Computer Exercises in System Identification
Regularization

This version: 2023-08-28

REGLERTEKNIK

AUTOMATIC CONTROL

LINKÖPING

1 Regularization for System Identification

The prediction error approach is to estimate the parameters of a model by

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} V(\theta), \tag{1}$$

where the criterion function $V(\theta)$ is often a weighted norm of the prediction errors

$$\varepsilon(k, \theta) = y(k) - \hat{y}(k|\theta). \tag{2}$$

This is a very general approach which can be applied to both linear and nonlinear model structures.

When the dimension of θ is large, numerical problems might arise and the variance of the parameters estimates can increase. One way to cope with these problems is to include a regularization term in the estimation criterion

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\ V(\theta) + \lambda(\theta - \theta^*)^T R(\theta - \theta^*). \tag{3}$$

A Bayesian interpretation is that θ has a prior distribution that is Gaussian with mean θ^* and covariance matrix proportional to $\frac{1}{\lambda}R^{-1}$. Therefore, a way of thinking about the regularization term conceptually is that θ^* represents an initial guess for the unknown parameter vector and that λR decides the confidence in this guess. An often used special case is when $\theta^* = 0$, which is referred to as Tikhonov regularization. The special case when $\theta^* = 0$ and $R = I$ is called Ridge regression.

Note that the covariance matrix of θ is denoted by Σ in the lecture notes and that R is inversely proportional to Σ. A different notation is used here in order to be coherent with the System Identification Toolbox.

1.1 Bias/variance trade-off in FIR-modelling

First, we will consider the problem to estimate the impulse response of a linear system as a finite impulse response (FIR) model. FIR models are estimated with the command

```
m = arx(z,[0, nb, 0]);
```

in the System Identification Toolbox. The choice of n_b is important because a large n_b is required in order to capture slowly decaying impulse responses without losing accuracy but a large n_b also gives many model parameters which increases the estimation uncertainty. This means that the choice of n_b is essentially a trade-off between bias and variance.

Exercises for Section 1.1

Let the true system be a second-order Butterworth filter

$$G(q) = \frac{0.02008 + 0.04017q^{-1} + 0.02008q^{-2}}{1 - 1.561q^{-1} + 0.6414q^{-2}}. \tag{4}$$

In a simulation experiment, 1000 data points have been collected based on this system with a sample time of 1 second. In order to load this data into your Matlab workspace, run

```
load regularizationExampleData.mat eData
z = eData;
```

1. By looking at the system (4), can you determine a suitable model order for your FIR model? Is the true system included in the model class? Plot the impulse response of the true system. To plot the impulse response, the function calls

   ```
   [y,t] = impulse(G);
   plot(t, y);
   ```

   can be used. Here G can be either a system or model object.

2. Estimate an FIR model of high order, for example $n_b = 50$, and compare the estimated model's impulse response with the impulse response of the true system.

3. Try some lower values of n_b and see if you can get an impulse response with less variance.

4. Now try to get a good bias/variance trade-off using ridge regression for an FIR model of order 50. This can be done with the series of commands

```
aopt = arxOptions;
aopt.Regularization.Lambda =1 ;
m50r = arx(z, [0, 50, 0], aopt);
```

Again, compare the estimated model's impulse response with the impulse response of the true system.

5. As a final thing, try to estimate good values for the regularization parameters, R and λ, prior to the model estimation using the commands

```
ropt = arxRegulOptions('RegularizationKernel','DC');
[Lambda,R] = arxRegul(z,[0 50 0], ropt);
```

   Here 'DC' determines which regularization kernel is used. The estimated values for R and λ can be used in the model estimation with the commands

```
aopt = arxOptions;
aopt.Regularization.Lambda = Lambda;
aopt.Regularization.R = R;
m50tc = arx(z,[0 50 0],aopt);
```

1.2 Regularized ARX-models for Estimating State-space Models

To estimate the impulse response from data a state-space model on innovation form can be used. Assuming that the data is saved as z and has unit sampling time, this can be done with the command

```
mn = ssest(z,n,'Ts',1)
```

The catch is to determine a suitable model order, n. There are two commonly used methods for this

1. **Cross validation (CV)**: Divide the dataset into two parts, one for estimation and one for validation. Estimate models for $n = 1, \ldots, M$ using the estimation dataset and evaluate the fit to the validation dataset using the compare command

```
copt = compareOptions('InitialCondition', 'z');
[~,fitn] = compare(zv, mn, copt);
```

Determine the order n that maximizes the fit. Then reestimate the model using the whole data record.

2. **Use the Akaike criterion (AIC)**: Estimate models for orders $n = 1, \dots, M$ using the whole dataset and then pick that model that minimizes

   ```
   aic(mn);
   ```

An alternative way of doing this is to first estimate an impulse response of high order and then use model reduction techniques. In the System Identification Toolbox, this can be done with the command

```
mr = ssregest(z, n)
```

which first estimates an FIR model of automatically chosen high order and then performs balanced model reduction to the chosen order, n, as explained in the lecture notes.

Exercises for Section 1.2

Now, consider a different system. This time a 30:th order linear system with colored measured measurement noise

$$y(k+1) = G(q)y(k) + H(q)e(k). \tag{5}$$

In a simulation experiment, 210 data points have been collected based on this system with a sample time of 1 second. In order to load the true system and the data into your /matlab workspace, run

```
load regularizationExampleData.mat m0
load regularizationExampleData.mat m0simdata
z = m0simdata;
```

1. Plot the impulse responses of G and H

```
impulse(m0);
impulse(noise2meas(m0));
```

2. Find the CV choice of the model order ($M = 30$ is sufficient).
3. Find the AIC choice of the model order ($M = 30$ is sufficient).
4. Compare the impulse responses of the estimated models with the impulse responses of the true system.
5. Estimate a high-order ARX model (for example $n_a = 5$, $n_b = 60$ and $n_k = 0$) using regularization and let the function **arxRegul** determine the regularization parameters. Compare the impulse responses of this model with the impulse responses of the models estimated before.
6. Now, try to perform balanced model reduction on this ARX model. This can be done with the command

   ```
   mredn = balred(idss(marx),n);
   ```

   How small n can you choose and still obtain accurate impulse responses?

1.3 Bias/Variance Trade-off in Grey-Box Models

When performing greybox estimation it is common that prior physical knowledge is available about the parameters. In order to obtain a balanced trade-off between this prior information and the information in the collected data, regularization is a prime tool.

Exercises for Section 1.3

Consider a DC motor with static gain K, time constant τ and transfer function to shaft angle

$$G(s) = \frac{K}{s(1 + s\tau)}. \tag{6}$$

Assume that we from prior knowledge believe that K is about 4 and τ is about 1. In a simulation experiment, data was collected using this system.

Both the angle and the angular velocity were measured. The true parameters in this simulation were $K = 2.2$ and $\tau = 0.85$. In order to load the data into your MATLAB workspace, run

```
load regularizationExampleData.mat motorData motorData_NoiseFree
z = motorData;
z0 = motorData_NoiseFree;
```

Here z_0 is a dataset collected without any disturbances.

1. Let the angle be denoted by x_1 and the angular velocity by x_2 and cast (6) on state-space form.
2. Write an **idgrey** model file for your found state-space model as

   ```
   function [A,B,C,D] = dcmotor(tau, K, Ts)
   A = ;
   B = ;
   C = ;
   D = ;
   end
   ```

   An idgrey object can then be obtained with the command

   ```
   mi = idgrey('dcmotor', {'time constant',1; 'gain',4}, 'c');
   ```

   where 'dcmotor' is the name of the **idgrey** model file and the guessed parameter values have been inserted as initial values. This model is adapted to the observed data by

   ```
   m = greyest(z, mi);
   ```

   To see your estimated parameters, run

   ```
   getpar(m)
   ```

   What values for K and τ do you obtain?

3. Compare the response of you estimated model with the output of the true system using the disturbed dataset, z. Also, include the response of the initial model m_i, with guessed parameter values, in the comparison.

4. Perform the same comparison using the undisturbed dataset, z_0, for validation.

5. To merge the data information with the prior information we can use regularization

```
gopt = greyestOptions;
gopt.Regularization.Lambda = 100;
gopt.Regularization.R = [1000, 1];
gopt.Regularization.Nominal = 'model';
mr = greyest(motorData, mi, gopt)
```

   where R is chosen to reflect that τ is better known than K. The option **gopt.Regularization.Nominal = 'model'** means that the regularization center, θ^* in (3), is the initial model in the estimation call, m_i. Compare the response of the model m_r with the previous models, both using disturbed and undisturbed validation data.

Solutions with Discussion

The solutions below are not necessarily the only correct ones. Depending on choices during data processing and estimation, your results may be perfectly correct even if they do not exactly agree with the discussion below.

Solutions for Section 1.1

1. The true system is not of FIR-type. The impulse response has decayed to zero after less than 50 samples which means that $n_b = 50$ should be sufficient.

2. The impulse response of the FIR-model with $n_b = 50$ has high variance and does not fit the true system well.

3. For example $n_b = 13$ gives a fairly good result.

4. The variance is reduced as compared to the unregularized estimates.
5. This reduces the variance even more. It should be mentioned that the price with regularization is a bias in the response estimate. This problem is seemingly insignificant in this example.

Solutions for Section 1.2

1. -
2. Using the 150 first datapoints for estimation, the CV choice is $n = 4$.
3. The AIC choice is $n = 12$.
4. The CV choice gives a model fit of 82.38% for the impulse response in G and 73.70% for the impulse response in H. The AIC choice gives a model fit of 83.01% for the impulse response in G and 59.20% for the impulse response in H. This is if all the data is used for estimation. To find the fit for the noise model you can run

   ```
   y0 = impulse(noise2meas(m0), t);
   yh = impulse(noise2meas(m), t);
   100*goodnessOfFit(yh, y0, 'NRMSE')
   ```

   if you run MATLAB from 2019 or earlier and

   ```
   y0 = impulse(noise2meas(m0), t);
   yh = impulse(noise2meas(m), t);
   100*(1-goodnessOfFit(yh, y0, 'NRMSE'))
   ```

   if you run MATLAB from 2020 or later.
5. The high-order ARX model gives a model fit of 81.95% for the impulse response in G and 75.97% for the impulse response in H.
6. Reducing the order of the model to $n = 7$ gives a fit of 82.20% for the impulse response in G and 76.36% for the impulse response in H, which is about as good as the initial high-order ARX model.

Solutions for Section 1.3

1. A state-space representation is

$$\dot{x}_1 = x_2, \tag{7a}$$

$$\dot{x}_2 = -\frac{1}{\tau}x_2 + \frac{K}{\tau}u. \tag{7b}$$

2. The ODE-file can be written as

```
function [A,B,C,D] = dcmotor(tau, K, Ts)
A = [0 1;0 -1/tau];
B = [0; K/tau];
C = eye(2);
D = [0;0];
end
```

   We get the estimates $\hat{\tau} = 0.57448$ and $\hat{K} = 2.1379$.

3. Refining the model with data gives an improved fit in angle from 9.658% to 29.5% and and an improved fit in angular velocity from 0.9975% to 4.162%. Here the comparison is done with disturbed data.

4. Refining the model with data gives an improved fit in angle from 20.68% to 94.31% and and an improved fit in angular velocity from 25.6% to 85.29. Here the comparison is done with undisturbed data.

5. A comparison using disturbed data gives a fit for the regularized model of 29.4% in angle and 3.854% in angular velocity. Notably, this is worse than the model obtained without regularization.

6. A comparison using undisturbed data gives a fit for the regularized model of 97.49% in angle and 94.44% in angular velocity. The fact that regularization improves the adaption to undisturbed data but not to disturbed data shows that regularization is useful for avoiding over-fitting.

References

[1] T. Glad and L. Ljung. *Reglerteknik. Grundläggande teori*. Studentlitteratur, 2006.

[2] A. Hansson and L. Ljung . *Some topics in system identification.* 2020.

[3] L. Ljung, T. Glad and A. Hansson. *Modeling and identification of Dynamic Systems.* Studentlitteratur, 2021.