

Short notes on B-Spline Interpolation, Approximation and Reconstruction filters

Hasan Ayaz

School of Biomedical Engineering Science and Health Systems,
Drexel University

To implement 3rd order B-Spline interpolation, the basis function $\beta^3(x)$ is used. $\beta^3(x)$ is merely 4 times self-convolution of the $\beta^0(x)$ basis function which is a centered-rectangle around origin. After convolution, $\beta^3(x)$ can be formulated as in table 1.

Table 1¹

$$\beta^3(x) = \begin{cases} 2/3 - |x|^2 + |x|^3 / 2 & 0 \leq |x| < 1 \\ (2 - |x|)^3 / 6 & 1 \leq |x| < 2 \\ 0 & 2 \leq |x| \end{cases}$$

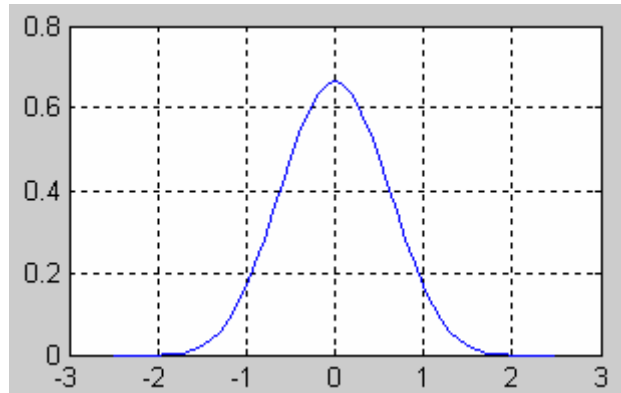


Figure 1. Cubic B-spline

Discrete B-spline kernel b_m^n is obtained by sampling the B-spline of degree n, expanded by a factor m. Discrete B-Spline kernel can be formulated as in eq 1.

$$b_m^n = \beta^n(x/m) \Big|_{x=k} \xleftrightarrow{z} B_m^n(z) = \sum_{k \in \mathbb{Z}} b_m^n(k) z^{-k} \quad \text{eq 1}$$

Now, for interpolation, we want to find the coefficients $c(k)$ for the given sample $s(k)$ so that when we combine the B-splines with coefficients $c(k)$, we have a perfect fit of $s(k)$

¹ Unser M., "Splines, A perfect fit for signal and image processing", *IEEE Signal Processing Magazine*, November 1999, 22-38

$$\sum_{l \in \mathbb{Z}} c(l) \beta^n(x-l) \Big|_{x=k} = s(k) \quad \text{eq 2}$$

Using discrete B-splines, eq 2 can be written as the convolution form.

$$s(k) = b_1^n * c(k) \quad \text{eq 3}$$

So, c(k) can be calculated by eq 4

$$c(k) = (b_1^n)^{-1} * s(k) \quad \text{eq 4}$$

For our case n=3. Thus by sampling cubic B-spline at -1, 0 and 1, we can find the discrete B-spline generating kernel. Sample values are [1/6 2/3 1/6]. So the z-transform of this will lead $(z+4+z^{-1})/6$. To find c(k), we need the inverse of this, that is $6 / (z+4+z^{-1})$

To apply this filter, it is decomposed into two other, one causal and the other anti-causal. After finding c(k) we can go back to our original signal by convolving with [1/6 2/3 1/6].

Also, we can do interpolation through any intermediate value using eq.1. Figure 2 shows the graph where 'o's represent the actual sample point. Any intermediate value can be calculated and plotted, and we can have a continuous output signal.

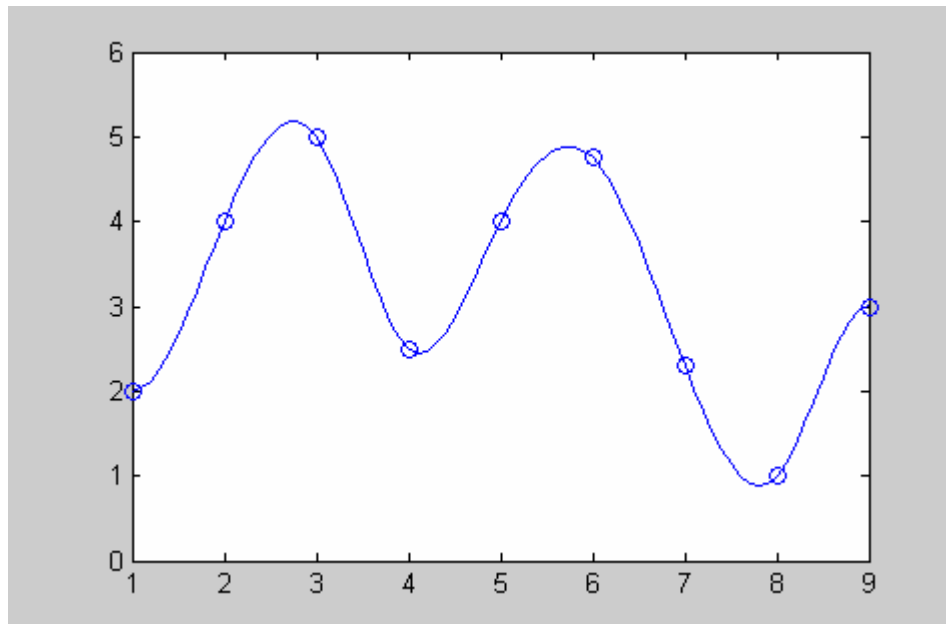


Figure 2. Cubic B-Spline interpolation

Another and faster way of doing the interpolation is to use the method depicted in figure 4. When this method is applied to the above sample set, we find the interpolated middle points as shown in figure 3 where red 'o' represent the original values and blue 'x' shows the interpolated dataset.

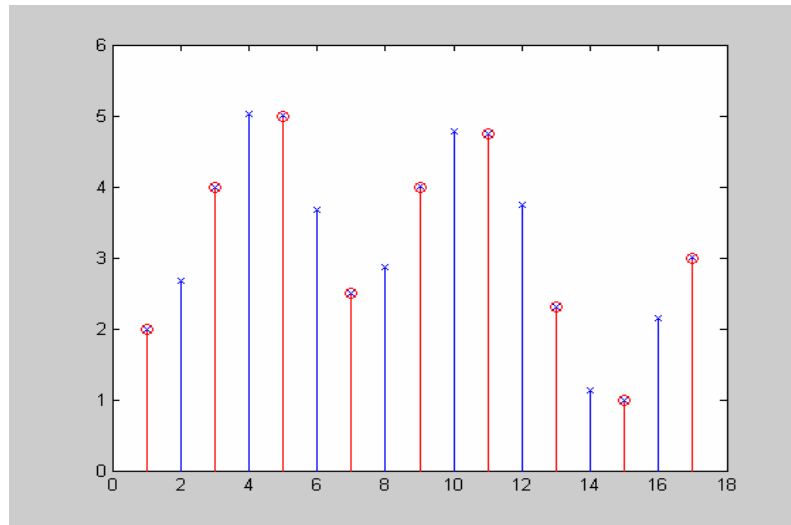


Figure 3

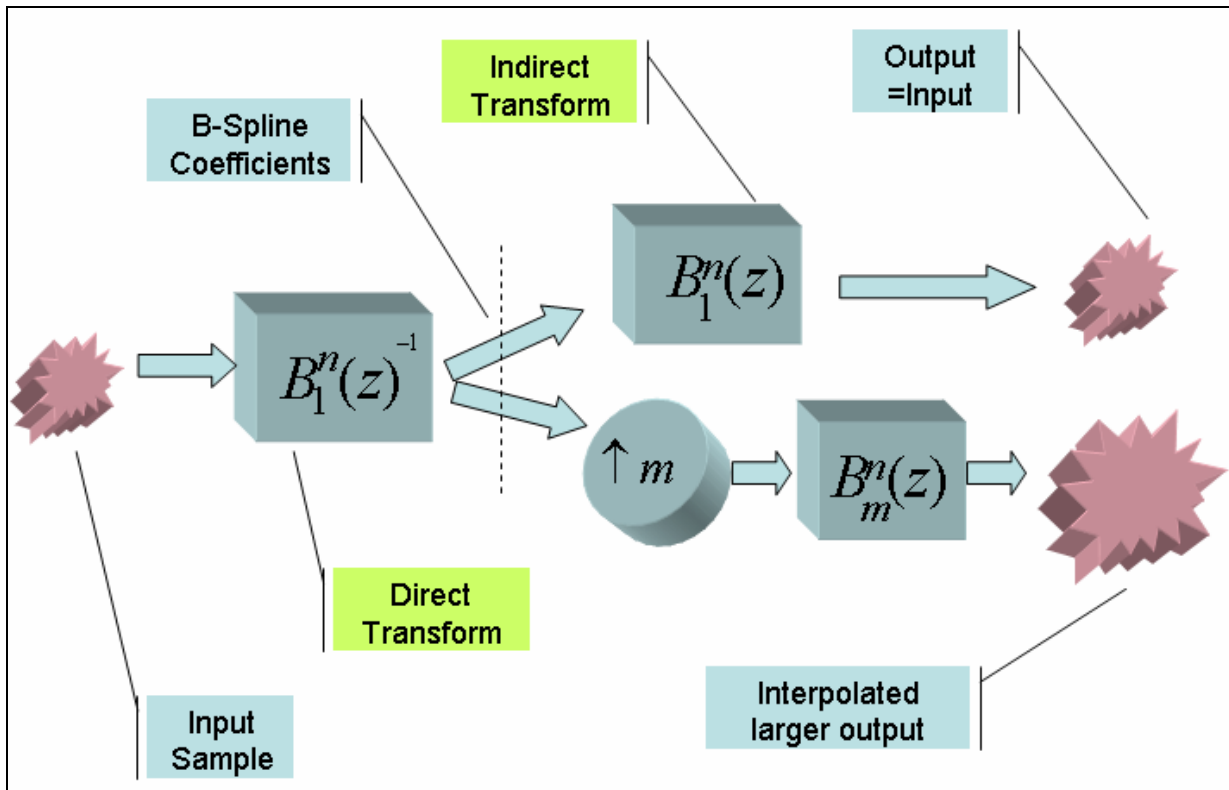


Figure 4

To implement approximation filter, the algorithm described in Unser, Aldroubi and Eden, 1993² is used. The block diagram of the algorithm is shown in figure 5. First

² Unser, M., Aldroubi, A., Eden, M., "B-Spline Signal Processing: Part II – Efficient Design and Applications", *IEEE Transactions on Signal Processing* February 1993, Vol 41. No. 2 p834-848

the signal is convolved with the n^{th} order (in our case 3) B-spline that is sub-sampled with m . Then, decimation with m is done. After decimation a post-filter is applied. This post filter is calculated by minimizing the approximation error and it is formulated as in equation 5.

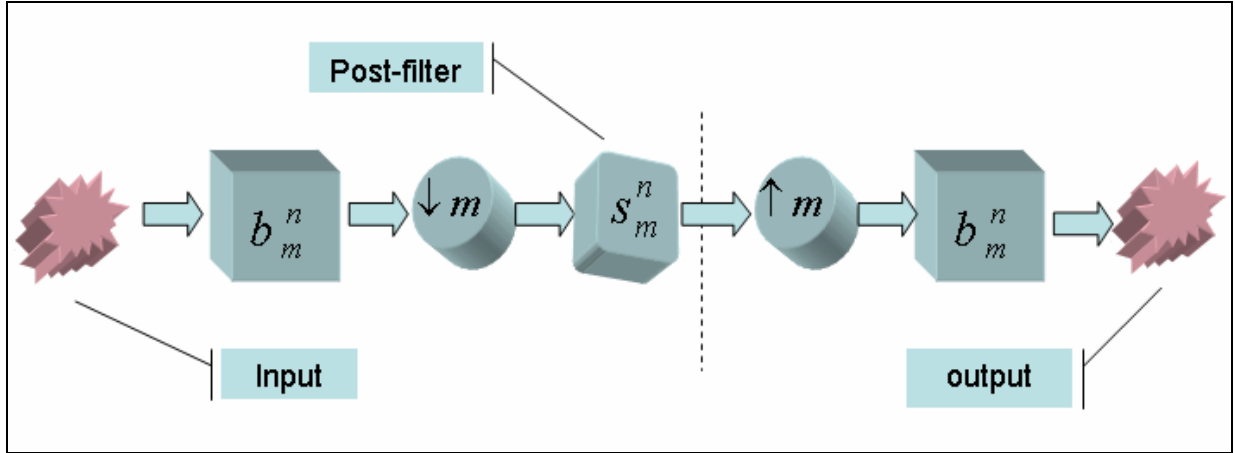


Figure 5. Least Squares polynomial spline approximation & reconstruction of a discrete signal

$$s_m^n(k) = ([b_m^n * b_m^n]_{\downarrow m})^{-1}(k) \quad \text{eq 5}$$

To implement the post filter, we can use iterative equations similar to those in finding the B-spline coefficients. The general form of the equations are described in “Unser, Aldroubi and Eden”, “B-Spline Signal Processing: Part II” in section I-B. For our case $n=3$ and $m=2$. Sampling cubic B-spline of figure 1 give us the following

$b_2^3 = [0.0208 \quad 0.1667 \quad 0.4792 \quad 0.6667 \quad 0.4792 \quad 0.1667 \quad 0.0208]$ where middle is the origin.

Convolving this with itself and then taking one sample from every two, we have

$$([b_2^3 * b_2^3]_{\downarrow 2}) = [0.0004 \quad 0.0477 \quad 0.4718 \quad 0.9601 \quad 0.4718 \quad 0.0477 \quad 0.0004]$$

To invert this, we can take the z-transform of it and then we will have the z-transform of our post-filter. The poles of this z-transform are -0.529604, -0.122309 and -0.0100731. Using these poles and the iterative equations can be formulated to get the post-filter. The results of an arbitrary signal are shown in figures 6 and 7. Red ‘*’s are original values and blue ‘o’'s are approximated values.

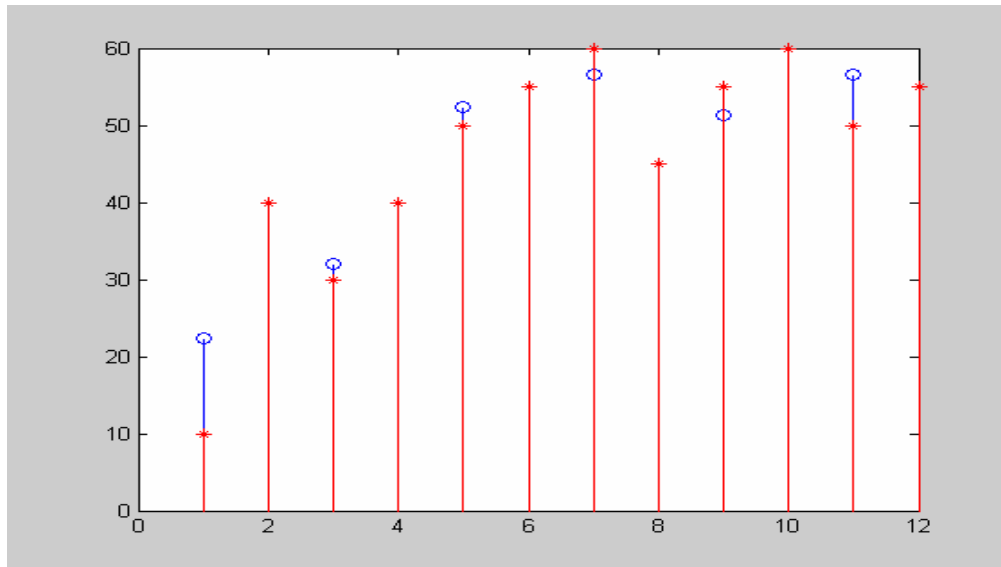


Figure 6

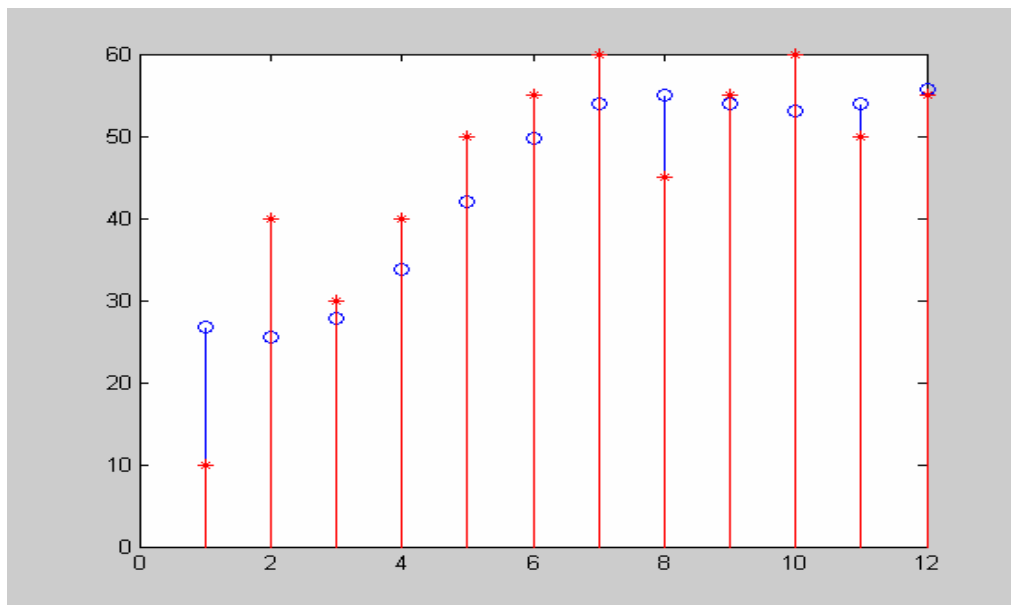


Figure 7

References

- [1] Unser M., "Splines, A perfect fit for signal and image processing", *IEEE Signal Processing Magazine*, November 1999, 22-38
- [2] Unser, M., Aldroubi, A., Eden, M., "B-Spline Signal Processing: Part I – Theory", *IEEE Transactions on Signal Processing February 1993*, Vol 41. No. 2 p821-833
- [3] Unser, M., Aldroubi, A., Eden, M., "B-Spline Signal Processing: Part II – Efficient Design and Applications", *IEEE Transactions on Signal Processing February 1993*, Vol 41. No. 2 p834-848