```
r=[1+0.9^(2)  -0.9  0 0 ]
```

← ~~ACF~~ = $[\gamma(0), \gamma(1), \gamma(2), \gamma(3)]$ för $X_t = Z_t - 0.9 Z_{t-1}$

ACVF

```
r =

    1.8100   -0.9000         0         0

GA=toeplitz(r)

GA =

    1.8100   -0.9000         0         0
   -0.9000    1.8100   -0.9000         0
         0   -0.9000    1.8100   -0.9000
         0         0   -0.9000    1.8100

[C v]=ldl(GA)
```

COVARIANCE MATRIX Γ_4

← lower diagonal decomposition av Γ_n using MATLAB SIGNAL PROCESSING TOOLBOX

```
C =

    1.0000         0         0         0
   -0.4972    1.0000         0         0
         0   -0.6606    1.0000         0
         0         0   -0.7404    1.0000

v =

    1.8100         0         0         0
         0    1.3625         0         0
         0         0    1.2155         0
         0         0         0    1.1436
```

$\Gamma_4 = C_4 D_4 C_4^T$

$= D_4$

```
innov(GA,3)
```

← ST 2543 HOME PAGE

```
ans =

   -0.4972         0         0
   -0.6606         0         0
   -0.7404         0         0

v.*eye(4)

ans =

    1.8100         0         0         0
         0    1.3625         0         0
         0         0    1.2155         0
         0         0         0    1.1436

D=v.*eye(4)

D =

    1.8100         0         0         0
         0    1.3625         0         0
         0         0    1.2155         0
         0         0         0    1.1436

C*D*C'

ans =

    1.8100   -0.9000         0         0
   -0.9000    1.8100   -0.9000         0
         0   -0.9000    1.8100   -0.9000
         0         0   -0.9000    1.8100

GA

GA =

    1.8100   -0.9000         0         0
```

A CHECK

~~GA = D*~~ $GA = C * D * C'$

```
   -0.9000    1.8100   -0.9000         0
         0   -0.9000    1.8100   -0.9000
         0         0   -0.9000    1.8100

help ldl
 <strong>ldl</strong>   Block <strong>ldl</strong>' factorization for Hermitian indefinite matrices.
    Assuming that A is a Hermitian matrix (that is, A == A'),
    [L,D] = <strong>ldl</strong>(A) stores a block diagonal matrix D and a "psychologically
    lower triangular matrix" (i.e. a product of unit lower triangular and
    permutation matrices) in L so that A = L*D*L'.  The block diagonal matrix
    D has 1x1 and 2x2 blocks on its diagonal.  This syntax is not valid for
    sparse A.

    [L,D,P] = <strong>ldl</strong>(A) returns unit lower triangular matrix L, block diagonal D,
    and permutation matrix P so that P'*A*P = L*D*L'.  This is equivalent
    to [L,D,P] = <strong>ldl</strong>(A,'matrix').

    [L,D,p] = <strong>ldl</strong>(A,'vector') returns the permutation information as a
    vector instead of a matrix.  That is, p is a row vector such that
    A(p,p) = L*D*L'.

    L = <strong>ldl</strong>(A) returns only the "psychologically lower triangular matrix" L
    as in the two output form.  The permutation information is lost, as is
    the block diagonal factor D.  This syntax is not valid for sparse A.

    By default, <strong>ldl</strong> references only the diagonal and lower triangle of A,
    and assumes that the upper triangle is the complex conjugate transpose
    of the lower triangle.  Therefore [L,D,P] = <strong>ldl</strong>(TRIL(A)) and
    [L,D,P] = <strong>ldl</strong>(A) both return the exact same factors.

    [U,D,P] = <strong>ldl</strong>(A,'upper') references only the diagonal and upper triangle
    of A and assumes that the lower triangle is the complex conjugate
    transpose of the upper triangle.  This call returns a unit upper triangular
    matrix U such that P'*A*P = U'*D*U (assuming that A is Hermitian, and not
    just upper triangular).  Similarly, [L,D,P] = <strong>ldl</strong>(A,'lower') gives
    the default behavior.

    [U,D,p] = <strong>ldl</strong>(A,'upper','vector') returns the permutation information
    as a vector, as does [L,D,p] = <strong>ldl</strong>(A,'lower','vector').

    [L,D,P,S] = <strong>ldl</strong>(A) returns unit lower triangular L, block diagonal D,
    permutation matrix P, and scaling matrix S such that P'*S*A*S*P =
    L*D*L'.  This syntax is only available for real sparse matrices, and
    only the lower triangle of A is referenced. <strong>ldl</strong> uses MA57 for real
    sparse A.

    [L,D,P,S] = <strong>ldl</strong>(A,THRESH) uses THRESH as the pivot tolerance in MA57.
    THRESH must be a double scalar lying in the interval [0, 0.5]. The
    default value for THRESH is 0.01.  Using smaller values of THRESH may
    give faster factorization times and fewer entries, but it may also
    result in a less stable factorization. This syntax is only available
    for real sparse matrices.

    [U,D,p,S] = <strong>ldl</strong>(A,THRESH,'upper','vector') sets the pivot tolerance and
    returns upper triangular U and permutation vector p as described above.

    See also <a href="matlab:help chol">chol</a>, <a href="matlab:help lu">lu</a>, <a
href="matlab:help qr">qr</a>.

    Reference page in Help browser
       <a href="matlab:doc ldl">doc ldl</a>

inv(C)

ans =

    1.0000         0         0         0
    0.4972    1.0000         0         0
    0.3285    0.6606    1.0000         0
    0.2432    0.4891    0.7404    1.0000
```

INNOVATIONS ALGORITHM


```
r=[1+0.9^(2)   -0.9  zeros(1,98)];
GA=toeplitz(r);
[C v]=ldl(GA);
y=simarma([], -0.9, 100,0.5);
e=inv(C)*y';
yp=(C-eye(100))*e;
i=[1:1:100];
plot(i,y,'r',i,yp,'b')
plot(e)
std(e)

ans =

    0.6822

std(e)^2

ans =

    0.4654

v(1)

ans =

    1.8100

1+0.9^2

ans =

    1.8100

diary off
```

ACVF (100 VALUES) FOR $\underline{X}_t = Z_t - 0.9 Z_{t-1}$

$\Gamma_n = (\gamma(i-j))$

FACTORIZES $\Gamma_n = C_n D_n C_n^T$ $n = 5000$

SIMULATES 5000 SAMPLES OF $\underline{X}_t = Z_t - 0.9 Z_{t-1}$

COMPUTES THE INNOVATIONS

$\underline{\varepsilon}_n = A \underline{X}_n = C^{-1} \underline{X}_n$

COMPUTES $\hat{\underline{X}}_n = (C - I)\underline{\varepsilon}_n = C\underline{\varepsilon}_n - \underline{\varepsilon}_n$

$(= \underline{X}_n - \underline{\varepsilon}_n)$

estimate $\hat{\sigma}^2$ $(= 0.5)$

$V(1) = \gamma(1)$

```
y=simarma([], -0.9, 100,0.5);
z=acvf(y);
GA=toeplitz(z);
[C v]=ldl(GA);
e=inv(C)*y';
yp=(C-eye(100))*e;
i=[1:1:100];
plot(i,y,'r',i,yp,'b')
f=acvf(e);
f1=acf(f,1);

plot(e)
std(e)^2

ans =

    0.3152

f2=armaacvf([], -09, 100, 0.5);
GA=toeplitz(f2);
[C v]=ldl(GA);
e=inv(C)*y';
yp=(C-eye(100))*e;
plot(i,y,'r',i,yp,'b')
plot(f2)
f2=armaacvf([], -0.9, 100, 0.5);
GA=toeplitz(f2);
[C v]=ldl(GA);
e=inv(C)*y';
yp=(C-eye(100))*e;
plot(i,y,'r',i,yp,'b')
plot(f2)
r=[1+0.9^(2)  -0.9  0 0 0 0 0]

r =

    1.8100   -0.9000         0         0         0         0         0

r=[1+0.9^(2)  -0.9  0 0 0 0 0]

r =

    1.8100   -0.9000         0         0         0         0         0

GA=toeplitz(r)

GA =

    1.8100   -0.9000         0         0         0         0         0
   -0.9000    1.8100   -0.9000         0         0         0         0
         0   -0.9000    1.8100   -0.9000         0         0         0
         0         0   -0.9000    1.8100   -0.9000         0         0
         0         0         0   -0.9000    1.8100   -0.9000         0
         0         0         0         0   -0.9000    1.8100   -0.9000
         0         0         0         0         0   -0.9000    1.8100

[C v]=ldl(GA);
[C v]=ldl(GA)

C =

    1.0000         0         0         0         0         0         0
   -0.4972    1.0000         0         0         0         0         0
         0   -0.6606    1.0000         0         0         0         0
         0         0   -0.7404    1.0000         0         0         0
         0         0         0   -0.7870    1.0000         0         0
         0         0         0         0   -0.8169    1.0000         0
         0         0         0         0         0   -0.8374    1.0000


v =
```

COMPUTES $\hat{\gamma}(k)$ THE SAMPLE COVARIANCE

FORMS $\hat{\Gamma}_n$

FINDS $\hat{C}_n$

PREDICTIONS USING $\hat{C}_n$

$\hat{X}_n = (\hat{C}_n - I)\xi_n$

COMPUTES $\gamma(k)$ ACVF FROM -0.5

THE PARAMETER VALUE -0.9

$X_t = Z_t - 0.9 Z_{t-1}$

BD, pp. 73-77

```
    1.8100         0         0         0         0         0         0
         0    1.3625         0         0         0         0         0
         0         0    1.2155         0         0         0         0
         0         0         0    1.1436         0         0         0
         0         0         0         0    1.1017         0         0
         0         0         0         0         0    1.0748         0
         0         0         0         0         0         0    1.0564
```

```
z=iddata(y',[],1);
M=armax(z,[ 0 1])
```

SYSTEM IDENTIFICATION TOOLBOX

```
M =
Discrete-time MA model:  y(t) = C(z)e(t)
  C(z) = 1 - 0.9351 z^-1

Sample time: 1 seconds

Parameterization:
Polynomial orders:   nc=1
   Number of free coefficients: 1
   Use "polydata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Estimated using ARMAX on time domain data "z".
Fit to estimation data: 23.39% (prediction focus)
FPE: 0.4902, MSE: 0.4755
diary off
```

$$\gamma(l) = \frac{0.5}{1-0.3^2}(0.3)^l \quad l = 1,2,3,4,5$$

ACVF of $X_t - 0.3X_{t-1} = Z_t$

```
r=0.5*(0.3.^l)/(1-0.3^2)

r =

    0.1648    0.0495    0.0148    0.0045    0.0013

GA=toeplitz(r)

GA =

    0.1648    0.0495    0.0148    0.0045    0.0013
    0.0495    0.1648    0.0495    0.0148    0.0045
    0.0148    0.0495    0.1648    0.0495    0.0148
    0.0045    0.0148    0.0495    0.1648    0.0495
    0.0013    0.0045    0.0148    0.0495    0.1648

[C v]=ldl(GA)

C =

    1.0000         0         0         0         0
    0.3000    1.0000         0         0         0
    0.0900    0.3000    1.0000         0         0
    0.0270    0.0900    0.3000    1.0000         0
    0.0081    0.0270    0.0900    0.3000    1.0000


v =

    0.1648         0         0         0         0
         0    0.1500         0         0         0
         0         0    0.1500         0         0
         0         0         0    0.1500         0
         0         0         0         0    0.1500

inv(C)
ans =

    1.0000         0         0         0         0
   -0.3000    1.0000         0         0         0
    0.0000   -0.3000    1.0000         0         0
   -0.0000         0   -0.3000    1.0000         0
         0         0    0.0000   -0.3000    1.0000

diary off
```

$\Gamma_5 = C_5 D_5 C_5^T$

C_5

D_5

$\underline{\text{inv}(C)} = C_5^{-1} = \Theta_5$

HENCE:

$\hat{X}_t = \phi X_{t-1} = 0.3 X_{t-1}$

AS IS KNOWN.

(SEE HANDOUT ON LINEAR ESTIMATION)

PREDICTION $\quad \hat{\varepsilon}_n = \Theta_n X_n$

$\hat{X}_n = (C_n - I)\hat{\varepsilon}_n = (C_n - I)\Theta_n X_n$

$\Theta_n C_n = I$

$= X_n - \Theta_n X_n$

$= (I - \Theta_n) X_n$