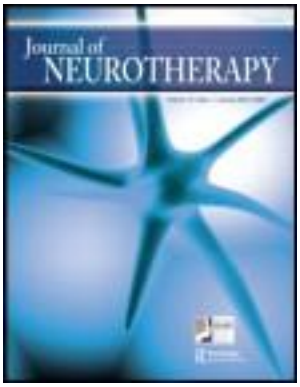


Journal of Neurotherapy: Investigations in Neuromodulation, Neurofeedback and Applied Neuroscience

Notes on EEG Resampling by Natural Cubic Spline Interpolation

Marco Congedo MA [a] , Cem Özen MS [b] & Leslie Sherlin BA [c]

[a] Department of Psychology , The University of Tennessee , Knoxville, TN

[b] Department of Physics and Astronomy , The University of Tennessee , Knoxville, TN

[c] Harold Abel School of Psychology, Capella University , Minneapolis, MN

Published online: 08 Sep 2008.

To cite this article: Marco Congedo MA , Cem Özen MS & Leslie Sherlin BA (2002) Notes on EEG Resampling by Natural Cubic Spline Interpolation, Journal of Neurotherapy: Investigations in Neuromodulation, Neurofeedback and Applied Neuroscience, 6:4, 73-80

To link to this article: http://dx.doi.org/10.1300/J184v06n04_08

Notes on EEG Resampling by Natural Cubic Spline Interpolation

Marco Congedo, MA
Cem Özen, MS
Leslie Sherlin, BA

ABSTRACT. Resampling of digitized electroencephalographic data allows changing the sampling rate with minimal distortion of the signal. Useful applications of the procedure include compatibility among diverse hardware and software and the customization of data analysis. The natural cubic spline interpolation procedure is introduced in a discursive fashion. A formal presentation is provided in the appendix.

KEYWORDS. Spline interpolation, cubic spline, natural spline, sample rate, EEG

Research and clinical practice in quantitative electroencephalography (qEEG) requires expertise in diverse fields. On occasion the re-

---

Marco Congedo is affiliated with the Department of Psychology, and Cem Özen is affiliated with the Department of Physics and Astronomy, The University of Tennessee, Knoxville, TN.

Leslie Sherlin is affiliated with the Harold Abel School of Psychology, Capella University, Minneapolis, MN.

Address correspondence to: Marco Congedo, MA, Brain Research and Psychophysiology Laboratory, Department of Psychology, The University of Tennessee, Knoxville, TN 37996-0900.

This research has been partially supported by Nova Tech EEG, Inc. Graphic material was created by computer software that is property of Nova Tech EEG, Inc.

Journal of Neurotherapy, Vol. 6(4) 2002

searcher or practitioner will have to face problems of a technical nature. One of these problems is the resampling of a digitized signal. In this article we introduce the problem and the most popular procedure to solve it, the natural cubic spline interpolation. While the article is introductory, the appendix treats the issue in a more formal way for the benefit of the interested reader.

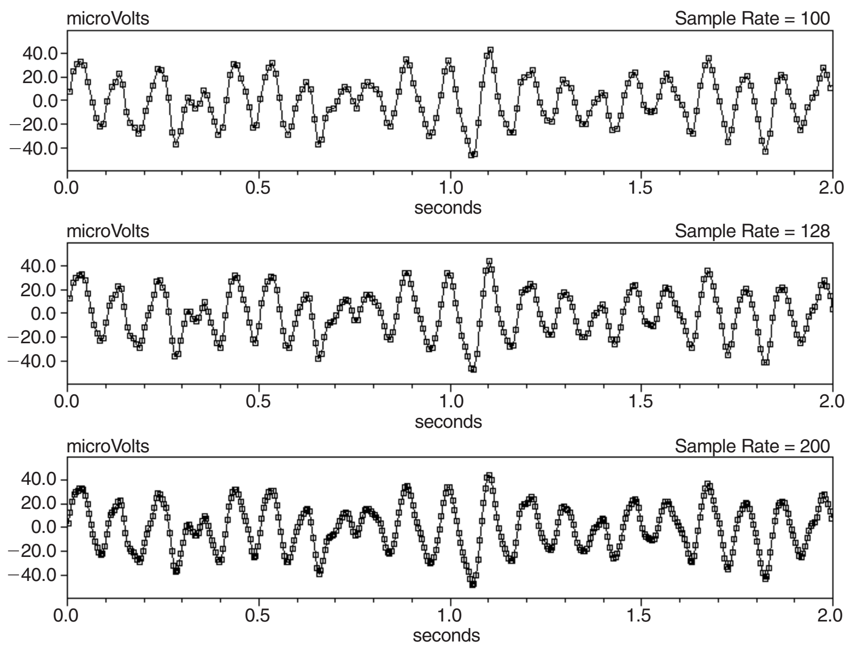
There are at least two situations in which the resampling of the EEG is useful. One is to ensure compatibility among EEG data acquisition devices and software for the comparison to EEG norms, better known as normative databases. The other is to customize the data analysis, or refine the frequency resolution. Unfortunately, in the EEG industry there is no standard for the sampling rate and file format. As a result, it is usually difficult and sometimes impossible to share EEG data and to analyze the data with diverse kinds of software. Virtually all data acquisition machines have their own format for data files. Unless the manufacturer provides the user with a file conversion for the ASCII format (the standard alphanumeric format), it is problematic to export the data to other software. Another difficulty is the disagreement concerning the "sampling rate" used. Currently the EEG signal is either acquired as an analog signal and then digitized, or it is acquired in digital form. The digitized EEG consists of equally spaced samples of the underlying signal. The number of samples per second is called the sample rate (SR). If the sample rate is sufficiently high, then the loss of information determined by the sampling process is negligible. More specifically, following the well-known Shannon's sampling theorem (Lynn & Fuerst, 1989), when digitizing a signal we are able to capture frequencies up to half the SR (folding frequency). The maximum frequency contained in an analog signal is called Nyquist frequency, and the minimum sample rate we need in order to represent this frequency in a digitized signal is called the Nyquist rate (half the Nyquist frequency). For example, the reason why the popular compact disc has a SR = 44,100 is because in this way one can reconstitute a sound up to its 22,050 Hertz (Hz) component, which is close enough to the highest human auditory threshold. Some EEG data acquisition machines (e.g., Lexicor) support sample rates that are multiples of 2 (e.g., 64, 128, 256 . . .) while others like Cadwell and Neuroscan support sample rates that are multiples of 10 (e.g., 100, 200 . . .). To submit EEG data to a normative database the sample rate has to be compatible with the sample rate used in the database. The diversity of database standards sometimes prevents this possibility. EEG resampling consists of converting the SR of a signal while leaving the signal intact. Resampling the EEG provides compatibility among data acquisition machines and software. For instance one can acquire and analyze EGG data at the desired sample rate, resample, and import the data into other software programs.

Another case in which EEG resampling is useful is when we want to customize the data analysis process. It is well known that Fast Fourier Transform (FFT) algorithms require data segment lengths (epic lengths) to be a power of 2 in length (usually 128, 256, or 512 samples). The Fourier coefficients allow estimates of amplitude or power of the signal at equally spaced discrete frequencies. In the case of an EEG signal the number of discrete frequencies is equal to the epoch length divided by two. The frequency resolution for a given EEG file equals the sample rate divided by the epoch length. For example, supposing the EEG file was recorded at 128 samples per second, and that each epoch of the file has 256 samples, the FFT will provide estimates at 256/2 = 128 discrete frequencies and the frequency resolution would be 128/256 = 0.5. Thus, the first estimation will be at 0.5 Hz; the second at 1 Hz; the third at 1.5, and so on up to the folding frequency (64 Hz). The reader can verify that there will be 128 discrete frequencies (Beauchamp, 1973; Brillinger, 1975). The frequency resolution can be adjusted by manipulating either the epoch length (if the software allows) or the sample rate (by resampling). The frequency resolution increases with longer epochs and smaller sample rates. Although a higher frequency resolution is undoubtedly desirable, it is not possible to increase the frequency resolutions ad libitum. The limits are dictated by the assumption of signal stationarity for the FFT (DeLurgio, 1998) and by the sampling theorem. The assumption of signal stationarity is less tenable for an FFT performed on longer epochs. The SR, as reported above, cannot be smaller than the Nyquist rate. Typically a frequency resolution of 0.5 (epoch length = 512 and SR = 256) or 0.25 (epoch length = 512 and SR = 128) is the smallest resolution that can be reasonably achieved.

EEG resampling is simple in principle. The EEG is a continuous signal constituted by oscillation of potential differences over time. In plotting the EEG, the abscissa is plotted as time and the ordinate plotted as voltage. Digitized samples are a sequence of number pairs: one for the time and the other for the voltage. Each sample is the instantaneous recording of voltage at equally spaced time intervals. Samples are plotted as points in two dimensional spaces. Connecting these points results in the EEG as it is typically displayed by EEG software (Figure 1).

In order to resample the EEG we have to specify the new interval to which a new set of x-coordinates corresponds and estimate the y-values (voltage) for the new samples. This is usually obtained by interpolating

FIGURE 1. Two seconds of EEG as recorded digitally from occipital site O2 in a 26-year-old woman. The chart in the middle is the original recording (Sample Rate = 128). Squares represent digital samples. The first and the third chart show the result of natural cubic spline interpolation resampling. The first chart shows the result of down sampling (from 128 SR to 100 SR). The third chart shows the result of up sampling (from 128 to 200).



the value of the function (signal) in the gap between known data-points (original samples). Cubic Spline interpolation is a special case of cubic polynomial interpolation. In addition to polynomial interpolation, spline interpolation requires the first derivative of the underlying function to be continuous at the known data-points. In other words, the interpolated function will not reconstruct artificial spikes, but will assume the function to be smooth, an assumption that suits extra-cranial EEG data that are to be processed by Fourier analysis (though it is not a good idea to perform morphological analysis on resampled data). Cubic denotes a polynomial model of the third order. The third order model is a suitable compromise between precision and simplicity of computations. The spline interpolation algorithm requires the second derivative at the two end-points to be specified. Usually these are unknown. A common choice is to set the second derivative at the end points to zero. If so, the procedure is called natural spline.

Figure 1 shows two seconds of data recorded with digital equipment from occipital site O2 in a 26-year-old woman. The original recording had a sampling rate of 128 and is displayed in the middle chart of Figure 1. The first chart in Figure 1 shows the same data after down sampling from SR = 128 to SR = 100. The last chart shows the same data after up sampling from SR = 128 to SR = 200. The signal is almost identical in the three charts, illustrating the little distortion introduced by the resampling method.

A few final comments on EEG resampling: up sampling does not pose any theoretical problem. However it is important to keep in mind that up sampling cannot increase the Nyquist Frequency (see above) since this is set during the data acquisition process and cannot be increased off-line. Additionally, we have seen that, keeping the epoch length fixed, up sampling reduces the frequency resolution. As such, up sampling is seldom of interest. On the other hand, down sampling requires care with the sampling theorem. For instance, if it is desired to down sample to 64 samples per second, the signal first should be low-pass filtered at 32 Hz to prevent aliasing. In this example we would increase the frequency resolution, but we would lose all frequency components above 32 Hz. As a final note, after resampling it is advisable to remove the end points or even a few samples at the end of the signal, since natural spline errors occur there more likely as compared to the middle portion of the signal. For this reason it is important to resample the entire EEG recording, and not to perform the process epoch-by-epoch.

REFERENCES

Beauchamp, K. G. (1973). *Signal processing using analog and digital techniques*. London: George Allen & Unwin.

Brillinger, D. R. (1975). *Time series. Data analysis and theory*. New York: Holt, Rinehart, and Winston.

DeLurgio, S. A. (1998). *Forecasting principle and applications*. New York: McGraw-Hill.

Devries, P. L. (1994). *A first course in computational physics*. New York: John Wiley & Sons.

Lynn, P. A., & Fuerst, W. (1989). *Introductory digital signal processing with computer applications*. New York: John Wiley & Sons.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1996). *The art of scientific computing in Fortran 77, Vol. 1 of Fortran Numerical Recipes (2nd Ed.)*. Cambridge University Press.

APPENDIX

Given the value of a function at a set of data points $x_1, x_2, \ldots, x_N$ with $x_1 < x_2 < \ldots x_{n-1} < x_n$, we are interested in estimating the value of the function at an arbitrary point in the interval $[x_1, x_n]$.

The interpolation scheme must use appropriate functional forms to mimic the function in a plausible way. There are many different classes of functional forms, however not all of them have continuous derivatives. When the continuity of derivatives are important, spline interpolation is usually a good choice since by construction they have the desired continuity.

The number of data points minus one is called the *order* of interpolation. Increasing the order does not usually increase the accuracy especially in the case of a widely used class of interpolating functions, the class of polynomials. This is largely due to the fact that added points cause superfluous wiggles between the tabulated values. One can alternatively use a local interpolation scheme using a finite number of nearest-neighbor points, but the price of this approach is having discontinuous derivatives. Now imagine that we have the tabulated points x_1, x_2, . . . and so on in the increasing order, and we are interpolating "locally" first in the region $x_1 < x < x_2$. As x varies from x_1 to x_2, the function and its derivatives will vary smoothly. As x increases beyond x_2, we want to interpolate in the region $x_2 < x < x_3$. Since now we have a different set of interpolation points, we have a different approximating polynomial. This is why this scheme suffers from a discontinuous change in the derivatives.

If the continuity of derivatives is a concern, once can use the spline interpolation scheme. The requirement that the first derivatives across boundaries between interpolation intervals be continuous, makes the spline interpolation more than a "local" interpolation technique. In other words, the nonlocality serves to guarantee global smoothness in the interpolated function up to some order of derivative.

Below we are going to give a simple outline of the cubic spline interpolation scheme.

Given a tabulated function $f(x_i)$, $i = 1, 2, \ldots, N$, let us consider a particular interval $x_j < x < x_{j+1}$. Now let us define the cubic interpolating function $p(x)$ as:

$$p(x) = a_j(x - x_j)^3 + b_j(x - x_j)^2 + c_j(x - x_j) + d_j \qquad (1)$$

Now assume for a moment that, in addition to the tabulated values $f(x_i)$, we also know the tabulated values of the function's second derivative $f''(x_i)$.

For avoiding cumbersome notation, let us introduce:

$$p_j = p(x_j);\ h_j = x_{j+1} - x_j;\ f_j = f(x_j) \tag{2}$$

The approximating polynomial should give exact results at the tabulated points, i.e., the end points of the interval:

$$p_j = f_j = d_j;\ p_{j+1} = a_j h_j^3 + b_j h_j^2 + c_j h_j + p_j \tag{3}$$

The first and second derivatives of the polynomial are:

$$p'(x) = 3a_j(x - x_j)^2 + 2b_j(x - x_j) + c_j \tag{4}$$

and

$$p''(x) = 6a_j(x - x_j) + 2b_j \tag{5}$$

The coefficients a_j and b_j in Equation (1) can be expressed in terms of the second derivative of the polynomial by writing Equation (5) at the points $x = x_j$ and $x = x_{j+1}$:

$$p''(x_j) = p_j'' = 2b_j \Rightarrow b_j = \frac{p_j''}{2} \tag{6}$$

$$p''(x_{j+1}) = p_{j+1}'' = 6a_j h_j + 2b_j \Rightarrow a_j = \frac{1}{6}\frac{p_{j+1}'' - p_j''}{h_j} \tag{7}$$

Using Equation (5) the coefficient c_j can now be written as:

$$c_j = \frac{p_{j+1} - p_j}{h_j} - \frac{h_j p_{j+1}'' + 2h_j p_j''}{6} \tag{8}$$

Once the coefficients a_j, b_j, c_j and d_j are determined one can construct the interpolating function, Equation (1), and its derivative, Equation (4). The second derivatives f_j'' can be determined with the use of Equation (4) by substituting the coefficients. We have $n-1$ interpolation intervals, hence $n-2$ boundary points. Enforcing the continuity of the p' at the boundary points yields $n-2$ equations. We need two more equations to obtain a unique solution for n unknown f_j'''s. These two are given by applying Equation (4) at the boundary points. All these linear equations can be put into a matrix form, which turns out to be tridiagonal:

$$\begin{pmatrix} 2h_1 & h_1 & & & & & \\ h_1 & 2(h_1+h_2) & h_2 & & & & \\ & h_2 & 2(h_2+h_3) & h_3 & & & \\ & & . & . & . & & \\ & & & & & & \\ & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & & h_{n-1} & 2h_{n-1} \end{pmatrix} \begin{pmatrix} p_1'' \\ p_2'' \\ p_3'' \\ . \\ . \\ p_{n-1}'' \\ p_n'' \end{pmatrix}$$

$$= \begin{pmatrix} 6\frac{p_2-p_1}{h_1} - 6p_1' \\ 6\frac{p_3-p_2}{h_2} - 6\frac{p_2-p_1}{h1} \\ 6\frac{p_4-p_3}{h_3} - 6\frac{p_3-p_2}{h_2} \\ . \\ . \\ 6\frac{p_n-p_{n-1}}{h_{n-1}} - 6\frac{p_{n-1}-p_{n-2}}{h_{n-2}} \\ -6\frac{p_n-p_{n-1}}{h_{n-1}} + 6p_n' \end{pmatrix} \tag{9}$$

Equation (9) defines the cubic spline interpolation problem. Note that the array in the right hand side of the equation has the first and last registers dependent on the first derivatives. If they are not known, they are usually chosen as zero. Splines obtained this way are called *the natural splines*.

For a more detailed yet practical discussion of the material, readers can consult Press et al. (1996) and DeVries (1994).