System simulation with HOPSAN NG

Petter Krus, Robert Braun, Peter Nordin, Karl Pettersson
Department of Management and Engineering
Linköping University
The HOPSAN Development

- FORTRAN
- Pascal
- 1972 C
- 1983 C++
- 1995 Java
- 1995 Python

- DSH
- DSH+
- Bath fp

- 67 CSSL (Continuous System Simulation Language)
- ACSL (Advanced Continuous System Simulation Language)
- AMESIM
- Simulink
- Dymola
- Modelica

- 80
- 85
- 90
- 95
- 00
- 05
- 10

- Mainframe Computers
- Workstations
- PC Computers

- 1972
- 1983
- 1995

- C
- C++
- Java
- Python
Simulation in the Product Lifecycle

- **System level design**
  - System simulation for design optimization and analysis
    - *High Speed Simulation HSS*
  - Human in the loop simulation, *Real time simulation RTS*

- **Subsystem design**
  - System simulation for design *HSS*
  - Hardware in the loop simulation *HWIL, RTS*

- **Prototype testing and evaluation**
  - Dynamic testing using *HWIL, RTS*

- **Operation**
  - Training simulators, *RTS*
  - Embedded simulation models for condition monitoring and control
    - *RTS and faster than real time simulation, FRTS*
  - Mission planning
    - *Faster than real time simulation, FRTS*
Industrial Partners and Applications

Hardware in the loop systems
- Prevas,
- National Instrument

Helicopters
- Cybaero AB

Construction Machines
- Volvo CE

Rock drills
- Atlas Copco
Goals and Objectives

- This project is aimed at development, and demonstration of simulation technologies for real-time simulation (RTS), or faster than real-time simulation (FRTS) of high-fidelity models. This includes methods to utilize multi-core architectures in an efficient way.

- To apply the methodologies on real industrial problems for feedback, evaluation, and technology transfer. Hardware in the loop simulation for next generation of energy efficient wheel loaders. Faster than real times simulation of unmanned helicopter systems, and high fidelity modeling of rock-drill equipment, and deployment of models on multi-core hardware.
Research Areas

- Real-time Simulation (RTS), and Faster than Real Time Simulation (FRTS) Technologies
  - Distributed modeling
  - Parallelization of simulation models for multi-core processors
  - Hardware in the loop simulation
- Model fidelity and management
- Principles for parameterization
Industrial Applications

- Hardware in the Loop Simulation of Heavy Vehicles
- Real Time, and faster than real time simulation of unmanned helicopters
- High speed, high fidelity simulation for optimization of rock drill equipment
The usefulness of simulation

Simulation is going through a rapid transition from simple problem dependent models, to highly complex, multi domain, problem independent models.
Hopsan-NG (Next Generation)

- Bidirectional delay-lines
- Modelica support is under development
- Genuine team work
- Released today!

Friday afternoon = “happy hour”
Example, Valve-line system
Signal flow modelling
Power port modelling

Line (capacitance)  node connection  Valve (resistance)

effort p, f, u  flow q, v, i  effort p, f, u  flow q, v, i  effort p, f, u  flow q, v, i

E.g.
HOPSAN
Dymola
AMESIM
Simulation X
Power port model (HOPSAN)
Block diagram model
Typical domains dynamic system simulation

- Oil hydraulics
- Gas, pneumatics
- Electrical power
- Mechanical
- Allow control systems to be integrated
Pump-motor system
The equations for the pump are:
\[ q_{p1} = D_p q_p \]
\[ q_{p2} = -q_{p1} \]
\[ T_p = \frac{p_1 - p_2}{D_p} + B_p q_p \]

The equations for the volumes (lumped transmission lines)
\[ p_1 = \frac{q_{m1} + q_{p1}}{C_{s1}} \]
\[ p_2 = \frac{q_{m2} + q_{p2}}{C_{s2}} \]

The equations for the motor
\[ q_{m2} = D_m q_m \]
\[ q_{m1} = -q_{m2} \]
\[ T_m = \frac{p_1 - p_2}{D_m} - B_m q_m \]

Mechanical Load
\[ J_m q_m = T_m - T_L \]
The system of equations

\[ p_1 = \frac{D_m q_m - D_p q_p}{C_{s1}} \]

\[ p_2 = \frac{D_p q_p - D_m q_m}{C_{s2}} \]

\[ q_m = -\frac{p_1 + p_2 + D_m T_L + B_m D_m q_m}{D_m J_m} \]

\[ T_p = -\frac{-p_1 + p_2 - B_p D_p q_p}{D_p} \]

\[ q_{m1} = -D_m q_m \]
\[ q_{p2} = -D_p q_p \]
\[ T_m = -\frac{-p_1 + p_2 + B_m D_m q_m}{D_m} \]

\[ q_{p1} = D_p q_p \]
\[ q_{m2} = D_m q_m \]
Differential algebraic equations

\[ F : y, x, t \]

\[ K : y, \frac{d^2 y}{dt^2}, \ldots, \frac{d^n y}{dt^n}, t \geq 0 \]

\[ y_0 + t \left( \frac{1}{2} y_{n+1} + \left( y_n + y_{n+1} \right) \right) \]

Trapezoidal rule
F(y,dy/dt) and y for the transmission
Differential algebraic equations

\[
\frac{d}{dt} = \frac{2}{h} + q^{-1} h
\]

Bilinear transform

\[ qy = y + t \]

Use Newton-Rapson

\[ J_{i,j,n} = \frac{G_i}{y_j} \]
Differential algebraic equations

Symbolic operations to generate executable model

Implicit model

\[ F(y, \frac{dy}{dt}) \]

Differential algebraic (DAE)

Add integrator states

G(y)

Algebraic system

Order reduction etc.

Newton-Raphson

Model with numerical solver

Symbolic operations to generate executable model

Implicit model

\[ F(y, \frac{dy}{dt}) \]

Differential algebraic (DAE)

Bilinear transform

Partial differentiation

J(y)

Model with numerical solver

Algebraic system

G(y)

Newton-Raphson

Order reduction etc.
$G(y,t)$
Jacobian of the lumped time discrete algebraic system

\[
J =
\begin{pmatrix}
hD_p & 0 & 0 & -h & h & 0 & 0 & 0 & 0 \\
0 & h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & h & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -h & 0 & 2C_{s1} & 0 & -h & 0 & 0 & 0 \\
0 & 0 & h & 0 & 2C_{s2} & 0 & h & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & h & 0 & 0 & 2D_m \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & h & 0 \\
0 & 0 & 0 & -h & h & 0 & 0 & hD_m & 2B_m D_m \\
0 & 0 & 0 & -h^2 & h^2 & 0 & 0 & 0 & 2h B_m D_m + 4D_m J_m
\end{pmatrix}
\]
Modelling and Simulation in the Extended Enterprise

- Projects where several companies form partnerships, they might be competitors in other
- Full system simulation necessary, but a minimum of proprietary knowledge should be transferred.
  - Exchange of complete un-compiled model information is not an alternative.
- Co-simulation or exchange of precompiled modules are attractive in this context
Collaborative Modelling and simulation

Subsystem/design team

Subsystem/design team

Subsystem/design team

Subsystem/design team

Subsystem interface

System/system integration group
Requirements on simulation for optimisation in a collaborative context

- Fast simulation. Linear scaling
- No individual has a total overview of the system dynamics.
- Numerical properties must be maintained when subsystems are put together. Robustness is an absolute requirement.
- Non-linearities in one components should not affect performance in other subsystems.
- Deterministic simulation time is preferred for optimisation
Development in modelling and simulation in complex mechanical systems

- Modelling and simulation of most of the Space Shuttle hydraulic system in the 70’s. (HYTRAN).

- Hardware performance up 100 times/10years

- System simulation development has been remarkably slow in progress.
Bilateral Delay Lines
or Transmission Line Modelling (TLM)


- P. B. Johns and M.O'Brien. 'Use of the transmission line modelling (t.l.m) method to solve nonlinear lumped networks.' The Radio Electron and Engineer. 1980.

Distributed Model Structure

- Individual simulation setup
  - Usage of Internet to physically separate the models. Useful in a customer-supplier relation
- Enables interdisciplinary model development and simulation
- Efficient for large systems
1990 Real Time Parallel Simulation of Fluid Power System

Fig. 6. System that was simulated using distributed processing.
Execution of Parallel Model

Fig. 5. Distributed simulation of a system with three parallel processes including one physical.

Fig 7. Program listing of the parallel simulation.
The transmission line

Can represent:
Electrical line, hydraulic line, pneumatic line, spring.
Both capacitance and inductance.
Block diagram of transmission line
Differential algebraic equations

\[ F = \frac{-q_p B_p D_p p_a + p_b}{D_p} + T_p \]

\[ y = \begin{cases} i_k \frac{T_p}{k} \end{cases} \]
Time discrete algebraic system with distributed parameters
Compare: Differential algebraic equations with lumped parameters
Jacobian of the distributed time discrete algebraic system

\[
J = \begin{bmatrix}
  hD_p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & h & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & -Z_c & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & -Z_c & 0 & 1 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 1 & 0 & -Z_c & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -Z_c \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & h & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & hD_m \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2hB_mD_m + 4J_mD_m
\end{bmatrix}
\]
Compare: Jacobian of the lumped time discrete algebraic system

\[
J = \begin{bmatrix}
  hD_p & 0 & 0 & -h & h & 0 & 0 & 0 & 0 \\
  0 & h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & h & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & -h & 0 & 2C_{s1} & 0 & -h & 0 & 0 & 0 \\
  0 & 0 & h & 0 & 2C_{s2} & 0 & h & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & h & 0 & 0 & 2D_m \\
  0 & 0 & 0 & 0 & 0 & 0 & h & 0 & -2D_m \\
  0 & 0 & 0 & -h & h & 0 & 0 & hD_m & 2B_m D_m \\
  0 & 0 & 0 & -h^2 & h^2 & 0 & 0 & 0 & 2h B_m D_m + 4D_m J_m 
\end{bmatrix}
\]
Laminar restrictor with transmission line boundaries

\[ \begin{align*}
q_2 &= \frac{p_1 - p_2}{R_v} \\
q_1 &= -q_2 \\
p_1 &= c_1 + q_1 Z_{c_1} \\
p_2 &= c_2 + q_2 Z_{c_2}
\end{align*} \]

\[ \begin{align*}
q_2 &= \frac{c_1 - c_2}{R_v + Z_{c_1} + Z_{c_2}} \\
q_1 &= -q_2 \\
p_1 &= c_1 + q_1 Z_{c_1} \\
p_2 &= c_2 + q_2 Z_{c_2}
\end{align*} \]
Laminar restrictor with transmission line boundaries using Newton Raphson

$$G = F = \begin{pmatrix} K_c p_1 + K_c p_2 + q_2 \\ q_1 + q_2 \\ p_1 - c_1 - Z_{c1} q_1 \\ p_2 - c_2 - Z_{c2} q_2 \end{pmatrix}$$

$$y = \begin{pmatrix} q_2 \\ q_1 \\ p_1 \\ p_2 \end{pmatrix} \quad J = \begin{pmatrix} 1 & 0 & K_c & K_c \\ 1 & 1 & 0 & 0 \\ 0 & -Z_{c1} & 1 & 0 \\ -Z_{c2} & 0 & 0 & 1 \end{pmatrix}$$
Blockdiagram of laminar restrictor

\[
\frac{1}{R_v + Z_{c_1} + Z_{c_2}}
\]
Blockdiagram of pressure source

Pressure source

$q_1$

$Z_c=0$

$c_1$

$p_0$
Blockdiagram of orifice connected to a line and a pressure source
Quantum physics and simulation

- Limited speed of signal propagation (300 000 km/s)
- The uncertainty principle (limited resolution)
- Planck time 10^{-43} sec
- Planck length 10^{-33} m
- Quantum gravity: Space and time as a discrete event system
The transmission line

\[ q_1 \quad q_2 \]

\[ p_1 \]

\[ p_2 \]

\[ c_1 \]

\[ c_2 \]
The transmission line as a general integrator

\[ y = x_1 + x_2 \]

\[ y_1 = y_2 + h \]

\[ y_2 = y_1 + h \]
Comparison transmission line - trapezoidal rule

\[ p_1 H t \leq p_2 H t - T L + \frac{Z_c q_2}{2} H t - T L + \frac{Z_c q_1}{2} H t - T L \]

\[ p_1 H t \leq p_2 H t - 2 T L + \frac{Z_c q_1}{2} H t - 2 T L + \frac{Z_c q_2}{2} H t - T L \]

Set \( q_2 = 0 \). (blocked outlet)

Compared to the trapezoidal method for integration

\[ y_{h+t} = y_t + \frac{1}{2} h \int_{t}^{t+h}, f \right., h = t \]

where

\[ y = f \int_{t}^{t+h}, t \]

These equations are the same if \( T = h/2 \)
Modelling of capacitance using the trapezoidal method and unit transmission line element.
The transmission line equation can be written:

\[ P_1 \alpha^s T = \tilde{\alpha}^s T P_1 + Z_c \alpha^s T Q_1 + \tilde{\alpha}^s T Q_1 \]

Solving for \( P_1 \) yields:

\[ P_1 = \frac{1 + \tilde{\alpha}^2 s T Z_c}{-1 + \tilde{\alpha}^2 s T} \]

The transfer function \( G \) is defined as

\[ G_{\text{line}} = \frac{P_1}{Q_1 Z_c} = \frac{1 + \tilde{\alpha}^2 s T}{-1 + \tilde{\alpha}^2 s T} \]

In the same way the transfer function for the trapezoidal rule is obtained as

\[ G_{\text{trapez}} = \frac{Y}{F} = \frac{1}{2} \frac{1 + \tilde{\alpha}^h}{-1 + \tilde{\alpha}^h} \]

The exact solution is of course

\[ G_{\text{exact}} = \frac{1}{s} \]
Exact integration, trapez and transmission line

Abs[$G$]

Transmission line

Exact

Trapezoidal
Distributed modelling

- Maintain the physical structure of the system in the model
- Use the finite signal propagation speed to numerical advantage
- *Distributed* parameters, *distributed* solver and allows for *distributed* processing
Subsystem management

System model

Simulation of system between time $T_0$ and $T_1$ with time step $h$

Subsystem 1

Subsystem 2

Subsystem 3

Subsystem n

Request variables for $t+h$ given variables at $t$

Subsystem

Simulation of subsystem between time $t$ and $t+h$ with local time step

Subsubsystem 1

Subsubsystem 2

Subsubsystem n
Component generator (Mathematica)

Electric motor example
Generation of Executable Model using in-line transformation

Symbolic operations to generate executable model:
- Implicit model
  \( F(y, dy/dt) \)
- Differential algebraic (DAE)

Bilinear transform

Model with numerical solver:
- Algebraic system
  \( G(y) \)
- Newton-Raphson
  \( J(y) \)

Partial differentiation
//Iterative solution using Newton-Raphson
for(iter=1;iter<=mNoiter;iter++)
{
    //Motor
    //Differential-algebraic system of equation parts

    //Assemble differential-algebraic equations
    systemEquations[0] = wmr1 + (mTimestep*(-(iel2*mKe) + tormr1 + mTimestep*mTm0*limit(wmr1/mwc,-1.,1.))/(2*mJm + mBm*mTimestep) + delayedPart[1][1];
    systemEquations[1] = thetamr1 - (mTimestep*wmr1)/2. + delayedPart[2][1];
    systemEquations[2] = iel2 + (-uel1 + uel2 + mKe*wmr1)/mRa;
    systemEquations[3] = -cel1 + uel1 + iel2*Zcel1;
    systemEquations[5] = -cmr1 + tormr1 - wmr1*Zcmr1;

    //Calculate the delayed parts
    delayParts1[1] = (-(iel2*mKe*mTimestep) + mTimestep*tormr1 - 2*mJm*wmr1 + mBm*mTimestep*wmr1 + mTimestep*mTm0*limit(wmr1/mwc,-1.,1.))/(2*mJm + mBm*mTimestep);
    delayParts2[1] = (-2*thetamr1 - mTimestep*wmr1)/2.;
    delayParts3[1] = delayParts1[1];
    delayParts4[1] = delayParts2[1];
    delayParts5[1] = delayParts1[1];

    //Solving equation using LU-factorisation
    ludcmp(jacobianMatrix, order); solvlu(jacobianMatrix, systemEquations, deltaStateVar, order);

    for(i=0;i<6;i++)
    {
        stateVar[i] = stateVark[i] - jsyseqnweight[iter - 1] * deltaStateVar[i];
    }

    for(i=0;i<6;i++)
    {
        stateVark[i] = ...
    }

    delayParts1[1] = delayParts1[1];
    delayParts2[1] = delayParts2[1];
    delayParts3[1] = delayParts3[1];
    delayParts4[1] = delayParts4[1];
    delayParts5[1] = delayParts5[1];
Example: Aircraft System

The aircraft attitudes during an S-maneuver.

Angular position and reference position of the rudder actuator.
Figure 5. Simulated mission.
Altitude scale is amplified 20 times for the plotting.
(6000 sec simulated in 105 seconds (normal PC), time step 0.01 sec)
Can be used to evaluate functional and performance characteristics.