ECE 532 - HW5 - Fall 2017 - Rebecca Willett

Table of Contents

Homework Assignment 5

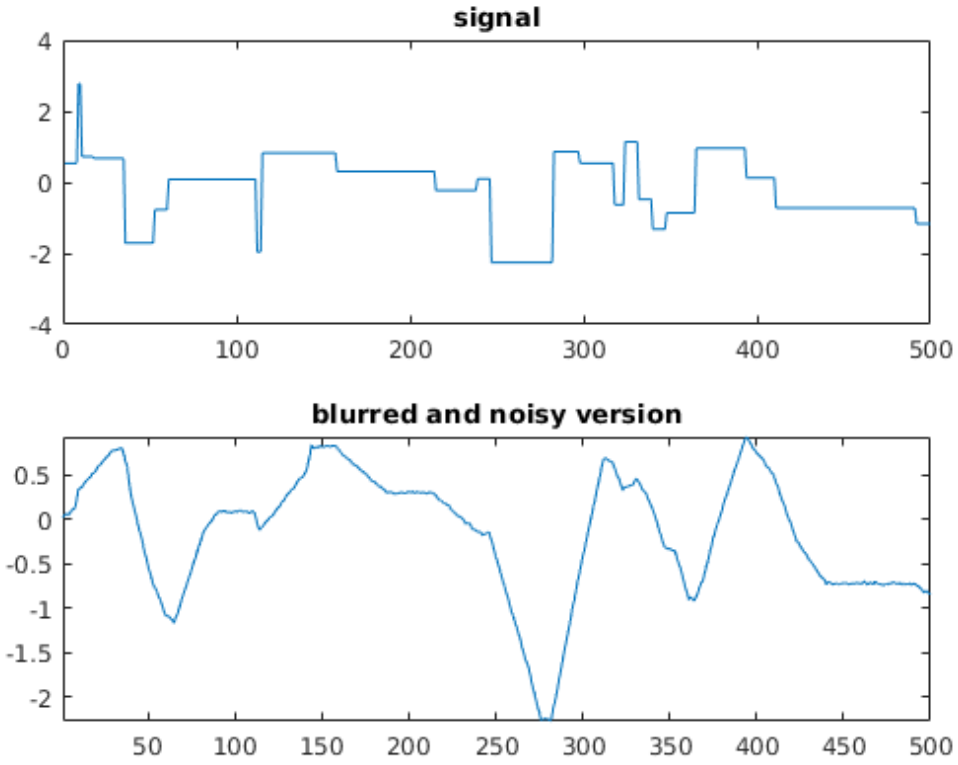
Completed by Roumen Guha

```
close all
clear all
```

Problem 2

```
rng(0) % initialize random seed

n = 500;
k = 30;
sigma = 0.01;

% generate random piecewise constant signal
w = zeros(n,1);
w(1) = randn;
for i = 2:n
    if (rand < 0.95)
        w(i) = w(i - 1);
    else
        w(i) = randn;
    end
end

% generate k-point averaging function
h = ones(1, k)/k;

% make X matrix for blurring
m = n+k-1;
for i = 1:m
    if i <= k
```

```
        X(i, 1:i) = h(1:i);
    else
        X(i, i-k+1:i) = h;
    end
end
X = X(:, 1:n);

% blurred signal + noise
y = X*w + sigma*randn(m, 1);

% plot
figure(1)
subplot(211)
plot(w)
title('signal')

subplot(212)
plot(y(1:n))
axis('tight')
title('blurred and noisy version')
```



(a)

```
errorRate = @(y, y_hat) sum((y - y_hat).^2);

[U, S, V] = svd(X, 'econ');
```

(i) Standard Least-Squares

```
w_hat = (X' * X) \ X' * y;
y_hat = X * w_hat;

errorRate_leastSquares = errorRate(y, y_hat)
```

```
errorRate_leastSquares =

    0.0032
```

(ii) Truncated Singular Value Decomposition (SVD)

```
w_hat = (V * inv(S) * U') * y;
y_hat = X * w_hat;

errorRate_truncatedSVD = errorRate(y, y_hat)
```

```
errorRate_truncatedSVD =

    0.0032
```

(iii) Regularized Least-Squares (LS)

```
lambda = 0.0000001;

w_hat = (V / (S' * S + lambda * eye(500)) * S' * U') * y;
y_hat = X * w_hat;

errorRate_regularizedLS = errorRate(y, y_hat)
```

```
errorRate_regularizedLS =

    0.0032
```

(b)

It seems as though regularization works best when sigma and k are large, so when there is a lot of noise or blurring.

Published with MATLAB® R2017a