Ingegneria dell 'Automazione - Sistemi in Tempo Reale

Selected topics on discrete-time and sampled-data systems

Luigi Palopoli

palopoli@sssup.it - Tel. 050/883444

Outline

- Effects of delay
- Discrete equivalents
- Multirate Systems

An example

- Consider a first order system $\dot{x} = \lambda x + u$

An example

- Consider a first order system $\dot{x} = \lambda x + u$
- $\Gamma_1 = \int_{mT}^{T} e^{\lambda\eta} d\eta = \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda}$, $\Gamma_2 = \frac{e^{m\lambda T} - 1}{\lambda}$

An example

- Consider a first order system $\dot{x} = \lambda x + u$
- $\Gamma_1 = \int_{mT}^{T} e^{\lambda \eta} d\eta = \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda}$, $\Gamma_2 = \frac{e^{m \lambda T} - 1}{\lambda}$
- Augmented discrete-time system:

$$\begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} e^{\lambda T} & \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} + \begin{bmatrix} \frac{e^{m \lambda T} - 1}{\lambda} \\ 1 \end{bmatrix} u(k)$$

An example

- Consider a first order system $\dot{x} = \lambda x + u$
- $\Gamma_1 = \int_{mT}^{T} e^{\lambda\eta} d\eta = \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda}$, $\Gamma_2 = \frac{e^{m\lambda T} - 1}{\lambda}$
- Augmented discrete-time system:

$$\begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} e^{\lambda T} & \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} + \begin{bmatrix} \frac{e^{m\lambda T} - 1}{\lambda} \\ 1 \end{bmatrix} u(k)$$

- Control by static feedback $u(k) = kx(x)$ yields the closed loop dynamic:

$$\begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} e^{\lambda T} + k\frac{e^{m\lambda T} - 1}{\lambda} & \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda} \\ k & 0 \end{bmatrix} z$$

An example

- Consider a first order system $\dot{x} = \lambda x + u$
- $\Gamma_1 = \int_{mT}^{T} e^{\lambda \eta} d\eta = \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda}$, $\Gamma_2 = \frac{e^{m\lambda T} - 1}{\lambda}$
- Augmented discrete-time system:

$$\begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} e^{\lambda T} & \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} + \begin{bmatrix} \frac{e^{m\lambda T} - 1}{\lambda} \\ 1 \end{bmatrix} u(k)$$

- Control by static feedback $u(k) = kx(x)$ yields the closed loop dynamic:

$$\begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} e^{\lambda T} + k\frac{e^{m\lambda T} - 1}{\lambda} & \frac{e^{\lambda T} - e^{\lambda m T}}{\lambda} \\ k & 0 \end{bmatrix} z$$

- Stability can be studied- via Jury criterion - on

$$z^2 - (e^{\lambda T} + k\frac{e^{m\lambda T} - 1}{\lambda})z - k\frac{e^{\lambda T} - e^{\lambda m T}}{\lambda}$$

Solution

- $a_2 < 1 \Rightarrow k > -\frac{\lambda}{e^{\lambda T} - e^{\lambda m T}}$

Solution

- $a_2 < 1 \Rightarrow k > -\frac{\lambda}{e^{\lambda T} - e^{\lambda m T}}$
- $a_2 > -1 - a_1 \Rightarrow k < -\lambda$

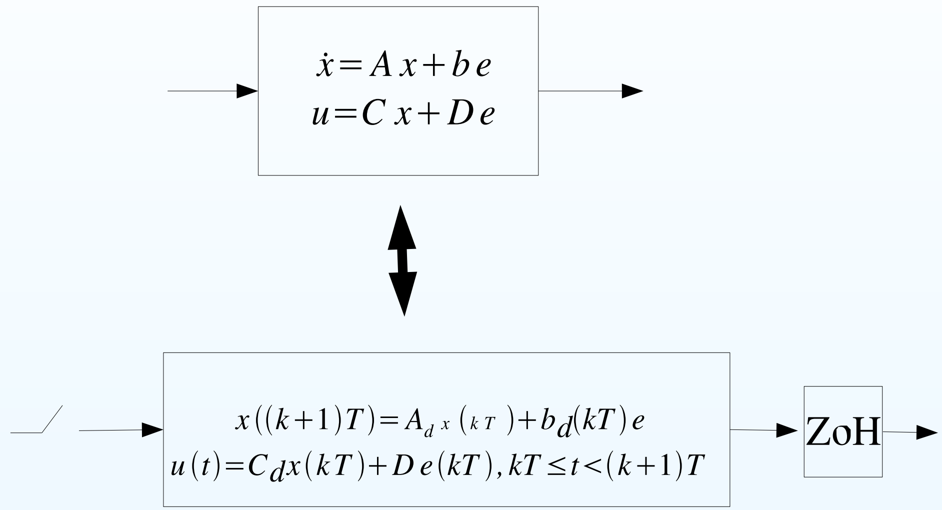
Solution

- $a_2 < 1 \Rightarrow k > -\frac{\lambda}{e^{\lambda T} - e^{\lambda m T}}$
- $a_2 > -1 - a_1 \Rightarrow k < -\lambda$
- $a_2 > -1 + a_1 \Rightarrow \begin{cases} k > -\lambda \frac{e^{\lambda T}+1}{2e^{\lambda m T} - e^{\lambda T} - 1} & \text{if } m > \frac{\log \frac{1+e^{\lambda T}}{2}}{\lambda T} \\ k > \frac{e^{\lambda T}+1}{-2e^{\lambda m T} + e^{\lambda T} + 1}\lambda & \text{Otherwise} \end{cases}$

Outline

- Effects of delay
- Discrete equivalents
- Multirate Systems

Discrete Equivalent



- Given a continuous time controler (A, B, C, D) find a discrete time controller (A_d, B_d, C_d, D_d) that is a good approximation for the CT controller.
- Roughly speaking the problem is finding a good numeric integration algorthm....
- There are different ways for doing it:
  - Forward integration rule
  - Backward integration rule
  - Trapezoidal integration rule
  - ZoH equivalent

Forward integration

- Approximate

$$\dot{x} \approx \frac{x((k+1)T) - x(kT)}{T}$$

- ..that leads to:

$$\frac{x((k+1)T)-x(kT)}{T} = Ax(kT) + Be(kT), u(kT) = Cx(kT) + De(kT) \rightarrow$$
$$x((k+1)T) = (I + AT)x(kT) + Be(kT), u(kT) = Cx(kT) + De(kT)$$

- The rule maps the left laplace half-plane into a square of side 1: a stable system might become unstable!

Backward integration

- Approximation: $\dot{x}(t) \approx \frac{x(kT)-x((k-1)T)}{T}$
- Looking at the state evolution: $\frac{x(kT)-x((k-1)T)}{T} = Ax(kT) + Be(kT)$
- ...from which

$$(I - AT)x(kT) = x((k-1)T) + BTe(kT)$$

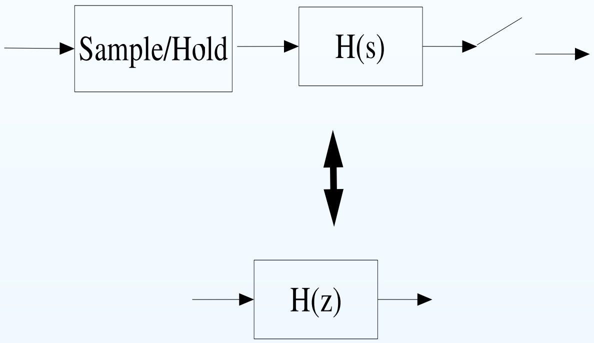
- Introduce $w(kT) = x((k-1)T)$, we get:

$$w((k+1)T) = x(kT) = (I - AT)^{-1}w(kT) + (I - AT)^{-1}BTe(KT)$$

- Output of the controller

$$y(kT) = Cw((k+1)T) + De(kT) = C(I - AT)^{-1}w(kT) + (C(I - AT)^{-1}BT + D)e(kT)$$

ZoH equivalent



- The idea is here to find a discrete equivalent replicating the behaviour of the continuous time controller if it was framed into a ZoH-Sampler scheme
- we know how to do this....

Implementation Scheme

Data structure

```
struct DTSys {
    int nx, ny, nu;
    double * A;
    double * B;
    double * C;
    double * D;
    double * x; /* Current State */
    double * lx; /* Working variable */
}
```

Integration Step

```
void DTStep(struct DTsys * s, double * e, double * u) {
        outputUpdate(s, e, u);
        stateUpdate(s, e);
}
void outputUpdate(struct DTsys * s, double * e, double * u) {
        int i,j; double * x = s->x;
        for (i=0; i < s->ny;i++) { /*output update */
                u[i] = 0;
                for (j=0; j < s->nx;j++)
                        u[i] += s->C[i*s->nx+j]*x[i];
                for (j=0; j < s->nu;j++)
                        u[j] += s->D[i*s->nu+j]*e[j];
        }
};
```

Integration step (continued)

```
void stateUpdate(struct DTsys * s, double * e) {
    int i,j; double * x = s->x; double * lx = s->lx;
    for (i=0; i < s->nx;i++) { /*State update */
        lx[i] = 0;
        for (j=0; j < s->nx;j++)
            lx[i] += s->A[i*s->nx+j]*x[i];
        for (j=0; j < s->nu;j++)
            lx[i] += s->B[i*s->nu+j]*e[j];
        }
    }
    s->x = lx;s->lx = x;
    }
}
```

Initialization

```
double A = {.,.,.,.};/*rows and columns of A*/
double B ={.,.,.,.};/*rows and columns of B*/
double C = {.,.,.,.};
double D = {.,.,.,.};
double x = {.,.};/*Initial state vector */
double Ix = {.,.};/*Working variable as big as the state vector */
struct DTsys mySystem = {/*Create and initialize a system structure */
      2,2,2,A,B,C,D,x,Ix
}
```

Task structure

```
TASK() {
    double e[2];
    double u[2];
    ...
    for(;;) {
        getE(e);/* Get sensor readings in e*/
        /* e is ouput of the plant
        and input to the controller *
        DTStep(&mySystem,e,u);
        putU(u);/*Write to the actuators */
    }
}
```

Consideration

- Is this the best we can do?
- Not at all!!
- We are unnecssearily delaying the emission of the new output!
- A better solution:

```
TASK() {
    ...
    for(;;) {
        getE(e);/* Get sensor readings in e*/
        outputUpdate(&mySystem,e,u);
        putU(u);/*Write to the actuators */
        stateUpdate(&mySystem,e);
    }
}
```

Outline

- Effects of delay
- Discrete equivalents
- Multirate Systems

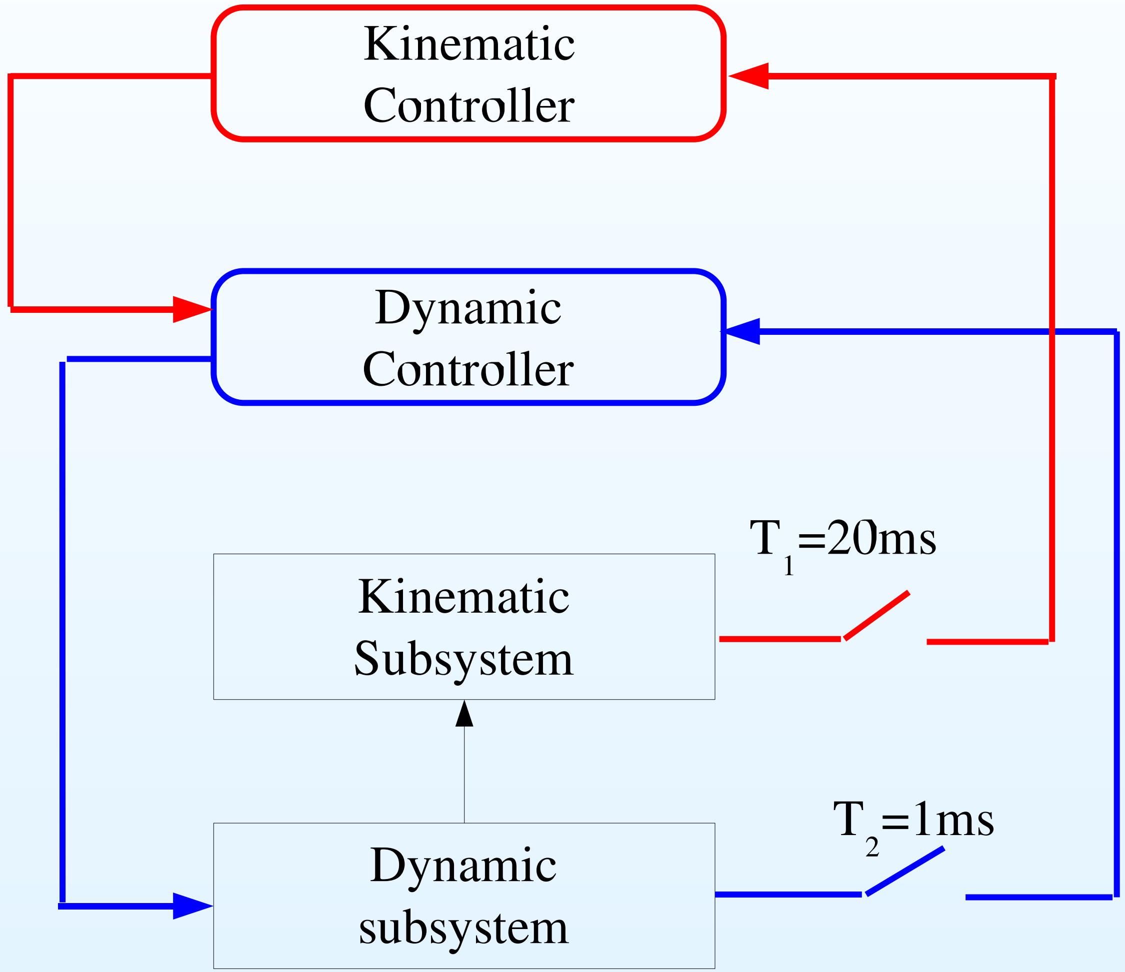
Outline

- Multirate and periodic systems
  - Modeling
  - Stability
  - LTI representations

Motivation

- Frequently a system is comprised of several subsystems
- In some cases data can be collected with different frequencies
- It can be useful to design several interconnected controllers activated at different frequencies

Example



Kinematic Controller
Dynamic Controller
T1=20ms
Kinematic Subsystem
T2=1ms
Dynamic subsystem

Example (continued)

- Assumption: sampling periods are integre multiples of the smallest period
- Let $x_1 \in \mathbb{R}^{n_1}$ be kinematic variables, and $x_2(k) \in \mathbb{R}^{n_2}$ be dynamic variables
- Equation (sampled at the slowest period):

$$x_1(k+1) = A_{1,\,1}\, x_1(k) + A_{1,\,2}\, x_2(k)$$
$$x_2(k+1) = A_{2,\,2}\, x_2(k) + B_2\, u(k)$$

- The different periods for data measurements can be modeled assuming a periodic C matrix: $y(k) = C(k)\left[\begin{smallmatrix} x_1(k) \\ x_2(k) \end{smallmatrix}\right]$ where

$$C(k) = \begin{cases} I_{(n_1+n_2)\times(n_1+n_2)} & k = hT, T = T_1/T_2, h \in \mathbb{N} \\ \left[\begin{smallmatrix} 0 & I_{n_1 \times n_1} \end{smallmatrix}\right], & \text{otherwise.} \end{cases}$$

- Observe: the dimensions of the output space change in time...

Example (continued)

- Closed loop dynamics
  - we can assume that data communicated from the kinematic to dynamic controller are held throughout the period T_2
  - to modify this sample and hold intracontroller mechanism we need a further state variable
  - The resulting clodes loop dyanmics will be something like:

$$x(k+1) = A_c(k)x(k)$$

where matrix A_c is periodic with period $T = T_1/T_2$

Periodic systems

- From the example above it is clear that multirate control systems can easily be modeled by the use of periodic (i.e. periodically timevarying systems)
- Plants can be modeled as:

  $$x(k+1) = A(k)x_(k) + B(k)u(k)$$
  $$y(k) = C(k)x(k) + D(k)u(k)$$

  where $A(k+T) = A(k), B(k+T) = B(k), C(k+T) = C(k), D(k+T) = D(k)$

- In particular closed loop autonomous systems are modeled as $x(k+1) = A(k)x(k), A(k+T) = A(k)$

Outline

- Multirate and periodic systems
  - Modeling
  - Stability
  - LTI representations

Stability results

- Consider the autonomous periodic system

$$x(k+1) = A(k)x(k), \; A(k+T) = A(k) \tag{0}$$

- Consider the subsequence $\tilde{x}(h) = x(k_0 + hT)$ for some k_0; the sequence is described by

$$\tilde{x}(h+1) = \tilde{A}_{k_0}\tilde{x}(h), \; \text{where } \tilde{A}_{k_0} = A(k_0 + T - 1)A(k_0 + T - 2)...A(k_0) \tag{0}$$

- Fact: System (1) is [Asymptotically] stable $\Leftrightarrow$ System (2) is
  - $\Rightarrow$ obvious
  - $\Leftarrow$ (intuition) The intersampling evolution (due to the finitiness of the period) is norm-bounded: there exist a constant M such that $x(k) \leq M\tilde{x}(h), \forall k \in [hT, h(T+T) - 1[$, so if $\tilde{x}(h)$ vanishes so dows $x(k)$

Stability results

- More formally...:
  - Lemma: the eigenvalues of the matrix $\tilde{A}_{k_0} = A(k_0+T-1)A(k_0+T-2)...A(k_0)$, called *monodromy matrix*, are indpendent of k_0
  - System (1) is stable if and only if the monodromy matrix is Schur-stable (i.e. its eignecvalues are all in the unit circle)
- Based on the observation that the stability can evaluated on the periodic subsampled sequence we can build lyapunov inspired criteria...
- Example: the system is A.S. if there exist periodic definite matrices $P(k)$ that respect the following linear matrix inequalities:
$$\begin{aligned} &A^T(0)P(1)A(0) - P(0) \prec 0 \\ &A^T(1)P(2)A(1) - P(1) \prec 0 \\ &\ldots \\ &A^T(T-1)P(0)A(T-1) - P(T-1) \prec 0 \end{aligned}$$
- With some trick (Schur complements) the above can be used as a synthesis tool for finding stabising gains (by using Linear Matrix inequalities solvers)

Outline

- Multirate and periodic systems
  - Modeling
  - Stability
  - LTI representations

LTI representations

- Consider the system:

$$
\begin{aligned}
&x(k+1) = A(k)x_(k) + B(k)u(k) \\
&y(k) = C(k)x(k) + D(k)u(k) \\
&\text{where } A(k+T) = A(k), B(k+T) = B(k), C(k+T) = C(k), D(k+T) = D(k) \\
&x \in \mathbb{R}^n, y \in \mathbb{R}^p u \in \mathbb{R}^m
\end{aligned} \tag{0}
$$

- We want to find LTI equivalent systems for which important properties are preserved

Lifting

- The idea of considering periodically extracted subsequences for stability can be exploited also in general terms
- The lifted representation for the system is achieved by the following steps:
  - sampling the evolution of the state with the same periodicitiy of the system
  - grouping inputs used throughout a period and output emitted throughout a period into augemented vectors
  - Lifted state: $\hat{x}^{(h_0)}(h) = x(h_0 + hT)$
  - Lifted inputs:

$$\hat{u}^{(h_0)}(k) = \begin{bmatrix} u(h_0 + hT) \\ u(h_0 + hT + 1) \\ \dots \\ u(h_0 + hT + T - 1) \end{bmatrix}, \ \hat{y}^{(h_0)}(k) = \begin{bmatrix} y(h_0 + hT) \\ y(h_0 + hT + 1) \\ \dots \\ y(h_0 + hT + T - 1) \end{bmatrix}$$

Lifting (continued)

- the lifted system is described by (h_0 specification omitted):

$$\begin{aligned} \hat{x}(h+1) &= \hat{A}\hat{x}(h) + \hat{b}\hat{u}(h) \\ \hat{y}(h) &= \hat{C}\hat{x}(h) + \hat{D}\hat{u}(h) \end{aligned} \qquad (0)$$

where

$$\hat{A} = A(h_0 + T - 1) \dots A(h_0)$$

$$\hat{b} = [\hat{b}_1 \hat{b}_2 \dots \hat{b}_T], \text{ with } \hat{b}_i = A(h_0 + T - 1) \dots A(h_0 + i)B(h_0 + i - 1$$

$$\hat{C} = \begin{bmatrix} \hat{c}_1 \\ \hat{c}_2 \\ \dots \\ \hat{c}_T \end{bmatrix}, \text{ with } \hat{c}_i = C(h_0 + i - 1)A(h_0 + i - 2) \dots A(h_0)$$

$$\hat{D} = \{\hat{d}_{i,j}\} \text{ with } i, j = 1, 2, \dots T \text{ and}$$

$$\hat{d}_{i,j} = \begin{cases} C(h_0 + i - 1)A(h_0 + i - 2 \dots A(h_0 + j)B(h_0 + j - 1) & i < j \\ 0, & i \geq j \end{cases}$$

Lifting (continued)

- The reachability space of $[A(), B()]$ a $h = h_0$ coincides with the reachibility space of $[\hat{A}, \hat{B}]$
- The pair $[A(), B()]$ is stabilisable iff the pair $[\hat{A}, \hat{B}]$ is
- The observability space of $[A(), B()]$ at $h = h_0$ coincides with the observability subspace of $[\hat{A}, \hat{C}]$
- The pair $[A(), C()]$ is detectable iff the pair $[\hat{A}, \hat{B}]$ is