Exercise on signal forecasting using an LTI model

Ivan Markovsky

1. *Prediction using a model*

   (a) *State space approach* Given a state space representation of a discrete-time autonomous system $\mathscr{B}(A,C)$ of order $\mathtt{n}$ and a finite, $T \geq \mathtt{n}$ samples long, trajectory

   $$y_{\mathrm{p}} := \big(y(1),\ldots,y(T)\big)$$

   of that system, find the next T_{f} samples

   $$y_{\mathrm{f}} := \big(y(T+1),\ldots,y(T+T_{\mathrm{f}})\big)$$

   of the given trajectory, *i.e.*, find $y(T+1),\ldots,y(T+T_{\mathrm{f}})$ such that

   $$y := \big(y(1),\ldots,y(T),y(T+1),\ldots,y(T+T_{\mathrm{f}})\big)$$

   is a trajectory of $\mathscr{B}(A,C)$.

   (b) *Polynomial approach* Solve the problem of 1a using a polynomial representation of the system $\mathscr{B}(P) = \mathscr{B}(A,C)$. Assume that the highest power coefficient of P is I.

   (c) *Simulation example* Implement your solutions of 1a and 1b in Matlab and apply them on the data in the file

   `http://homepages.vub.ac.be/ imarkovs/elec3035/tutorial2.mat`

   (use the command `load` to read the data) The file contains the following variables: `yp`, `a`, `c`, `p`, `Tf`, and `yp_noisy`, which correspond to col $\big(y_{\mathrm{p}}(1),\ldots,y_{\mathrm{p}}(T)\big)$, $y_{\mathrm{p}}(t) \in \mathbb{R}$, A, C, col$(p_n,\ldots,p_0)$, T_f, and a noise corrupted version of `yp` needed in 1d.

   (d) *Simulation example with noisy sequence* y_{p} Apply your solutions of 1a and 1b on the noise corrupted version of the data sequence y_{p}, stored in the variable `yp_noisy`. Compare the obtained predictions with the exact trajectory y_{f}, obtained in 1c.

Solution:

(a) *State space approach* A trajectory y of an autonomous system $\mathscr{B}(A,C)$ is completely specified by an initial condition $x(1)$, so the problem of predicting the future part y_{f} of the trajectory y from its given past y_{p} is equivalent to the problem of determining the initial condition $x(1)$ of y from y_{p}.

From the general expression of a response of an autonomous system

$$y(t_1) = CA^{t_1-t_2}x(t_2)$$

we have a system of equations for the unknown initial condition $x(1)$

$$\underbrace{\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(T) \end{bmatrix}}_{y_{\mathrm{p}}} = \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{T-1} \end{bmatrix}}_{\mathscr{O}_{\mathrm{p}}} x(1). \tag{1}$$

In order to be able to determine $x(1)$ uniquely from y_{p}, the matrix $\mathscr{O}_{\mathrm{p}}$, called an extended observability matrix of the system $\mathscr{B}(A,C)$, should have full column rank. Note that $\mathscr{O}_{\mathrm{p}}$ is $T\mathtt{p} \times \mathtt{n}$, where $\mathtt{p} = \text{row dim}(C)$

is the number of outputs. Under the assumption $T \geq \mathtt{n}$ (*i.e.*, "enough data is given") the matrix $\mathscr{O}_\mathrm{p}$ has the right dimension for being full column rank.

The condition $T\mathtt{p} \geq \mathtt{n}$ is necessary but not sufficient for $\mathscr{O}_\mathrm{p}$ to be full column rank. The extended observability matrix $\mathscr{O}_\mathrm{p}$ depends on the system parameters A and C, so an extra condition is needed on the matrices A and C, *i.e.*, on the state space representation of the system. This condition is so important that it is given the name *observability*.

Provided that we have enough data $T \geq \mathtt{n}/\mathtt{p}$ and the representation $\mathscr{B}(A,C)$ is observable, we can uniquely determine $x(1)$ from y_f via

$$x(1) = (\mathscr{O}_\mathrm{p}^\top \mathscr{O}_\mathrm{p})^{-1} \mathscr{O}_\mathrm{p} y_\mathrm{p}.$$

Then we predict y_f using $y(t) = CA^{t-1}x(1)$:

$$\underbrace{\begin{bmatrix} y(T+1) \\ \vdots \\ y(T_\mathrm{f}) \end{bmatrix}}_{y_\mathrm{f}} = \underbrace{\begin{bmatrix} CA^T \\ \vdots \\ CA^{T+T_\mathrm{f}-1} \end{bmatrix}}_{\mathscr{O}_\mathrm{f}} x(1).$$

The final answer is

$$y_\mathrm{f} = \mathscr{O}_\mathrm{f}(\mathscr{O}_\mathrm{p}^\top \mathscr{O}_\mathrm{p})^{-1} \mathscr{O}_\mathrm{p} y_\mathrm{p} = \mathscr{O}_\mathrm{f} \left(\sum_{\tau=0}^{T-1} CA^\tau (A^\tau)^\top C^\top \right)^{-1} \mathscr{O}_\mathrm{p} y_\mathrm{p}.$$

(b) *Polynomial approach*

In this case we consider the polynomial representation

$$P_0 y(t) + P_1 y(t+1) + \cdots + P_{\ell-1} y(t+\ell-1) + y(t+\ell) = 0, \qquad \text{for all } t \in \mathbb{Z}.$$

Because of the assumption that the highest power coefficient P_ℓ is I, we can find for each t, $y(t)$ as a linear combination of $y(t-1), \ldots, y(t-\ell)$

$$y(t) = -\big(P_0 y(t-\ell) + P_1 y(t-\ell+1) + \cdots + P_{\ell-1} y(t-1)\big). \tag{2}$$

Assuming that $T \geq \ell$, we can apply this formula recursively and "extend" y_p to y_f, *i.e.*, we simulate the response y_f corresponding to the initial conditions $y(T), y(T-1) \ldots, y(T-\ell+1)$, which is the end part of y_f. For this to be possible, we need $T \geq \ell$. It can be shown that this condition follows from the assumption $T \geq \mathtt{n}$.

(c) *Simulation example* Here is Matlab code for the prediction methods described in 1a and 1b.

sspredict.m

```
% Prediction of an scalar autonomous response, using a model
% State-space approach

function yf = ss_predict(yp,a,c,Tf)

T = length(yp);
n = size(c,2);

% Construct the extended observability matrix
O = c;
for i = 1:T+Tf-1
  O = [O; O(end,:)*a];
end

Op = O(1:T,:);
Of = O(T+1:end,:);

yf = Of * pinv(Op'*Op) * Op' * yp;
```

polpredict.m

```
% Polynomial approach

function yf = pol_predict(yp,p,Tf)

T = length(yp);
L = length(p) - 1;

y = yp;
for i = 1:Tf
  y(T+i) = - fliplr(p(2:end)) * y(T+i-L:T+i-1);
end
yf = y(T+1:end);
```

The obtained simulation result with the given exact data is shown in Figure 1, left. Up to numerical errors, the results obtained by the two methods are the same.

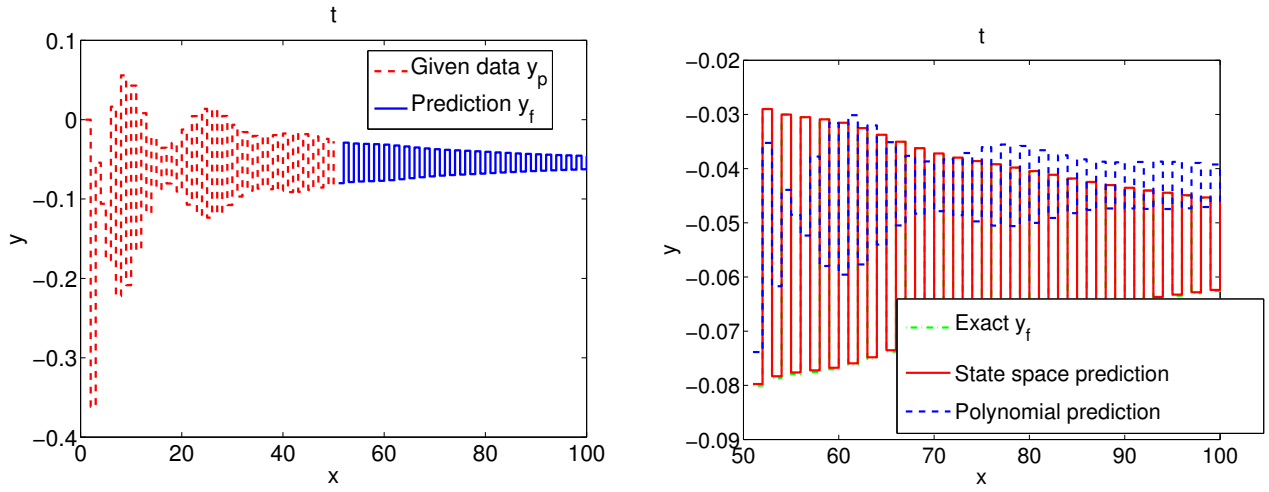

Figure 1: Simulation result for problems 1c and 1d.

(d) *Simulation example with noisy sequence* y_p

The predictions from noisy data are shown in Figure 1, right. The state space method gives much better result than the polynomial method. The explanation for this is that in 1c we use the whole sequence y_p in order to predict y_f, while in 1d we use only the last 16 samples (and ignore the other 34 samples). Note that in the noisy case, the system (1) is incompatible and we compute the least squares approximate solution. The least squares approximation is the reason for the good performance of the method of 1c. The polynomial approach can also be modified so that it uses all given data and does least squares approximation, so the difference in the performance of the two methods is not in the approach (polynomial vs state space) but in the particular way we solved the problem in the two cases.

□

2. *Autonomous system identification*

   (a) Given a scalar sequence

   $$y_{\mathrm{d}} := \big(y_{\mathrm{d}}(1), \ldots, y_{\mathrm{d}}(T)\big)$$

   and a natural number n, the aim of the considered identification problem is to find an autonomous system, such that the given sequence is a trajectory of that system. Under what conditions a solution exists. If a solution exists, when it is unique.

   (b) *Simulation example* Implement your solution in Matlab and apply it on the sequence

   $$y_{\mathrm{d}} = (1,2,4,7,13,24,44,81).$$

   What is the data generation rule?

   (c) *Prediction without model* Consider a prediction problem without a given model. Under what conditions it is possible to predict y_{f} from y_{p} (see Problem 1) without having a model? Implement your method in Matlab and test it on the exact and noisy sequences.

Solution:

(a) We aim to determine parameters $P_0, P_1, \ldots, P_\ell$, so that

$$P_0 y_{\mathrm{d}}(t) + P_1 y_{\mathrm{d}}(t+1) + \cdots + P_\ell y_{\mathrm{d}}(t+\ell) = 0, \qquad \text{for } t = 1, \ldots, T-\ell \tag{3}$$

holds. Using matrix vector notation, the system of equations (3) can be written in a matrix form as

$$\begin{bmatrix} P_0 & P_1 & \cdots & P_\ell \end{bmatrix} \begin{bmatrix} y_{\mathrm{d}}(1) & y_{\mathrm{d}}(2) & y_{\mathrm{d}}(3) & \cdots & y_{\mathrm{d}}(T-\ell) \\ y_{\mathrm{d}}(2) & y_{\mathrm{d}}(3) & y_{\mathrm{d}}(4) & \cdots & y_{\mathrm{d}}(T-\ell+1) \\ \vdots & \vdots & \vdots & & \vdots \\ y_{\mathrm{d}}(\ell+1) & y_{\mathrm{d}}(\ell+2) & y_{\mathrm{d}}(\ell+3) & \cdots & y_{\mathrm{d}}(T) \end{bmatrix} = 0. \tag{4}$$

Assuming that the highest power coefficient of P is I, we can rewrite (4) as

$$\underbrace{\begin{bmatrix} P_1 & P_2 & \cdots & P_\ell \end{bmatrix}}_{P} \underbrace{\begin{bmatrix} y_{\mathrm{d}}(2) & y_{\mathrm{d}}(3) & \cdots & y_{\mathrm{d}}(T-\ell+1) \\ y_{\mathrm{d}}(3) & y_{\mathrm{d}}(4) & \cdots & y_{\mathrm{d}}(T-\ell+2) \\ \vdots & \vdots & & \vdots \\ y_{\mathrm{d}}(\ell+1) & y_{\mathrm{d}}(\ell+2) & \cdots & y_{\mathrm{d}}(T) \end{bmatrix}}_{H} = -\underbrace{\begin{bmatrix} y_{\mathrm{d}}(1) & y_{\mathrm{d}}(2) & \cdots & y_{\mathrm{d}}(T-\ell) \end{bmatrix}}_{Y_{\mathrm{d}}}. \tag{5}$$

If (5) is solvable, then the autonomous system defined by

$$y(t) + P_1 y(t+1) + \cdots + P_{\ell-1} y(t+\ell-1) + y(t+\ell) = 0, \qquad \text{for all } t \in \mathbb{Z}$$

has as a trajectory the given sequence y_{d}. For uniqueness we need H to be full row rank. The identification procedure is to compute the solution of the system $PH = Y_{\mathrm{d}}$, *i.e.*,

$$P = Y_{\mathrm{d}} H^\top (HH^\top)^{-1}.$$

(b) *Simulation example* A Matlab function implementing the method of 2a is:

ident.m

```
% Identification of a scalar autonomous system

function p = ident(y,L)

H = hankel(y(1:L+1),y(L+1:end));
```

```
Y = -H(1,:);
H = H(2:end,:);

x = Y * H' * inv(H*H');
p = [1 x];

% Convert p to the Matlab convention of decending powers
p = fliplr(p);
p = p / p(1);
```

Applied on the given data it gives

$$p = \begin{bmatrix} 1 & -1 & -1 & -1 \end{bmatrix}.$$

Since in this case the system (5) is solvable and A is full column rank, p is an exact solution and is unique. Therefore the data generating rule is

$$y(t+3) = y(t+2) + y(t+1) + y(t), \qquad y(-2) = 1,\ y(-1) = 0,\ y(0) = 0.$$

The sequence y_{d} is obtained from $\mathscr{B}(p)$ under initial condition $(1,0,0)$.

(c) *Prediction without model* Yes, provided

- the given data y_{p} is exact (this guarantees existence of solution of (5)), and
- the matrix A is full row rank (this is an implicit condition on y_{p}).

Under these assumptions we can identify the data generating system from the data and apply the solutions of Problem 1. In the exact of exact data the method "works well", however, the noisy case is more challenging. (I will propose it as a 3rd year project.)

□