Stochastic Adaptive Control (02421)

Lecture 11

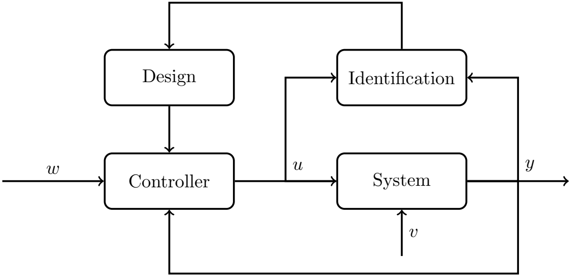
Tobias K. S. Ritschel

Assistant Professor

Section for Dynamical Systems, DTU Compute

Lecture Plan

1. ARX models
2. ARX prediction + control
3. ARX estimation
4. ARX model validation + adaptive control
5. ARMAX control
6. ARMAX estimation + adaptive control
7. Systems and control theory
8. Stochastic systems + Kalman filtering
9. SS estimation (recursive) + control
10. SS control
11. SS estimation (batch)
12. SS estimation (recursive)
13. SS nonlinear control



Today's Agenda

- Least-squares estimation
- Maximum likelihood estimation
- Cramér-Rao lower bound

Least-squares estimation

System

System

$$x_{t+1} = A(\theta)x_t + B(\theta)u_t + G(\theta)v_t, \tag{1}$$

$$y_t = C(\theta)x_t + D(\theta)u_t + F(\theta)e_t, \tag{2}$$

Stochastic vectors

$$x_0 \sim N(m_0(\theta), P_0(\theta)), \qquad v_t \sim N(0, I), \qquad e_t \sim N(0, I) \tag{3}$$

Note: The process and measurement noise are *standard* normal such that we can linearize wrt. θ in the recursive formulation

Least-squares parameter estimation problem

State prediction

$$\hat{x}_{t+1} = A\hat{x}_t + Bu_t, \qquad \hat{x}_0 = m_0 \tag{4}$$

Output prediction

$$\hat{y}_t = C\hat{x}_t + Du_t \tag{5}$$

Residuals

$$\epsilon_t = y_t - \hat{y}_t \tag{6}$$

Least-squares parameter estimation problem

$$\hat{\theta} = \arg\min_{\theta} J_N(\theta; Y_N), \qquad J_N(\theta; Y_N) = \frac{1}{2}\sum_{t=0}^{N} \epsilon_t^T \epsilon_t \tag{7}$$

This problem is nonlinear in the parameters, θ, and the solution must be approximated numerically, e.g., using Matlab's `fmincon`.

How would you estimate the parameters that only affect F and G?

Think about it for yourself for one minute and then discuss with the person next to you for one minute

Matlab's `fmincon`

Syntax

```
x = fmincon(fun,x0,A,b)
x = fmincon(fun,x0,A,b,Aeq,beq)
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub)
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options)
x = fmincon(problem)
[x,fval] = fmincon(___)
[x,fval,exitflag,output] = fmincon(___)
[x,fval,exitflag,output,lambda,grad,hessian] = fmincon(___)
```

Description

Nonlinear programming solver.
Finds the minimum of a problem specified by

$$\min_x f(x) \text{ such that } \begin{cases} c(x) \leq 0 \\ ceq(x) = 0 \\ A \cdot x \leq b \\ Aeq \cdot x = beq \\ lb \leq x \leq ub, \end{cases} \tag{8}$$

b and beq are vectors, A and Aeq are matrices, c(x) and ceq(x) are functions that return vectors, and f(x) is a function that returns a scalar. f(x), c(x), and ceq(x) can be nonlinear functions. x, lb, and ub can be passed as vectors or matrices; see Matrix Arguments.

Link: `https://de.mathworks.com/help/optim/ug/fmincon.html`

Matlab's `fmincon`: Least-squares objective function

```matlab
function JN = least_squares_objective_function(theta, Y, U, m0, p)
    % Create system matrices (provided by the user)
    [A, B, C, D] = p.create_system_matrices(theta, p);

    % Initial state
    xhatt = m0;

    % Objective function
    JN = 0;

    for t = 1:N+1 % (the real t is actually 0:N)
        % Manipulated input and output
        ut = U(:, t);
        yt = Y(:, t);

        % Predict output and residual
        yhatt    = C*xhatt + D*ut;
        epsilont = yt - yhatt;

        % Predict state
        xhattp1 = A*xhatt + B*ut;

        % Add to objective function
        JN = JN + 0.5*(epsilont'*epsilont);

        % Update states
        xhatt = xhattp1;
    end
end
```

Maximum likelihood estimation

Likelihood

Multiplication rule

$$P(A,B) = P(A|B)P(B), \qquad P(A,B|C) = P(A|B,C)P(B|C) \tag{9}$$

Probability density function

$$p(y_N, y_{N-1}, \ldots, y_0|\theta) = p(y_N|y_{N-1}, \ldots, y_0, \theta)p(y_{N-1}, \ldots, y_0|\theta) \tag{10}$$

Likelihood

$$\mathcal{L}(\theta) = p(y_N, y_{N-1}, \ldots, y_0|\theta) = p(y_0|\theta)\prod_{t=1}^{N} p(y_t|y_{t-1}, \ldots, y_0, \theta) \tag{11}$$

Log-likelihood

$$\ln \mathcal{L}(\theta) = \ln p(y_0|\theta) + \sum_{t=1}^{N} \ln p(y_t|y_{t-1}, \ldots, y_0, \theta) \tag{12}$$

Likelihood for normally distributed variables

Probability density of normal distribution

$$p(y_t|y_{t-1},\dots,y_0,\theta) = \frac{1}{\sqrt{(2\pi)^{n_y}\det P_{y,t|t-1}}}\exp\left(-\frac{1}{2}\epsilon_t^T P_{y,t|t-1}^{-1}\epsilon_t\right) \quad (13)$$

Residuals

$$\epsilon_t = y_t - \hat{y}_{t|t-1} \quad (14)$$

Logarithm of probability density of normal distribution

$$\ln p(y_t|y_{t-1},\dots,y_0,\theta) = -\frac{n_y}{2}\ln 2\pi - \frac{1}{2}\ln\det P_{y,t|t-1} - \frac{1}{2}\epsilon_t^T P_{y,t|t-1}^{-1}\epsilon_t \quad (15)$$

Log-likelihood

$$\mathcal{L}(\theta) = -\frac{(N+1)n_y}{2}\ln 2\pi - \frac{1}{2}\sum_{t=0}^{N}\left(\ln\det P_{y,t|t-1} + \epsilon_t^T P_{y,t|t-1}^{-1}\epsilon_t\right) \quad (16)$$

Maximum likelihood estimation

Maximum likelihood estimation

$$\hat{\theta} = \arg\max_{\theta} \quad \mathcal{L}(\theta) \tag{17}$$

Equivalent formulation

$$\hat{\theta} = \arg\min_{\theta} \quad J_N(\theta), \qquad\qquad J_N(\theta) = -\mathcal{L}(\theta) \tag{18}$$

Negative log-likelihood function

$$\boxed{J_N(\theta) = \frac{(N+1)n_y}{2} \ln 2\pi + \frac{1}{2} \sum_{t=0}^{N} \left(\ln \det P_{y,t|t-1} + \epsilon_t^T P_{y,t|t-1}^{-1} \epsilon_t \right)} \tag{19}$$

Key differences to least-squares objective function

1. Determinant of **covariance** is penalized
2. Residuals are weighted by the inverse of the **covariance**

Kalman filter equations

Measurement update (vectors)

$$\boxed{\begin{aligned} \hat{y}_{t|t-1} &= C\hat{x}_{t|t-1} + Du_t, \\ \epsilon_t &= y_t - \hat{y}_{t|t-1}, \\ \hat{x}_{t|t} &= \hat{x}_{t|t-1} + \kappa_t \epsilon_t \end{aligned}}$$

(20) (21) (22)

Measurement update (matrices)

$$\boxed{\begin{aligned} P_{y,t|t-1} &= CP_{t|t-1}C^T + R_2, \\ P_{xy,t|t-1} &= P_{t|t-1}C^T, \\ \kappa_t &= P_{xy,t|t-1}P_{y,t|t-1}^{-1}, \\ P_{t|t} &= P_{t|t-1} - \kappa_t P_{xy,t|t-1}^T \end{aligned}}$$

(23) (24) (25) (26)

Time update

$$\boxed{\begin{aligned} \hat{x}_{t+1|t} &= A\hat{x}_{t|t} + Bu_t, \\ P_{t+1|t} &= AP_{t|t}A^T + R_1 \end{aligned}}$$

(27) (28)

Matlab's `fmincon`: Negative log-likelihood objective function

```matlab
function JN = maximum_likelihood_objective_function(theta, Y, U, m0, P0, p)
    % Create system matrices (provided by the user)
    [A, B, G, C, D, F] = p.create_system_matrices(theta, p);

    % Initial state and covariance
    xhatttm1 = m0;
    Pttm1    = P0;

    % Objective function
    JN = 0.5*numel(Ybar)*log(2*pi);

    for t = 1:N+1 % (the real t is actually 0:N)
        % Manipulated input and output
        ut = U(:, t);
        yt = Y(:, t);

        % Predicted output and covariance, residual, and Kalman gain
        yhatttm1 = C*xhatttm1 + D*ut;
        Pyttm1   = C*Pttm1*C' + R2;
        epsilont = yt - yhatttm1;
        kappat   = Pttm1*C'/Pyttm1;

        % Measurement update
        xhattt = xhatttm1 + kappat*epsilont;
        Ptt    = Pttm1 - kappat*C*Pttm1;

        % Time update
        xhatttp1 = A*xhattt + B*ut;
        Pttp1    = A*Ptt*A' + R1;

        % Add to objective function
        JN = JN + 0.5*(log(det(Pyttm1)) + epsilont'*(Pyttm1\epsilont));

        % Update states
        xhatttm1 = xhattp1; Pttm1 = Pttp1;
    end
end
```

Cramér-Rao lower bound

Cramér-Rao lower bound

Lower bound on individual parameter variances[1]

$$\mathrm{Cov}(\hat{\theta}) \succeq F^{-1} \tag{29}$$

where $A \succeq B$ means that $A - B$ is positive semidefinite.

Fisher's information matrix

$$F_{ij} = \mathbb{E}\left[\frac{\partial J_N}{\partial \theta_i}(\theta; Y_N)\frac{\partial J_N}{\partial \theta_j}(\theta; Y_N)\right] \tag{30}$$

Typically, maximum likelihood estimators are *efficient*, which means that equality holds in the bound

$$\boxed{F_{ij} = \left(\frac{\partial m_Y}{\partial \theta_i}\right)^T P_Y^{-1}\frac{\partial m_Y}{\partial \theta_j} + \frac{1}{2}\,\mathrm{Tr}\left(P_Y^{-1}\frac{\partial P_Y}{\partial \theta_i}P_Y^{-1}\frac{\partial P_Y}{\partial \theta_j}\right)} \tag{31}$$

[1] Theorem 4.4 in the book by A. van den Bos, 2007. Parameter estimation for scientists and engineers. Wiley.

The prediction problem: Compact notation (recap)

Compact notation

$$X_N = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{bmatrix}, \quad Y_N = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{bmatrix}, \quad U_N = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \end{bmatrix}, \quad V_N = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_N \end{bmatrix}, \quad E_N = \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_N \end{bmatrix}, \tag{32}$$

$$\Phi_{xx} = \begin{bmatrix} I \\ A \\ \vdots \\ A^N \end{bmatrix}, \quad \Gamma_{xu} = \begin{bmatrix} 0 & & & \\ B & 0 & & \\ \vdots & \ddots & \ddots & \\ A^{N-1}B & \cdots & B & 0 \end{bmatrix}, \quad \Gamma_{xv} = \begin{bmatrix} 0 & & & \\ G & 0 & & \\ \vdots & \ddots & \ddots & \\ A^{N-1}G & \cdots & G & 0 \end{bmatrix}, \tag{33}$$

$$\Phi_{yx} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^N \end{bmatrix}, \quad \Gamma_{yu} = \begin{bmatrix} D & & & \\ CB & D & & \\ \vdots & \ddots & \ddots & \\ CA^{N-1}B & \cdots & CB & D \end{bmatrix}, \quad \Gamma_{yv} = \begin{bmatrix} 0 & & & \\ CG & 0 & & \\ \vdots & \ddots & \ddots & \\ CA^{N-1}G & \cdots & CG & 0 \end{bmatrix}, \tag{34}$$

$$\Gamma_{ye} = \begin{bmatrix} F & & & \\ & F & & \\ & & \ddots & \\ & & & F \end{bmatrix}, \quad R_V = I, \tag{35}$$

$$X_N = \Phi_{xx} x_0 + \Gamma_{xu} U_N + \Gamma_{xv} V_N, \qquad V_N \sim N(0, R_V), \tag{36}$$

$$Y_N = \Phi_{yx} x_0 + \Gamma_{yu} U_N + \Gamma_{yv} V_N + \Gamma_{ye} E_N, \qquad E_N \sim N(0, R_E) \tag{37}$$

*The superscript N on the matrices and the dependencies on the parameters, θ, have been omitted for brevity of notation.

Distribution of prediction

Distribution of output prediction

$$Y_N \sim N(m_Y, P_Y) \tag{38}$$

Mean

$$m_Y = \mathbb{E}[Y_N] = \Phi_{yx} m_0 + \Gamma_{yu} U_N \tag{39}$$

Covariance

$$P_Y = \mathrm{Cov}(Y_N) = \Phi_{yx} P_0 \Phi_{yx}^T + \Gamma_{yv} R_V \Gamma_{yv}^T + \Gamma_{ye} R_E \Gamma_{ye}^T \tag{40}$$

Negative log-likelihood function (multiplication rule not used)

$$\begin{aligned} J_N(\theta) = -\ln \mathcal{L}(\theta; Y_N) = \frac{(N+1)n_y}{2} \ln(2\pi) + \frac{1}{2} \ln \det P_Y \quad (41) \\ + \frac{1}{2}(Y_N - m_Y)^T P_Y^{-1} (Y_N - m_Y) \quad (42) \end{aligned}$$

Linear algebra hints

Problem: `PY\epsilon` might give imprecise results (and a warning)

Use an LDL factorization ($P_Y = LDL^T$)

$$LZ = \epsilon, \qquad Z = DL^T X, \tag{43}$$

$$DY = Z, \qquad Y = L^T X, \tag{44}$$

$$L^T X = Y \tag{45}$$

Questions?