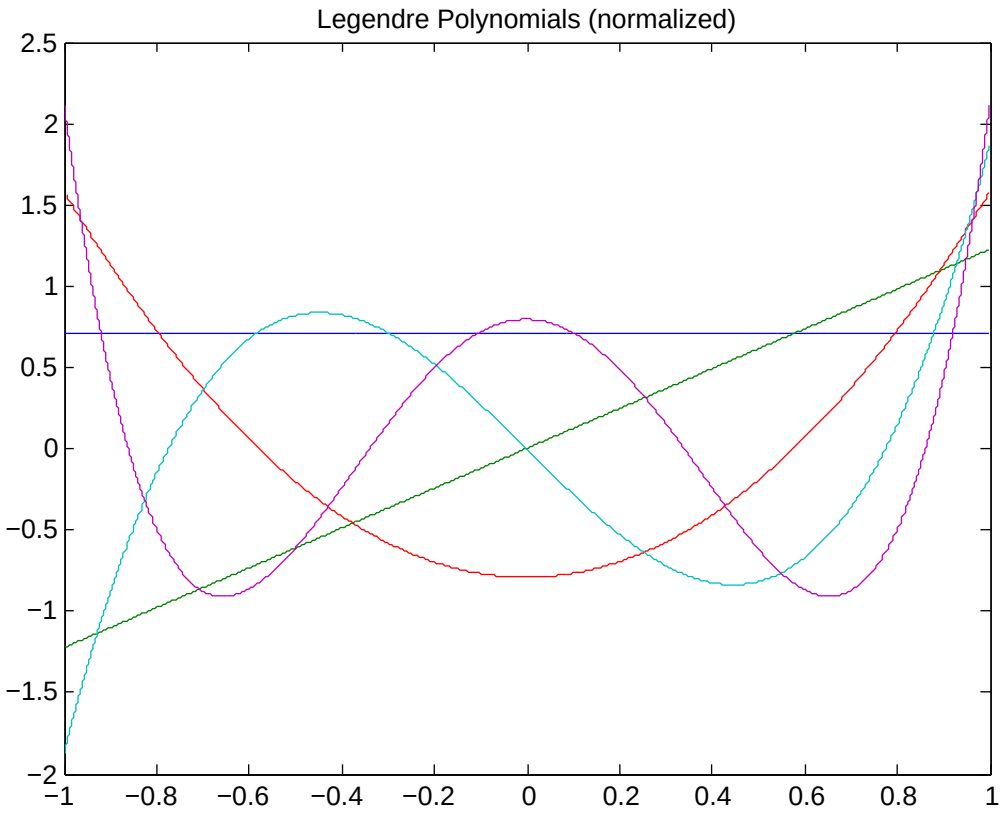
PROBLEM #1

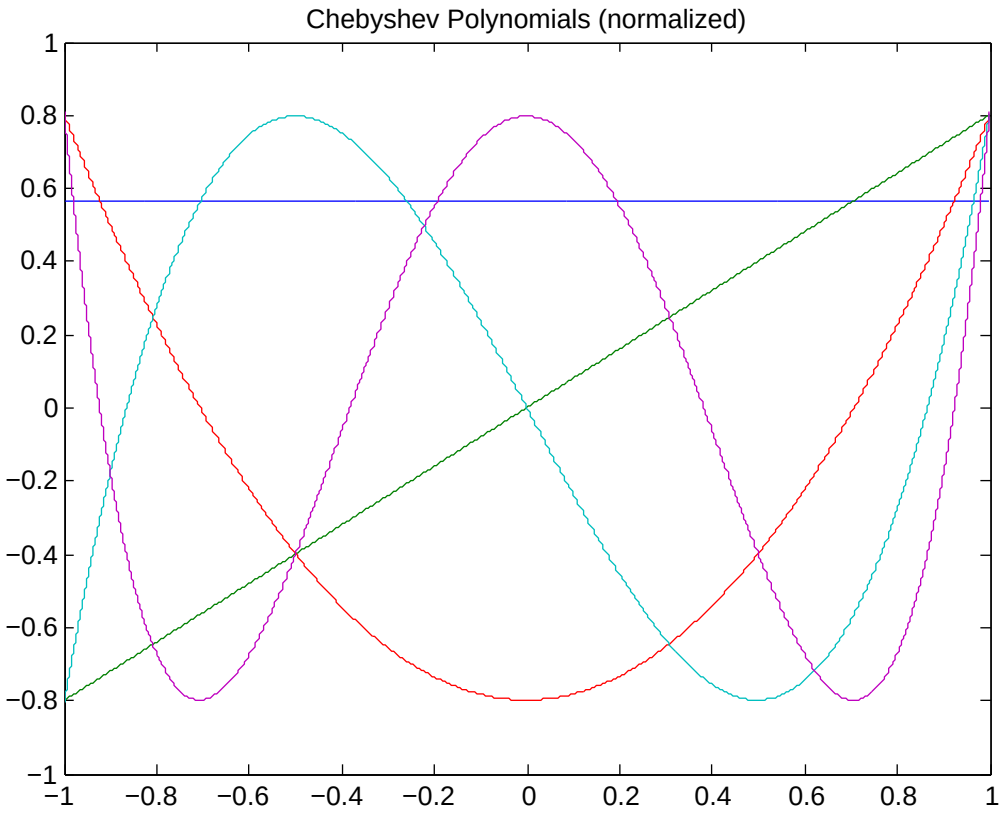
MATLAB CODE

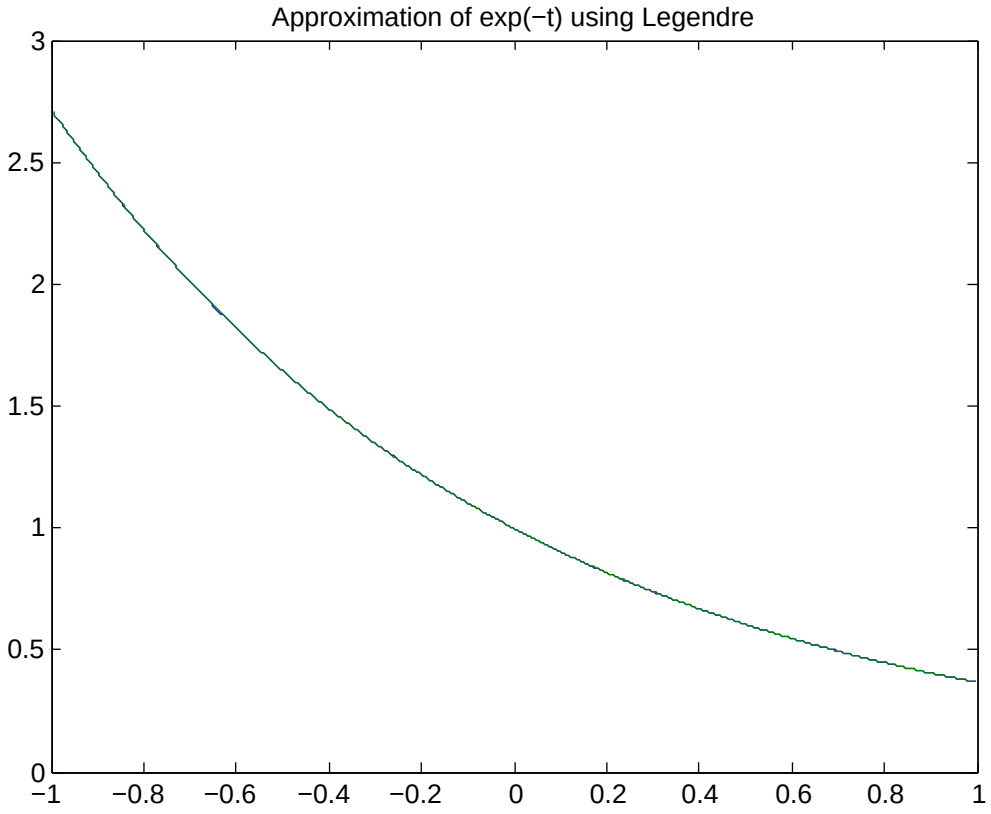
```
%
%    Hw #3
%    Problem #1
%
%    ECEn 671, Fall 2010
%    Professor Neal K. Bangerter
%

clear;
close all;

% set up our linspace for t
n = 10000;
t = linspace(-0.9999, 0.9999, n);  % 0.9999 to avoid problem with weighting
for Chebyshev
dt = t(2) - t(1);
for kk = 1:5
    p(:, kk) = t.^(kk-1);  % generate our original p vectors 1, t, t^2, etc.
end

%
% now generate the Legendre polynomials through Gram-Schmidt
%

% initialize e to p and set up q.
e = p;
q = zeros(n, 5);

% normalize the first legendre polynomial
q(:,1) = e(:,1)/sqrt(sum(e(:,1).*e(:,1))*dt);  % sqrt(sum(x.*x)*dt)
approximates norm!

% now compute the rest
for kk = 2:5
    for ll = 1:kk-1
        e(:,kk) = e(:,kk) - sum(p(:,kk).*q(:,ll))*dt*q(:,ll);
    end
    q(:,kk) = e(:,kk)/sqrt(sum(e(:,kk).*e(:,kk))*dt);
end

% now plot the Legendre polynomials (normalized)
figure;
plot(t, q(:,1), t, q(:,2), t, q(:,3), t, q(:,4), t, q(:,5));
title('Legendre Polynomials (normalized)');


%
% now generate the Chebyshev polynomials through Gram-Schmidt
%

% initialize e to p and set up q_c.
e = p;
q_c = zeros(n, 5);

% initialize the weighting
w = 1./sqrt(1-t.^2);
```

```
% normalize the first Chebyshev polynomial
q_c(:,1) = e(:,1)/sqrt(sum(e(:,1).*w'.*e(:,1))*dt);  % sqrt(x'*x*dt)↵
approximates norm!

% now compute the rest
for kk = 2:5
    for ll = 1:kk-1
        e(:,kk) = e(:,kk) - sum(p(:,kk).*w'.*q_c(:,ll))*dt*q_c(:,ll);
    end
    q_c(:,kk) = e(:,kk)/sqrt(sum(e(:,kk).*w'.*e(:,kk))*dt);
end

% now plot the Chebyshev polynomials (normalized)
figure;
plot(t, q_c(:,1), t, q_c(:,2), t, q_c(:,3), t, q_c(:,4), t, q_c(:,5));
title('Chebyshev Polynomials (normalized)');


%
% Now do the fits to both the Legendre and Chebyshev polynomials
%
x = exp(-t)';

% Compute cross correlation matrices for both Legendre and Chebyshev
p_l = zeros(5,1);
p_c = zeros(5,1);
for kk = 1:5
    p_l(kk) = sum(x.*q(:,kk))*dt;
    p_c(kk) = sum(x.*q_c(:,kk))*dt;
end

% Compute the Grammian R for both Legendre and Chebyshev cases
R_l = zeros(5);
R_c = zeros(5);
for kk = 1:5
    for ll = 1:5
        R_l(kk,ll) = sum(q(:,kk).*q(:,ll))*dt;
        R_c(kk,ll) = sum(q_c(:,kk).*q_c(:,ll))*dt;
    end
end

% Now compute the optimal coefficients c for each case
c_l = (R_l^-1)*p_l;
c_c = (R_c^-1)*p_c;

% Now plot each of the cases
figure;
plot(t, x, t, c_l'*q');
title('Approximation of exp(-t) using Legendre');

figure;
plot(t, x, t, c_c'*q_c');
title('Approximation of exp(-t) using Chebyshev');

% Compute the norms of the error vectors in each case
```

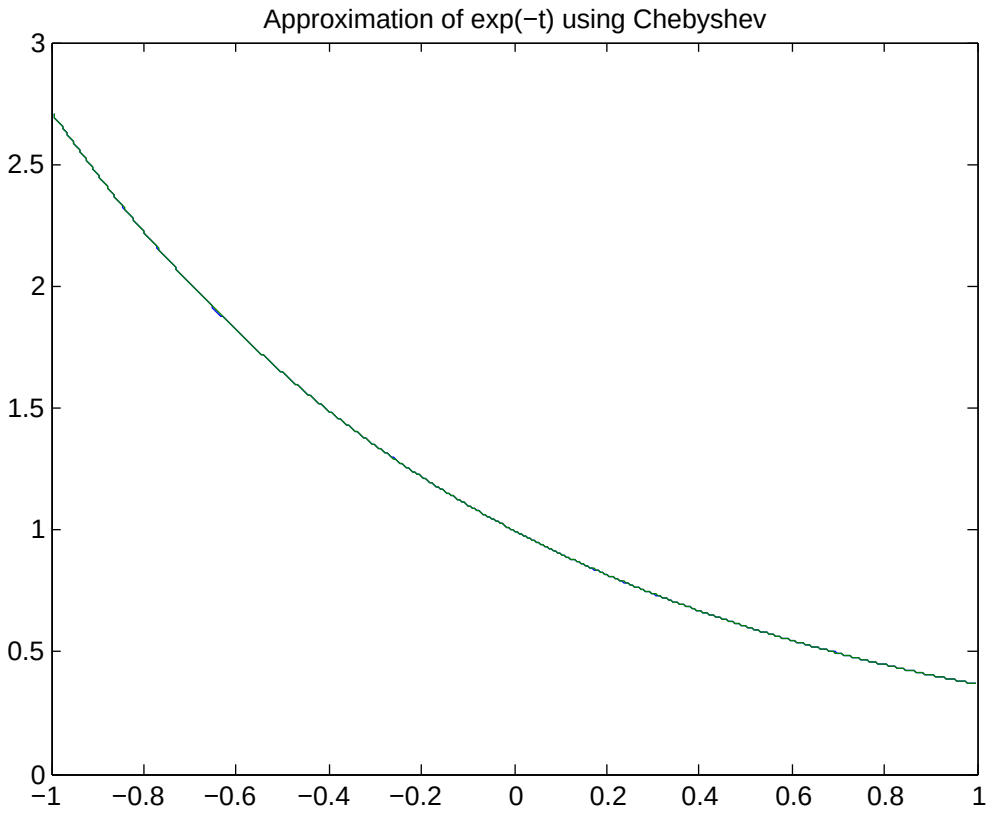
```
error_l = x' - c_l'*q';
error_c = x' - c_c'*q';

norm_error_l = sqrt(sum(error_l.*error_l)*dt);
norm_error_c = sqrt(sum(error_c.*error_c)*dt);

cond_R_l = cond(R_l);
cond_R_c = cond(R_c);
```


Legendre Polynomials (normalized)
2.5
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1


Chebyshev Polynomials (normalized)
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1



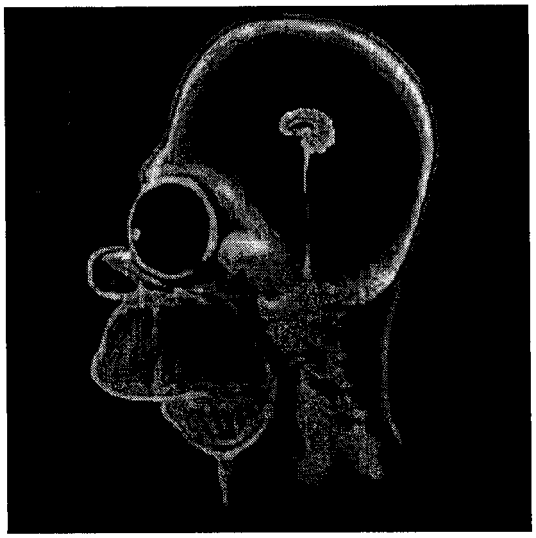

```
Norm of error is 0.00047
Condition number of R is 1.0
```



Norm of error is 0.7771
Condition number of R is 4.6,
so we expect more error than
in the Legendre case where R
was 1.0. (But only because
we are doing this numerically!
If we weren't using finite
precision arithmetic, we would
expect the error to be
identical since both the
Legendre and Chebyshev polynomials
span the same space!)

```
%
%   Homework #3:  Problem #2 solution
%

clear;
close all;

%   Load in the data
load('prob2.mat');

%   Display the original x
figure; imshow(x);

%   Reshape A and x into column vectors
A = reshape(A, 256^2, 20);
x = reshape(x, 256*256, 1);

%   Take the projection of x onto A
tmp = (A'*A);
tmp = tmp^(-1);
tmp = tmp*A'*x;
x_hat = A*tmp;
clear tmp;

%   Compute error e = x - x_hat
e = x - x_hat;

%   Scale and reshape e
e = e/max(e);
e = reshape(e, 256, 256);

%   And display e
figure; imshow(e);
```

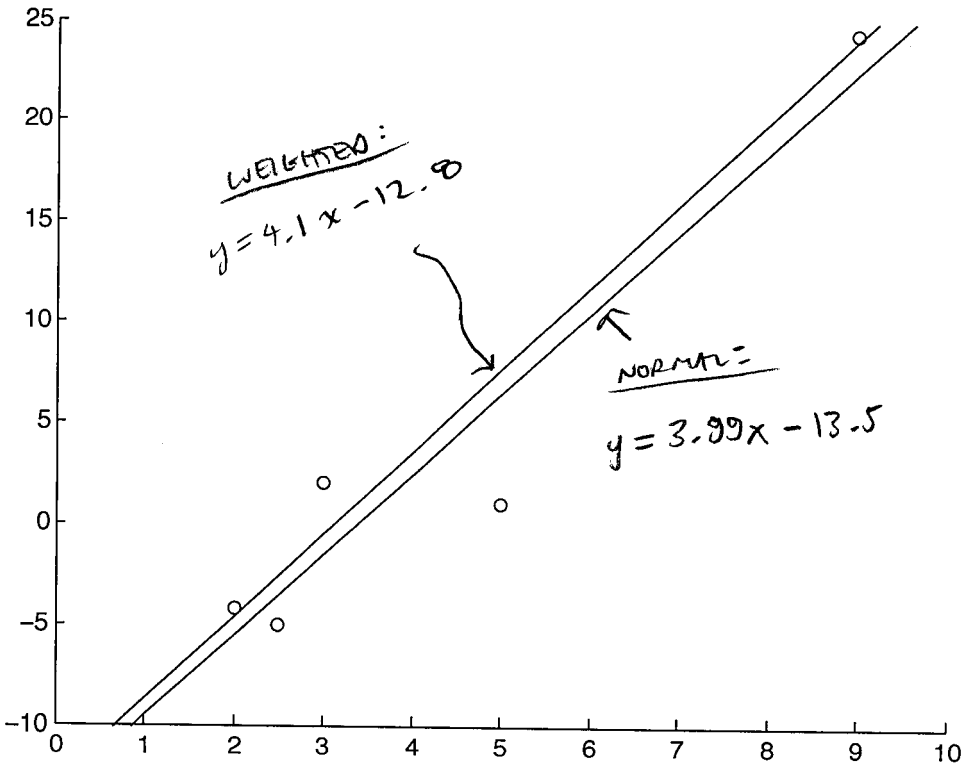
PROBLEM #2



THE PERFECT ORIGINAL IMAGE IS NOT RECOVERED BECAUSE THE ORIGINAL IMAGE WAS NOT ORTHOGONAL TO ALL OF OUR BASIS IMAGES. WE HAVE RECOVERED AN IMAGE THAT IS ORTHOGONAL TO EACH OF THE BASIS IMAGES, SO WE HAVE LOST THE PART OF HOMER THAT WAS SPANNED BY OUR BASIS SET.

PROBLEM 3.8-3


```
%
%   Homework #3:  Problem 3.8-3 solution
%

x = [2 2.5 3 5 9]';
y = [-4.2 -5 2 1 24.3]';

% Part (a) - plot the data
figure;
scatter(x,y);
axis([0 10 -10 25]);

% Part (b) - determine best least-squares line
A = [x ones(length(x),1)];
c = (A'*A)^(-1)*A'*y;

hold on;
x_ls = 0:10;
y_ls = c(1)*x_ls + c(2);
plot(x_ls, y_ls);

% Part (c) - determine a weighted best least-squares line and plot
W = diag([10 1 1 1 10]);
c_w = (A'*W*A)^(-1)*A'*W*y;

y_wls = c_w(1)*x_ls + c_w(2);
plot(x_ls, y_wls);
```



3.8-6:

FORMULATE $y \approx c e^{ax}$ AS A LINEAR REGRESSION PROBLEM.

TAKE LOG:

$$\ln y = \ln(c e^{ax})$$

$$\ln y = \ln(c) + ax$$

$$\begin{bmatrix} \ln y_1 \\ \ln y_2 \\ \vdots \\ \ln y_m \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_m \end{bmatrix} \begin{bmatrix} \ln c \\ a \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{bmatrix}$$

ONCE $\ln c$ AND a ARE FOUND,

$y \approx c e^{ax}$ CAN BE FOUND!

3.8-10:

THE APPROXIMATION ERROR E IS GIVEN BY:

$$E = Y - \sum_{i=1}^{m} c_i X_i$$

BY THE ORTHOGONALITY PRINCIPLE, WE WANT:

$$\langle E, X_j \rangle = 0 \quad \text{FOR} \quad j = 1, 2, \ldots, m$$

OR:

$$\left\langle Y - \sum_{i=1}^{m} c_i X_i \, , \, X_j \right\rangle = 0 \quad \text{FOR } j = 1, 2, \ldots, m$$

$$\langle Y, X_j \rangle - \left\langle \sum_{i=1}^{m} c_i X_i, X_j \right\rangle = 0 \quad , \; j = 1, 2, \ldots, m$$

$$\sum_{i=1}^{m} c_i \langle X_i, X_j \rangle = \langle Y, X_j \rangle \quad , \; j = 1, 2, \ldots, m$$

$$\sum_{i=1}^{m} c_i \, tr\left(X_i X_j^H\right) = tr\left(Y X_j^H\right) \quad , \; j = 1, 2, \ldots, m$$

WE CAN WRITE IN MATRIX FORM AS:

$$\begin{bmatrix} tr(X_1 X_1^H) & tr(X_2 X_1^H) & \cdots & tr(X_m X_1^H) \\ tr(X_1 X_2^H) & & & \\ \vdots & & & \\ tr(X_1 X_m^H) & & \cdots & tr(X_m X_m^H) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix} = \begin{bmatrix} tr(Y X_1^H) \\ tr(Y X_2^H) \\ \vdots \\ tr(Y X_m^H) \end{bmatrix}$$

3.9-12:

$f = \{1, 1, 2, 3, 5, 8, 13\}$ $\quad m = 2$

a) COVARIANCE METHOD:

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \\ 13 & 8 \end{bmatrix} \qquad R = A^H A = \begin{bmatrix} 1 & 2 & 3 & 5 & 8 & 13 \\ 1 & 1 & 2 & 3 & 5 & 8 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \\ 13 & 8 \end{bmatrix} = \begin{bmatrix} 272 & 168 \\ 168 & 104 \end{bmatrix}$$

AUTOCORRELATION METHOD:

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \\ 13 & 8 \\ 0 & 13 \end{bmatrix} \qquad R = A^H A = \begin{bmatrix} 1 & 1 & 2 & 3 & 5 & 8 & 13 & 0 \\ 0 & 1 & 1 & 2 & 3 & 5 & 8 & 13 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \\ 13 & 8 \\ 0 & 13 \end{bmatrix} = \begin{bmatrix} 273 & 168 \\ 168 & 273 \end{bmatrix}$$

b) WE HAVE:

$n =$	1	2	3	4	5	6	7
$f[n] =$	1	1	2	3	5	8	13

WE WANT:

$d[n] =$	2	3	5	8	13

SINCE WE ONLY HAVE ENOUGH REMAINING DATA FOR $d[2]$ TO $d[6]$, WE HAVE TO USE AN ABBREVIATED A MATRIX FOR THE COVARIANCE METHOD.

COVARIANCE METHOD:

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \end{bmatrix} \qquad R = A^H A = \begin{bmatrix} 1 & 2 & 3 & 5 & 8 \\ 1 & 1 & 2 & 3 & 5 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \end{bmatrix} = \begin{bmatrix} 103 & 64 \\ 64 & 40 \end{bmatrix}$$

$$\underline{d} = \begin{bmatrix} 2 \\ 3 \\ 5 \\ 8 \\ 13 \end{bmatrix}$$

$$\underline{c} = (A^H A)^{-1} A^H \underline{d} = R^{-1} A^H \underline{d}$$

$$\underline{c} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$ ⟸ VERY GOOD PREDICTOR FOR FIBONACCI!

AUTOCORRELATION METHOD:

THIS IS GOING TO PERFORM VERY POORLY, SINCE THE ASSUMPTION THAT ANY DATA WE DON'T HAVE IS ZERO IS A BAD ONE...

WE ASSUME: $f[n]$ =

$n=$	0	1	2	3	4	5	6	7	8	9
	0	1	1	2	3	5	8	13	0	0

AND: $d[n]$ =

	1	2	3	5	8	13	0	0

WE HAVE: (LIKE IN PART (a))

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 2 & 1 \\ 3 & 2 \\ 5 & 3 \\ 8 & 5 \\ 13 & 8 \\ 0 & 13 \end{bmatrix}$$

$$R = A^H A = \begin{bmatrix} 273 & 168 \\ 168 & 273 \end{bmatrix}$$

$$\underline{d} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 5 \\ 8 \\ 13 \\ 0 \\ 0 \end{bmatrix}$$

$$\underline{c} = (A^H A)^{-1} A^H \underline{d} = R^{-1} A^H \underline{d}$$

$$\underline{c} = \begin{pmatrix} 0.6132 \\ 0.0036 \end{pmatrix}$$

c) THE MINIMUM LEAST-SQUARES ERROR IS GIVEN BY TAKING $\|\underline{e}\|^2$ WHERE:

$$\underline{d} = A\,\underline{c} + \underline{e}$$

So:

$$\underline{e} = \underline{d} - A\,\underline{c}$$

↑ WE COMPUTED THESE ABOVE.

COVARIANCE METHOD:

$$\|\underline{e}\|^2 = \|\underline{d} - A\,\underline{c}\|^2 = \boxed{0}$$

AUTOCORRELATION METHOD:

$$\|\underline{e}\|^2 = \|\underline{d} - A\,\underline{y}\|^2 = \boxed{168.6}$$