Centrifuge Rotor Models
A Comparison of the Euler-Lagrange and the Bond Graph Modeling Approach

Jose J. Granda, Ph.D., Professor
Department of Mechanical Engineering
California State University - Sacramento

Jayant Ramakrishnan, Ph.D.

Louis H. Nguyen
Aeroscience & Flight Mechanics Division
NASA Johnson Space Center

AIAA-Houston Annual Technical Symposium 2006
NASA Johnson Space Center - Gilruth Center
May 19, 2006
Topics

- Objectives
- Modeling Approach Comparisons
  - Euler-Lagrange Approach
  - Bond Graph Methods
- Model Structures
  - Simple Oscillator
  - Two Mass System
  - Discussion
- Application
- Summary
Objectives

◆ Demonstrate the similarities and differences in traditional models and bond graph methods
◆ Utilize the bond graph methods for independent verification of models
◆ Develop and promote the bond graph modeling capability here at JSC
Acknowledgements

- We wish to acknowledge the help of Murugan Subramaniam, AkimaTechlink Systems for many helpful discussions on these topics.
Dynamical Systems can be modeled in many ways. Some of the methods are:

- Euler Lagrange Methods where the second order differential equations are derived using free body diagrams and then transformed using appropriate state variables to a system of first order equations. These systems generally have a predefined structure and their solution methods are robust and well documented.

- Bond Graph method which is based on power and causality flow between inertial, resistive and capacitive elements and equations are directly generated in the first order state space. The choice of variables and modeling methodology results in a different structure of matrices. Programs such as CAMP-G can directly generate the equations of motion and the MATLAB M-files from the graphical input of the system structure.
Bond Graphs and Physical Variables

Power Flow Concept

A imposes effort on B, B responds with a flow

Causality Concept

A imposes effort on A, A responds with a flow
### Physical Systems Variable Types

<table>
<thead>
<tr>
<th>Variables</th>
<th>Hydraulic</th>
<th>Mechanical Rotation</th>
<th>Mechanical Translation</th>
<th>Electrical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressure (P)</td>
<td>[N/m²]</td>
<td>Volume flow (Q)</td>
<td>Volume [m³]</td>
<td>Flux linkage [V·s]</td>
</tr>
<tr>
<td>Torque (T)</td>
<td>[N·m]</td>
<td>Angular velocity (ω)</td>
<td>Angle [rad]</td>
<td>Angular Momentum [N·m·s]</td>
</tr>
<tr>
<td>Displacement (x)</td>
<td>[m]</td>
<td>Flow</td>
<td>Displacement (x)</td>
<td>Momentum [N·s]</td>
</tr>
<tr>
<td>Force (F)</td>
<td>[N]</td>
<td>Current (i)</td>
<td>Charge [q]</td>
<td></td>
</tr>
<tr>
<td>Voltage (V)</td>
<td>[Volts V]</td>
<td>Amperes A</td>
<td>[A·s]</td>
<td></td>
</tr>
<tr>
<td>Current (i)</td>
<td>[A·s]</td>
<td>Amperes A</td>
<td>[A]</td>
<td></td>
</tr>
<tr>
<td>Angular velocity (ω)</td>
<td>[rad/s]</td>
<td>Angle [rad]</td>
<td>[rad]</td>
<td></td>
</tr>
<tr>
<td>Displacement (x)</td>
<td>[m]</td>
<td>Flow</td>
<td>[m]</td>
<td></td>
</tr>
<tr>
<td>Velocity (v)</td>
<td>[m/s]</td>
<td>Flow</td>
<td>Displacement (x)</td>
<td></td>
</tr>
</tbody>
</table>
Bond Graphs and Block Diagrams

Block Diagram

A → B
flow (f) effort (e)

A ← B

Bond Graph Notation

A → B
flow (f) effort (e)

A ← B
effort (e) flow (f)

power flow effort
Simple Oscillator Differential Equations
Newton’s Law approach

Second order differential equation

\[ m \ddot{x} = F - kx - b \frac{dx}{dt} \]
\[ m \ddot{x} = b \frac{dx}{dt} + kx = F \]

Let
\[ x_1 = x \]
\[ x_2 = \frac{dx}{dt} \]

Then
\[ \frac{dx_1}{dt} = x_2 \]
\[ \frac{dx_2}{dt} = \frac{1}{m} (F - kx_1 - bx_2) \]

Second order differential equation
Represented as two first order differential equations
State Space Form
Simple Oscillator State Space Form

Defining $u = F$ and the outputs as states, we get in a state space form

\[
\frac{dx_1}{dt} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u
\]

\[
y_1 = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} u
\]

\[
y_2 = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} u
\]
Simple Oscillator Bond Graph Model

Using Inertia, Compliance and resistive elements, the physical system is modeled as shown. Here the I’s and C’s each produce first order differential equations.

In the bond graph model shown, the index for momentum (p) and displacement (q) variables are taken from the bond number in the figure. For example Inertia [ I ] connects to the 1 junction through bond 2.

Power Flow in the Simple Oscillator
Simple Oscillator Differential equations derived from the Bond Graph

State Variables are $q_4$ and $p_2$
$q_4$ = spring deformation
$p_2$ = momentum of mass [1]

\[
\frac{dq_4}{dt} = f_4 = f_3 = f_2 = \frac{1}{I} p_2
\]

\[
\frac{dp_2}{dt} = e_2 = e_1 - e_3 - e_4
\]

Substituting
\[
\frac{dp_2}{dt} = SE - R \frac{dq_4}{dt} - \frac{1}{C} q_4
\]

yielding
\[
\frac{dp_2}{dt} = SE - \frac{R}{I} p_2 - \frac{1}{C} q_4
\]
Simple Oscillator State Space equations from the Bond Graph Model

\[
\begin{bmatrix}
\frac{dq_4}{dt} \\
\frac{dp_2}{dt}
\end{bmatrix} =
\begin{bmatrix}
0 & \frac{1}{I} \\
-\frac{1}{C} & -\frac{R}{I}
\end{bmatrix}
\begin{bmatrix}
q_4 \\
p_2
\end{bmatrix}
+ \begin{bmatrix}
0 \\
1
\end{bmatrix} SE
\]

\[
y =
\begin{bmatrix}
y_1 \\
y_2 \\
qu_1 \\
qu_2
\end{bmatrix} =
\begin{bmatrix}
1 & 0 \\
0 & \frac{1}{m}
\end{bmatrix}
\begin{bmatrix}
q_4 \\
p_2
\end{bmatrix}
+ \begin{bmatrix}
0 \\
0
\end{bmatrix} F
\]
Two Mass Model

\[
\begin{align*}
\frac{dx_1}{dt} &= \frac{-1}{k_1} F_1 + \frac{1}{b_1} V_g \\
\frac{dx_2}{dt} &= \frac{-1}{k_2} F_2 \\
V_g &= \frac{1}{b_2} F_2
\end{align*}
\]
Bond Graph Model

SE₁: F₁

\[ \dot{x}_1 \]

₁

₀

\[ (x_1 - x_2) \]

₁

₀

\[ C₁: 1/k₁ \]

I₁: M₁

₁

₀

R₁: b₁

SE₂: F₂

\[ \dot{x}_2 \]

₁

₀

\[ (x_1 - x_2) \]

₁

₀

\[ C₂: 1/k₂ \]

I₂: M₂

₁

₀

R₂: b₂

SF: Vg

Velocity relations at 1 junctions
Summation of Forces at 0 Junctions
Basic Equations, Power Flow and Causality

\[ SE_1 = -m_1 g \]
\[ I_3 = m_1 \]
\[ C_5 = 1/k_1 \]
\[ R_6 = b_1 \]
\[ I_9 = m_2 \]
\[ SE_8 = -m_2 g \]
\[ C_{12} = 1/k_2 \]
\[ R_{13} = b_2 \]
\[ SF_{15} = v_g \]
\[ e_1 = e_2 + e_3 \]
\[ f_1 = f_3 = f_2 \]

System Description

Power Flow

<table>
<thead>
<tr>
<th>Bond</th>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SE₁</td>
<td>1₁₁₂₃</td>
</tr>
<tr>
<td>2</td>
<td>1₁₁₂₃</td>
<td>0₂₄₇</td>
</tr>
<tr>
<td>3</td>
<td>1₁₁₂₃</td>
<td>I₃</td>
</tr>
</tbody>
</table>

Causality Flow

<table>
<thead>
<tr>
<th>Bond</th>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SE₁</td>
<td>1₁₁₂₃</td>
</tr>
<tr>
<td>2</td>
<td>0₂₄₇</td>
<td>1₁₁₂₃</td>
</tr>
<tr>
<td>3</td>
<td>1₁₁₂₃</td>
<td>I₃</td>
</tr>
</tbody>
</table>
Equations Assembly

Equations at the first O junction are

\[ e_4 = e_5 + e_6 \]
\[ f_4 = f_5 = f_6 \]

Equations at the second 1 junction are

\[ f_2 = f_4 + f_7 \]
\[ e_2 = e_4 = e_7 \]

Continuing to the next junction

\[ e_8 + e_7 = e_9 + e_{10} \]
\[ f_7 = f_8 = f_9 = f_{10} \]
Equation (1)
\[
\begin{align*}
\frac{dq_5}{dt} &= f_5 = f_4 \\
\frac{dq_5}{dt} &= f_2 - f_7 \\
\frac{dq_5}{dt} &= f_3 - f_9 \\
\frac{dq_5}{dt} &= \left(\frac{1}{I_3}\right)p_3 - \left(\frac{1}{I_9}\right)p_9
\end{align*}
\]

Equations (3 and 4)
\[
\begin{align*}
\frac{dp_3}{dt} &= SE_1 - \left(\frac{1}{C}\right)q_5 - \left(\frac{R_6}{I_3}\right)p_3 + \left(\frac{R_6}{I_9}\right)p_9 \\
\frac{dp_9}{dt} &= q_5 \left(\frac{1}{C}\right) + \left(\frac{R_6}{I_3}\right)p_3 - \left(\frac{R_6}{I_9}\right)p_9 + SE_8 - q_{12} \left(\frac{1}{C_{12}}\right) - \left(\frac{p_9}{I_9 - SF_{15}}\right)R_{13}
\end{align*}
\]

Equation (2)
\[
\begin{align*}
\frac{dq_{12}}{dt} &= f_{12} = f_{11} \\
\frac{dq_{12}}{dt} &= f_{10} - f_{14} \\
\frac{dq_{12}}{dt} &= f_9 - f_{15} \\
\frac{dq_{12}}{dt} &= \left(\frac{1}{I_9}\right)p_9 - SF_{15}
\end{align*}
\]
These equations can be converted to the first order state space form using standard approaches.
Bond Graph Modeling – Graphical Tools

Graphical Bond Graph Input

CAMP-G

CAMPGMOD
Controls the simulation, defines parameters, input vectors and initial conditions

CAMPEQU
Generates MATLAB Function containing the state space form of the differential equations, derivatives and the state vectors used to perform the integration

CAMPGSYM
Generates MATLAB Function containing System Symbolic Matrices A,B,C,D Output equations Transfer Functions

OUTPUTS
ISS Centrifuge Rotor

- Simple model depicted as a 5 Degree-of-freedom (10 first order equation) system.

- Vibration isolation system consisting of springs and dampers for the translational motion (3 dof), and rotational springs (2-dof) for the tilting motion of the stator.
ISS Centrifuge Rotor

Bond Graph computer Model
ISS Centrifuge Rotor  
External excitation forces and torques
CAMPGMOD (Sample)
Define Physical Parameters

% CAMP/MATLAB - GENERATED MODEL DESCRIPTION:
% The following files have been generated
% campgmod.m => m file containing model parameters
% initial conditions, sources and simulation controls
% campgequ.m => m function containing the system
% first order differential equations
% campgsym.m => m file containing system matrices in symbolic form
% For simulation and control, edit these files
% Enter values for physical parameters, initial conditions, inputs and time controls
% in places where the ?? marks appear
% Standard generalized variables in Bond Graph notation used.

% ......CAMPGMOD.M - MATLAB MODEL INPUT FILE ......
% clear
% more on
% ......System Physical Parameters........
% global R31 R32 I34 T36x37 R39 R40 I42 R45 R46 I48 T50x51 ...
% T55x56 T60x61 I66 I67 C69 C72 C84 C85 C86

% ...... Initial conditions .......
% Q69IN= 0 ;  % Initial angular displacement about y axis
% Q72IN= 0 ;  % Initial angular displacement about z axis
% Q85IN= 0 ;  % Initial displacement along y axis
% Q86IN= 0 ;  % Initial displacement along z axis
% Q84IN= 0 ;  % Initial displacement along x axis
% P66IN= 0 ;  % Initial angular momentum about the y axis
% P67IN= 0 ;  % Initial angular momentum about the z axis
% P42IN= 0 ;  % Initial linear momentum along the y axis
% P48IN= 0 ;  % Initial linear momentum along the z axis
% P34IN= 0 ;  % Initial linear momentum along the x axis
% initial = [Q69IN; Q72IN; Q85IN; Q86IN; Q84IN; P66IN; P67IN; ...
% P42IN; P48IN; P34IN] ;
% ......System Physical Parameters........
% global R31 R32 I34 T36x37 R39 R40 I42 R45 R46 I48 T50x51 ...
% T55x56 T60x61 I66 I67 C69 C72 C84 C85 C86
% Position parameters. Two angles phi_y, phi_z

% Position parameters. Two angles phi_y, phi_z
CAMPGEQU (Example) Solution of linear or nonlinear differential equations

- % Forcing Function for the 5 DOF Case
- % Modeled after CR Simulation Report - Murugan - Equations  _------
- %
- % Initialization
- %
- % These are TEST numbers only - For actual model use the w profile from the
- % spin up and calculate wdot from the same model; Also update values for
- % alpha, gamma and epsilon; Beta is as selected in the report.
- %
- beta= 45*2*pi/360; % Radians
- %w=0.1; % rad/s
- %wdot= 0.01; % rad/(s)(s)
- Jxy = 1; % Inertia (TBU)
- Jxz = 1; % Inertia (TBU)
- Ip = 1; % Inertia (TBU)
- It = 0.1; % Inertia (TBU)
- gamma = 30*2*pi/360;
- alpha = 1;
- M = 1404; % Mass Rotor-Stator
- e= 0.01; % Epsilon Parameter
- %
- % Force and Torque Computation
- %
- % Inside the logical if loop to account for change
- %
- % Apply Forces and Torques to the 5 DOF CR Model
- %
- if t <= 0
- w=0;
- SE75= 0 ; % newtons
- SE87= 0 ; %newtons
- SE89= 0 ;
- SE90= 0 ;
- SE91= 0 ;
- elseif t >= 0 & t <= 40
- w=(0.7/40*t)*2*pi; % (radians/sec)
- wdot=.7/40;
- Fx = 0; % Force Units
- Fy = M*e*(w^2*cos(beta+w*t)+wdot*sin(beta+w*t)); % Force Units
- Fz = M*e*(w^2*cos(beta+w*t)+wdot*sin(beta+w*t)); % Force Units
- Tx = 0; % Torque units
- Ty = -Jxy*wdot+(Ip-It)*alpha*w^2*cos(gamma);
- Tz = -Jxz*wdot+(Ip-It)*alpha*w^2*sin(gamma);
- SE75= Fy ; % newtons
- SE87= Fz ; %newtons
- SE89= Tz ;
- SE90= Ty ;
- SE91= Fx ;
- else
- w=(0.7)*2*pi; % (radians/sec) constant
- wdot=0 ;
- Fx = 0; % Force Units
- Fy = M*e*(w^2*cos(beta*w*t)+wdot*sin(beta*w*t)); % Force Units
- Fz = M*e*(w^2*cos(beta*w*t)+wdot*sin(beta*w*t)); % Force Units
- Tx = 0; % Torque units
- Ty = -Jxy*wdot+(Ip-It)*alpha*w^2*cos(gamma);
- Tz = -Jxz*wdot+(Ip-It)*alpha*w^2*sin(gamma);
- SE75= Fy ; % newtons
- SE87= Fz ; %newtons
- SE89= Tz ;
- SE90= Ty ;
- SE91= Fx ;
- end
% System Differential Equations-First Order Form
% Derivatives vector
% p_qdot = [dQ69; dQ85; dQ86; dP66; dP67; dP42; ...
% dP48; dP34];
% Symbolic expressions:
% Derivatives (dp,dq) and output variables (e,f),
% Generate A, B matrices corresponding to states p's and q's,

% dQ69 = SF68 - P66/I66
% A(1,:) = [sb, sb, sb, sb, sb, sb, sb, sb, sb, sb];
% B(1,:) = [sb, sb, sb, sb, sb, sb, sb, sb, sb, sb];
% D(1,:) = [sb, sb, sb, sb, sb, sb, sb, sb, sb, sb];
% B(1,:) = [0, 0, 0, 0, -1/I66, 0, 0, 0, 0, 0];
% A(1,:) = [0, 0, 0, 0, 0, -1/I66, 0, 0, 0, 0];
% B(1,:) = [0, 0, 0, 0, 1, 0, 0, 0, 0, 0];
% A(1,:) = [0, 0, 0, 0, 0, 0, -1/I66, 0, 0, 0];
% B(1,:) = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
% dP66 = SF29 * R31 * T36 * x37 - P34 * L34 * R31 * T36 * x37 - P66 * I66 * T36 * x37 * R31 * T36 * x37 - ...
% P67 * I67 * T50 * x51 * R31 * T36 * x37 + SF29 * R32 * T36 * x37 - P34 * I34 * R32 * T36 * x37 - ...
% P66 * I66 * T36 * x37 * R32 * T36 * x37 - P67 * I67 * T50 * x51 + Q66 * C69 + SE89;
% A(2,:) = [1/C69, 0, 0, +1/C86 * T60 * x61, +1/C84 * T36 * x37, -1/C84 * T36 * x37 * R31 * T36 * x37 - ...
% 1/I66 * T36 * x37 * R32 * T36 * x37 - 1/I66 * T60 * x61 * R45 / T60 * x61 - ...
% 1/I67 * T50 * x51 * R31 * T36 * x37 - 1/I67 * T50 * x51 * R32 * T36 * x37, 0 - ...
% 1/I48 * R45 / T60 * x61 - ...
% 1/I48 * R45 / T60 * x61 - 1/I34 * R31 * T36 * x37 - 1/I34 * R32 * T36 * x37 - ...
% B(2,:) = [1/R31 * T36 * x37 + 1/R32 * T36 * x37, 0 + 1/R45 / T60 * x61 + 1/R46 / T60 * x61, ...
% 0, 0, 0, +1, 0];
% dP67 = SF29 * R31 * T50 * x51 - P34 * I34 * R31 * T36 * x37 + Q66 * C69 + SE89;
% A(3,:) = [1/C72, -1/C85 * T55 * x56, 0 + 1/C84 * T50 * x51 - ...
% 1/C85 * T55 * x56 + Q72 / C72 + SE90;
% B(3,:) = [1/R31 * T50 * x51 + 1/R32 * T50 * x51 - ...
% 1/R39 * T55 * x56 - 1/R40 * T55 * x56, 0];
% dP42 = SF35 * R39 * T55 * x56 + P42 / I42 * R39 * T55 * x56 - 1/I67 * T55 * x56 * R40 / T60 * x61 - ...
% SF35 * R40 * T55 * x56 + P42 / I42 * R40 * T55 * x56 - P67 / I67 * T55 * x56 * R40 / T55 * x56 - ...
% Q85 / C89 * T55 * x56 + Q72 / C72 + SE90;
% B(4,:) = [1/R31 * T50 * x51 + 1/R32 * T50 * x51 - ...
% 1/R39 * T55 * x56 - 1/R40 * T55 * x56, 0];
% dP46 = SF29 * R31 * T36 * x37 - P34 * I34 * R31 * T36 * x37 - P66 * I66 * T36 * x37 * R31 * T36 * x37 - ...
% P67 / I67 * T50 * x51 * R31 * T36 * x37 + SF29 * R32 * T36 * x37 - P34 * I34 * R32 * T36 * x37 - ...
% P66 * I66 * T36 * x37 * R32 * T36 * x37 + Q64 / C84 * T36 * x37 + ...
% SF43 * R45 / T60 * x61 - P48 / I48 * R45 / T60 * x61 - P66 / I66 * T60 * x61 * R45 / T60 * x61 - ...
% SF43 * R46 / T60 * x61 - P48 / I48 * R46 / T60 * x61 - P66 / I66 * T60 * x61 * R46 / T60 * x61 - ...
% Q66 / C69 * T60 * x61 + Q69 / C69 + SE89;
% A(5,:) = [1/C69, 0, 0, +1/C86 * T60 * x61, +1/C84 * T36 * x37, -1/C84 * T36 * x37 * R31 * T36 * x37 - ...
% 1/I66 * T36 * x37 * R32 * T36 * x37 - 1/I66 * T60 * x61 * R45 / T60 * x61 - ...
% 1/I67 * T50 * x51 * R31 * T36 * x37 - 1/I67 * T50 * x51 * R32 * T36 * x37, 0 - ...
% 1/I48 * R45 / T60 * x61 - ...
% 1/I48 * R45 / T60 * x61 - 1/I34 * R31 * T36 * x37 - 1/I34 * R32 * T36 * x37 - ...
% B(5,:) = [1/R31 * T36 * x37 + 1/R32 * T36 * x37, 0 + 1/R45 / T60 * x61 + 1/R46 / T60 * x61, ...
% 0, 0, 0, +1, 0];
% dP67 = SF29 * R31 * T50 * x51 - P34 * I34 * R31 * T36 * x37 + Q66 * C69 + SE89;
% A(6,:) = [1/C72, -1/C85 * T55 * x56, 0 + 1/C84 * T50 * x51 - ...
% 1/C85 * T55 * x56 + Q72 / C72 + SE90;
% B(6,:) = [1/R31 * T50 * x51 + 1/R32 * T50 * x51 - ...
% 1/R39 * T55 * x56 - 1/R40 * T55 * x56, 0];
% dP67 = SF29 * R31 * T50 * x51 - P34 * I34 * R31 * T36 * x37 + Q66 * C69 + SE89;
% A(7,:) = [1/C72, -1/C85 * T55 * x56, 0 + 1/C84 * T50 * x51 - ...
% 1/C85 * T55 * x56 + Q72 / C72 + SE90;
% B(7,:) = [1/R31 * T50 * x51 + 1/R32 * T50 * x51 - ...
% 1/R39 * T55 * x56 - 1/R40 * T55 * x56, 0];
% dP42 = SF35 * R39 * T55 * x56 + P42 / I42 * R39 * T55 * x56 - 1/I67 * T55 * x56 * R40 / T60 * x61 - ...
% SF35 * R40 * T55 * x56 + P42 / I42 * R40 * T55 * x56 - P67 / I67 * T55 * x56 * R40 / T55 * x56 - ...
% Q85 / C89 * T55 * x56 + Q72 / C72 + SE90;
% B(8,:) = [1/R31 * T50 * x51 + 1/R32 * T50 * x51 - ...
% 1/R39 * T55 * x56 - 1/R40 * T55 * x56, 0];
System Matrices from Bond Graph $A(:,1:7)$

### $A(:,1:5)$ (All Rows, First 5 Columns)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$1/C69$</td>
<td>0</td>
<td>0</td>
<td>$1/C86/T60x61$,</td>
<td>1/C84/T36x37</td>
<td>$1/C86/T60x61$,</td>
<td>1/C84/T36x37</td>
</tr>
<tr>
<td>0</td>
<td>$1/C72$,</td>
<td>-1/C85*T55x56,</td>
<td>0</td>
<td>$1/C84/T50x51$</td>
<td>$1/C84/T50x51$</td>
<td>$1/C84/T50x51$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1/C86,</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$1/C84$</td>
</tr>
</tbody>
</table>

### $A(:,6:7)$ All rows, Columns 6 & 7

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1/I66,</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
System Matrices from Bond Graph

\[ A(:,8:10) \]

\[
\begin{bmatrix}
0, & 0, & 0 \\
0, & 0, & 0 \\
-1/I42, & 0, & 0 \\
0, & -1/I48, & 0 \\
0, & 0, & -1/I34 \\
0, & -1/I48*R45/T60x61-1/I48*R46/T60x61, & -1/I34*R31/T36x37-1/I34*R32/T36x37 \\
1/I42*R39*T55x56+1/I42*R40*T55x56, & 0, & -1/I34*R31/T50x51-1/I34*R32/T50x51 \\
-1/I42*R39-1/I42*R40, & 0, & 0 \\
0, & -1/I48*R45-1/I48*R46, & 0 \\
0, & 0, & -1/I34*R31-1/I34*R32
\end{bmatrix}
\]
# System Matrices from Bond Graph

**B(:,1:10)**

### B(:,1:5) All rows, Columns 1-5

- \[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

- \[
\begin{bmatrix}
R31/T36x37+R32/T36x37 & 0 & R45/T60x61+R46/T60x61, \\
0 & 0 & 0 \\
\end{bmatrix}
\]

- \[
\begin{bmatrix}
R31/T50x51+R32/T50x51, & -R39*T55x56-R40*T55x56, \\
0 & 0 & 0 \\
\end{bmatrix}
\]

- \[
\begin{bmatrix}
0 & R39+R40, & 0, & 0, & 0 \\
0 & 0, & R45+R46, & 0, & 0 \\
R31+R32, & 0, & 0, & 0, & 0 \\
\end{bmatrix}
\]
### Eigenvalues Comparison (6th Order State Space)

<table>
<thead>
<tr>
<th>#</th>
<th>Euler/Newton Approach</th>
<th>Bond Graph Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Eigenvalue</strong></td>
<td><strong>ω rad/s</strong></td>
</tr>
<tr>
<td>1</td>
<td>-0.3960 ± 0.7391i</td>
<td>0.8385</td>
</tr>
<tr>
<td>2</td>
<td>-0.2046 ± 0.8273i</td>
<td>0.8522</td>
</tr>
<tr>
<td>3</td>
<td>-0.2089 ± 0.8353i</td>
<td>0.8611</td>
</tr>
</tbody>
</table>

**Locked rotational degrees of freedom**
## Eigenvalues Comparison (10th Order State Space)

<table>
<thead>
<tr>
<th>#</th>
<th>Eigenvalue</th>
<th>ω rad/s</th>
<th>ζ Damping Factor</th>
<th>Eigenvalue</th>
<th>ω rad/s</th>
<th>ζ Damping Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.3960 ± 0.7391i</td>
<td>0.8385</td>
<td>0.4722</td>
<td>-0.3960 ± 0.7391i</td>
<td>0.8385</td>
<td>0.4723</td>
</tr>
<tr>
<td>2</td>
<td>-0.1857 ± 0.8142i</td>
<td>0.8351</td>
<td>0.2223</td>
<td>-0.1838 ± 0.8127i</td>
<td>0.8332</td>
<td>0.2205</td>
</tr>
<tr>
<td>3</td>
<td>-0.1888 ± 0.8217i</td>
<td>0.8431</td>
<td>0.2239</td>
<td>-0.1877 ± 0.8208i</td>
<td>0.8419</td>
<td>0.2229</td>
</tr>
<tr>
<td>4</td>
<td>-0.1642 ± 3.2507i</td>
<td>3.2548</td>
<td>0.0504</td>
<td>-0.1880 ± 3.3196i</td>
<td>3.3249</td>
<td>0.0565</td>
</tr>
<tr>
<td>5</td>
<td>-0.1737 ± 3.2771i</td>
<td>3.2816</td>
<td>0.0529</td>
<td>-0.1885 ± 3.3192i</td>
<td>3.3245</td>
<td>0.0566</td>
</tr>
</tbody>
</table>
Summary

- Bond Graph Method provides an elegant alternative graphical method for modeling dynamical systems.
- Results for complex systems like the Centrifuge Rotor correlate well with other approaches.
- CAMP-G generates MATLAB M-file automatically
  - System Equations
  - System Equations in a symbolic form
  - Transfer Functions
- Easily combine bond graph models with mathematical and analysis capabilities of MATLAB
References


