Identification of Continuous-Time Systems Utilizing Lebesgue-Sampled Data

Identification of Continuous-Time Systems Utilizing Lebesgue-Sampled Data

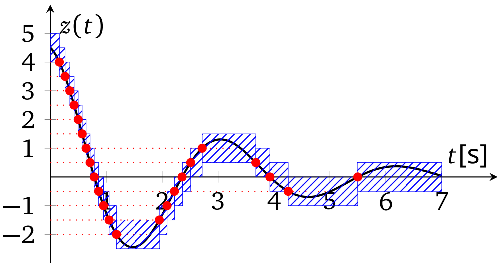
R. A. González, A. L. Cedeño, M. Coronel, J. C. Agüero and C. R. Rojas

ERNSI 2022

Lebesgue sampling

Key idea: sample only when a signal crosses fixed thresholds

- Reduces data-rate requirements
- Analysis and control design are more complicated
- To avoid spurious sampling, send-on-delta strategy can be used



Problem formulation

Continuous-time SISO system in state-space form:

$$\mathrm{d}\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t)\mathrm{d}t + \mathbf{B}u(t)\mathrm{d}t + \mathrm{d}\mathbf{w}(t)$$
$$z(t) = \mathbf{C}\mathbf{x}(t) + Du(t)$$

- $\mathbf{w}(t)$: Wiener process of incremental covariance $\mathbf{Q}$
- $u(t)$: known input signal
- $\mathbf{A}, \mathbf{B}, \mathbf{C}, D$ and $\mathbf{Q}$: unknown constant matrices

Assume $z(t)$ is sampled as above, with time-stamps $\{t_l\}_{l=1}^M$ and threshold distance h

How can we estimate $\mathbf{A}, \mathbf{B}, \mathbf{C}, D$ and $\mathbf{Q}$ using $\{u(t)\}_{t\in[t_1,t_M]}$ and $\{z(t_l)\}_{l=1}^M$ and the intersample behavior knowledge?

EM for Lebesgue-sampled data

Approach: small sampling period Δ + Delta domain. Relabeling $k=1$ to $k=N := \lfloor \frac{t_M}{\Delta} \rfloor - \lfloor \frac{t_1}{\Delta} \rfloor + 1$ and discretizing:

$$\mathrm{d}\mathbf{x}_k^+ = \Delta \mathbf{A}_{\text{in}}\mathbf{x}_k + \Delta \mathbf{B}_{\text{in}} u_k + \mathrm{d}\mathbf{w}_k^+$$

$$z_k = \mathbf{C}\mathbf{x}_k + Du_k + \epsilon e_k \leftarrow \text{added to ensure full rank matrices}$$

$$y_k = \mathcal{Q}_h\{z_k\} \leftarrow \text{set-valued function describing Lebesgue sampling}$$

with $\mathbf{A}_{\text{in}} = (e^{\mathbf{A}\Delta} - \mathbf{I})/\Delta$, $\mathbf{B}_{\text{in}} = \Delta^{-1}\int_0^\Delta e^{\mathbf{A}\tau}\mathbf{B}\mathrm{d}\tau$, and noise vector covariance $\Delta \mathbf{Q}_{\text{in}}\delta_{k-l}^K$, where $\mathbf{Q}_{\text{in}} = \Delta^{-1}\int_0^\Delta e^{\mathbf{A}\tau}\mathbf{Q}e^{\mathbf{A}^\top\tau}\mathrm{d}\tau$.

Note that $(\mathbf{A}_{\text{in}}, \mathbf{B}_{\text{in}}, \mathbf{Q}_{\text{in}}) \approx (\mathbf{A}, \mathbf{B}, \mathbf{Q})$ for small Δ

$\Longrightarrow$ derive EM for fast-sampled Delta domain equivalent!

Parameter estimates $\mathbf{A}, \mathbf{B}, \mathbf{C}, D, \mathbf{Q}$ described in $\boldsymbol{\theta}$

E-step: Q function

$$-Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}_i) = -\mathbb{E}\{\log \mathrm{p}(\mathbf{x}_{1:N+1}, z_{1:N}) | y_{1:N}, \hat{\boldsymbol{\theta}}_i\}$$
$$\sim L_0(\hat{\boldsymbol{\theta}}_i) + N \log\det(\mathbf{Q}_{\text{in}})$$
$$+\frac{1}{\epsilon^2}\left(\begin{bmatrix}\mathbf{C}^\top \\ D\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Gamma}_{xx} & \boldsymbol{\Gamma}_{ux}^\top \\ \boldsymbol{\Gamma}_{ux} & \Gamma_{uu}\end{bmatrix}\begin{bmatrix}\mathbf{C}^\top \\ D\end{bmatrix} - 2\begin{bmatrix}\mathbf{C}^\top \\ D\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Gamma}_{xz} \\ \Gamma_{uz}\end{bmatrix}\right)$$
$$+\operatorname{tr}\left\{\mathbf{Q}_{\text{in}}^{-1}\left(\begin{bmatrix}\mathbf{A}_{\text{in}}^\top \\ \mathbf{B}_{\text{in}}^\top\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Gamma}_{xx} & \boldsymbol{\Gamma}_{ux}^\top \\ \boldsymbol{\Gamma}_{ux} & \Gamma_{uu}\end{bmatrix}\begin{bmatrix}\mathbf{A}_{\text{in}}^\top \\ \mathbf{B}_{\text{in}}^\top\end{bmatrix} - \begin{bmatrix}\mathbf{A}_{\text{in}}^\top \\ \mathbf{B}_{\text{in}}^\top\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Gamma}_{x\delta} \\ \boldsymbol{\Gamma}_{u\delta}\end{bmatrix} - \begin{bmatrix}\boldsymbol{\Gamma}_{x\delta} \\ \boldsymbol{\Gamma}_{u\delta}\end{bmatrix}^\top \begin{bmatrix}\mathbf{A}_{\text{in}}^\top \\ \mathbf{B}_{\text{in}}^\top\end{bmatrix}\right)\right\}$$

where $\boldsymbol{\Gamma}$ matrices are conditional expectations w.r.t. $y_{1:N}, \hat{\boldsymbol{\theta}}_i$.

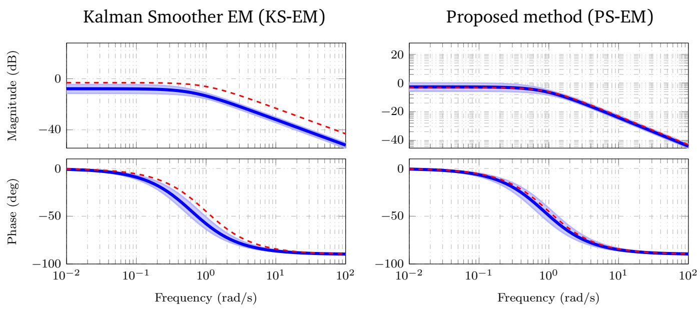
M-step: EM iterations $\hat{\boldsymbol{\theta}}_{i+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}_i)$ are given by

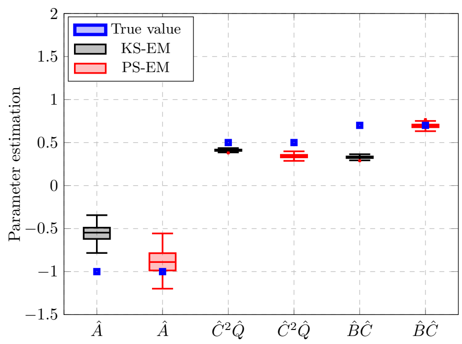
$$\begin{bmatrix}\mathbf{A}_{i+1} & \mathbf{B}_{i+1} \\ \mathbf{C}_{i+1} & D_{i+1}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\Gamma}_{x\delta} & \boldsymbol{\Gamma}_{xz} \\ \boldsymbol{\Gamma}_{u\delta} & \Gamma_{uz}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Gamma}_{xx} & \boldsymbol{\Gamma}_{ux}^\top \\ \boldsymbol{\Gamma}_{ux} & \Gamma_{uu}\end{bmatrix}^{-1}$$

$$\mathbf{Q}_{i+1} = \frac{\Delta}{N}\left(\boldsymbol{\Gamma}_{\delta\delta} - \begin{bmatrix}\boldsymbol{\Gamma}_{x\delta} \\ \boldsymbol{\Gamma}_{u\delta}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{\Gamma}_{xx} & \boldsymbol{\Gamma}_{ux}^\top \\ \boldsymbol{\Gamma}_{ux} & \Gamma_{uu}\end{bmatrix}^{-1}\begin{bmatrix}\boldsymbol{\Gamma}_{x\delta} \\ \boldsymbol{\Gamma}_{u\delta}\end{bmatrix}\right)$$

How to compute $\boldsymbol{\Gamma}$? Particle filtering/smoothing (or Gaussian Sum filtering/smoothing)

Simulation studies





References

[1] R. A. González, A. L. Cedeño, M. Coronel, J. C. Agüero and C. R. Rojas. "Identification of Continuous-time State-Space Systems Utilizing Lebesgue-Sampled Data". In preparation.

Department of Mechanical Engineering, Control Systems Technology