EE185524

Tracking of Linear System with Delay

Yurid Eka Nugraha

TSP DTE FTEIC ITS

Introduction: network induced delays

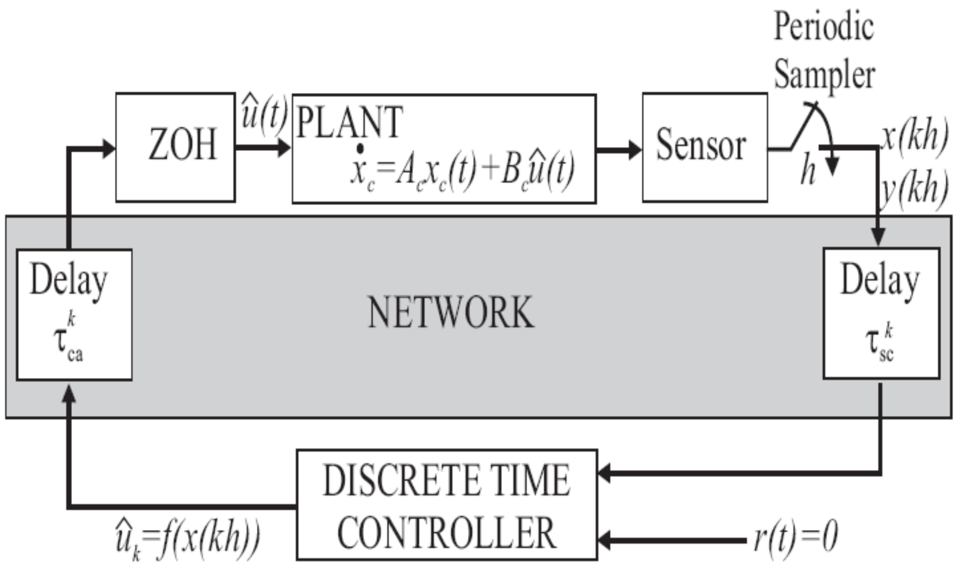
Information flow in the control loop is delayed due to

- buffering,
- access contention (the time a node waits until it gets access to the network),
- computation and transmission delays,
- etc.

Network-induced delays in NCS appear in the information flow between

- Sensor and controller $\{\tau_{\mathrm{sc}}(k)\}$, (controller receives "outdated" information about process behavior)
- Controller and actuator $\{\tau_{\mathrm{ca}}(k)\}$, (control action cannot be applied "on time" and the controller does not know the exact instance the calculated control signal will be received by the actuator)

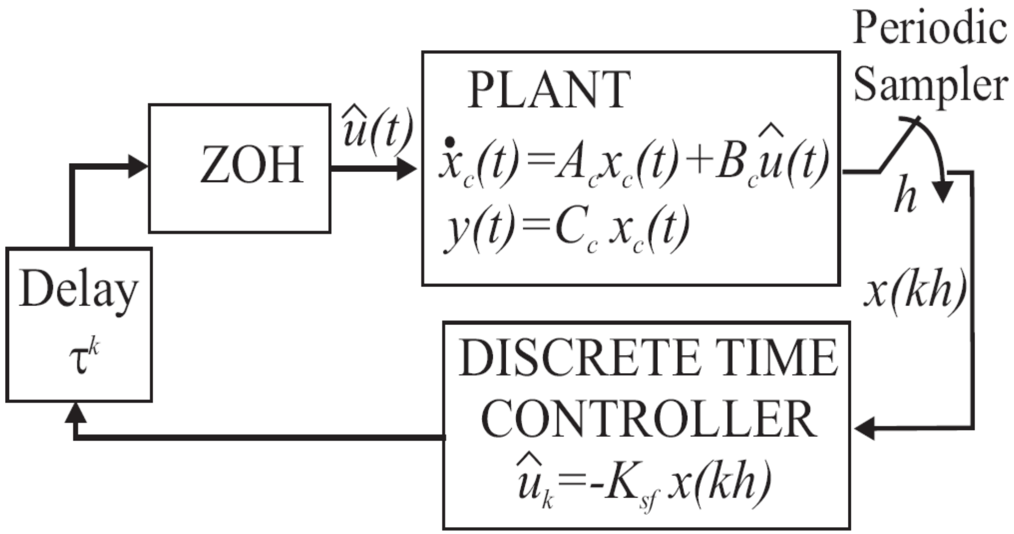
Introduction: network induced delays



Network-induced delays in NCS
$\tau_{sc}(k)$ and $\tau_{ca}(k)$

When a static LTI controller is employed, can lump the two delays into

$$\tau[k] = \tau_{sc}[k] + \tau_{ca}[k]$$



Tracking control design for NCS

The usual approach for NCS Analysis & Design:

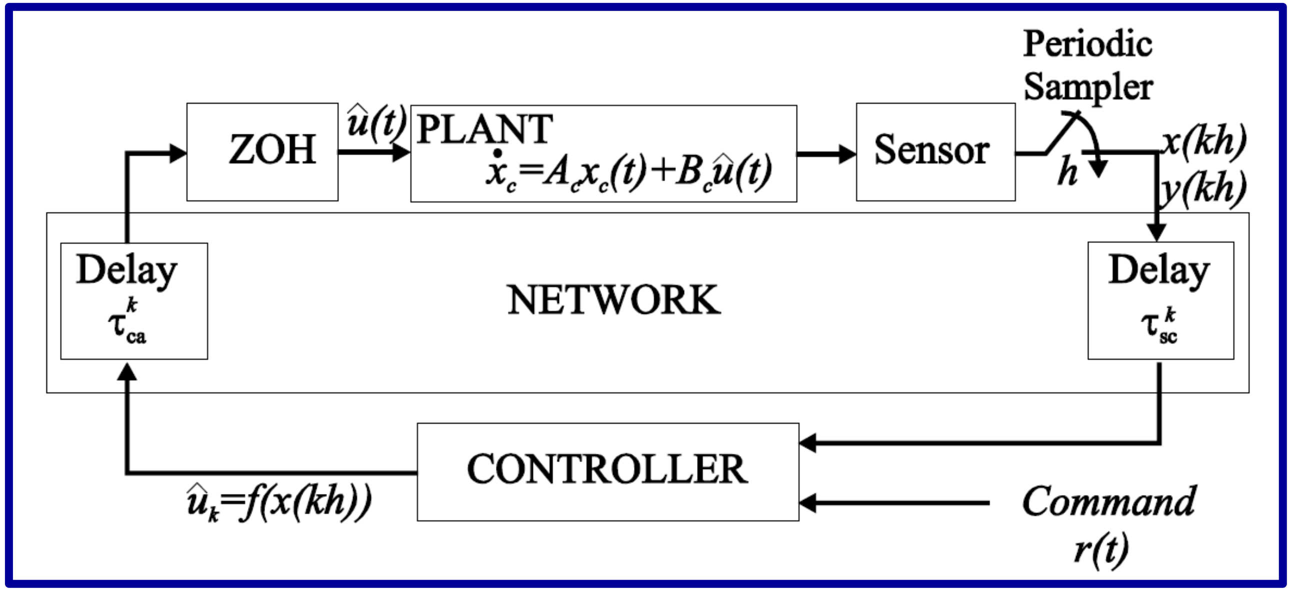
- design a controller ignoring the network, then
- analyze stability, performance and robustness with respect to the effects of network-delays and scheduling policy…(usually via the selection of an appropriate scheduling protocol).

Tracking control design?

- most of the NCS publications concerns regulation: "design a controller which brings the output/state to zero"
- many results on tracking for **T**ime **D**elayed **S**ystems (TDS) but cannot be applied as it is to NCS due to the "Network-centric" nature of NCS e.g.
  - special nature of delays in NCS,
  - the fundamental issue of "Packet Loss/Drops",
  - etc.

NCS modeling in linear system

NCS with network-induced delays in the actuation and sensor path



Assumptions made ...

- Dynamics: a combination of a continuous–time LTI plant with a discrete–time controller
- Time Invariant controller → can lump $\boldsymbol{\tau}_{sc}(k)$, $\boldsymbol{\tau}_{ca}(k)$, into $\boldsymbol{\tau}^k = \boldsymbol{\tau}_{sc}(k) + \boldsymbol{\tau}_{ca}(k)$
- Single source of uncertainty and performance degradation → **the lumped transmission delay $\boldsymbol{\tau}^k$.**
- No plant uncertainties or nonlinearities - No packet drops

NCS modeling in linear system

In practice:

- the dynamics of the NCS under investigation is a combination of a continuous–time uncertain/nonlinear plant with a discrete–time ("sampled-data") controller.
- the sampler is **time-driven**, whereas both controller and actuator are **event-driven**
- some packets are lost or intentionally dropped (contain obsolete/useless info)

Delays $\tau^k_{\mathrm{sc}}, \tau^k_{\mathrm{ca}}, \tau^k < h$

- $\tau^k_{\mathrm{sc}} = \tau_{\mathrm{sc}}[k]$: delay experienced by a state or output sample x(kh), y(kh), sampled at "kh" and presented –after a delay $\boldsymbol{\tau^k_{sc}}$ to the event–driven remote controller for control computation purposes.

- $\tau^k_{\mathrm{ca}} = \tau_{\mathrm{ca}}[k]$: delay experienced by the control–action, computed immediately after its reception at time instance kh+ $\boldsymbol{\tau^k_{sc}}$ until it is transmitted via the network to the Z.O.H (and finally presented to the event–driven actuator).

- $\boldsymbol{\tau}^k$: Total delay within the kth sampling period, i.e. the time **from** the instant when the sampling node samples sensor data from the plant **to** the instant when actuators exert a control action –**whose computation was based on this sample**– to the plant.

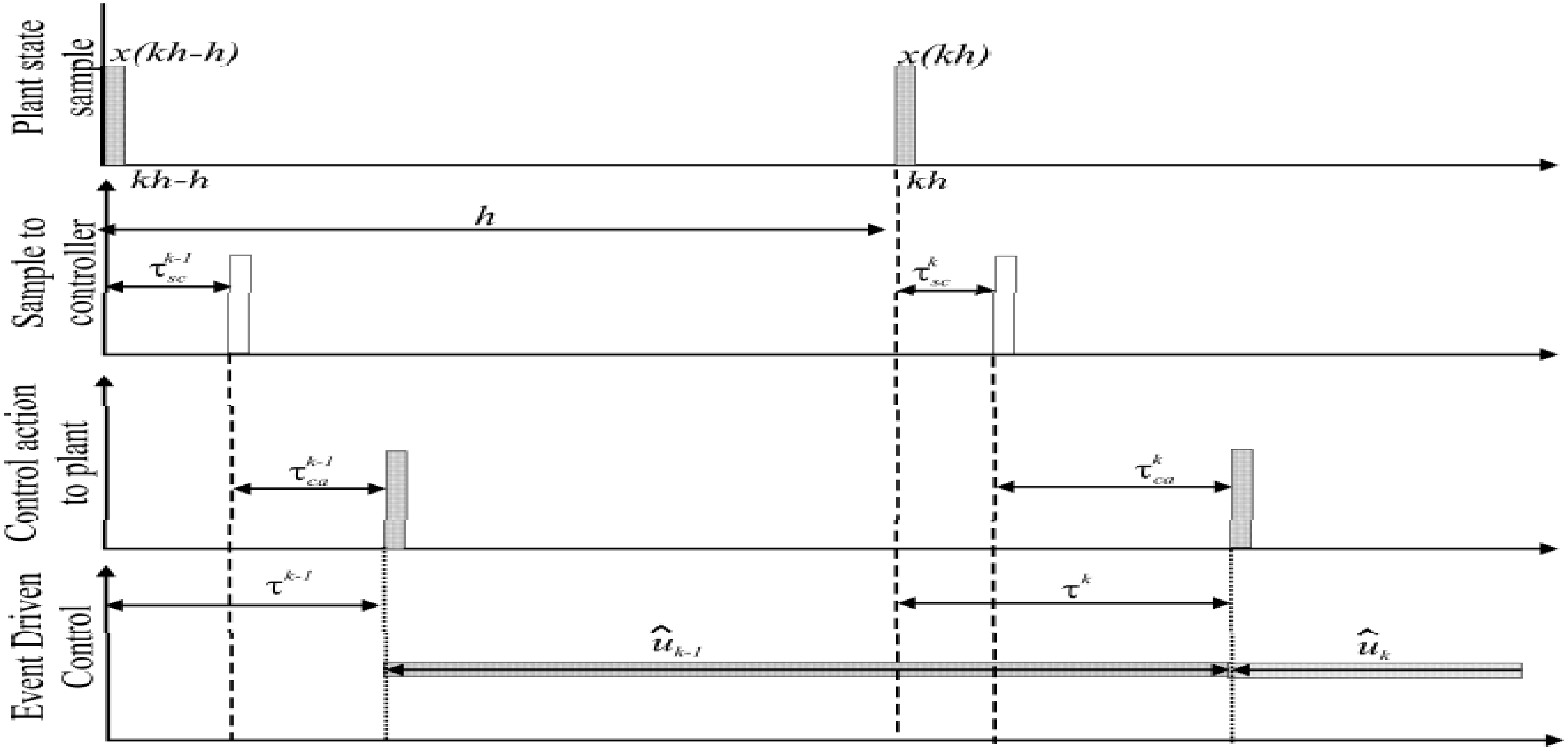
- $\tau[k] = \tau_{\mathrm{sc}}[k] + \tau_{\mathrm{ca}}[k]$

(since a static time invariant control law is employed)

Known Bounds:

$$\mathbf{0 \leq \boldsymbol{\tau}_{min} < \boldsymbol{\tau}^k < \boldsymbol{\tau}_{max} \leq h}$$

NCS timing diagram



NCS modeling: Discrete controller

$$
\begin{aligned}
\dot{x}(t) &= A_c x(t) + B_c \hat{u}(t), \quad y(t) = C_c x(t) \\
& \quad t \in \left[kh + \tau^k, kh + h + \tau^{k+1}\right) \\
& \quad 0 \le \tau_{\min} < \tau^k < \tau_{\max} \le h \\
\hat{u}(t) &= \begin{cases} \hat{u}_{k-1}, & t \in \left[kh - h + \tau^{k-1}, \ kh + \tau^k\right) \\ \hat{u}_k, & t \in \left[kh + \tau^k, \ kh + h + \tau^{k+1}\right) \end{cases}.
\end{aligned}
$$

- $\hat{u}(t)$: the "most recent" control action presented to the event–driven actuator at the time instance t within a sampling period [kh, kh + h) & can take two values $\hat{u}_k$ or $\hat{u}_{k-1}$
- $\hat{u}(t)$ **experiences a "jump" at the <u>uncertain or unknown</u> time instance kh+ τ^k**, changing from $\hat{u}_{k-1}$ into $\hat{u}_k$ (uncertain actuation instance)
- <u>**Very Complicated Dynamics**</u> → Impulse Delayed Systems, Asynchronous Dynamical Systems, Hybrid Systems, etc. even for the regulation case (r=0)

NCS modeling: the issue of network-induced delays

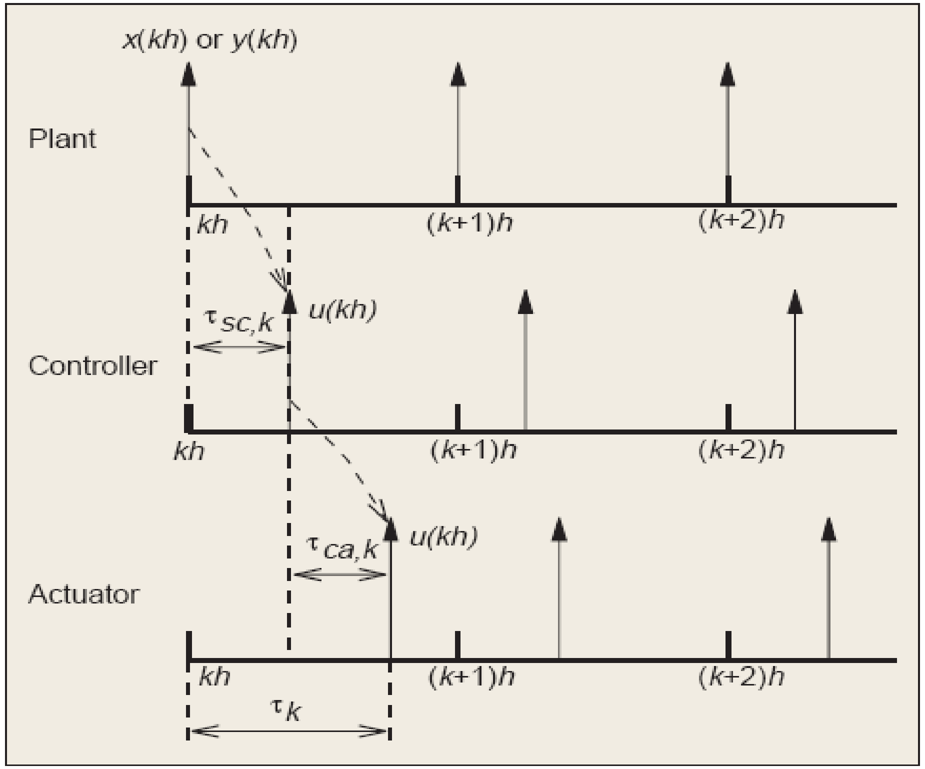


Figure 5. *Network-induced delay.*

NCS Timing Diagram form Zhang & Branicky paper (IEEE Control Systems Magazine, Febr.2001).

the symbol "$u(kh)$" denotes the actuation that takes place at $\underline{kh + \tau^k}$ and its <u>value</u> is $\underline{u(kh) =} -Kx(kh)$

Hence (unless $\boldsymbol{\tau}^k$ is constant) it is not possible to treat the ensuing NCS in a standard "sampled data" or "time-delayed" setting. Instead a "hybrid" setup should rather be used.

Discretization of NCS state equation

$$\tau^k < h \rightarrow x_k = x(kh)$$

$$\mathbf{x_{k+1} = \Phi\, x_k + \Gamma_0(\tau^k)\, \hat{u}_k + \Gamma_1(\tau^k)\, \hat{u}_{k-1}} \qquad (\Sigma 1)$$

- Notation x_k, x_{k-1},… denotes the values x(kh), x(kh-h), … of the periodically sampled discrete–time signal coming out of the sampler. The same notation for y_k, y_{k-1},…

- We keep the "hat" notation for $\hat{u}_k$,$\hat{u}_{k-1}$ as a reminder of the asynchronous, ("jump") nature of these signals.

- $\tilde{O}_n$:*n*-column zero vector, I_n is the *n x n* identity matrix, 0_n is the *n x n* zero matrix.

- M^T: the transpose of a matrix. M > 0 (< 0) means that M is positive (negative) definite.

Discretization of NCS state equation

- Exact discretization between equidistant sampling instances → finite dimensional dynamics...
- The uncertain time varying delay $\boldsymbol{\tau}^{\mathbf{k}}$ can still take any (out of infinite) values within the allowable interval
- The uncertainty of $\boldsymbol{\tau}^{\mathbf{k}}$ → generates an uncertainty in the actuation instance →
  - System matrices $(\Gamma_0(\tau^k), \Gamma_1(\tau^k))$ are uncertain
  - Presence of a delayed input term $\hat{u}_{k-1}$

Exact discretization despite the jump nature of $\hat{u}(t)$

State Equation: $\boxed{\dot{x}(t) = A_c x(t) + B_c \hat{u}(t),\ y(t) = C_c x(t) \quad (1)}$ leads to

$$x(kh+h) = \exp(A_c h)x(kh) + \int_{kh}^{kh+\tau^k} \exp(A_c(kh+h-s))B_c \hat{u}_{k-1} + \int_{kh+\tau^k}^{kh+h} \exp(A_c(kh+h-s))B_c \hat{u}_k ds$$

Define the three matrices Φ, Γ_0, Γ_1

$$\Phi = \exp(A_c h), \quad \Gamma_1(\tau^k) = \int_{kh}^{kh+\tau^\kappa} \exp(A_c \overset{\lambda}{\overline{(kh+h-s)}})B_c ds, \quad \Gamma_0(\tau^\kappa) = \int_{kh+\tau^\kappa}^{kh+h} \exp(A_c \overset{\lambda}{\overline{(kh+h-s)}})B_c ds$$

$x(kh+h) = \Phi x(kh) + \Gamma_0(t^k)\hat{u}_k + \Gamma_1(t^k)\hat{u}_{k-1}$, where: $\Gamma_0(\tau^k) = \int_0^{h-\tau^k} \exp(A_c\lambda)B_c d\lambda$, $\Gamma_1(\tau^k) = \int_{h-\tau^k}^{h} \exp(A_c\lambda)\mathrm{B}_c d\lambda$

we have used the following: $\lambda = kh+h-s$ ($so\ d\lambda = -ds$ since h $const.$) and changing the variable of integration into $d\lambda = -ds$, the new limits of integration are $(h-\tau^k)$ and 0 so we get the simplified expression for Γ_0 :

$$\Gamma_0(\tau^k) = \int_{kh+\tau^\kappa}^{kh+h} \exp(A_c(kh+h-s))B_c ds = \int_{h-\tau^k}^{0} \exp(A_c\lambda)\mathrm{B}_c d(-\lambda) = \int_0^{h-\tau^k} \exp(A_c\lambda)\mathrm{B}_c d\lambda$$

Exact discretization despite the jump nature of $\hat{u}(t)$

Similarly from the definition of Γ_1, using the same change of variables as previously

$(\lambda = kh + h - s \Rightarrow d\lambda = -ds)$ the new limits of integration are h and $(h - \tau^k)$ so we get :

$$\Gamma_1(\tau^k) = \int_{kh}^{kh+\tau^\kappa} \exp(A_c(kh+h-s))B_c ds = \int_{h}^{h-\tau^k} \exp(A_c\lambda)\mathrm{B}_c d(-\lambda) = \int_{h-\tau^k}^{h} \exp(A_c\lambda)\mathrm{B}_c d\lambda$$

Moreover assuming there is no uncertainty on output matrix we have : $y(kh) = C_c x(kh)$ or $y_k = C_c x_k$

A usefull identity for calculating integrals of matrix exponential functions

$(such\ as\ \Gamma_0(\tau^k))$.

$$\exp\left(\begin{bmatrix} X_{n\ x\ n} & Y_{n\ x\ n} \\ 0_{m\ x\ n} & 0_{m\ x\ m} \end{bmatrix} t\right) = \begin{bmatrix} e^{(X\ t)} & \int_0^t e^{(Xr)} Y dr \\ 0_{m\ x\ n} & I_m \end{bmatrix}$$

To Compute $\Gamma_0(\tau^k) = \int_0^{h-\tau^\kappa} e^{(A_c\lambda)}\mathrm{B}_c d\lambda$ we can use above identity as:

$$\Gamma_0(\tau^k) = [I_n \quad \overline{0}^T]\, e^{\left(\begin{bmatrix} Ac & Bc \\ \overline{0} & 0 \end{bmatrix}(h - \tau^k)\right)} \begin{bmatrix} \overline{0}^T \\ 1 \end{bmatrix}$$ Identity I.

$where\ \overline{0} = 0_{1\ x\ n}$ the zero row vector with n columns

Exact discretization despite the jump nature of $\hat{u}(t)$

$$\Gamma_0(\tau^k) = \int_0^{h-\tau^k} \exp(A_c\lambda)B_c d\lambda, \quad (3.A).$$

$$\Gamma_1(\tau^k) = \int_{h-\tau^k}^{h} \exp(A_c\lambda)B_c d\lambda. \quad (3.B).$$

From 2nd equation: $\Gamma_1(\tau^k) = \int_{h\text{-}\tau^k}^{h} e^{(A_c\lambda)}\mathrm{B}_c d\lambda = \int_{h\text{-}\tau^k}^{0} e^{(A_c\lambda)}\mathrm{B}_c d\lambda + \int_0^h e^{(A_c\lambda)}\mathrm{B}_c d\lambda$

$$= -\Gamma_0(\tau^k) + \int_0^h e^{(A_c\lambda)}\mathrm{B}_c d\lambda$$

Decomposing the delays

$$\tau^k = \tau^\circ + \tau^k_\Delta$$

Examples:

- $\tau^o = \tau_{avg}$

$$\begin{aligned}\tau^k &= \left(\frac{\tau_{\max} + \tau_{\min}}{2}\right) + \left(\frac{\tau_{\max} - \tau_{\min}}{2}\right)\delta \\ &= \tau_{avg} + \left(\frac{\tau_{\max} - \tau_{\min}}{2}\right)\delta, \quad |\,\delta\,| \leq 1\end{aligned}$$

- τ^o is chosen as constant and known («*semi-arbitrary*»)
- Use of "Min Max" techniques for selection of τ^o
- The nominally delayed system, Stability Analysis and Controller Synthesis depend on the (user's) choice of τ^o

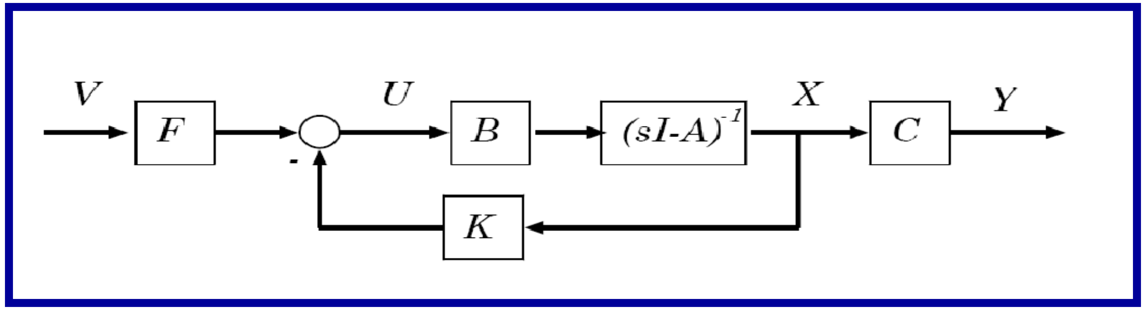
Decomposing the delays

$$\begin{aligned}\Gamma_0(\tau^k) &= \int_0^{h-\tau^k} \exp(A_c\lambda)B_c d\lambda \\ &= \int_0^{h-\tau^\circ} \exp(A_c\lambda)B_c d\lambda + \int_{h-\tau^\circ}^{h-\tau^k} \exp(A_c\lambda)B_c d\lambda \\ &\triangleq \Gamma_0(\tau^\circ) + \Delta\Gamma_0(\tau^k) \quad (4.C).\end{aligned}$$

$$\begin{aligned}\Gamma_1(\tau^k) &= \int_{h-\tau^k}^{h} \exp(A_c\lambda)B_c d\lambda \\ &= \int_{h-\tau^k}^{h-\tau^\circ} \exp(A_c\lambda)B_c d\lambda + \int_{h-\tau^\circ}^{h} \exp(A_c\lambda)B_c d\lambda \\ &\triangleq \Delta\Gamma_1(\tau^k) \;+\; \Gamma_1(\tau^\circ) \\ &= [-\Delta\Gamma_0(\tau^k)] + \left[-\Gamma_0(\tau^\circ) + \int_0^h \exp(A_c\lambda)B_c d\lambda\right] (4.D)\end{aligned}$$

Design of simple (set point) tracking controllers

- The reference signal to be tracked by the output is (piecewise) constant (a "set point")
- **Assumption**: both the plant and the controller are CT LTI systems
- Since the controller is time invariant, can lump the delays into
- A "naïve" tracking controller consists of two parts: Feedback & Feedforward $\tau[k] = \tau_{\rm sc}[k] + \tau_{\rm ca}[k]$
  - The feedback part $(-Kx(t))$ assures closed-loop stability $u(t) = -Kx(t) + Fr$
  - The feedforward part (F) assures that the static gain is "1" (Stable Transfer Function from r to y)



Design of simple (set point) tracking controllers

Considering the continuous–time Linear Time Invariant system

$$\dot{x}(t) = A_c x(t) + B_c u(t), \quad y(t) = C_c x(t),$$

the control law

$$u(t) = -Kx(t) + Fr(t)$$

guarantees asymptotic output tracking if the feedback gain K is a stabilizing state–feedback gain ($A_c - B_c K$ is Hurwitz stable) and the feedforward gain F is selected as

$$F = -\{C_c(A_c - B_c K)^{-1} B_c\}^{-1}$$

or equivalently (use "Matrix Inversion Lemma")

$$F = (K A_c^{-1} B_c - I)(C_c A_c^{-1} B_c)^{-1}$$

Suffers from three drawbacks ("naïve"):

- the plant must not contain integrators (system matrix A is nonsingular)
- cannot handle disturbances and/or model uncertainties (it is NOT Robust)
- number of inputs $\geq$ number of outputs ("overactuation")

Robustness of tracking performance under NCS

Numerical Example 1: a networked stable & minimum phase system

open–loop stable continuous–time system $G(s) = \frac{2}{s^2+3s+2}$, with state space description:

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \end{aligned}$$

- A stable & minimum phase (=zeros in LHP) system
- Infinite Gain Margin
- "Lightly Damped" = stable poles close to the Imaginary axis → "damping ratio" is small → damped oscillative open-loop behaviour (typical in aerospace and "flexible space structure" applications)
- Tracking controller was designed via LQR with $R = 1$, $Q = 1000 * I_2$

  $u(t) = -30.63x_1(t) - 30.63x_2(t) + 31.63r$

Robustness of tracking performance under NCS

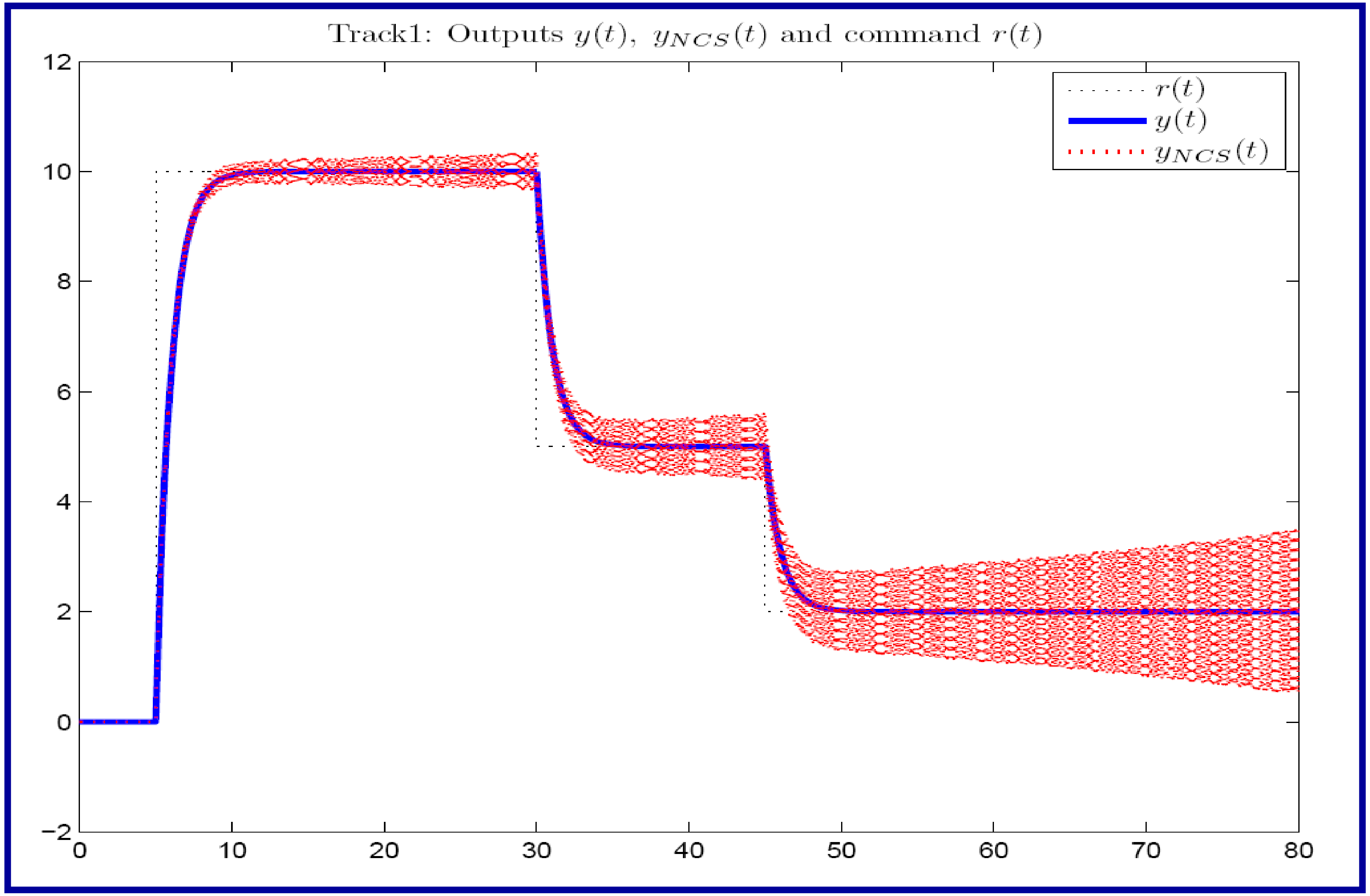
The Networked Version with constant delay $\boldsymbol{\tau}^{k}$

$$\tau_{sc} = \tau_{ca} = 0.0131 \text{ s} \rightarrow \quad \tau^{k} = \tau_{sc} + \tau_{ca} = 0.0262\text{s}$$

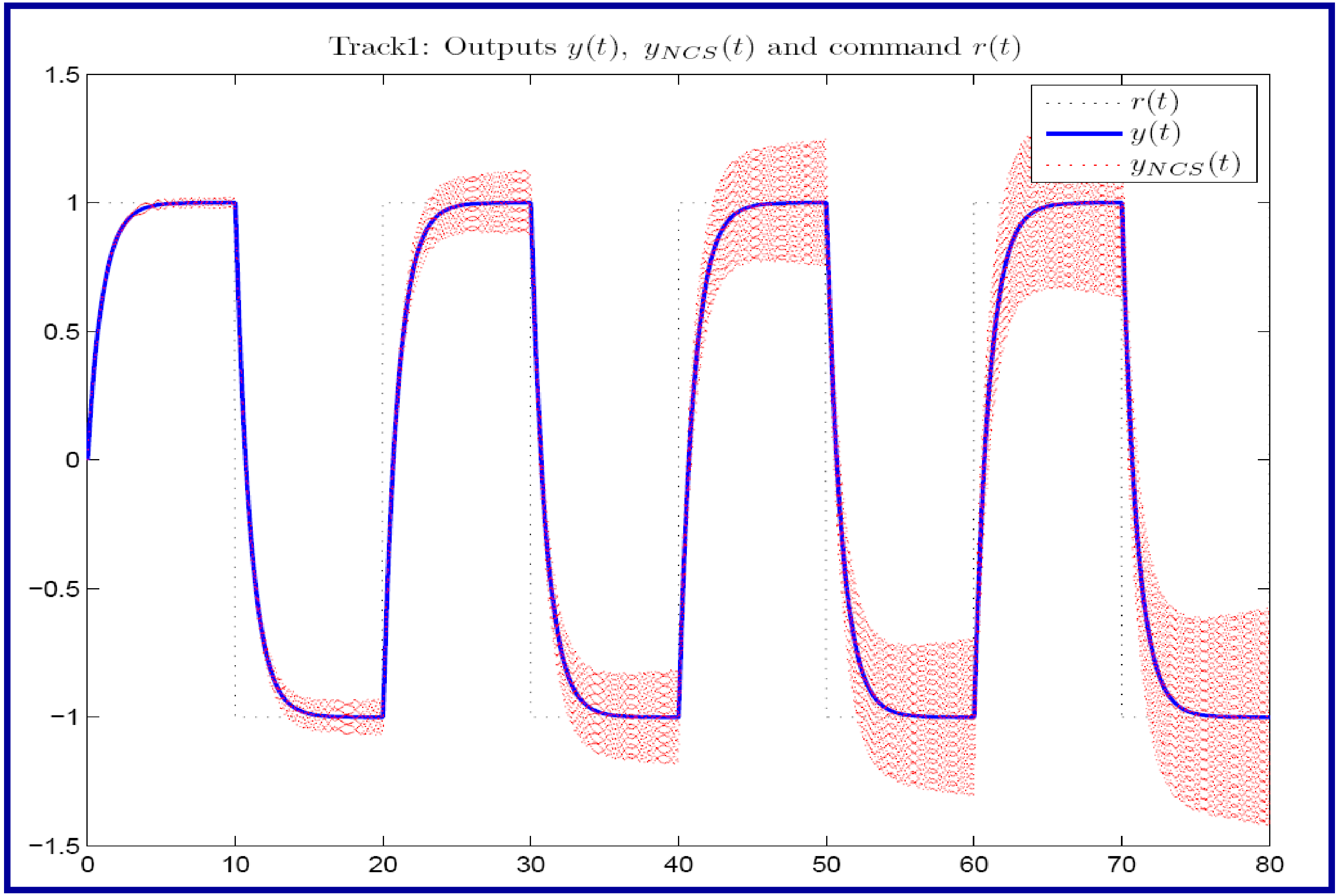
- Assuming $\tau^{k} \leq h$ this corresponds (for the discrete time control case) to a sampling frequency of 38Hz: a relatively "slow sampling"...
- "slow sampling" is typical for NCS (fast sampling → increases # of packets → increases network traffic → increases chances for collisions → packet loss/drops)
- **7th order Pade Approximation** used in simulations for the constant time-delay
- Reference signal(s) $\boldsymbol{r}$ are (piecewise) constant:
  - combination of step functions or
  - square pulse with period slower than the system's time constants

Robustness of tracking performance under NCS

- Simulation needs time for Instability to occur



Robustness of tracking performance under NCS



Robustness of tracking performance under NCS (uncertain delay)

- The Networked Version with **uncertain time-varying delay τ^k** varying between $\tau_{min} = 0$ and $\tau_{max} = 0.0312\text{s} < h$ corresponding to a sampling frequency 32 Hz

- Implementation used in simulations:

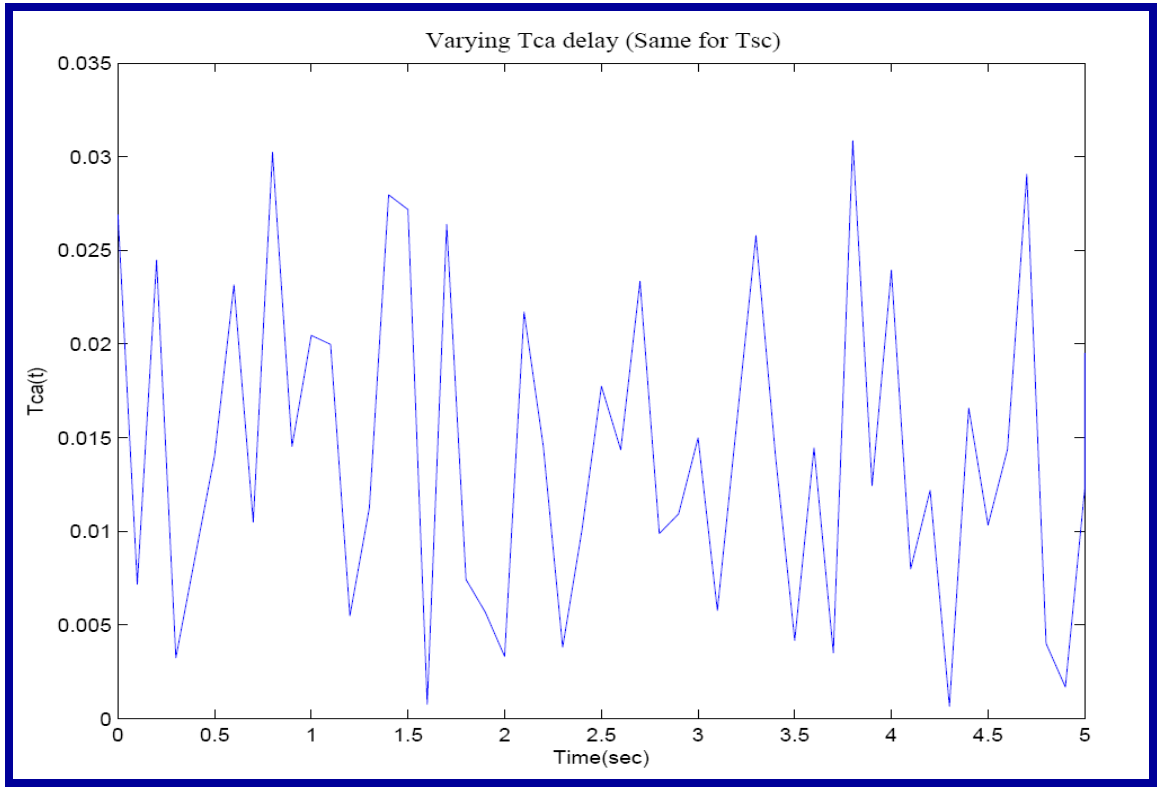
$$\boldsymbol{\tau^k = \tau^o + \delta\, \tau_{unc}}, \qquad \boldsymbol{|\delta| < 1}$$

with $\tau^o = \tau_{avg} = (\tau_{max} + \tau_{min})/2 = 0.0156$ s being the "mean value" (a constant nominal delay) and $|\delta|<1$ being a random variable of uniform distribution.

$$\begin{aligned}\tau^k &= \left(\frac{\tau_{\max} + \tau_{\min}}{2}\right) + \left(\frac{\tau_{\max} - \tau_{\min}}{2}\right)\delta \\ &= \tau^\circ + \left(\frac{\tau_{\max} - \tau_{\min}}{2}\right)\delta\end{aligned}$$

Robustness of tracking performance under NCS (uncertain delay)

Numerical Example 1: a networked stable & minimum phase system with uncertain (time-varying) delay $0=\tau_{min} \le \tau^k \le \tau_{max} = 0.0312$s



An instance of the actual uncertain varying delay used in simulations

$$\tau^k = 0.0156 + 0.0156\,\delta \quad |\delta| < 1$$

$$\boldsymbol{\tau^k = \tau^o + \delta\, \tau_{unc}, \quad |\delta| < 1}$$

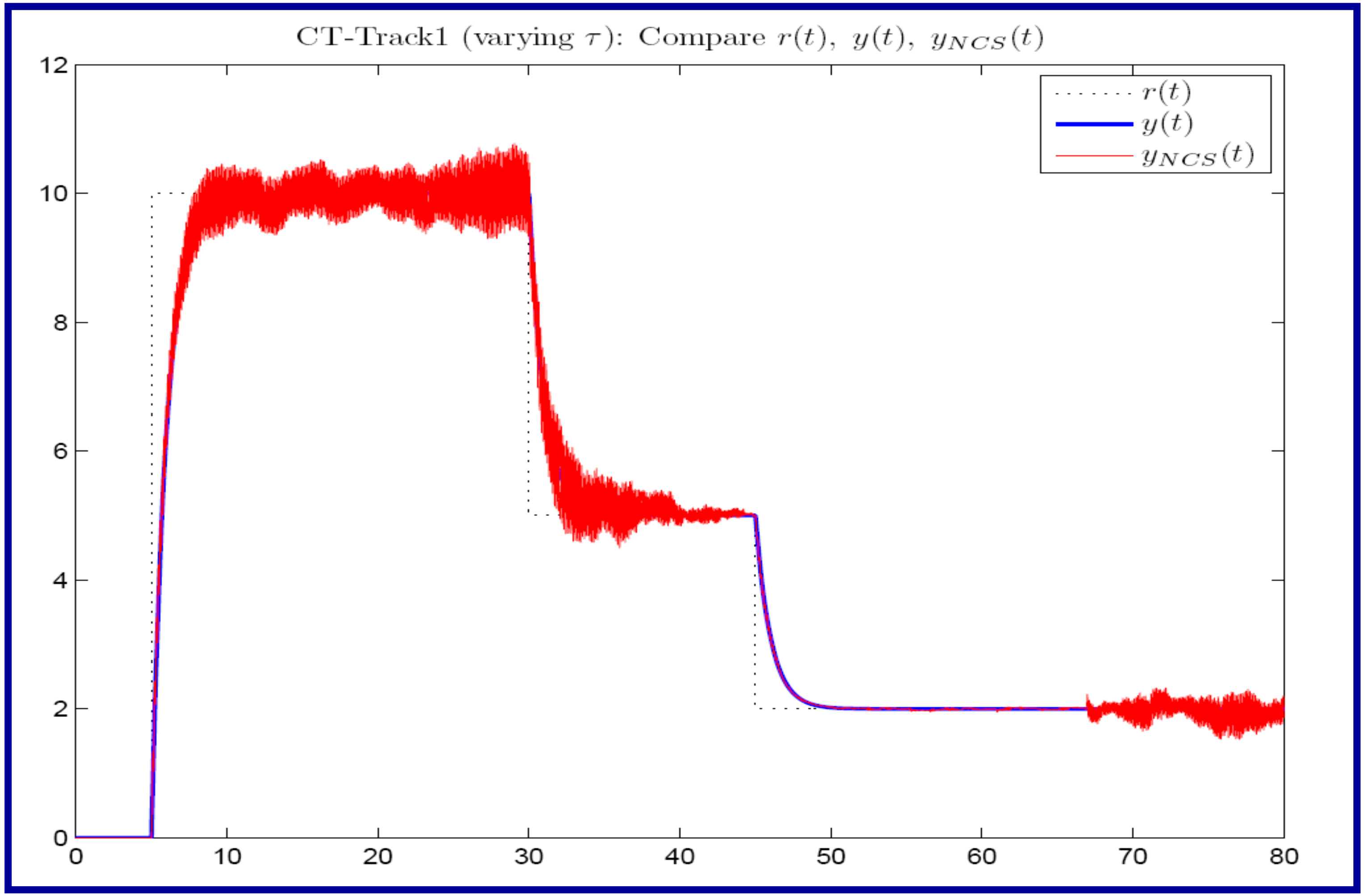
$$\tau^o = \tau_{avg} = (\tau_{max} + \tau_{min})/2$$

$$\tau^k = \left(\frac{\tau_{max}+\tau_{min}}{2}\right) + \left(\frac{\tau_{max}-\tau_{min}}{2}\right)\delta$$

$$= \tau^o + \left(\frac{\tau_{max}-\tau_{min}}{2}\right)\delta$$

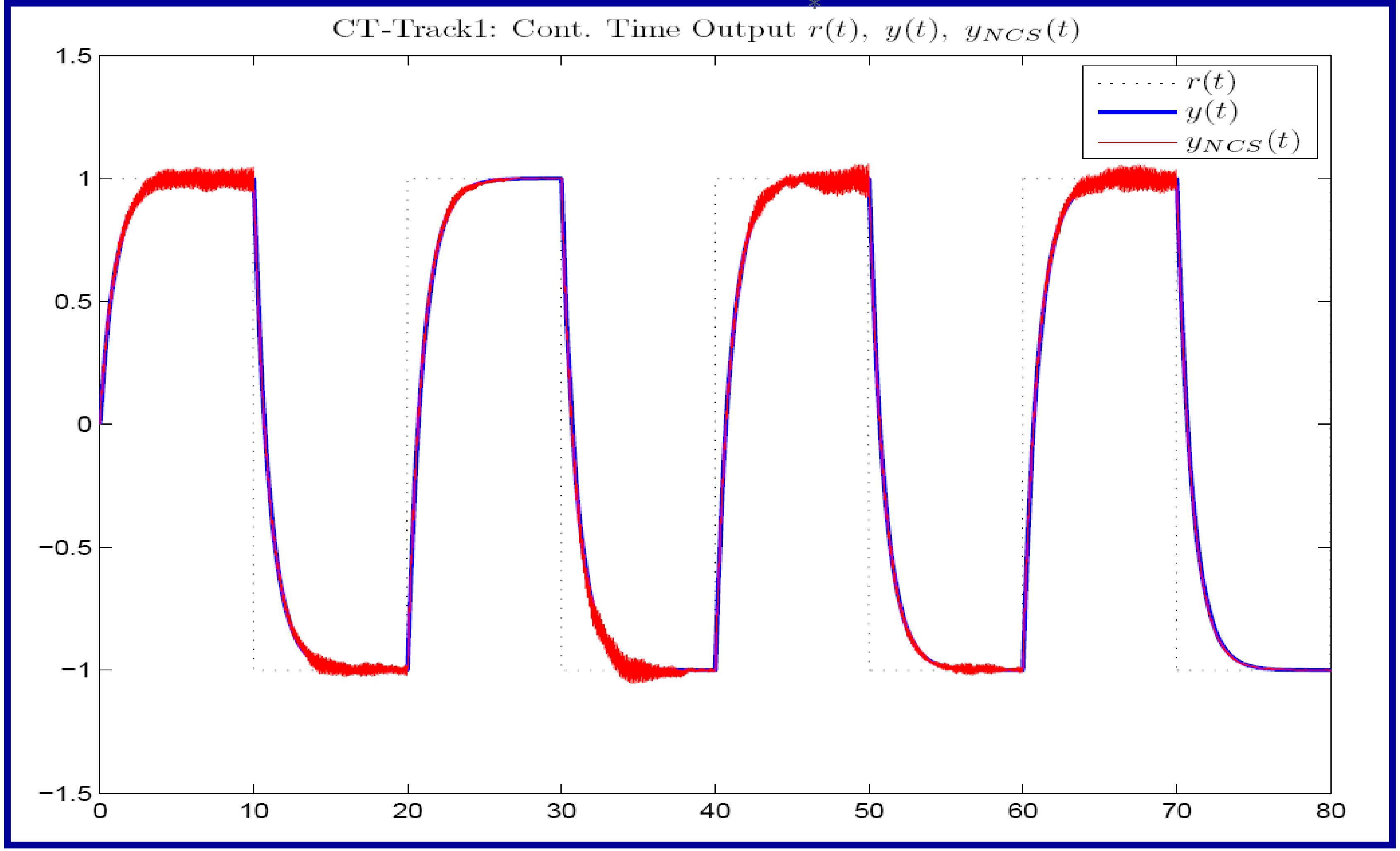
Robustness of tracking performance under NCS (uncertain delay)

Numerical Example 1: a networked stable & minimum phase system with uncertain (time-varying) delay $\tau^k = 0.0156 + 0.0156 * \delta \quad |\delta| < 1$



Robustness of tracking performance under NCS (uncertain delay)

Numerical Example 1: a networked stable & minimum phase system with uncertain (time-varying) delay $\tau^k = 0.0156 + 0.0156 \; \delta^* \quad |\delta| < 1$



Robustness of tracking performance under NCS (unstable system)

the open–loop unstable "benchmark" system $G(s) = \frac{9}{(s-3)^2}$ with state–space description

$$
\begin{aligned}
\dot{x}(t) &= \begin{bmatrix} 0 & 1 \\ -9 & -6 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 9 \end{bmatrix} u(t), \\
y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)
\end{aligned}
$$

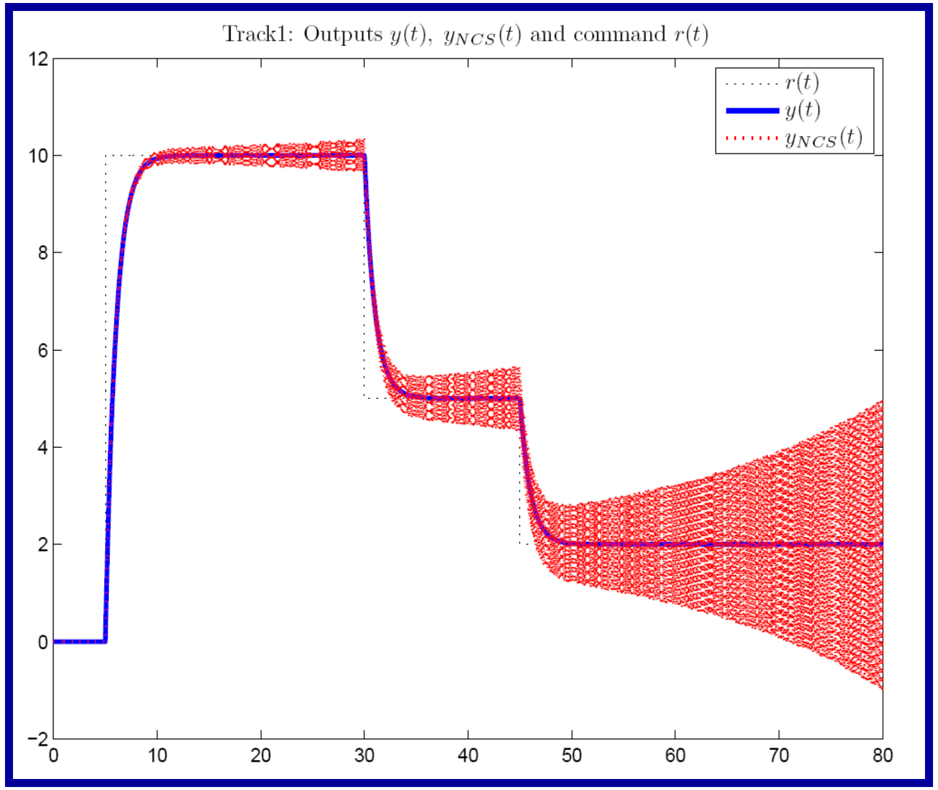
- SPTC was designed via LQR with R=1, Q=100*I_2

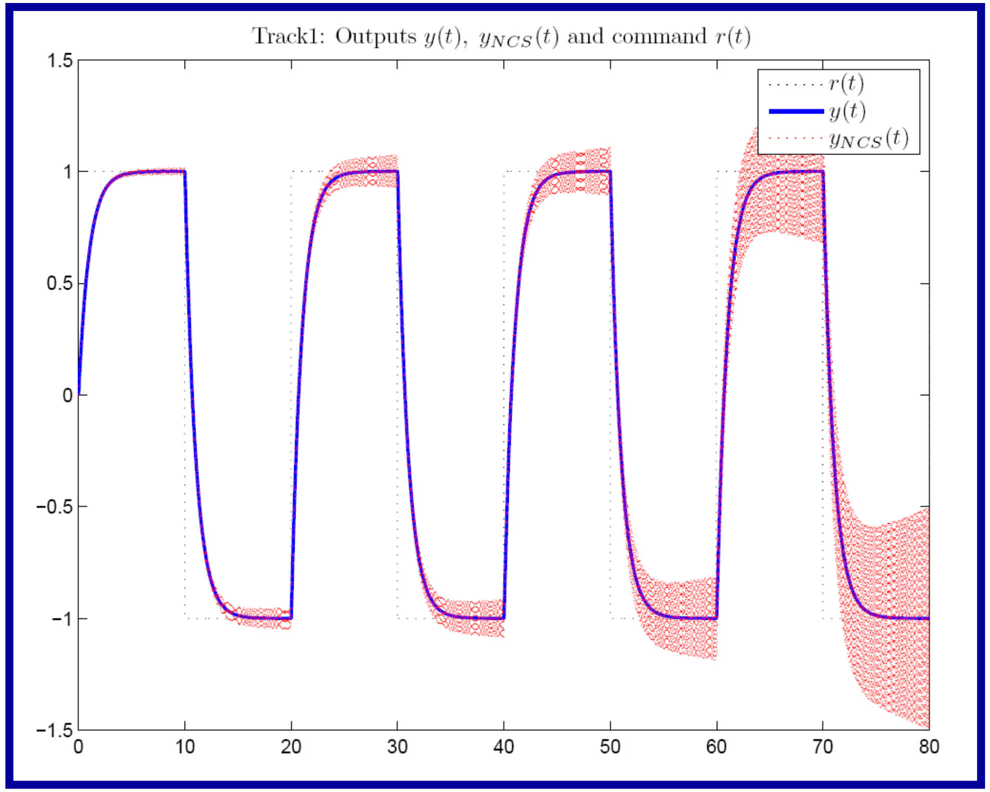
$$u(t) = -9.05x_1(t) - 10.78x_2(t) + 10.05r$$

- gives perfect tracking in the absence of delays
- The Q matrix was selected small in order to avoid high feedback gains...and yet

Robustness of tracking performance under NCS (unstable system)

constant delay: $\tau^k = \tau_{\text{sc}} + \tau_{\text{ca}} = 0.0155$





Robustness of tracking performance under NCS

We can deduce useful conclusions (despite the lack of a mathematically rigorous approach)

- ➢ Clearly a more sophisticated approach is needed for the design of NCS tracking controllers
- ➢ Cannot pretend that the "delays are not there" - must take them into account in the design phase.
- ➢ Cannot compromise stability (avoid large gains) - rule of thumb for Time-Delayed - Systems

Conclusions and Future Work

1. The constant delay case (contrary to intuition) is as detrimental to tracking performance as the varying delay case.

2. The feedback gain must be kept “small”.

- If LQR is employed: extensive trial-and-error simulations with various “Q” matrices must be carried out for the entire delay range to ensure (at least) stability.
- Tracking for the case of unstable plants and/or lightly damped plants is not trivial.

3. Unstable plants: always difficult to enforce tracking (with or without delays).

4. Tracking controllers in discrete-time: special attention is needed due to the interplay between (1) sampling period and delay and (2) the “asynchronous/jump” nature of the control signal

Conclusions and Future Work

- Generalize achieved results for MIMO NCS plants with multiple delays, Parametric Uncertainties & Actuator constraints
- The use of Robust Control Methodologies (H_∞ or "Guaranteed Cost") for the design of Feedback Gain
- The employment of Integral Action (apart from feedback and feedforward terms) in the tracking control Algorithm(s).
- NCS's indeed constitute a very interesting and rich field of control systems both in theoretical results as well as in future applications.