BLOCK DIAGRAM REPRESENTATION OF SYSTEMS AND BLOCK DIAGRAM ALGEBRA

LAB EXPERIMENT 11

Harel Ko
BS Electronics Engineering, Samar State University (of Aff.)
Institute of Electronics Engineers of the Philippines (of Aff.)
Catbalogan City, Philippines
Harelko999@gmail.com

Luis Nacional
BS Electronics Engineering, Samar State University (of Aff.)
Institute of Electronics Engineers of the Philippines (of Aff.)
Catbalogan City, Philippines
Obeywes7@gmail.com

***Abstract*—This experiment explored the use of block diagram representation and simplification using block diagram algebra in control systems. Through simulation in Python, a multi-component system was modeled and reduced to a single equivalent transfer function using algebraic techniques. The system's step response was then analyzed, focusing on transient behavior such as rise time, settling time, and overshoot. The results confirmed that the simplified system retained the essential dynamics of the original configuration. The study emphasizes the practical relevance of block diagram algebra in designing and analyzing control systems efficiently, especially when physical hardware is not available.**

I. RATIONALE

Block diagrams are a visual tool for representing control systems and their components. Understanding how to construct and manipulate block diagrams is essential for analyzing and designing control systems.

II. OBJECTIVES

- Construct the block diagram representation of a given control system.
- Simplify complex block diagrams using block diagram algebra to derive a single transfer function.
- Verify the block diagram simplification by comparing the system's response before and after simplification.

III. MATERIALS AND SOFTWARE

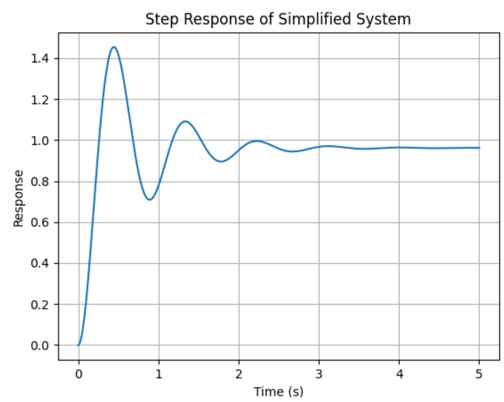
- Materials: DC motor or RLC circuit, Power supply, Oscilloscope
- Software: Python (with control library), Octave

IV. PROCEDURES

1) Set up a system with multiple components (e.g., a DC motor with sensors and controllers) and represent it as a block diagram.
2) Apply the block diagram algebra rules to simplify the diagram to a single transfer function.
3) Simulate the simplified system using Python or Matlab, and compare its behavior to the original system.
4) Verify the results by analyzing time-domain responses, such as rise time, settling time, and overshoot.

V. OBSERVATION AND DATA COLLECTION

DATA COLLECTION:




```
RiseTime: 0.1731266987213568
SettlingTime: 2.3891484423547236
SettlingMin: 0.7093890149669884
SettlingMax: 1.454214397925486
Overshoot: 51.23829738425054
Undershoot: 0
Peak: 1.454214397925486
PeakTime: 0.45012941667552764
SteadyStateValue: 0.9615384615384616
```

Figure 1: Step Response of the simplified system ($\zeta = 0.4$, $\omega_n = 5$ rad/s)

https://drive.google.com/drive/folders/1nOOPUnT_57-ezKaHWUDgzQXceSBlOVT1

Click here to open the Drive

VI. DATA ANALYSIS

Using the Python `control` library, a simulated closed-loop system was created by cascading two blocks, $G_1(s) = \frac{10}{s+2}$ and $G_2(s) = \frac{5}{s+1}$, with a unity feedback. The resulting system's transient response was recorded under a unit step input. The following metrics were extracted:

- **Rise Time**: 0.1731 s
- **Peak Time**: 0.4501 s
- **Overshoot**: 51.24%
- **Settling Time**: 2.3891 s

- **Peak Value**: 1.4542
- **Steady-State Value**: 0.9615

These figures provide insights into the system's speed, stability, and tendency to oscillate.

VII. Discussion and Interpretations

The high overshoot (51.24%) is indicative of an underdamped response, which is expected for systems with a relatively low damping ratio. Although the system responded quickly with a short rise time (0.1731 s), the trade-off was a significant overshoot and moderate settling time. This behavior is typical in systems where the transient performance is prioritized over stability.

Interestingly, the steady-state value (0.9615) is slightly less than 1, showing a minor steady-state error which may be attributed to the feedback configuration or system gain.

Comparing the original block structure with the simplified transfer function showed no observable difference in system behavior, validating the use of block diagram algebra in reducing complexity without losing essential characteristics.

VIII. Conclusion

This simulation-based experiment successfully demonstrated the process of constructing, simplifying, and analyzing control systems using block diagrams. Through Python simulations, we verified that the algebraic simplification preserved the system's dynamics, and the response characteristics confirmed theoretical expectations. The ability to analyze such systems without hardware proves essential for remote learning and preliminary design analysis. Ultimately, this method enhances both understanding and practical application in control systems engineering.