Note

Click here to download the full example code

Pairwise alignment

Shows the usage of the elastic registration to perform a pairwise alignment.

```
# Author: Pablo Marcos Manchón
# License: MIT

# sphinx_gallery_thumbnail_number = 5


import skfda
import matplotlib.pyplot as plt
import matplotlib.colors as clr
import numpy as np
```

Given any two functions f and g, we define their pairwise alignment or registration to be the problem of finding a warping function γ^* such that a certain energy term $E[f, g \circ \gamma]$ is minimized.

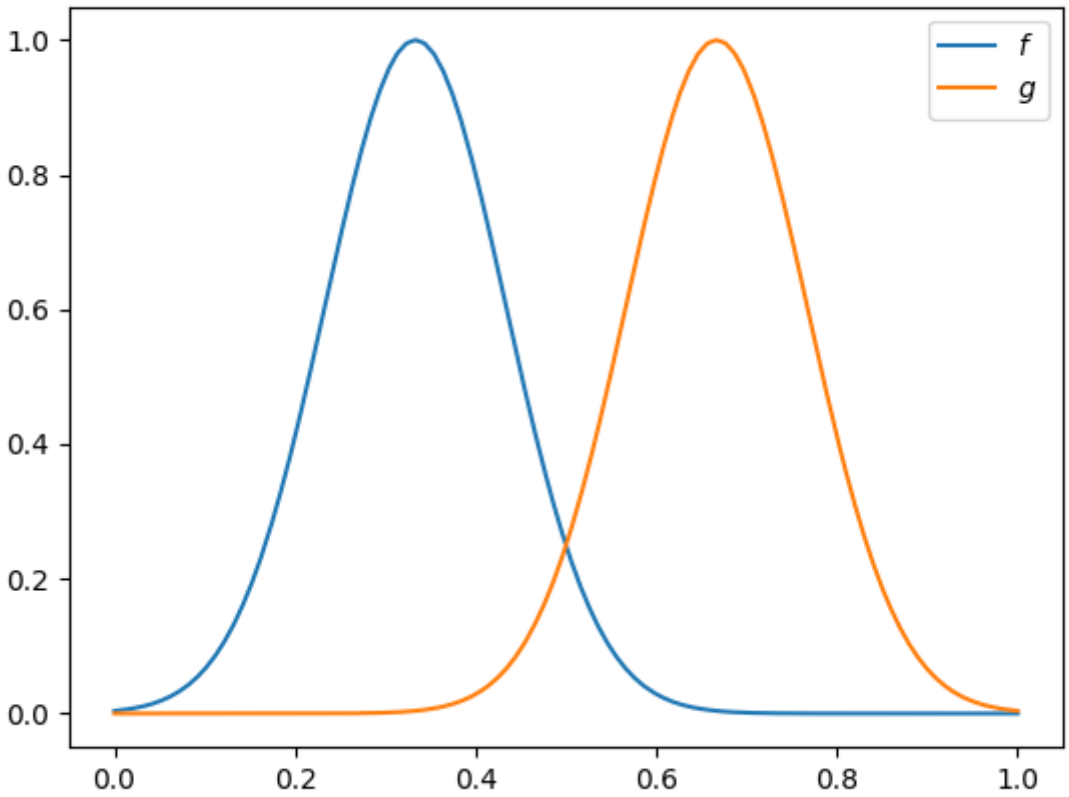
$$\gamma^* = *argmin_{\gamma \in \Gamma} E[f \circ \gamma, g]$$

In the case of elastic registration it is taken as energy function the Fisher-Rao distance with a penalisation term, due to the property of invariance to reparameterizations of warpings functions.

$$E[f \circ \gamma, g] = d_{FR}(f \circ \gamma, g)$$

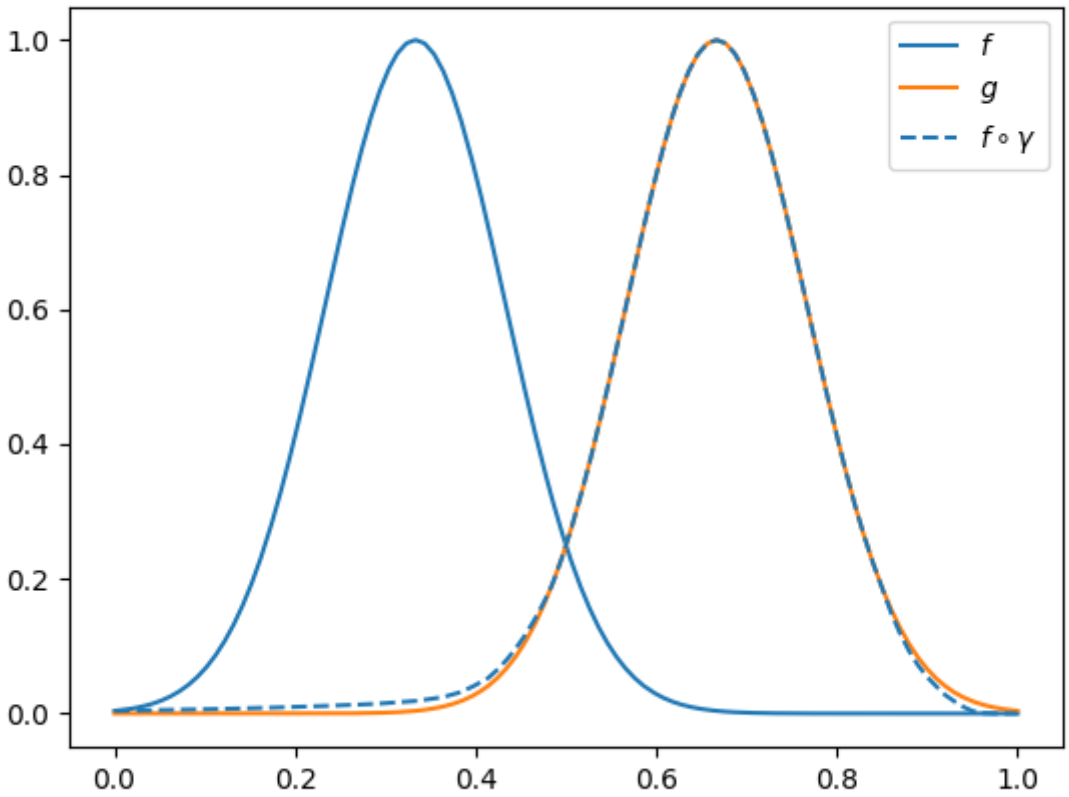
Firstly, we will create two unimodal samples, f and g, defined in [0, 1] wich will be used to show the elastic registration. Due to the similarity of these curves can be aligned almost perfectly between them.

```
# Samples with modes in 1/3 and 2/3
fd = skfda.datasets.make_multimodal_samples(n_samples=2, modes_location=[1/3,2/3],
                                            random_state=1, start=0, mode_std=.01)

fd.plot()
plt.legend(['$f$', '$g$'])
```



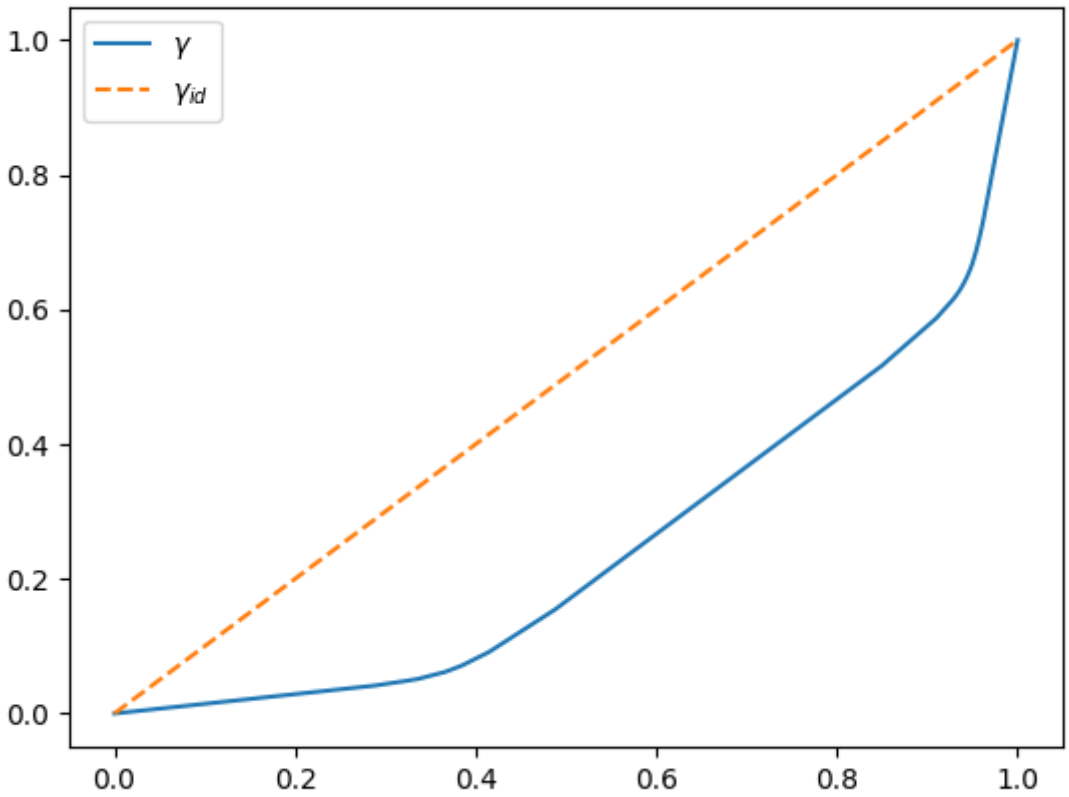
In this example g will be used as template and f will be aligned to it. In the following figure it is shown the result of the registration process, wich can be computed using `elastic_registration`.

```
f, g = fd[0], fd[1]

# Aligns f to g
fd_align = skfda.preprocessing.registration.elastic_registration(f, g)


plt.figure()

fd.plot()
fd_align.plot(color='C0', linestyle='--')


# Legend
plt.legend(['$f$', '$g$', '$f \\circ \\gamma $'])
```



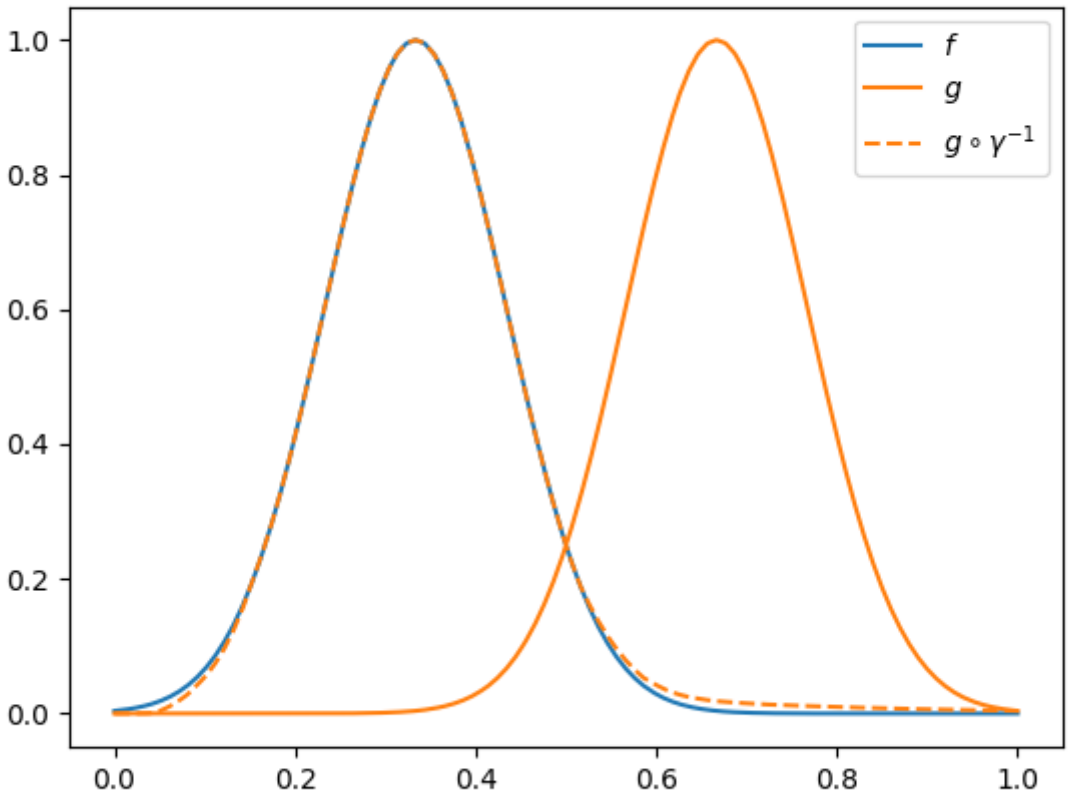
The non-linear transformation γ applied to f in the alignment can be obtained using `elastic_registration_warping`.

```
# Warping to align f to g
warping = skfda.preprocessing.registration.elastic_registration_warping(f, g)

plt.figure()

# Warping used
warping.plot()

# Plot identity
t = np.linspace(0, 1)
plt.plot(t, t, linestyle='--')

# Legend
plt.legend(['$\\gamma$', '$\\gamma_{id}$'])
```



The transformation necessary to align g to f will be the inverse of the original warping function, γ^{-1}. This fact is a consequence of the use of the Fisher-Rao metric as energy function.

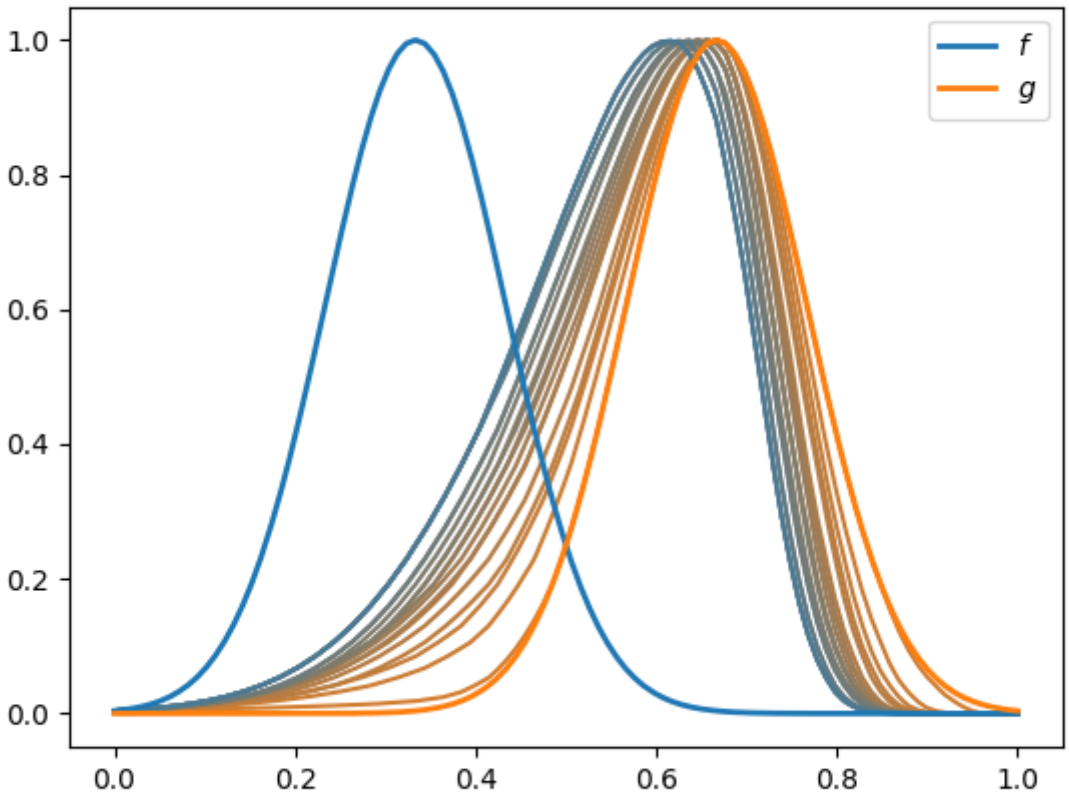
```
warping_inverse = skfda.preprocessing.registration.invert_warping(warping)


plt.figure()

fd.plot(label='$f$')
g.compose(warping_inverse).plot(color='C1', linestyle='--')


# Legend
plt.legend(['$f$', '$g$', '$g \\circ \\gamma^{-1} $'])
```



The amount of deformation used in the registration can be controlled by using a variation of the metric with a penalty term $\lambda \mathcal{R}(\gamma)$ wich will reduce the elasticity of the metric.

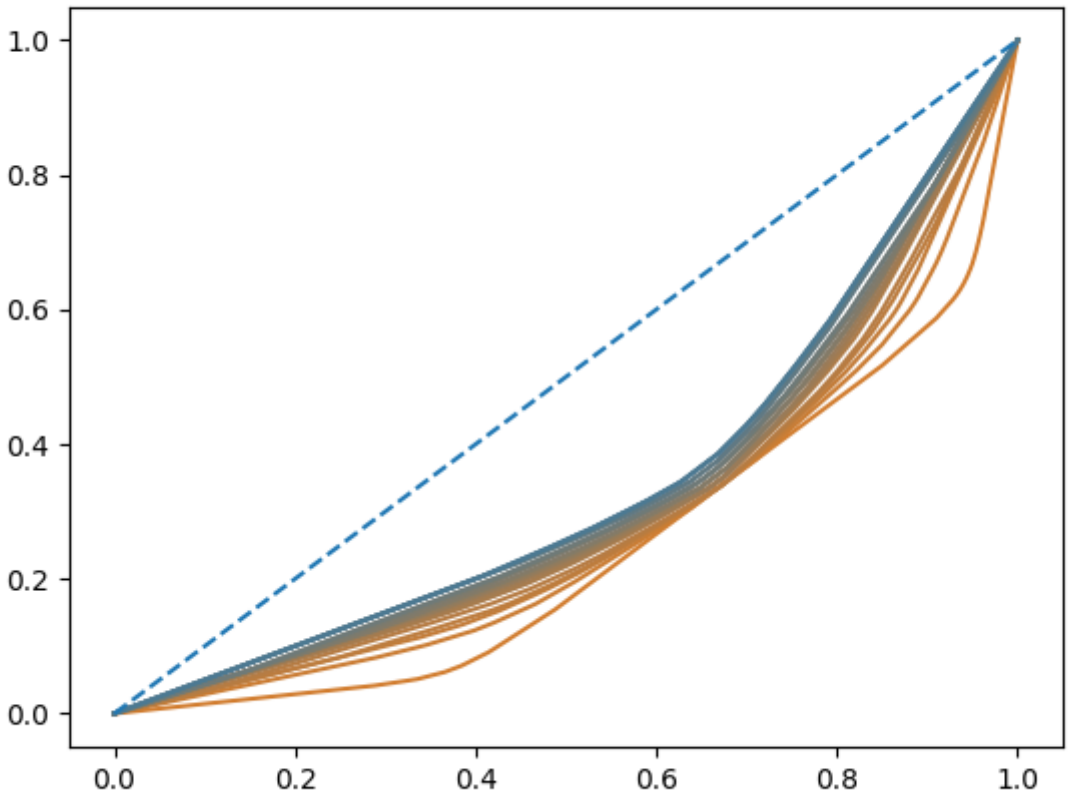
The following figure shows the original curves and the result to the alignment varying λ from 0 to 0.2.

```
# Values of lambda
lambdas = np.linspace(0, .2, 20)

# Creation of a color gradient
cmap = clr.LinearSegmentedColormap.from_list('custom cmap', ['C1','C0'])
color = cmap(.2 + 3*lambdas)

plt.figure()

for lam, c in zip(lambdas, color):
    # Plots result of alignment
    skfda.preprocessing.registration.elastic_registration(f, g, lam=lam).plot(color=c)


f.plot(color='C0', linewidth=2., label='$f$')
g.plot(color='C1', linewidth=2., label='$g$')

# Legend
plt.legend()
```



This phenomenon of loss of elasticity is clearly observed in the warpings used, since as the term of penalty increases, the functions are closer to γ_{id}.

```
plt.figure()

for lam, c in zip(lambdas, color):
    skfda.preprocessing.registration.elastic_registration_warping(f, g,
lam=lam).plot(color=c)

# Plots identity
plt.plot(t,t,  color='C0', linestyle="--")
```

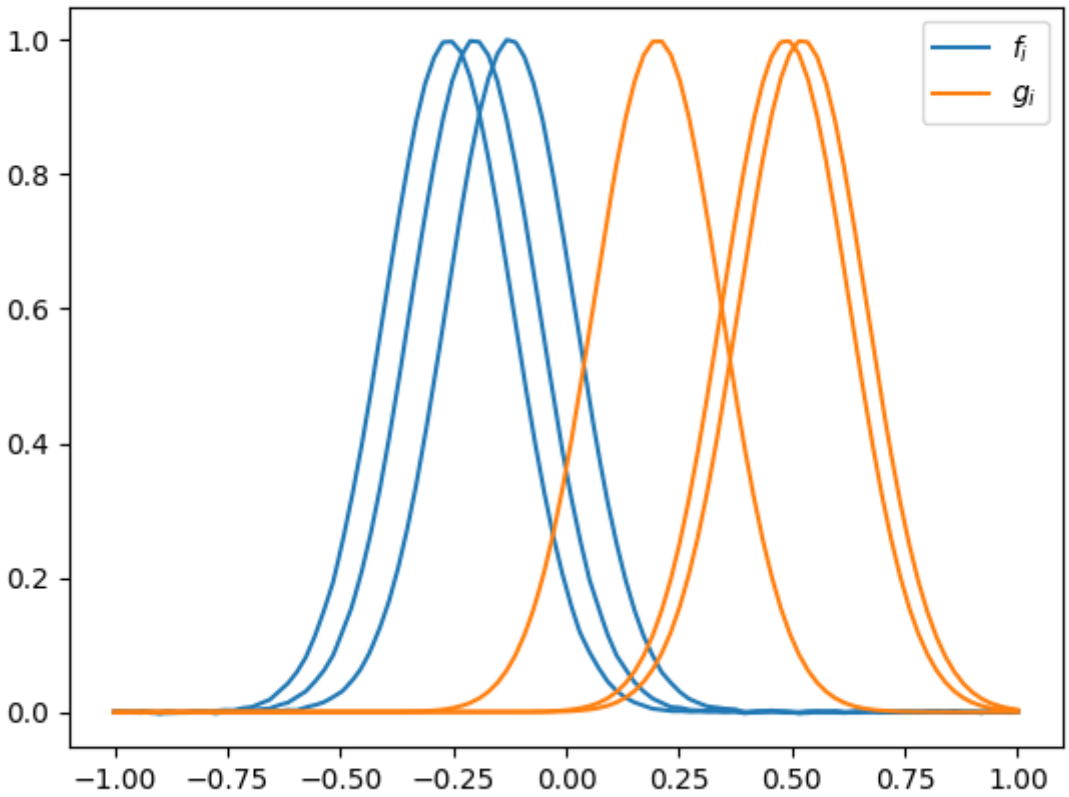


We can perform the pairwise of multiple curves at once. We can use a single curve as template to align a set of samples to it or a set of templates to make the alignemnt the two sets.

In the elastic registration example it is shown the alignment of multiple curves to the same template.

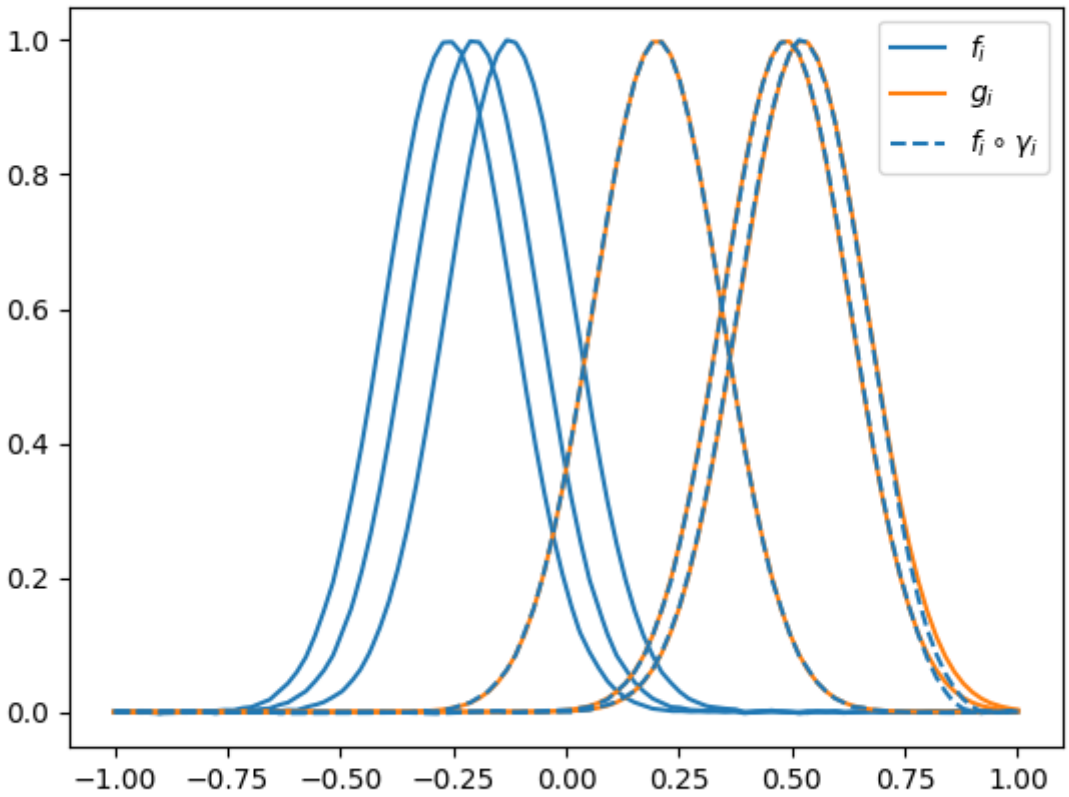
We will build two sets with 3 curves each, $\{f_i\}$ and $\{g_i\}$.

```
# Creation of the 2 sets of functions
state = np.random.RandomState(0)

location1 = state.normal(loc=-.3, scale=.1, size=3)
fd = skfda.datasets.make_multimodal_samples(n_samples=3, modes_location=location1,
                                            noise=.001 ,random_state=1)

location2 = state.normal(loc=.3, scale=.1, size=3)
g = skfda.datasets.make_multimodal_samples(n_samples=3, modes_location=location2,
                                           random_state=2)

# Plot of the sets
plt.figure()

fd.plot(color="C0", label="$f_i$")
fig, ax = g.plot(color="C1", label="$g_i$")

l = ax[0].get_lines()
plt.legend(handles=[l[0], l[-1]])
```



The following figure shows the result of the pairwise alignment of $\{f_i\}$ to $\{g_i\}$.

```
plt.figure()

# Registration of the sets
fd_registered = skfda.preprocessing.registration.elastic_registration(fd, g)

# Plot of the curves
fig, ax = fd.plot(color="C0", label="$f_i$")
l1 = ax[0].get_lines()[-1]
g.plot(color="C1", label="$g_i$")
l2 = ax[0].get_lines()[-1]
fd_registered.plot(color="C0", linestyle="--", label="$f_i \\circ \\gamma_i$")
l3 = ax[0].get_lines()[-1]

plt.legend(handles=[l1, l2, l3])

plt.show()
```



- Srivastava, Anuj & Klassen, Eric P. (2016). Functional and shape data analysis. In *Functional Data and Elastic Registration* (pp. 73-122). Springer.
- J. S. Marron, James O. Ramsay, Laura M. Sangalli and Anuj Srivastava (2015). Functional Data Analysis of Amplitude and Phase Variation. Statistical Science 2015, Vol. 30, No. 4

Total running time of the script: (0 minutes 2.988 seconds)

Download Python source code: plot_pairwise_alignment.py

Download Jupyter notebook: plot_pairwise_alignment.ipynb