Least squares B-spline approximation with applications to geospatial point clouds

Alireza Amiri-Simkooei, Fatemeh Esmaeili, Roderik Lindenbergh

This readme file explains how the least-squares B-spline approximation (LSBSA) can be applied to 1D curve fitting, 2D surface, and 3D manifold problems. For this purpose, the three simulated examples will be explained for which the Python and Matlab codes were also presented.

1. Curve fitting problem

Step 1: Simulate m number of observations $y = v = f(u)$.

For the curve fitting problem, we introduce $m = 503$ points along the u axis at $u_i = 0, 0.05, 0.10, \dots, 25.1$, $i = 1, \dots, 503$ (here we use equally spaced data, but this is not a requirement for LSBSA). The values are indeed between 0 and 8π with sampling interval of 0.05. For each of the data points u_i, the known function values are obtained from $v_i = f(u_i) = \sin(u_i) + \operatorname{sinc}(u_i)$. A normally distributed noise with a standard deviation of $\sigma = 0.05$ is then added to the v values. This will make the final observations $y = v + noise$, to be approximated by LSBSA. The data points are then $(u_i, y_i), i = 1, \dots, m$.

Step 2: Specify the knots along the u-axis (locations where the pieces meet each other).

LSBSA needs the user-defined knots along the u-axis. In this example we introduced 14 knots along $0 \leq u \leq 8\pi$ axis with intervals of 2, making in total 14 knouts as: $knotu = [0, 2, 4, \dots, 24, 25.1]$. Note the last knot was added to guarantee that all observation points are located in the knots' definition domain. The number of cells in this example is $14 - 1 = 13$. Note also that the knots do not need to be equally spaced, so for example $knotu = [-1, 3.1, 5.8, \dots, 23.5, 26.0]$ could also be an option. The interval between knots is a tuning parameter, and can be specified by the user for the best performance.

Step 3: Specify the B-spline's degree (linear, quadratic, cubic, quartic, etc).

The degree of the B-spline polynomials is specified here. For example, we may use linear, quadratic, cubic, or quartic B-splines leading to $degu = 1, 2, 3, \text{or } 4$. Other degrees can accordingly be used. We may also tune this variable to selecting an optimal value for the $degu$.

Step 4: Construct the A design matrix for the given data points based on B-splines cells and order.

This step establishes the $m \times n$ design matrix A of the linear model $y = Ax + e$, where m is the number observations and n is the number of B-spline coefficients. M is the length of vector u, and n is the number of knots $knotu$ plus the degree $degu$ minus 1, i.e. $n = \operatorname{len}(knotx) + degx - 1$. For the example considered we have $m = 503$ and for the four cases we deal with $n = 14$ (linear), $n = 15$ (quadratic), $n = 16$ (cubic), and $n = 17$ (quartic).

Step 5: Apply the least squares method to estimate B-spline coefficients.

The coefficients of B-splines are then estimated using the least squares method. In the example considered we used the covariance matrix as $Q_y = I_m$, where I_m is an identity matrix of size m, so we have $\hat{x} =$

$(A^T A)^{-1} A^T y$. The estimates for observations are $\hat{y} = A\hat{x}$ and for the residuals are $\hat{e} = y - A\hat{x}$. Having the residuals estimated, the unknown variance component σ^2 is estimated by the least squares method as

$$\hat{\sigma}^2 = \frac{\hat{e}^T \hat{e}}{m - n}$$

The standard deviation $\hat{\sigma}$ explains the quality of the fit. It captures both the approximation error (for instance, caused by coarse knots or the use of low-degree B-splines like linear ones) and the measurement errors (such as simulated noise).

Step 6: Predict v (y) values for u positions specified by user using the estimated coefficients.

After estimation of the B-spline coefficients, it is possible to predict the function value at any given point. We may then define a new set of u values, u_p's, to predict the v values, , $v_p = f(u_p)$. For the example considered we have used the $u_p = 0, 0.1, 0.2, \dots, 25.1$, resulting in total $m_p = 252$ regularly spaced value. To compute the $f(u_p)$ values, we first need to compute the A_p matrix, from

$$A_p \;=\; \texttt{bs.A\_matrix\_1D}(u_p, knotu, degu)$$

The predicted values $y_p = v_p$ can be obtained as $y_p = A_p\hat{x}$.

Step 7: Apply the quality control measures and adjust the tuning parameters.

The quality control measures and tuning of the parameters are important step to obtain reliable results. For example we can apply hypothesis testing to identify outlying observations, or we can investigate whether the estimated $\hat{\sigma}$ is sufficiently small for the approximation problem. If not, we may adjust the $knotu$ and $degu$ to appropriate values and repeat the procedure.

Step 8: Present the approximation results.

For presenting the results we can may present the original data points, the least squares residuals, the estimated observations, or the predicted values. In this curve fitting problem, the predicted values follow the estimated values because u_p is a subset of u, i.e. $u_p \in u$. This can be different for surface and manifold fitting problems (see below).

2. Surface fitting problem

Step1: Simulate m number of observations z=(u, v) .

For the surface fitting problem, we Generate $m = 20000$ random positions $(u_i, v_i), i = 1{:}\,20000$. The values are $-2 \le u, v \le 2$ and $u_{min} = -2, u_{max} = 2$ and $v_{min} = -2, v_{max} = 2$ were fixed boundaries along u, v axes. The known surface function values are obtained from $z_i = u_i \exp(-{u_i}^2 - {v_i}^2)$. The data points are then $(u_i, v_i, z_i), i = 1{:}\,20000$. The $y_i = z_i, i = 1, \dots, m$ are then the observations to be approximated by LSBSA.

Step 2: Specify the knots along u, v axes.

2D LSBSA needs the user-defined knots along u, v axes. In this example, we introduced 11 knots along $-2 \le u \le 2$ axis with intervals of 0.4 and 11 knots along $-2 \le v \le 2$ axis with intervals of 0.4 ($knotu$, $knotv$), making total number of 121 cells. Note that it is not necessary for the knots to have equal intervals along u or v axes, so for example 0.4×0.6 could be an option for the cells size. Last knots along u, v axes was added to guarantee that all observation points are located in the knots' definition domain. The interval between knots is a tuning parameter, and can be specified by the user for the best performance.

Step 3: Specify the B-spline's degree along u and v axes (linear, quadratic, cubic, quartic, etc.).

The degree of the B-spline polynomials along each u and v axis is specified here. For example, we may use linear, quadratic, cubic, or quartic B-spline along axes leading to $degu = 1, 2, 3, 4$, $degv = 1, 2, 3, 4$. Other degrees can accordingly be used. We may also tune this variable to selecting an optimal value for the $degu, degv$. Note that the degree of the B-spline along each axis can be different from the other axis.

Step 4: Construct the A design matrix for the given data points based on B-splines cells and order.

This step establishes the $m \times n$ design matrix A of the linear model $y = Ax + e$, where m is the number observations and n is the number of B-spline coefficients. m is the length of vector u and v, and $n = n_u \times n_v$, where: $n_u = \text{len}(knotu) + degu - 1$ and $n_v = \text{len}(knotv) + degv - 1$. For the example considered we have $m = 20000$. Number of 11 knots along each u, v axis and cubic order for B-splines make a total number of $(11 + 2) \times (11 + 2) = 169$ unknown coefficients.

Step 5: Apply the least squares method to estimate B-spline coefficients.

The coefficients of B-splines are then estimated using the least squares method. In the example considered we used the covariance matrix as $Q_y = I_m$, where I_m is an identity matrix of size m, so we have $\hat{x} = (A^T A)^{-1} A^T y$. The estimated observations are $\hat{y} = A\hat{x}$ and for the residuals are $\hat{e} = y - A\hat{x}$. Having the residuals estimated, the unknown variance component σ^2 is estimated by the least squares method as

$$\hat{\sigma}^2 = \frac{\hat{e}^T \hat{e}}{m - n}$$

The standard deviation $\hat{\sigma}$ explains the quality of the fit. It captures both the approximation error (for instance, caused by coarse knots or the use of low-degree B-splines like linear ones) and the measurement errors (such as simulated noise). Note that here we did no introduce simulated noise to this example.

Step 6: Predict $z = f(u, v)$ values for u, v positions specified by user using the estimated coefficients.

After estimation of the B-spline coefficients, it is possible to predict the function value at any given point. We may then define a new set of u and v values, u_p's, v_p's, to predict the z values, $z_p = f(u_p, v_p)$. For the example considered we have used a regular grid $u_p = -2{:}\,0.05{:}\,2$ and $v_p = -2{:}\,0.05{:}\,2$ of 6561 points. Note that to use the LSBSA, the grid points and their coordinate components must be in vector format. We first need to compute the matrix A_p from

$$A_p = \texttt{bs.A\_matrix\_2D}(u_p, v_p, knotu, knotv, degu, degv)$$

The predicted values $y_p = z_p$ can be obtained as $y_p = A_p \hat{x}$.

Step 7: Apply the quality control measures and adjust the tuning parameters.

The quality control measures and tuning of the parameters are important step to obtain reliable results. For example, we can apply hypothesis testing to identify outlying observations, or we can investigate whether the estimated $\hat{\sigma}$ is sufficiently small for the approximation problem. If not, we may adjust the $knotu, knotv$ and $degu$, $degv$ to appropriate values and repeat the procedure.

Step 8: Present the approximation results.

For presenting the results we may present the original data points, the least squares residuals, the estimated observations, or the predicted values of a regular grid points. Here we present predicted values for the regular $u_p = -2{:}\,0.05{:}\,2$ and $v_p = -2{:}\,0.05{:}\,2$ grid points to display a 3D surface. Note that to display the estimated 3D surface, the vector format of the grid points and their y_p values must be converted to the mesh grid format.

3. Manifold fitting problem

Step1: Simulate m number of observations z=(u, v, t) .

For the manifold fitting problem, we generate $m = 10000$ random 3D space-time locations $(u_i, v_i, t_i), i = 1{:}\,10000$. The points are generated over the cube $-2 \leq u, v \leq 2$ and $0 \leq t \leq 4$ along u, v and t axes. The known surface function values are obtained from $z_i = (1 + 0.05\, t_i)u_i \exp(-{u_i}^2 - {v_i}^2) + 0.02\, u_i v_i t_i$. The data points are then $(u_i, v_i, t_i, z_i), n = 1, \ldots, m$, so 4D point cloud data to be approximated by 3D LSBSA manifold fitting ($z_i = y_i = f(u_i, v_i, t_i)$).

Step 2: Specify the knots along u, v, t axes.

LSBSA needs the user-defined knots along the along u, v, t axis. In this example we introduced 9 knots along each $-2 \leq u, v \leq 2$ axis with 0.5 intervals ($knotu$, $knotv$), and time intervals of 0.5 for $knott$ in time between $0 \leq t \leq 4$ making in total 729 3D manifold cells. Note that it is not necessary for the knots to have equal intervals along the u, v axes, so for example $0.4 \times 0.6 \times 0.5$ could be an also option for the cells size. Note the last knots along u, v,t axis was added to guarantee that all observation points are located in the knots' definition domain. The interval between knots is a tuning parameter, and can be specified by the user for the best performance.

Step 3: Specify the B-spline's degree along u ,v and t axes (linear, quadratic, cubic, quartic, etc.).

The degree of the B-spline polynomials along each u,v, t axes is specified here. For example, we may use linear, quadratic, cubic, or quartic B-spline along axes leading to $degu = 1, 2, 3, 4$, $degv = 1, 2, 3, 4$ and $degt = 1, 2, 3, 4$. Other degrees can accordingly be used. We may also tune this variable to selecting an optimal value for the $degu, degv, degt$. Note that the degree of the B-spline along each axis can be different from the other axes.

Step 4: Construct the A design matrix for the given data points based on B-splines cells and order.

This step establishes the $m \times n$ design matrix A of the linear model $y = Ax + e$, where m is the number observations and n is the number of B-spline coefficients. m is the length of vector u, v and t, and

$$n = n_u \times n_v \times n_t$$

where: $n_u = \text{len}(knotu) + degu - 1$, $n_v = \text{len}(knotv) + degv - 1$, and $n_t = \text{len}(knott) + degt - 1$. For the example considered we have $m = 10000$. Number of 9 knots along each u, v and t axis and the cubic-degree B-splines make a total number of $n = (9+2) \times (9+2) \times (9+2) = 1331$ unknown coefficients to be estimated.

Step 5: Apply the least squares method to estimate B-spline coefficients.

The coefficients of B-splines are then estimated using the least squares method. In the example considered we used the covariance matrix as $Q_y = I_m$, where I_m is an identity matrix of size m, so we have $\hat{x} = (A^T A)^{-1} A^T y$. The estimated observations are $\hat{y} = A\hat{x}$ and for the residuals are $\hat{e} = y - A\hat{x}$. Having the residuals estimated, the unknown variance component σ^2 is estimated by the least squares method as

$$\hat{\sigma}^2 = \frac{\hat{e}^T \hat{e}}{m-n}$$

The standard deviation $\hat{\sigma}$ explains the quality of the fit. It captures both the approximation error (for instance, caused by coarse knots or the use of low-degree B-splines like linear ones) and the measurement errors (such as simulated noise). Here we did not introduce noise to generated data.

Step 6: Predict $z = f(u, v, t)$ values for u, v positions at epoch t specified by user using the estimated coefficients.

After estimation of the B-spline coefficients, it is not possible to predict the function value at any given point. We may then define a new set of u, v and t values u_p's, v_p's, t_p's to predict the z values, $z_p = f(u_p, v_p, t_p)$. For the example considered we have used a regular grid $u_p = -2{:}\,0.1{:}\,2$ and $v_p = -2{:}\,0.1{:}\,2$ of 1681 points at $t = 1,2,3,4$ time instances. Note that to use the LSBSA, the grid points and their coordinate components must be in vector format. We first need to compute the A_p matrix, from

$$A_p \ = \texttt{bs.A\_matrix\_3D}(u_p, v_p, t_p, knotu, knotv, knott, degu, degv, degt)$$

The predicted values $y_p = z_p$ can be obtained as $y_p = A_p\hat{x}$.

Step 7: Apply the quality control measures and adjust the tuning parameters.

The quality control measures and tuning of the parameters are important step to obtain reliable results. For example, we can apply hypothesis testing to identify outlying observations, or we can investigate whether the estimated $\hat{\sigma}$ is sufficiently small for the approximation problem. If not, we may adjust the $knotu, knotv$, $knott$ and $degu$, $degv, degt$ to appropriate values and repeat the procedure.

Step 8: Present the approximation results.

For presenting the results we can may present the original data points, the least squares residuals, the estimated observations, or the predicted values of the regular grid points at a certain time instance. Here we present predicted values for the $u_p = -2{:}\,0.1{:}\,2$ and $v_p = -2{:}\,0.1{:}\,2$ of regular grid points to display a 3D surface at $t = 1,2,3,4$ frames. Note that to display the estimated 3D surface, the vector format of the grid points and their z values must be converted to the mesh grid format.