SFWR ENG 3DX4 Summary

Instructor: Dr. Lawford

Course: SFWR ENG 3DX4

*Math objects made using* [*MathType*](http://www.dessci.com/en/products/mathtype/)*; graphs made using* [*Winplot*](http://math.exeter.edu/rparris/winplot.html)*.*

Table of Contents

[Introduction to Systems 2](#_Toc410868835)

[Laplace 2](#_Toc410868836)

[Transfer Functions 3](#_Toc410868837)

[Electrical 3](#_Toc410868838)

[Component stuff 3](#_Toc410868839)

[Mesh Analysis 4](#_Toc410868840)

[Cramer’s Rule 4](#_Toc410868841)

[OP-Amps 4](#_Toc410868842)

[Mechanical 4](#_Toc410868843)

[Translational Systems 4](#_Toc410868844)

[Rotational Systems 5](#_Toc410868845)

[Signals 6](#_Toc410868846)

[Final Value Theorem 7](#_Toc410868847)

[Graph Stuff 8](#_Toc410868848)

[Non-/Linear Systems 9](#_Toc410868849)

[Block Diagrams 9](#_Toc410868850)

[State Space Equations 10](#_Toc410868851)

[Transfer Function -> State Space 10](#_Toc410868852)

Note: the following summaries may be useful:

* [SFWR ENG 2MX3](https://drive.google.com/open?id=0BxW61uJyyN8TTWx5d0gzQW9ZUzQ&authuser=0)
* [ENGINEER 3N03](https://docs.google.com/document/d/117z1qGbrDJJV9bx57CQ4SxEL8Ws8oL27bM7-NgFHNKU/edit)
* [TRON 3TA4](https://drive.google.com/file/d/0BxW61uJyyN8TLTR4UV9fYVdBeEU/view?usp=sharing)

I may review to clarify or correct, but mostly I will omit those things.

# Introduction to Systems

Systems can be represented by **block diagrams** to make it easier to marginalize the different parts of the systems.

**Transducer**: converts any form of energy to electrical signals

# Laplace

Useful for…

Time begins when your signal begins



Initial conditions:

* *c*(0)

**Time domain** (*t*): variables are lower case, e.g. *f* (*t*)

**Frequency domain** (*s*): variables are upper case, e.g. *F* (*s*)

**Transfer function**:

When doing the inverse Laplace, it’s useful to break your fractions up so that you can

**Strictly Stable**: it will eventually get back to the initial position

**Marginally Stable**:

**Unstable**: it will progressively get worse



# Transfer Functions

## Electrical

### Component stuff

**Impedence**:



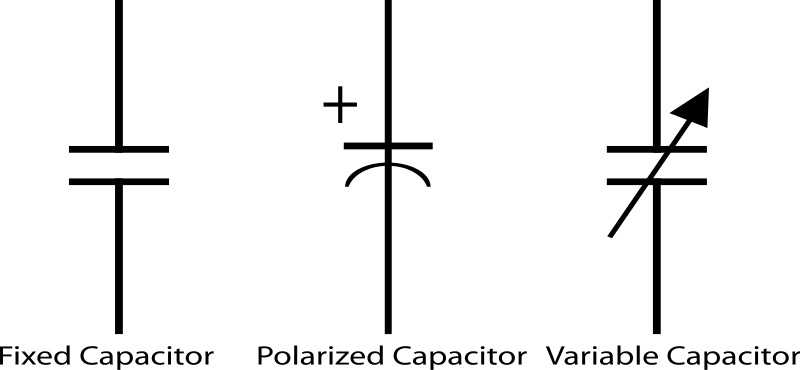
**Current**

****

**Voltage**



Polarized capacitors: Z is positive when current is going from − to +, but negative from + to −

[](https://upload.wikimedia.org/wikipedia/commons/thumb/1/1c/Types_of_capacitor.svg/800px-Types_of_capacitor.svg.png)

**admittance**:







### Mesh Analysis

Add the voltages, where V = IZ

### Cramer’s Rule





### OP-Amps

## Mechanical

**Translational systems**:

**Rotational Systems**:

**Newton’s Second Law of Motion**: Σ *f = Ma*





### Translational Systems

For sure make a free-body diagram

Also, include “inertial force in it” = , which is in the opposite direction of the applied force

#### e.g.











#### Spring

Spring is like a capacitor

**Force displacement**:

#### Viscous Damper

Using viscous fluid to slow something down

Viscous Damper is like a resistor

**Force displacement**:

#### Mass

Mass is like a inductor

**Force displacement**:

### Rotational Systems

**Impedence**: 



* Each θ is on an inertia block. The impedences connected to the motion at θ include the impedences directly to the left and right of the inertia block.
* When finding the sum of impedences between 2 θ’s only count the impedences on wires that don’t go through other θ’s, i.e. 0 if no direct connection





#### Motors

**Meshing Gears**: 

[N]: number of teeth

When gears are lined up

**Applied Armature Voltage** [ea]:

**Armature Resistance** [Ra]:

**Motor Torque Constant** [Kt]:

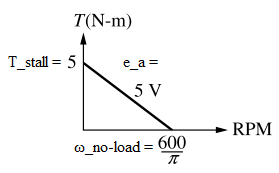
**Back EMF Constant** [Kb]:

**No load speed** [ωno-load]: when the voltage line touches the x-axis



**Stall torque**[Tstall]: when angular velocity reaches 0





[Ja]: motor J

[JL]: load J

[Jm]:g

[Dm]: coefficient of viscous dampening

## Signals

**Transducer**: anything that converts energy to electrical energy

**Transmitter**: long distances

Unstable systems have ∞ steady state error

**Steady-state error** [*e∞*]:



### Final Value Theorem

**Final value theorem**: finds steady state error



So and you’re given F(s), so just multiply by s and find the limit.

There are limitations as to where you can use this theorem. It is dependent on the location of the poles.

#### 1) Right half plane



System is unstable: e+ 🡪 ∞

#### 2) Imaginary Axis − Origin



Unstable: ei 🡪 Oscillatory system, so limit will be average, i.e. midpoint

#### 3) Left Half Plane



Stable: e− converges to 0, but makes transfer function 0 for every single pole

#### 4) Origin



Stable: integrator, i.e. 1/s, so 

Don’t use this theorem if any poles are 1 or 2.

### Graph Stuff

**Rise time** [*Tr*]: time between 10% and 90% of final value (*c*final)

**Peak time** [*Tp*]: time it takes to get to highest peak (*c*max)

**Settling time** [*Ts*]: how long it takes to get to the steady state within ±2%

**Percent overshoot** [%OS]: how much further is the peak from the final



**Time Constant** [τ]: the time it takes the system’s step response to reach 1 – 1/e = 63.2% of cfinal

**Second-order**:







For each pole,



**Forced response**: when a = 0

**Natural response**: when a > 0

**Nonminium-phase system**: Initially the system starts in the wrong direction, then stabilizes at the right place

# Non-/Linear Systems

* Op Amps are linear
* If you don’t have enough voltage, your motor magnets won’t have enough power to switch poles, so they require a minimum voltage

You can’t model non-linear systems, until you linearize it. To do this, we find the slope and approximate the equation of the line, using y=mx+b

**Proportional-Integral-Derivative (PID)**:

If your gears are vibrating, your PID is probably too high

# Block Diagrams

A way of representing a system

**Summing junction**: could be an X or +, but usually an X in this course

**Cascade**: subsystems in series are multiplied

**Parallel**: parallel subsystems have a *summing junction* at the end, so you just add everything together

**Feedback**: positive feedback is bad

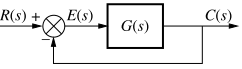


Positive:

Negative: 

Simplification:

**Unity Feedback**: when the feedback path has multiplicative value of 1



# State Space Equations

Yeah, you think you know them from 2MX3, but you don’t really know them. Apparently the ABCD variables actually have names.

* **System Matrix [A]**:
* **Input Matrix [B]**:
* **Output Matrix [C]**:
* **Feedforward Matrix [D]**:

# Transfer Function -> State Space

**Phase Variable Approach**:

The *n* state variables will consist of:

* *y*
* the derivatives of *y*