

Robust Modelling Using Bi-Lateral Delay Lines for High Speed Simulation of Complex Systems

Petter Krus

Division of Fluid and Mechatronic Systems, Linköping University, SE-581 83 Linköping, Sweden

Abstract: A very effective method for modelling and simulation of large complex dynamic systems is represented by distributed modelling using transmission line elements (or bi-lateral delay lines). This method evolves naturally for calculation of pressures when hydraulic pipelines are modelled with distributed parameters, and it can be used to effectively partition the model to use local solvers for the differential equations in each component or subsystem. It is also applicable to other physical systems, such as mechanical, electrical, gas etc.

One interesting application for distributed solvers using bi-lateral delay lines is in real time simulation, since they are very robust and usually quite large simulation time steps can be used. Modelling for real-time applications puts special requirements on robustness in the numerical methods used. In real-time applications there is no room for decreasing time step in numerically critical stages. Furthermore, if a system is relying on a real-time simulation for its functionality, failure in the numerical properties is unacceptable. It is also in many applications possible to simulate the system faster than real time, which means that high fidelity system simulation can be used to plan ahead in control applications, and for simulation based optimisation.

Since solvers can be embedded in components or subsystems, it is very straightforward to implement parallel processing using the multi-core processors which now is the standard for desk top computers. There is also an increasing need in many situations to resolve the time scale to finer details, where the effect of wave propagation needs to be modelled.

Keywords: System simulation, transmission lines

INTRODUCTION

The interest in real-time simulation (RTS) has increased in recent years. There are several reasons for this; more functionality in products is realized through embedded software, and real-time simulation is needed in hardware-in-the-loop simulation (HWIL), which is a key technology for validation of such control systems. Also, real-time simulation is needed in human in the loop simulation (HIL), which is used both in product design and in training simulators. There are also application in the operation phase of a product, e.g. model based diagnostic systems and in model-predictive control systems. Currently, specialized, highly simplified, and often linear models are used for real-time simulation.

As a result there is increasing demands for real-time in a range of industrial applications. However, since commercial platforms for multi-domain system simulation, use architectures that are designed for off-line simulation, it is difficult if not impossible, to use anything but, highly simplified, often linear models, for real-time simulation. With the rapid development in hardware performance, real-time simulation, should no longer be a problem, but methods and tools for generating real-time simulation of high-fidelity, fully non-linear multi-domain system, is lacking. The large application for off-line simulation models in today's product development has promoted the application of central numerical solution methods. With the advent of real-time simulation models it becomes increasingly obvious that these models will differ greatly from off-line simulation models. Still, it is desirable to keep the two sets of models as close as possible.

In many real-time applications it is an advantage if the system model can be composed by distributed simulations of subsystems on different processors. Analyzing systems of components originating from different vendors may even make distributed simulations an absolute necessity, in order to limit disclosure of information. Partitioning of simulation models for parallel simulation is also useful in order to take advantage of multi-core processor architectures rapidly becoming the norm for desk top computing. High speed simulation also enables, faster than real-time simulation (FRTS), which can be used in e.g. control systems based on prediction of future responses to alternative actions (Anagnostopoulos et.al 2003). Furthermore, it also enables the use of simulation based optimisation also for larger systems.

In the recent decade, system simulation has had something of a breakthrough, where large systems can be modelled with a high degree of fidelity, i.e. Lantto et al.. However, the technologies used in commercial software are decidedly off-line technologies that do not allow for real time simulation, this includes centralized solver as the normal approach to simulation. Although great advances have been made in the development of algorithms and software, this approach suffers from inherently poor scaling. I.e. execution time grows more than linear with system size.

In contrast, distributed modelling, where solvers are embedded in subsystem, and even component models, has almost linear scaling properties. Special considerations are needed, however, to connect the subsystems to each other in a way that maintains stability properties, and do not introduce unwanted numerical effects. Technologies based on

bilateral delay lines, see Auslander 1968, (or transmission line modelling, TLM) have been developed and used for a long time at Linköping University, and has successfully been implemented in the HOPSAN simulation package, which at time of writing is the only simulation package that utilise the technology, within mechanical engineering, and fluid power. It has also been demonstrated for parallel simulation in Krus et al 1990 and subsequently by Burton et al 1994. Although the method has its roots already in the sixties, it has never been widely adopted, probably because its advantages are not evident for small systems, and that wave-propagation is regarded as a marginal phenomena in most areas, and thus not generally well understood.

Simulation using bilateral delay lines is also highly suited to simulate systems where wave propagation is an issue. One particular area is in rock drill equipment, where a high fidelity representation of wave propagation phenomena in both mechanical, and hydraulic parts, is fundamental for describing the functionality of the system.

Using distributed solvers with bi-lateral delay lines as connection elements, gives a physically motivated partitioning of the system. In this way component models can be numerically insulated from each other, which provide highly robust numerical properties. This technique is also useful for high speed simulation of systems, and has been used successfully for simulation based optimization, where the system is simulated a large number of times with different parameter sets. The use of transmission line elements for partitioning of systems is a non-exclusive approach. Conventional simulation techniques can still be used within the subsystems. This means that transmission line elements can be used to connect simulation models developed in different simulation packages. Using distributed solvers also has the advantage that it allows a model to be assembled from precompiled modules. This can be highly valuable in collaborative system design, since it does not require disclosure of the source code, when providing a module to partners.

Differential Algebraic Systems

A general approach to represent a system is to represent it as a differential algebraic system. This also allows for algebraic loops. The simulation language Modelica [6] is a language that is based on this form.

$$F(x, \dot{x}, u, t) = 0 \quad (1)$$

where x is the variable vector, u is an input vector, and t is time.

However, Eq. (1) implies that the system essentially has to be written in state space form, something that may be considered as too limited. Many relationships are usually given in transfer function form, which makes it more natural to allow for higher derivatives. The system can then instead be expressed as

$$F\left(y, \frac{dy}{dt}, \frac{d^2y}{dt^2}, \dots, \frac{d^2y}{dt^2}, t\right) = 0 \quad (2)$$

This also has the advantage that the variable vector is reduced, since y is shorter than x , y contains a subset of the states in x . It should, however, be pointed out that it is only possible to impose strong non-linearities (such as limitations on the state variables) represented in the y vector. Also all variables that are of any interest must be included in the y vector otherwise they will not be computed explicitly. Finally high order differentials should be avoided since the equations becomes numerically ill conditioned if the word length is limited.

In order to solve the dynamic part of the system in a numerically stable way, the trapezoidal rule can be used. Using the trapezoidal rule the time differential is solved as:

$$x(t+h) = x(t) + \frac{1}{2}h(x(t) + x(t+h)) \quad (3)$$

A more effective way of using the trapezoidal rule is to reformulate it in the form known as the bilinear transform.

$$\frac{d}{dt} = \frac{2(1-q^{-1})}{h(1+q^{-1})} \quad (4)$$

where q in this context represents the time displacement operator such that:

$$qy = y(t+h) \quad (5)$$

Using the bilinear transform in Eq. (4) means that it can be rewritten as a function G of y and old states.

$$G(y(t), y(t-h), \dots, y(t-nh), u(t), u(t-h), \dots, u(t-nh), t) = 0 \quad (6)$$

When solving the system all the old values $y(t-h)\dots y(t-nh)$ can be regarded as constants since they have already been established in previous time steps. Likewise the input vector is also known. Equation (6) is therefore rewritten as:

$$G(y(t), t) = 0 \tag{7}$$

In order to solve this system of equations in a numerically stable way, the Jacobian matrix is needed, which is defined as:

$$J_{ijk} = \frac{\partial G_i(y_k(t))}{\partial y_j} \tag{8}$$

The equation can then be solved numerically using Newton-Raphson iteration.

$$y_{k+1} = y_k(t) - J_k(t)^{-1} G(y_k(t)) \tag{9}$$

Since an iterative procedure is used, there is a potential for performance loss due to the number of iterations needed to solve the system. However, the values from the previous time step can be used as start values.

$$\begin{aligned} y_0(t) &= y(t-h) \\ J_0(t) &= J(t-h) \end{aligned} \tag{10}$$

If the system is linear, the system can be solved in only one iteration, and it is usually sufficient with only one iteration even for non-linear systems, especially if a small time step is used. There are, however, situations when input signals changes suddenly, e.g. a valve is changed step wise during one time step, that requires more than one iteration. In practice, however, it has been found that two iterations increase the tolerance against non-linearities dramatically, while a further increase to three iteration gives only minor improvement. Two iterations have therefore been found to be something near to an optimum for almost all situations. For implementation it is better to use LU-decomposition rather than using the matrix inverse of the Jacobian.

Provided the system is reasonably linear (slow variation of J, Eq. (9) is an A-stable method. However, in reality, rather large variations of J can be tolerated. Even pure discontinuities can also be handled satisfactory using the above approach, when fixed-time step is used (as in real-time simulation).

Eq. (9) also illustrates a dilemma associated with all numerically stable methods. They need knowledge of the Jacobian, and if the system is stiff and highly non-linear this must be updated very often. We also realize that the computational burden is much more than linearly dependent on the size of the system. This makes these methods unsuitable for large problems. Eq. (9) is, however, very effective for solving small systems which makes it very suitable for solving subsystems in a distributed modelling context.

Example

As an example the modelling of a simple hydrostatic transmission consisting of a pump and motor with an inertia load is considered.

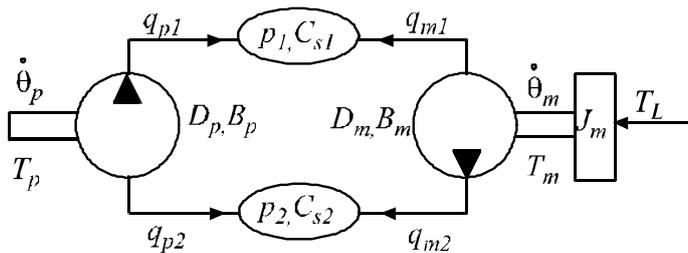


Figure 1. Hydraulic pump-motor example.

The vector F for the system equations becomes

$$F(y, \dot{y}, u, t) = \begin{pmatrix} \dot{\theta}_p B_p + \frac{P_a - P_b}{D_p} T_p \\ -\dot{\theta}_p D_p + q_{p1} \\ \dot{\theta}_p D_p + q_{p2} \\ \dot{p}_1 - \frac{q_{m1} + q_{p1}}{C_{s1}} \\ \dot{p}_2 - \frac{q_{m2} + q_{p2}}{C_{s2}} \\ -\dot{\theta}_m D_m + q_{m1} \\ \dot{\theta}_m D_m + q_{m2} \\ -\dot{\theta}_m B_m + \frac{P_a - P_b}{D_m} T_m \\ \ddot{\theta}_m + \frac{B_m}{J_m} \dot{\theta}_m - \frac{P_a - P_b}{J_m D_m} + \frac{T_L}{J_m} \end{pmatrix} \quad (11)$$

The variable vector y is

$$y = \begin{pmatrix} T_p \\ q_{p1} \\ q_{p2} \\ p_1 \\ p_2 \\ q_{m1} \\ q_{m2} \\ T_m \\ \theta_m \end{pmatrix} \quad (12)$$

The variable vector u is

$$u = \begin{pmatrix} \dot{\theta}_p \\ T_L \end{pmatrix} \quad (13)$$

Using bilinear transform to convert the time continuous differential algebraic system into a discrete time system yields $G(y, t) =$

$$\begin{pmatrix} DS[1, -h p_a + h p_b + D_p (h T_p + 2 B_p \theta_p)] - h p_a + h p_b + h D_p T_p - 2 B_p D_p \theta_p \\ DS[1, h q_{p1} + 2 D_p \theta_p] + h q_{p1} - 2 D_p \theta_p \\ DS[1, h q_{p2} - 2 D_p \theta_p] + h q_{p2} + 2 D_p \theta_p \\ DS[1, -2 C_{s1} p_a - h (q_{m1} + q_{p1})] + 2 C_{s1} p_a - h (q_{m1} + q_{p1}) \\ DS[1, -2 C_{s2} p_b + h (q_{m2} + q_{p2})] + 2 C_{s2} p_b + h (q_{m2} + q_{p2}) \\ DS[1, h q_{m1} - 2 D_m \theta_m] + h q_{m1} + 2 D_m \theta_m \\ DS[1, h q_{m2} + 2 D_m \theta_m] + h q_{m2} - 2 D_m \theta_m \\ DS[1, -h p_a + h p_b + D_m (h T_m - 2 B_m \theta_m)] - h p_a + h p_b + h D_m T_m + 2 B_m D_m \theta_m \\ DS[1, 2 (-h^2 p_a + h^2 p_b + D_m (h^2 T_L - 4 J_m \theta_m))] + DS[2, -h^2 p_a + h^2 p_b + D_m (h^2 T_L - \\ 2 h B_m \theta_m + 4 J_m \theta_m)] - h^2 p_a + h^2 p_b + h^2 D_m T_L + 2 h B_m D_m \theta_m + 4 D_m J_m \theta_m \end{pmatrix} \quad (14)$$

where DS is the delay step function defined as

$$DS(n, x(t)) = x(t - nh) \quad (15)$$

The Jacobian of this system is:

$$J = \begin{pmatrix} h D_p & 0 & 0 & -h & h & 0 & 0 & 0 & 0 \\ 0 & h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -h & 0 & 2 C_{s1} & 0 & -h & 0 & 0 & 0 \\ 0 & 0 & h & 0 & 2 C_{s2} & 0 & h & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & h & 0 & 0 & 2 D_m \\ 0 & 0 & 0 & 0 & 0 & 0 & h & 0 & -2 D_m \\ 0 & 0 & 0 & -h & h & 0 & 0 & h D_m & 2 B_m D_m \\ 0 & 0 & 0 & -h^2 & h^2 & 0 & 0 & 0 & 2 h B_m D_m + 4 D_m J_m \end{pmatrix} \quad (16)$$

Eq. (9) can then be used to solve this system in each time step. Although it is possible to use Eq. (9) directly it is wise to replace the inverse of the Jacobian by using LU-decomposition instead, there are also a few other actions that can be done in order to further enhance the efficiency of the solver. In general, however, the effort to solve the system increases more than linear with the system size.

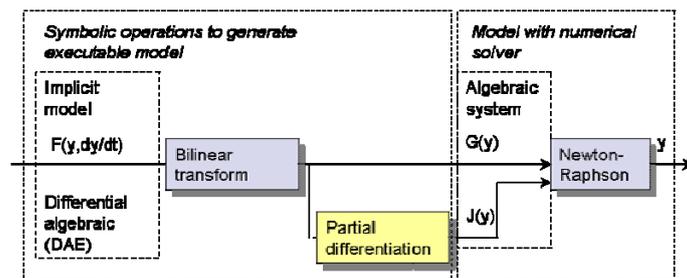


Figure 2. Scheme of transformations

Figure 2. shows a scheme of the transformations involved. The differential algebraic system DAE is transformed into time discrete form using bilinear transform. In Elmqvist et al 1995 the concept of *inline integration* is introduced, where extra equations are introduced to perform the integration. The scheme here is related but would be more appropriately called *inline transformation*, and *inline integration* can be viewed as a special case of that. The Jacobian J is obtained by symbolic partial differentiation of the time discrete system G . G and the Jacobian J are used to solve the system in each time step using the Newton-Raphson method for solving the system in each time step.

Distributed Modelling For Simulation of Fluid Power Components and Systems

Distributed parameters

Simulation of fluid power systems are characterized by difficulties such as very strong nonlinearities, stiff differential equations and a high degree of complexity. Using conventional integration techniques it is often necessary to use very small time steps in order to be able to deal with numerically stiff problems, and strong nonlinearities.

If the state variables in the system are to be unique for each subsystem and not shared by other subsystems, distributed parameters (variables) must be introduced. This can be accomplished if the propagation of waves in connecting components, such pipes, in the system is considered.

A very suitable method for modelling and simulation of large complex dynamic systems is represented by distributed modelling using transmission line elements. The origin of this concept goes back at least to Auslander 1968 [1] who first introduced transmission lines (or bi-lateral delay lines). This method evolves naturally for calculation of pressures when pipelines are modelled with distributed parameters. This approach was adopted for simulation of fluid power systems with long lines in the HYTRAN program already in the seventies.

A related method is the transmission line modelling method (TLM) presented by Johns and O'Brien (Ref. [2]) for simulation of electrical networks.

Johns and O'Brien pointed out that an important aspect of modelling using transmission line elements is that most of the numerical errors introduced by an ordinary solver are avoided. The errors made due to the introduction of transmission line elements, are better described as modelling errors.

An attractive feature with this is that laws of conservation of mass and energy still hold for the solution, since there always exist a plausible physical system for the model, although the line lengths may vary compared to the original system. This also implies that the user may tolerate a larger numerical error since, generally, quite large modelling errors are present anyway (errors of the order of 10% are generally considered acceptable from an engineering point of view).

The Unit Transmission Line Element

In transmission line modelling the basic dynamic element is the unit transmission line. In the HOPSAN package this is used to connect different components to each other. In the general case it can be used to model both capacitances and inductances. In the HOPSAN-package, however, it is used primarily to represent capacitances (oil volumes and mechanical springs).

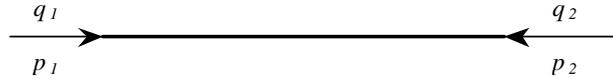


Figure 3. Transmission line

The complete set of equation that describes a lossless transmission line are:

$$\begin{aligned} p_1(t) &= p_2(t-T) + Z_c(q_1(t) + q_2(t-T)) \\ p_2(t) &= p_1(t-T) + Z_c(q_2(t) + q_1(t-T)) \end{aligned} \tag{17}$$

Here Z_c is the characteristic impedance of the line, p and q are pressures and flows respectively. T is the time delay in the line. Note that the main property of these equations is the time delay they introduce in the communication between the ends. Introducing

$$\begin{aligned} c_1(t) &= p_2(t-T) + Z_c q_2(t-T) \\ c_2(t) &= p_1(t-T) + Z_c q_1(t-T) \end{aligned} \tag{18}$$

Here c is the wave variables that represent information that has been transmitted from the other side of the transmission line. With these, the following set of equations is obtained.

$$\begin{aligned} p_1(t) &= c_1(t) + Z_c q_1(t) \\ p_2(t) &= c_2(t) + Z_c q_2(t) \end{aligned} \tag{19}$$

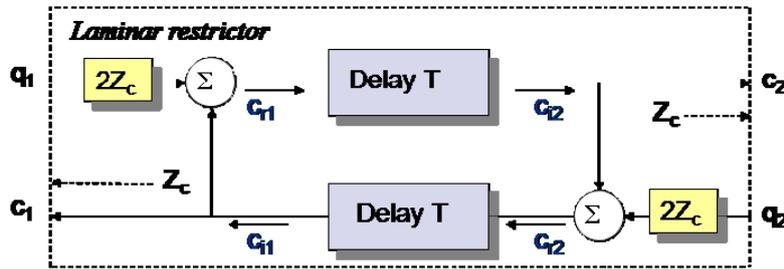


Figure 4. Block diagram of transmission line.

An interesting observation is found if c_2 in Eq. (18) is substituted with Eq. (19) and the outlet at 2 is blocked.

$$p_1(t) = p_1(t-2T) + Z_c(q_1(t-2T) + q_1(t)) \tag{20}$$

Compared to the trapezoidal method for integration

$$y(t+h) = y(t) + \frac{h}{2}(\dot{y}(t) + \dot{y}(t+h)) \tag{21}$$

Where h is the simulation time step. These equations are the same if $T=h/2$!

The relationship between flow entering a volume and the pressure can be written as:

$$\dot{p} = \frac{q}{C} \tag{22}$$

where C is the capacitance. Identification yields

$$Z_c = \frac{h}{C} \tag{23}$$

The implication of this is that the trapezoidal method is used to integrate pressure in a volume (capacitance) between two components; this corresponds to introducing a short pipe instead of a pure capacitance.

The introduction of a transmission line element in place of a capacitance can therefore be viewed as a kind of integration method. In general it can be written such that the integration of an equation as (24) is performed by splitting the variable y into the two variables y_1 and y_2 .

$$\dot{y} = f(x_1 + x_2) \quad (24)$$

These values are the same in steady state, and the difference between them can be regarded (and a measure of) a numerical error. They are calculated using the following equations.

$$\begin{aligned} y_1(t) &= y_2(t-h) = hf(x_1(t) + x_2(t-h)) \\ y_2(t) &= y_1(t-h) = hf(x_2(t) + x_1(t-h)) \end{aligned} \quad (25)$$

y_1 is then of course used at the equations associated with x_1 and y_2 with x_2 .

It should be noted that in order to further improve the numerical properties of the transmission line element damping can be introduced as in ref. [3]. This greatly improves the behaviour with no significant side effects. Since a transmission line also has inductance, an unwanted parasitic inductance will result from using a transmission line to represent a capacitance. The inductance L can be calculated as:

$$L = hZ_c = \frac{h^2}{C} \quad (26)$$

As can be seen it rapidly diminishes with step size. It also means that a transmission line can be used to replace an inductance with a resulting parasitic capacitance as a side effect.

System simulation using transmission line elements

Using this substitution for the pressures p_1 and p_2 in the hydraulic transmission example yields the new system (instead of Eq. (11)) of equations as:

$$F = \begin{pmatrix} \frac{-\dot{\theta}_p B_p D_p - p_a + p_b}{D_p} + T_p \\ q_{p1} - \dot{\theta}_p D_p \\ \dot{\theta}_p D_p + q_{p2} \\ -DS(1, p_{a2}) + p_{a1} - (q_{p1} - DS(1, q_{m1})) Z_c \\ -DS(1, p_{b2}) + p_{b1} - (q_{p2} - DS(1, q_{m2})) Z_c \\ -DS(1, p_{a1}) + p_{a2} - (q_{m1} - DS(1, q_{p1})) Z_c \\ -DS(1, p_{b1}) + p_{b2} - (q_{m2} - DS(1, q_{p2})) Z_c \\ \dot{\theta}_m D_m + q_{m1} \\ q_{m2} - \dot{\theta}_m D_m \\ \frac{\dot{\theta}_m B_m D_m - p_a + p_b}{D_m} + T_m \\ \dot{\theta}_m + \frac{\dot{\theta}_m B_m D_m + T_L D_m - p_a + p_b}{D_m J_m} \end{pmatrix} \quad (27)$$

Here the variable p_1 has been split into the variables p_{1a} and p_{1b} across the transmission line, and the same with p_2 . The variable vector becomes

$$y = \begin{pmatrix} T_p \\ q_{p1} \\ q_{p2} \\ P_{1a} \\ P_{1b} \\ P_{2a} \\ P_{2b} \\ q_{m1} \\ q_{m2} \\ T_m \\ \theta_m \end{pmatrix} \quad (28)$$

After bilinear transformation the system becomes

$$G(y, t) = \begin{pmatrix} DS[1, -h p_a + h p_b + D_p (h T_p + 2 B_p \theta_p)] - h p_a + h p_b + h D_p T_p - 2 B_p D_p \theta_p \\ DS[1, h q_{p1} + 2 D_p \theta_p] + h q_{p1} - 2 D_p \theta_p \\ DS[1, h q_{p2} - 2 D_p \theta_p] + h q_{p2} + 2 D_p \theta_p \\ -DS[1, p_{a2}] + P_{a1} + (DS[1, q_{m1}] - q_{p1}) Z_c \\ -DS[1, P_{b2}] + P_{b1} + (DS[1, q_{m2}] - q_{p2}) Z_c \\ -DS[1, P_{a1}] + P_{a2} + (DS[1, q_{p1}] - q_{m1}) Z_c \\ -DS[1, P_{b1}] + P_{b2} + (DS[1, q_{p2}] - q_{m2}) Z_c \\ DS[1, h q_{m1} - 2 D_m \theta_m] + h q_{m1} + 2 D_m \theta_m \\ DS[1, h q_{m2} + 2 D_m \theta_m] + h q_{m2} - 2 D_m \theta_m \\ DS[1, -h p_a + h p_b + D_m (h T_m - 2 B_m \theta_m)] - h p_a + h p_b + h D_m T_m + 2 B_m D_m \theta_m \\ DS[1, 2(-h^2 p_a + h^2 p_b + D_m (h^2 T_L - 4 J_m \theta_m))] + DS[2, -h^2 p_a + h^2 p_b + \\ D_m (h^2 T_L - 2 h B_m \theta_m + 4 J_m \theta_m)] - h^2 p_a + h^2 p_b + h^2 D_m T_L + \\ 2 h B_m D_m \theta_m + 4 D_m J_m \theta_m \end{pmatrix} \quad (29)$$

The corresponding Jacobian then becomes:

$$J = \begin{pmatrix} h D_p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -Z_c & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -Z_c & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -Z_c & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -Z_c & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & h & 0 & 0 & 2 D_m \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & h & 0 & 0 & -2 D_m \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h D_m & 0 & 2 B_m D_m \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 h B_m D_m + 4 J_m D_m & 0 \end{pmatrix} \quad (30)$$

Note that this system can be partitioned into two uncoupled systems that consequently can be solved independently from each other in each time step. This means that the system can be solved using two instances of Newton-Raphson, equation (9), that are numerically insulated from each other. The price for this is that the Jacobian has become slightly larger with the introduction of two variables for p_a and p_b respectively.

Modelling of Components

In order to demonstrate the principle of component modelling the very simple laminar orifice with the resistance R_v is shown.

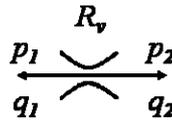


Figure 5. Laminar restrictor

The pressure-flow relationship is:

$$q_2 = \frac{p_1 - p_2}{R_v} \quad (31)$$

The following equations are solved at the component (connected to lines)

$$\begin{aligned} q_2 &= \frac{p_1 - p_2}{R_v} \\ q_1 &= -q_2 \\ p_1 &= c_1 + q_1 Z_{c1} \\ p_2 &= c_2 + q_2 Z_{c2} \end{aligned} \quad (32)$$

Here R_v is the resistance of the orifice, Z_{c1} and Z_{c2} are the characteristic impedances of the lines connected to the orifice. Being a non-dynamic linear system, these equations can be solved algebraically for q_1 and q_2 .

The equations used in the executable component model will thus be :

$$\begin{aligned} q_2 &= \frac{c_1 + c_2}{R_v + Z_{c1} + Z_{c2}} \\ q_1 &= -q_2 \\ p_1 &= c_1 + q_1 Z_{c1} \\ p_2 &= c_2 + q_2 Z_{c2} \end{aligned} \quad (33)$$

A comparison with Eq.(32) and Eq.(36) shows that the adoption to transmission lines has the same effect on the equations as adding restrictors with the resistance Z_c . As a consequence, it is rather uncomplicated to modify any component or simulator to adapt to transmission lines. The same principle is valid also for mechanical nodes. However, since a resistor is a non-dynamic component, one algebraic state is usually introduced for each connector.

The figure below shows the block diagram of the restrictor

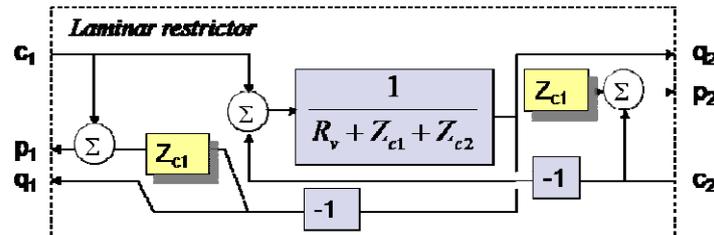


Figure 6. Block diagram of restrictor.

The laminar resistor is a simple linear model that can be derived analytically. The more general approach is using Newton-Raphson Eq.(9).

Modelling of Systems

These components can be connected for system simulation. Here also a pressure source at the right end has been added. Modelling of mechanical springs is performed in exactly the same way as the volume, since they also represent pure capacitances. The only difference is that it handles speed, instead of flow, and force, instead of pressure. The modelling of two dimensional mechanical systems is presented in detail in Krus 1995 [8].

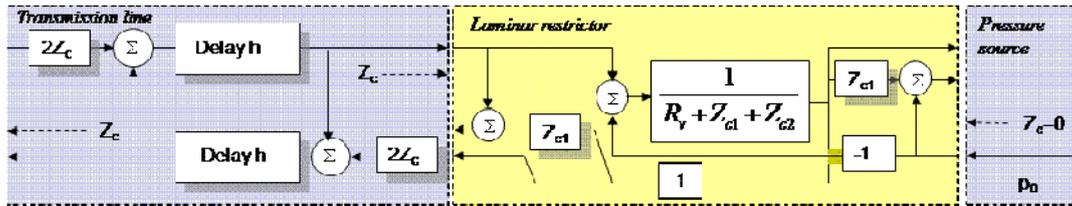


Figure 7. Assembled block diagram, showing transmission line – laminar restrictor – pressure boundary condition.

Test Example

As a test a simple spring mass system is simulated.



Figure 8. Mass spring test system.

The spring is modelled as a bi-directional delay line and is connected to a fixed point to the left and to a pure inertia load to the right. The mass is integrated using bilinear transform (equivalent to the trapezoidal rule). As can be seen the time step can be increased to 0.1 seconds and the system still shows benign behaviour, only the frequency is shifted slightly. If the time step is increased to one second the response is distorted, and much slower than the exact solution due to the influence of parasitic inductance from equation (26), but the system is still stable. If this had been a very fast component in a system, and had not been of prime interest, this behaviour could still be tolerated.

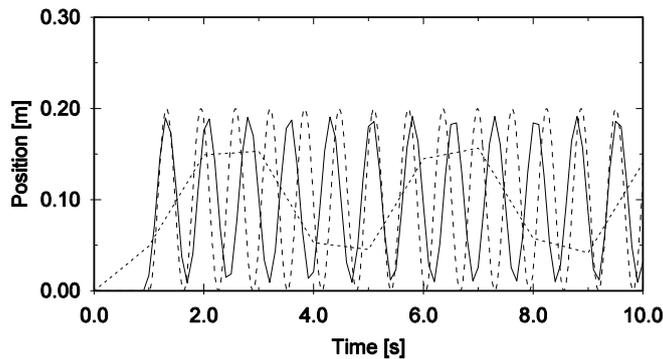


Figure 9. Simulated result of spring mass system with spring represented by transmission line. Dashed line is $h=0.01$ sec, filled is $h=0.1$ sec and dotted line is $h=1$ sec.

As a reference the spring is modelled to simply yield a force to corresponding to the compression of the spring. For slow variations, or very small time step, this should give the same result. However, since the spring is modelled as a separate component there is a time delay of one time step from the force calculated in the spring is affecting the mass, and an updated position of the mass is sent back to the spring. As a result, even with the smallest time step of one millisecond, the system show considerable divergence, and is completely unstable for a time step of 10 milliseconds. To be able to use larger time steps, it would be necessary to use one solver for the whole model.

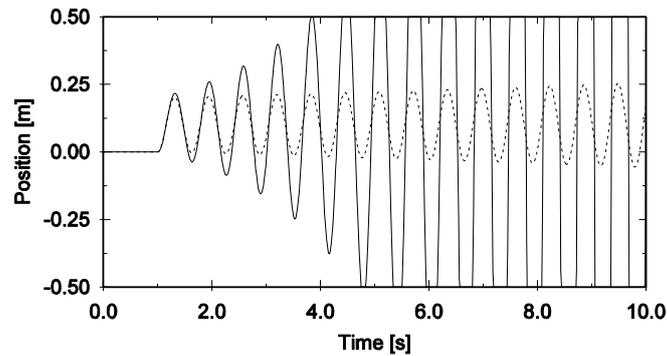


Figure 10. Simulation result of mass spring system with spring force calculated as a function of compression. Time step $h = 0.001$ seconds is dotted. $h = 0.01$ seconds is filled.

Event Free Modelling

In real time simulation it is not practical to handle events by finding zero-crossings and restart solvers when e.g. limitations in variables are hit. Therefore other mechanisms need to be used. Using the approach with differential algebraic equations that are solved through bilinear transformation and Newton-Raphson using an analytical Jacobian, it might seem that the Jacobian would be extremely difficult to derive, since most manually written models involve a great deal of conditions and jumps as they are written in a procedural style. The introduction of an automated approach means that a functional programming style is imposed (the derivative of a function can always be defined, except in singular points). Therefore all conditions have to be represented by functions. The algorithm

```
if(cond)then
```

```
  a = a1
```

```
  b = b1
```

```
  c = c1
```

```
else
```

```
  a = a2
```

```
  b = b2
```

```
  c = c2
```

```
end if
```

can be transformed into

```
a=If(cond,a1,a2)
```

```
b=If(cond,b1,b2)
```

```
c=If(cond,c1,c2)
```

Since the If function is a piecewise continuous function the differential is defined and can be used in the Jacobian. An advantage with this style of programming is that the variables a, b, and c in the example are forced to be defined for both cases, and cannot be forgotten as in normal procedural programming. This approach does not remove the events as such but there is no notion of events in the code since everything is hidden in the functions.

Application

The technique described here can be used to connect large systems for high-speed simulation. One example is given by the aircraft system model shown in the figure below. This particular model has been used to demonstrate optimization of flight control system including actuator system [12].

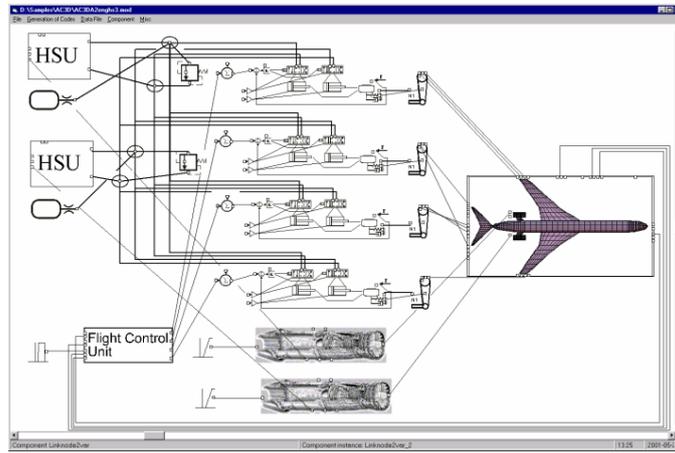


Figure 11. Aircraft simulation model

Another application is to use transmission line elements for connecting different solvers to each other in co-simulation. An example of simulation of a complete wheel loader can be found in [3] and [9].

The HOPSAN simulation package has been developed at the division since the late 1970s, and has played an important role in research projects over time. It has also been used widely in the industry. In recent years commercial software has become increasingly available, and those have also been used at the division. In 2009 the development of a new simulation platform, HOPSAN NG, was initiated. This is an object-oriented C++ application with focus on multi-core support and compatibility.

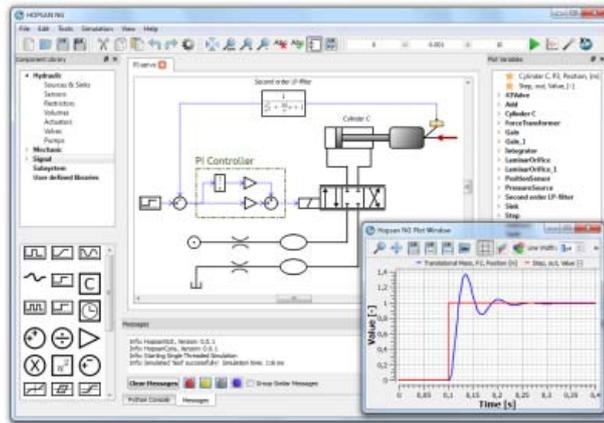


Figure 12. The HOPSAN-NG simulation package

In this paper the method of using transmission lines for partitioning complex system models has been described. Using such elements it is possible to use highly robust distributed solvers on small subsystems, which are then connected to each other using the transmission lines, for system simulation. As a result highly robust models are achieved that can be used also for real time simulation. Using transmission lines to numerically partition a system is a non-exclusive approach that can be used together with conventional solvers. In this way it is possible to obtain a very robust system modelling concept that that has been described here, which is also very useful also for real time simulation. It is also a very useful technique for connecting different simulation software in co-simulation. The transmission line modelling is implemented successfully in the HOPSAN-NG simulation software.

CONCLUSIONS

In this paper the method of using transmission lines for partitioning complex system models has been described. Using such elements it is possible to use highly robust distributed solvers on small subsystems, which are then connected to each other using the transmission lines, for system simulation. As a result highly robust models are achieved that can be used also for real time simulation. Using transmission lines to numerically partition a system is a non-exclusive approach that can be used together with conventional solvers. In this way it is possible to obtain a very robust system modelling concept that that has been described here, which is also very useful also for real time simulation. It is also a very useful technique for connecting different simulation software in co-simulation. The transmission line modelling is implemented successfully in the HOPSAN-NG simulation software.

REFERENCES

- [1] Auslander D. M., 'Distributed System Simulation with Bilateral Delay-Line Models' *Journal of Basic Engineering*, Trans. ASME p195-p200, June 1968.
- [2] Johns P. B. 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.
- [3] Krus P., A. Jansson, J-O. Palmberg, K. Weddfeldt. "Distributed Simulation of Hydromechanical Systems". Third Bath International Fluid Power Workshop, Bath, UK 1990.
- [4] Burton, J. D., Edge, K. A., Burrows, C. R., 1993. Partitioned simulation of hydraulic systems using transmission-line modelling. In: ASME WAM, New Orleans, LA, USA. Publ by ASME, New York, NY, USA. (Using Transputer technology)
- [5] Anagnostopoulou D. and M. Nikolaidou. Timing issues and experiment scheduling in faster-than-real-time simulation. *SIMULATION*, 79(11):613–625, November 2003.
- [6] Fritzson P. "Principles of Object Oriented modelling and Simulation with Modelica 2.1", Wiley-IEEE Press. 2003.
- [7] Elmqvist H., M. Otter, F. E. Cellier. "Inline Integration: a New Mixed Symbolic/Numerical Approach for Solving Differential Algebraic Equation Systems". European Simulation Multiconference, Prague, Czech Republic, 1995.
- [8] Krus P., "An Automated Approach for Creating Components and Subsystems for Simulation of Distributed Systems", Ninth Bath International Fluid Power Workshop, Bath, UK 1996.
- [9] Krus P, K Weddfelt, J-O Palmberg, "Fast Pipeline Models for Simulation of Hydraulic Systems". *Journal of Dynamic Systems, Measurement and Control*, Trans. ASME, March 1994.
- [10] Larsson J., Interoperability in Modelling and Simulation. Linköping Studies in Science and Technology Dissertation 2003.
- [11] Lantto B., H. Ellström, Hampus Gavel, M. Jarelande, Sören Steinkellner, A. Järlestål and M. Landberg, "Modeling and Simulation of Gripen's Fluid Power Systems", Proceedings of "Recent advances in aerospace actuation systems and components", Toulouse, France, 2004.
- [12] Krus P. "Simulation Based Optimisation for Aircraft Systems", SAE 2003 Transactions Journal of Aerospace.

RESPONSIBILITY NOTICE

The author is the only responsible for the printed material included in this paper.