Fitting Smooth Spline to Data

Matthew Kelly

Tufts University - Optimal Control for Robotics

February 7, 2018

This document contains the mathematical details for constructing a smooth spline through a data set. In addition to fitting the data, the spline should be smooth (jerk-minimizing) and satisfy known boundary conditions precisely. The resulting spline can be generated by solving a linear system ($x = A\backslash b$).

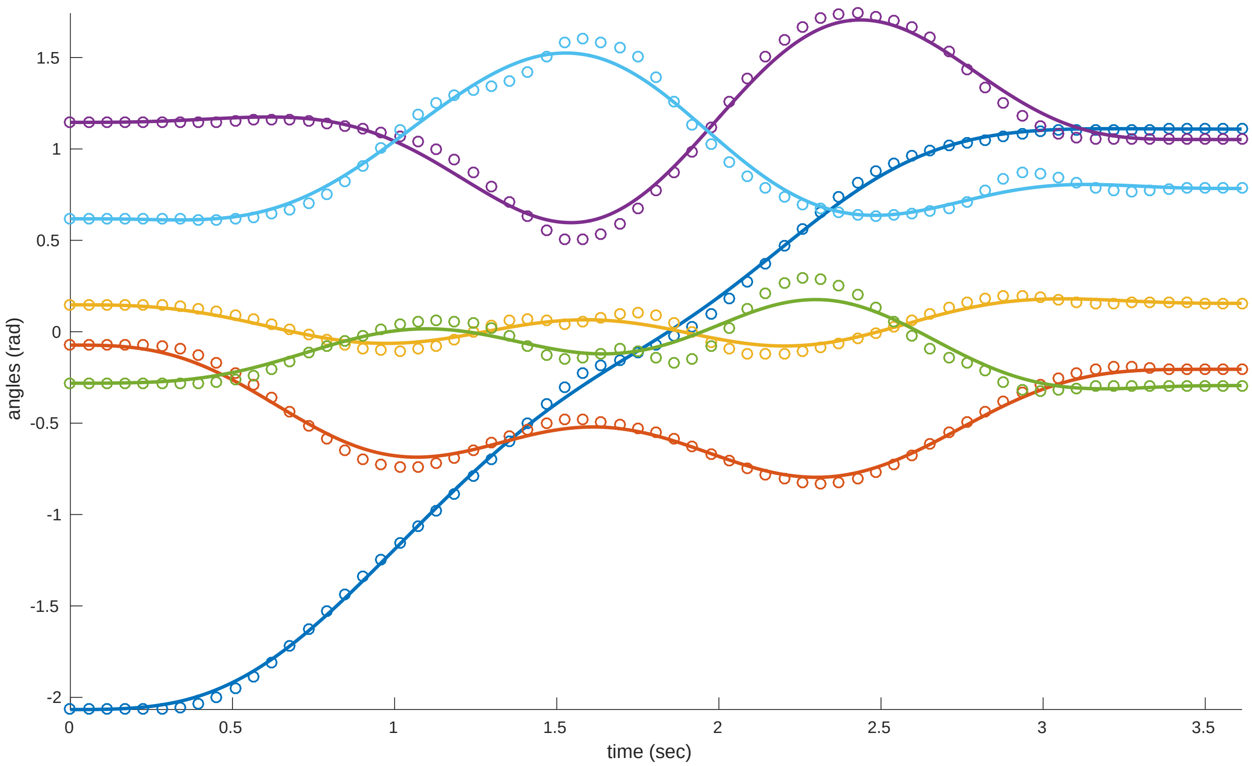


Figure 1: A spline that is fit to a six-dimensional data set, trading off between data-fitting and minimizing jerk. The slope and curvature at the start and end are set to zero.

1 Problem Statement

Simply put: fit a smooth function through a sequence of data points, *e.g.* Figure 1. More precisely:

- **Given:** a time-stamped set of points in space.
- **Find:** a smooth vector function.
- **Minimize:** integral of curvature-rate-squared and sum or squared-error.
- **Subject to:** value, slope, and curvature at the boundaries.

We will use $\boldsymbol{x}(t)$ to represent value as a function of time. The data set is $\{t_i, \bar{\boldsymbol{x}}_i\}$, which gives measured value at each time stamp. The time-domain of the data set is $[0, T]$.

1.1 Objective Function

The objective function is a weighted combination of two terms. The first is the sum of the squared-error between the candidate vector function and the points in the data set. The second is a smoothing-term, minimizing the integral of the third derivative of the function.

$$\boldsymbol{J}\big(\boldsymbol{x}(t)\big) \;=\; \frac{T}{N}\sum_{i=0}^{N}\big(\boldsymbol{x}(t_i) - \bar{\boldsymbol{x}}_i\big)^2 \;+\; \alpha\int_0^T \dddot{\boldsymbol{x}}^2(t)\,dt \tag{1}$$

1.2 Constraints

The vector function $\boldsymbol{x}(t)$ must be smooth: continuous value, slope, and curvature.

$$\boldsymbol{x}(t) \in \mathcal{C}^2 \tag{2}$$

We also require that the value, slope, and curvature be prescribed at the initial and final times. In practice these boundary constraints can be dropped if not required by the end-user.

$$\boldsymbol{x}(0) = \boldsymbol{x}_0 \qquad\qquad \boldsymbol{x}(T) = \boldsymbol{x}_T \tag{3}$$
$$\dot{\boldsymbol{x}}(0) = \dot{\boldsymbol{x}}_0 \qquad\qquad \dot{\boldsymbol{x}}(T) = \dot{\boldsymbol{x}}_T \tag{4}$$
$$\ddot{\boldsymbol{x}}(0) = \ddot{\boldsymbol{x}}_0 \qquad\qquad \ddot{\boldsymbol{x}}(T) = \ddot{\boldsymbol{x}}_T \tag{5}$$

2 Method

In this section we cover the precise details of how to construct the objective and constraint equations, given that we represent $\boldsymbol{x}(t)$ as a cubic spline.

2.1 Assumptions

To make this problem tractable, we will make two simplifying assumptions:

- the trajectory will be a cubic spline
- the knot points of the spline will be given

2.2 Outline

The problem outlined in Section 1 can be posed as a quadratic program, where $\mathcal{H}$ and $\mathbf{f}$ are the quadratic and linear cost matrices, $\mathcal{A}$ and $\mathbf{b}$ are the equality constraints, and $\boldsymbol{z}$ is the vector of decision variables: the coefficients of the cubic spline.

$$\begin{aligned} \text{minimize:} \qquad & \tfrac{1}{2}\boldsymbol{z}^T\mathcal{H}\boldsymbol{z} + \boldsymbol{f}^T\boldsymbol{z} && (6) \\ \text{subject to:} \qquad & \mathcal{A}\boldsymbol{z} = \boldsymbol{b} && (7) \end{aligned}$$

This is a special quadratic program: it does not have inequality constraints. As a result, this quadratic program can be solved as the linear system shown below, where $\boldsymbol{w}$ is a vector of Lagrange multipliers for the constraint equations.

$$\begin{bmatrix} \mathcal{H} & \mathcal{A}^T \\ \mathcal{A} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{f} \\ \boldsymbol{b} \end{bmatrix} \tag{8}$$

Once we have constructed the matrices $\mathcal{H}$, $\mathbf{f}$, $\mathcal{A}$, and $\mathbf{b}$, we can then solve the linear system (8) for the decision variables to $\boldsymbol{z}$: the spline coefficients.

2.3 Block Matrices

Note that the matrices $\mathcal{H}$ and $\mathcal{A}$ are large and sparse, as shown in Figure 2. These matrices should be implemented as sparse matrices, and the linear system solved using a sparse solver. The matrices $\mathcal{H}$ and $\mathcal{A}$ are constructed from smaller blocks, where each block corresponds to the coefficients of a single segment. For example, $\mathcal{H}_j$ is the 4×4 block in the $\mathcal{H}$ matrix that corresponds to the spline segment j. Similarly, $\mathcal{A}_{k,j}$ is the 3×4 block of coefficients for the k^{th} set of constraints on the coefficients of spline segment j.

2.4 Spline Definition

We will represent the trajectory as a cubic spline, with knot points T_j where $j \in \{0 \dots M\}$. A single segment of the spline is given below, where j is the segment index, and $\tau = t - T_j$ is the time since the start of the segment. The function $\boldsymbol{x}(t)$ is constructed by chaining together each of the segments $\boldsymbol{x}_j(t)$ in order.

$$\boldsymbol{x}_j(\tau) = \boldsymbol{A}_j\tau^3 + \boldsymbol{B}_j\tau^2 + \boldsymbol{C}_j\tau + \boldsymbol{D}_j \tag{9}$$

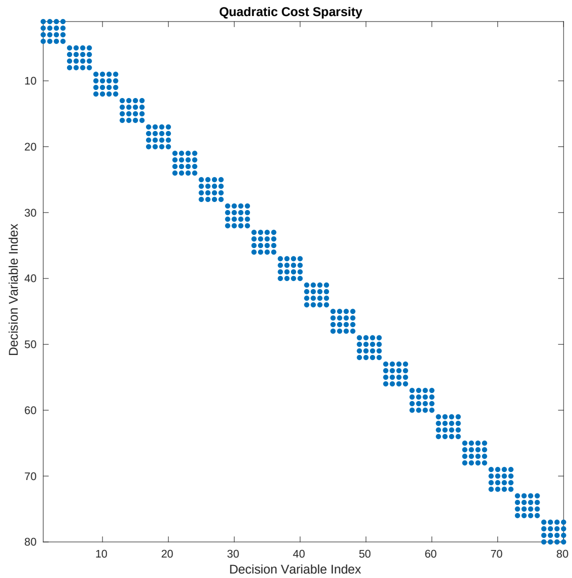
The derivatives of this spline are easily computed, as shown below:

$$\dot{\boldsymbol{x}}_j(\tau) = \frac{d}{dt}\boldsymbol{x}_j(\tau) = 3\boldsymbol{A}_j\tau^2 + 2\boldsymbol{B}_j\tau + \boldsymbol{C}_j \tag{10}$$

$$\ddot{\boldsymbol{x}}_j(\tau) = \frac{d}{dt}\dot{\boldsymbol{x}}_j(\tau) = 6\boldsymbol{A}_j\tau + 2\boldsymbol{B}_j \tag{11}$$

$$\dddot{\boldsymbol{x}}_j(\tau) = \frac{d}{dt}\ddot{\boldsymbol{x}}_j(\tau) = 6\boldsymbol{A}_j \tag{12}$$

We will represent the coefficients of a single segment of the spline as $\boldsymbol{z}_j = [\boldsymbol{D}_j, \boldsymbol{C}_j, \boldsymbol{B}_j, \boldsymbol{A}_j]^T$, and the coefficients of the entire spline as $\boldsymbol{z} = [\boldsymbol{z}_0, \boldsymbol{z}_1, \dots, \boldsymbol{z}_{M-1}]^T$.



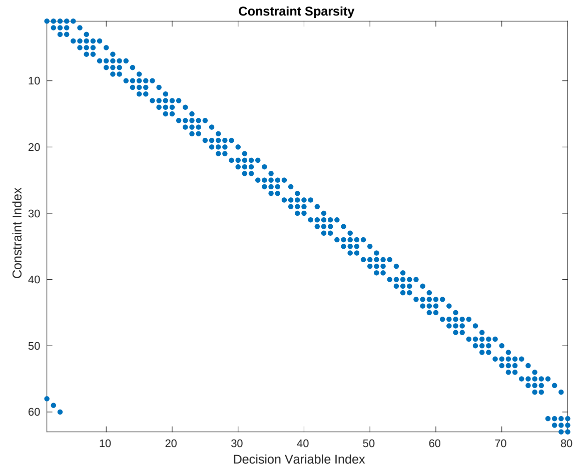


Figure 2: The sparsity pattern for the spline shown in Figure 1. It has 20 cubic segments and thus 4*20 = 80 decision variables. Notice that the quadratic cost matrix is perfectly block diagonal. The first 3*(20-1) rows of the constraint matrix are perfectly block diagonal, representing the continuity constraints. The final six rows are different, representing the boundary constraints.

2.5 Spline Continuity

The spline coefficients are constructed such that value, slope, and curvature are continuous. This can be written as a set of three equations at each knot point, where $h_j = T_{j+1} - T_j$ is the duration of segment j.

$$\boldsymbol{x}_j(h_j) - \boldsymbol{x}_{j+1}(0) = \boldsymbol{0} \tag{13}$$

$$\dot{\boldsymbol{x}}_j(h_j) - \dot{\boldsymbol{x}}_{j+1}(0) = \boldsymbol{0} \tag{14}$$

$$\ddot{\boldsymbol{x}}_j(h_j) - \ddot{\boldsymbol{x}}_{j+1}(0) = \boldsymbol{0} \tag{15}$$

These equations form the blocks on the diagonals of the $\mathcal{A}$ matrix. Each set of three constraints at a knot point populates two blocks of the constraint matrix, one for the lower segment and one for the upper segment.

$$\begin{bmatrix} \mathcal{A}_{k,j} & \mathcal{A}_{k,j+1} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{z_j} \\ \boldsymbol{z_{j+1}} \end{bmatrix} = \boldsymbol{b}_k \tag{16}$$

$$\mathcal{A}_{k,j} = \begin{bmatrix} 1 & h_j & h_j^2 & h_j^3 \\ 0 & 1 & 2h_j & 3h_j^2 \\ 0 & 0 & 2 & 6h_j \end{bmatrix} \qquad \mathcal{A}_{k,j+1} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -2 & 0 \end{bmatrix} \tag{17}$$

$$\boldsymbol{z}_j = \begin{bmatrix} \boldsymbol{D}_j \\ \boldsymbol{C}_j \\ \boldsymbol{B}_j \\ \boldsymbol{A}_j \end{bmatrix} \qquad \boldsymbol{z}_{j+1} = \begin{bmatrix} \boldsymbol{D}_{j+1} \\ \boldsymbol{C}_{j+1} \\ \boldsymbol{B}_{j+1} \\ \boldsymbol{A}_{j+1} \end{bmatrix} \qquad \boldsymbol{b}_k = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix} \tag{18}$$

2.6 Boundary Constraints

The boundary constraints are similar to the continuity equations, but they use only a single block in the $\mathcal{A}$ matrix and a non-zero block in the $\boldsymbol{b}$ matrix. To keep notation simple through this section, we will define $L = M - 1$ to be the index of the final segment of the spline.

$$\boldsymbol{x}_0(0) = \boldsymbol{x}_0 \qquad \boldsymbol{x}_L(h_L) = \boldsymbol{x}_T \tag{19}$$

$$\dot{\boldsymbol{x}}_0(0) = \dot{\boldsymbol{x}}_0 \qquad \dot{\boldsymbol{x}}_L(h_L) = \dot{\boldsymbol{x}}_T \tag{20}$$

$$\ddot{\boldsymbol{x}}_0(0) = \ddot{\boldsymbol{x}}_0 \qquad \ddot{\boldsymbol{x}}_L(h_L) = \ddot{\boldsymbol{x}}_T \tag{21}$$

The lower boundary equation can be written:

$$\mathcal{A}_{\ell,0} \cdot \boldsymbol{z}_{\boldsymbol{0}} = \boldsymbol{b}_\ell \tag{22}$$

$$\mathcal{A}_{\ell,0} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \end{bmatrix} \qquad \boldsymbol{z}_0 = \begin{bmatrix} \boldsymbol{D}_0 \\ \boldsymbol{C}_0 \\ \boldsymbol{B}_0 \\ \boldsymbol{A}_0 \end{bmatrix} \qquad \boldsymbol{b}_\ell = \begin{bmatrix} \boldsymbol{x}_0 \\ \dot{\boldsymbol{x}}_0 \\ \ddot{\boldsymbol{x}}_0 \end{bmatrix} \tag{23}$$

The upper boundary equation can be written:

$$\mathcal{A}_{\ell+1,L} \cdot \boldsymbol{z}_{\boldsymbol{L}} = \boldsymbol{b}_{\ell+1} \tag{24}$$

$$\mathcal{A}_{\ell+1,L} = \begin{bmatrix} 1 & h_L & h_L^2 & h_L^3 \\ 0 & 1 & 2h_L & 3h_L^2 \\ 0 & 0 & 2 & 6h_L \end{bmatrix} \qquad \boldsymbol{z}_L = \begin{bmatrix} \boldsymbol{D}_L \\ \boldsymbol{C}_L \\ \boldsymbol{B}_L \\ \boldsymbol{A}_L \end{bmatrix} \qquad \boldsymbol{b}_{\ell+1} = \begin{bmatrix} \boldsymbol{x}_T \\ \dot{\boldsymbol{x}}_T \\ \ddot{\boldsymbol{x}}_T \end{bmatrix} \tag{25}$$

2.7 Objective Function

Unlike the constraint matrices, the objective function matrices are made of up a sum of many components. A single segment will have a term from the minimum-jerk component, as well as a term for each point in the data set that is on the time-domain of that segment. Thus, we can rewrite the objective function (1) as sum over segments:

$$\boldsymbol{J} = \sum_{j=0}^{M-1} \boldsymbol{J}_j \tag{26}$$

Similarly, the cost of a single segment can be constructed as follows, where $\mathcal{I}$ is the set of points such that $t_i \in [T_j, T_{j+1})$.

$$\boldsymbol{J}_j = \alpha J_j^{\text{smooth}} + \frac{T}{N} \sum_{i \in I_j} J_j^{\text{i}} \tag{27}$$

In practice, the segment cost is implemented by summing blocks of the $\mathcal{H}$ and $\boldsymbol{f}$ matrices. These blocks are thus computed:

$$\boldsymbol{H}_j = \alpha \boldsymbol{H}_j^{\text{smooth}} + \frac{T}{N} \sum_{i \in I_j} \boldsymbol{H}_j^{\text{i}} \tag{28}$$

$$\boldsymbol{f}_j = \alpha \boldsymbol{f}_j^{\text{smooth}} + \frac{T}{N} \sum_{i \in I_j} \boldsymbol{f}_j^{\text{i}} \tag{29}$$

2.8 Data-Fitting

The data-fitting term in the objective function is given by:

$$\frac{T}{N} \sum_{i=0}^{N} \big(\boldsymbol{x}(t_i) - \bar{\boldsymbol{x}}_i\big)^2 \tag{30}$$

We will construct the equations for a single point $\{t_i, \bar{\boldsymbol{x}}_i\}$ from the data set, where $t_i \in [T_j, T_{j+1})$. In other words, the point is on segment j. The squared-error between the point and the segment is thus given by:

$$J_j^i = \Big(\boldsymbol{x}_j(t_i - T_j) - \bar{\boldsymbol{x}}_i\Big)^2 \tag{31}$$

After doing a bit of algebra, the block matrices for this equation are given below, where $\tau_{ij} = t_i - T_j$ is the time between the data point i and segment j. Note that $\mathcal{H}_j^i$ is symmetric. These matrices can them be computed for each point in the data set.

$$\mathcal{H}_j^i = \begin{bmatrix} 1 & \tau_{ij} & \tau_{ij}^2 & \tau_{ij}^3 \\ \tau_{ij} & \tau_{ij}^2 & \tau_{ij}^3 & \tau_{ij}^4 \\ \tau_{ij}^2 & \tau_{ij}^3 & \tau_{ij}^4 & \tau_{ij}^5 \\ \tau_{ij}^3 & \tau_{ij}^4 & \tau_{ij}^5 & \tau_{ij}^6 \end{bmatrix} \qquad \boldsymbol{f}_j^i \begin{bmatrix} -\bar{\boldsymbol{x}}_i \\ -\bar{\boldsymbol{x}}_i\,\tau_{ij} \\ -\bar{\boldsymbol{x}}_i\,\tau_{ij}^2 \\ -\bar{\boldsymbol{x}}_i\,\tau_{ij}^3 \end{bmatrix}^T \tag{32}$$

2.9 Smoothing Objective

We enforce a smooth spline by adding a term to minimize the integral of the curvature-rate-squared. This is simple to compute for a cubic spline:

$$\dddot{\boldsymbol{x}}_j^2(t) = 36\boldsymbol{A}_j^2 \tag{33}$$

The corresponding blocks in the $\mathcal{H}$ and $\mathbf{f}$ matrices are:

$$\mathcal{H}_j^{\text{smooth}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 36 \end{bmatrix} \qquad \boldsymbol{f}_j^{\text{smooth}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^T \tag{34}$$