High-Speed Modem Algorithms 13

13.1 OVERVIEW

In high-speed data communication systems, there often arises the need for digital signal processing techniques. In the implementation of medium-speed (up to 2400 bps) to high-speed (4800 bps and higher) modems, certain effects of the limited-bandwidth communications channel (typically a voice-band telephone line) present themselves as obstacles. The most notable of these effects is intersymbol interference, which is the "smearing together" of the transmitted symbols over a time-dispersive channel (Lucky, et al, 1968). This effect is a problem in virtually all pulse-modulation systems, including pulse-amplitude modulation (PAM), frequency-shift keying (FSK), phase-shift keying (PSK), and quadrature-amplitude modulation (QAM) systems.

The basic action in most methods of reducing the effects of intersymbol interference is to pass the received signal through a filter that approximates the inverse transfer function of the communications channel; this process is called equalization. The implementation of an equalizer usually depends upon the speed of the modem. For medium-speed modems (generally PSK) "compromise" equalization is often adequate. Compromise equalization is performed using a short transversal filter with fixed coefficients that compensate for a wide range of channel characteristics. High-speed modems (generally QAM) usually require adaptive equalization using an adaptive filter to compensate for the excessively wide range of channel characteristics encountered in the switched telephone network (Qureshi, 1982).

13.2 SP COMPLEX-VALUED TRANSVERSAL FILTER

In the implementation of PSK and QAM modems, two double-sideband suppressed-carrier AM signals are sent by the transmitter and separated at the receiver. Orthogonal (quadrature) carrier signals are used for modulation and demodulation. It is customary to represent the in-phase and quadrature components of the received signal as the real and imaginary parts of a complex signal. Thus, the equalizer will operate upon this complex signal in order to reduce the effects of intersymbol interference. In practice, the equalizer may be inserted either before (passband equalization) or after (baseband equalization) the demodulation of the received signal (Qureshi, 1982).

The subroutine shown in Listing 13.1 presents an FIR filter routine for complex-valued data and coefficients that could be used to implement an equalizer. This routine implements the same sum-of-products operation as the nonadaptive (fixed-coefficient) FIR filter presented in Chapter 5; it has been modified to operate upon complex values. The filter is described by the equation on the next page.

$$y(n) = \sum_{k=0}^{N-1} h_k \, x(n-k)$$

The first loop, *realloop*, computes the real output by computing the sum of products of the real data values and the real coefficients, and subtracting the sum of products of the imaginary data values and the imaginary coefficients. The second loop, *imagloop*, is similar in that it computes the imaginary output as the sum of products of the real data values and the imaginary coefficients, added to the sum of products of the imaginary data values and the real coefficients. The outputs in both cases are rounded and conditionally saturated.

```
.MODULE cfir_sub;

{  Single-Precision Complex FIR Filter Subroutine

   Calling Parameters
        I0 —> Oldest data value in real delay line (Xr’s)
        L0 = filter length (N)
        I1 —> Oldest data value in imaginary delay line (Xi’s)
        L1 = filter length (N)
        I4 —> Beginning of real coefficient table (Hr’s)
        L4 = filter length (N)
        I5 —> Beginning of imaginary coefficient table (Hi’s)
        L5 = filter length (N)
        M0,M4 = 1
        AX0 = filter length minus one (N-1)
        CNTR = filter length minus one (N-1)
```

```
   Return Values
        I0 —> Oldest data value in real delay line
        I1 —> Oldest data value in imaginary delay line
        I4 —> Beginning of real coefficient table
        I5 —> Beginning of imaginary coefficient table
        SR1 = real output (rounded and conditionally saturated)
        MR1 = imaginary output (rounded and conditionally saturated)

   Altered Registers
        MX0,MY0,MR,SR1

   Computation Time
        2 × (N-1) + 2 × (N-1) + 13 + 8 cycles

   All coefficients and data values are assumed to be in 1.15 format.
}

.ENTRY  cfir;

cfir:   MR=0, MX0=DM(I1,M0), MY0=PM(I5,M4);
        DO realloop UNTIL CE;
             MR=MR-MX0*MY0(SS), MX0=DM(I0,M0), MY0=PM(I4,M4);    {Xi × Hi}
realloop:    MR=MR+MX0*MY0(SS), MX0=DM(I1,M0), MY0=PM(I5,M4);    {Xr × Hr}
        MR=MR-MX0*MY0(SS), MX0=DM(I0,M0), MY0=PM(I4,M4);         {Last Xi × Hi}
        MR=MR+MX0*MY0(RND);                                      {Last Xr × Hr}
        IF MV SAT MR;
        SR1=MR1;                                                 {Store Yr}
        MR=0, MX0=DM(I0,M0), MY0=PM(I5,M4);
        CNTR=AX0;
        DO imagloop UNTIL CE;
             MR=MR+MX0*MY0(SS), MX0=DM(I1,M0), MY0=PM(I4,M4);    {Xr × Hi}
imagloop:    MR=MR+MX0*MY0(SS), MX0=DM(I0,M0), MY0=PM(I5,M4);    {Xi × Hr}
        MR=MR+MX0*MY0(SS), MX0=DM(I1,M0), MY0=PM(I4,M4);         {Xr × Hi}
        MR=MR+MX0*MY0(RND);                                      {Xi × Hr}
        IF MV SAT MR;                                            {MR1=Yi}
        RTS;
.ENDMOD;
```

Listing 13.1 Single-Precision Complex FIR Filter

13.3 COMPLEX-VALUED STOCHASTIC GRADIENT

As mentioned previously, non-adaptive or compromise equalization is usually only adequate in medium-speed modems. High-speed modems require the equalizer coefficients to be adapted because of changing channel characteristics. In fact, even many 2400-bps modems incorporate adaptive equalization.

Although many adaptive filtering algorithms exist, virtually all adaptive equalizers in high-speed modems utilize the stochastic gradient (SG) algorithm (described in Chapter 5). This is primarily because it generally provides adequate performance and requires the least computation for a given filter order as compared to the other adaptive algorithms. Using the SG algorithm, filter coefficients at time T, $c_j(T)$, are adapted through the following equation:

$$c_j(T + 1) = c_j(T) + ß e_c(T)\, y^*(T - j + 1)$$

In this equation, $e_c(T)$ is the estimation error formed by the difference between the signal it is desired to estimate, d(T), and a weighted linear combination of the current and past input values y(T).

$$e_c(T) = d(T) - \sum_{j=1}^{n} c_j(T)\, y(T - j + 1)$$

The value $y(T - j + 1)$ represents the past value of the input signal “contained” in the jth tap of the transversal filter. For example, y(T), the present value of the input signal, corresponds to the first tap and $y(T - 42)$ corresponds to the forty-third filter tap. The step size ß controls the “gain” of the adaptation.

The coefficients are usually adapted during some training period after connection has been established. This involves the transmission of some known training sequence to the modem, during which time the equalizer adapts its coefficients according to a synchronized version of the received training sequence. Upon completion of the training period, slight variations in the channel characteristics may be tracked by performing the adaptation based on the estimate of the received symbol. This is referred to as decision-directed adaptation, and in some cases it is relied upon to perform the initial adaptation as well.

A subroutine for performing adaptation of complex FIR filter coefficients according to the stochastic gradient algorithm is given in Listing 13.2. In this subroutine, the cache memory is utilized very effectively, since four program memory accesses are made each time through the loop.

```
.MODULE csg_sub;

{  Single-Precision Complex SG Update Subroutine

   Calling Parameters
        I0 —> Oldest data value in real delay line                  L0 = N
        I1 —> Oldest data value in imag delay line                  L1 = N
        I4 —> Beginning of real coefficient table                   L4 = N
        I5 —> Beginning of imag coefficient table                   L5 = N
        MX0 = real part of Beta × Error
        MX1 = imag part of Beta × Error
        M0,M5 = 1
        M4=0
        M1= -1
        CNTR = Filter length (N)

   Return Values
        Coefficients updated
        I0 —> Oldest data value in real delay line
        I1 —> Oldest data value in imaginary delay line
        I4 —> Beginning of real coefficient table
        I5 —> Beginning of imaginary coefficient table

   Altered Registers
        MY0,MY1,MR,SR,AY0,AY1,AR

   Computation Time
        6 × N + 10 cycles

   All coefficients and data values are assumed to be in 1.15 format.
}
```

(listing continues on next page)

```
.ENTRY  csg;

csg:    MY0=DM(I0,M0);                          {Get Xr}
        MR=MX0*MY0(SS), MY1=DM(I1,M0);          {Er × Xr, get Xi}
        DO adaptc UNTIL CE;
           MR=MR+MX1*MY1(RND), AY0=PM(I4,M4);   {Ei × Xi, get Hr}
           AR=AY0-MR1, AY1=PM(I5,M4);           {Hr-Er × Xr+Ei × Xi, get Hi}
           PM(I4,M5)=AR, MR=MX1*MY0(SS);        {Store Hr, Er × Xi}
           MR=MR-MX0*MY1(RND), MY0=DM(I0,M0);   {Ei × Xr, get Xr}
           AR=AY1-MR1, MY1=DM(I1,M0);           {Hi-Er × Xi-Ei × Xr, get Xi}
adaptc:    PM(I5,M5)=AR, MR=MX0*MY0(SS);        {Store Hi, Er × Xr}
        MODIFY(I0,M1);
        MODIFY(I1,M1);
        RTS;
.ENDMOD;
```

Listing 13.2 Single-Precision Complex Stochastic Gradient

13.4 EUCLIDEAN DISTANCE

In the receiver of a high-speed modem, some method must be established for determining to which values from the space of possibilities the real and imaginary parts of the received sample correspond. In some QAM modems with large signal constellations, this can be a rather non-trivial process. For example, the CCITT V.29 standard calls for a 16-point signal constellation. One means of determining the value of samples is the Euclidean distance measure. This method involves computing the distance (error) between the received sample value and all possible candidates for the transmitted sample, given the signal constellation. The error is given by the following equation:

$$e(j) = ((x_r - c_r(j))^2 + (x_i - c_i(j))^2)^{1/2}$$

In this equation, the error e(j) is the distance between the received signal value, *x*, and the jth signal constellation value, *c*, in the real-imaginary plane. The value of *j* for which e(j) is minimum then selects the constellation point.

A subroutine for computing the Euclidean distance is shown in Listing 13.3. The *ptloop* loop is executed once for each point in the given signal constellation. The (squared) distance between *x* and each point is computed. AF is loaded with this value if it is less than the previous minimum, and the index corresponding to that constellation value (obtained from the current CNTR value) is stored in SI. After all distances have been computed, SI contains the index of the point that corresponds to the minimum e(j). This index can be used to select the constellation value.

```
.MODULE dist_sub;

{  Euclidean Distance Subroutine

   Calling Parameters
        I1 —> Start of constellation (C) table
        AX0 contains Xr
        AX1 contains Xi
        L1 = length of constellation table
        M0 = 1
        M1 = -1
        CNTR = length of constellation table

   Return Values
        SI contains the decision index j
        AF contains the minimum distance (squared)
        I1 —> Beginning of constellation table

   Altered Registers
        AY0,AY1,AF,AR,MX0,MY0,MY1,MR,SI

   Computation Time
        10 × N + 5 (maximum)
}
.ENTRY  dist;

dist:   AY0=32767;                      {Init min distance to largest possible value}
        AF=PASS AY0, AY0=DM(I1,M0);     {Get Cr}
        DO ptloop UNTIL CE;
           AR=AX0-AY0, AY1=DM(I1,M0);   {Xr-Cr, Get Ci}
           MY0=AR, AR=AX1-AY1;          {Copy Xr-Cr, Xi-Ci}
           MY1=AR;                      {Copy Xi-Ci}
           MR=AR*MY1(SS), MX0=MY0;      {(Xi-Ci)², Copy Xr-Cr}
           MR=MR+MX0*MY0 (RND);         {(Xr-Cr)²}
           AR=MR1-AF;                   {Compare with previous minimum}
           IF GE JUMP ptloop;
           AF=PASS MR1;                 {New minimum if MR1<AF}
           SI=CNTR;                     {Record the constellation index}
ptloop:    AY0=DM(I1,M0);
        MODIFY(I1,M1);                  {Point back to beginning of table}
        RTS;
.ENDMOD;
```

Listing 13.3 Euclidean Distance

13.5 REFERENCES

Lucky, R. W.; Salz, J.; and Weldon, E. J., Jr. 1968. *Principles of Data Communication*. New York: McGraw-Hill.

Qureshi, S. U. H. 1982. Adaptive Equalization. *IEEE Communications*. March 1982. P. 9-16.