60017 PERFORMANCE ENGINEERING

Controlling a computer system

This lecture

- Input-output models
- Introduction to Control theory

Computer system control

Computer systems are increasingly complex. Similarly to autoregressive models, we can build input-output (I-O) models to bypass this complexity.

I-O models can then be combined with classic control theory:

- The methods we study are frequently used in data centers, networks, protocols, and virtualized systems.
- Better suited to simple control functions, i.e., assign CPU shares, shape network traffic, reduce network congestion, ...

Computer system control

Benefits of classic control theory[1]:

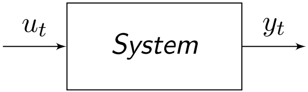
- Relatively easy to learn and use
- Established theory and techniques
- Success stories in computer science (e.g., TCP/IP)

Limitations of classic control theory in computing applications:

- Due to their programming, computer systems can behave in erratic ways not observed before.
- Underpinning maths uses tools, such as transforms, that hinder the intuition.
- More common in automation of physical systems, not all computer scientists know it.

[1] See also: X. Zhu, What does control theory bring to systems research? SIGOPS OSR, Jan 2009.

Discrete-time input-output models



- y_t: output signal at time $t = 1, 2, \ldots$
- u_t: input signal at time $t = 1, 2, \ldots$

Question: If u_t is a control signal, how to model the influence of u_t on y_t without knowing the system internal dynamics?

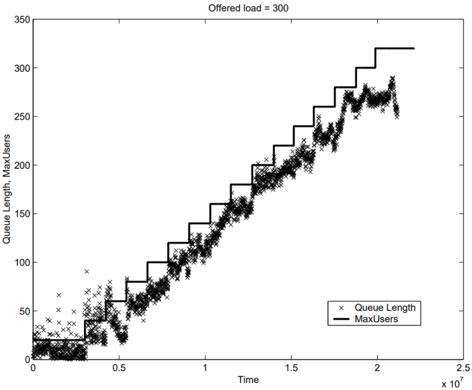
We may then try to capture the system input-output (I-O) behaviour via an autoregressive model, e.g,

$$y_t = \phi_1 y_{t-1} + \theta_1 u_{t-1}$$

Note that, contrary to the general $AR(1)$, we here assume that all signals are deterministic (no noise).

Example: I-O model for an admission control system

We vary the MaxUsers control option (u_{t-1}) in a web server, and observe the resulting queue-length y_t:



The data can be fitted (using linear regression) to the I-O model:

$$y_t = 0.43y_{t-1} + 0.47u_{t-1}$$

Compact representation via z-transforms

We represent a time series y_t, $t \geq 0$, by the (unilateral) z-transform

$$Y(z) = \mathcal{Z}\{y_t\} = y_0 + y_1/z + y_2/z^2 + \ldots = \sum_{k=0}^{+\infty} y_k z^{-k}$$

where $z \in \mathbb{C}$, with the convention that $y_j = 0$ for $j < 0$.

We may imagine $Y(z)$ as a "clothesline" for the y_t values.

Properties

Important properties of the z-transform include:

- Scaling: $\mathcal{Z}\{\alpha y_t\} = \alpha \mathcal{Z}\{y_t\}$
- Linearity: $\mathcal{Z}\{\alpha y_t + \beta u_t\} = \alpha \mathcal{Z}\{y_t\} + \beta \mathcal{Z}\{u_t\}$
- Convolution: $\mathcal{Z}\{y_t\}\mathcal{Z}\{u_t\} = \mathcal{Z}\{\sum_{k=-\infty}^{+\infty} y_k u_{t-k}\}$
- Time shift: $\mathcal{Z}\{y_{t-k}\} = z^{-k}\mathcal{Z}\{y_t\}$

Thanks to the above properties, the z-transform translates complex manipulations of discrete signals into simple algebraic operations, i.e., the formalism avoids the explicit use of infinite series or convolutions that are required in the time domain.

Exercise: prove the time shift property for $k = 1$.

The z-transform for a input-output model

Let us consider again the input-output model

$$y_t = \phi_1 y_{t-1} + \theta_1 u_{t-1}$$

Let $Y(z) = \mathcal{Z}\{y_t\}$ and $U(z) = \mathcal{Z}\{u_t\}$ be the z-transforms of y_t and u_t. It is then possible to prove that

$$Y(z) = H(z)U(z) \quad \text{where} \quad H(z) = \frac{\theta_1}{z - \phi_1}$$

$H(z)$ is called the transfer function and represents the effect of the input signal u_t on the output signal y_t.

Exercise: prove the result by taking the z-transform of both sides of $y_t - \phi_1 y_{t-1} = \theta_1 u_{t-1}$.

More on the transfer function

In more complex I-O models, $H(z)$ is in general a rational function

$$H(z) = \frac{N(z)}{D(z)}$$

where $N(z)$ is a polynomial with roots called zeros and $D(z)$ is a polynomial with roots called poles.

- The value $H(1)$ gives the steady-state gain, i.e., the long-term ratio between output and input signals.
- The output of a system decays geometrically with its poles λ_k.
  - Let $\lambda = \max_k |\lambda_k|$ be the spectral radius. If $\lambda > 1$, the system will be unstable.
  - The closer λ is to zero, the faster will be the system settling time, i.e., the time to adjust to a change in the input (typically, within 2% of the steady-state). The settling time may be coarsely estimated as $T = -4/\ln\lambda$.
- Zeros affect the transient response and the blocking of outputs ($y_t = 0$).

Controlling a system

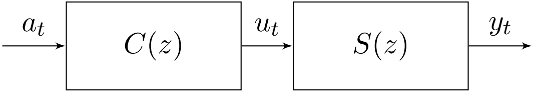
How can we change the poles of a system?

- We can couple the system with a controller so to regulate its input and force the required output.
- The original system can be tested to obtain its transfer function $S(z)$.
- The controller will simply read input signals and send output signals, as determined by its transfer function $C(z)$.

Controller design involves choosing $C(z)$ and a control architecture.

Open-loop control

In this control strategy, the controller regulates the input to the system:



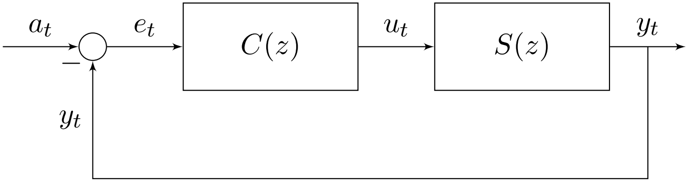
- a_t: input signal at time $t = 1, 2, \ldots$
- u_t: control signal at time $t = 1, 2, \ldots$
- y_t: output signal at time $t = 1, 2, \ldots$

The transfer function for this serial system can be shown to be

$$H(z) = C(z)S(z)$$

Simple to build, but unreliable since the controller cannot monitor the system output.

Closed-loop control



- e_t: error signal at time $t = 1, 2, \ldots$

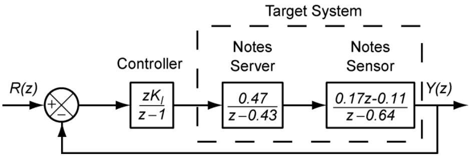
The transfer function for this system can be shown to be

$$H(z) = \frac{C(z)S(z)}{1 + C(z)S(z)}$$

This form of control is common to dampen changes.

More accurate than open-loop control, but can lead to instabilities or oscillations if $C(z)$ is chosen incorrectly.

Case study[2]



The objective is to maintain a reference queue length encoded by $R(z)$. The forward path has transfer function

$$C(z)A(z)B(z) = K_I \frac{z}{z-1} \cdot \frac{0.47}{z-0.43} \cdot \frac{0.17z-0.11}{z-0.64}$$

Accounting for the closed loop, the system has transfer function

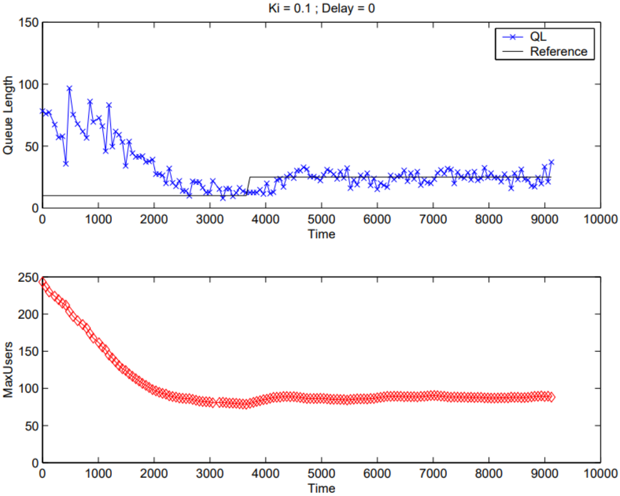
$$H(z) = \frac{C(z)A(z)B(z)}{1 + C(z)A(z)B(z)}$$

we can assign the parameter K_I to control the poles.

[2] Y. Diao et al. A Control Theory Foundation for Self-Managing Computing Systems, IEEE JSAC, 2005.

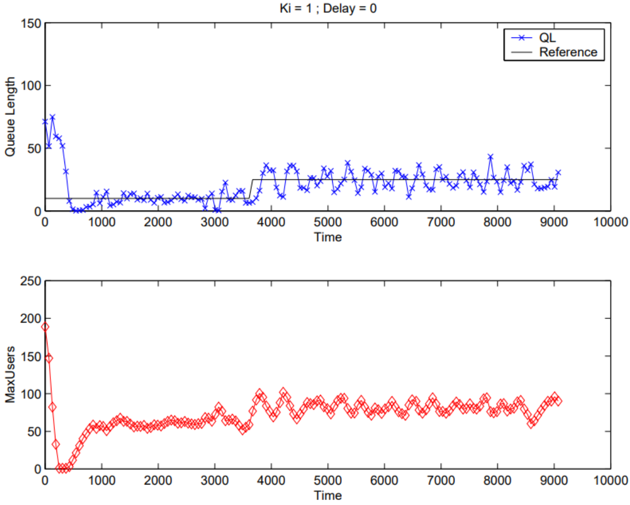
Case study

If we choose $K_I = 0.1$ the controller has spectral radius 0.98 and regulates the system slowly but smoothly (geometrically):



Case study

If we choose $K_I = 1$ the controller settle time is greatly reduced thanks to a pole at 0.84, but there is greater variance:



Case study

If we choose $K_I = 5$ the poles include a complex pair and the controller starts to oscillate: