A New Phase Unwrapping Algorithm

JOSÉ M. TRIBOLET, STUDENT MEMBER, IEEE

Abstract—A new phase unwrapping algorithm is proposed that combines the information contained in both the phase derivative and the principal value of the phase into an adaptive numerical integration scheme. This new algorithm has proven itself to be very reliable and it can be easily incorporated in standard homomorphic signal processors.

I. INTRODUCTION

HOMOMORPHIC signal processing, a nonlinear technique based on linear filtering operations on the complex cepstrum [1], was introduced in 1965 by Oppenheim and has been quite successfully applied to a variety of problems, namely, speech, seismic, and EEG data processing.

The computation of the complex cepstrum requires the definition of a continuous phase curve for the input data [1]. This phase curve is commonly called the unwrapped phase and it can be rigorously defined in terms of the integral of its derivative [1].

Conventional phase unwrapping has been done either by appropriate processing of the principal value of the phase (Schafer's algorithm) or by numerical integration of the phase derivative.

The algorithm we propose here is an adaptive numerical integration scheme that combines the information contained in both the phase derivative and the principal value of the phase, as follows: at each frequency, a set of permissible phase values is defined by adding integer multiples of 2π to the principal value of the phase. The selection of one of these values as the value of the unwrapped phase at this frequency is done with the help of a phase estimate formed by numerical integration of the phase derivative with a given step interval. This step interval is adapted until the phase estimate becomes arbitrarily close to one of the permissible phase values. This new algorithm has proven itself to be very reliable, at least as far as our experience with speech, seismic, and lunar data is concerned.

II. THE UNWRAPPED PHASE: THEORY

Let $x(n)$ be a sequence and $X(e^{j\omega})$ its Fourier transform. Let

$$\begin{aligned} X(e^{j\omega}) &= X_R(e^{j\omega}) + X_I(e^{j\omega}) \\ &= |X(e^{j\omega})| \exp\left[j \arg\left[X(e^{j\omega})\right]\right] \end{aligned} \tag{1}$$

where $|X(\cdot)|$ is the magnitude of the Fourier transform and $\arg[X(\cdot)]$ is its phase. Let $\hat{x}(n)$ be the complex cepstrum of $x(n)$ and $\hat{X}(e^{j\omega})$ its Fourier transform. Then

$$\hat{X}(e^{j\omega}) = \log X(e^{j\omega}) = \log|X(e^{j\omega})| + j \arg\left[X(e^{j\omega})\right] \tag{2}$$

where we have assumed that the complex logarithm is defined so that (2) represents a proper Fourier transform. In that case, the derivative of $\hat{X}(e^{j\omega})$ is well defined:

$$\frac{d\hat{X}(e^{j\omega})}{d\omega} = \frac{d \log X(e^{j\omega})}{d\omega} = \frac{dX(e^{j\omega})/d\omega}{X(e^{j\omega})}, \tag{3}$$

and it follows that the derivative of $\arg[X(e^{j\omega})]$ is given by the imaginary part of (3):

$$\frac{d \arg\left[X(e^{j\omega})\right]}{d\omega} = \frac{X_R(e^{j\omega})X_I'(e^{j\omega}) - X_I(e^{j\omega})X_R'(e^{j\omega})}{|X(e^{j\omega})|^2} \tag{4}$$

where the prime denotes $d/d\omega$ and

$$X'(e^{j\omega}) = X_R'(e^{j\omega}) + jX_I'(e^{j\omega}) = -j \text{ FT } \{n x(n)\}. \tag{5}$$

The phase $\arg[X(e^{j\omega})]$ can thus be unambiguously defined in terms of its derivative $\arg'[X(e^{j\omega})]$ as

$$\arg\left[X(e^{j\omega})\right] = \int_0^{\omega} \arg'\left[X(e^{j\eta})\right] d\eta \tag{6}$$

with initial conditions given by

$$\arg\left[X(e^{j0})\right] = 0. \tag{7}$$

Such phase functions are commonly called the *unwrapped phase* of $x(n)$.

In order for the complex logarithm in (2) to be properly defined, $\arg[X(e^{j\omega})]$ must be a continuous and odd function. From its definition in (6) and (7), we conclude that the unwrapped phase is indeed a continuous function; it will also be an odd function whenever the mean of the phase derivative $\langle \arg'[X(e^{j\omega})] \rangle$, defined as

$$\langle \arg'\left[X(e^{j\omega})\right] \rangle = \frac{1}{2\pi} \int_0^{2\pi} \arg'\left[X(e^{j\omega})\right] d\omega, \tag{8}$$

equals zero. When such is not the case, phase unwrapping must be followed by the removal of the linear phase component induced by $\langle \arg'[X(e^{j\omega})] \rangle$.

III. CONVENTIONAL PHASE UNWRAPPING: BASIC ISSUES

In spite of the fact that the unwrapped phase is very precisely defined by (6), one soon realizes that such an equation can never be exactly implemented in a digital computer. It is therefore of interest to investigate how to approximate it or how to alternatively define the unwrapped phase. Perhaps

Manuscript received March 5, 1976; revised October 12, 1976. This work was supported in part by the Advanced Research Projects Agency monitored by ONR under Contract N00014-75-C-0951-NR 049-308 and in part by the National Science Foundation under Grant ENG71-02319-A02.

The author is with the Department of Electrical Engineering and Computer Science, Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139.

the most straightforward approximation to (6) is to do numerical integration of the phase derivative on a set of uniformly spaced frequencies

$$\Omega = \{\omega_k = \frac{2\pi}{N} k, \quad k = 0,1,\cdots,N-1\} \tag{9}$$

since, through the use of the fast Fourier transform (FFT), it is possible to evaluate very efficiently $\arg'[X(e^{j\omega_k})]$, $\forall \omega_k \in \Omega$. This method depends very critically on the size of the integration step $2\pi/N$. Unfortunately, it is not possible to estimate *a priori* how large the FFT size N should be in order to accurately unwrap the phase. As shall be illustrated in Section V, numerical integration may lead to very significant errors, even for reasonably large FFT's.

Another approach that has been taken quite often is to compute the principal value of the phase

$$\mathrm{ARG}\,[X(e^{j\omega_k})] = \{\arg [X(e^{j\omega_k})]\}_{\mathrm{mod}\ 2\pi}, \quad \forall \omega_k \in \Omega \tag{10}$$

using inverse tangent routines on $X(e^{j\omega})$ and then unwrap by appropriately adding multiples of 2π to the principal value until the discontinuities induced by the modulo 2π operation are removed. This algorithm relies on the detection of such discontinuities, which is done by computing the difference between the principal values of the phase at two adjacent frequencies ω_{k-1} and ω_k. Whenever this difference is greater than a given threshold, we say that a discontinuity is present. This procedure will yield the unwrapped phase whenever the frequency sampling is fine enough so that the difference between any two adjacent samples of the unwrapped phase is always less than the prespecified threshold. In such a case, the unequivocal sorting between natural variations of the phase and the discontinuities induced by the modulo 2π operation is possible. An example where such sorting will be equivocal is the following. Consider a sequence $x(n)$ that has one zero very close to the unit circle at $\omega_z = (\omega_k + \omega_{k-1})/2$. The phase will then change by approximately π between ω_k and ω_{k-1}, increasing or decreasing depending on the position of the zero relative to the unit circle. This situation is illustrated in Fig. 1. At this sampling rate, the algorithm shall not be able to distinguish between a phase increase of π, which corresponds to the presence of a discontinuity induced by the modulo 2π operation between ω_{k-1} and ω_k, and a decrease of π, which corresponds to the absence of such discontinuity.

This situation often arises in practice, as shall be illustrated in Section V. This is one of the many examples one could think of where the information contained in the samples of the principal value of the phase is not enough to allow a reliable operation of this algorithm.

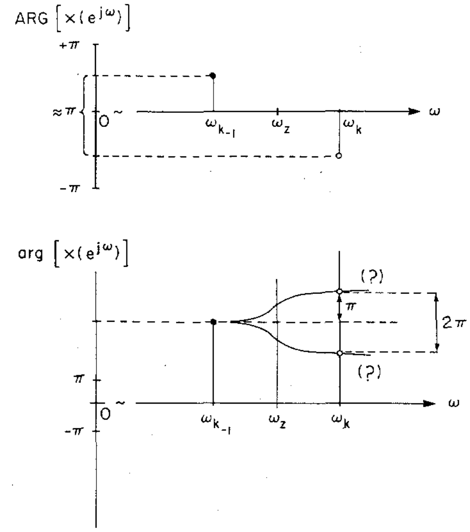

Fig. 1. Ambiguities in detecting discontinuities of the principal value of the phase.

IV. Phase Unwrapping by Adaptive Integration

A. Basic Idea

Let Ω_1 be an arbitrary frequency value and $\mathrm{ARG}\,[X(e^{j\Omega_1})]$ be the principal value of the phase at Ω_1.

The set of permissible phase values at Ω_1 is then given by

$$\{\mathrm{ARG}\,[X(e^{j\Omega_1})] + 2\pi l, \quad \forall l, \text{integer}\}. \tag{11}$$

The phase unwrapping problem amounts to determining the correct integer value $l_c(\Omega_1)$ such that

$$\arg [X(e^{j\Omega_1})] = \mathrm{ARG}\,[X(e^{j\Omega_1})] + 2\pi\, l_c(\Omega_1). \tag{12}$$

This is done through the use of numerical integration of the phase derivative. We shall adopt here the trapezoidal integration rule. Assuming the unwrapped phase to be known at a frequency $\Omega_0 < \Omega_1$, we shall define a phase estimate at Ω_1, $\tilde{\arg}\,[X(e^{j\Omega_1})|\Omega_0]$ by

$$\tilde{\arg}\,[X(e^{j\Omega_1})|\Omega_0] = \arg [X(e^{j\Omega_0})] + \frac{\Omega_1 - \Omega_0}{2}\,[\arg'[X(e^{j\Omega_0})] + \arg'[X(e^{j\Omega_1})]]. \tag{13}$$

Clearly, this estimate improves as the step interval $\Delta\Omega = \Omega_1 - \Omega_0$ becomes smaller. We shall define the phase estimate at Ω_1 to be consistent if it lies within a predefined distance of one of the permissible phase values at Ω_1, that is, if

$$\exists l_c(\Omega_1) \ni |\tilde{\arg}\,[X(e^{j\Omega_1})|\Omega_0] - \mathrm{ARG}\,[X(e^{j\Omega_1})] + 2\pi\, l_c(\Omega_1)| < \mathrm{THLD} < \pi. \tag{14}$$

The basic idea of this algorithm is thus to adapt the step size $\Delta\Omega$ until a consistent phase estimate is found. The resultant $l_c(\Omega_1)$ in (14) is used in (12) to form the unwrapped phase at Ω_1. This unwrapped phase is then used to form $\tilde{\arg}\,[X(e^{j\Omega_2})|\Omega_1]$, $\Omega_2 > \Omega_1$, and so on.

B. Implementation

The basic idea described previously requires the computation of the phase derivative and of the principal value of the phase on a set of nonequally spaced frequencies. For this algorithm to be practical, one must take full advantage of the FFT algorithm and reduce the number of extra discrete Fourier transform (DFT) computations to a reasonably small number. Let us denote by

$$\{\omega_k = (2\pi/N)k, \quad k = 0,1,\cdots,N-1\} \tag{15}$$

the set of uniformly spaced frequencies with interval $2\pi/N$

where $N = 2^M$ (or, in general, any highly composite number). The phase derivative and the principal value of the phase at these frequencies may then be computed using FFT's to evaluate the DFT's of $x(n)$ and $nx(n)$. At each ω_k, a phase estimate is initially formed by one-step trapezoidal integration, starting at ω_{k-1}. If the resultant estimate is not consistent, the adaptive integration scheme is applied within the interval $[\omega_{k-1}, \omega_k]$. The step size adaptation was carefully designed to minimize the number of extra DFT's required. The search for consistency is done by consecutively splitting the step interval in half. As the required phase derivatives and principal values are computed, they are stored in a stack fashion. As soon as a consistent estimate is found, the correspondent data are moved out of the stack to a register that holds the most recent consistent estimate of the phase at some frequency within $[\omega_{k-1}, \omega_k]$. New estimates are always formed by integrating from the most recent estimate to the frequency correspondent to the top of the stack. We present next a Fortran program along these lines.

C. A Fortran Program

See program shown below.

```
C                 TITLE:  CCEPS                 09/01/76              JMT
C
C                 PURPOSE:
C                           To compute the complex cepstrum cx(n)
C                           of a sequence x(n).
C
C                 AUTHOR:  Jose M. Tribolet
C                          room 36-387    M.I.T.
C                          Cambridge, MA  02139
C
C                 USAGE:
C                       Call CCEPS(X,NX,CXE,CXO,NCX,ISNX,ISFX,YR,YI)
C
C                 DESCRIPTION OF PARAMETERS:
C                        X:Real array containing the sequence x(n).
C                        NX:Length of x(n).
C                        CXE:Real array containing the even-indexed
C                            samples of cx(n).
C                            This array must be of dimension NCX/2+1.
C                        CXO:Real array containing the odd-indexed
C                            samples of cx(n).
C                            This array must be of dimension NCX/2+1.
C                        NCX:Number of complex cepstrum samples to be
C                            computed.Program will evaluate cx(n)for
C                            -NCX/2-1<n<NCX/2.Since radix 2 FFTs are
C                            used,NCX must be a power of 2.
C                            NCX must be greater or equal to NX
C                        ISNX:Integer value that is either -1 or +1
C                            depending on whether the data x(n) had
C                            to be sign reversed or not.
C                        ISFX:Integer value that indicates the amount
C                            of shifting that had to be performed on
C                            the data x(n).
C                        YR,YI:Two real auxiliary arrays,each of
C                            dimension NCX/2+1.
C
C                 SUBROUTINES CALLED:R2FFT(XR,XI,NPTS,ITYPE)
C
C                 R2FFT is a radix 2 FFT,with the
C                        following characteristics:
C                 XR,XI:Real and imaginary arrays to be transformed.
C                 NPTS:Number of complex points to be transformed,
C                      i.e.,the number of elements in XR or XI:must
C                      be an integer power of 2.
C                 ITYPE:Indicates type of FFT and direction:
C                 ITYPE=+1    FFT for real 2*NPTS-point sequence:
C                             Calling program supplies NPTS-point
C                             complex sequence formed by loading odd-
C                             -indexed samples of real sequence into
C                             XR and even-indexed samples of real se-
C                             quence into XI(index=1,2,...,2*NPTS).
C                             Program computes the first NPTS of the
C                             2*NPTS-point DFT of the 2*NPTS-point re-
C                             al sequence.Since the DFT of a real N-
C                             -point sequence is Hermitian symmetric
C                             mod N,the first transform point and the
C                             (NPTS+1)th transform point are both pure-
C                             ly real.By convention,the real part of the
C                             (NPTS+1)th transform point will be retur-
C                             ned in the imaginary part of the first
C                             transform point.The DFT is thus uniquely
C                             determined.
C                 ITYPE=-1    Inverse FFT for 2*NPTS-point Hermitian
```

```
C                                          symmetric complex DFT sequence:
C                                          Calling program supplies the first NPTS of
C                                          the input sequence with the convention that
C                                          the real part of the (NPTS+1)th point be
C                                          stored in the imaginary part of the first
C                                          point.The desired inverse DFT is purely
C                                          real:its odd-indexed points are given by
C                                          XR and its even-indexed points by XI
C                                          (indexed=1,2,...,2*NPTS).
C
C

      SUBROUTINE CCEPS(X,NX,CXE,CXO,NCX,ISNX,ISFX,YR,YI)
      DIMENSION X(1),CXE(1),CXO(1),YR(1),YI(1)
      DIMENSION ISK(13),SK1(13),SK2(13)
      INTEGER SP
      COMPLEX CO,C1,C2
      DOUBLE PRECISION A,B,C,D,E
      DATA TWOPI/6.283185307/

C
C             Initialization
C

      N=12
      L=2**N
      NPTS=NCX/2
      PI=TWOPI/2.
      H=TWOPI/FLOAT(2*NCX)
      H1=H/L
      THLD1=.8*PI
      THLD2=.5*PI
      ISNX=+1

C
C             Transform x(n)
C
C             Load CXE and CXO arrays.
C
      NHALF=NX/2
      DO 10 I=1,NHALF
      J=2*I-1
      CXE(I)=X(J)
10    CXO(I)=X(J+1)
      IF((NX-2*NHALF).EQ.0) GO TO 20
      NHALF=NHALF+1
      CXE(NHALF)=X(NX)
      CXO(NHALF)=0.
20    IF(NHALF.EQ.NPTS)GO TO 35
      NHALF=NHALF+1
C
      DO 30 I=NHALF,NPTS
      CXE(I)=0.
30    CXO(I)=0.
C
C               FFT
C
35    CALL R2FFT(CXE,CXO,NPTS,+1)
      CXE(NPTS+1)=CXO(1)
      CXO(1)=0.
      CXO(NPTS+1)=0.
C
C               Transform n*x(n)
C
C               Load YR and YI arrays.
C
      NHALF=NX/2
      DO 40 I=1,NHALF
      J=2*I-1
      YR(I)=(J-1)*X(J)
40    YI(I)=J*X(J+1)
      IF((NX-2*NHALF).EQ.0) GO TO 50
      NHALF=NHALF+1
      YR(NHALF)=(NX-1)*X(NX)
      YI(NHALF)=0.
50    IF(NHALF.EQ.NPTS)GO TO 65
      NHALF=NHALF+1
      DO 60 I=NHALF,NPTS
      YR(I)=0.
```

```
60        YI(I)=0.
C
C                 FFT
C
65        CALL R2FFT(YR,YI,NPTS,+1)
          YR(NPTS+1)=YI(1)
          YI(NPTS+1)=0.
          YI(1)=0.
C
C                 Check if sign reversal is required.
C
          If(CXE(1).LT.O.) ISNX=-1
C
C                 Compute logmagnitude:store in CXE
C                 Compute phase derivative:store in YR.
C                 Compute phase principal value:store in YI.
C                    For w=(TWOPI/NCX)*I,I=0,1,...NPTS.
C
          DVTMN=0.
          DO 70 I=1,NPTS+1
          A=CXE(I)
          B=CXO(I)
          C=YR(I)
          D=YI(I)
          E=A*A+B*B
          YR(I)=-((A*C+B*D)/E)
          DVTMN=DVTMN+YR(I)
C
          CXE(I)=DLOG(E)/2
          IF(ISNX.EQ.1)YI(I)=DATAN2(B,A)
          IF(ISNX.EQ.-1)YI(I)=DATAN2(-B,-A)
70        CONTINUE
          DVTMN=(2*DVTMN-YR(1)-YR(NPTS+1))/NCX
          ELPINC=2*DVTMN*H
C
C
C               Phase unwrapping by adaptive integration.
C
C


          PH=0.
          DO 150 I=2,NPTS+1
C
C               Store unwrapped phase in CXO.
C

          CXO(I-1)=PH

C
C               Form phase estimate at w=(TWOPI/NCX)*(I-1) by
C               trapezoidal integration with step size TWOPI/NCX.
C

          PHAINC=H*(YR(I)+YR(I-1))
          IF(ABS(PHAINC-ELPINC).GT.THLD1)GO TO 90
          PH=CXO(I-1)+PHAINC
C
C               Check consistency of estimate.
C

          A0=(PH-YI(I))/TWOPI
          A1=IFIX(A0)*TWOPI+YI(I)
          A2=A1+SIGN(TWOPI,A0)
          A3=ABS(A1-PH)
          A4=ABS(A2-PH)
          IF(A3.GT.THLD2.AND.A4.GT.THLD2) GO TO 90

C
C               Phase estimate was consistent.
C

          PH=A1
          IF(A3.GT.A4)PH=A2
          IF(ABS(PH-CXO(I-1)).GT.PI)GO TO 90
```

```
      GO TO 150
C
C           Phase estimate was not consistent:
C                   Adapt step size.
C
C           Initiate software stack.
C
90    SP=1
      ISK(1)=L+1
      SK1(1)=YI(I)
      SK2(1)=YR(I)
C
C           Initiate registers
C
      IB=1
      B1=CXO(I-1)
C
      B2=YR(I-1)
C
C           If software stack dimension does not
C           allow further step reduction,STOP.
C
100   IF((ISK(SP)-IB).GT.1) GO TO 110
      STOP 'FAILED'

C
C           Define intermediate frequencies (I.F.):
C                   w=(TWOPI/NCX)*(I-2+(K-1)/L)
C
110   K=(ISK(SP)+IB)/2
C
C           Compute DFTs of x(n) and n*x(n) AT I.F..
C

      A0=TWOPI*(I-2.+(K-1)/FLOAT(L))/FLOAT(NCX)
      C1=(0.,0.)
      C2=(0.,0.)
      DO 120 J=1,NX
      ARG=A0*(J-1)
      CO=CMPLX(COS(ARG),-SIN(ARG))*X(J)
      C1=C1+CO
120   C2=C2+CO*(J-1)
C
C           Compute phase derivative and the
C           principal value of the phase AT I.F..
C

      SP=SP+1
      ISK(SP)=K
      A=REAL(C1)
      B=AIMAG(C1)
      C=REAL(C2)
      D=AIMAG(C2)
      IF(ISNX.EQ.+1) SK1(SP)=DATAN2(B,A)
      IF(ISNX.EQ.-1) SK1(SP)=DATAN2(-B,-A)
      SK2(SP)=-(A*C+B*D)/(A*A+B*B)

C
C           Evaluate estimate at I.F.
C
130   DELTA=H1*(ISK(SP)-IB)
      PHAINC=DELTA*(B2+SK2(SP))
      IF(ABS(PHAINC-DELTA*2*DVTMN).GT.THLD1)GO TO 100
      PH=B1+PHAINC
C
C           Check consistency of estimate at I.F.
C
      A0=(PH-SK1(SP))/TWOPI
      A1=IFIX(A0)*TWOPI+SK1(SP)
      A2=A1+SIGN(TWOPI,A0)
      A3=ABS(A1-PH)
      A4=ABS(A2-PH)
      IF(A3.LT.THLD2.OR.A4.LT.THLD2)GO TO 140
```

```
C
C             Estimate was not consistent:Reduce step size.
C
C
        GO TO 100
C
C             Estimate was consistent:Update registers.
C
140     PH=A1
        IF(A3.GT.A4)PH=A2
        IF(ABS(PH-B1).GT.PI)GO TO 100
        IB=ISK(SP)
        B1=PH
        B2=SK2(SP)
        SP=SP-1

C
C             When software stack is empty,the unwrapped
C             phase at w= TWOPI*(I-1)/NCX is held in the
C             B1 register.
C

        IF(SP.NE.0) GO TO 130
        PH=B1

C
C             End of step size adaptation.
C
150     CONTINUE
C
C
C             End of Phase Unwrapping
C
C
C             Remove Linear Phase Component
C

        ISFX=(ABS(PH*2/TWOPI)+.1)
        IF(PH.LT.0.0) ISFX=-ISFX
        H=PH/NPTS
        DO 160 I=2,NPTS
160     CXO(I)=CXO(I)-H*(I-1)
C
C             Compute cx(n):IFFT.
C

        CXO(1)=CXE(NPTS+1)
        CALL R2FFT(CXE,CXO,NPTS,-1)
        RETURN
        END
```

V. An Example

Consider the signal shown in Fig. 2. In Fig. 3 its phase derivative is depicted, corresponding to a FFT size of $N = 2048$, which amounts to four times the data length. Let us, for example, focus on the low frequencies where large spikes of the phase derivative occur. Table I contains the first 25 samples of the phase derivative (scaled by π/N), the principal value of the phase, and the unwrapping results as given by numerical integration, Schafer's algorithm (with a threshold set for 4.5), and adaptive integration.

Note that both the numerical integration and the adaptive integration results progress with a similar trend, although the accumulation of error in the numerical integration estimate, noticeable after the occurrence of the large derivative spike at sample $k = 21$, causes a significant deviation of these estimates from the set of permissible estimates as previously defined. This deviation will be propagated from this point to higher frequencies, leading to an unacceptable result for the unwrapped phase.

Schafer's algorithm has also failed to successfully unwrap the phase. For example, at sample $k = 10$, it actually predicts an increase in the phase from -13.86 to -9.71 against the evidence offered by the phase derivative at that point. The same happens at samples $k = 15$ and $k = 21$. The results of adaptive integration are shown in Fig. 4 after the removal of the linear phase component. The unwrapped phase agrees everywhere with the information available. The phase values are selected from the permissible phase values at each frequency and they do follow the trend predicted by the phase derivative.

VI. Final Comments

Phase unwrapping is, as stated in this paper, a well-defined operation. Any stable signal $x(n)$ without zeros on the unit

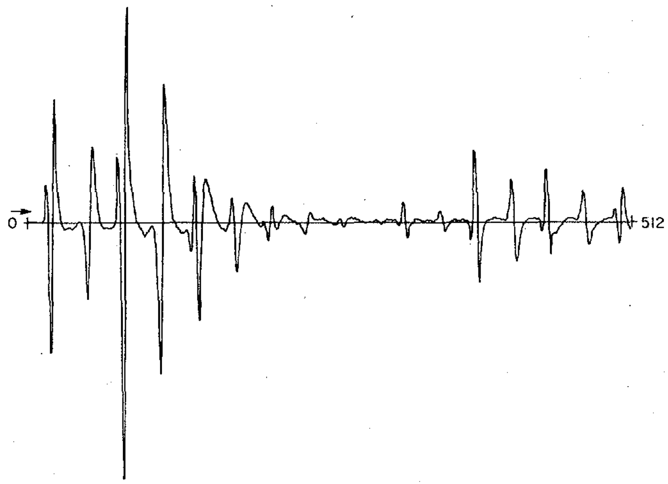

Fig. 2. A signal $x[n]$.

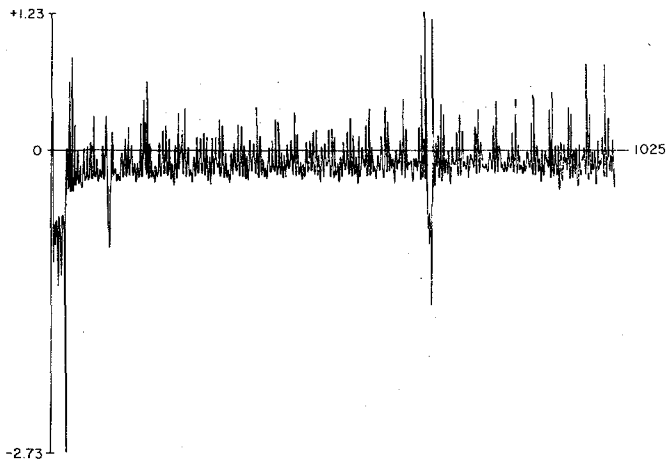

Fig. 3. Phase derivative $\arg'[X(e^{j\omega})]$ scaled by π/N for $\omega = (2\pi/N)k$, $k = 0, 1, \cdots, (N/2) + 1, N = 2048$.

circle will have a unique unwrapped phase $\arg[X(e^{j\omega})]$. The phase unwrapping method proposed here performs significantly better than conventional methods.

This method has been used in the homomorphic analysis of speech and seismic data with encouraging results. It is hoped that it will contribute to further research on the use of homomorphic signal processing.

References

[1] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

TABLE I
COMPARISON OF DIFFERENT PHASE UNWRAPPING TECHNIQUES

$\omega_k = \frac{2\pi}{N} k$	Phase Derivative (Scaled by π/N)	Principal Value of The Phase	Unwrapping Results		
			Numerical Integration	Schafer's Algorithm THLD=4.5	Adaptive Integration
0	- .86	.00	0.0	.00	.00
1	- .76	-1.64	- 1.62	- 1.64	- 1.64
2	- .67	-3.06	- 3.06	- 3.06	- 3.06
3	- .70	1.86	- 4.43	- 4.42	- 4.42
4	- .86	.31	- 6.00	- 5.97	- 5.97
5	- .87	1.49	- 7.73	- 7.77	- 7.77
6	- .69	-3.03	- 9.28	- 9.31	- 9.31
7	- .64	1.95	-10.61	-10.61	-10.61
8	- .74	.61	-11.99	-11.95	-11.95
9	-1.24	-1.30	-13.96	-13.86	-13.86
10	- .80	2.85	-16.00	- 9.71	-16.00
11	- .62	1.48	-17.42	-11.08	-17.37
12	- .60	.26	-18.65	-12.30	-18.58
13	- .66	- .98	-19.91	-13.55	-19.83
14	- .94	-2.50	-21.51	-15.07	-21.35
15	-1.14	1.45	-23.59	-11.11	-23.68
16	- .74	- .33	-25.46	-12.90	-25.47
17	- .64	-1.69	-26.84	-14.26	-26.82
18	- .61	-2.93	-28.09	-15.50	-28.07
19	- .59	2.15	-29.28	-16.70	-29.25
20	- .64	.96	-30.50	-17.88	-30.45
21	-2.73	-1.54	-33.87	-20.39	-32.96
22	- .58	2.52	-37.19	-16.32	-35.18
23	- .49	1.47	-38.38	-17.38	-36.23
24	- .45	.52	-39.31	-18.32	-37.17

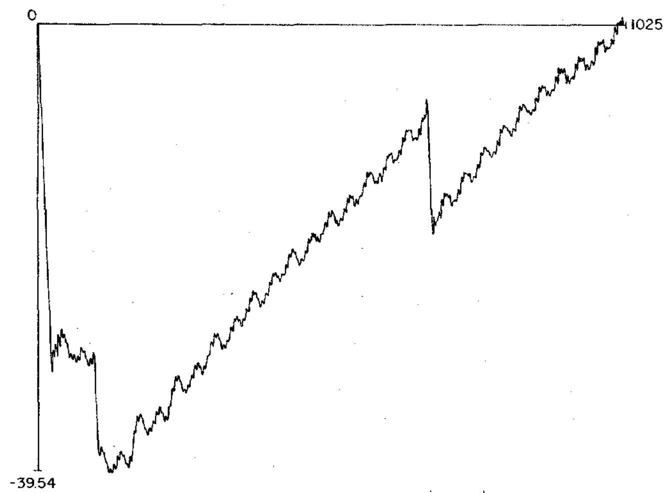

Fig. 4. Unwrapped phase $\arg[X(e^{j\omega})]$ after the removal of the linear phase component.