Module 1 Summary: Software Engineering

Focused on abstraction and modularity in software engineering.

Topics: procedures, data structures, objects, state machines

Lab Exercises: implementing robot controllers as state machines

Abstraction and Modularity: Combinators
- Cascade: make new SM by cascading two SM's
- Parallel: make new SM by running two SM's in parallel
- Select: combine two inputs to get one output

Themes: PCAP
- Primitives – Combination – Abstraction – Patterns

Module 2 Preview: Signals and Systems

Focus next on analysis of feedback and control systems.

Topics: difference equations, system functions, controllers.

Lab exercises: robotic steering

themes: modeling complex systems, analyzing behaviors

Analyzing (and Predicting) Behavior

Today we will start to develop tools to analyze and predict behavior.

Example (design Lab 2): use sonar sensors (i.e., currentDistance) to move robot desiredDistance from wall.

Analyzing (and Predicting) Behavior

Make the forward velocity proportional to the desired displacement.

Example: class wallFinder(sm.SM):
... startState = None
... def getNextValues(self, state, inp):
...     desiredDistance = 0.5
...     currentDistance = inp.sonars[3]
...     return (state,io.Action(fvel=?,rvel=0))

Find an expression for fvel.
Check Yourself

Which expression for \texttt{fvel} has the correct form?

1. \texttt{currentDistance}  
2. \texttt{currentDistance-desiredDistance}  
3. \texttt{desiredDistance}  
4. \texttt{currentDistance/desiredDistance}  
5. none of the above

Check Yourself

Which plot best represents \texttt{currentDistance}?

Example: Mass and Spring

\[
x(t) \rightarrow \text{mass \& spring system} \rightarrow y(t)
\]

Example: Tanks

\[
r_0(t) \rightarrow \text{tank system} \rightarrow r_2(t)
\]

Example: Cell Phone System

\[
sound \text{ in} \rightarrow \text{cell phone system} \rightarrow sound \text{ out}
\]

Signals and Systems: Widely Applicable

The Signals and Systems approach has broad application: electrical, mechanical, optical, acoustic, biological, financial, ...
6.01: Introduction to EECS I

Lecture 3
February 15, 2011

Signals and Systems: Modular
The representation does not depend upon the physical substrate.

The representation does not depend upon the physical substrate.

© FreeFoto.com. CC by-nc-nd.
This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/fairuse.

sound in

sound out

cell phone

E/M
tower
optic fiber
tower
E/M
cell phone

sound in

sound out

Composite system

Component and composite systems have the same form, and are analyzed with same methods.

The Signals and Systems Abstraction
Our goal is to develop representations for systems that facilitate analysis.

signal in

system

signal out

Examples:
- Does the output signal overshoot? If so, how much?
- How long does it take for the output signal to reach its final value?

Continuous and Discrete Time
Inputs and outputs of systems can be functions of continuous time

or discrete time.

We will focus on discrete-time systems.

Difference Equations
Difference equations are an excellent way to represent discrete-time systems.

Example:

\[ y[n] = x[n] - x[n-1] \]

Difference equations can be applied to any discrete-time system; they are mathematically precise and compact.

We will use the unit sample as a “primitive” (building-block signal) to construct more complex signals.

Difference Equations
Difference equations are mathematically precise and compact.

Example:

\[ y[n] = x[n] - x[n-1] \]

Let \( x[n] \) equal the “unit sample” signal \( \delta[n] \),

\[ \delta[n] = \begin{cases} 
1, & \text{if } n = 0; \\
0, & \text{otherwise.} 
\end{cases} \]

\[ x[n] = \delta[n] \]

We will use the unit sample as a “primitive” (building-block signal) to construct more complex signals.
Step-By-Step Solutions

Difference equations are convenient for step-by-step analysis.

Find $y[n]$ given $x[n] = \delta[n]$:

- $y[-1] = x[-1] - x[-2]$ = 0 - 0 = 0
- $y[0] = x[0] - x[-1]$ = 1 - 0 = 1
- $y[1] = x[1] - x[0]$ = 0 - 1 = -1

\[
\begin{align*}
   x[n] &= \delta[n] \\
   \ldots &
\end{align*}
\]

\[
\begin{align*}
   y[n] &= x[n] - x[n-1] \\
   \ldots &
\end{align*}
\]

Multiple Representations of Discrete-Time Systems

Block diagrams are useful alternative representations that highlight visual/graphical patterns.

Difference equation:

- $y[n] = x[n] - x[n-1]$

Block diagram:

- [Block diagram image]

Same input-output behavior, different strengths/weaknesses:
- **difference equations** are mathematically compact
- **block diagrams** illustrate signal flow paths

Check Yourself

DT systems can be described by difference equations and/or block diagrams.

Difference equation:

- $y[n] = x[n] - x[n-1]$

Block diagram:

- [Block diagram image]

In what ways are these representations different?

From Samples to Signals

Lumping all of the (possibly infinite) samples into a **single object** – the signal – simplifies its manipulation.

This lumping is analogous to:
- representing coordinates in three-space as points
- representing lists of numbers as vectors in linear algebra
- creating an object in Python

Operators manipulate signals rather than individual samples.

- [Block diagram image]

Nodes represent whole signals (e.g., $X$ and $Y$).

The boxes **operate** on those signals:
- Delay = shift whole signal to right 1 time step
- Add = sum two signals
- * = multiply by $-1$

**Signals** are the primitives.

Operators are the means of combination.
Operator Notation
Symbols can now compactly represent diagrams.
Let $\mathcal{R}$ represent the right-shift operator:
$$ Y = \mathcal{R} \{ X \} \equiv \mathcal{R}X $$
where $X$ represents the whole input signal ($x[n]$ for all $n$) and $Y$ represents the whole output signal ($y[n]$ for all $n$)

Representing the difference machine $\text{Delay}^{-1} + X \ Y$ with $\mathcal{R}$ leads to the equivalent representation
$$ Y = X - \mathcal{R}X = (1 - \mathcal{R})X $$

Operator Notation: Check Yourself
Let $Y = \mathcal{R}X$. Which of the following is/are true:
1. $y[n] = x[n]$ for all $n$
2. $y[n + 1] = x[n]$ for all $n$
3. $y[n] = x[n + 1]$ for all $n$
4. $y[n - 1] = x[n]$ for all $n$
5. none of the above

Operator Representation of a Cascaded System
System operations have simple operator representations.
Cascade systems $\rightarrow$ multiply operator expressions.

Using operator notation:
$$ Y_1 = (1 - \mathcal{R})X $$
$$ Y_2 = (1 - \mathcal{R})Y_1 $$
Substituting for $Y_1$:
$$ Y_2 = (1 - \mathcal{R})(1 - \mathcal{R})X $$

Operator Algebra
Operator expressions expand and reduce like polynomials.

Using difference equations:
$$ y_2[n] = y_1[n] - y_1[n - 1] $$
$$ = (x[n] - x[n - 1]) - (x[n - 1] - x[n - 2]) $$
$$ = x[n] - 2x[n - 1] + x[n - 2] $$

Using operator notation:
$$ Y_2 = (1 - \mathcal{R})Y_1 = (1 - \mathcal{R})(1 - \mathcal{R})X $$
$$ = (1 - \mathcal{R})^2X $$
$$ = (1 - 2\mathcal{R} + \mathcal{R}^2)X $$

Operator Approach
Applies your existing expertise with polynomials to understand block diagrams, and thereby understand systems.

Operator Algebra
Operator notation facilitates seeing relations among systems.
"Equivalent" block diagrams (assuming both initially at rest):

Equivalent operator expression:
$$ (1 - \mathcal{R})(1 - \mathcal{R}) = 1 - 2\mathcal{R} + \mathcal{R}^2 $$
Operator Algebra

Operator notation prescribes operations on signals, not samples: e.g., start with $X$, subtract 2 times a right-shifted version of $X$, and add a double-right-shifted version of $X!$

\[
X : \begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6
\end{array}
\]

\[-2RX : \begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6
\end{array}
\]

\[+R^2X : \begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6
\end{array}
\]

\[y = X - 2RX + R^2X : \begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6
\end{array}
\]

Expressions involving $R$ obey many familiar laws of algebra, e.g., commutativity.

\[R(1 - R)X = (1 - R)RX\]

This is easily proved by the definition of $R$, and it implies that cascaded systems commute (assuming initial rest)

\[
X \quad \text{Delay} \quad \oplus \quad Y
\]

is equivalent to

\[
X \quad \text{Delay} \quad \oplus \quad Y
\]

Operator Algebra

Multiplication distributes over addition.

Equivalent systems

\[
X \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad -1 \quad Y
\]

\[
X \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad -1 \quad \text{Delay} \quad \oplus \quad Y
\]

Equivalent operator expression:

\[R(1 - R) = R - R^2\]

Operator Algebra

The associative property similarly holds for operator expressions.

Equivalent systems

\[
X \quad \text{Delay} \quad + \quad 2 \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad -1 \quad \text{Delay} \quad + \quad 1 \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad + \quad 2 \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad -1 \quad \text{Delay} \quad -1 \quad \text{Delay} \quad + \quad Y
\]

Equivalent operator expression:

\[(1 - R)R(2 - R) = (1 - R)(R(2 - R))\]

Check Yourself

How many of the following systems are equivalent?

\[
X \quad \text{Delay} \quad + \quad 2 \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad + \quad 4 \quad \text{Delay} \quad + \quad Y
\]

\[
X \quad \text{Delay} \quad 4 \quad \text{Delay} \quad + \quad Y
\]

Explicit and Implicit Rules

Recipes versus constraints.

\[
X \quad \text{Delay} \quad -1 \quad Y = (1 - R)X
\]

Recipe: output signal equals difference between input signal and right-shifted input signal.

\[
X \quad \text{Delay} \quad Y = RY + X
\]

\[
(1 - R)Y = X
\]

Constraints: find the signal $Y$ such that the difference between $Y$ and $RY$ is $X$. But how?
**Example: Accumulator**

Try step-by-step analysis: it always works. Start “at rest.”

$$x[n] \rightarrow \begin{array}{c}
\rightarrow \\
\text{Delay} \\
\rightarrow \\
\rightarrow \\
\text{y}[n] \\
\end{array}$$

Find \(y[n]\) given \(x[n] = \delta[n]\):
- \(y[0] = x[0] + y[-1] = 1 + 0 = 1\)
- \(y[1] = x[1] + y[0] = 0 + 1 = 1\)

Thus \(y[n] = \delta[n]\)

 Persistent response to a transient input!

**Example: Accumulator**

These systems are equivalent in the sense that if each is initially at rest, they will produce identical outputs from the same input.

\[(1 - R) Y_1 = X_1 \quad \iff \quad Y_2 = (1 + R + R^2 + R^3 + \cdots) X_2\]

**Proof:** Assume \(X_2 = X_1\):

\[
Y_2 = (1 + R + R^2 + R^3 + \cdots) X_2
\]

\[
= (1 + R + R^2 + R^3 + \cdots) X_1
\]

\[
= (1 + R + R^2 + R^3 + \cdots) (1 - R) Y_1
\]

\[
= ((1 + R + R^2 + R^3 + \cdots) - (R + R^2 + R^3 + \cdots)) Y_1
\]

\[
= Y_1
\]

It follows that \(Y_2 = Y_1\).

**Example: Accumulator**

The reciprocal of \(1 - R\) can also be evaluated using synthetic division.

\[
1 - R \left[ \begin{array}{c}
1 + R + R^2 + R^3 + \cdots
\end{array} \right]
\]

\[
\frac{1}{1 - R}
\]

\[
\frac{R}{R - R^2}
\]

\[
\frac{R^2}{R^2 - R^3}
\]

\[
\frac{R^3}{R^3 - R^4}
\]

Therefore

\[
\frac{1}{1 - R} = 1 + R + R^2 + R^3 + R^4 + \cdots
\]

**Example: Accumulator**

The response of the accumulator system could also be generated by a system with infinitely many paths from input to output, each with one unit of delay more than the previous.

\[
Y = (1 + R + R^2 + R^3 + \cdots) X
\]

**Example: Accumulator**

The system functional for the accumulator is the reciprocal of a polynomial in \(R\).

\[
X \rightarrow \begin{array}{c}
\rightarrow \\
\text{Delay} \\
\rightarrow \\
\rightarrow \\
\text{y} \\
\end{array}
\]

\[
(1 - R) Y = X
\]

The product \((1 - R) \times (1 + R + R^2 + R^3 + \cdots)\) equals 1.

Therefore the terms \((1 - R)\) and \((1 + R + R^2 + R^3 + \cdots)\) are reciprocals.

Thus we can write

\[
\frac{Y}{X} = \frac{1}{1 - R} = 1 + R + R^2 + R^3 + R^4 + \cdots
\]

**Check Yourself**

A system is described by the following operator expression:

\[
\frac{Y}{X} = \frac{1}{1 + 2R}
\]

Determine the output of the system when the input is a unit sample.
Linear Difference Equations with Constant Coefficients

Any system composed of adders, gains, and delays can be represented by a difference equation.

\[ y[n] + a_1 y[n-1] + a_2 y[n-2] + a_3 y[n-3] + \cdots = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] + b_3 x[n-3] + \cdots \]

Such a system can also be represented by an operator expression.

\[ (1 + a_1 R + a_2 R^2 + a_3 R^3 + \cdots) Y = (b_0 + b_1 R + b_2 R^2 + b_3 R^3 + \cdots) X \]

We will see that this correspondence provides insight into behavior. This correspondence also reduces algebraic tedium.

Check Yourself

Determine the difference equation that relates \( x[\cdot] \) and \( y[\cdot] \).

1. \( y[n] = x[n - 1] + y[n - 1] \)
2. \( y[n] = x[n - 1] + y[n - 2] \)
3. \( y[n] = x[n - 1] + y[n - 1] + y[n - 2] \)
5. none of the above

Signals and Systems

Multiple representations of discrete-time systems.

**Difference equations**: mathematically compact.

\[ y[n] = x[n] - x[n - 1] \]

**Block diagrams**: illustrate signal flow paths.

**Operator representations**: analyze systems as polynomials.

\[ Y = (1 - R) X \]

**Labs**: representing signals in python controlling robots and analyzing their behaviors.