NEUVIEME COLLOQUE SUR LE TRAITEMENT DU SIGNAL ET SES APPLICATIONS

NICE du 16 au 20 MAI 1983

FAST DESIGN OF WAVESHAPING FILTERS FOR SEISMIC SIGNAL DECONVOLUTION

D.Manolakis*, G.Carayannis**, N.Kalouptsidis*** and C.Halkias*

RESUME

Cet article présente des algorithmes rapides pour dessiner des"least squares shaping" et "spiking" filters. Ces filters sont utilisés à la deconvolution des signaux sismiques pour augmenter la resolution des sismogrammes par l'élimination des reflections multiples des périodes courtes ou longues. Ils sont ainsi utilisés trop par l'industrie d'exploration de pétrole dans la terre ou la mer. La necessité des algorithmes rapides est très grande, à cause de la quantité excessive des traces sismiques, qui sont ramassées par les programmes d'exploration.
L'algorithme le plus efficace est l'algorithme de Simpson, qui est developé exclusivement en cas des signaux stationnaires (méthode d'auto-corrélation). Les algorithmes nouveaux, qui sont presentés ici, sont plus rapides que l'algorithme de Simpson, et permettent un calcul récursif de l'erreur. Une méthode rapide est aussi presentée pour determiner le filtre avec la distance optimale de prévision-Un nombre de routines FORTRAN qui realisent les algorithmes proposés,sont finallement presentées.

SUMMARY

This paper deals with fast algorithms for the design of least-squares shaping and spiking filters. These filters are extensively used for seismic signal deconvolution to increase the resolution of seismograms through the elimination of multiple reflections with either short or long periods. Thus they are universally used by the petroleum exploration industry to attenuate ghost reflections and reverberations both on land and sea. Due to the tremendous amount of data collected in seismic exploration programs, fast algorithms for the computation of waveshaping filters are extremely important. The most efficient algorithm currently in use is the widely known Simpson's sideways recursion developed exclusively for the case of stationary signals (autocorrelation method). In this paper we provide new algorithms featured by increased speed (compared to Simpson's algorithm) and a recursive error computation capability. In addition a fast scheme for determining the predictor with the optimum prediction distance is given. Finally a set of FORTRAN routines, which realize the introduced algorithms, is included.

* National Technical University of Athens, Dept. of Electrical Engineering,42 Patission Ave.,Athens 147, Greece.

** Council of Europe, 67006-Strasbourg, France.

*** University of Athens, Dept.of Electronics, 104 Solonos St., Athens 144, Greece.

FAST DESIGN OF WAVESHAPING FILTERS FOR SEISMIC SIGNAL DECONVOLUTION

1. INTRODUCTION

Shaping filters, spiking filters and l-steps ahead predictors are extensively used in seismic signal processing. As it is well-known a major difficulty in the interpretation of seismograms is the lack of resolution due to the smearing of the input wavelet and/or the presense of multiple reflections /1/, /2/.

One way to increase the resolution in seismic recordings is the use of shaping filters, which are FIR Wiener filters designed by parametric Least Squares methods /1/. These filters "try" to convert the smeared wavelet into a more sharp form which will allow the resolving of overlapping seismic signals. Obviously, the ideal desired shape would be the unit sample sequence ("spike"). Shaping filters designed to convert a given wavelet into a Delta sequence are known as spiking filters /1/. Moreover, l-steps ahead predictors provide an effective deconvolution operator for the elimination of multiple reflections, with either short or long periods, from the seismograms.Thus they are universally used by the petroleum exploration industry to attenuate ghost reflections and reverberations both on land and sea /1/, /2/.

As a criterion of performance in FIR Wiener filtering and prediction is used the minimum value of the squared error which is a function of both the filter's order p and the lag l between the input signal and the desired response.

In this paper we present FORTRAN routines for the efficient design of shaping filters. These programs realize the fast algorithms introduced in /3/ and are faster than corresponding routines available in /4/. For a given value of order p supply the shaping filter corresponding to the value of l attaining the minimum total error (optimum lag filter). A routine for the design of optimum l-steps ahead predictors is also included. The design of spiking filters is treated in /3/, where additional information about the various algorithms can be found. The surboutine listings are given in the Appendix.

2. SHAPING FILTERS

The time domain input-output relationship for a FIR filter, of order p, is given by

$$y(n) = -\sum_{j=1}^{p} c_j\, x(n+1-j) = -\underline{c}_p^t\, \underline{x}(n) \tag{1}$$

where

$$\underline{c}_p = [\, c_1 \; c_2 \; \cdots \; c_p \,]^t \tag{2}$$

and

$$\underline{x}(n) = [\, x(n) \; x(n-1) \; \cdots \; x(n-p+1) \,]^t \tag{3}$$

Suppose that x(n) is a finite duration wavelet having length N. Then the convolution y(n) of c_j, x(n) can have at most N+p-1 nonzero terms. Writing (1) for n= =0,1,...., N+p-2 results to

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(p-1) \\ \vdots \\ y(N-1) \\ \\ y(N+p-1) \end{bmatrix} = - \begin{bmatrix} x(0) & & & O \\ x(1) & x(0) & & \\ \vdots & & \ddots & \\ x(p-1) & \cdots & & x(0) \\ \vdots & & & \vdots \\ x(N-1) & \cdots & & x(N-p) \\ & & \ddots & \vdots \\ O & & & x(N-1) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{p-1} \end{bmatrix} \tag{4}$$

or more compactly

$$\underline{y} = -X\, \underline{c}_p \tag{5}$$

Let

$$\underline{z} = [\, z(0) \; z(1) \; \cdots \; z(N+p-2) \,]^t \tag{6}$$

be a desired response signal. Then the error vector is

$$\underline{e} = \underline{z} - \underline{y} = \underline{z} + X\,\underline{c}_p \tag{7}$$

where

$$\underline{e} = [\, e(0) \; e(1) \; \cdots \; e(N+p-2) \,]^t \tag{8}$$

Minimization of the total error

$$E_p = \underline{e}^t \underline{e} \tag{9}$$

leads to the following set of normal equations /1/,/3/

$$R_p\, \underline{c}_p = -\underline{d}_p \tag{10}$$

where

$$R_p = X^t X \tag{11}$$

$$\underline{d}_p = X^t \underline{z} \tag{12}$$

The matrix R_p is symmetric Toeplitz with elements given by

$$r_j = \sum_{n=0}^{N-1-j} x(n)\,x(n+j) \quad , j=0,1,\cdots,p-1 \tag{13}$$

The elements of $\underline{d}_p$ are the crosscorrelation samples

$$r_{xz}(j) = \sum_{n=0}^{N-1-j} x(n)\,z(n+j) \quad , j=0,1,\cdots,p-1 \tag{14}$$

The minimum value of E_p is

$$E_p = \underline{z}^t \underline{z} + \underline{c}_p^t\, \underline{d}_p \tag{15}$$

or using (10)-(12)

$$E_p = \underline{z}^t\underline{z} - \underline{z}^t X(X^tX)^{-1}X^t\underline{z} \qquad (16).$$

From (16), (12), (14) it follows that E_p depends on the lag between x(n) and z(n). This means that even if the shape of z(n) is unchanged the value of E_p will be different if we shift z(n) relative to x(n). To make this point more clear suppose that we want to shape x(n) into the form of a wavelet s(n), $0 \le n \le M-1$. Then we set z(n) = s(n+l) where l is the value of lag. Since E_p depends on l we set E_p^l to show this dependance and using (16) we take

$$E_p^l = \sum_{n=0}^{M-1} s^2(n) - \underline{z}^t X(X^tX)^{-1}X^t\underline{z} \qquad (17)$$

If we shift s(n) in such a way as all its samples are outside of the interval $0 \le n \le N+p-2$, then $\underline{z} = \underline{0}$. In this case $\underline{c}_p = \underline{0}$ and the error E_p is fully irreducible. To give a possibility to a shaping filter to reduce E_p^l should at least one nonzero term of s(n) lies to the interval $0 \le n \le N+p-2$ since in this case the desired response vector $\underline{Z}$ is non-zero. To found the allowed range for l we consider the two extreme cases:

$$z(0) = s(M-1)\ ,\quad z(n) = 0\ ,\quad 1 \le n \le N+p-2 \qquad (18)$$

$$z(N+p-2) = s(0)\ ,\quad z(n) = 0\ ,\quad 0 \le n \le N+p-3 \qquad (19)$$

which combined with z(n) = s(n+l) supply that

$$\begin{aligned} &l_1 \le l \le l_2 \\ &l_1 = -(N+p-2) \\ &l_2 = M-1 \end{aligned} \qquad (20)$$

The filter $\underline{c}_p^l$ corresponding to lag l is given by

$$R_p\,\underline{c}_p^l = -\underline{d}_{l,l+p-1} \qquad (21)$$

where

$$\underline{d}_{l,l+p-1} = [r_{xz}(0)\ r_{xz}(1) \cdots r_{xz}(p-1)]^t \qquad (22)$$

$$= [r_{xs}(l)\ r_{xs}(l+1) \cdots r_{xs}(l+p-1)]^t \qquad (23)$$

and

$$r_{xs}(l) = \sum_n x(n)\,s(n+l) \qquad (24)$$
$$l_1 \le l \le l_2+p-1$$

It can be easily seen that $r_{xw}(l)$ is the convolution of x(n) with s(-n). Thus it has only N+M-1 non zero terms in the interval $-(N-1) \le l \le (M-1)$ and can be easily computed with subroutine COREL. The elements of the autocorrelation matrix R_p are obtained using subroutine AUTOC.

Thus to design the shaping filter with the optimum lag we have to solve the family of systems

$$R_p[\underline{c}_p^{l_1}\ \underline{c}_p^{l_1+1} \cdots \underline{c}_p^{l_2}] = -[\underline{d}_{l_1,l_1+p-1} \cdots \underline{d}_{l_2,l_2+p-1}] \qquad (25)$$

and then choose the filter which attains the minimum value of E_p^l. A fast solution of this problem is proposed in the next section.

3. FAST DESIGN OF SHAPING FILTERS

In this section we describe a new efficient method for the design of shaping filters which is based on the fast algorithms for FIR Wiener filters with optimum lag introduced in /3/.

These algorithms give the possibility to compute recursively either $\underline{c}_p^{l_1}, \underline{c}_p^{l_1+1}, \ldots, \underline{c}_p^{l_2}$ or $E_p^{l_1}, E_p^{l_1+1}, \ldots, E_2^{l_2}$ or both. This can be achieved using the following procedures proved in /3/.

PROCEDURE 1: (Initialization)

Solve the system

$$R_p\,\underline{c}_p^l = -\underline{d}_{l,l+p-1} \qquad (26)$$

for $l = l_1$

using the Levinson algorithm:

$$\underline{c}_{m+1}^l = \begin{bmatrix}\underline{c}_m^l \\ 0\end{bmatrix} + k_{m+1}^l \begin{bmatrix}\underline{w}_m \\ 1\end{bmatrix} \qquad (27)$$

$$k_{m+1}^l = -\beta_m^l / \alpha_m \qquad (28)$$

$$\beta_m^l = \underline{w}_m^t\,\underline{d}_{l,l+m-1} + d_{l+m} \qquad (29)$$

$$\alpha_m = \alpha_{m-1} + \beta_{m-1}^w k_m^w \qquad (30)$$

$$J\underline{w}_{m+1} = \begin{bmatrix}J\underline{w}_m \\ 0\end{bmatrix} + k_{m+1}^w \begin{bmatrix}\underline{w}_m \\ 1\end{bmatrix} \qquad (31)$$

$$k_{m+1}^w = -\beta_m^w / \alpha_m \qquad (32)$$

$$\beta_m^w = \underline{w}_m^t\,\underline{r}_m + r_{m+1} \qquad (33)$$

where

$$\underline{r}_m = [r_1\ r_2 \cdots r_m]^t \qquad (34)$$

and

$$J = \begin{bmatrix} 0 & & 1 \\ & \cdot^{\cdot^{\cdot}} & \\ 1 & & 0 \end{bmatrix} \qquad (35)$$

The quantities $\underline{w}_{p-1}$, α_{p-1} will be required in all the subsequent stages of the algorithm and so their storage is necessary.

PROCEDURE 2 (Computation of the succeeding filter c_p^{l+1}).

Given $\underline{c}_p^{l}$ compute $\underline{c}_p^{l+1}$ using the following coupled "step-down/step-up" recursions:

"step-down" recursion

$$\begin{bmatrix} J\,\underline{c}_{p-1}^{\ell+1} \\ 0 \end{bmatrix} = J\,\underline{c}_p^{\ell} - \hat{k}_p^{\ell} \begin{bmatrix} \underline{w}_{p-1} \\ 1 \end{bmatrix} \qquad (36)$$

"step-up" recursion

$$\underline{c}_p^{\ell+1} = \begin{bmatrix} \underline{c}_{p-1}^{\ell+1} \\ 0 \end{bmatrix} + k_p^{\ell+1} \begin{bmatrix} \underline{w}_{p-1} \\ 1 \end{bmatrix} \qquad (37)$$

where

$$k_p^{\ell+1} = -\beta_{p-1}^{\ell+1} / \alpha_{p-1} \qquad (38)$$

and

$$\beta_{p-1}^{\ell+1} = \underline{d}^{t}_{\ell+1,\ell+p-1}\, \underline{w}_{p-1} + d_{\ell+p} \qquad (39)$$

Note that using PROCEDURE 2 for $l=l_1+1$ until $l=l_2$ we can compute recursively all shaping filters.

The recursive computation of error can be independently done using the following procedures /3/:

PROCEDURE A (Initialization)

- CALL PROCEDURE 1 for $l=l_1$
- Compute $E_p^{l_1}$ using

$$E_p^{\ell_1} = \sum_{n=0}^{M-1} s^2(n) + \underline{d}^{t}_{\ell_1,\ell_1+p-1}\, \underline{c}_p^{\ell_1} \qquad (40)$$

PROCEDURE B (Compute E_p^{l+1} from E_p^{l})

This is achieved using the formulas

$$\hat{\beta}_{p-1}^{\ell} = \underline{w}^{t}_{p-1}\, J\, \underline{d}_{\ell+1,\ell+p-1} + d_{\ell} \qquad (41)$$

$$\hat{k}_p^{\ell} = -\hat{\beta}_{p-1}^{\ell} / \alpha_{p-1} \qquad (42)$$

$$\beta_{p-1}^{\ell+1} = \underline{d}^{t}_{\ell+1,\ell+p-1}\, \underline{w}_{p-1} + d_{\ell+p} \qquad (43)$$

$$k_p^{\ell+1} = -\beta_{p-1}^{\ell+1} / \alpha_{p-1} \qquad (44)$$

$$E_p^{\ell+1} = E_p^{\ell} + \left(\hat{k}_p^{\ell} - k_p^{\ell+1}\right)\left(\hat{k}_p^{\ell} + k_p^{\ell+1}\right)\alpha_{p-1} \qquad (45)$$

Procedure 1 requires $2p^2+p$ MAD (Multiplications and Divisions) and Procedure 2 requires $3(p-1)+1$ MAD for each value of l. In contrast the well-known Simpson's sideways recursion requires $4(p-1)+2$ MAD /5/. Procedure B requires only $2p+2$ MAD for each value of l.

The algorithm we propose for the computation of optimum lag shaping filter is:

- CALL PROCEDURE A for $l=l_1$
- COMPUTE $E_p^{l_1}$, $E_p^{l_1+1}$, ..., $E_p^{l_2}$ using PROCEDURE B
- FIND the minimum value of E_p^{l} and the corresponding lag l^*
- COMPUTE $\underline{c}_p^{l^*}$ using the Levinson algorithm

This is achieved using the FORTRAN subroutine SFWOL.

The call statement is

```
CALL SFWOL (X,LX,S,LS,C,IP,Y,LY,ER,LAG,R,D,W)
```

The subroutine inputs are

$$X = [x(0)\; x(1)\; \cdots\; x(N-1)]^{t}$$
$$LX = N$$
$$S = [s(0)\; s(1)\; \cdots\; s(M-1)]^{t}$$
$$LS = M$$
$$IP = P$$

The subroutine outputs are

$$C = [c_1\, c_2\; \cdots\; c_p]^{t}$$
$$Y = [y(0)\; y(1)\; \cdots\; y(LY)]$$
$$LY = N+P-1$$
$$E = [E_p^{\ell_1}\; E_p^{\ell_1+1} \cdots E_p^{\ell_2}]$$
$$LAG = \ell^*$$
$$W = [w_{p-1}\; w_{p-2}\; \cdots\; w_1]^{t}$$

D = auxiliary array with dimension
$$LD = M+N+2P-3$$

To demonstrate the use of subroutine SFWOL we give next a numerical example. We use SFWOL to design the shaping filter with optimum lag for the mixed-phase wave-let given in /4/.

The subroutine inputs are

$$X = [50.,-65.,\, 28.,\, 68.,\, 6.,\, -9.,\, -2.]^{t}$$
$$LX = 7$$
$$S = [0.5\,,\, 0.8\,,\, 1.0\,,\, 0.8\,,\, 0.5]^{t}$$
$$LS = 5$$
$$IP = 5$$

On return the subroutine supplies

$$\ell^* = -3$$

```
C(1)= 0.5955E-02          E(-10)= 0.1000E 01
C(2)= 0.1171E-01          E( -9)= 0.9985E 00
C(3)= 0.1337E-01          E( -8)= 0.9961E 00
C(4)= 0.1123E-01          E( -7)= 0.9747E 00
C(5)= 0.6009E-02          E( -6)= 0.7854E 00
Y( 1)= 0.2978E 00         E( -5)= 0.5178E 00
Y( 2)= 0.1984E 00         E( -4)= 0.1993E 00
Y( 3)= 0.7430E-01         E( -3)= 0.7690E-01
Y( 4)= 0.4251E 00         E( -2)= 0.1695E 00
Y( 5)= 0.7768E 00         E( -1)= 0.5112E 00
Y( 6)= 0.8501E 00         E(  0)= 0.7393E 00
Y( 7)= 0.8950E 00         E(  1)= 0.9094E 00
Y( 8)= 0.3322E 00         E(  2)= 0.9527E 00
Y( 9)=-0.9179E-01         E(  3)= 0.9700E 00
Y(10)=-0.7655E-01         E(  4)= 0.9777E 00
Y(11)=-0.1202E-01
```

which are in agreement with the results given in /4/.

An alternative approach would be to use the subroutine EPROC2 given in /3/. This method however requires the storage of all filters $\underline{c}_p^{l_1}, \underline{c}_p^{l_1+1} \dots\dots \underline{c}_p^{l_2}$.

4. DESIGN OF l-STEPS AHEAD PREDICTORS

An l-steps ahead predictor is defined by

$$y(n) = -\sum_{j=1}^{p} a_j^{\ell}\, x(n-j) \tag{46}$$

and

$$z(n) = x(n+\ell) \quad , \quad \ell \geqslant 1 \tag{47}$$

It can be easily seen that the optimum LS predictor is again given by (21) which takes the form

$$R_p\, \underline{a}_p^{\ell} = -\underline{r}_{\ell,\ell+p-1} \tag{48}$$

where

$$\underline{a}_p^{\ell} = [\, a_1^{\ell}\ a_2^{\ell}\ \dots\ a_p^{\ell}\,]^t \tag{49}$$

$$\underline{r}_{\ell,\ell+p-1} = [\, r(\ell)\ r(\ell+1)\ \dots\ r(\ell+p-1)\,]^t \tag{50}$$

Thus to compute the optimum prediction distance for $1 \leq l \leq l_{max}$ we need the autocorrelation function r_j for $j=0,1,\dots,p+l_{max}-1$.
Subroutine LSPRE solves the optimum prediction distance problem using subroutines LEVINS and PROCB. The subroutine inputs are

$$X = [\, x(0)\ x(1)\ \dots\ x(N-1)\,]^t$$
$$LX = N$$
$$IP = p$$
$$LMAX = \ell_{max}$$

The subroutine outputs are

$$A = [\, a_1\ a_2\ \dots\ a_p\,]^t$$
$$W = [\, w_{p-1}\ w_{p-2}\ \dots\ w_1\,]^t$$
$$R = [\, r_0\ r_1\ \dots\ r_{p+\ell_{max}-1}\,]^t$$
$$L = \text{optimum prediction distance}$$
$$Y = [\, y(0)\ y(1)\ \dots\ y(N+p-2)\,]^t = \text{actual output}$$
$$E = [\, E_p^1\ E_p^2\ \dots\ E_p^{\ell_{max}}\,]$$

If subroutine LSPRE is called with

$$X = [\, 50.,\ -65.,\ 28.,\ 68.,\ 6.,\ -9.,\ -2.\,]^t$$
$$LX = 7$$
$$IP = 5$$
$$LMAX = 2$$

On return supplies

$$L = 1$$

$$E = \begin{bmatrix} 0.813141E\ 00 \\ 0.886663E\ 00 \end{bmatrix}$$

$$A = \begin{bmatrix} -0.298801E\ 00 \\ -0.326304E\ 00 \\ 0.107930E\ 00 \\ 0.898610E-01 \\ 0.115025E\ 00 \end{bmatrix} \qquad Y = \begin{bmatrix} -0.149400E\ 02 \\ 0.310682E\ 01 \\ 0.182398E\ 02 \\ -0.319774E\ 02 \\ -0.210492E\ 02 \\ 0.311012E\ 01 \\ 0.135132E\ 02 \\ 0.804208E\ 01 \\ -0.334461E\ 00 \\ -0.121494E\ 01 \\ -0.230049E\ 00 \end{bmatrix}$$

A more efficient implementation could be achieved using subroutine DURBIN /6/ instead of LEVINS for the initialization step (i.e. for l=1).

5. CONCLUSION

This paper was dealt with the efficient design of waveshaping filters with optimum lag and optimum l-steps ahead predictors. The Toeplitz case arising under a stationarity assumption or when both pre- and post-windowing are used was examined. The more general near-to-Toeplitz case is treated in /7/.

REFERENCES

1. Robinson, E.A. and Treitel S., Geophysical Signal Analysis,Prentice Hall, 1980.
2. Silvia, M.T. and Robinson,E.A.,Devoncolution of Geophysical Time Series in the Exploration for Oil and Natural Gas, Elsevier, 1979.
3. Manolakis, D., Kalouptsidis,N. and Carayannis G., "Fast algorithms for discrete-time Wiener filters with optimum lag",IEEE Trans. on ASSP, Febr. 1983.
4. Robinson,E.A.,Multichannel Time Series Analysis with Digital Computer Programs,Holden-Day,1978.
5. Wiggins,R.A. and Robinson,E.A., "Recursive Solution to the Multichannel Filtering Problem", J.Geoph.Res. Vol.70, pp.1885-1891, April 1965.
6. Carayannis,G.,Kalouptsidis,D. and Manolakis,D.,"Fast Recursive Algorithms for a Class of Linear Equations" IEEE Trans.on ASSP, Vol.30,No.2.,pp 227-239,April82.
7. Kalouptsidis,N., Carayannis,G., and Manolakis D., "Systems of Equations with Near-to-Toeplitz or near to-Hankel parameters and applications to signal processing", Proc. IEEE Int. Conf. on ASSP, Boston,USA, April 1983.

APPENDIX

Next are given the listings of the various subroutines used in this paper. The call statement for subroutine LEVINS which is given in /6/ should be changed to

SUBROUTINE LEVINS (IP, X,D,A,W,ALPHA)

```
      SUBROUTINE SFWOL(X,LX,S,LS,C,IP,Y,LY,E,LAG,D,R,W)
      DIMENSION X(1),S(1),C(1),Y(1),E(1),D(1),W(1),R(1)
      LY=LX+IP-1
      LE=LY+LS-1
      LD=LE+IP-1
      CALL AUTOC(X,LX,R,IP)
      DO 10 I=1,LD
      D(I)=0.
10    CONTINUE
      CALL COREL(X,LX,S,LS,D(IP),LR)
      JHELP=LR+IP-1
      DO 20 I=IP,JHELP
      D(I)=-D(I)
20    CONTINUE
      SP=0.
      DO 30 I=1,LS
      SP=SP+S(I)*S(I)
30    CONTINUE
      E(1)=SP
      CALL PROCA(IP,R,D,C,W,1,E(1),ALPHA)
      ERROR=E(1)
      LAG=1
      DO 40 L=2,LE
      LL=L
      CALL PROCB(IP,W,D,LL,ERROR,ALPHA)
      E(L)=ERROR
      IF(E(L).LT.E(LAG)) LAG=L
40    CONTINUE
      DO 50 L=1,LE
      E(L)=E(L)/SP
50    CONTINUE
      IF(LAG.GT.1)
     +CALL LEVINS(IP,R,D(LAG),C,W,ALPHA)
      LAG=LAG-LY
      CALL CONVL(IP,C,LX,X,LY,Y)
      RETURN
      END
```

```
      SUBROUTINE LSPRE(X,LX,IP,LMAX,A,W,R,L,Y,E)
      DIMENSION X(1),A(1),R(1),Y(1),W(1),E(1)
      L2=IP+LMAX
      CALL AUTOC(X,LX,R,L2)
      DO 10 I=2,L2
      Y(I-1)=-R(I)
10    CONTINUE
      CALL LEVINS(IP,R,Y,A,W,ALPHA)
      ERROR=R(1)
      DO 20 I=1,IP
      ERROR=ERROR+A(I)*Y(I)
20    CONTINUE
      E(1)=ERROR
      L=1
      DO 30 I=2,LMAX
      LL=I
      CALL PROCB(IP,W,Y,LL,ERROR,ALPHA)
      E(I)=ERROR
      IF(E(I).LT.E(L)) L=I
30    CONTINUE
      DO 40 I=1,LMAX
      E(I)=E(I)/R(1)
40    CONTINUE
      IF(L.GT.1)CALL LEVINS(IP,R,Y(L),A,W,ALPHA)
      CALL CONVL(IP,A,LX,X,LY,Y)
      RETURN
      END
```

```
      SUBROUTINE AUTOC(X,N,R,L)
      DIMENSION X(1),R(1)
      DO 1 K=1,L
      R(K)=0.
      NK=N-K+1
      DO 1 ND=1,NK
      NDK=ND+K-1
      R(K)=R(K)+X(ND)*X(NDK)
1     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PROCB(IP,W,D,L,ERROR,ALPHA)
      DIMENSION W(1),D(1)
      JIM=IP-1
      L=L-1
      BETAF=D(L)
      DO 1 M=1,JIM
      BETAF=BETAF+W(M)*D(L+M)
1     CONTINUE
      RKF=-BETAF/ALPHA
      BETA=D(L+IP)
      DO 2 M=1,JIM
      BETA=BETA+D(L+M)*W(IP-M)
2     CONTINUE
      RK=-BETA/ALPHA
      ERROR=ERROR+(RKF-RK)*(RKF+RK)*ALPHA
      RETURN
      END
```

```
      SUBROUTINE PROCA(IP,R,D,A,W,LZERO,ERROR,ALPHA)
      DIMENSION R(1),D(1),A(1),W(1)
      CALL LEVINS(IP,R,D(LZERO),A,W,ALPHA)
      DO 1 M=1,IP
      ERROR=ERROR+A(M)*D(LZERO-1+M)
1     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE COREL(X,LX,Y,LY,RXY,LR)
      DIMENSION X(1),Y(1),RXY(1)
      LR=LX+LY-1
      DO 1 N=1,LR
      RXY(N)=0.
1     CONTINUE
      DO 2 I=1,LX
      DO 2 J=1,LY
      N=LR+2-I-J
      RXY(N)=RXY(N)+X(I)*Y(LY+1-J)
2     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CONVL(LA,A,LX,X,LY,Y)
      DIMENSION A(1),X(1),Y(1)
      LY=LA+LX-1
      DO 1 N=1,LY
1     Y(N)=0.
      DO 2 I=1,LA
      DO 2 J=1,LX
      N=I+J-1
2     Y(N)=Y(N)+A(I)*X(J)
      RETURN
      END
```