

Single Channel Backus Type Predictive Deconvolution Operator

(Existing Rock Solid Images Technology)

Dr. M. Turhan (Tury) Taner

15 April 1974

Introduction:

Peg-Leg multiples are the type of multiples where one or more of the travel paths, from surface to reflector and back is of different time lag than the others. The particular type we are interested here is the one described and analyzed by Backus (1959). He suggested that reverberation patterns following deeper reflectors are more complex than the multiples of the water bottom reflections, this complication is caused by the seismic waves traveling through the reverberation zone twice; once going down and once coming up. Thus, a three-point operator would suppress such reverberations. This approach, though practical for analogue operations, has several drawbacks;

- 1) Multiple lags must be accurately known for each application,
- 2) Water bottom or main multiple generating interface must be a simple, sharp interface, and
- 3) Reverberation on the down-going side must be identical to the reverberation on the receiver side.

In simple case water bottom multiple generation is given by;

$$1) \quad R(z) = \frac{1}{(1 + rz^n)} \quad , \text{ where}$$

r = water bottom reflection coefficient

n = travel time in the water layer in terms of time samples,

R(z) = Water bottom reverberation response (given in z-transform)

Since we pass through the water layer twice, Backus gives the total response as the auto-convolution of the water layer response;

$$2) \quad W(z) = R^2(z) = \frac{1}{(1 + rz^n)^2} \quad .$$

Therefore the deconvolution operator will be the inverse of equation (2);

$$3) \quad D(z) = [W(z)]^{-1} = (1 + rz^n)^2 = 1 + 2rz^n + r^2z^{2n} \quad ,$$

Thus, the three point operator is the inverse of the water layer reverberation. In real cases, where the water bottom is not a simple reflector, then, we need to compute multi-point deconvolution operators, as we do with other deconvolution operator computation. Kunetz and Fourman (1968) gave a solution for multi-point operators; three sets of M point operators, equally spaced. He also indicated that the solution in this case be obtained economically by the Toeplitz type matrices. We will use Kunetz' method of reordering matrices for our formulation.

Let us assume that the water bottom response is H(z) , rather than simple r, then the reverberation is represented by ,

$$4) \quad W(z) = \frac{1}{(1 + H(z)z^n)^2} \quad , \text{ and its inverse is,}$$

$$5) \quad D(z) = [W(z)]^{-1} = (1 + H(z)z^n)^2 = 1 + 2H(z)z^n + H(z)^2 z^{2n} .$$

Equation (5) indicate two sets of operators, the second set as twice the length of the first set. For the sake of simplicity, the least squares computations will be formed for two sets of M+1 point operators.

Computation of Two Cluster Decon Operators:

We wish to compute two sets of M+1 point operators a(i) and b(i) that minimizes;

$$5) \quad \mathbf{e}^2 = \sum_{i=I_1}^{I_2} \left\{ f(i+L) - \sum_{k=0}^M [a(k)f(i-k) + b(k)f(i-L-k)] \right\}^2 .$$

To determine the minimum, partial derivatives with respect to each unknown are set to zero;

$$6) \quad \begin{aligned} \partial \mathbf{e}^2 / \partial a(m) &= \sum_{i=I_1}^{I_2} [f(i+L).f(i-m)] - \sum_i f(i-m) \sum_{k=0}^M [a(k)f(i-k)] + \sum_i f(i-m) \sum_{k=0}^M [b(k)f(i-L-k)] = 0 \\ \partial \mathbf{e}^2 / \partial b(m) &= \sum_{i=I_1}^{I_2} [f(i+L).f(i-L-m)] - \sum_i f(i-L-m) \sum_{k=0}^M [a(k)f(i-k)] + \sum_i f(i-L-m) \sum_{k=0}^M [b(k)f(i-L-k)] = 0 \end{aligned}$$

for m=0,1,2,3,...,M. which results in the Normal equations. These normal equations may be solved by two different method. One of the method is to use Conjugate Gradient method as described by Koehler and Taner (1985) and the second is forming a Toeplitz matrix approximation and solving them by a modified Levinson recursion.

By the very nature of the computation, normal equations are non-negative Hermitian type (symmetric) matrices. They can, thus, be solved by the Conjugate gradient method. Main computation, as given in Koehler (1985) article, consists of multiplying the square matrix of the normal equations with the conjugate vector matrix. The rest of the operations involve simple updating the error, conjugate and solution vectors. Koehler has shown that matrix product is implied in the equation (6) is in the form of a convolution followed by a correlation (right most two parts of the equation 6.). We will leave the details of the conjugate gradient method to the 1985 article.

The second solution uses the modified Levinson algorithm. To do this we have to modify the order to form Toeplitz type matrices. It is interesting to note that, here we are using two cluster of operators on the same trace. If we consider each operator cluster separately, then it becomes a two-channel 2-D predictive deconvolution operator design problem. (See Davies and Mercado)

If we change the order of summation (in the conventional computation manner), assume long window lengths and stationarity, we can write this expression in terms of the autocorrelation function. This means that ;

$$A(\mathbf{a}-\mathbf{b}) = A(\mathbf{b}-\mathbf{a}) = \sum_i f(t+\mathbf{a})f(t+\mathbf{b}) = \sum_i f(t+\mathbf{b})f(t+\mathbf{a}) ,$$

then, we can write the normal equations in the form of;

$$8) \quad \begin{aligned} \sum_i [a(k)A(k-m) + b(k)A(L+k-m)] &= A(L+m) \\ \sum_i [a(k)A(L-k+m) + b(k)A(k-m)] &= A(2L+m) , \end{aligned}$$

for m=0,1,2,3,...,M. This produces a symmetric matrix of 2[M+!] by 2[M+1] with a natural partitioning for unknowns a and b. Matrix in this form is not of Toeplitz type, thus its solution would have to be by conventional methods.

$$9) \begin{bmatrix} A(0) & A(1)\dots & A(M) & A(L) & A(L+1)\dots & A(L+M) \\ A(1) & A(0)\dots & A(M-1) & A(L-1) & A(L)\dots & A(L+M-1) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ A(M) & A(M-1)\dots & A(0) & A(L-M) & A(L-M+1)\dots & A(L) \\ \hline A(L) & A(L-1)\dots & A(L-M) & A(0) & A(1)\dots & A(M) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ A(L+M) & A(L+M-1)\dots & A(L) & A(M) & A(M-1) & A(0) \end{bmatrix} \begin{bmatrix} a(0) \\ a(1) \\ \dots \\ a(M) \\ \hline b(0) \\ b(1) \\ \dots \\ b(M) \end{bmatrix} = \begin{bmatrix} A(L) \\ A(L+1) \\ \dots \\ A(L+M) \\ \hline A(2L) \\ A(2L+1) \\ \dots \\ A(2L+M) \end{bmatrix}$$

If, however, we partitioned a little differently, we can produce a Toeplitz type of matrix, which could be formed and solved rather easily by the Levinson recursion (modified by Dr. Fulton Koehler).

Kunetz have shown that by changing the order of equation (9) we can obtain the Toeplitz type of matrix. Instead of writing the equation in order of a(0),a(1),...and b(0),b(1),b(2),..., we write it in order of the sequence numbers of each operator, as, a(0),b(0), a(1),b(1),a(2),b(2),.....a(M),b(M).

$$10) \begin{bmatrix} A(0) & A(L) & A(1) & A(L+1) & A(2) & A(L+2) & \dots & \dots & A(M) & A(L+M) \\ A(L) & A(0) & A(L-1) & A(1) & A(L-2) & A(2) & \dots & \dots & A(L-M) & A(M) \\ A(1) & A(L-1) & A(0) & A(L) & A(1) & A(L+1) & \dots & \dots & A(M-1) & A(L+M-1) \\ A(L+1) & A(1) & A(L) & A(0) & A(L-1) & A(1) & \dots & \dots & A(L-M+1) & A(M-1) \\ A(2) & A(L-2) & A(1) & A(L-1) & A(0) & A(L) & \dots & \dots & A(M-2) & A(L+M-2) \\ A(L+2) & A(2) & A(L+1) & A(1) & A(L) & A(0) & \dots & \dots & A(L-M+2) & A(M-2) \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ A(M) & A(L-M) & A(M-1) & A(L-M+1) & \dots & \dots & \dots & \dots & A(0) & A(L) \\ A(L+M) & A(M) & A(L+M-1) & A(M-1) & \dots & \dots & \dots & \dots & A(L) & A(0) \end{bmatrix} \begin{bmatrix} a(0) \\ b(0) \\ a(1) \\ b(1) \\ a(2) \\ b(2) \\ \dots \\ \dots \\ \dots \\ a(M) \\ b(M) \end{bmatrix} = \begin{bmatrix} A(L) \\ A(2L) \\ A(L+1) \\ A(2L+1) \\ A(L+2) \\ A(2L+2) \\ \dots \\ \dots \\ \dots \\ A(L+M) \\ A(2L+M) \end{bmatrix}$$

We now introduce new variables. Let;

$$X(I) = \begin{bmatrix} a(I) \\ b(I) \end{bmatrix}, \quad Y(I) = \begin{bmatrix} A(L+I) \\ A(2L+I) \end{bmatrix} \quad \text{and} \quad r(I) = \begin{bmatrix} A(I) & A(L+I) \\ A(L-I) & A(I) \end{bmatrix}, \quad \text{then equation 10 becomes,}$$

$$11) \begin{bmatrix} r(0) & r(1) & r(2) & \dots & r(M) \\ r^T(1) & r(0) & r(1) & \dots & r(M-1) \\ r^T(2) & r^T(1) & r(0) & \dots & r(M-2) \\ \dots & \dots & \dots & \dots & \dots \\ r^T(M) & r^T(M-1) & r^T(M-2) & \dots & r(0) \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ \dots \\ X(M) \end{bmatrix} = \begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ \dots \\ Y(M) \end{bmatrix},$$

where T represent the transpose matrix. Thus;

$$r^T(I) = \begin{bmatrix} A(I) & A(L-I) \\ A(L+I) & A(I) \end{bmatrix} \quad \text{is the transpose of} \quad r(I) = \begin{bmatrix} A(I) & A(L+I) \\ A(L-I) & A(I) \end{bmatrix}$$

Equation 11 can now be solved by a modified Levinson algorithm. (Algorithm is given by Fulton Koehler)
 Notice that each element of the square matrix is a 2 by 2 matrix and column vector elements are 2 component column matrices.

Modified Levinson Recursion Algorithm;

Let us assume that we have two operators of length 'N', equivalent to unit-step prediction error operators of the Levinson algorithm. These operators satisfy the following equations. The first operator V(I) to satisfy;

$$12) \quad \begin{bmatrix} r(0) & r(1) & r(2) & \dots & r(N-1) \\ r^T(1) & r(0) & r(1) & \dots & r(N-2) \\ r^T(2) & r^T(1) & r(0) & \dots & r(N-3) \\ \dots & \dots & \dots & \dots & \dots \\ r^T(N-1) & r^T(N-2) & r^T(N-3) & \dots & r(0) \end{bmatrix} \begin{bmatrix} V_{N-1}(0) \\ V_{N-1}(1) \\ V_{N-1}(2) \\ \dots \\ V_{N-1}(N-1) \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{1,N-1} \\ [0] \\ [0] \\ \dots \\ [0] \end{bmatrix}$$

$$\text{where;} \quad V_{N-1}(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad V_{N-1}(I) = \begin{bmatrix} v_{N-1,1}(I) \\ v_{N-1,2}(I) \end{bmatrix},$$

$$\text{and} \quad \mathbf{a}_{1,N-1} = \begin{bmatrix} \mathbf{a}_{1,1,N-1} \\ \mathbf{a}_{1,2,N-1} \end{bmatrix}, \quad [0] = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

A second operator W(I) to satisfy;

$$13) \quad \begin{bmatrix} r(0) & r(1) & r(2) & \dots & r(N-1) \\ r^T(1) & r(0) & r(1) & \dots & r(N-2) \\ r^T(2) & r^T(1) & r(0) & \dots & r(N-3) \\ \dots & \dots & \dots & \dots & \dots \\ r^T(N-1) & r^T(N-2) & r^T(N-3) & \dots & r(0) \end{bmatrix} \begin{bmatrix} W_{N-1}(0) \\ W_{N-1}(1) \\ W_{N-1}(2) \\ \dots \\ W_{N-1}(N-1) \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{2,N-1} \\ [0] \\ [0] \\ \dots \\ [0] \end{bmatrix}$$

$$\text{where;} \quad W_{N-1}(0) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad W_{N-1}(I) = \begin{bmatrix} w_{N-1,1}(I) \\ w_{N-1,2}(I) \end{bmatrix},$$

$$\text{and} \quad \mathbf{a}_{2,N-1} = \begin{bmatrix} \mathbf{a}_{2,1,N-1} \\ \mathbf{a}_{2,2,N-1} \end{bmatrix}$$

Now, let us extend both of the operators by one element by adding one zero column element to the last row, and form the product between the extended operators and the Toeplitz hyper-matrix extended by one column and by one row.

$$14) \quad \begin{bmatrix} r(0) & r(1) & r(2) & \dots & r(N) \\ r^T(1) & r(0) & r(1) & \dots & r(N-1) \\ r^T(2) & r^T(1) & r(0) & \dots & r(N-2) \\ \dots & \dots & \dots & \dots & \dots \\ r^T(N-1) & r^T(N-2) & r^T(N-3) & \dots & r(1) \\ r^T(N) & r^T(N-1) & r^T(N-2) & \dots & r(0) \end{bmatrix} \begin{bmatrix} V_{N-1}(0) \\ V_{N-1}(1) \\ V_{N-1}(2) \\ \dots \\ V_{N-1}(N) \\ [0] \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{1,N-1} \\ [0] \\ [0] \\ \dots \\ [0] \\ \mathbf{b}_{1,N-1} \end{bmatrix}.$$

As we see from equation (12) that the right hand side of equation columns of equation (14) with exception of the last row will remain the same. The last row will be an arbitrary column matrix,

$$\mathbf{b}_{1,N-1} = \begin{bmatrix} \mathbf{b}_{1,1,N-1} \\ \mathbf{b}_{1,2,N-1} \end{bmatrix}.$$

We will form the expansion of the second operator the same way;

$$15) \quad \begin{bmatrix} r(0) & r(1) & r(2) & \dots & r(N) \\ r^T(1) & r(0) & r(1) & \dots & r(N-1) \\ r^T(2) & r^T(1) & r(0) & \dots & r(N-2) \\ \dots & \dots & \dots & \dots & \dots \\ r^T(N-1) & r^T(N-2) & r^T(N-3) & \dots & r(1) \\ r^T(N) & r^T(N-1) & r^T(N-2) & \dots & r(0) \end{bmatrix} \begin{bmatrix} W_{N-1}(0) \\ W_{N-1}(1) \\ W_{N-1}(2) \\ \dots \\ W_{N-1}(N) \\ [0] \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{2,N-1} \\ [0] \\ [0] \\ \dots \\ [0] \\ \mathbf{b}_{2,N-1} \end{bmatrix}$$

$$\text{where; } \mathbf{b}_{2,N-1} = \begin{bmatrix} \mathbf{b}_{2,1,N-1} \\ \mathbf{b}_{2,2,N-1} \end{bmatrix}.$$

Let us the sign tilda (\tilde{X}) to indicate order reversing of the elements of a column matrix. That is if

$$X = \begin{bmatrix} a(1) \\ a(2) \end{bmatrix}, \text{ then the order reversed will be, } \tilde{X} = \begin{bmatrix} a(2) \\ a(1) \end{bmatrix}.$$

We can also show that, order reversed $r(k)X(I)$ is $r^T(k)\tilde{X}(I)$.

For example

$$r(k)X(I) = \begin{bmatrix} A(k) & A(L+k) \\ A(L-k) & A(k) \end{bmatrix} \begin{bmatrix} a(1) \\ b(1) \end{bmatrix} = \begin{bmatrix} A(k)a(I) + A(L+k)b(I) \\ A(L-k)a(I) + A(k)b(1) \end{bmatrix}, \text{ in order reversed will be}$$

$$\{r(k)X(I)\}^{\sim} = \begin{bmatrix} A(L-k)a(I) + A(k)b(I) \\ A(k)a(I) + A(L+k)b(1) \end{bmatrix}. \text{ We can show the same result is obtained by ;}$$

$$r^T(k)\tilde{X}(I) = \begin{bmatrix} A(k) & A(L-k) \\ A(L+k) & A(k) \end{bmatrix} \begin{bmatrix} b(1) \\ a(1) \end{bmatrix} = \begin{bmatrix} A(k)b(I) + A(L+k)a(I) \\ A(L+k)b(I) + A(k)a(1) \end{bmatrix}.$$

Let us show that these operators with additional zero elements be represented as V_{N-1}^+ and W_{N-1}^+ .

We wish to form new N+1 element operators so that,

$$16) \quad V_N = V_{N-1}^+ + \mathbf{s}_{11}\tilde{V}_{N-1}^+ + \mathbf{s}_{21}\tilde{W}_{N-1}^+, \\ W_N = W_{N-1}^+ + \mathbf{s}_{12}\tilde{V}_{N-1}^+ + \mathbf{s}_{22}\tilde{W}_{N-1}^+,$$

where $\mathbf{s}_{11}, \mathbf{s}_{21}, \mathbf{s}_{12}, \mathbf{s}_{22}$ are scale factor we wish to determine so that the product of expanded Toeplitz hyper matrix with the operators will have zero elements at the last row.

Let subscript N represent the matrices with N elements; then;

$$17) \quad [R_N][V_N] = \begin{bmatrix} \mathbf{a}_{1,N-1} + \mathbf{s}_{11}\tilde{\mathbf{b}}_{1,N-1} + \mathbf{s}_{21}\tilde{\mathbf{b}}_{2,N-1} \\ [0] \\ [0] \\ \mathbf{b}_{1,N-1} + \mathbf{s}_{12}\tilde{\mathbf{a}}_{1,N-1} + \mathbf{s}_{22}\tilde{\mathbf{a}}_{2,N-1} \end{bmatrix} \text{ and,}$$

$$18) \quad [R_N][W_N] = \begin{bmatrix} \mathbf{a}_{2,N-1} + \mathbf{s}_{12}\tilde{\mathbf{b}}_{1,N-1} + \mathbf{s}_{21}\tilde{\mathbf{b}}_{2,N-1} \\ \mathbf{b}_{2,N-1} + \mathbf{s}_{12}\tilde{\mathbf{a}}_{1,N-1} + \mathbf{s}_{22}\tilde{\mathbf{a}}_{2,N-1} \end{bmatrix} \cdot$$

We will choose \mathbf{s} values so that the last rows of equations (17) and (18) will be equal to zero.

$$19) \quad \begin{aligned} \mathbf{b}_{1,N-1} + \mathbf{s}_{11}\tilde{\mathbf{a}}_{1,N-1} + \mathbf{s}_{21}\tilde{\mathbf{a}}_{2,N-1} &= 0, \\ \mathbf{b}_{2,N-1} + \mathbf{s}_{12}\tilde{\mathbf{a}}_{1,N-1} + \mathbf{s}_{22}\tilde{\mathbf{a}}_{2,N-1} &= 0. \end{aligned}$$

Note that all elements of equation (19) are two element matrices, so we can write them in expanded form as (let us omit the subscript N-1 for the sake of clarity);

$$20) \quad \begin{aligned} \mathbf{b}_{1,1} + \mathbf{s}_{11}\mathbf{a}_{1,2} + \mathbf{s}_{21}\mathbf{a}_{2,2} &= 0, \\ \mathbf{b}_{1,2} + \mathbf{s}_{11}\mathbf{a}_{1,1} + \mathbf{s}_{21}\mathbf{a}_{2,1} &= 0, \\ \mathbf{b}_{2,1} + \mathbf{s}_{12}\mathbf{a}_{1,2} + \mathbf{s}_{22}\mathbf{a}_{2,2} &= 0, \\ \mathbf{b}_{2,2} + \mathbf{s}_{12}\mathbf{a}_{1,1} + \mathbf{s}_{22}\mathbf{a}_{2,1} &= 0. \end{aligned}$$

This could be written in the matrix form as;

$$21) \quad \begin{bmatrix} \mathbf{a}_{11} & \mathbf{a}_{21} \\ \mathbf{a}_{12} & \mathbf{a}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{s}_{11} & \mathbf{s}_{12} \\ \mathbf{s}_{21} & \mathbf{s}_{22} \end{bmatrix} = \begin{bmatrix} -\mathbf{b}_{12} & -\mathbf{b}_{22} \\ -\mathbf{b}_{11} & -\mathbf{b}_{21} \end{bmatrix} \cdot$$

We that the inverse of the $[\mathbf{a}]$ matrix is,

$$22) \quad \begin{bmatrix} \mathbf{a}_{11} & \mathbf{a}_{21} \\ \mathbf{a}_{12} & \mathbf{a}_{22} \end{bmatrix}^{-1} = \frac{-1}{(\mathbf{a}_{11}\mathbf{a}_{22} - \mathbf{a}_{12}\mathbf{a}_{21})} \begin{bmatrix} -\mathbf{a}_{22} & \mathbf{a}_{21} \\ -\mathbf{a}_{12} & -\mathbf{a}_{11} \end{bmatrix}, \text{ therefore;}$$

$$23) \quad \begin{bmatrix} \mathbf{s}_{11} & \mathbf{s}_{12} \\ \mathbf{s}_{21} & \mathbf{s}_{22} \end{bmatrix} = \frac{1}{(\mathbf{a}_{11}\mathbf{a}_{22} - \mathbf{a}_{12}\mathbf{a}_{21})} \begin{bmatrix} -\mathbf{a}_{12} & \mathbf{a}_{21} \\ -\mathbf{a}_{12} & -\mathbf{a}_{11} \end{bmatrix} \begin{bmatrix} \mathbf{b}_{12} & \mathbf{b}_{22} \\ \mathbf{b}_{11} & \mathbf{b}_{21} \end{bmatrix} \cdot$$

Since we have all \mathbf{s} scales by the equation (23), we can extend the operators V and W by one additional matrix element by equation (16). These new operators will be N+1 elements long and will now satisfy the expressions;

$$24) \quad [R_N][V_N] = \begin{bmatrix} \mathbf{a}_{1,N} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \text{ and } [R_N][W_N] = \begin{bmatrix} \mathbf{a}_{2,N} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \text{ where}$$

$$25) \quad \begin{aligned} \mathbf{a}_{1,N} &= \mathbf{a}_{1,N-1} + \mathbf{s}_{11}\tilde{\mathbf{b}}_{1,N-1} + \mathbf{s}_{21}\tilde{\mathbf{b}}_{2,N-1}, \\ \mathbf{a}_{2,N} &= \mathbf{a}_{2,N-1} + \mathbf{s}_{12}\tilde{\mathbf{b}}_{1,N-1} + \mathbf{s}_{22}\tilde{\mathbf{b}}_{2,N-1}. \end{aligned}$$

This will form the expanded operators, which are similar to the single cluster unit-step prediction error operators. The second recursion will extend the general (two cluster) operators by one element. Thew will have to satisfy;

$$26) \quad [R_{N-1}][X_{N-1}] = \begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ \dots \\ Y(N-1) \end{bmatrix} .$$

Similar to the V and W operators, if we expand the X operator by one element by adding a zero [0] to the end, we will get;

$$27) \quad [R_N][X^+_{N-1}] = \begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ \dots \\ Y(N-1) \\ \mathbf{g}_{N-1} \end{bmatrix} .$$

We will now form the extended prediction operator X by scaled combination of the V and W operators;

$$28) \quad X_N = X^+_{N-1} + \mathbf{r}_1 \tilde{V}_N + \mathbf{r}_2 \tilde{W}_N ,$$

where, the product of this operator with the extended Toeplitz will be ,

$$29) \quad [R_N][X_N] = \begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ \dots \\ Y(N-1) \\ \mathbf{g}_{N-1} + \mathbf{r}_1 \tilde{\mathbf{a}}_{1,N} + \mathbf{r}_2 \tilde{\mathbf{a}}_{2,N} \end{bmatrix}$$

Therefore, we will select the scalars $\mathbf{r}_1, \mathbf{r}_2$ so that;

$$30) \quad \mathbf{g}_{N-1} + \mathbf{r}_1 \tilde{\mathbf{a}}_{1,N} + \mathbf{r}_2 \tilde{\mathbf{a}}_{2,N} = Y(N) ,$$

or, more explicitly;

$$31) \quad \begin{aligned} \mathbf{g}_{1,N-1} + \mathbf{r}_1 \mathbf{a}_{1,2,N} + \mathbf{r}_2 \mathbf{a}_{2,2,N} &= A(L+N) \\ \mathbf{g}_{2,N-1} + \mathbf{r}_1 \mathbf{a}_{1,1,N} + \mathbf{r}_2 \mathbf{a}_{2,1,N} &= A(2L+N) \end{aligned} .$$

We compute the scalars by equation (31) which will extend the operators, that will satisfy the extended matrix product ;

$$32) \quad [R_N][X_N] = \begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ \dots \\ Y(N) \end{bmatrix} .$$

We repeat these recursion for V,W and X until desired length of M+1 point operators have been computed. A Fortran listing of this recursion will be included with this report.

It should be remembered that the operators computed here are the Prediction operators. As with the conventional prediction error operators, we will have to negate these operators and used them their proper prediction lags and subtract the results from the original data.

Conclusion:

We have presented here the computation of Backus type two cluster operators by the method of least mean error square method. We have also shown a modified Levinson type of recursive computation method given by Koehler to compute the desired operators.

The advantage of these two cluster operators is its insensitivity to water layer transit times and changes of thickness on the source and the receiver side. Two cluster operators should be chosen by the study of the autocorrelation functions. By selecting the lags and operator lengths, we concentrate on the water layer reverberations. If we use one cluster long operator that covers both the prediction zones, we may loose the effectiveness of these operators by inclusion of other types of multiples into consideration. The least squares method minimizes overall error, and can not discriminate a particular multiple. By selecting the clusters properly, we can somewhat influence the method to consider certain set of multiples.

We have, also, pointed out that the lags used in the operator computation can be independent of each other. We can change the 2L prediction lag with any other lag and the recursion described here will work the same way. Enclosed subroutine is written with two different input for each lag.

References:

Backus, M. M. , 1959, Water Reverberations – Their Nature and Elimination, *Geophysics*, 24, pp 233-261.

Davies, E. B. and Mercado, E. J., 1968, Multi-channel deconvolution filtering of field recorded seismic data: *Geophysics*, 33, no. 5, 711-722.

Kunetz, G. and Fourman, J. M., 1968, Efficient Deconvolution of Marine Seismic Records, *Geophysics*, 33, pp. 412-423.

Koehler, F. and Taner, M. T., 1985, The use of conjugate-gradient algorithms in the computation of predictive deconvolution operators *: *Geophysics*, 50, 2752-2758.

Treitel, S., 1970, Principles of digital multi-channel filtering, *Geophysics*, v.35, 785-811.

(UNDEBUGGED FORTRAN LISTING OF LEVINSON-KOEHLER RECURSION)

```

C*****
C      SUBROUTINE SEIS25(AC, LAG1, LAG2, MFL, AF, BF )
C*****
C
C+SEIS25
C
C-FUNCTION
C  SUBROUTINE TO COMPUTE TWO-CLUSTER BACKUS TYPE OPERATOR
C  BY LEVINSON-KOEHLER RECURSION
C
C-CALLING SEQUENCE:
C
C  CALL SEIS25(AC,LAG1,LAG2,MFL,AF,BF)
C
C-ARGUMENTS:
C
C  INPUT ARGUMENST:
C
C  AC(I) = AUTOCORRELATION FUNCTION (IT SHOULD BE AT LEAST
C  LAG2+MFL LONG.
C  LAG1 = PREDICTION LAG OF THE FIRST CLUSTER ( IN SAMPLES )
C  LAG2 = PREDICTION LAG OF THE SECOND CLUSTER ( LAG2 > LAG1 )
C  MFL = LENGTH OF EACH OPERATOR CLUSTER, IN SAMPLES.
C
C-OUTPUT:
C
C  AF(I) = OPERATORS OF THE FIRST CLUSTER
C  BF(I) = OPERATORS OF THE SECOND CLUSTER, EACH MFL SAMPLES LONG.
C  IF MFL < 0 , SUBROUTINE COULD NOT COMPUTE OPERATORS, CHECK
C  THE VALIDITY OF THE AUTOCORRELATION FUNCTION.
C  THERE IS A LIMIT OF 1000 SAMPLES PER OPERATOR.
C
C-DESCRIPTION:
C
C  SUBROUTINE WILL COMPUTE TWO CLUSTER PREDICTION OPERATORS
C  WITH GIVEN PREDICTION LAGS. USER SHOULD REVERSE POLARITIES
C  AND ADD A WEIGHT 1. TO ZERO LAG FOR PREDICTION ERROR OPERATION.
C  SEE TECHNICAL REPORT BY TANER DATED 15 APRIL 1974.
C
C-ORIGINAL: 19-APRIL-1974 BY M.T. TANER
C
C++
C-----
C      IMPLICIT NONE
C      SAVE
C-----
C      REAL AC(1:*),AF(1:*),BF(1:*),
*      AC1(1000),AC2(1000),CC(1000),FX(1000),ALAM1,ALAM2
*      AX(1000),BX(1000),AZ(1000),BZ(1000),SC11,SC12,SC21,SC22
C
C      REAL*8 ALP11,ALP12,ALP21,ALP22,BETA11,BETA12,BETA21,BETA22,
*      DALP,GAM1,GAM2,EPSI11,EPSI12,EPSI21,EPSI22
C      INTEGER K1,K2,N1,IN1,L1,L2,NOTIM,I,LAG1,LAG2,MFL,MFL2,K,KFL
C
C---- GENERATE FIRST TWO ROWS OF THE MODIFIED TOEPLITZ MATRIX AND
C  MODIFIED RIGHT HAND SIDE
C
C      K1 = 1

```

```

K2 = LAG2 - LAG1 + 1
N1 = K2
IN1 = 1
L1 = LAG1 + 1
L2 = LAG2 + 1
MFL2 = 2 * MFL
C
C----- SET UP MODIFIED MATRIX
C
DO 10 I = 1, MFL2, 2
  CC(I) = AC(L1)
  CC(I+1) = AC(L2)
  AC1(I) = AC(K1)
  AC1(I+1) = AC(K2)
  AC2(I) = AC(N1)
  AC2(I+1) = AC(K1)
  L1 = L1 + 1
  L2 = L2 + 1
  K1 = K1 + 1
  K2 = K2 + 1
  N1 = N1 - IN1
  IF( N1 .GE. 1 ) GO TO 10
  IN1 = -IN1
  N1 = 1
10 CONTINUE
C
C----- MODIFIED TOEPLITZ IS SET UP.
C
  NOTIM = 0
100 CONTINUE
C
C----- CLEAR AUX ARRAYS
C
DO 120 I = 1, MFL
  FX(I) = 0.
  AX(I) = 0.
  BX(I) = 0.
  AZ(I) = 0.
  BZ(I) = 0.
120 CONTINUE
C
C----- COMPUTE INITIAL VALUES
C
FX(1) = (CC(1)*AC1(1)-CC(2)*AC1(2))/(AC1(1)**2-AC1(2)**2)
FX(2) = (CC(2)*AC1(1)-CC(1)*AC1(2))/(AC1(1)**2-AC1(2)**2)
AX(1) = 1.0
AX(2) = 0.0
BX(1) = 0.0
BX(2) = 1.0
AZ(1) = 1.0
AZ(2) = 0.0
BZ(1) = 0.0
BZ(2) = 1.0
C
C----- INITIAL PREDICTION ERROR ESTIMATE
C
ALP11 = AC1(1)
ALP12 = AC2(1)
ALP21 = AC1(2)
ALP22 = AC2(2)
DALP = ALP11*ALP22-ALP12*ALP21
EPSI12 = ALP21/DALP

```

```

      EPSI11 = ALP22/DALP
      EPSI22 = ALP11/DALP
      EPSI21 = ALP12/DALP
C
C----- MAIN LOOP FOR GENERAL RECURSION
C
      DO 500 K = 2, MFL
C
C----- RECURSION FOR UNIT STEP PREDICTION ERROR OPERATOR
C
      BETA11 = 0.0
      BETA12 = 0.0
      BETA21 = 0.0
      BETA22 = 0.0
      GAMA1 = 0.0
      GAMA2 = 0.0
      KFL = (K-1)*2
      L2 = 2*K
      L1 = L2-1
      DO 200 I = 1, KFL, 2
      BETA11 = BETA11 + AX(I)*AC1(L1)+AX(I+1)*AC2(L1)
      BETA21 = BETA21 + BX(I)*AC1(L1)+BX(I+1)*AC2(L1)
      BETA12 = BETA12 + AX(I)*AC1(L2)+AX(I+1)*AC2(L2)
      BETA22 = BETA22 + BX(I)*AC1(L2)+BX(I+1)*AC2(L2)
      GAMA1 = GAMA1 + FX(I)*AC1(L1)+FX(I+1)*AC2(L1)
      GAMA2 = GAMA2 + FX(I)*AC1(L2)+FX(I+1)*AC2(L2)
C
      AZ(I) = AX(I)
      AZ(I+1) = AX(I+1)
      BZ(I) = BX(I)
      BZ(I+1) = BX(I+1)
      L1 = L1-2
      L2 = L2-2
200 CONTINUE
C
C----- COMPUTE SCALARS FOR EXPANSION OF OPERATORS BY ONE ELEMENT
C
      SC11 = EPSI11*BETA12 + EPSI12*BETA11
      SC12 = EPSI11*BETA22 + EPSI12*BETA21
      SC21 = EPSI21*BETA12 + EPSI22*BETA11
      SC22 = EPSI21*BETA22 + EPSI22*BETA21
C
C----- UPDATE AND EXPAND
C
      DO 300 I = 3, KK, 2
      AX(I) = AX(I) + SC11*AZ(LL) + SC21*BZ(LL)
      AX(I+1) = AX(I+1) + SC11*AZ(LL-1) + SC21*BZ(LL-1)
      BX(I) = BX(I) + SC12*AZ(LL) + SC22*BZ(LL)
      AX(I+1) = AX(I+1) + SC12*AZ(LL-1) + SC22*BZ(LL-1)
      LL = LL - 1
300 CONTINUE
C
C----- UPDATE ALPHA
C
      ALP11 = ALP11 + SC11*BETA12 + SC21*BETA22
      ALP12 = ALP12 + SC11*BETA11 + SC21*BETA21
      ALP21 = ALP21 + SC12*BETA12 + SC22*BETA22
      ALP22 = ALP22 + SC12*BETA11 + SC22*BETA21
C
C----- CHECK TO SEE IF PREDICTION ERROR REMAINS NO-NEGATIVE
C
      DALP = ALP11*ALP22 - ALP12*ALP21

```

```

      IF( DALP .LE. 0. ) GO TO 700
C
C-----RECURSION IS GOOD, NOW EXPAND PREDICTION OPERATOR
C
      EPSI12 = ALP21/DALP
      EPSI11 = ALP22/DALP
      EPSI21 = ALP12/DALP
      EPSI22 = ALP11/DALP
      ALAM1 = EPSI11*(GAMA2 - CC(KK)) + EPSI12*(GAMA1-CC(KK-1))
      ALAM2 = EPSI21*(GAMA2 - CC(KK)) + EPSI22*(GAMA1-CC(KK-1))
      LL = KK
      DO 400 I = 1, KK
      FX(I) = FX(I) + ALAM1*AX(LL) + ALAM2*BX(LL)
      LL = LL - 1
400 CONTINUE
C
C----- ONE TERM EXPANSION IS COMPLETED -- GO TO THE NEXT STEP
C
500 CONTINUE
C
C----- OPERATORS ARE COMPLETED - NOW TRASFER TO THE OUTPUT ARRAYS
C
      LL = 1
      DO 600 I = 1, MFL
      AF(I) = FX(LL)
      BF(I) = FX(LL+1)
      LL = LL + 2
600 CONTINUE
C
C----- ALL DONE -----
C
      RETURN
C
C+++++++ THIS IS CORRECTION OR ERROR RETURN AREA
C
700 CONTINUE
      NOTIM = NOTIM + 1
      AC1(1) = AC1(1)*1.005
      AC2(2) = AC1(1)
      IF(NOTIM .LT. 5 ) GO TO 100
C
C----- WE COULD NOT COMPUTE A GOOD OPERATOR - CHECK AUTOCORRELATION
C
      MFL = -MFL
      RETURN
      END

```