International school
Linear System Theory, Control and Matrix Computations

Exercises for Lectures 3, 5, and 9

Ivan Markovsky

Problem 1 *Distance measure modulo translation, rotation, and scaling*

Consider two contours $\mathscr{C}_1$ and $\mathscr{C}_2$ in $\mathbb{R}^2$, specified by N matching points

$$p^{(i)} \in \mathscr{C}_1 \subset \mathbb{R}^2 \quad \leftrightarrow \quad q^{(i)} \in \mathscr{C}_2 \subset \mathbb{R}^2, \qquad i = 1, \dots, N.$$

An example with $N = 6$ points is shown in Figure 1.

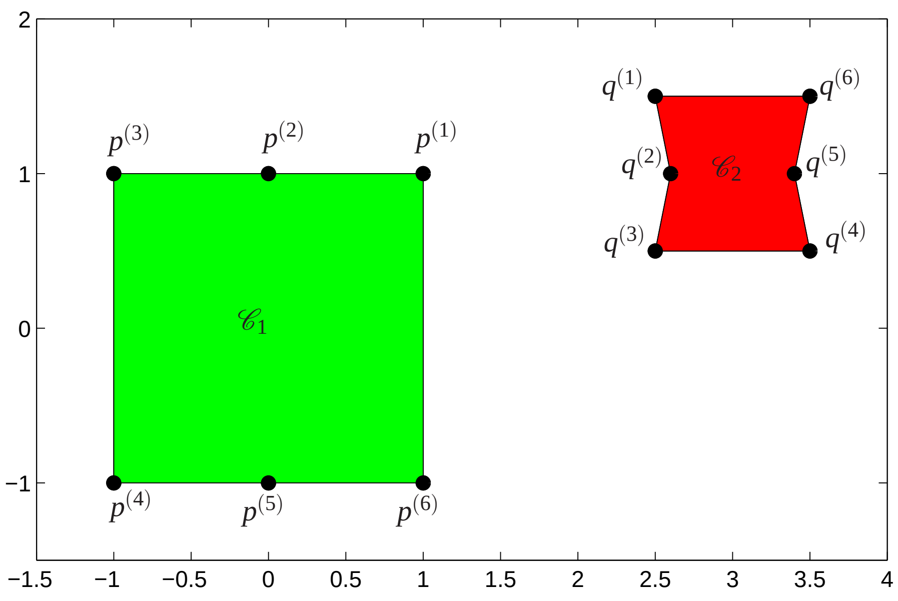

Figure 1: Contours $\mathscr{C}_1$ and $\mathscr{C}_2$ with 6 matching points $p^{(i)} \leftrightarrow q^{(i)}$.

Next, we define a distance measure $\operatorname{dist}(\mathscr{C}_1, \mathscr{C}_2) \in [0, \infty)$ between $\mathscr{C}_1$ and $\mathscr{C}_2$ modulo translation, rotation, and scaling of the contours. Denote by $\mathscr{A}_{a,\theta,s}$ the operator that translates by the vector $a \in \mathbb{R}^2$, rotates by the angle $\theta \in [-\pi, \pi)$ rad, and scales by the factor $s > 0$, i.e.,

$$\mathscr{A}_{a,\theta,s}(p) = s \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} p + a.$$

A possible formulation of $\operatorname{dist}(\mathscr{C}_1, \mathscr{C}_2)$ that gives a tractable problem is

$$\operatorname{dist}(\mathscr{C}_1, \mathscr{C}_2) := \min_{\substack{a \in \mathbb{R}^2,\ s > 0 \\ \theta \in [-\pi, \pi)}} \sqrt{\sum_{i=1}^{N} \left\| p^{(i)} - \mathscr{A}_{a,\theta,s}(q^{(i)}) \right\|_2^2}. \tag{1}$$

1. Prove that problem (1) is equivalent to the following least squares problem

$$\text{minimize}_{(a_1,a_2,b_1,b_2)\in\mathbb{R}^4} \left\| \begin{bmatrix} p_1^{(1)} \\ p_2^{(1)} \\ \vdots \\ p_1^{(N)} \\ p_2^{(N)} \end{bmatrix} - \begin{bmatrix} 1 & 0 & q_1^{(1)} & -q_2^{(1)} \\ 0 & 1 & q_2^{(1)} & q_1^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & q_1^{(N)} & -q_2^{(N)} \\ 0 & 1 & q_2^{(N)} & q_1^{(N)} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{bmatrix} \right\|_2, \tag{2}$$

where the relation between the parameters (b_1, b_2) of (1) and the parameters of (θ, s) of (2) is given by

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = s \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \theta \\ s \end{bmatrix} = \begin{bmatrix} \sin^{-1}(b_2/\sqrt{b_1^2+b_2^2}) \\ \sqrt{b_1^2+b_2^2} \end{bmatrix}. \tag{3}$$

2. Compute $d_{12} := \text{dist}(\mathscr{C}_1, \mathscr{C}_2)$ and $d_{21} := \text{dist}(\mathscr{C}_2, \mathscr{C}_1)$ and the corresponding transformation parameters a, θ, and s, where

$$\begin{bmatrix} p^{(1)} & p^{(2)} & p^{(3)} & p^{(4)} & p^{(5)} & p^{(6)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & -1 & 0 & 1 \\ 1 & 1 & 1 & -1 & -1 & -1 \end{bmatrix},$$
$$\begin{bmatrix} q^{(1)} & q^{(2)} & q^{(3)} & q^{(4)} & q^{(5)} & q^{(6)} \end{bmatrix} = \begin{bmatrix} 2.5 & 2.6 & 2.5 & 3.5 & 3.4 & 3.5 \\ 1.5 & 1.0 & 0.5 & 0.5 & 1.0 & 1.5 \end{bmatrix}.$$

Comment on the results.

□

Problem 2 *Algorithms for exact system identification*

Consider an LTI system $\mathscr{B}_0$ and a trajectory $w_\text{d} = (u_\text{d}, y_\text{d}) \in \mathscr{B}_0$ (the subscript 0 stands for "true data generating system"). In lecture 5, we outlined the following algorithms for exact system identification:

- $w_\text{d} \mapsto R(\xi)$, where $\widehat{\mathscr{B}} := \ker\big(R(\xi)\big)$ is the identified model,
- $w_\text{d} \mapsto H$, where H contains the first samples of the impulse response of $\widehat{\mathscr{B}}$,
- $w_\text{d} \mapsto \mathscr{O}_{\ell_\text{max}+1}(A,C) \mapsto (A,B,C,D)$, where (A,B,C,D) is an input/state/output representation of $\widehat{\mathscr{B}}$, and
- $w_\text{d} \mapsto \big(x_\text{d}(1), \dots, x_\text{d}(\mathtt{n}_\text{max}+\mathtt{m}+1)\big) \mapsto (A,B,C,D)$.

The purpose of this exercise is to apply the MATLAB implementations `w2r` and `uy2x2ss` of the algorithms $w_\text{d} \mapsto R(\xi)$ and $w_\text{d} \mapsto \big(x_\text{d}(1), \dots, x_\text{d}(\mathtt{n}_\text{max}+\mathtt{m}+1)\big) \mapsto (A,B,C,D)$ for exact identification on the data `exactid_data.mat`. Assuming that the data generating system $\mathscr{B}_0$ is controllable and has an upper bound on the lag $\ell_\text{max} = 5$, argue that the identified models, say $\widehat{\mathscr{B}}_1$ and $\widehat{\mathscr{B}}_2$, are MPUM for the data w_d. Conclude that $\widehat{\mathscr{B}}_1 = \widehat{\mathscr{B}}_2 = \mathscr{B}_0$. Verify that the models $\widehat{\mathscr{B}}_1$ and $\widehat{\mathscr{B}}_2$ are exact for w_d, i.e., $w_\text{d} \in \widehat{\mathscr{B}}_1$ and $w_\text{d} \in \widehat{\mathscr{B}}_2$, and are equivalent, i.e., $\widehat{\mathscr{B}}_1 = \widehat{\mathscr{B}}_2$.

□

Problem 3 *Algorithms for approximate system identification*

The purpose of this exercise is to apply a simple version of the algorithm for approximate identification of lecture 9 on the flutter data `flutter.dat` from the Database for the Identification of Systems (DAISY)

`http://homes.esat.kuleuven.be/~smc/daisy/`

(The data is locally available together with the other functions necessary for this exercise.) Consider the misfit

$$\text{misfit}(w_\text{d}, \mathscr{B}) := \min_{\widehat{w}} \|w_\text{d} - \widehat{w}\|_2 \quad \text{subject to} \quad \widehat{w} \in \mathscr{B} \tag{4}$$

between the data w_d and a model $\mathscr{B}$. Geometrically, $\mathrm{misfit}(w_\mathrm{d},\mathscr{B})$ is the orthogonal projection of w_d on $\mathscr{B}$. Assuming that $\mathscr{B}$ is controllable, let $\mathscr{B} = \mathrm{image}\big(M(\sigma)\big)$ be a minimal image representation. In terms of the parameter M, the constraint $\widehat{w} \in \mathscr{B}$ becomes $\widehat{w} = M(\sigma)l$, for some latent variable l. In matrix form,

$$\widehat{w} = \mathscr{T}_T(M)l, \qquad \text{where} \quad \mathscr{T}_T(M) := \begin{bmatrix} M_0 & & & \\ M_1 & M_0 & & \\ \vdots & M_1 & \ddots & \\ M_\ell & \vdots & \ddots & M_0 \\ & M_\ell & & M_1 \\ & & \ddots & \vdots \\ & & & M_\ell \end{bmatrix} \in \mathbb{R}^{\mathtt{w}T\times(T+\ell)}.$$

Then the misfit computation problem (4) is equivalent to the standard least squares problem

$$\text{minimize} \quad \text{over } l \quad \|w_\mathrm{d} - \mathscr{T}_T(M)l\| \tag{5}$$

and is implemented in MATLAB in the functions `misfit` and `misfit2`. The latter does not exploit the structure of the matrix $\mathscr{T}_T(M)$, while the former exploits the banded (but not Toeplitz) structure of $\mathscr{T}_T(M)$.

1. Consider the model $\mathscr{B} = \ker\big(R(\sigma)\big)$, defined by

   $$R(\xi) = R_0 + \xi R_1 + \xi^2 R_2 + \xi^3 R_3,$$

   where

   $$\begin{aligned} R_0 &= \begin{bmatrix} -1.0666 & -0.7235 \end{bmatrix}, & R_1 &= \begin{bmatrix} 2.8745 & 2.3369 \end{bmatrix}, \\ R_2 &= \begin{bmatrix} -2.7278 & -2.5736 \end{bmatrix}, & R_3 &= \begin{bmatrix} 0.8924 & 1.0000 \end{bmatrix}. \end{aligned} \tag{6}$$

   (This model is computed by the `w2r` function.) Compute the misfit between the flutter data and $\mathscr{B}$ with `misfit` and `misfit2`, measuring the computation time (see `help toc` in MATLAB), and compare the obtained answers. Comment on the results.

2. *Misfit minimization* Next we consider the misfit minimization problem

   $$\widehat{\mathscr{B}}_\mathrm{gtls} := \arg\min_{\widehat{\mathscr{B}}} \quad \mathrm{misfit}(w_\mathrm{d},\mathscr{B}) \quad \text{subject to} \quad \widehat{\mathscr{B}} \in \mathscr{L}^{\mathtt{w}}_{\mathtt{m},\ell}, \tag{7}$$

   where the number of inputs $\mathtt{m} < \mathtt{w}$ and the lag ℓ are given natural numbers. Considering, again the image representation and restricting for simplicity to the SISO case, i.e., $\mathtt{w} = 2$ and $\mathtt{m} = 1$, (7) is equivalent to

   $$\text{minimize} \quad \text{over } M \in \mathbb{R}^{2(\ell+1)\times 1} \quad \mathrm{misfit}\big(w_\mathrm{d}, \mathrm{image}\left(M(\sigma)\right)\big) \quad \text{subject to} \quad M \neq 0, \tag{8}$$

   which is a constrained nonlinear least-squares problem. Partition the parameter $M(\xi)$ of the image representation as $M =: \left[\begin{smallmatrix} p \\ q \end{smallmatrix}\right]$ and assumed that the polynomial p is monic (highest order coefficient is one). With this assumption, problem (8) becomes unconstrained

   $$\text{minimize} \quad \text{over } M' \in \mathbb{R}^{2\ell+1\times 1} \quad \mathrm{misfit}\big(w_\mathrm{d}, \mathrm{image}\left(\left[\begin{smallmatrix} M' \\ 1 \end{smallmatrix}\right](\sigma)\right)\big). \tag{9}$$

   The function `gtls` solves (9), using the function `fminsearch` from the Optimization Toolbox of MATLAB. (`fminsearch` implements the Nelder-Mead local optimization method.)

   Partition the flutter data set into identification, e.g., first 60%, and validation, e.g., remaining 40%, parts. Compute a locally optimal model with lag $\ell = 3$ for the identification part of the data, using as an initial approximation (6). Validate the identified model by computing the misfit on the validation part of the data. Show the best fit of the validation data by the identified model.

3. *Compare with PEM* A classical method for system identification is the prediction error method (PEM) and a popular implementation of the PEM method is the function `pem` from the System Identification Toolbox of MATLAB. Similarly to the misfit minimization method (7), the PEM method is based on local optimization, starting from an initial approximation. However, the PEM method minimizes a different cost function. Using its default settings, the `pem` is applied on the identification part of the flutter data by

   ```
   >> load flutter.dat
   >> sys_pem = pem(iddata(flutter(1:600,2),flutter(1:600,1)),3);
   ```

   (see `help pem` and `help iddata` for details on the usage of these functions). The following script validates the PEM model using the `misfit` function

   ```
   >> sys_pem = ss(sys_pem); % convert sys_pem from the idss to the ss format
   >> % apply misfit on the deterministic part of the model
   >> [mv_pem,wh_pem] = misfit(flutter(601:end),sys_pem(1,1));
   ```

   A validation function from the System Identification Toolbox is `compare`. The following script validates the GTLS model using `compare`

   ```
   >> % convert sys_gtls from the ss to the idss format and apply compare
   >> compare(iddata(flutter(601:end,2),flutter(601:end,1)),idss(sys))
   ```

   Using the functions `misfit` and `compare`, validate the `pem` and `gtls` identified models on the validation part of the data. Repeat the experiment for different partitionings of the data into identification and validation parts. Comment on the results.

□