Computing the center of a shape

The notion of center for a shape in $\mathbb{R}^d$ depends on which representation of the shape you are considering. If you view the shape as a point cloud or as a landmark-based shape analysis $p_1, \ldots, p_n \in \mathbb{R}^d$, then the center can be computed using the mean

$$\frac{1}{n}\sum_{i=1}^{n} p_i$$

where the sum is applied using standard vector addition.

However, if you view the shape as a continuous parametrized curve $\beta : [0,1] \to \mathbb{R}^n$, then the center is defined using an integral

$$\int_0^1 \beta(t)dt$$

In this exercise, we will compare the two different shape representations.

- load the `closed_curves_2D/toydata.mat` toy
- see `task_000003` and use `data['C'][0,:,0]`,`data['C'][1,:,0]` for the points.

Compute the center as a landmark-base shape space

- consider using `numpy.sum` to sum over axis 1. You should get a single point in $\mathbb{R}^2$.
- consider using built in `numpy.ndarray.shape` variable to get the number of points you are summing over.

Compute the center using parametrized curve based shape space

Let $\beta : [0,1] \to \mathbb{R}^2$ denote your parametrized shape curve. Let $0 = t_1 < t_2 < t_n = 1$ denote the time points you have observed values $\beta(t_1), \ldots, \beta(t_n)$. If β is smooth, then one can approximate it using a piecewise-linear curve.

- Compute the length of each line segment for $i = 1 \ldots n-1$, compute line segment distances
$$|\beta(t_i) - \beta(t_i - 1)|$$
  - consider using `nump.ndarray` built in indexes to get all but the first and all but the last element in the array. See https://numpy.org/doc/stable/user/basics.indexing.html#basics-indexing.
  - consider using `numpy.sum` to sum over axis 0. You should get a vector in $\mathbb{R}^(n-1)$ representing each length.

- Compute the cumulative sum `numpy.cumsum`
- use `numpy.trapz` to compute the integral of the parametrized curve over axis 1.
  - you should get a single point if done correct.

Plot the centered landmark shape and the centered curve on the same plot

Translate the points in `data['C'][0,:,0]`,`data['C'][1,:,0]` using the mean computed in the landmark-base shape space and use `matplotlib.pyplot.scatter` to plot it.

In the same figure (before you use `matplotlib.pyplot.show`), translate the points in `data['C'][0,:,0]`,`data['C'][1,:,0]` using the center computed for the parametrized curve based shape space. Use `matplotlib.pyplot.plot` to plot it as a curve.

You should see two distinct (slightly offset) shapes plotted .