Documentation of the spline representation in FOCUS

Nicola Lonigro, Caoxiang Zhu

April 2022

The paper has been published and you can find it at https://doi.org/10.1088/1741-4326/ac2ff3

1 Spline representation

This file contains all the functions required to operate the spline representation. The function

- *eval_spline_basis* computes the third order basis functions $B_i(x)$, that are then used to define the spline as

$$S(x) = \sum_i C_i B_i(x) \tag{1}$$

  with C_i the control points of the spline, that are the geometric degree of freedom optimized by the code. They are computed iteratively from the Cox-De Boor algorithm

$$B_{i,0}(x) = \begin{cases} 1, & t_i < x < t_{i+1} \\ 0, & otherwise \end{cases} \tag{2}$$

$$B_{i,k}(x) = \frac{x - t_i}{t_{i+k} - t_i} B_{i,k-1}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} B_{i+1,k-1}(x) \tag{3}$$

  where the t_i are the elements of the knot vector characterizing the spline.

- *eval_spline_basis1* computes the first derivative of the third order basis function with respect to the parameter x. The computed values are stored as a property of the *Splines* structure, to be used in the derivatives of the cost functions.

$$\frac{\partial B_{i,3}(x)}{\partial x} = \begin{cases} \frac{1}{t_{i+3}-t_i}\left\{B_{i,2}(x) + \frac{x-t_i}{t_{i+2}-t_i}\left[B_{i,1}(x) + \frac{x-t_i}{t_{i+1}-t_i}\right]\right\} & t_i < x < t_{i+1} \\ \frac{1}{t_{i+3}-t_i}\left\{B_{i,2}(x) + \frac{x-t_i}{t_{i+2}-t_i}\left[B_{i,1}(x) - \frac{x-t_i}{t_{i+2}-t_{i+1}}\right] - \frac{1}{t_{i+3}-t_{i+1}}\left[B_{i+1,1}(x) - \frac{t_{i+3}-x}{t_{i+2}-t_{i+1}}\right]\right\} - \frac{1}{t_{i+4}-t_{i+1}}\left\{B_{i+1,2}(x) - \frac{t_{i+4}-x}{t_{i+3}-t_{i+1}}\left[B_{i+1,1}(x) + \frac{x-t_{i+1}}{t_{i+2}-t_{i+1}}\right]\right\} & t_{i+1} < x < t_{i+2} \\ \frac{1}{t_{i+3}-t_i}\left\{B_{i,2}(x) - \frac{x-t_i}{t_{i+3}-t_{i+1}}\left[B_{i+1,1}(x) + \frac{t_{i+3}-x}{t_{i+3}-t_{i+2}}\right]\right\} - \frac{1}{t_{i+4}-t_{i+1}}\left\{B_{i+1,2}(x) - \frac{t_{i+4}-x}{t_{i+3}-t_{i+1}}\left[B_{i+1,1}(x) - \frac{x-t_{i+1}}{t_{i+3}-t_{i+2}}\right] + \frac{1}{t_{i+4}-t_{i+2}}\left[B_{i+2,1}(x) - \frac{t_{i+4}-x}{t_{i+3}-t_{i+2}}\right]\right\} & t_{i+2} < x < t_{i+3} \\ \frac{1}{t_{i+4}-t_{i+1}}\left\{-B_{i+1,2}(x) - \frac{t_{i+4}-x}{t_{i+4}-t_{i+2}}\left[B_{i+2,1}(x) + \frac{t_{i+4}-x}{t_{i+4}-t_{i+3}}\right]\right\} & t_{i+3} < x < t_{i+4} \end{cases}$$

- *eval_spline_basis2* computes the second derivative of the third order basis function with respect to the parameter x. The computed values are stored as a property of the *Splines* structure, to be used in the derivatives of the cost functions.

$$\frac{\partial^2 B_{i,3}(x)}{\partial x^2} = \begin{cases} \frac{2}{t_{i+3}-t_i}\frac{1}{t_{i+2}-t_i}\left[B_{i,1}(x) + 2\frac{x-t_i}{t_{i+1}-t_i}\right] & t_i < x < t_{i+1} \\ \frac{2}{t_{i+3}-t_i}\left\{\frac{1}{t_{i+2}-t_i}\left[B_{i,1}(x) - \frac{x-t_i}{t_{i+2}-t_{i+1}}\right] - \frac{1}{t_{i+3}-t_{i+1}}\left[B_{i+1,1}(x) - \frac{t_{i+3}-x}{t_{i+2}-t_{i+1}}\right] - \frac{x-t_i}{t_{i+2}-t_{i+1}}\left[\frac{1}{t_{i+2}-t_i} + \frac{1}{t_{i+3}-t_{i+1}}\right] - \frac{1}{t_{i+4}-t_{i+1}}\left[B_{i+1,1}(x) + \frac{x-t_{i+1}}{t_{i+2}-t_{i+1}} - \frac{t_{i+4}-x}{t_{i+2}-t_{i+1}}\right]\right\} & t_{i+1} < x < t_{i+2} \\ \frac{2}{t_{i+3}-t_{i+1}}\left\{-\frac{1}{t_{i+3}-t_i}\left[B_{i+1,1}(x) + \frac{t_{i+3}-x}{t_{i+3}-t_{i+2}} - \frac{x-t_i}{t_{i+3}-t_{i+2}}\right] - \frac{1}{t_{i+4}-t_{i+1}}\left[B_{i+1,1}(x) - \frac{x-t_{i+1}}{t_{i+3}-t_{i+2}}\right]\right\} + \frac{2}{t_{i+4}-t_{i+1}}\left\{\frac{1}{t_{i+4}-t_{i+2}}\left[B_{i+2,1}(x) - \frac{t_{i+4}-x}{t_{i+3}-t_{i+2}}\right] - \frac{t_{i+4}-x}{t_{i+3}-t_{i+2}}\left[\frac{1}{t_{i+3}-t_{i+1}} + \frac{1}{t_{i+4}-t_{i+2}}\right]\right\} & t_{i+2} < x < t_{i+3} \\ \frac{2}{t_{i+4}-t_{i+1}}\frac{1}{t_{i+4}-t_{i+2}}\left[B_{i+2,1}(x) + 2\frac{t_{i+4}-x}{t_{i+4}-t_{i+3}}\right] & t_{i+3} < x < t_{i+4} \end{cases}$$

- *enforce_spline_periodicity* is a function that ensures the spline is periodic, by making sure the first and last control points are the same.

2 Cost function for straight outer coils

This cost function constraints the curvature on the outer side of the coil to obtain straighter coils. It uses a similar expression to the curvature cost function, but it is only applied to the outer part of the coil and uses an independent set of parameters to constrain the curvature on the outer part of the coil more strongly than on the inner part.
The expression of the cost function is

$$f_S = \frac{1}{N_c}\sum_{i=1}^{N_c}\frac{1}{L_i}\int_{t_{min}}^{t_{max}} W(t)(p_{1,2}(\kappa_i, \kappa_{0,s}) + \sigma_s \kappa_i^{\gamma_s})dt \ , \tag{4}$$

where $W(t)$ is a weight function that equals 1 on the outer part of the coil and 0 on the inner part. The representation of the coils is a function of t in $[t_{min}, t_{max}]$. This is equal to $[0, 2\pi]$ for the Fourier representation and [0,1] for the Spline representation. For a description of the remaining part of the expression, representing the constraint on the curvature of each point affected by the penality, see the documentation on the curvature function.
The weight function is given by

$$W(t) = \begin{cases} 1, & P_{xy}(t) > P_m + \beta P_d \\ 0, & otherwise \end{cases} \tag{5}$$

where P_m is the mean squared distance of the coil projection in the x-y plane from a user-defined point (x_0, y_0), usually corresponding to the center of the stellarator, and P_d is a measure for half the maximum radius of the coil. Considering the distance of the projection of a point along the coil in the x-y plane from the user-defined point (x_0, y_0)

$$P_{xy}(t) = (x_i(t) - x_0)^2 + (y_i(t) - y_0)^2 \ , \tag{6}$$

they are defined as

$$P_m = \overline{P_{xy}} \ , \qquad P_d = \frac{\max(P_{xy}) - P_m}{2} \ . \tag{7}$$

β is a user-defined parameter that can be used to tweak the section of the coil affected by the optimization.

With a value $\beta = 0$ the section of the coil with a projected distance greater than the mean value for that coil will all be affected while for a value $\beta = 2$ the new cost function will not be applied to any point. The value of β is read from the input parameter *straight_coeff* while x_0 and y_0 are read from *origin_surface_x* and *origin_surface_y* respectively.

The derivatives are computed as in the case of the curvature function, but only applied on the outer part of the coil.

The implementation of the cost function is completely independent from the curvature function, so a different expression for this penalty could be implemented in the future without affecting the curvature cost function.

How To Use

The cost function is controlled trough some parameters that are specific to this cost function:

- *weight_str* : Sets the "weight" (i.e. strength) of the constraint
- *straight_coeff* : Sets β, determining which part of the coil to consider as the outer part.
- *origin_surface_x*,*origin_surface_y* : set x_0 and y_0.

and some that are analogous of the ones used in the curvature cost function, but can be set independently

- *penfun_str*
- *str_alpha*
- *str_beta*
- *str_gamma*
- *str_sigma*