Primo report

Giacomo Longo e Roberta Tassara

14 Novembre 2019

1 Analisi di regressione

1.1 Introduzione

Il primo caso che analizzeremo riguarda la realizzazione di una stima mediante un approssimatore universale

$$h(x) = \sum_{i=0}^{p} c_i x^i$$

Tale approssimatore costruisce funzioni del tipo

$$h(x) = c_0$$

$$h(x) = c_1 x + c_0$$

$$h(x) = c_2 x^2 + c_1 x + c_o$$

La funzione ricercata deve minimizzare l'errore empirico

$$\min_{\underline{c}} \{\frac{1}{n} \sum_{i=1}^{n} (y_i - h(x_i))^2\} \qquad \underline{c} = \{c_0, ..., c_p\}$$

che rappresenta la somma delle distanze tra i dati osservati (y_i) e quelli della curva ottima ($h(x_i)$).
Scrivendo la funzione in forma matriciale, derivandola rispetto a c e ponendo il tutto = 0 si ottiene

$$\underline{c} = (X^T X)^+ X \underline{y}$$

che corrisponde al minimo globale

1.2 Analisi al variare di p con $y = 1 - x^2$

1.2.1 Parametri di simulazione

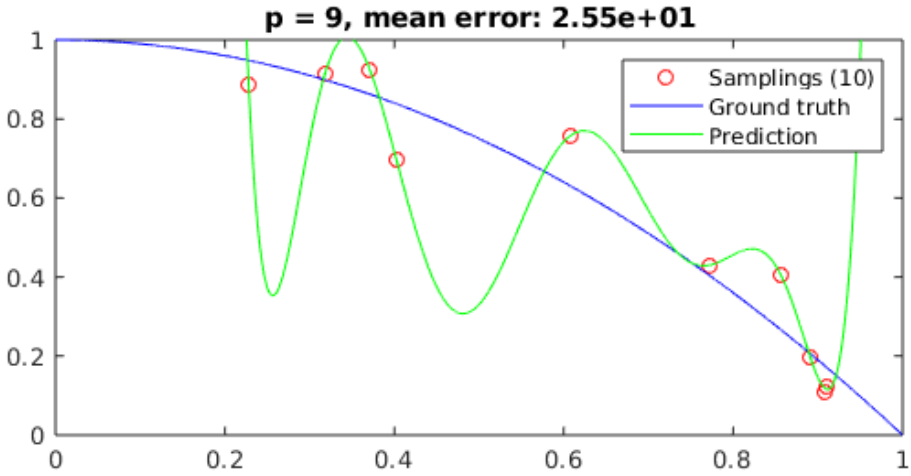
Dimensione dell'asse delle x: 10000 punti
Intervallo dell'asse delle x: $0 \leq x \leq 1$
Punti campione: 10
Intensitá di rumore: 10%

1.2.2 $p > n - 1$

L'analisi é stata omessa in quanto non rilevante: per tracciare n punti é sufficiente un polinomio di grado $n - 1$

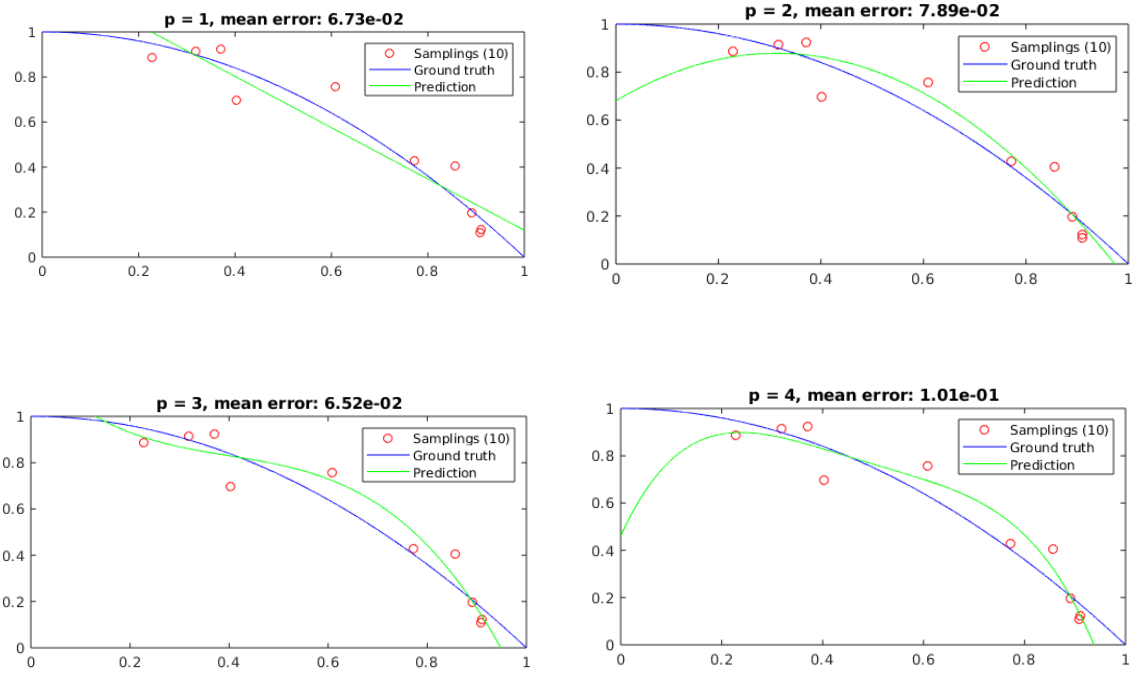
1.2.3 $p = n - 1$

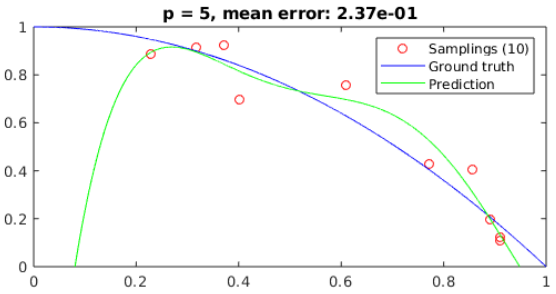
Se $p = n-1$ la funzione estimante(in verde) attraversa perfettamente tutti i punti campionati(in rosso), tuttavia non rappresenta una buona approssimazione della funzione originale (in blu). Infatti l'errore medio individuato é particolarmente alto.
Tale risultato, anche in prossimitá dei campioni con valori molto simili alla funzione reale, la funzione stimata si comporta in modo molto differente con vistose oscillazioni dovute ai termini di grado elevato.

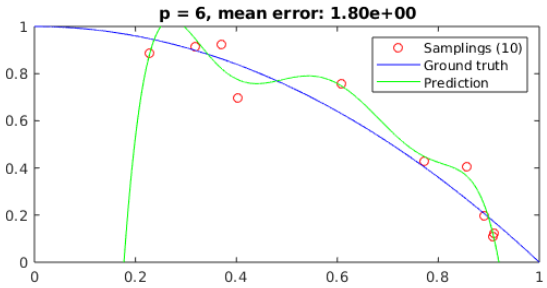


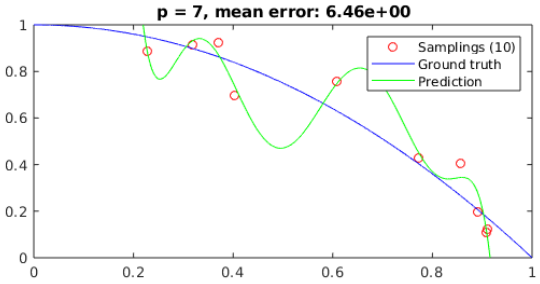
1.2.4 $p < n - 1$

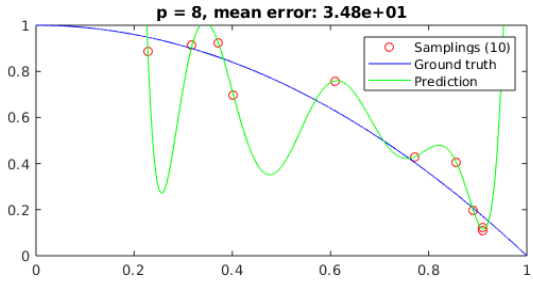
Per bassi valori di p, la funzione risultante é semplice.
Tra i vari valori di p, in questa particolare istanza, il valore che approssima meglio la funzione originale é $p = 3$ per il quale si ha il piú basso errore medio.
Questo significa che per questa funzione, l'errore di approssimazione é inferiore a quello di stima.
Colpisce anche che il miglior stimatore non coincida con il grado della funzione originaria, ció é dovuto al rumore.




p = 5, mean error: 2.37e-01
Samplings (10)
Ground truth
Prediction


p = 6, mean error: 1.80e+00
Samplings (10)
Ground truth
Prediction


p = 7, mean error: 6.46e+00
Samplings (10)
Ground truth
Prediction


p = 8, mean error: 3.48e+01
Samplings (10)
Ground truth
Prediction

1.3 Codice

```
%% Cleanup
clear % Clear variables
clc % Clean console
close all % Close all plots

%% Original function (ground truth)
num_true = 10000;
x_true = linspace(0,1,num_true); % Generation of the X axis
y_true = groundTruth(x_true);

figure
hold on
grid on

tiledlayout(3,3) % Arrange plots in a grid

%% Sampling from ground truth
num_samples = 10; % Number of points to sample from the function
sigma = 0.05; % Noise strength
x_sampling = rand(num_samples, 1); % Take num_samples points on the x-axis
y_sampling = sampleWithGaussianNoise(x_sampling, sigma); % Sample and add gaussian noise

%% Learning from sample data
% We now build a function like sum over i {c * x^i}
p = num_samples - 1; % Degree of the approximating polynomial (should be between 1 and num_samples - 1)
learn_X = zeros(num_samples, p+1); % Matrix initialization
for i = 1:p+1 % The logical choice here would be 0 to p but MATLAB doesn't like that
    learn_X(:,i) = x_sampling .^ (i - 1); % Calculation of x^i for all xs
end
for current_p = 1:p
    % Learning phase
    learn_X_current = learn_X(:,1:current_p+1); % Rescale matrix
    learn_C_current = (learn_X_current' * learn_X_current) \ (learn_X_current' * y_sampling); % Recalculate coefficients

    % Forward phase
    forward_X_current = zeros(num_true, current_p + 1);
    for i = 1:current_p+1
        forward_X_current(:,i) = x_true .^ (i - 1);
    end
    forward_Y_current = forward_X_current * learn_C_current;

    average_error = mean(abs(forward_Y_current - y_true'));

    nexttile;
    plot(x_sampling, y_sampling, 'ro', x_true, y_true, 'b', x_true, forward_Y_current, 'g-');
    axis([min(x_true), max(x_true), min(y_true), max(y_true)]) % Scale axis to ground truth function
    title(sprintf('p = %d, mean error: %.2e', current_p, average_error));
    legend(sprintf('Samplings (%d)', num_samples), 'Ground truth', 'Prediction');
end

%% Definitions
function y = groundTruth(x)
    y = 1 - x.^2;
end
function y = sampleWithGaussianNoise(x, sigma)
    y = groundTruth(x) + sigma * randn(size(x));
end
```