSION



CHAPTER

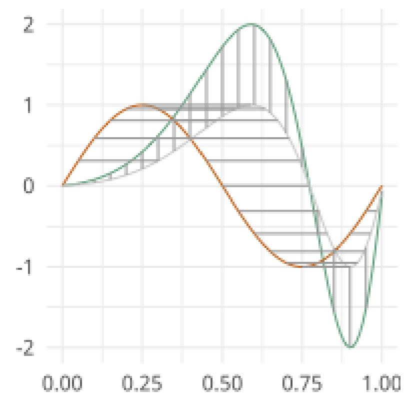
SEVEN

ELASTIC PRINCIPAL COMPONENT REGRESSION



EIGHT

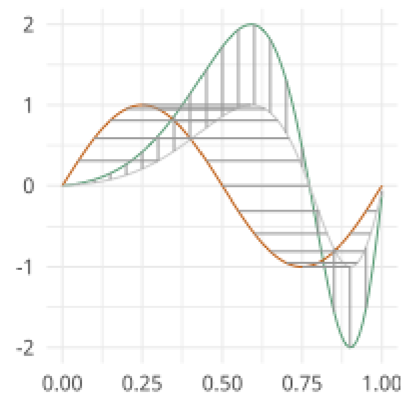
ELASTIC FUNCTIONAL CHANGEPOINT



CHAPTER

NINE

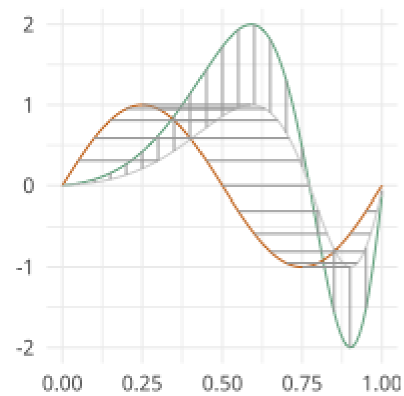
ELASTIC GLM REGRESSION



CHAPTER

TEN

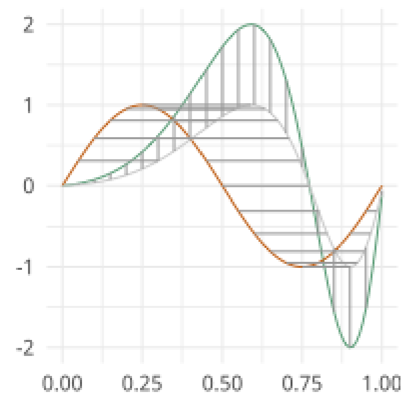
ELASTIC FUNCTIONAL TOLERANCE BOUNDS



CHAPTER

ELEVEN

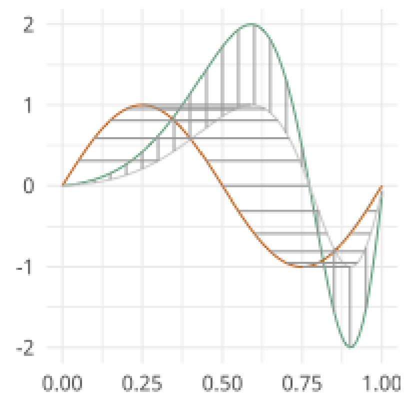
ELASTIC FUNCTIONAL CLUSTERING



CHAPTER

TWELVE

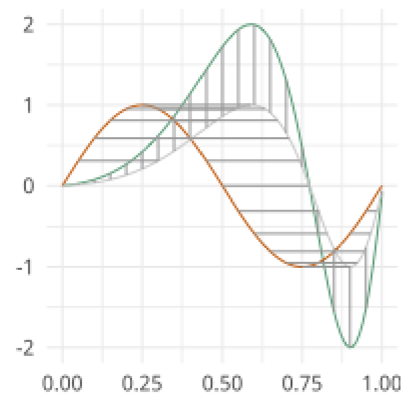
ELASTIC IMAGE WARPING



CHAPTER

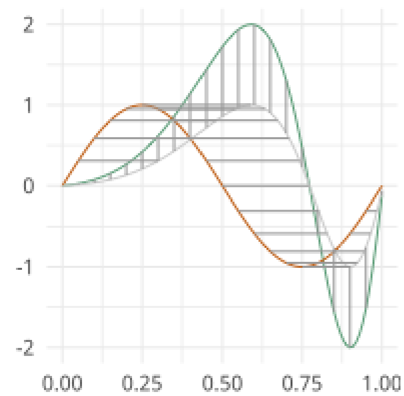
THIRTEEN

CURVE REGISTRATION



CHAPTER FOURTEEN

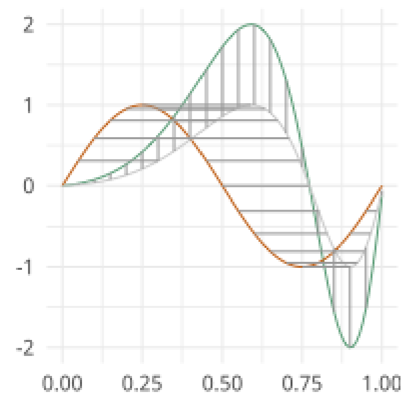
SRVF GEODESIC COMPUTATION



CHAPTER

FIFTEEN

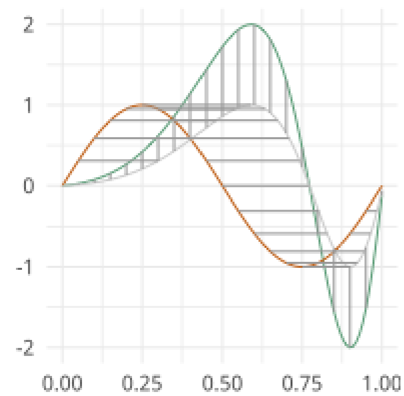
UTILITY FUNCTIONS



CHAPTER

SIXTEEN

CURVE FUNCTIONS



CHAPTER SEVENTEEN

UMAP EFDA METRICS

Distance metrics for functions and curves in R^n for use with UMAP (https://github.com/lmcinnes/umap)

moduleauthor:: J. Derek Tucker <jdtuck@sandia.gov>

`umap_metric.`**`efda_distance`**(*q1*, *q2*, *alpha=0*)

” calculates the distances between two curves, where q2 is aligned to q1. In other words calculates the elastic distances/ This metric is set up for use with UMAP or t-sne from scikit-learn

Parameters

- **q1** – vector of size N
- **q2** – vector of size N
- **alpha** – weight between phase and amplitude (default = 0, returns amplitude)

Return type
scalar

Return dist
amplitude distance

`umap_metric.`**`efda_distance_curve`**(*beta1*, *beta2*, *closed*)

” calculates the distances between two curves, where beta2 is aligned to beta1. In other words calculates the elastic distance. This metric is set up for use with UMAP or t-sne from scikit-learn

Parameters

- **beta1** – vector of size n*M
- **beta2** – vector of size n*M
- **closed** –

(0) if open curves and (1) if closed curves

Return type
scalar

Return dist
shape distance

CHAPTER

EIGHTEEN

REFERENCES

Tucker, J. D. 2014, Functional Component Analysis and Regression using Elastic Methods. Ph.D. Thesis, Florida State University.

Robinson, D. T. 2012, Function Data Analysis and Partial Shape Matching in the Square Root Velocity Framework. Ph.D. Thesis, Florida State University.

Huang, W. 2014, Optimization Algorithms on Riemannian Manifolds with Applications. Ph.D. Thesis, Florida State University.

Srivastava, A., Wu, W., Kurtek, S., Klassen, E. and Marron, J. S. (2011). Registration of Functional Data Using Fisher-Rao Metric. arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W. and Srivastava, A. (2013). Generative models for functional data using phase and amplitude separation. Computational Statistics and Data Analysis 61, 50-66.

J. D. Tucker, W. Wu, and A. Srivastava, "Phase-Amplitude Separation of Proteomics Data Using Extended Fisher-Rao Metric," Electronic Journal of Statistics, Vol 8, no. 2. pp 1724-1733, 2014.

J. D. Tucker, W. Wu, and A. Srivastava, "Analysis of signals under compositional noise With applications to SONAR data," IEEE Journal of Oceanic Engineering, Vol 29, no. 2. pp 318-330, Apr 2014.

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

S. Kurtek, A. Srivastava, and W. Wu. Signal estimation under random time-warpings and nonlinear signal alignment. In Proceedings of Neural Information Processing Systems (NIPS), 2011.

Wen Huang, Kyle A. Gallivan, Anuj Srivastava, Pierre-Antoine Absil. "Riemannian Optimization for Elastic Shape Analysis", Short version, The 21st International Symposium on Mathematical Theory of Networks and Systems (MTNS 2014).

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

W. Xie, S. Kurtek, K. Bharath, and Y. Sun, A geometric approach to visualization of variability in functional data, Journal of American Statistical Association 112 (2017), pp. 979-993.

Lu, Y., R. Herbei, and S. Kurtek, 2017: Bayesian registration of functions with a Gaussian process prior. Journal of Computational and Graphical Statistics, 26, no. 4, 894–904.

Lee, S. and S. Jung, 2017: Combined analysis of amplitude and phase variations in functional data. arXiv:1603.01775 [stat.ME], 1–21.

J. D. Tucker, J. R. Lewis, and A. Srivastava, "Elastic Functional Principal Component Regression," Statistical Analysis and Data Mining, vol. 12, no. 2, pp. 101-115, 2019.

J. D. Tucker, J. R. Lewis, C. King, and S. Kurtek, "A Geometric Approach for Computing Tolerance Bounds for Elastic Functional Data," Journal of Applied Statistics, 10.1080/02664763.2019.1645818, 2019.

T. Harris, J. D. Tucker, B. Li, and L. Shand, "Elastic depths for detecting shape anomalies in functional data," Technometrics, 10.1080/00401706.2020.1811156, 2020.

M. K. Ahn, J. D. Tucker, W. Wu, and A. Srivastava. "Regression Models Using Shapes of Functions as Predictors" Computational Statistics and Data Analysis, 10.1016/j.csda.2020.107017, 2020.

J. D. Tucker, L. Shand, and K. Chowdhary. "Multimodal Bayesian Registration of Noisy Functions using Hamiltonian Monte Carlo", Computational Statistics and Data Analysis, accepted, 2021.

X. Zhang, S. Kurtek, O. Chkrebtii, and J. D. Tucker, "Elastic k-means clustering of functional data for posterior exploration, with an application to inference on acute respiratory infection dynamics", arXiv:2011.12397 [stat.ME], 2020.

Q. Xie, S. Kurtek, E. Klassen, G. E. Christensen and A. Srivastava. Metric-based pairwise and multiple image registration. IEEE European Conference on Computer Vision (ECCV), September, 2014

J. D. Tucker and D. Yarger, "Elastic Functional Changepoint Detection of Climate Impacts from Localized Sources", Envirometrics, 10.1002/env.2826, 2023.

CHAPTER

NINETEEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX