Chapter 1

Computer labs

1.1 Computer lab 1: Least Squares Estimation: do's and don'ts

The computerlabs for the course on system identification are mandatory, that is, we need to have for each of the students registered for the course a small report with the answers to the questions asked throughout the exercise sessions. Attendance of the computerlabs itself is not insisted on. The deadline for the computerlab report is at the end of the last lecture of Part I. It is understood that each student needs to hand in one signed copy, but cooperation is encouraged. A good report consists of a one page summary containing the findings for each computerlab, perhaps containing a motivating graph. No code is to be sent in.

1.1.1 Least Squares

At first, let us study how a least squares estimator behaves numerically.

Basis Functions

The first example considers a linear model of a stationary system. That is, set $n = 100$, and assume we have n samples $\{y_t\}_{t=1}^n$. We try to 'explain' these by relating them to possible signals such that one has for all $t = 1, 2, \ldots, n$ that

$$y_t \approx w_1 \sin(\omega_1 t) + w_1 \sin(\omega_2 t), \tag{1.1}$$

where he frequencies $\omega_1 = 0.1$ and $\omega_2 = 0.1 + \lambda$. Assume the observations are generated for $t = 1, 2, \ldots, n$ as

$$y_t = \sin(0.1t) + e_t, \tag{1.2}$$

where $\{e_t\}_t$ is white noise with zero mean and unit variance. This is generated using the command `randn` in MATLAB. At first, fix $\lambda = 0.1$, and let us try to find the parameters b_1 and b_2 in model eq. (1.1). This can be done using a least squares optimization problem in $\theta = (w_1, w_2)$ as

$$\hat{\theta} = \underset{\theta=(w_1,w_2)}{\operatorname{argmin}} \frac{1}{2} \sum_{t=1}^{n} \left(y_t - w_1 \sin(\omega_1 t) - w_2 \sin(\omega_2 t)\right)^2. \tag{1.3}$$

This problem is solved by finding a solution $\hat{\theta}$ of $\theta = (w_1, w_2)^T$ satisfying the corresponding normal equations.

$$\begin{bmatrix} \sum_{t=1}^{t} \sin(\omega_1 t)\sin(\omega_1 t) & \sum_{t=1}^{t} \sin(\omega_1 t)\sin(\omega_2 t) \\ \sum_{t=1}^{t} \sin(\omega_2 t)\sin(\omega_1 t) & \sum_{t=1}^{t} \sin(\omega_2 t)\sin(\omega_2 t) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} \sum_{t=1}^{t} \sin(\omega_1 t) y_t \\ \sum_{t=1}^{t} \sin(\omega_2 t) y_t \end{bmatrix}. \tag{1.4}$$

or in matrix notation,

$$\mathbf{R}\theta = \mathbf{r}, \tag{1.5}$$

In MATLAB generate the input signals as

```
>> n=100; U12 = [sin(0.1*(1:n)') sin((0.1+lambda)*(1:n)')]
```

and

```
>> y = sin(0.1*(1:n)') + randn(n,1)
```

The elements can be computed in MATLAB as `>> R=U12'*U12 >> r=U12'*y` where `U12` is an n by 2 matrix containing the elements of the two input vectors, and `y` is a vector containing all the observed values. The solution to the normal equations $\mathbf{R}\theta = \mathbf{r}$ or $\theta = \mathbf{R}^{-1}\mathbf{r}$ is found by MATLAB as

```
>> thetahat = inv(R)*r
```

other ways to implement this are

1. using the 'backslash' operator.

   ```
   >> thetahat = R\r
   ```

2. The Penrose pseudo-inverse

   ```
   >>  thetahat = pinv(R)*r
   ```

3. The QR decomposition.

   ```
   >>  [U,S]=qr(R); thetahat = S\(U*r)
   ```

4. The 'backslash' operator

   ```
   >>  thetahat = U12\y
   ```

Now we can investigate what numerical procedures are good in case we have different values for λ. If λ goes to zero, then the matrix $\mathbf{R}$ becomes ill-conditioned and the very formulation of the least squares estimation problem runs into problems. Let us see how this works.

1. The condition number of a matrix $\mathbf{R}$ is calculated in MATLAB as `cond(R)`. Can you plot the value of this condition-number in case when $\gamma = 0.1, 0.01, 0.001, 0.0001, 0.00001$.

2. Calculate for each of those values the least squares estimation $\hat{\theta}$. Which approach to calculate this breaks down the earliest, and which one does not?

FIR Example

Let us now apply this technique towards a simple dynamical model relating a given input signal $\{u_t\}_t$ to a given output signal $\{y_t\}_t$. Consider the model for all $t = 1, 2, \ldots, n$

$$y_t = b_1 u_t + b_2 u_{t-1} + e_t = (b_1 + b_2 q^{-1}) u_t + e_t, \tag{1.6}$$

where $n = 100$. Here $\omega = 1$ and

$$u_t = \sin(\omega t), \tag{1.7}$$

and $\{e_t\}_t$ is white noise with zero mean and unit variance. This is generated using the command `randn` in MATLAB. Then the parameters $\theta = (b_1, b_2)$ can be estimated as $\hat{\theta} = (\hat{b}_1, \hat{b}_2)$ where $\hat{\theta}$ solves the problem

$$\hat{\theta} = \underset{\theta=(b_1,b_2)}{\operatorname{argmin}} \frac{1}{2} \sum_{t=2}^{n} \left(y_t - b_1 u_t - b_2 u_{t-1}\right)^2. \tag{1.8}$$

This problem is solved by finding the solution θ to the corresponding normal equations.

$$\begin{bmatrix} \sum_{t=2}^{t} u_t u_t & \sum_{t=2}^{t} u_t u_{t-1} \\ \sum_{t=2}^{t} u_{t-1} u_t & \sum_{t=2}^{t} u_t u_t \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} \sum_{t=2}^{t} u_t y_t \\ \sum_{t=2}^{t} y_t u_{t-1} \end{bmatrix}. \tag{1.9}$$

For which values of ω do the estimates run into ill-conditioning problems?

1.1.2 Dynamic Models

Let us do some basic manipulations using the `filter` and the `pzmap` command to get an idea on how to simulate using simple dynamic models.

Let for the sake of this exercise $\{u_t\}_t$ be of length $n = 100$:

```
>> u = randn(n,1);
```

Let us now push this signal through a FIR(2) system given as

$$S_1 : \; y_t = (q^{-1} + 0.5 q^{-1}) u_t \tag{1.10}$$

This can be implemented in MATLAB as

```
b=[0, 1, 0.5]; y=filter(b,1,u);
```

Let us do the same for the ARX(0,2) system

$$S_2 : \; (1 - 1.5 q^{-1} + 0.7 q^{-2}) y_t = u_t \tag{1.11}$$

This is implemented as

```
>> a=[1, -1.5, 0.7]; y=filter(1,a,u);
```

Then the ARX(2,2)

$$S_3 : \; (1 - 1.5 q^{-1} + 0.7 q^{-2}) y_t = (q^{-1} + 0.5 q^{-1}) u_t \tag{1.12}$$

is simulated as

```
>> a=[1, -1.5, 0.7];  b=[0, 1, 0.5]; y=filter(b,a,u);
```

Finally, we simulate the ARMAX(2,2,2) system given as

$$S_4: \ (1 - 1.5q^{-1} + 0.7q^{-2})y_t = (q^{-1} + 0.5q^{-1})u_t + (1 - q^{-1} + 0.2q^{-2})e_t, \tag{1.13}$$

where $\{e_t\}_t$ is zero mean, unit variance white noise generated as `>> e = randn(n,1)`. The simulation of the system is performed by

```
>> a=[1,-1.5, 0.7];  b=[0, 1, 0.5]; c=[1, -1, 0.2];
>> y=filter(b,a,u) + filter(c,a,e);
```

The previous command is used later quite intensively in order to generate signals useful for our experiments. Similar tools can also be used to analyze different properties of this model. Herefor the object `idmodel` in the system identification toolbox is useful. To see how this works encode the system S_4 in MATLAB as follows

```
>> m = idmodel(a,b,c)
```

The properties of this system can be inspected for example using the commands

```
>> bode(m); nyquist(m); pzmap(m); impulse(m);
```

All this functions are integrated in the `lti viewer` called as

```
>> ltiview(m)
```

How can you call different views of the given system? Can you modify a parameter of a in S_4 in order to get an unstable system? What poles and zeros (give an exampl) are needed in order to get a system with large oscillations (slow damping)?

1.1.3 Nonparametric Techniques

Nonparametric schemes are characterized by the property that the results are given as curves, tables or graphs, rather than in the form of a model. In this section we will study three different non-parametric methods, namely: transient, correlation and spectral analysis.

In order to illustrate the method, let us consider the following system

$$(1 - 0.8q^{-1})\ y_t = u_{t-1} + e_t, \tag{1.14}$$

where $\{e_t\}_t$ is white noise with zero mean and variance $\sigma^2 = 1$. In the simulations below, 100 experiments are conducted. This is in order to illustrate if systematic error and uncertainty are characteristic to a method. From one experiment, it might be hard to decide wether the method is bad, or if we just had a 'bad day'.

Transient Analysis

We begin by illustrating a transient analysis. System S_1 with a step input $u_t = 1(t > 0)$ is simulated, and the response is plotted. The task can be done using the MATLAB code which is available as the file `lab13a.m`.

Discuss briefly the benefits and drawbacks with the step response analysis. Is it easy to determine the system dynamics (e.g. time constants, static gain and resonance frequencies) from the step response?

Correlation Analysis

Next we consider a correlation analysis. Let the input $\{u_t\}_t$ be white binary noise taking values $u_t = \pm 1$ and length $n = 100$. Then use correlation analysis to estimate the impulse response of the system for lags $\tau = 0, 1, 2, \ldots, 20$. Compare with the true values. This can be done using the MATLAB file `lab13b.m`. Discuss briefly the results from the correlation analysis.

Now repeat this task, but using a more low-frequency character. Use the input signal

$$u'_t = a \frac{1}{1 - 0.8q^{-1}} u_t \tag{1.15}$$

where u_t is as defined in the previous exercise, and $a = \sqrt{1 - 0.8^2}$ is such that the variance of $\{u'_t\}$ equals that of $\{u_t\}_t$. This example is run in MATLAB using the code `lab13c.m`. It is clear that in this case the estimate $\hat{h}$ is severely biased. Why is this so? (Hint: what is implicitly assumed in the previous exercise).

This problem can be solved by using the Wiener-Hopf technique. This is implemented in the MATLAB command `cra.m`. Compare the result which you get with `lab13d` with the one we have before.

Spectral Analysis

Next we will use data from the previous exercise and apply a spectral analysis. The resulting estimates are compare to the ones we had before using a Bode plot. The spectral analysis is implemented in the command `spa` which uses by default a Hamming window of length M, see `lab13e`. Discuss how the choice of M affects the estimate. Moreover, how is the estimate affected by the fact that we use a low frequency input u'_t? Would there be any difference if we use white noise instead?

Least Squares Again

Compare the results from the correlation analysis and the spectral analysis with the one obtained by application of a least squares estimate based on the ARX model

$$y_t = a y_{t-1} + b u_{t-1} + e_t, \ \forall t = 1, 2, \ldots, n. \tag{1.16}$$

Give then a concise answer to the following questions using MATLAB simulation.

- What is the objective function?
- Can you write down the corresponding normal equations?
- How would you solve this set of equations in MATLAB?
- What is the impulse response corresponding to the parameters $\hat{a}$ and $\hat{b}$ (with a MATLAB command)?
- Is the Bode diagram close to the true one?