Università
della
Svizzera
italiana

Faculty
of Informatics

May 29, 2018

Stochastic Methods

Assignment 5

Sumeet Gyanchandani

1 Plotting True Affiliations

The following plot depicts true affiliations of time-series to one of the 3 distinct clusters given by colors red, blue and green.

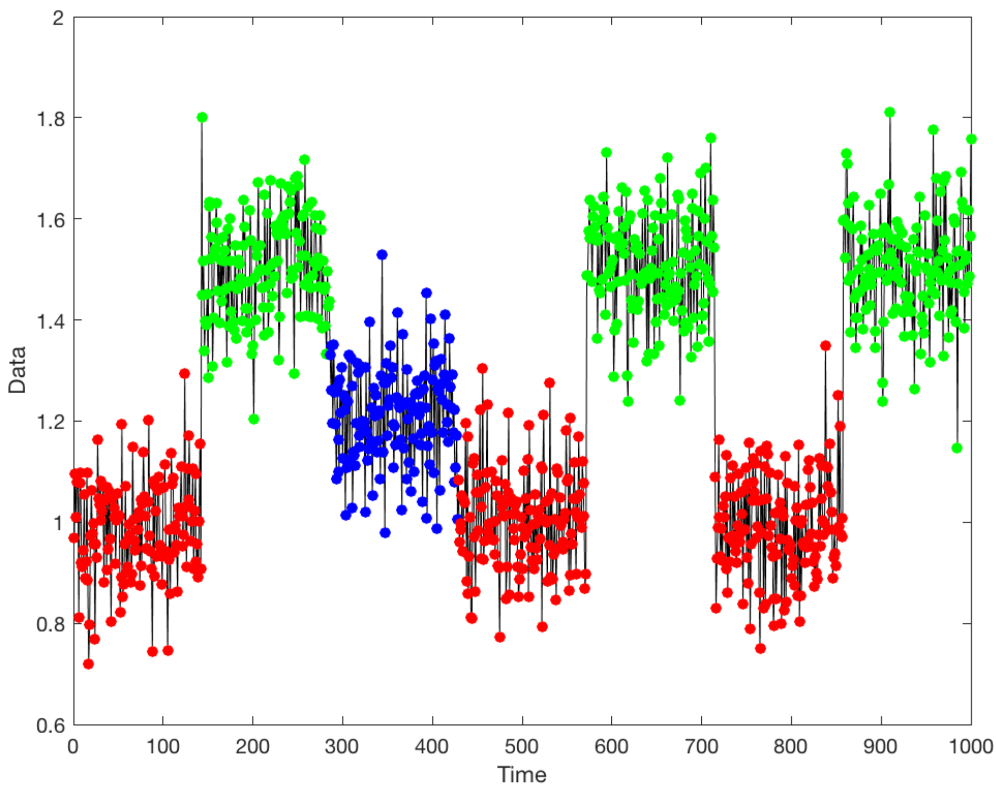

Figure 1. Clustered time-series

2 Standard K-Means Clustering

The following is the plot of the estimated affiliation vector IDX.

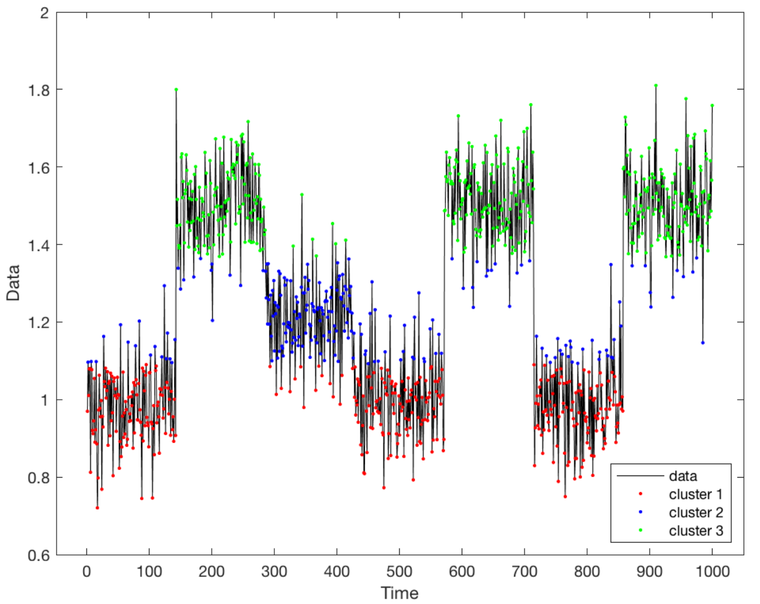

Figure 2. Estimated clusters

For the given sample, the clustering is decent, except for the boundary points between two clusters.

3 K-Means-H1 Method

The following is the result of the K-Means using H1 regularizations.

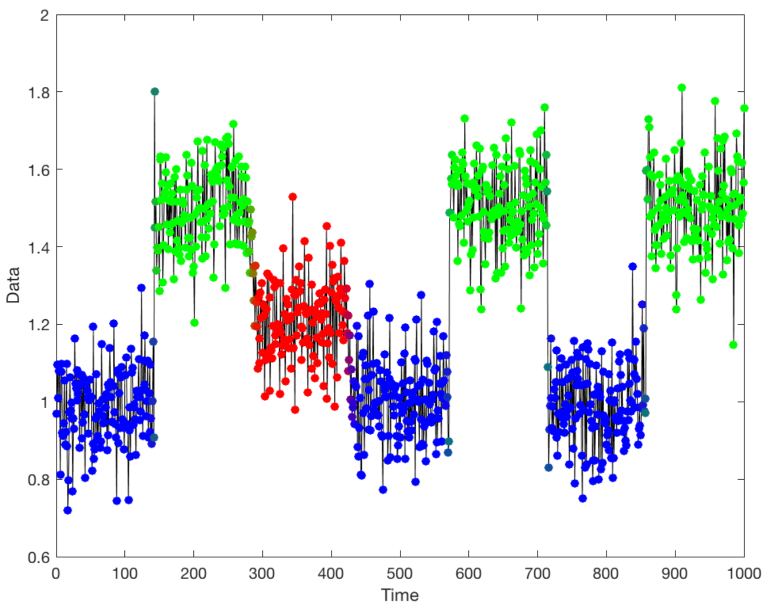

Figure 3. Resulting clusters

The resulting clusters of H1 regularization are more accurate than the standard K-Means clustering.

Listing 1. h1reg.m

```
clear all;
close all;

%Loading data
load('data_sample');

buffer(:,1) = cell2mat(gamma_sol(:,1));
buffer(:,2) = cell2mat(gamma_sol(:,2));
buffer(:,3) = cell2mat(gamma_sol(:,3));

%Parameters
[T,K] = size(buffer);
epssqr = 10;
epsilon = sqrt(epssqr);
maxiters = 1000;

%True Affiliation
figure;
plot(x,'k');
hold on;
y = [1:T];
pointsize = 30;
scatter(y,x,pointsize,buffer,'filled');
xlabel('Time');
ylabel('Data');
hold off;

%Standard K-Means
rng(3);
[IDX, Centroids] = kmeans(x', 3);
figure;
plot(x,'k');
hold on;
gscatter(y,x,IDX,'rbg','...');
legend('data','cluster 1','cluster 2','cluster 3','Location','southeast');
xlabel('Time');
ylabel('Data');
hold off;

% K-Means-H1
dimensions = size(x,1);
C = randn(K, dimensions);
maxiters = 1000;
distances = zeros(T, K);
gamma = zeros(T*K, 1);

% Creating Hessian matrix
Hblock = 2*diag([0.5;ones(T-2,1);0.5]) - diag(ones(T-1,1),1) - diag(ones(T-1,1),-1);
H = sparse(K*T,K*T);
for k = 1:K
H((k-1)*T+1:k*T,(k-1)*T+1:k*T) = Hblock;
end
H = epssqr*H;

BE = zeros(T,K*T);
for k = 1:K
BE(:,(k-1)*T+1:k*T) = eye(T);
end
CE = ones(T,1);

lb = zeros(T*K,1);
ub = ones(T*K,1);
```

```
L = Inf; change = Inf;
it = 0;
X = x';

while it < maxiters && change > epsilon
it = it + 1;
f = zeros(T*K,1);
for t = 1:T
for k = 1:K
f((k-1)*T+t) = distance(C(k,:), X(t,:));
end
end

for k = 1:K
distances(1:T,k) = f(k*T);
end

options = optimoptions('quadprog','Algorithm','interior-point-convex','Display','off');
[gamma,~,~,~,lambda] = quadprog(H,f,[],[],BE,CE,lb,[],[],options);
for k = 1 : K
gammak = gammak_block(gamma,T,k);

if sum(gammak) == 0
C(k,:) = zeros(1,dimensions);
else
for j = 1:dimensions
C(k,j) = dot(gammak,X(:,j))/sum(gammak);
end
end
end

temp = L;
L = compute_L(gamma,T,distances);
change = abs(L - temp);
end

%Plotting resulting affiliations
for k=1:K
buffer2(:,k) = gammak_block(gamma,T,k);
end

figure;
plot(x,'k');
hold on;
y = [1:T];
scatter(y,x,pointsize,buffer2,'filled');
xlabel('Time');
ylabel('Data');
hold off;

function gammak = gammak_block(gamma,T,k)
gammak = gamma((k-1)*T+1:k*T);
end

function L = compute_L(gamma,T,distances)
K = length(gamma)/T;
L = 0;
for k=1:K
gammak = gammak_block(gamma,T,k);
L = L + dot(gammak,distances(:,k));
end
end
```