Linear prediction

Lab 13, SDP

Table of contents

1 Objective

Experiment with linear prediction for simple signals

2 Theoretical notions

Linear prediction represents the estimation of a sample of the signal $x[n]$ as a linear combination of N previous samples:

$$x[n] \approx a_1 x[n-1] + a_2 x[n-2] + ... + a_N x[n-n]$$

Signals that approximately follow such a relationship are called “autoregressive” (AR). N represents the order of the autoregressive model.

In Matlab, the function `lpc()` estimates the coefficients a_k (refer to the documentation).

An alternative and more accurate method is provided in the function `lpc_exact()` along with the laboratory work.

3 Matlab examples

3.1 Generating some simple signals

Linear signal:

```
x = linspace(0,20,50);
plot(x)
```

Sinusoidal signal:

```
n = 0:50;
f = 0.1;
x = sin(2*pi*f*n);
plot(x)
```

Exponential signal $a^n u[n]$:

```
n = 0:20;
x = (1/2).^m;
plot(x)
```

Noise:

```
x = randn(1,100);   % or use rand()
plot(x)
```

3.2 Computing LPC coefficients

```
x = 1:1:10
order = 5;
a = lpc(x, order);
```

3.3 Prediction using the LPC coefficients

```
x = 1:1:10
order = 5;
a = lpc(x, order);

% Predict value at n=11 and append to x:
x(11) = sum(x(n-1:-1:n-ordin) .* (-a(2:end)))
```

4 Theoretical exercises

1. Consider the system described by the following equation:

$$y[n] = 0.8y[n-1] + x[n] + x[n-1],$$

where $x[n]$ is a stationary random process with a mean of 0 and autocorrelation $\gamma_{xx}[m] = \left(\frac{1}{2}\right)^{|m|}$.

   a. Determine the power spectral density (PSD) of the output $y[n];
   b. Determine the autocorrelation function $\gamma_{yy}[m]$ of the output;
   c. Determine the variance σ_y^2 of the output.

5 Practical exercises

1. Predicție liniară pe un semnal liniar

   - Generați un semnal liniar crescător de 200 eșantioane, cu pantă constantă $\Delta = 0.5$, prima valoare fiind 5.

     Folosiți `linspace()` sau `start:step:stop`.

- Modelăm semnalul ca un proces autoregresiv de ordin 4, AR(4). Calculați coeficienții de predicție a_k cu funcția Matlab `lpc()`.
- Pe baza coeficienților de predicție, folosind relația de predicție, preziceți următoarele 200 eșantioane ale semnalului. Afișați întregul semnal rezultat (400 eșantioane)

  Puteți folosi o relație de forma `sum(x(n-1:-1:n-ordin) .* (-a(2:end)))`
- Utilizați funcția `lpc_exact()` în locul `lpc()`. Ce se observă ?
- Generați același semnal crescător cu lungime 400 eșantioane direct cu formula inițială. Afișați pe aceeași figură semnalul acesta și semnalul precedent (2 x 400 eșantioane).

  Ce calitate are porțiunea prezisă, comparativ?
- Schimbați ordinul modelului în AR(1), AR(2), AR(3), AR(10. Ce se observă ?

  Care este cel mai mic ordin pentru care predicția reușește?

2. Predicție liniară pe diverse semnale.

   Repetați ex. precedent pentru un semnal de forma:

   - Semnal exponențial: $x[n] = (0.9)^n u[n]$. Porniți de la un semnal de lungime 50, și estimați următoarele 50 eșantioane.
   - Semnal sinusoidal: $x[n] = 3 \cdot \sin(2 * \pi * f * n)u[n], f = 0.05$. Porniți de la un semnal de lungime 50, și estimați următoarele 50 eșantioane.
   - Sinusoidă exponențială: $x[n] = 0.8^n \cdot \sin(2 * \pi * f * n)u[n], f = 0.2$. Lungime 50 + 50
   - Semnal sinusoidal atenuat: $x[n] = \frac{\sin(2*\pi*f*n)}{2*\pi*f*n}u[n], f = 0.05$. Porniți de la un semnal de lungime 50, și estimați următoarele 50 eșantioane.
   - Semnal de tip zgomot alb gaussian (AWGN, generat cu `randn()`). $x[n] = 0.8^n \cdot sin(2 * \pi * f * n)u[n], f = 0.2$. Lungime 500 + 100. Apoi lungime 20 + 100.
   - Semnal de tip zgomot alb uniform (generat cu `rand()`). $x[n] = 0.8^n \cdot sin(2 * \pi * f * n)u[n], f = 0.2$. Lungime 500 + 100. Apoi lungime 20 + 100.
   - Semnal sinusoidal in zgomot alb: $x[n] = 2 \cdot sin(2 * \pi * f * n)u[n] + AWGN, f = 0.05$. Lungime 100 + 100.
   - Semnalul `mtlb` încărcat cu `load mtlb;`. Estimați următoarea secundă de semnal audio.
   - Primele 150 de eșantioane din semnalul `mtlb`. Estimați următoarea secundă de semnal audio.

3. Reducerea zgomotului prin predicție.

   Generați un semnal de forma:

   $x[n] = \sin(2 * \pi * f * n) +$ zgomot alb.

   Calculați coeficienții de predicție, și apoi estimați fiecare eșantion din semnalul $x[n]$ pe baza eșantioanelor precedente. Afișați semnalul astfel obținut ($x_2[n]$) cu semnalul original pe aceeași figură. Ce se observă?

4. Detecția vocii (Voice Activity Detector).

   - Încărcați semnalul audio `data_slow.wav` (cu `audioread()`), afișați-l grafic și redați-l audio.

   a. Utilizați funcția `buffer()` pentru a împărți semnalul în ferestre cu lungimea de aproximativ 25ms.
   b. Modelați fiecare segment semnalul ca un proces aleator AR(12), și găsiți coeficienții liniari de predicție pentru fiecare segment.
   c. Pentru fiecare segment, calculați energia coeficienților de predicție (suma coeficienților la pătrat). Afișați secvența de valori obținută.
   d. De pe grafic, alegeți un prag convenabil pentru a diferenția segmentele de voce de cele de pauză.
   e. Eliminați segmentele din semnal care sunt de pauză, și reuniți segmentele rămase întrun-un semnal întreg. Ascultați semnalul astfel obținut.

5. Repetați exercițiul anterior, dar adăugând peste semnalul inițial zgomot AWGN. Până la ce nivel de zgomot se obțin rezultate bune?

6. Incărcați imaginea `lena512.bmp`. Transformați în 0 valorile de pe liniile $20 : 30$, coloanele de la 100 la 150.

   Refaceți imaginea în felul următor: pentru fiecare linie separat, modelați primele 100 eșantioane cun un proces AR(10), apoi estimați cele 50 valori lipsa care urmează.

   Afișați imaginea astfel obținută.

6 Întrebări finale

1. TBD