FEBRUARY 1987 VOL. 12, NO. 2

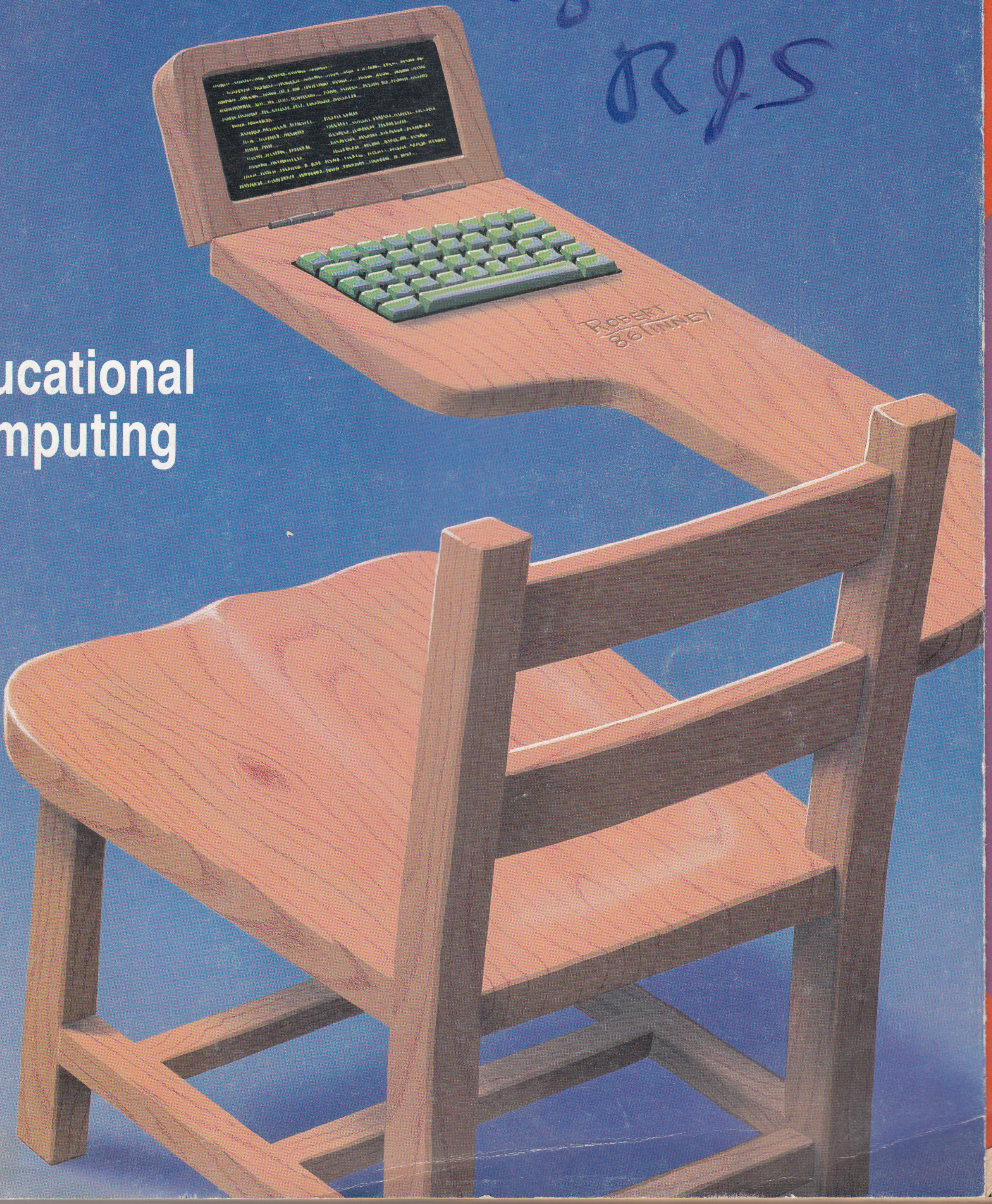
$3.50 IN UNITED STATES
$4.50 IN CANADA / £1.75 IN U.K.
A McGRAW-HILL PUBLICATION
0360-5280

BYTE

THE SMALL SYSTEMS JOURNAL®

Pg 137
RJS

Educational Computing



Robert J. Sciamanda

Another Approach to Data Compression

Use these BASIC programs to explore the Nyquist sampling theorem

Suppose you have a large quantity of data obtained by hardware measurements, software simulation, or any other method. The numbers may correspond to physical measurements or to values of a known or unknown mathematical function. Do you need to keep all this data? Or can you keep only samples of it and later reconstruct the original data set to within a specified accuracy, using the samples?

I've written a set of BASIC programs that answer these questions by using the Nyquist sampling theorem to find the lowest sampling rate that can be used to reconstruct the original data to a preset degree of accuracy.

The Nyquist sampling theorem says that a finite-bandwidth function can be reconstructed exactly from a set of sampled values:

$$f(n\tau),\ n = -\infty, \ldots, -1, 0, +1, \ldots, +\infty$$

so long as the sampling frequency, $1/\tau$, is greater than twice the highest frequency present in the Fourier spectrum of $f(t)$.

The reconstruction algorithm is given by the "Nyquist sum":

$$f(t) = \sum_{n=-\infty}^{+\infty} \frac{f(n\tau)\sin[(\pi/\tau)(t - n\tau)]}{(\pi/\tau)(t - n\tau)} .$$

Hardware implementations of Nyquist sampling and reconstruction are familiar; the hardware embodiment of the equation above is in fact the low-pass smoothing filter used in digital-to-analog conversion hardware.

In the case at hand, the frequency content of the original function is unknown. However, you can use the Nyquist sum equation to find a suitable value of τ (the sampling interval) through a trial-and-error approach, trying larger and larger values of τ until the observed error of the Nyquist sum exceeds a preset limit.

The BASIC Programs

There are five programs in all. Initially you must run them in sequence.

Listing 1 generates a file named DATA, which contains values from a typical function. The original data values are preceded by count N.

The program in listing 2 reads N+1 values from the file into the array A( ). The program then asks you to specify the accuracy to which this data must be reconstructed.

The software sampling interval L is then set equal to 2, and the Nyquist sum equation is used to reconstruct A( ) from the subset consisting of every Lth item in A( ). If this reconstruction is successful to within the specified accuracy, L is incremented by 1 and the reconstruction process is repeated.

With each iteration of this process, reconstruction is attempted from a subset of fewer data samples until the accuracy requirement fails. The original data count N, the largest successful L value, and the compressed data (every Lth item in A( )) are then written into the disk file CDATA.

Listing 3 reverses the process of listing 2, reconstructing the original data set to within the specified accuracy from the compressed data file CDATA.

The original data count, the sampling interval, and the compressed data are input from CDATA as N, L, and the array B( ), respectively. The Nyquist sum equation is then applied to reconstruct the original data set, which is written into the disk file RDATA.

RDATA is identical in format to the original data file DATA; that is, the first entry is the data count N, followed by the N+1 reconstructed data values.

Listing 4 provides an explicit check of the compression-reconstruction process. After running the reconstruction program (listing 3), run listing 4 to see a side-by-side comparison of the original and reconstructed data sets.

[Editor's note: *BYTE added listing 5, which plots the points from DATA and RDATA on the screen. The listing requires an IBM PC or compatible machine with BASICA and high-resolution graphics.*]

All listings (except 5) were written in Microsoft GW-BASIC for a Zenith Z-150. They should run without modification on IBM PC compatibles and with only minor changes on other computers equipped with BASIC and a disk drive.

Using the test data generated by listing 1 and electing an accuracy of 0.005, listing 2 (the compression algorithm) ran in 45 seconds on a Z-150 and found the largest successful sampling interval to be L = 3.

Will It Work with Real Data?

How much your real data can be compressed by this method—or whether it can be compressed at all—is a function both of your required accuracy and of the frequency content of the data. The more erratic the data variations, the higher the component frequencies and the higher the required sampling frequency.

An important factor contributing to the frequency content of the original data is the smoothness of the transitions to and from the value 0 at the beginning and end of the data domain. The value 0 is pre-

continued

Robert J. Sciamanda is a physicist at American Sterilizer Co. (2424 West 23rd St., AMSCO 620, Erie, PA 16514). Previously he taught physics at Gannon University and designed automated inspection systems for EG&G at the Idaho National Engineering Laboratory.

Listing 1: *A program to generate a data file for testing the Nyquist compression technique. The program evaluates the Gaussian function*

$$10e^{-(5 - .2i)^2}, \quad i = 0 \text{ to } 50.$$

```
10 OPEN "O",#1,"DATA" :REM Make a test data file by taking
20 PRINT#1,50         :REM 51 samples from a Gaussian
30 D=.2               :REM curve centered at i=25.
40 FOR I=0 TO 50
50 A=10*EXP(-(5-D*I)^2)
60 PRINT#1,A
70 NEXT I
80 CLOSE
```

Listing 2: *This program reads the sample data file called DATA, prompts the user for the required accuracy, and then determines whether the data can be compressed and reconstructed using the Nyquist sampling theorem. If possible, the compressed data is written to a file called CDATA.*

```
10 OPEN "I",#1,"DATA"
20 INPUT#1,N              :REM Get data count.
30 DIM A(N)
40 FOR I=0 TO N           :REM Get original data set.
50 INPUT#1,A(I)
60 NEXT I
70 CLOSE
80 INPUT "Enter desired accuracy ";E
90 FOR L=2 TO INT(N/2-.5)
100 W=3.141593/L
110 FOR I=1 TO N          :REM Reconstruct missing values.
120 IF I MOD L=0 THEN 190 :REM Branch at sampled values.
130 G=0
140 FOR J=0 TO N STEP L :REM The Nyquist sum.
150 M=W*(I-J)
160 G=G+A(J)*SIN(M)/M
170 NEXT J
180 IF ABS(G-A(I))>E THEN 210 :REM Sum done; test
     accuracy.
190 NEXT I     :REM If ok, reconstruct next value.
200 NEXT L     :REM Increment sampling interval.
210 L=L-1      :REM Highest successful sampling interval.
220 IF L>1 THEN 260 :REM L=1 means no compression
     possible.
230 PRINT "For an accuracy of +/-";E;"all of this data
     must be kept."
240 PRINT "No compressed data file (CDATA) will be
     generated."
250 END
260 OPEN "O",#1,"CDATA" :REM Create compressed data file.
270 PRINT#1,N,L :REM Write data count, sampling interval.
280 FOR J=0 TO N STEP L :REM Write compressed data set.
290 PRINT#1,A(J)
300 NEXT J
310 CLOSE
320 L$="th"
330 IF L=2 THEN L$="nd"
340 IF L=3 THEN L$="rd"
350 PRINT "Every ";L;L$;" data value has been kept in the
     compressed data file (CDATA)."
360 PRINT "The original data set can be reconstructed to
     an accuracy of +/-";E
```

Listing 3: *This program reads compressed data from CDATA and uses it to reconstruct the original data by applying the Nyquist sampling theorem.*

```
10 OPEN "I",#1,"CDATA" :REM Compressed data.
20 INPUT#1,N,L          :REM Get count, sampling inverval.
30 K=INT(N/L):DIM B(K)
40 FOR I=0 TO K         :REM Get compressed data.
50 INPUT#1,B(I)
60 NEXT I
70 CLOSE
80 OPEN "O",#1,"RDATA" :REM Create reconstructed data.
90 PRINT#1,N            :REM Write data count.
100 W=3.141593/L
110 FOR I=0 TO N        :REM Reconstruction
120 IF I MOD L = 0 GOTO 190 :REM Branch at sampled values.
130 G=0
140 FOR J=0 TO K        :REM The Nyquist sum.
150 M=W*(I-J*L)
160 G=G+B(J)*SIN(M)/M
170 NEXT J
180 GOTO 200   :REM Sum done; store this value.
190 G=B(I/L)
200 PRINT#1,G :REM Write reconstructed value to file.
210 NEXT I    :REM Go reconstruct next value.
220 CLOSE     :REM Done
230 PRINT "The reconstructed data file is RDATA"
```

Listing 4: *This program displays data from the original and reconstructed data files.*

```
10 OPEN "I",#1,"DATA"   :REM Original data file.
20 OPEN "I",#2,"RDATA"  :REM Reconstructed data file.
30 PRINT "  DATA          RDATA       Error"
40 IF EOF(1) THEN CLOSE: END
50 INPUT#1,A            :REM Get original data value.
60 INPUT#2,B            :REM Get reconstructed value.
70 ER=ABS(B-A)  :REM Calculate error.
80 PRINT USING "#.#####^^^^  #.#####^^^^  #.####";A,B,ER
90 GOTO 40
```

Listing 5: *This program plots the data from DATA and RDATA on the screen (high-resolution capability required). Note that the scale and offsets may differ for the two plots.*

```
10 SCREEN 0,0,0 :REM Text screen
20 SZ=4-INT(-(640+7)*200/32) :REM Size of graphics array.
30 DIM SC(SZ)          :REM To hold graphics screen.
40 VRES=200: HRES=640
50 REM
60 YES=(1=1)
70 NO=(1=0)
80 SS=NO                      : REM Screen not saved yet
90 FILES
100 LINE INPUT "Name the input file ";FI$
110 IF FI$=NU$ THEN END
120 OPEN FI$ FOR INPUT AS 1
130 INPUT #1,N
140 PRINT FI$; " contains ";N+1; "values"
150 INPUT #1,Y
160 MINY=Y: MAXY=Y
170 FOR K=1 TO N
180 INPUT #1,Y
190 IF Y>MAXY THEN MAXY=Y
```

continued

```
200 IF Y<MINY THEN MINY=Y
210 NEXT K
220 CLOSE
230 PRINT "Values range from ";MINY; "to "; MAXY
240 PRINT "Press any key to continue";
250 WHILE INKEY$=NU$: WEND
260 YSCALE=(VRES-1)/ABS(MAXY-MINY)
270 XSCALE=(HRES-1)/N
280 CLS
290 SCREEN 2  :REM Graphics screen
300 IF SS THEN PUT (0,0),SC :REM Restore screen if it has
     been saved previously.
310 REM
320 OPEN FI$ FOR INPUT AS 1
330 INPUT #1,N
340 INPUT#1,Y
350 PSET (0,VRES-1-(Y-MINY)*YSCALE)   :REM Plot 1st point.
360 FOR X=1 TO N
370 INPUT #1,Y
380 LINE -(X*XSCALE,VRES-1-(Y-MINY)*YSCALE) :REM Connect.
390 NEXT X
400 CLOSE
410 GET (0,0)-(639,199),SC
420 WHILE INKEY$=NU$: WEND   :REM Hold til key pressed.
430 SS=YES    :REM Screen has been saved.
440 SCREEN 0,0,0       :REM Go back to text screen.
450 GOTO 90
```

sumed for values "before" and "after" the N+1 data values. This is because in the reconstruction of each function value, the Nyquist sum equation requires the use of sampled values over the complete domain of the function, that is, from $n = -\infty$ to $n = +\infty$. When the series is truncated, as it must be in any practical implementation, the algorithm operates as if all the truncated sample values are 0.

In both hardware and software implementations, it is common practice to preprocess the data, using filtering and windowing techniques to limit the high-frequency content. Filtering eliminates or smooths out exceptional fluctuations or discontinuities in the data, while windowing techniques modulate the truncation of the data domain, providing smooth transitions to and from the value 0. Such techniques allow sampling at a lower frequency, but at the risk of doing violence to the data. The trade-off must be made on a case-by-case basis, assessing the extent to which the high-frequency content is artificial, that is, due to noise or introduced by truncation of the data domain.

For the Curious

It is instructive to use the algorithm of the Nyquist sum equation to reconstruct typical functions (truncated sinusoids, Gaussians, etc.) from sample sets extracted at a variety of sampling frequencies to exhibit the so-called aliasing effects of undersampling. These effects are best studied by comparing plots of the original and reconstructed functions.

To create data from a particular function, change listing 1. Note that the first value written to the file must be a count that is one less than the number of data values to follow. For instance, if the file contains 10 values, the count should be 9.

To control the sampling frequency without regard to Nyquist accuracy limitations, change listing 2, lines 80–100, to

```
80 PRINT "Sampling rate (2 to ";
    INT(N/2-.5); ")";
90 INPUT L: IF L < 2 OR L >
    INT(N / 2-.5) THEN 80
100 OPEN "O", #1, "CDATA"
```

and delete lines 110 through 260 and line 360.

Run the modified program to generate the sampled data file CDATA, then run listing 3 to attempt a reconstruction of the original data. Use listings 4 and 5 to compare the original and the reconstructed data.

Intellectually, the theory behind the Nyquist theorem is intriguing and elegant, and it requires only a knowledge of Fourier transform theory. Carson Chen's "An Introduction to the Sampling Theorem" (National Semiconductor Application Note 236) is a concise and readily available synopsis of Nyquist theory and includes an excellent bibliography. Chen's explanation is included in the *Nationa Semiconductor Data Conversion/Acquisition Databook* (Santa Clara, CA: Nationa Semiconductor Corp., 1984, page 12.46)

On the practical side, the Nyquist technique deserves more attention and development as a software calculation tool alongside traditional numerical interpolation. ■