ARTICLE IN PRESS

Engineering Science and Technology, an International Journal xxx (2016) xxx–xxx

HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

Full Length Article

Recursive B-spline approximation using the Kalman filter

Jens Jauch *, Felix Bleimund, Stephan Rhode, Frank Gauterin

Institute of Vehicle System Technology, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany

ARTICLE INFO

Article history:
Received 14 April 2016
Revised 15 September 2016
Accepted 20 September 2016
Available online xxxx

Keywords:
Recursive
B-spline
Approximation
Kalman filter
Shift
Online

ABSTRACT

This paper proposes a novel recursive B-spline approximation (RBA) algorithm which approximates an unbounded number of data points with a B-spline function and achieves lower computational effort compared with previous algorithms. Conventional recursive algorithms based on the Kalman filter (KF) restrict the approximation to a bounded and predefined interval. Conversely RBA includes a novel shift operation that enables to shift estimated B-spline coefficients in the state vector of a KF. This allows to adapt the interval in which the B-spline function can approximate data points during run-time.

1. Introduction

A B-spline function is a piecewise defined polynomial function with several beneficial properties such as numerical stability of computations, local effects of coefficient changes and built-in smoothness between neighboring polynomial pieces [2, chap. 1]. A common application of B-spline functions, curves and surfaces is fitting of data points. Fitting can either be interpolation or approximation. An interpolating B-spline function passes through the data points, whereas an approximating B-spline function minimizes the residuals between the function and the data but does not pass through the data points in general. The representation using B-splines is popular in computer-aided design, modeling and engineering as well as computer graphics for the geometry of curves, objects and surfaces [3]. It is also used for planning trajectories of computer controlled industrial machines [4] and robots [5,6].

Fitting B-spline functions can be determined by the weighted least squares (WLS) method. It is often used in offline applications, where all data points are available at once.

The Kalman filter (KF) is an established method for estimating the state of a dynamic system. Applications include tracking, navigation, sensor data fusion and process control [7, pp. 4f.]. The KF can be seen as a generalization of the recursive least squares (RLS) method [8, p. 129]. RLS can compute an approximating B-spline function recursively meaning that the approximation is updated with each new data point. This is desired in online applications, in which data points are observed one after another.

1.1. Problem statement

The value of a B-spline function is given by the sum of basis functions (B-splines) weighted with their corresponding coefficients. Each B-spline is only nonzero within a certain bounded interval which causes that the definition range of a B-spline is bounded as well. If the magnitude of the data points is not exactly known or changes over time, data points can be outside the definition range. Such data points cannot be taken into account. Thereby the problem arises that the approximation might not reflect the data anymore.

Publications concerning the recursive data approximation with a B-spline function have not addressed this issue but have assumed a constant definition range. For example, the approaches based on the KF in [9,10] require that the KF state vector contains all coefficients that are estimated during the whole approximation procedure. Therefore the number of coefficients has to be bounded and specified in advance. As a result, these algorithms can only approximate data points that are within the bounded definition range determined at the beginning.

1.2. Contribution

We propose a novel B-spline approximation (RBA) algorithm that solves the approximation problem iteratively using a KF.

* Corresponding author.
E-mail addresses: jens.jauch@kit.edu (J. Jauch), felix.bleimund@kit.edu (F. Bleimund), stephan.rhode@kit.edu (S. Rhode), frank.gauterin@kit.edu (F. Gauterin).

☆ The associated MATLAB source code can be downloaded from [1].

http://dx.doi.org/10.1016/j.jestch.2016.09.015
2215-0986/© 2016 Karabuk University. Publishing services by Elsevier B.V.

RBA overcomes the current limitation of recursive algorithms based on the KF concerning the fixed approximation interval. The main contribution is to use the time update of the KF for a shift of estimated B-spline coefficients in the KF state vector in combination with a shift in the B-spline knot vector. The shift operation enables to shift the definition range such that it is always possible to take into account the latest data point for the approximation.

In online and offline applications, the shift operation allows to reduce the size of the state vector. As smaller state vector causes less computational effort. Table 1 displays the relevant features of different B-spline approximation methods.

1.3. Fitting algorithms for B-spline functions

Fitting B-spline functions can be computed by least squares (LS) methods [2,11,12]. With the standard formula in batch form, all data points have to be collected and then processed simultaneously. Therefore the number of data points n needs to be bounded. The computation usually involves a Cholesky or QR factorization and requires $\mathcal{O}(n)$ operations if one takes advantage of the banded matrix structure [13, pp. 327–331]. Such algorithms are stated in [13, pp. 117–121] and [14, pp. 152–160]. With the LS algorithm each data point influences the result to the same extend. The WLS algorithm allows to weight measurements relative to each other [2, pp. 119–123].

In online applications an ever-growing amount of data is common. LS algorithms for online applications can be subdivided into two groups: First, growing memory LS algorithms apply a weighting that forgets old data exponentially. Second, sliding window LS algorithms discard old data completely and require only finite storage [15]. Sliding window LS and sliding window WLS algorithms are proposed in [15–18], respectively. Re-computing the fitting function from scratch with each new data point is costly. Rank update and rank downdate methods allow to re-use an already known factorization for an efficient update of a solution when observations have been added or deleted [19–21].

With WLS the bounded definition range of B-spline functions does not present a problem because the number and position of B-splines can be changed if the fitting function is re-computed from scratch. Moreover, rank modification methods support adding or deleting matrix columns [20]. This allows to extend, shrink or shift the definition range of the B-spline function.

Recursive algorithms such as RLS (see [8, pp. 84–88]) usually require less computational power than batch algorithms because they use smaller matrices and vectors whose sizes do not depend on the number of data points. The recursive computation is also referred to as progressive, iterative or sequential. In [22] fitting B-spline curves and surfaces are iteratively constructed based on the idea of profit and loss modification without solving a linear system. The authors of [23] build on the progressive and iterative approximation technique for B-spline curve and surface fitting and prove that the proposed algorithm achieves a least squares fit to the data points. A recursive algorithm for optimal smoothing B-spline surfaces inspired by the RLS method is presented in [24]. Algorithms that involve a KF are stated in [9,10]. All recursive approaches mentioned assume a constant definition range.

1.4. Structure of the data set

$\{(s_t, \boldsymbol{y}_t)\}_{t=1,2,\dots,n}$ is a set of n data points. t denotes the time step at which data point $(s_t, \boldsymbol{y}_t)$ has been measured or observed. s_t is the value of the independent variable s at time step t. $\boldsymbol{y}_t = (y_{t,1}, y_{t,2}, \dots, y_{t,v}, \dots, y_{t,V_t})^\top$ is a vector of V_t measurements y that refer to s_t and may come from different sensors. $^\top$ denotes the transpose operation. $V_t \in \mathbb{N}$ is allowed to be different for each $\boldsymbol{y}_t$. We assume that $V_t \ll n \forall t$. The vector of all measurements $\boldsymbol{y}$ is composed as follows:

$$\boldsymbol{y}^\top = (\underbrace{y_{1,1}, \dots, y_{1,V_1}}_{=:\boldsymbol{y}_1^\top}, \dots, \boldsymbol{y}_t^\top, \dots, \underbrace{y_{n,1}, \dots, y_{n,V_n}}_{=:\boldsymbol{y}_n^\top}) \tag{1}$$

1.5. Outline

The remainder of this article is structured as follows: In Section 2.1 we introduce a B-spline function definition in matrix form. Section 2.2 describes the WLS approach followed by the KF algorithm in Section 2.3. Section 2.4 presents the novel RBA algorithm. Its effectiveness is demonstrated in comparison with the WLS solution by numerical examples in Section 3. We summarize the characteristics of RBA and draw our conclusions in Section 4.

2. Methods

2.1. B-spline functions

A B-spline function is a piecewise defined function. Its value is given by the weighted sum of J polynomial basis functions (B-splines) of degree d. The knot vector is $\boldsymbol{\kappa} = (\kappa_1, \kappa_2, \dots, \kappa_{J+d+1})$. We assume strictly increasing knot values ($\kappa_k < \kappa_{k+1}$, $k = 1, 2, \dots, J+d$). $\boldsymbol{\kappa}$ and d determine the number and shape of B-splines. The j-th B-spline $b_j(s), j = 1, 2, \dots, J$ is positive only for $s \in (\kappa_j, \kappa_{j+d+1})$ and zero elsewhere [2, pp. 37–42].

The following definitions originate from [2, pp. 47–50 & 65–70]: Let $[\kappa_\mu, \kappa_{\mu+1})$ be a spline interval and let μ denote the spline interval index with $d+1 \leqslant \mu \leqslant J$. For $s \in [\kappa_\mu, \kappa_{\mu+1})$, the B-splines $b_j(s), j = \mu - d, \dots, \mu$ can be nonzero. Their values for a specific $s \in [\kappa_\mu, \kappa_{\mu+1})$ can be summarized in the B-spline vector $\boldsymbol{b}_{\mu,d}(s) = (b_{\mu-d}(s), b_{\mu-d+1}(s), \dots, b_\mu(s)) \in \mathbb{R}^{1\times(d+1)}$ which can be computed according to (2):

$$\boldsymbol{b}_{\mu,d}(s) = \underbrace{\boldsymbol{B}_{\mu,1}(s)}_{\in\mathbb{R}^{1\times 2}} \underbrace{\boldsymbol{B}_{\mu,2}(s)}_{\in\mathbb{R}^{2\times 3}} \dots \underbrace{\boldsymbol{B}_{\mu,\delta}(s)}_{\in\mathbb{R}^{\delta\times(\delta+1)}} \dots \underbrace{\boldsymbol{B}_{\mu,d}(s)}_{\in\mathbb{R}^{d\times(d+1)}} \tag{2}$$

The B-spline matrix $\boldsymbol{B}_{\mu,\delta}(s) \in \mathbb{R}^{\delta\times(\delta+1)}$ is defined for each $\delta \in \mathbb{N}$ with $\delta \leqslant d$ and given by

$$\boldsymbol{B}_{\mu,\delta}(s) = \begin{bmatrix} \frac{\kappa_{\mu+1}-s}{\kappa_{\mu-1}-\kappa_{\mu+1-\delta}} & \frac{s-\kappa_{\mu+1-\delta}}{\kappa_{\mu+1}-\kappa_{\mu+1-\delta}} & 0 & \dots & 0 \\ 0 & \frac{\kappa_{\mu+2}-s}{\kappa_{\mu-2}-\kappa_{\mu+2-\delta}} & \frac{s-\kappa_{\mu+2-\delta}}{\kappa_{\mu+2}-\kappa_{\mu+2-\delta}} & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \dots \\ 0 & 0 & \dots & \frac{\kappa_{\mu+\delta}-s}{\kappa_{\mu+\delta}-\kappa_\mu} & \frac{s-\kappa_\mu}{\kappa_{\mu+\delta}-\kappa_\mu} \end{bmatrix}. \tag{3}$$

Table 1
Comparison of different B-spline approximation methods.

Feature	WLS (single call)	WLS (multiple calls)	RLS/KF	RBA
Number of processable data points n	bounded	unbounded	unbounded	unbounded
Time complexity	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Approximation interval	fixed	variable	fixed	variable
Determination of total number of coefficients being estimated	at beginning	during run-time	at beginning	during run-time

Please cite this article in press as: J. Jauch et al., Recursive B-spline approximation using the Kalman filter, Eng. Sci. Tech., Int. J. (2016), http://dx.doi.org/10.1016/j.jestch.2016.09.015