ECE 5412 Yucheng Chen Assignment 2

37. Convert the AR model into state space form:

Then, we compute the eigenvalues of the matrix , then we solve the equation

.

The eigenvalues of the matrix are .

To make sure the system is asymptotically stationary, we have .

If , meaning that are complex conjugate, then we have

Then we have

.

If , meaning that are real.

If , then we have , then we have

If , then we have , then we have



38. Choose

The MATLAB code is as follows

clc,clear,close all

a1 = -0.5;

a2 = 0.5;

N = 10000;

y = [];

y0 = 1;

y1 = 4;

ys = [y0,y1];

for i = 3:N

y = -a1\*ys(i-1)-a2\*ys(i-2) + randn(1);

ys = [ys,y];

end

theta0 = [0.8,0.8]';

rho = 0.999;

c = 2;

theta = theta0;

P = eye(2);

errs = [];

for i = 3:N

phi = [ys(i-1);ys(i-2)];

theta = theta + P\*phi/(rho/c + phi'\*P\*phi)\*(ys(i) - phi'\*theta);

P = 1/rho\*(P - P\*(phi\*phi')\*P/(rho/c + phi'\*P\*phi));

err = norm([theta(1)-a1,theta(2)-a2]);

errs = [errs,err];

end

plot(errs,'\*')

The error of the estimated and the true parameters are show below:



The algorithm seems to be very robust to different parameter initializations. Very different parameters end up with very close parameter estimation.

39. Using the Recursive Least Square estimator developed from problem 38. We iteration the model dimension from 1 to 20, then the corresponding AIC score of the dimension is shown as the following plot:



Consider that ,where is the number of data points, is likelihood of all data points and is the dimension of the model.

The BIC score versus the model dimension is shown as the following plot:



Both AIC and BIC scores tell that the model dimension estimate is 2.

The MATLAB code is shown as below:

clc,clear,close all

a1 = 0.5;

a2 = -0.5;

N = 10000;

y = [];

y0 = 1;

y1 = 4;

ys = [y0,y1];

for i = 3:N

y = a1\*ys(i-1)+a2\*ys(i-2) + randn(1);

ys = [ys,y];

end

dims = 1:20;

aics = [];

for d = dims

aic = computeAIC(ys,d,N);

aics = [aics,aic];

end

figure(1)

plot(aics,'\*')

xlabel('Model dimension')

ylabel('AIC score')

bics = [];

for d = dims

bic = computeBIC(ys,d,N);

bics = [bics,bic];

end

figure

plot(bics,'\*')

xlabel('Model dimension')

ylabel('BIC score')

function aic = computeAIC(ys,d,N)

theta0 = rand(1,d)';

rho = 0.999;

c = 2;

theta = theta0;

P = eye(d);

for i = (d+1):N

phi = [];

for j = 1:d

phi = [phi;ys(i-j)];

end

theta = theta + P\*phi/(rho/c + phi'\*P\*phi)\*(ys(i) - phi'\*theta);

P = 1/rho\*(P - P\*(phi\*phi')\*P/(rho/c + phi'\*P\*phi));

end

ysim = ys(1:d);

for i = (d+1):N

phi = [];

for j = 1:d

phi = [phi;ys(i-j)];

end

y = theta'\*phi;

ysim = [ysim,y];

end

aic = norm(ys-ysim) + 2\*d;

end

function bic = computeBIC(ys,d,N)

theta0 = rand(1,d)';

rho = 0.999;

c = 2;

theta = theta0;

P = eye(d);

for i = (d+1):N

phi = [];

for j = 1:d

phi = [phi;ys(i-j)];

end

theta = theta + P\*phi/(rho/c + phi'\*P\*phi)\*(ys(i) - phi'\*theta);

P = 1/rho\*(P - P\*(phi\*phi')\*P/(rho/c + phi'\*P\*phi));

end

ysim = ys(1:d);

for i = (d+1):N

phi = [];

for j = 1:d

phi = [phi;ys(i-j)];

end

y = theta'\*phi;

ysim = [ysim,y];

end

bic = log(N)\*d + 2\*(N/2\*log(2\*pi)+1/2\*norm(ys-ysim));

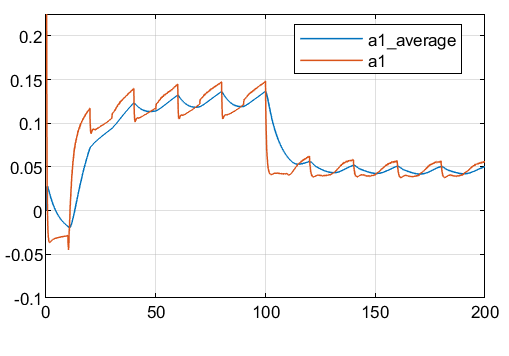
end

43. According to the example, the engine is modeled as follows:

Then the three parameters are estimated as .

The inertial is not explicitly included in the three parameters but we can observe the change of inertial by inspecting the change parameters.

We can observe the following the plot that is changing for the first 100 seconds and becomes a constant afterwards. This observation indicates that the inertia changes for the first 100 seconds, which sis as expected.



46.

Plug in , then we have

Plug in

I used a dataset of which has 4 columns and computed the variance estimate with number of constraints from 3 to 0. The results are shown in the following plot:



We can observe that as the number of constraints increases, the estimated variance of the model decreases.

47. The procedure is as follows:

Step 1 : Suppose is a matrix of K vectors of face images

Step 2: Compute and use PCA to compute matrices  such that and are diagonal.

Step3: is the first most significant eigenfaces. Then the features of the training faces are .

Step 4: Compute the feature of test image

Step 5:

I used 9 faces of different people as training set and another face image of the 9th person as the test case. Follow the described procedure above, the MATLAB code and final results are shown as below:

clc,clear,close all

addpath('faces/')

faces\_matrix = [];

for i = 1:9

im = imread(strcat('train',int2str(i),'.jpg'));

im = rgb2gray(im);

im = imresize(im,[45,45]);

im\_vec = double(im(:));

faces\_matrix = [faces\_matrix,im\_vec];

end

test\_im = rgb2gray(imread('test9.jpg'));

test\_im = imresize(test\_im,[45,45]);

faces\_matrix = faces\_matrix- mean(faces\_matrix,2);

test\_vec = double(test\_im(:)) - mean(faces\_matrix,2);

[U,D,V] = svd(faces\_matrix);

Ur = U(:,1:9);

features = D\*V';

features = features(1:9,1:9);

new\_feature = (test\_vec'\*Ur)';

errs = [];

for i = 1:9

face\_feature = features(:,i);

error = norm(face\_feature - new\_feature);

errs = [errs,error];

end

plot(errs,'\*');

xlabel('face # in training set')

ylabel('Sqaured Error');



The 9th sample has the least error, which is consistent with the expectation.s