Stanford Exploration Project, Report 105, September 5, 2000, pages 23–83

Stanford Exploration Project, Report 105, September 5, 2000, pages 23–83

Inverse B-spline interpolation

Sergey Fomel[1]

ABSTRACT

B-splines provide an accurate and efficient method for interpolating regularly spaced data. In this paper, I study the applicability of B-spline interpolation in the context of the inverse interpolation method for regularizing irregular data. Numerical tests show that, in comparison with lower-order linear interpolation, B-splines lead to a faster iterative conversion in under-determined problems and a more accurate result in over-determined problems. In addition, they provide a constructive method for creating discrete regularization operators from continuous differential equations.

INTRODUCTION

The problem of interpolating irregularly sampled data to regular grid (data regularization) can be recast as the inverse process with respect to interpolating regularly sampled data to irregular locations. Claerbout (1999) describes an iterative least-squares optimization approach to data regularization. The optimization is centered around two goals. The first goal is to minimize the power of the residual difference between the observed and predicted data. The second goal is to style the solution according to some predefined regularization criterion.

The ability of inverse interpolation to reach the data fitting goal depends on the accuracy of the forward interpolation operator. Forward interpolation is one of the classic problems in numerical analysis and has been studied extensively by generations of theoreticians and practitioners (Fomel, 1997b). The two simplest and most widely used methods are the nearest neighbor interpolation and linear interpolation. There are several approaches for constructing more accurate (albeit more expensive) linear forward interpolation operators: cubic convolution (Keys, 1981), local Lagrange, tapered sinc (Harlan, 1982), etc. Wolberg (1990) presents a detailed review of different conventional approaches.

Spline interpolation, based on representing the interpolated function by smooth piece-wise polynomials, has been in use for a long time (de Boor, 1978), but only recently Unser et al. (1993a,b) have discovered a way of implementing forward B-spline interpolation with an arbitrary order of accuracy in an efficient signal-processing fashion. The key idea is to implement the B-spline transform with recursive filtering. First, an efficient recursive filtering transforms regularly spaced data into spline coefficients, then the spline coefficients are interpolated onto irregular locations. B-spline interpolants exhibit a superior performance for any given order

[1]**email:** sergey@sep.stanford.edu

of accuracy in comparison with other methods of similar efficiency (Thévenaz et al., 2000).

In this paper, I study the applicability of B-spline interpolation in the context of the inverse interpolation method. In the first section, I review the forward interpolation problem and confirm the observations of Thévenaz et al. (2000) about the superior performance of B-splines. The second section introduces a constructive method of creating discrete regularization operators from B-splines and helical filtering (Claerbout, 1998). The method performance is evaluated with a simple numerical test. In conclusion, I summarize the benefits of using B-splines for data regularization.

FORWARD INTERPOLATION

Forward interpolation plays only a supplementary role in this paper, but it has many applications of its own in the seismic processing practice. It is sufficient to mention such applications as trace resampling, NMO, Kirchoff and Stolt migrations, log-stretch, radial transform, etc. Two simple examples appear at the end of this section.

The general form of a linear forward interpolation operator is

$$f(x) = \sum_{n \in N} W(x,n) f(n) \,, \tag{1}$$

where n is a point on a given regular grid N, x is a point in the continuum, $f(x)$ is the reconstructed continuous function, and $W(x,n)$ is a linear weight. Although in the discussion that follows, I refer to only the one-dimensional theory, a generalization to many dimensions is straightforward.

Nearest neighbor and beyond

The two simplest forms of the forward interpolation operators are the 1-point nearest neighbor interpolation with the weight

$$W(x,n) = \begin{cases} 1, & \text{for} \quad n-1/2 \leq x < n+1/2 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

and the 2-point linear interpolation with the weight

$$W(x,n) = \begin{cases} 1-|x-n|, & \text{for} \quad n-1 \leq x < n+1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Because of their simplicity, the nearest neighbor and linear interpolation methods are very practical and easy to apply. Their accuracy is, however, limited and may be inadequate for interpolating high-frequency signals. The shapes of interpolants (2) and (3) and their spectra are plotted in Figures 1 and 2. The spectra plots show that both interpolants act as low-pass filters, preventing the high-frequency energy from being correctly interpolated.

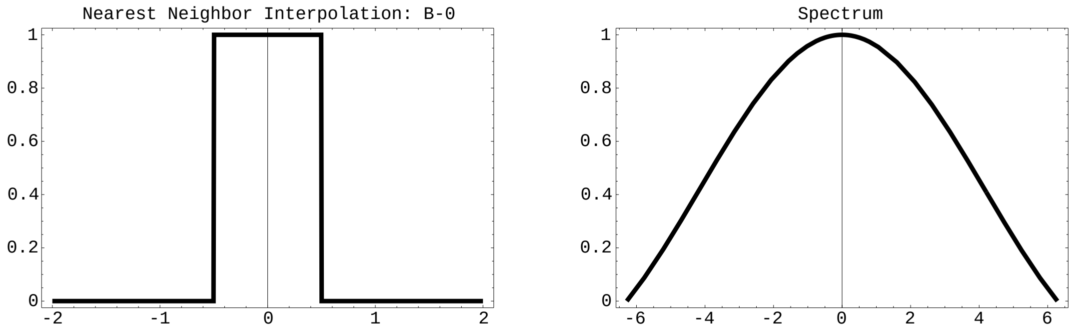


Figure 1: Nearest neighbor interpolant (left) and its spectrum (right). sergey2-nnint [CR]

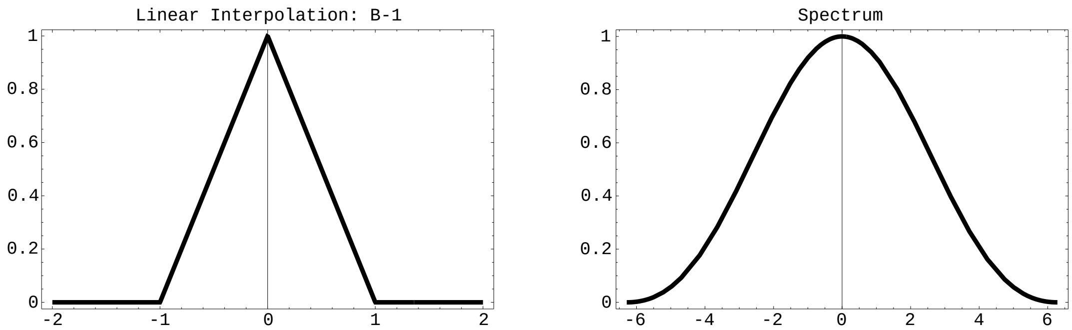


Figure 2: Linear interpolant (left) and its spectrum (right). sergey2-linint [CR]

On the other side of the accuracy scale, there is the infinitely long sinc interpolant:

$$W(x,n) = \frac{\sin\left[\pi\,(x-n)\right]}{\pi\,(x-n)}\;. \tag{4}$$

According to the sampling theorem (Kotel'nikov, 1933; Shannon, 1949), equation (4) provides an optimal interpolation for any band-limited signal. In practice, it is not directly applicable because of a prohibitively expensive computation. The shape of the sinc function and its spectrum are shown in Figure 3. The spectrum is identically equal to one in the Nyquist frequency band.

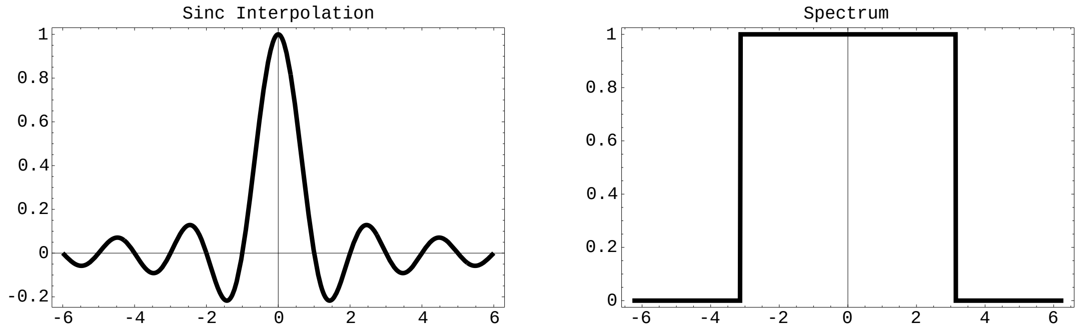


Figure 3: Sinc interpolant (left) and its spectrum (right). $\boxed{\text{sergey2-sincint}}$ [CR]

Several approaches exist for extending the nearest neighbor and linear interpolation to more accurate (albeit more expensive) methods. One example is the 4-point cubic convolution suggested by Keys (1981). The cubic convolution interpolant is a local piece-wise cubic function, which approximates the ideal sinc equation (4). Another popular approach is to taper the ideal sinc function in a local window. For example, one can use the Kaiser window (Kaiser and Shafer, 1980)

$$W(x,n) = \begin{cases} \dfrac{\sin\left[\pi\,(x-n)\right]}{\pi\,(x-n)}\,\dfrac{I_0\left(\alpha\sqrt{1-\left(\frac{x-n}{N}\right)^2}\right)}{I_0(\alpha)} & \text{for} \quad n-N < x < n+N \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where I_0 is the zero-order modified Bessel function of the first kind. The Kaiser-windowed sinc interpolant (5) has the adjustable parameter α, which controls the behavior of its spectrum. I have found empirically the value of $\alpha = 4$ to provide a spectrum that deviates from 1 by no more than 1% in a relatively wide band.

I compare the accuracy of different forward interpolation methods on a one-dimensional signal shown in Figure 4. The ideal signal has an exponential amplitude decay and a quadratic frequency increase from the center towards the edges. It is sampled at a regular 50-point grid and interpolated to 500 regularly sampled locations. The interpolation result is compared with the ideal one. Observing Figures 5, 6, and 7, we can see the interpolation error steadily decreasing as we go subsequently from 1-point nearest neighbor to 2-point linear, 4-point

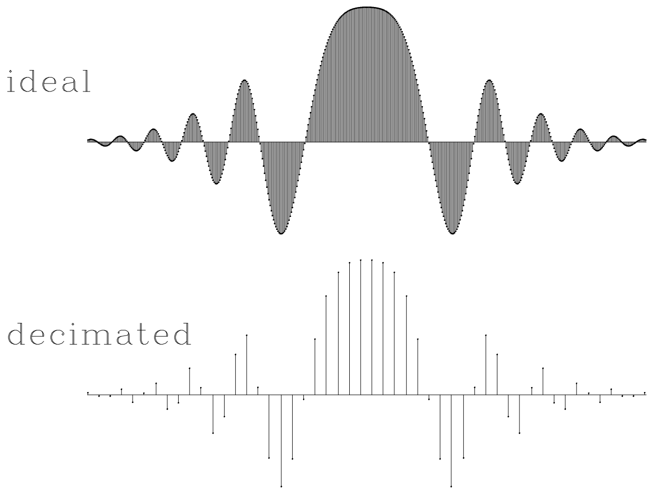


Figure 4: One-dimensional test signal. Top: ideal. Bottom: sampled at 50 regularly spaced points. The bottom plot is the input in a forward interpolation test. sergey2-chirp [ER]

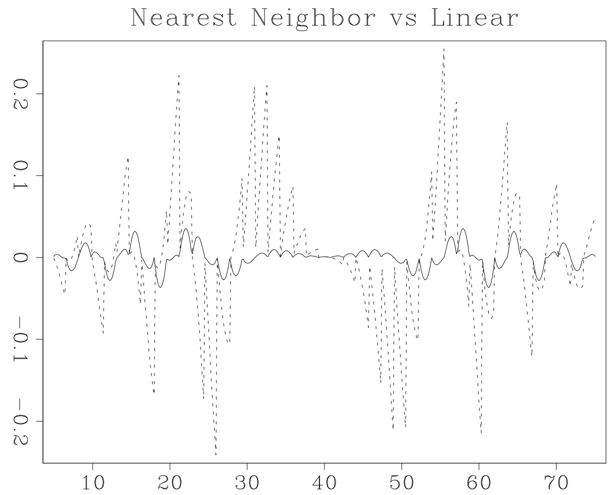


Figure 5: Interpolation error of the nearest neighbor interpolant (dashed line) compared to that of the linear interpolant (solid line). sergey2-binlin [ER]

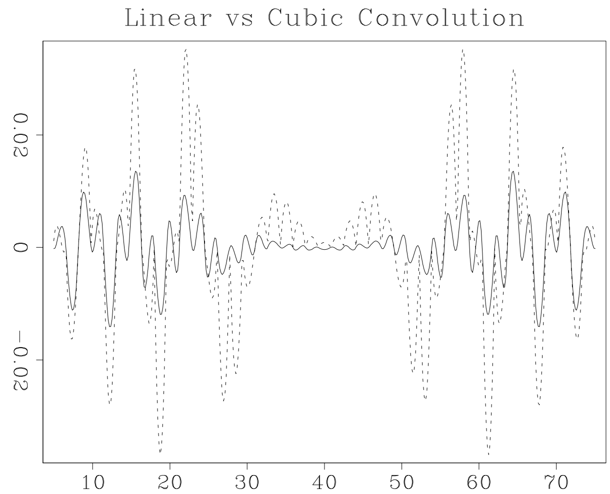


Figure 6: Interpolation error of the linear interpolant (dashed line) compared to that of the cubic convolution interpolant (solid line). sergey2-lincub [ER]

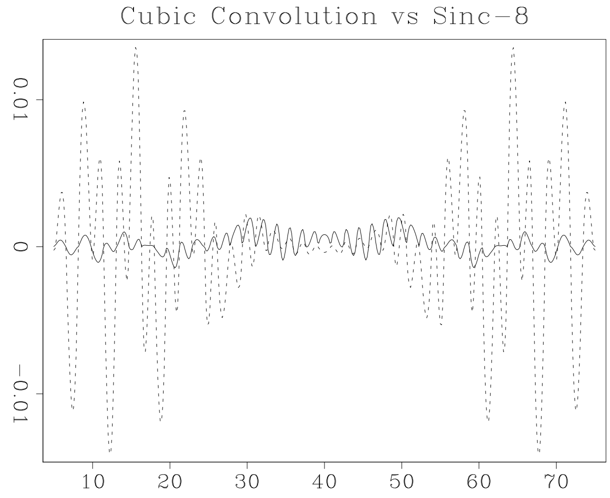


Figure 7: Interpolation error of the cubic convolution interpolant (dashed line) compared to that of the 8-point windowed sinc interpolant (solid line). sergey2-cubkai [ER]

cubic convolution, and 8-point windowed sinc interpolation. At the same time, the cost of interpolation grows proportionally to the interpolant length.

The differences among different methods are also clearly visible from the discrete spectra of the corresponding interpolants. The left plots in figures 8 and 9 show discrete interpolation responses: the function $W(x,n)$ for a fixed value of $x = 0.7$. The right plots compare the corresponding discrete spectra. We can see that the spectrum gets flatter and wider as the accuracy of the method increases.

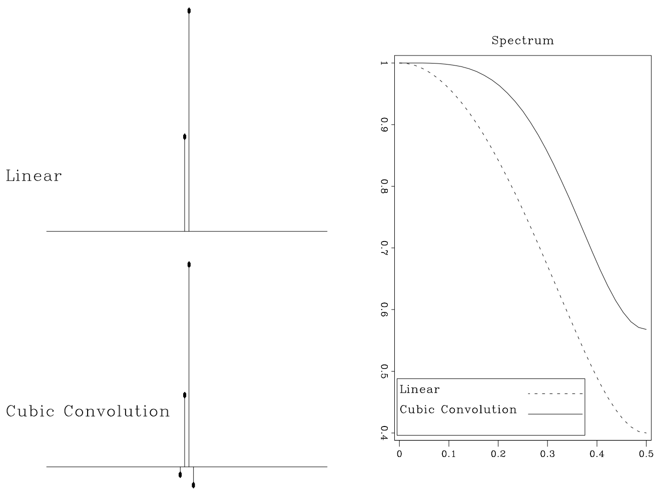


Figure 8: Discrete interpolation responses of linear and cubic convolution interpolants (left) and their discrete spectra (right) for $x = 0.7$. sergey2-speclincub [ER]

Interpolation and convolution

As I discussed in an earlier paper (Fomel, 1997b), a general approach for constructing the interpolant function $W(x,n)$ in equation (1) is to select an appropriate function basis for representing the function $f(x)$. The functional basis representation has the general form

$$f(x) = \sum_{k \in K} c_k \psi_k(x) \,, \tag{6}$$

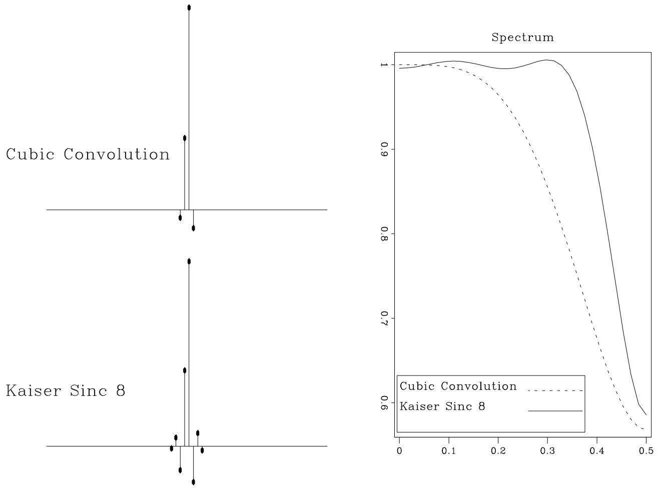


Figure 9: Discrete interpolation responses of cubic convolution and 8-point windowed sinc interpolants (left) and their discrete spectra (right) for $x = 0.7$. sergey2-speccubkai [ER]

where $\psi_k(x)$ are basis function, and c_k are the corresponding coefficients. Once an appropriate basis is selected, one can define the $W(x,n)$ function by means of the least squares method.

Unser et al. (1993a) noticed that the function basis idea has an especially simple implementation if the basis is convolutional and satisfies the equation

$$\psi_k(x) = \beta(x-k) \; . \tag{7}$$

In other words, the basis is constructed by integer shifts of a single function $\beta(x)$. Substituting formula (7) into equation (6) yields

$$f(x) = \sum_{k \in K} c_k \beta(x-k) \; . \tag{8}$$

Evaluating the function $f(x)$ in equation (8) at an integer value n, we obtain the equation

$$f(n) = \sum_{k \in K} c_k \beta(n-k) \; , \tag{9}$$

which has the exact form of a discrete convolution. The basis function $\beta(x)$, evaluated at integer values, is digitally convolved with the vector of basis coefficients to produce the sampled values of the function $f(x)$. We can invert equation (9) to obtain the coefficients c_k from $f(n)$ by inverse recursive filtering (deconvolution). In the case of a non-causal filter $\beta(n)$, an appropriate spectral factorization will be needed prior to applying the recursive filtering.

According to the convolutional basis idea, forward interpolation becomes a two-step procedure. The first step is the direct inversion of equation (9): the basis coefficients c_k are found by deconvolving the sampled function $f(n)$ with the factorized filter $\beta(n)$. The second step reconstructs the continuous (or arbitrarily sampled) function $f(x)$ according to formula (8). The two steps could be combined into one, but usually it is more convenient to apply them separately. I show a schematic relationship among different variables in Figure 10.

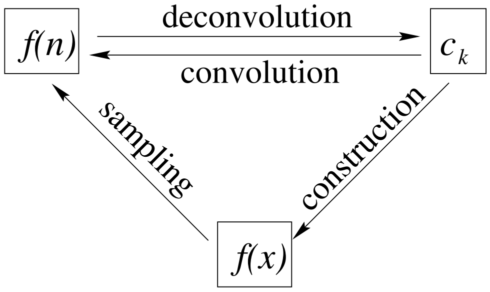


Figure 10: Schematic relationship among different variables for interpolation with a convolutional basis. sergey2-scheme [NR]

B-splines

B-splines represent a particular example of a convolutional basis. Because of their compact support and other attractive numerical properties, B-splines are a good basis choice for the forward interpolation problem and related signal processing problems (Unser, 1999).

B-splines of the order 0 and 1 coincide with the nearest neighbor and linear interpolants (2) and (3) respectively. B-splines $\beta^n(x)$ of a higher order n can be defined by a repetitive convolution of the zeroth-order spline $\beta^0(x)$ (the box function) with itself:

$$\beta^n(x) = \underbrace{\beta^0(x) * \cdots * \beta^0(x)}_{(n+1) \quad \text{times}} . \tag{10}$$

There is also the explicit expression

$$\beta^n(x) = \frac{1}{n!} \sum_{k=0}^{n+1} C_k^{n+1} (-1)^k (x + \frac{n+1}{2} - k)_+^n , \tag{11}$$

which can be proved by induction. Here C_k^{n+1} are the binomial coefficients, and the function x_+ is defined as follows:

$$x_+ = \left\{ \begin{array}{lcl} x, & \text{for} & x > 0 \\ 0, & \text{otherwise} & \end{array} \right. \tag{12}$$

As follows from formula (11), the most commonly used cubic B-spline $\beta^3(x)$ has the expression

$$\beta^3(x) = \left\{ \begin{array}{lcl} \left(4 - 6|x|^2 + 3|x|^3\right)/6, & \text{for} & 1 > |x| \geq 0 \\ (2-|x|)^3/6, & \text{for} & 2 > |x| \geq 1 \\ 0, & \text{elsewhere} & \end{array} \right. \tag{13}$$

The corresponding discrete filter $\beta^3(n)$ is a centered 3-point filter with coefficients 1/6, 2/3, and 1/6. According to the traditional method, a deconvolution with this filter is performed as a tridiagonal matrix inversion (de Boor, 1978). One can accomplish it more efficiently by spectral factorization and recursive filtering (Unser et al., 1993a). The recursive filtering approach generalizes straightforwardly to B-splines of higher orders.

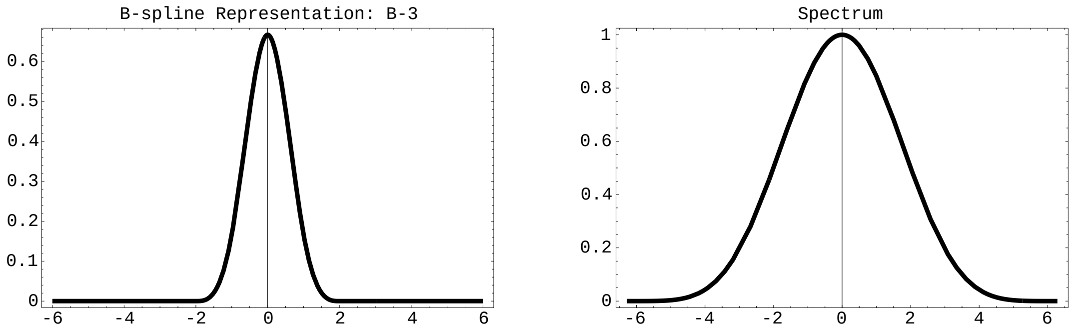


Figure 11: Third-order B-spline $\beta^3(x)$ (left) and its spectrum (right). sergey2-splint3 [CR]

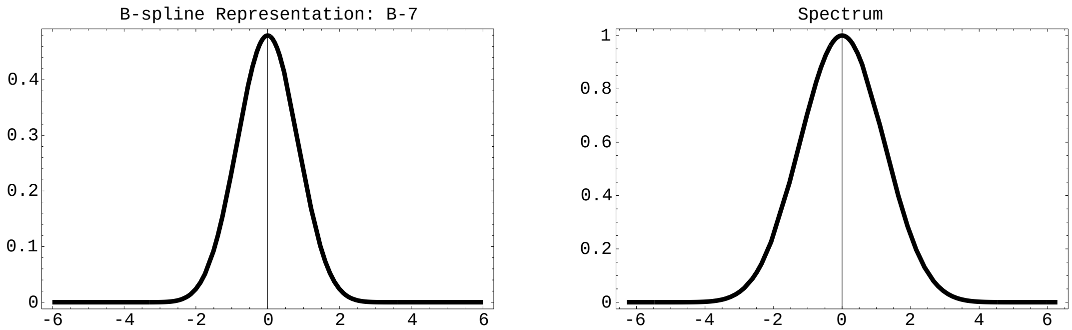


Figure 12: Seventh-order B-spline $\beta^7(x)$ (left) and its spectrum (right). sergey2-splint7 [CR]

Both the support length and the smoothness of B-splines increase with the order. In the limit, B-slines converge to the Gaussian function. Figures 11 and 12 show the third- and seventh-order splines $\beta^3(x)$ and $\beta^7(x)$ and their continuous spectra.

It is important to realize the difference between B-splines and the corresponding interpolants $W(x,n)$, which are sometimes called *cardinal splines*. An explicit computation of the cardinal splines is impractical, because they have infinitely long support. Typically, they are constructed implicitly by the two-step interpolation method, outlined in the previous subsection. The cardinal splines of orders 3 and 7 and their spectra are shown in Figures 13 and 14. As B-splines converge to the Gaussian function, the corresponding interpolants rapidly converge to the sinc function (4). A good convergence is achieved with the help of the infinitely long support, which results from recursive filtering at the first step of the interpolation procedure.

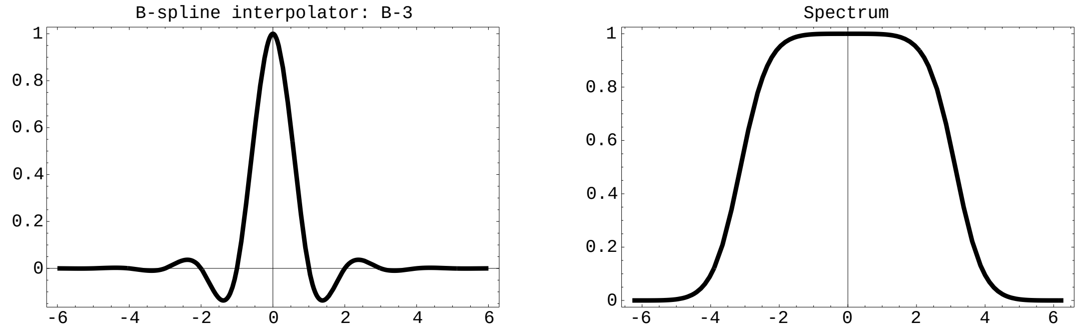


Figure 13: Effective third-order B-spline interpolant (left) and its spectrum (right). sergey2-crdint3 [CR]

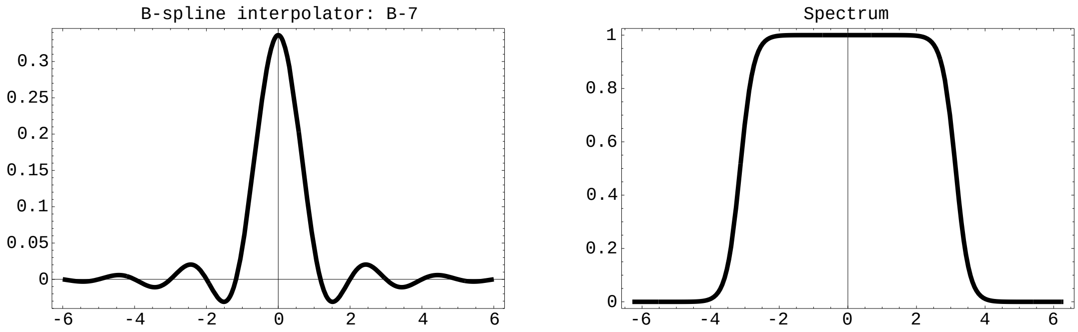


Figure 14: Effective seventh-order B-spline interpolant (left) and its spectrum (right). sergey2-crdint7 [CR]

In practice, the recursive filtering step adds only marginally to the total interpolation cost. Therefore, an n-th order B-spline interpolation is comparable in cost with any other method with an $(n+1)$-point interpolant. The comparison in accuracy usually turns out in favor of B-splines. Figures 15 and 16 compare interpolation errors of B-splines and other similar-cost methods on the example from Figure 4.

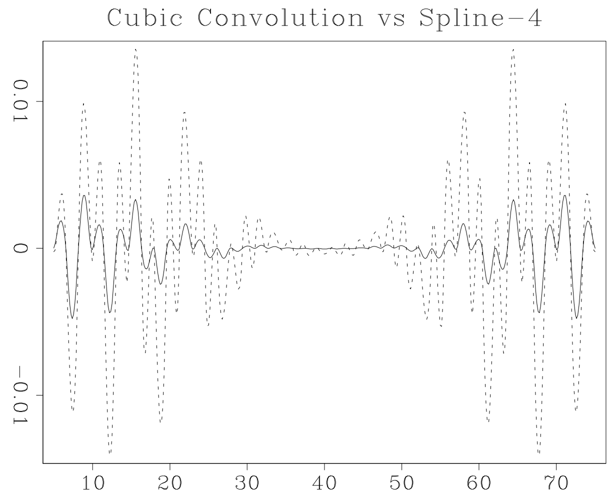


Figure 15: Interpolation error of the cubic convolution interpolant (dashed line) compared to that of the third-order B-spline (solid line). sergey2-cubspl [ER]

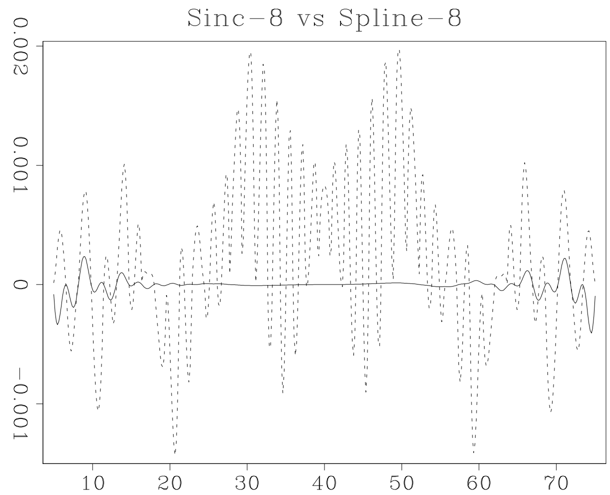


Figure 16: Interpolation error of the 8-point windowed sinc interpolant (dashed line) compared to that of the seventh-order B-spline (solid line). sergey2-kaispl [ER]

Similarly to Figures 8 and 9, we can also compare the discrete responses of B-spline interpolation with those of other methods. The right plots in Figures 17 and 18 show that the discrete spectra of the effective B-spline interpolants are genuinely flat at low frequencies and wider than those of the competitive methods. Although the B-spline responses are infinitely long because of the recursive filtering step, they exhibit a fast amplitude decay.

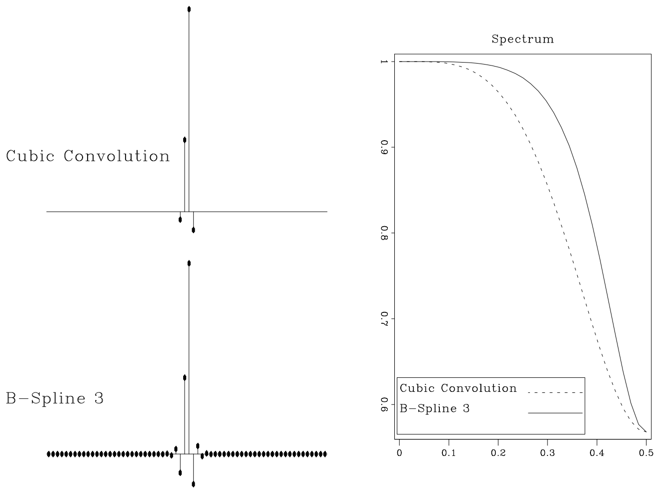


Figure 17: Discrete interpolation responses of cubic convolution and third-order B-spline interpolants (left) and their discrete spectra (right) for $x = 0.7$. sergey2-speccubspl [ER]

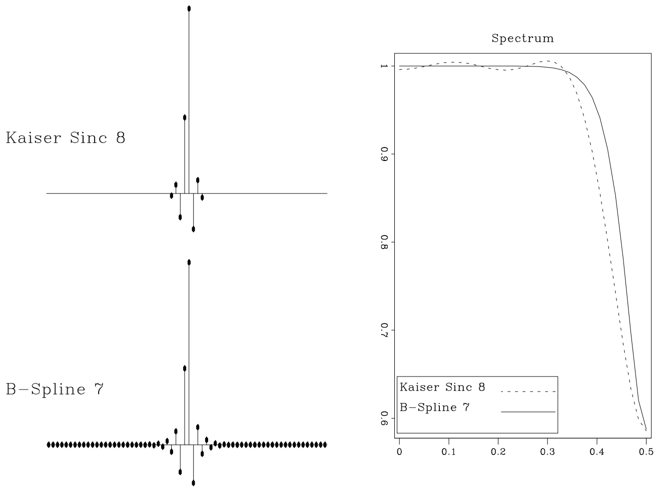


Figure 18: Discrete interpolation responses of 8-point windowed sinc and seventh-order B-spline interpolants (left) and their discrete spectra (right) for $x = 0.7$. sergey2-speckaispl [ER]

2-D example

For completeness, I include a 2-D forward interpolation example. Figure 19 shows a 2-D analog of function in Figure 4 and its coarsely-sampled version.

Figure 20 compares the errors of the 2-D nearest neighbor and 2-D linear (bi-linear) interpolation. Switching to bi-linear interpolation shows a significant improvement, but the error level is still relatively high. As shown in Figures 21 and 22, B-spline interpolation again outperforms other methods with comparable cost complexity. In all cases, I constructed 2-D interpolants by orthogonal splitting. Although the splitting method reduces computational

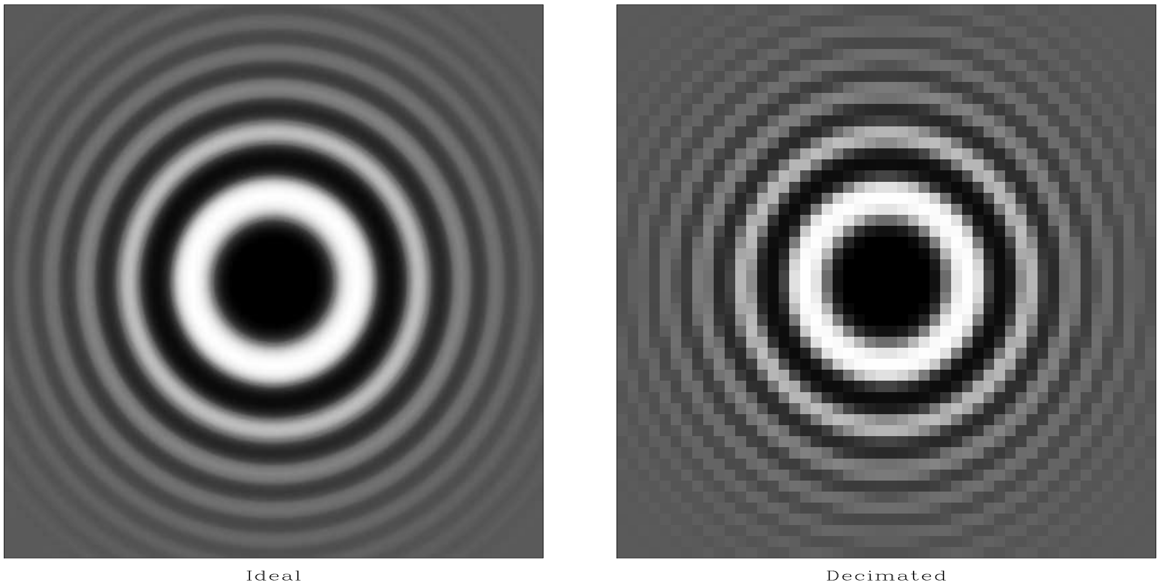


Figure 19: Two-dimensional test function (left) and its coarsely sampled version (right). sergey2-chirp2 [ER]

overhead, the main cost factor is the total interpolant size, which squares when going from 1-D to 2-D.

Beyond B-splines

It is not too difficult to construct a convolutional basis with better interpolation properties than those of B-splines, for example by sacrificing their smoothness. The following piecewise cubic function has a lower smoothness than $\beta^3(x)$ in equation (13) but slightly better interpolation behavior:

$$\mu^3(x) = \begin{cases} \left(10 - 13|x|^2 + 6|x|^3\right)/16, & \text{for} \quad 1 > |x| \geq 0 \\ (2-|x|)^2(5-2|x|)/16, & \text{for} \quad 2 > |x| \geq 1 \\ 0, & \text{elsewhere} \end{cases} \tag{14}$$

Figures 23 and 24 compare the test interpolation errors and discrete responses of methods based on the B-spline function $\beta^3(x)$ and the lower smoothness function $\mu^3(x)$. The latter method has a slight but visible performance advantage and a slightly wider discrete spectrum.

Blu et al. (1998) have developed a general approach for constructing non-smooth piecewise functions with optimal interpolation properties. However, the gain in accuracy is often negligible in practice. In the rest of this paper, I use the classic B-spline method.

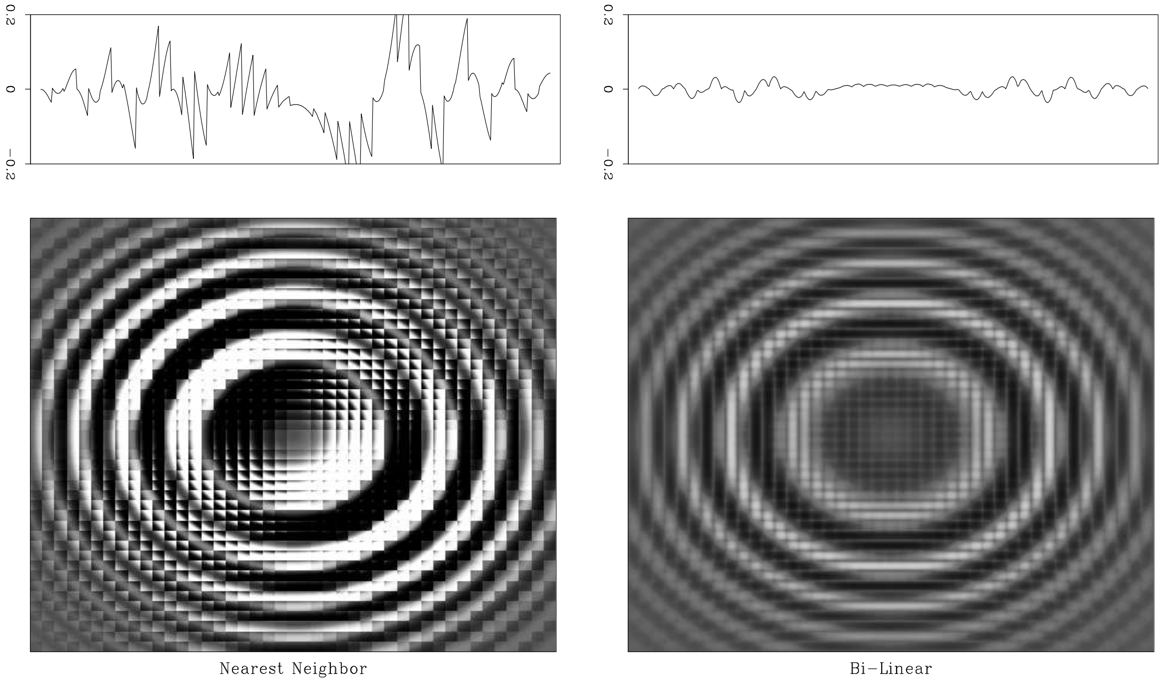


Figure 20: 2-D Interpolation errors of nearest neighbor interpolation (left) and linear interpolation (right). Top graphs show 1-D slices through the center of the image. sergey2-plcbinlin [ER]

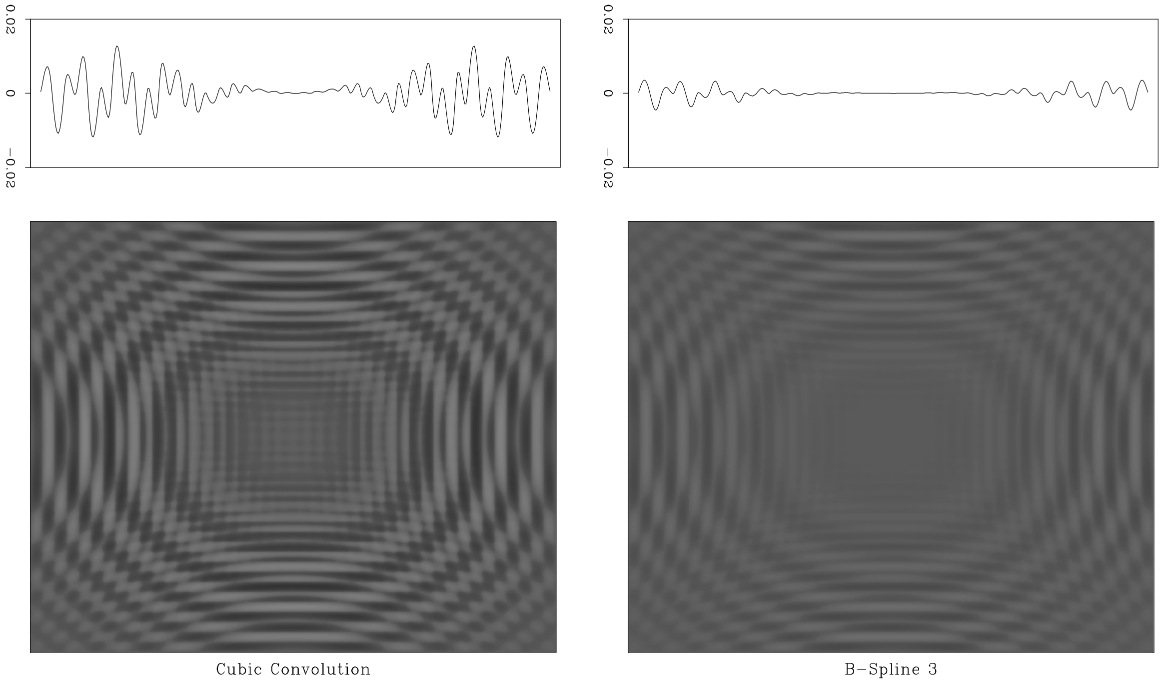


Figure 21: 2-D Interpolation errors of cubic convolution interpolation (left) and third-order B-spline interpolation (right). Top graphs show 1-D slices through the center of the image. sergey2-plccubspl [ER]

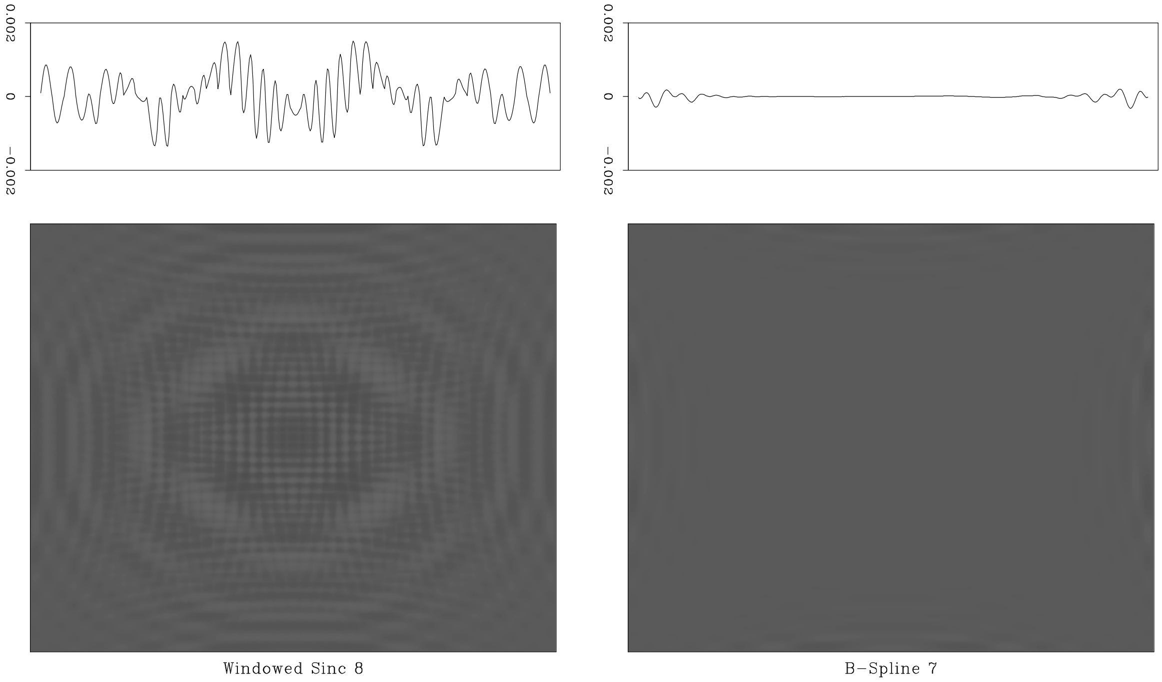


Figure 22: 2-D Interpolation errors of 8-point windowed sinc interpolation (left) and seventh-order B-spline interpolation (right). Top graphs show 1-D slices through the center of the images. sergey2-plckaispl [ER]

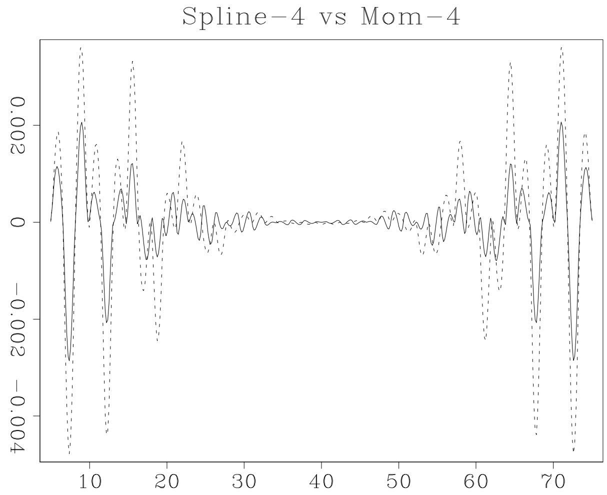


Figure 23: Interpolation error of the third-order B-spline interpolant (dashed line) compared to that of the lower smoothness spline interpolant (solid line). sergey2-splmom4 [ER]

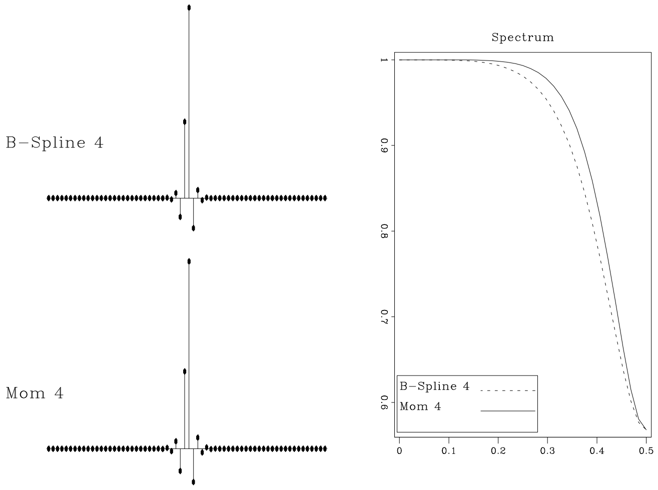


Figure 24: Discrete interpolation responses of third-order B-spline and lower smoothness spline interpolants (left) and their discrete spectra (right) for $x = 0.7$. sergey2-specsplmom4 [ER]

Seismic applications of forward interpolation

For completeness, I conclude this section with two simple examples of forward interpolation in seismic data processing. Figure 25 shows a 3-D impulse response of Stolt migration (Stolt, 1978), computed by using 2-point linear interpolation and 8-point B-spline interpolation. As noted by Ronen (1982) and Harlan (1982), inaccurate interpolation may lead to spurious artifact events in Stolt-migrated image. Indeed, we see several artifacts for the image with linear interpolation (the left plots in Figure 25.) The artifacts are removed by a more accurate interpolation method (the right plots in Figure 25.)

Another simple example is radial trace transform (Ottolini, 1982) Figure 26 shows a land shot gather contaminated by nearly radial ground-roll. As discussed by Claerbout (1983), Henley (1999), and Brown and Claerbout (2000), one can effectively eliminate ground-roll noise by applying radial trace transform, followed by high-pass filtering and the inverse radial transform. Figure 27 shows the result of the forward radial transform of the shot gather in Figure 26 in the radial band of the ground-roll noise and the transform error after going back to the original domain. Comparing the results of using linear and third-order B-spline interpolation, we see once again that the transform artifacts are removed with a more accurate interpolation scheme.

INVERSE INTERPOLATION AND DATA REGULARIZATION

In the notation of Claerbout (1999), inverse interpolation amounts to a least-squares solution of the system

$$\mathbf{L}\mathbf{m} \approx \mathbf{d}\,; \tag{15}$$

$$\epsilon \mathbf{A}\mathbf{m} \approx \mathbf{0}\,, \tag{16}$$

where $\mathbf{d}$ is a vector of known data $f(x_i)$ at irregular locations x_i, $\mathbf{m}$ is a vector of unknown function values $f(n)$ at a regular grid n, $\mathbf{L}$ is a linear interpolation operator of the general

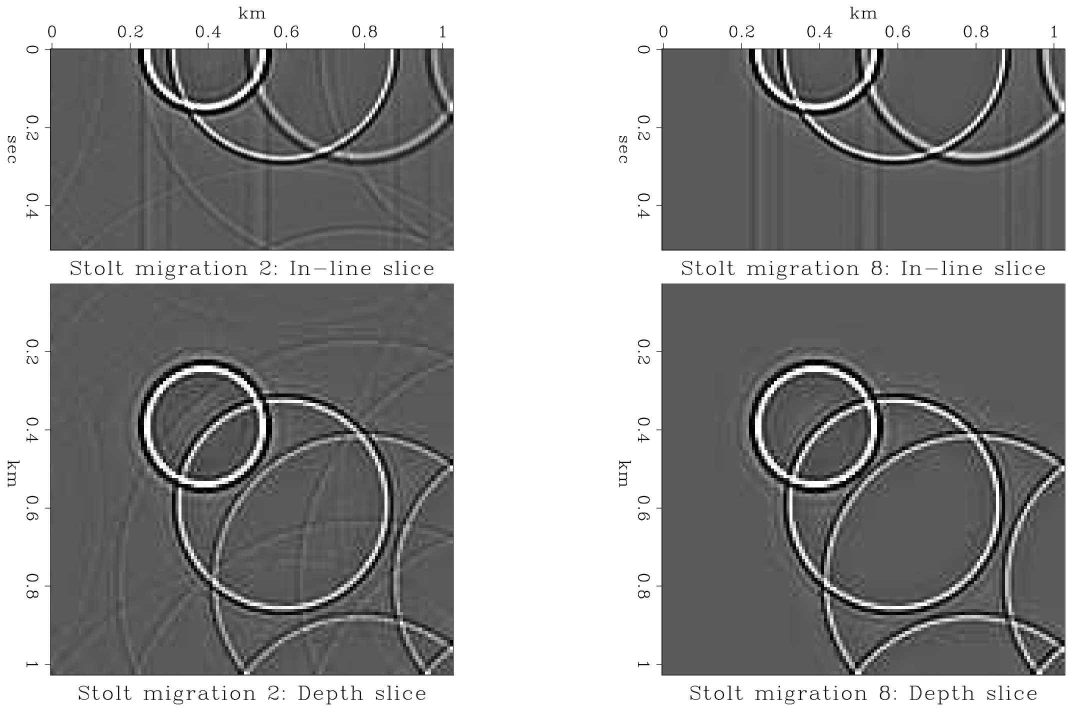


Figure 25: Stolt migration impulse response. Left: using linear interpolation. Right: using seventh-order B-spline interpolation. Migration artifacts are removed by a more accurate forward interpolation method. sergey2-stolt [ER]

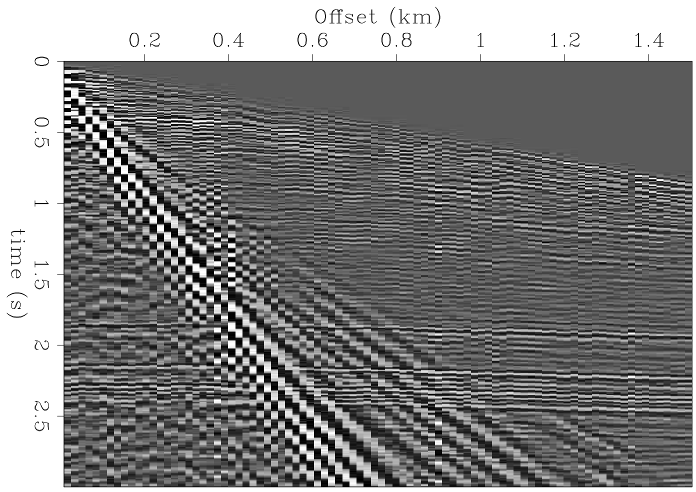


Figure 26: Ground-roll-contaminated shot gather used in a radial transform test sergey2-radialdat [ER]

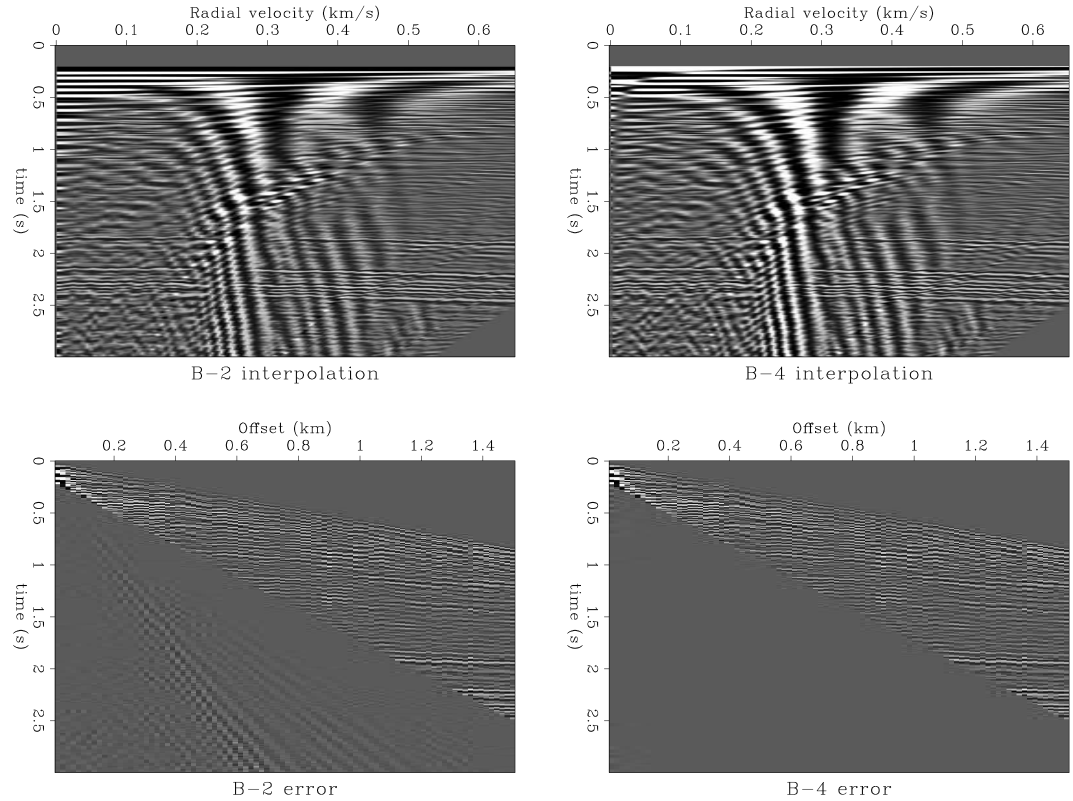


Figure 27: Radial trace transform results. Top: radial trace domain. Bottom: residual error after the inverse transform. The error should be zero in a radial band from 0 to 0.65 km/s radial velocity. Left: using linear interpolation. Right: using third-order B-spline interpolation. sergey2-radial [ER]

form (1), $\mathbf{A}$ is an appropriate regularization (model styling) operator, and ϵ is a scaling parameter. In the case of B-spline interpolation, the forward interpolation operator $\mathbf{L}$ becomes a cascade of two operators: recursive deconvolution $\mathbf{B}^{-1}$, which converts the model vector $\mathbf{m}$ to the vector of spline coefficients $\mathbf{c}$, and a spline basis construction operator $\mathbf{F}$. System (15-16) transforms to

$$\begin{aligned} \mathbf{F}\mathbf{B}^{-1}\mathbf{m} &\approx \mathbf{d}\,; && (17)\\ \epsilon\mathbf{A}\mathbf{m} &\approx \mathbf{0}\,. && (18) \end{aligned}$$

We can rewrite (17-18) in the form that involves only spline coefficients:

$$\begin{aligned} \mathbf{W}\mathbf{c} &\approx \mathbf{d}\,; && (19)\\ \epsilon\mathbf{A}\mathbf{B}\mathbf{c} &\approx \mathbf{0}\,. && (20) \end{aligned}$$

After we find a solution of system (19-20), the model $\mathbf{m}$ will be reconstructed by the simple convolution

$$\mathbf{m} = \mathbf{B}\mathbf{c}\,. \tag{21}$$

This approach resembles a more general method of model preconditioning (Fomel, 1997a).

The inconvenient part of system (19-20) is the complex regularization operator $\mathbf{AB}$. Is it possible to avoid the cascade of $\mathbf{B}$ and $\mathbf{A}$ and to construct a regularization operator directly applicable to the spline coefficients $\mathbf{c}$? In the following subsection, I develop a method for constructing spline regularization operators from differential equations.

Spline regularization

In many cases, the regularization (styling) condition originates in a continuous differential operator. For example, one can think of the gradient or Laplacian operator for regularizing smooth functions (Fomel, 2000b), plane-wave destructor for regularizing local plane waves (Fomel, 2000a), or the offset continuation equation for regularizing seismic reflection data (Fomel, 2000c).

Let us denote the continuous regularization operator by D. Regularization implies seeking a function $f(x)$ such that the least-squares norm of $D\left[f(x)\right]$ is minimum. Using the usual expression for the least-squares norm of continuous functions and substituting the basis decomposition (8), we obtain the expression

$$\left\|D\left[f(x)\right]\right\| = \int \left(D\left[f(x)\right]\right)^2 dx = \int \left(\sum_{k\in K} c_k D\left[\beta(x-k)\right]\right)^2 dx\,. \tag{22}$$

The problem of finding function $f(x)$ reduces to the problem of finding the corresponding set of basis coefficients c_k. We can obtain the solution to the least-squares optimization by differentiating the quadratic objective function (22) with respect to the basis coefficients c_k. This leads to the system of linear equations

$$\sum_{k \in K} c_k \int D\left[\beta(x-k)\right] D\left[\beta(x-j)\right] dx = \sum_{k \in K} c_k d_{j-k} = 0 \,, \tag{23}$$

where

$$d_j = \int D\left[\beta(x)\right] D\left[\beta(x-j)\right] dx \,. \tag{24}$$

Equation (23) is clearly a discrete convolution of the spline coefficients c_k with the filter d_j defined in equation (24). To transform the system (23) to a regularization condition of the form

$$\mathbf{Dc} \approx \mathbf{0} \,, \tag{25}$$

we need to treat the digital filter d_j as an autocorrelation and find its minimum-phase factor. Equation (25) replaces equation (20) in the inverse interpolation problem setting.

We have found a constructive way of creating B-spline regularization operators from continuous differential equations.

A simple regularization example is shown in Figure 28. The continuous operator D in this case comes from the theoretical plane-wave differential equation. I constructed the autocorrelation filter d_j according to formula (24) and factorized it with the efficient Wilson-Burg method on a helix (Sava et al., 1998). The figure shows three plane waves constructed from three distant spikes by applying an inverse recursive filtering with two different plane-wave regularizers. The left plot corresponds to using first-order B-splines (equivalent to linear interpolation). This type of regularizer is identical to Clapp's steering filters (Clapp et al., 1997) and suffers from numerical dispersion effects. The right plot was obtained with third-order splines. Most of the dispersion is suppressed by using a more accurate interpolation.

Test example

Now that we have all the problem pieces together, we can test the performance gain in the inverse interpolation problem (19)-(25) from the application of B-splines.

For a simple 1-D test, I chose the function shown in Figure 4, but sampled at irregular locations. To create two different regimes for the inverse interpolation problem, I chose 50 and 500 random locations. The two sets of points were interpolated to 500 and 50 regular samples respectively. The first test corresponds to an under-determined situation, while the second test is clearly over-determined. Figures 29 and 30 show the input data for the two test after normalized binning to the selected regular bins.

I solved system (19)-(25) by the iterative conjugate-gradient method, utilizing a recursive filter preconditioning (Fomel, 1997a) for faster convergence. The regularization operator $\mathbf{D}$

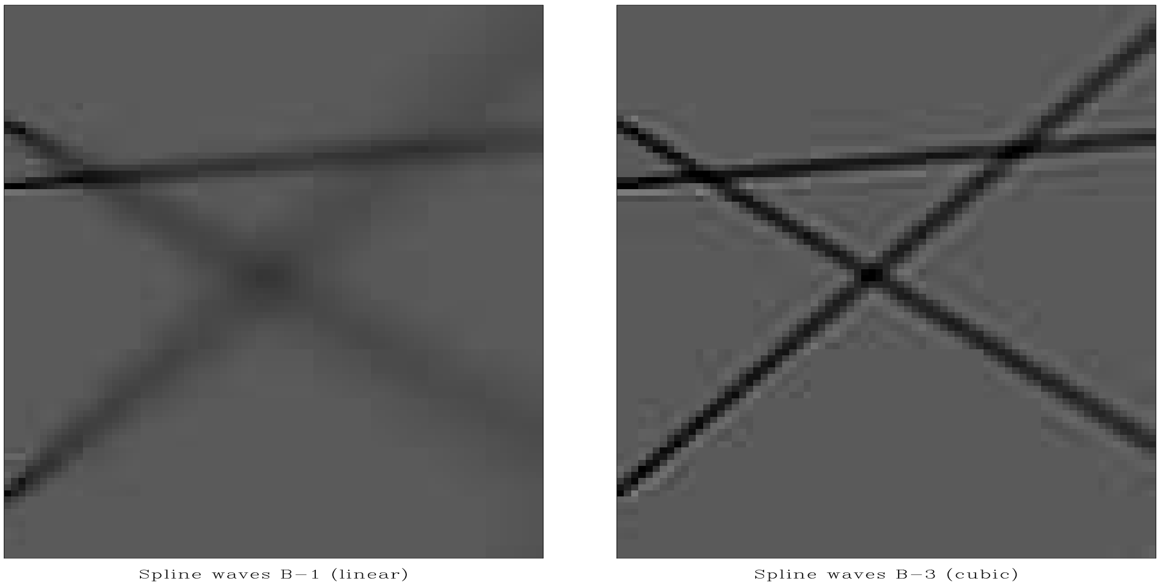


Figure 28: B-spline regularization. Three plane waves constructed by 2-D recursive filtering with the B-spline plane-wave regularizer. Left: using first-order B-splines (linear interpolation). Right: using third-order B-splines. sergey2-sthree [ER,M]

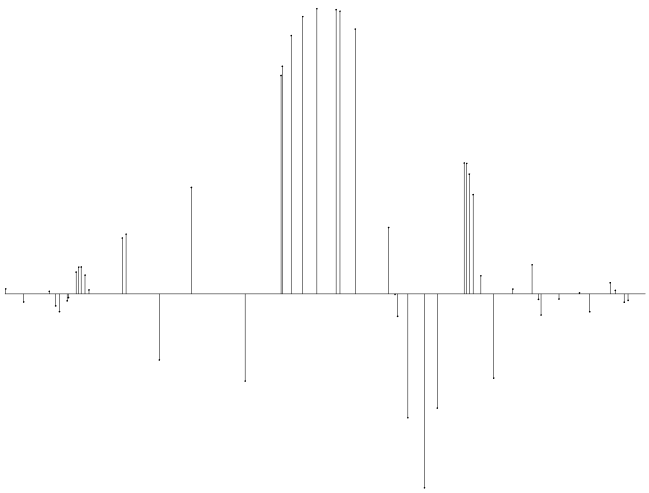


Figure 29: 50 random points binned to 500 regular grid points. The random data are used for testing inverse interpolation in an under-determined situation. sergey2-bin500 [ER]

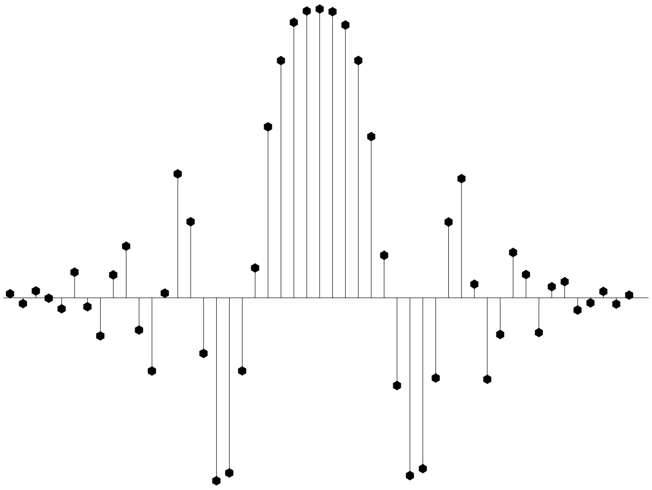


Figure 30: 500 random points binned to 50 regular grid points. The random data are used for testing inverse interpolation in an over-determined situation. sergey2-bin50 [ER]

was constructed by using the method of the previous subsection with the tension-spline differential equation (Smith and Wessel, 1990; Fomel, 2000b) and the tension parameter of 0.01.

The least-squares differences between the true and the estimated model are plotted in Figures 31 and 32. Observing the behavior of the model misfit versus the number of iterations and comparing simple linear interpolation with the third-order B-spline interpolation, we discover that

- In the under-determined case, both methods converge to the same final estimate, but B-spline inverse interpolation does it faster at earlier iterations. The total computational gain is not significant, because each B-spline iteration is more expensive than the corresponding linear interpolation iteration.
- In the over-determined case, both methods converge similarly at early iterations, but B-spline inverse interpolation results in a more accurate final estimate.

From the results of this simple experiment, it is apparent that the main advantage of using more accurate interpolation in the data regularization context occurs in the over-determined situation, when the estimated model is well constrained by the available data.

Application to 3-D seismic data regularization

In this subsection, I demonstrate an application of B-spline inverse interpolation for regularizing three-dimensional seismic reflection data. The dataset of this example comes from the North Sea and was used before for testing AMO (Biondi et al., 1998) and common-azimuth migration (Biondi, 1996). Figure 33 shows the midpoint geometry and the corresponding bin fold for a selected range of offsets and azimuths. The goal of data regularization is to create a regular data cube at the specified bins from the irregular input data, preprocessed by NMO. As typical of marine acquisition, the fold distribution is fairly regular but has occasional gaps caused by the cable feathering effect. The data cube after normalized binning (inverse nearest

Figure 31: Model convergence in the under-determined case. Dashed line: using linear interpolation. Solid line: using third-order B-spline. sergey2-norm500 [ER]

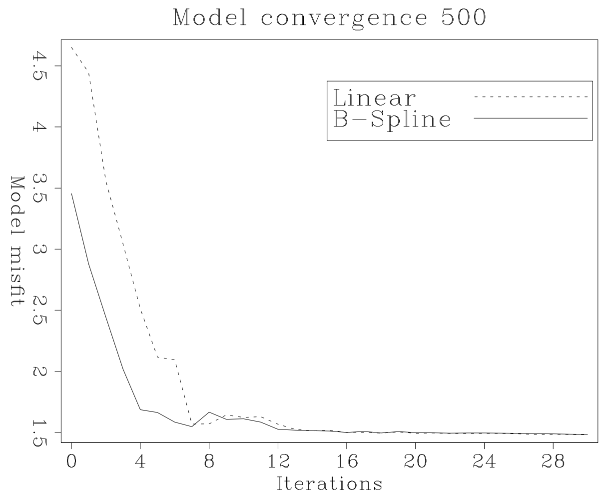
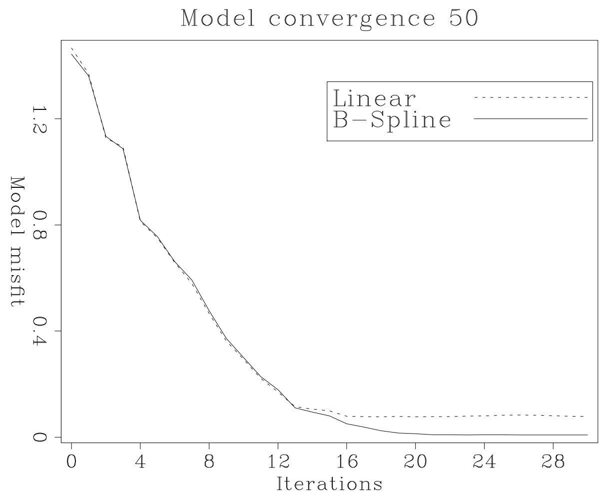


Figure 32: Model convergence in the over-determined case. Dashed line: using linear interpolation. Solid line: using third-order B-spline. sergey2-norm50 [ER]



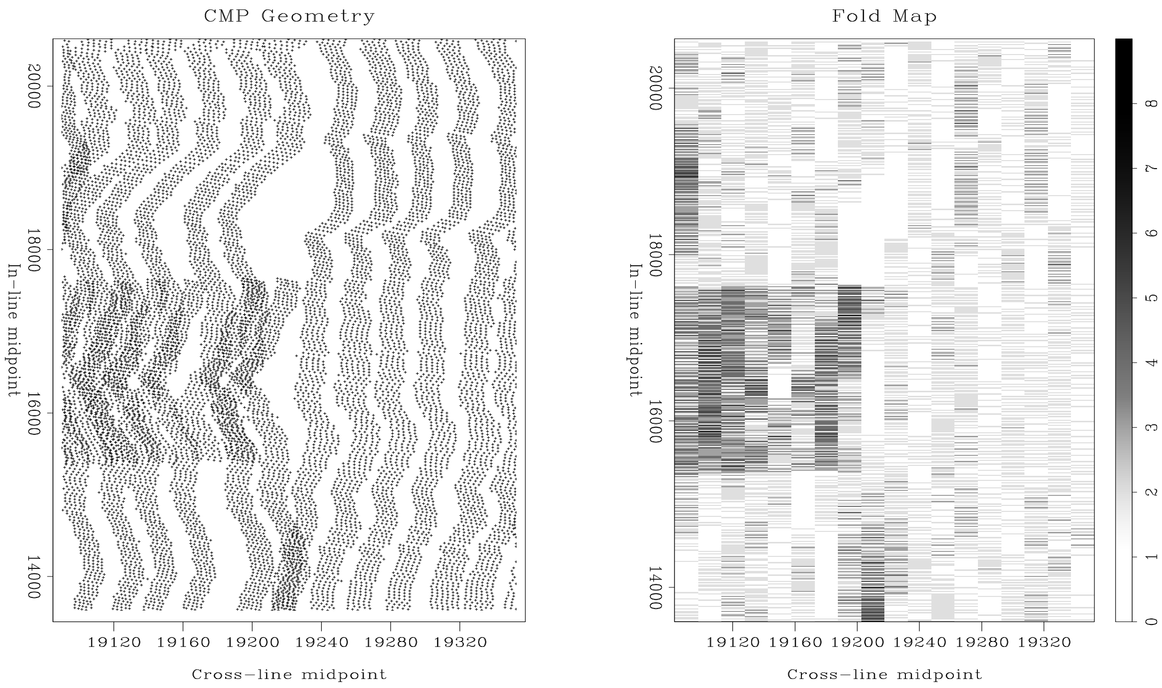


Figure 33: Midpoint geometry (left) and fold distribution (right) for the 3-D data test sergey2-cmpfold [ER]

neighbor interpolation) is shown in Figure 34. Binning works reasonably well in the areas of large fold but fails to fill the zero fold gaps and has an overall limited accuracy. Inverse interpolation using bi-linear interpolants significantly improves the result (Figure 35), and inverse B-spline interpolation improves the accuracy even further (Figure 36). In both cases, I regularized the data in constant time slices, using recursive filter preconditioning with plane-wave destructor filters analogous to those in Figure 28. The plane wave slope was estimated from the binned data with the method of Fomel (2000a). The inverse interpolation results preserve both flat reflection events in the data and steeply-dipping diffractions. When data regularization is used as a preprocessing step for common-azimuth migration (Biondi and Palacharla, 1996), preserving diffractions is important for correct imaging of sharp edges in the subsurface structure.

CONCLUSIONS

I have reviewed the B-spline forward interpolation method and confirmed the observation of Thévenaz et al. (2000) about its superior performance in comparison with other methods of similar cost. Whenever an accurate forward interpolation scheme is desired, B-splines can be an extremely valuable tool. B-spline forward interpolation involves two steps. The first step is recursive filtering, which results in a set of spline coefficients. The second step is a linear spline interpolation operator.

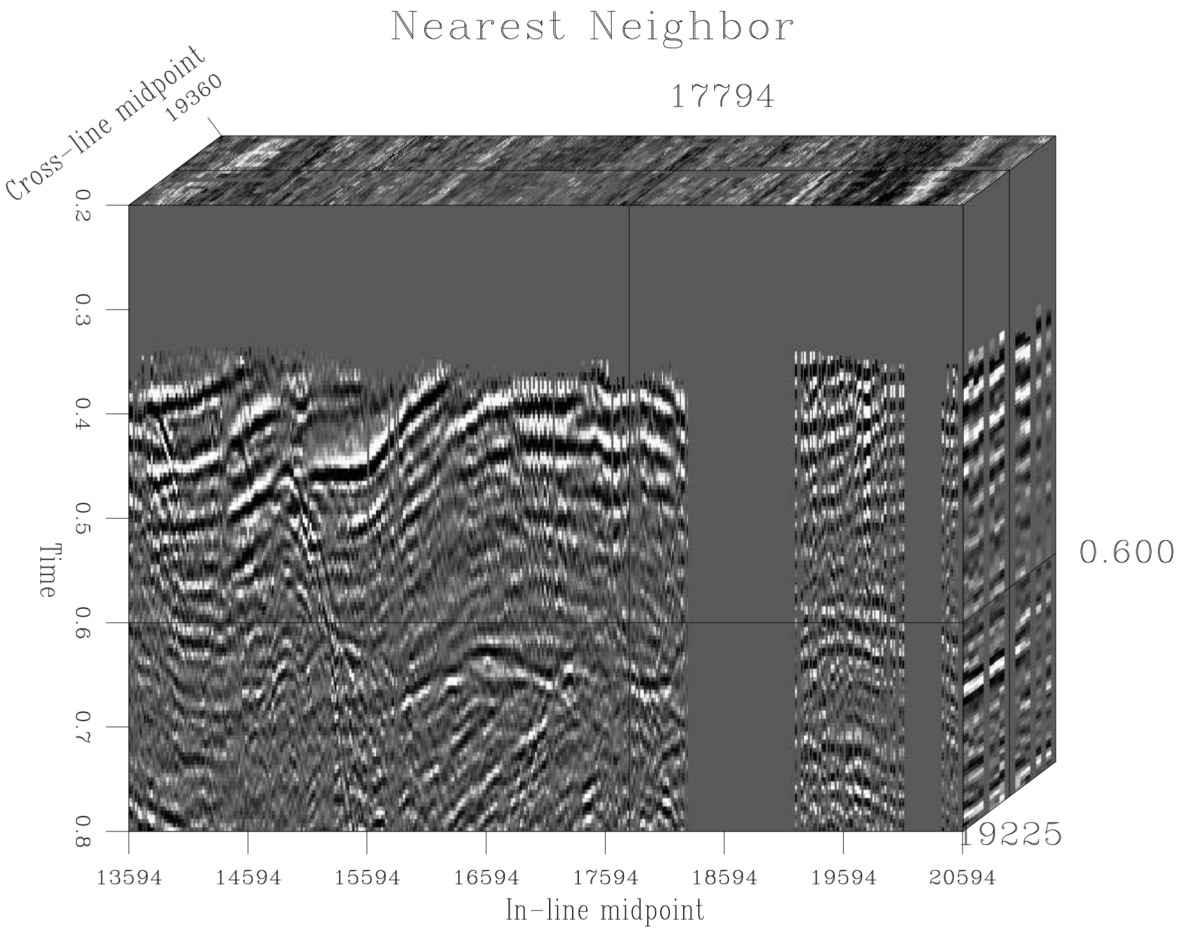


Figure 34: 3-D data after normalized binning sergey2-bin1 [ER]

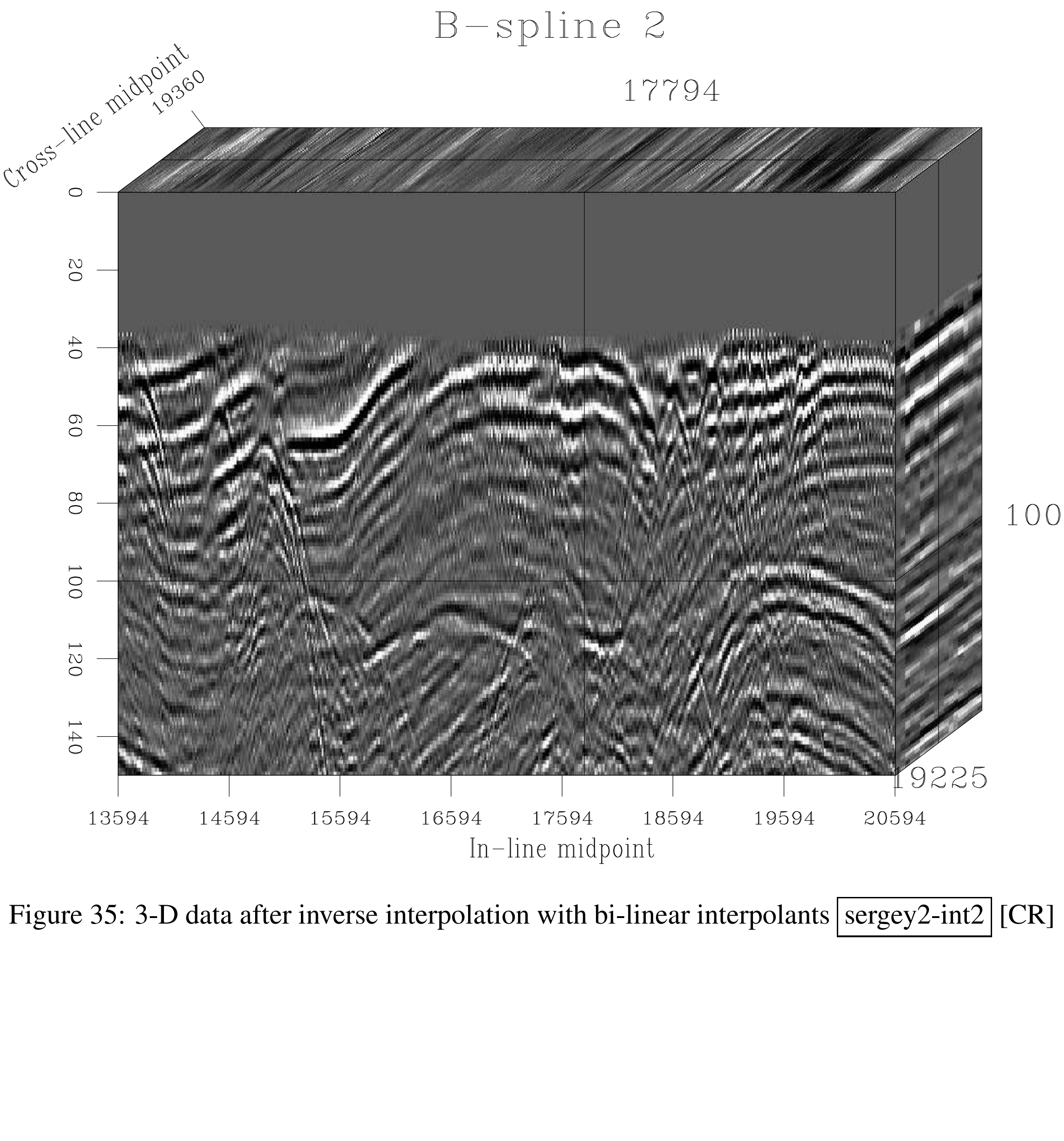


Figure 35: 3-D data after inverse interpolation with bi-linear interpolants sergey2-int2 [CR]

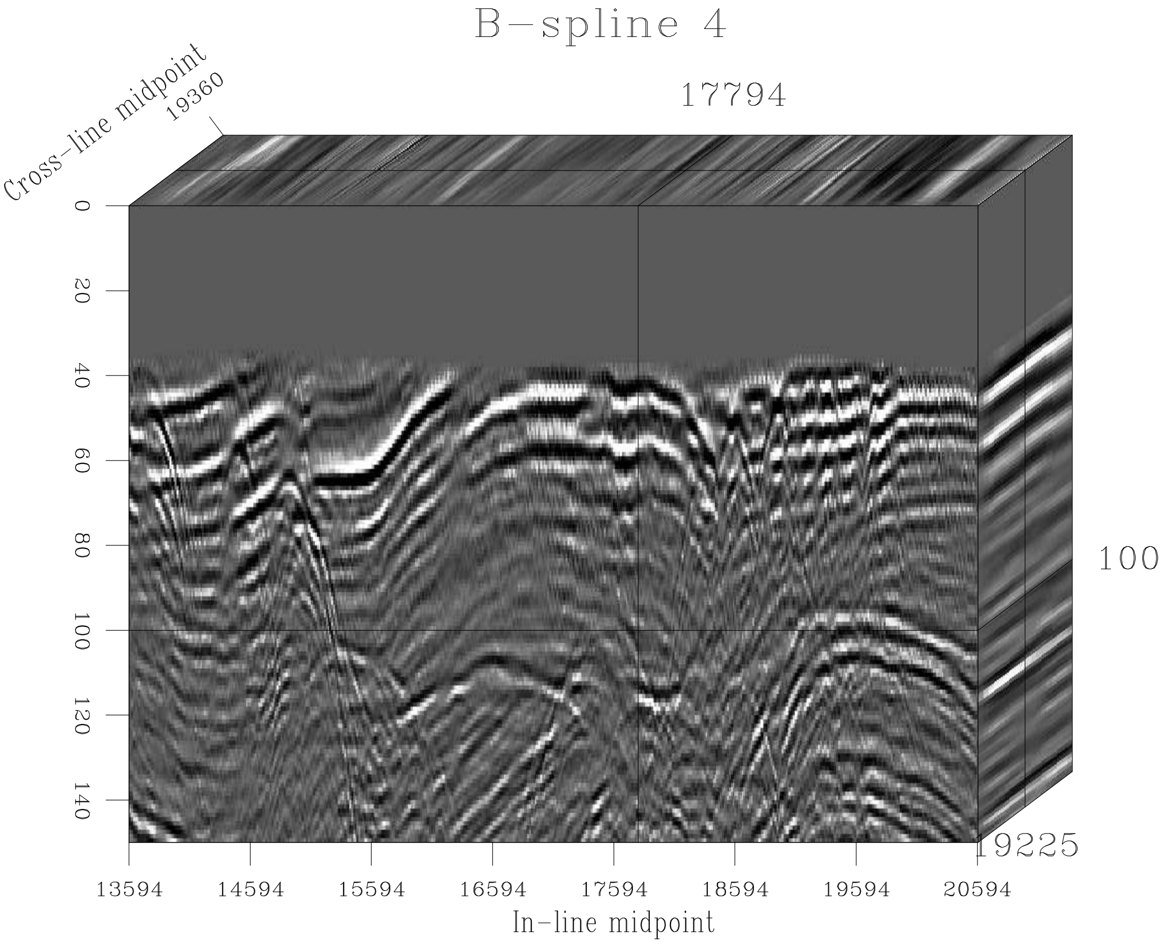


Figure 36: 3-D data after inverse interpolation with third-order B-spline interpolants sergey2-int4 [CR]

Analyzing the role of B-spline interpolation in data regularization, I have introduced a method of constructing B-spline discrete regularization operators from continuous differential equations.

Simple numerical experiments with B-spline inverse interpolation show that the main advantage of using a more accurate interpolation scheme occurs in an over-determined setting, where B-splines lead to a more accurate model estimates. In an under-determined setting, the B-spline inverse interpolation scheme converges faster at early iterations, but the total computational gain may be insignificant.

I have shown on a simple real data example that inverse B-spline interpolation can be used as an accurate method of data regularization for processing 3-D seismic reflection data.

ACKNOWLEDGMENTS

A short conversation with Dave Hale brought me to a better understanding of different forward interpolation methods.

The 3-D North Sea dataset was released to SEP by Conoco and its partners, BP and Mobil.

REFERENCES

Biondi, B., and Palacharla, G., 1996, 3-D prestack migration of common-azimuth data: Geophysics, **61**, no. 6, 1822–1832.

Biondi, B., Fomel, S., and Chemingui, N., 1998, Azimuth moveout for 3-D prestack imaging: Geophysics, **63**, no. 2, 574–588.

Biondi, B., 1996, Common-azimuth prestack depth migration of a North Sea data set: SEP–**93**, 1–14.

Blu, T., Thévenaz, P., and Unser, M., 1998, Minimum support interpolators with optimum approximation properties: Proc. IEEE Int. Conf. Image Processing, Chicago, IL, USA, October 4-7, 242–245.

Brown, M., and Claerbout, J., 2000, Ground roll and the Radial Trace Transform - revisited: SEP–**103**, 219–237.

Claerbout, J. F., 1983, Ground roll and radial traces: SEP–**35**, 43–54.

Claerbout, J., 1998, Multidimensional recursive filters via a helix: Geophysics, **63**, 1532–1541.

Claerbout, J., 1999, Geophysical estimation by example: Environmental soundings image enhancement: Stanford Exploration Project, http://sepwww.stanford.edu/sep/prof/.

Clapp, R. G., Fomel, S., and Claerbout, J., 1997, Solution steering with space-variant filters: SEP–**95**, 27–42.

de Boor, C., 1978, A practical guide to splines: Springer-Verlag.

Fomel, S., 1997a, On model-space and data-space regularization: A tutorial: SEP–**94**, 141–164.

Fomel, S., 1997b, On the general theory of data interpolation: SEP–**94**, 165–179.

Fomel, S., 2000a, Applications of plane-wave destructor filters: SEP–**105**, 1–26.

Fomel, S., 2000b, Helical preconditioning and splines in tension: SEP–**103**, 289–301.

Fomel, S., 2000c, Seismic data interpolation with the offset continuation equation: SEP–**103**, 237–254.

Harlan, W. S., 1982, Avoiding interpolation artifacts in Stolt migration: SEP–**30**, 103–110.

Henley, D. C., 1999, The radial trace transform: an effective domain for coherent noise attenuation and wavefield separation: 69th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1204–1207.

Kaiser, J. F., and Shafer, R. W., 1980, On the use of the I0-Sinh window for spectrum analysis: IEEE Trans. Acoustics, Speech and Signal Processing, **ASSP-28(1)**, 105.

Keys, R. G., 1981, Cubic convolution interpolation for digital image processing: IEEE Trans. Acoust., Speech, Signal Process., **ASSP-29**, 1153–1160.

Kotel'nikov, V. A., 1933, On the transmission capacity of "ether" and wire in electrocommunications: Izd. Red. Upr. Svyazi RKKA.

Ottolini, R., 1982, Migration of reflection seismic data in angle-midpoint coordinates: Ph.D. thesis, Stanford University.

Ronen, J., 1982, Stolt migration; interpolation artifacts: SEP–**30**, 95–102.

Sava, P., Rickett, J., Fomel, S., and Claerbout, J., 1998, Wilson-Burg spectral factorization with application to helix filtering: SEP–**97**, 343–351.

Shannon, C. E., 1949, Communication in the presense of noise: Proc. I.R.E., **37**, 10–21.

Smith, W. H. F., and Wessel, P., 1990, Gridding with continuous curvature splines in tension: Geophysics, **55**, no. 3, 293–305.

Stolt, R. H., 1978, Migration by Fourier transform: Geophysics, **43**, no. 1, 23–48.

Thévenaz, P., Blu, T., and Unser, M., 2000, Image interpolation and resampling:, *in* Handbook of Medical Image Processing , in press.

Unser, M., Aldroubi, A., and Eden, M., 1993a, B-spline signal processing: Part I – Theory: IEEE Transactions on Signal Processing, **41**, 821–832.

Unser, M., Aldroubi, A., and Eden, M., 1993b, B-spline signal processing: Part II – Efficient design and applications: IEEE Transactions on Signal Processing, **41**, 834–848.

Unser, M., 1999, Splines: a perfect fit for signal and image processing: IEEE Signal Processing Magazine, **16**, no. 6, 22–38.

Wolberg, G., 1990, Digital image warping: IEEE Computer Society Press.