EL2450 Hybrid and Embedded Control

Lecture 5: Implementation aspects

- Modeling and compensation for jitter, delay, loss
- Quantization and packet losses in state feedback

Today's Goal

You should be able to

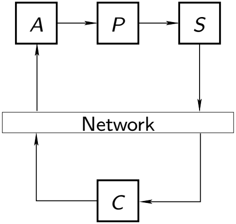
- Derive models for delay, jitter and loss
- Modify controllers to compensate for known and unknown delays
- Model and analyze packet losses and quantization effect in state feedback

Implementation Aspects

Computations and communications introduce imperfections, e.g.,

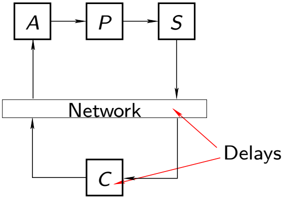
- Delays
- Packet Losses
- Quantization

Where do they appear in the control loop:



Time Delays

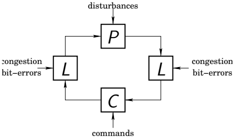
- Delays τ in communications and computations
- Delays are bad for control loops (avoid if possible)
- Delays can be known or unknown (influence control)
- Delay variation is denoted *jitter*
- Data loss (e.g., lost packet) can be interpreted as $\tau = \infty$



Communication Influence on Control Loop

Communication impose uncertainties

- Transmission delays
  - Data are delayed due to buffering and propagation delays
  - Delays are varying due to varying network load
- Data drops
  - Data are lost due to network protocol
  - Bit-errors in wireless links
  - Sudden loss of connection



Control Systems with Unknown Delays

Nyquist Criterion: Control system with phase margin φ_m at ω_c can have maximum (fixed) time delay

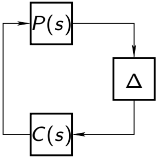
$$\tau < \varphi_m/\omega_c$$

Example $P(s) = 1/s^2$ with $C(s) = K(1 + T_d s)$, $K = 1$, $T_d = 1.4$, gives phase margin $\varphi_m = 1.13 = 65$ deg at $\omega_c = 1.54$. Then,

$$\tau < \varphi_m/\omega_c = 0.73$$

Stability under Unknown Time-Varying Delay

Theorem: Consider linear feedback system with Δ representing a delay $0 \le \tau(t) \le \tau_{\mathsf{max}}$. Closed-loop system stable if
$$\left|\frac{P(i\omega)C(i\omega)}{1+P(i\omega)C(i\omega)}\right| < \frac{1}{\tau_{\mathsf{max}}\omega},\ \forall \omega \in [0,\infty]$$



Proof is based on small gain theorem (see [L, paper 3])

Relations to Nyquist Criterion

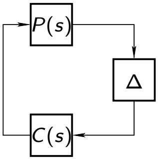
At $\omega = \omega_c$,

$$\left|\frac{P(i\omega)C(i\omega)}{1 + P(i\omega)C(i\omega)}\right| = \frac{1}{|1 - e^{i\varphi_m}|} \approx \frac{1}{\varphi_m}$$

Hence, closed-loop stability if

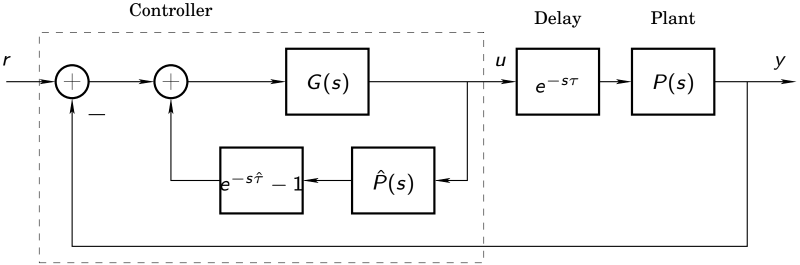
$$\frac{1}{\varphi_m} < \frac{1}{\tau_{\max}\omega_c}$$

Corresponds to Nyquist criterion for constant $\tau(t) = \tau_{\max}$



Control Systems with Known Delays

Known time delays can be compensated with **Smith predictor**:



Closed-loop system with $\hat{P} = P$ and $\hat{\tau} = \tau$: $y = \frac{PG}{1+PG} e^{-s\tau} r$

Design controller as if there were no time delay and then implement structure above

Example

Consider control design for

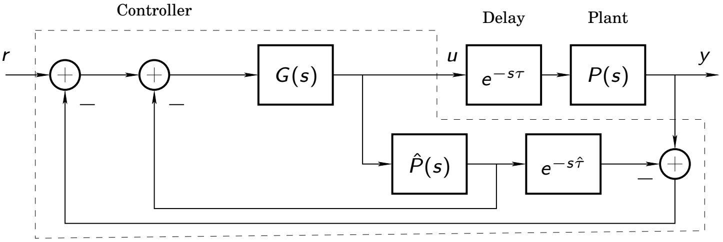
$$P(s)e^{-s\tau} = \frac{e^{-s\tau}}{s^2}$$

$G(s) = K(1 + T_d s)$, $K = 1$, $T_d = 1.4$, gives performance worse than the Smith Predictor.
Controller with Smith predictor (from u to $r - y$) is then given by

$$\begin{aligned} C(s) &= \frac{G(s)}{1 - P(s)G(s)(e^{-s\tau} - 1)} \\ &= \frac{Ks^2(1 + T_d s)}{s^2 + K(1 + T_d s) - K(1 + T_d s)e^{-s\tau}} \end{aligned}$$

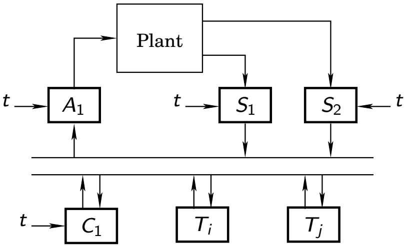
Interpretation of Smith Predictor

Compared to conventional PID control, the Smith predictor estimates old responses y. Alternative block diagram of Smith predictor:



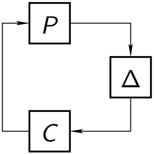
Time Stamps

Delays can be estimated from **time stamped data**, each node transmits data together with sampling time, e.g., $(y(t), t)$
If the receiving node is synchronized, it can determine and compensate time delay



Time Stamped Sensor Measurements

Suppose sensor measurements are delayed unknown and varying time $\tau(t)$
If sensor data $(y(t_s), t_s)$ is received at controller at time $t = t_c$, the current delay is $\tau(t) = t_c - t_s$, which can then be used in the control algorithm (cf., Smith predictor)



Important to take *all* delays into account (buffering, computation, propagation etc.)

Compensating Delays in State Feedback

Consider plant with state feedback

$$\dot{x}(t) = Ax(t) + Bu(t)$$

Sensor node sends $x(kh)$ to control node.
Suppose delay $\tau_k = \tau(kh) < h$ in Δ.
Controller has $x(kh)$ available and derives (draw time axis)

$$\bar{x}(kh + \tau_k) = e^{A\tau_k} x(kh) + \int_{kh}^{kh+\tau_k} e^{A(kh+\tau_k - s)} Bu(s) ds$$

and then

$$u(kh + \tau_k) = -L\bar{x}(kh + \tau_k)$$

Compensating Delays in Output Feedback

Consider plant

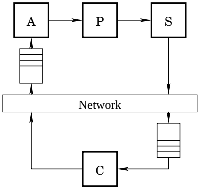
$$\dot{x}(t) = Ax(t) + Bu(t), \qquad y(t) = Cx(t)$$

Sensor node sends $y(kh)$ to control node with transmission delay $\tau_k = \tau(kh) < h$. Perform estimation and control in the following order:

$$\begin{aligned}
\bar{x}(kh) &= \hat{x}(kh) + K[y(kh) - C\hat{x}(kh)] \\
\bar{x}(kh+\tau_k) &= e^{A\tau_k}\bar{x}(kh) + \int_{kh}^{kh+\tau_k} e^{A(kh+\tau_k-s)}Bu(s)ds \\
u(kh+\tau_k) &= -L\bar{x}(kh+\tau_k) \\
\hat{x}(kh+h) &= e^{A(h-\tau_k)}\bar{x}(kh+\tau_k) + \int_{kh+\tau_k}^{kh+h} e^{A(kh+h-s)}Bu(s)ds
\end{aligned}$$

Delays Larger Than h

- Similar scheme can be applied for a large (known) delay $\tau(t) > h$, by extending the state of the estimator (cf., sampling of systems with delay in Lecture 2)
- Buffers can be introduced to handle out-of-order delivery
- It is possible to use late data to adjust old estimates
- But buffers may introduce time delay

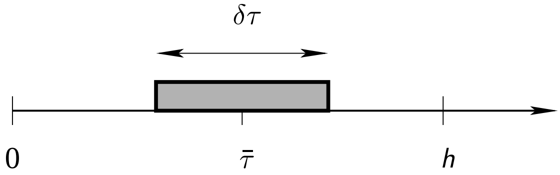


Jitter

Jitter δt is max deviation in time delay

$$\delta\tau = \tau_{\mathsf{max}} - \tau_{\mathsf{min}}$$

Introduce mean $\bar{\tau} = (\tau_{\mathsf{max}} + \tau_{\mathsf{min}})/2$



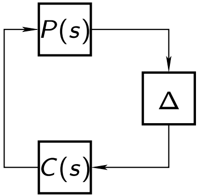
Stability with Jitter

Theorem: Consider linear feedback system with Δ representing a delay

$$0 \leq \bar{\tau} - \delta\tau/2 \leq \tau(t) \leq \bar{\tau} + \delta\tau/2$$

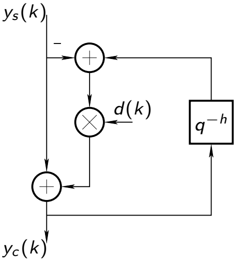
Closed-loop system stable if

$$\left| \frac{P(i\omega)C(i\omega)}{1 + P(i\omega)C(i\omega)e^{-i\omega\bar{\tau}}} \right| < \frac{\sqrt{2}}{\delta\tau \cdot \omega}$$



Data Loss Model

Let $d(k)$ be binary a stochastic variable, $y_s(k)$ the data packet transmitted at sensor node, and $y_c(k)$ received data packet
$d(k) = 1$ corresponds to packet loss and $d(k) = 0$ to no loss



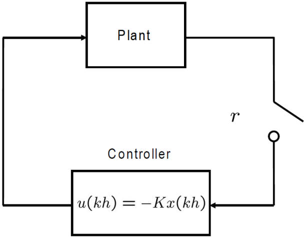
Observer with Loss

An observer handling data loss:

$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma u(k) + \begin{cases} K[y(k) - C\hat{x}(k)], & d(k) = 0 \\ 0, & d(k) = 1 \end{cases}$$

Similar adjustments can be made to the (stochastic) Kalman filter, in which $K = K(k)$ is time varying

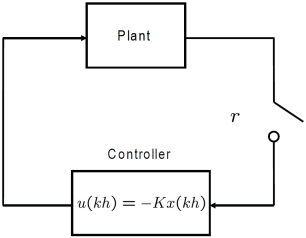
State Feedback Stability with Packet Losses



Plant dynamics: $x((k+1)h) = \Phi x(kh) + \Gamma u(kh)$

Controller: $u(kh) = -K\bar{x}(kh)$, where $\bar{x}(kh) = x(kh)$ if packet is transmitted, $\bar{x}(kh) = \bar{x}((k-1)h)$ otherwise.

State Feedback Stability with Packet Losses



Theorem Suppose that the closed-loop system without packet losses is stable, ie, the eigenvalues of $\Phi - \Gamma K$ are inside the unit circle. Let r be the successful packet reception rate. Then

- if the open-loop system is marginally stable, then the system is exponentially stable for all $0 < r \leq 1$.

State Feedback Stability with Packet Losses (cont.ed)

- if the open-loop system is unstable, then the system is exponentially stable for all

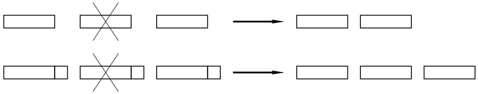
$$\frac{1}{1-\gamma_1/\gamma_2} < r \leq 1\,,$$

where $\gamma_1 = \log[\lambda^2_{\mathsf{max}}(\Phi - \Gamma K)]$, $\gamma_2 = \log[\lambda^2_{\mathsf{max}}(\Phi)]$

- Proofs in [ZBP]

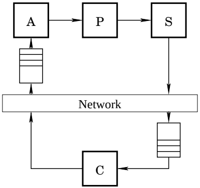
Error Correction

- Data drops are naturally handled through coding
- By introducing parity bits (redundancy), it is possible to reconstruct data at the receiver node even if some packets are lost
- The amount of redundancy $r(k)$ should be minimized in order to maximize data transmission, but still large enough to counteract data drops
- Redundancy $r(k)$ can be controlled based on feedback information on varying network load and other operating conditions



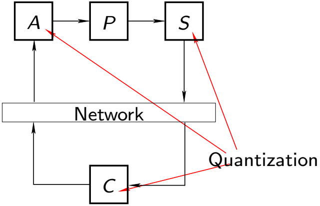
Delay or Loss

- There is a floating boundary between delay and drop
- TCP considers a packet to be lost after a specified time denoted `timeout`
- It can be better to drop data, than use old information for control
- Feedback is forgiving in the sense that communication drops and noise is often attenuated by controller



Quantization

- Quantization in AD converters
- Quantization of controller parameters
- Roundoff, overflow, and underflow in operations (addition etc.)
- Quantization in DA converters



State Feedback Stability with Quantization

- Quantization affects stability properties of nominal closed loop system.
- Can be analyzed in certain cases using Lyapunov techniques.
- Quantization induces error: compensated by feedback in some cases.

Review Lyapunov functions for continuous systems

A differentiable function $V : \mathbb{R}^n \to \mathbb{R}$ is a *Lyapunov function* for $\dot{x} = f(x), f(0) = 0$ if

1. $V(0) = 0$ and $V(x) > 0$, $\forall x \neq 0$
2. $\dot{V}(x) := \dfrac{\partial V}{\partial x} f(x) \leq 0$, $\forall x \neq 0$

Linear Systems $V(x) = x^T P x$, P positive definite, is a (quadratic) Lyapunov function for $\dot{x} = Ax$, if (and only if)

$$A^T P + PA = -Q, \qquad Q \text{ positive semidefinite}$$

because $\dot{V}(x) = x^T(A^T P + PA)x = -x^T Q x \leq 0$.

Stability Test

The solution $x^*(t) = 0$ for $\dot{x}(t) = f(x(t))$ is stable if there exists a Lyapunov function. It is asymptotically stable if, moreover, $\dot{V}(x)$ is negative definite.

Linear Systems A linear system $\dot{x}(t) = Ax(t)$ is asymptotically stable if (and only if) for any positive definite Q, there exists positive definite P such that

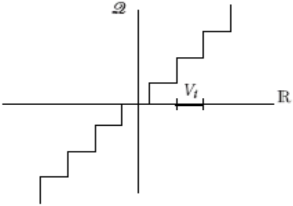
$$A^T P + PA = -Q$$

Stability LTI systems with errors

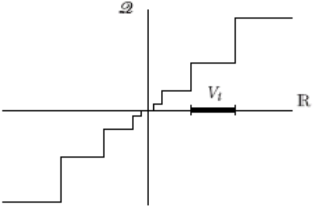
- Consider $\dot{x} = Ax + Bu$. Assume that $u = Kx$ is a stabilizing controller.
- Now assume measurement errors in the feedback, so that control input is $u = K(x + e)$.
- **Fact**: There exists a quadratic Lyapunov function V and $a, b > 0$ for which

$$\dot{V} \leq -a||x||^2 + b||x||||e||$$

Quantization Modelling



(a) Uniform quantizer. The scalar case.

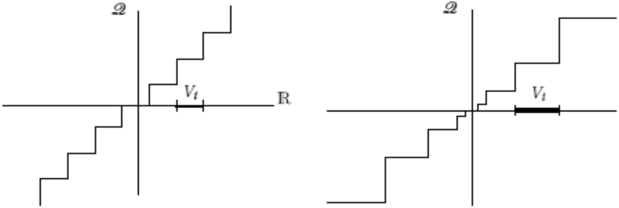


(b) Logarithmic quantizer. The scalar case.

In general, a quantizer $q : \mathbb{R}^n \to \mathcal{Q}$.

- Quantization regions are the sets $\{z \in \mathbb{R}^n | q(z) = i \in \mathcal{Q}\}$.
- Quantization range: highest value of signal that can be mapped to the quantizer.

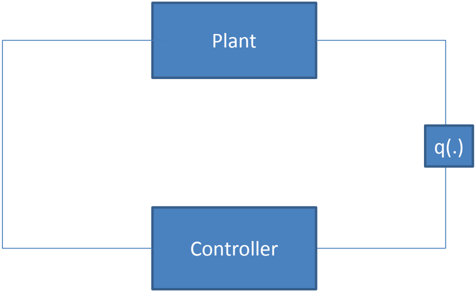
Quantization Error Modelling



(a) Uniform quantizer. The scalar case. (b) Logarithmic quantizer. The scalar case.

- Quantization error: $e(x(t)) = q(x(t)) - x(t)$.
- Uniform quantization: $||q(x(t)) - x(t)|| \leq \delta_u$.
- Logarithmic quantization: $||q(x(t)) - x(t)|| \leq \delta_l ||x(t)||$.

State Feedback Stability with Quantization



Here we consider state quantization. Input and output quantization is also possible.

State Feedback Stability with Quantization

- Plant: $\dot{x} = Ax + Bu$. Assume that $u = Kx$ is a stabilizing controller.
- Controller with quantization: $u = Kq(x)$. ("certainty equivalence controller").

Theorem Suppose that the closed-loop system without quantization is asymptotically stable. Then there exist sufficiently small δ_u, δ_l for which the quantized closed-loop system (i)is asymptotically stable, for the case of logarithmic quantizers, (ii) converges to a region around the equilibrium point, whose size depends on δ_u, for the case of uniform quantizers.
Note: proof based on Fact of slide 30.

State Feedback Stability with Quantization

- Can be extended to input ($u = q(Kx)$) and output quantization.
- Worst-case approach:maximum error is considered in the analysis.
- Alternative to consider stochastic approach (cf. Lecture 4).

Computer Arithmetics

- Control design and analysis are mainly based on high resolution and large range (floating-point) arithmetics: x, y, u supposed to be real valued
- Microcomputers in embedded systems may have fixed-point arithmetic

Analysis and design problems:

- What are the influences of limited word lengths (16 or 32 bits)?
- Is special attention needed in computations (overflow, roundoff)?
- Word length can sometimes be a choice, e.g., special-purpose VLSI circuits in consumer electronics

Floating-Point Arithmetics

IEEE 754 Standard: Numbers represented as

$$\pm a \cdot 2^b$$

where $0 \leq a < 2$ is the *significand*, and b the *exponent*

- Short real: 32 bits (Java/C: *float*)
  - 1 sign + 8 exponent + 23 significand
  - Range 2^{-126}–2^{128}
- Long real: 64 bits (Java/C: *double*)
  - 1 sign + 11 exponent + 52 significand
  - Range 2^{-1022}–2^{1024}

Supports infinity and NaN. Used by most processors, except some digital signal processors (DSP's)

Fixed-Point Arithmetics

- Word given in binary format
- Typical word lengths are 8, 16, and 32 bits
- 16 bits correspond to
  $\{-2^{15}, \ldots, 2^{15}-1\} = \{-32768, \ldots, 32767\}$

Computation properties:

- Result depends on order of computations
- Overflow risk
- Put attention to overflow characteristics (saturation)

Next Lecture

Event-based control and real-time systems

- Event-based control
- Real-time systems and scheduling