

COMPOSITION OF NUMERICAL INTEGRATORS IN THE HYFLOW FORMALISM

Fernando J. Barros

Department of Informatics Engineering
University of Coimbra
3030 Coimbra, PORTUGAL

ABSTRACT

The representation of complex Cyber-Physical Systems usually requires the ability to combine models described in different modeling formalisms and involving several numerical methods. While ordinary differential equations (ODEs) are a common formalism for representing continuous systems, their solution cannot be efficiently and accurately accomplished by a single numerical integrator. A plethora of methods have been developed to tackle several issues including efficiency, accuracy, and stiffness. Given the variety of methods, a common representation to enable their interoperability is still a major research challenge. In this paper, we introduce the Hybrid Flow System Specification formalism (HyFlow) as a general framework for describing numerical integrators in a modular form. Each numerical method is represented as a particular model in the formalism. Co-simulation is then enabled, since all HyFlow models can be composed by design. In this paper, we show examples demonstrating the co-simulation of geometric, pulse, and exponential numerical integrators.

1 INTRODUCTION

Ordinary differential equations (ODEs) play a fundamental role in the representation of continuous systems. However, their numerical simulation cannot be achieved by a single method since no universal algorithm has been able to produce accurate and efficient solutions for all kinds of ODEs. In fact, although ODEs provide a unifying representation for continuous systems, they can pose challenging problems when numerically integrated, including stiffness and the conservation of invariants, like system energy.

Traditional representations of ODEs rely on the analog computer paradigm and require no explicit representation of numerical methods (Henzinger 1996). In these approaches, the modeler only needs to describe the ODEs (Praehofer 1991), and in some cases to choose the numerical method for handling the overall set of ODEs (Fritzson 2015). Although at a first glance this is a good solution, since it frees users from numerical details, it has several limitations. The co-simulation is not guaranteed since ODEs need to be transformed and converted into a set of first-order ODEs and collectively solved by a single numerical integrator. Another limitation of these approaches is the difficulty in introducing new numerical integrators, since they are not explicitly represented. Given that these frameworks only offer ODEs as first class constructs, solutions requiring the combination of different families of numerical integrators can also not be described.

Several families of numerical methods have been developed to solve ODE systems. Methods include, for example, Adams and Backward Differentiate Formulas (BDF) (Hairer et al. 2000), the latter intended to integrate *stiff* ODEs. Both approaches require mapping of all ODEs into a system of first-order ODEs, and they both rely on the polynomial approximation of signals.

Unfortunately, these methods do not provide a universal solution for all kinds of ODEs. Energy-preserving systems, for example, require geometric integrations that exhibit excellent properties of energy conservation (Hairer et al. 2005). This property is fundamental to simulate some systems, like celestial mechanics, during long periods, and it cannot be guaranteed by other types of integrators. Stiffness,

although it can be addressed by BDF methods, can in some cases be more accurately and efficiently solved by exponential integrators (Cox and Matthews 2002; Hochbruck and Osterman 2011), which provide an explicit approach to the solution of stiff ODEs based on an exponential approximation of signals. This method is also interesting, since it challenges the traditional characterization of the stiffness phenomena, pointing stiffness as a limitation of the polynomial integrators rather than an intrinsic characteristic of the ODEs.

The use of a modeling formalism to explicitly represent numerical solvers was described by Zeigler and Lee (1998). This work uses DEVS as the underlying representation for continuous variables, while taking advantage of discrete-event semantics. The DEVS representation of continuous systems has evolved from piecewise constant segments to continuous segments described by polynomials (Migoni et al. 2013). However, a polynomial representation is not adequate to describe exponential integrators. Additionally, discrete event semantics is also not adequate to represent geometric integrators, since stepsize adaption cannot be controlled by conventional schemes based on the local error (Calvo and Sanz-Serna 1993).

In this paper, we exploit the Hybrid Flow System Specification formalism (HYFLOW) for providing a unifying representation for different families of numerical integrators. HYFLOW can represent hybrid systems as a combination of independent units that are known only by their interface. HYFLOW enables the co-simulation, since formalism preserves model modularity during simulation. In this paper, we demonstrate that HYFLOW can provide a unifying formalism for the co-simulation of exponential integrators, geometric integrators, hybrid generators, and piecewise constant signal integrators.

2 THE HYFLOW FORMALISM

The Hybrid Flow System Specification (HYFLOW) is a formalism aimed to represent hybrid systems with a time-variant topology (Barros 2003). HYFLOW achieves the representation of continuous variables using the concept of multi-sampling (Barros 2000; Barros 2002), while the representation of discrete events is based on the Discrete Event System Specification (DEVS) (Zeigler 1984). HYFLOW has two types of models: basic and network. Basic models provide state representation and transition functions. Network models are a composition of basic models and other network models. Given its definition, a network provides an abstraction for representing hierarchical systems.

2.1 The Basic HyFlow Model

We consider \widehat{B} as the set of names corresponding to basic HYFLOW models. A HYFLOW basic model associated with name $B \in \widehat{B}$ is defined by

$$M_B = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)$$

where

$X = X_c \times X_d$ is the set of input flow values

X_c is the set of continuous input flow values

X_d is the set of discrete input flow values

$Y = Y_c \times Y_d$ is the set of output flow values

Y_c is the set of continuous output flow values

Y_d is the set of discrete output flow values

P is the set of partial states (p-states)

$P_0 \subseteq P$ is the set of (valid) initial p-states

$\rho : P \rightarrow \mathbb{H}_0^+$ is the time-to-input function

$\omega : P \rightarrow \mathbb{H}_0^+$ is the time-to-output function

$S = \{(p, e) | p \in P, 0 \leq e \leq v(p)\}$ is the state set
 with $\tau(p) = \min\{\rho(p), \omega(p)\}$, the time-to-transition function
 $\delta : S \times X^\phi \rightarrow P$ is the transition function
 where $X^\phi = X_c \times (X_d \cup \{\phi\})$
 and ϕ is the null value (absence of value)
 $\Lambda_c : S \rightarrow Y_c$ is the continuous output function
 $\lambda_d : P \rightarrow Y_d$ is the partial discrete output function

The HYFLOW time base is the set of hyperreals numbers $\mathbb{H} = \{x + z\varepsilon | x \in \mathbb{R}, z \in \mathbb{Z}\}$, where ε is an infinitesimal value, such that $\varepsilon > 0$ and $\varepsilon < \frac{1}{n}$ for $n = 1, 2, 3, \dots$ (Goldblatt 1998). The set of positive hyperreals is defined by $\mathbb{H}_0^+ = \{h \in \mathbb{H} | h \geq 0\}$. The discrete output of a component described by a HYFLOW basic model is constrained to be null (ϕ) when in states (s, e) with $e \neq \omega(p)$.

Figure 1 depicts the typical trajectories of a HYFLOW component. At time t_1 the component in p-state p_0 samples its input since its elapsed time reaches $\rho(p_0) = e$. The component changes its p-state to $p_1 = \delta((p_0, \rho(p_0)), (x_1, \phi))$, where x_1 is the sampled value and no discrete flow is present.

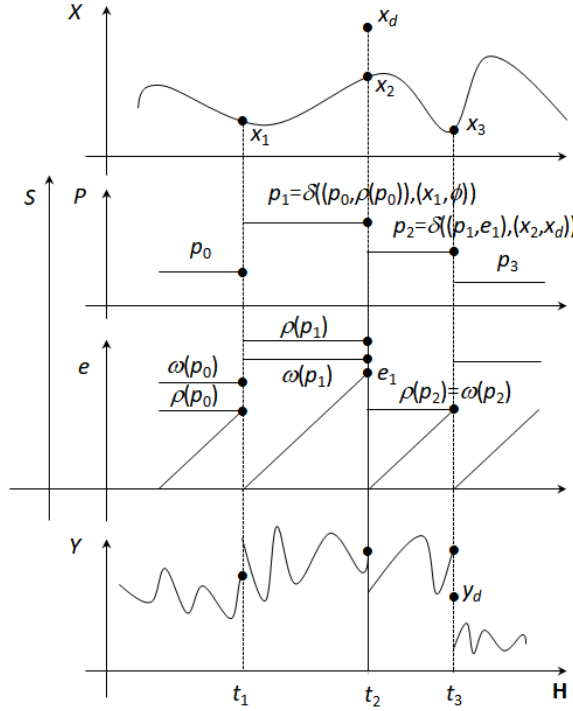


Figure 1: Basic HYFLOW component trajectories.

At time t_2 the discrete flow x_d is received by the component that changes to p-state $p_2 = \delta((p_1, e_1), (x_2, x_d))$, where x_2 is the continuous flow at t_2 . At time t_3 the component reaches the time-to-output time limit and it changes to p-state $p_3 = \delta((p_2, \omega(p_2)), (x_3, \phi))$. At this time the discrete flow $y_d = \lambda_d(p_2)$ is produced. Additionally, component continuous output flow is always present and given by $\Lambda_c(p, e)$. The formalism has shown be deterministic and closed under the coupling operation. The semantics of basic models is given by HYFLOW basic components described in the next section.

2.2 Co-Simulation of HyFlow Basic Models

To define HYFLOW co-simulation semantics we use the concept of HYFLOW component (Barros 2008). Components have their behavior governed by HYFLOW models but they establish the rules on how models are interpreted. Co-simulation has been defined in the context of discrete event systems (Zeigler 1984), where the *abstract simulator* concept has been introduced.

A HYFLOW basic component obeys to the *non-instantaneous assumption* (Barros 2016b), implying that a transition occurring at time t only takes effect at time $t^+ = t + \varepsilon$. This constraint enables the deterministic simulation of HYFLOW models (Barros 2008), particularly at times when a network model undergoes a change in the topology or when there are loops of zero-time delay. Considering $\widehat{\chi}$ as a set of component names, a basic component named $\chi \in \widehat{\chi}$ associated with model $M_B = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)$ is defined by:

$$\Xi_{\chi}^B = (\langle p, t, y \rangle, T, V, \Lambda, \Delta)$$

where

p is the component p-state

t is the time of component last transition

$y \in Y^{\phi}$ is the component output value, with $Y^{\phi} = Y_c \times (Y_d \cup \{\phi\})$

$T : \{\} \longrightarrow \mathbb{H}$ is the time of the next transition function, defined by:

$$T() = t + \min \{ \rho(p), \omega(p) \}$$

$V : \{\} \longrightarrow Y^{\phi}$ is the component output function, defined by:

$$V() = y$$

$\Lambda : \mathbb{H}$ is the component output action, defined by:

$$\Lambda(\tau) \triangleq$$

if $(\tau - t = \omega(p))$

$$y \leftarrow (\Lambda_c(p, \tau - t), \lambda_d(p))$$

else

$$y \leftarrow (\Lambda_c(p, \tau - t), \phi)$$

$\Delta : \mathbb{H} \times X^{\phi}$ is the component transition action, defined by:

$$\Delta(\tau, (x_c, x_d)) \triangleq$$

if $(\tau \neq T() \wedge x_d \neq \phi)$ **return**

$$p \leftarrow \delta((p, \tau - t), (x_c, x_d))$$

$$t \leftarrow \tau + \varepsilon$$

where by the non-instantaneous propagation, the component changes its state only at $\tau + \varepsilon$

The creation, at time t_0 , of a component named χ associated with the basic model $M_B = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)$ and model initial state $(p_0 \in P_0, e_0) \in S$, is achieved by:

$$\text{create-basic-component}(\chi \in \widehat{\chi}, B \in \widehat{B}, t_0 \in \mathbb{H}, (p_0 \in P_0, e_0) \in S) \triangleq (\langle p_0, t_0 - e_0, \Lambda(t_0) \rangle, T, V, \Lambda, \Delta)$$

HYFLOW components enable co-simulation, since they only use the information provided by the model interface. Additionally, since basic components can be combined into networks, the co-simulation is achieved by orchestrating the behavior of basic components (Barros 2016b). Next, we provide two examples of HYFLOW models: a square wave generator and an integrator of piecewise constant signals.

2.3 Example: Square Wave Generator

Generators provide a simple way for creating signals. A square wave is a hybrid signal that can be described by HYFLOW continuous and discrete flows. This signal is characterized by two piecewise constant segments that change periodically over time. However, for simulation efficiency, changes need to be signaled by a discrete flow (event). We consider a signal in the range $\{min, max\}$ with time-up and time-down intervals equal to tUp and $tDown$, respectively. The generator is described by the HYFLOW model:

$$M_{\square} = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)$$

where

$$\begin{aligned} X &= \{\} \times \{\}; Y = \mathbb{R} \times \mathbb{R} \\ P &= \{(max, min, tUp, tDown, b) | max, min, tUp, tDown \in \mathbb{R}; b \in \{\top, \perp\}\}; P_0 = P \\ \rho(max, min, tUp, tDown, b) &= \infty; \\ \omega(max, min, tUp, tDown, \top) &= tUp; \omega(max, min, tUp, tDown, \perp) = tDown \\ \delta(((max, min, tUp, tDown, \top), e), (x_c, x_d)) &= (max, min, tUp, tDown, \perp) \\ \delta(((max, min, tUp, tDown, \top), e), (x_c, x_d)) &= (max, min, tUp, tDown, \top) \\ \Lambda_c((max, min, tUp, tDown, \top), e < tUp) &= up \\ \Lambda_c((max, min, tUp, tDown, \top), e == tUp) &= down \\ \Lambda_c((max, min, tUp, tDown, \perp), e < tDown) &= down \\ \Lambda_c((max, min, tUp, tDown, \perp), e == tDown) &= up \\ \lambda_d(max, min, tUp, tDown, \top) &= down; \lambda_d(max, min, tUp, tDown, \perp) = up \end{aligned}$$

The generator issues a discrete flow for signaling a new segment, after a time-up or time-down period. Between pulses, the continuous flow is constant and given by the function Λ_c . At the end of a segment ($e == tUp$ or $e == tDown$), the continuous is made the same as the discrete flow.

2.4 Example: Pulse Integrator

The integration of ODEs plays a key role in describing many classes of dynamical systems. In previous work, we have developed general purpose ODE integrators (Barros 2015). However, more efficient algorithms can be used for particular types of signals. Piecewise constant flows, like the square wave described above, can be integrated with a simpler and more efficient solver. An integrator for piecewise constant signals can be represented by the HYFLOW model:

$$M_{\pi} = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)$$

where

$$\begin{aligned} X &= Y = \mathbb{R} \times \mathbb{R} \\ P &= \{(y, v, b) | y, v \in \mathbb{R}; b \in \{\top, \perp\}\}; P_0 = \{(y_0, 0, \top) \in P\} \\ \rho(y, v, b) &= \infty \\ \omega(y, v, \top) &= 0; \omega(y, v, \perp) = \infty \\ \delta(((y, v, \perp), e), (x_c, x_d)) &= (y + ve_{std}, x_c, \top) \\ \delta(((y, v, \top), e), (x_c, x_d)) &= (y, v, \perp) \\ \Lambda_c((y, v, b), e) &= y + ve_{std} \\ \lambda_d(y, v, b) &= y \end{aligned}$$

The model obtains its efficiency by exploiting the features of the input signal. Contrarily to integrators that need to handle arbitrary input segments, this model avoids sampling, because all changes in the input value are signaled by a discrete flow. Model continuous flow output is piecewise linear and described by the function Λ_c . This flow is available by other components that can sample it asynchronously at their own sampling rate. To facilitate model composition, all discrete flows received by the integrator are sent as discrete flow outputs. An associated component to this model needs to specify an initial value for the integral y_0 . The initial value of v is unknown and it is set to 0 in p-state p_0 . An associated component samples its input at simulation start to obtain v_0 .

2.5 HyFlow Network Model

HYFLOW network models are compositions of HYFLOW models (basic or other HYFLOW network models). Let \widehat{N} be the set of names corresponding to HYFLOW network models, with $\widehat{N} \cap \widehat{B} = \{\}$. Formally, a HYFLOW network model associated with name N is defined by

$$M_N = (X, Y, \eta)$$

where

N is the network name

$X = X_c \times X_d$ is the set of network input flows

X_c is the set of network continuous input flows

X_d is the set of network discrete input flows

$Y = Y_c \times Y_d$ is the set of network output flows

Y_c is the set of network continuous output flows

Y_d is the set of network discrete output flows

η is the name of the dynamic topology network executive

The executive model is a modified HYFLOW basic model, defined by

$$M_\eta = (X_\eta, Y_\eta, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d, \widehat{\Sigma}, \gamma)$$

where

$\widehat{\Sigma}$ is the set of network topologies

$\gamma: P \rightarrow \widehat{\Sigma}$ is the topology function

The network topology $\Sigma_\alpha \in \widehat{\Sigma}$, corresponding to the p-state $p_\alpha \in P$, is given by

$$\Sigma_\alpha = \gamma(p_\alpha) = (C_\alpha, \{I_{i,\alpha}\} \cup \{I_{\eta,\alpha}, I_{N,\alpha}\}, \{F_{i,\alpha}\} \cup \{F_{\eta,\alpha}, F_{N,\alpha}\})$$

where

C_α is the set of names associated with the executive state p_α

for all $i \in C_\alpha \cup \{\eta\}$

$I_{i,\alpha}$ is the sequence of influencers of i

$F_{i,\alpha}$ is the input function of i

$I_{N,\alpha}$ is the sequence of network influencers

$F_{N,\alpha}$ is the network output function

For all $i \in C_\alpha$

$$M_i = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)_i \text{ if } i \in \widehat{B}$$

$$M_i = (X, Y, \eta)_i \text{ if } i \in \widehat{N}$$

The topology of a network is defined by its executive through the topology function γ , which maps the executive p-state into network composition and coupling. Thus, topology adaption can be achieved by changing the executive p-state.

HYFLOW network models are simulated by HYFLOW network components that perform the orchestration of basic or other networks. Network co-simulation is achieved by a general communication protocol that relies only on the component interface. This protocol is completely independent from model details, enabling the composition of components that are interpreted as black boxes. A description of the HYFLOW network master co-simulation algorithm is provided by Barros (2008).

3 EXPONENTIAL TIME DIFFERENCING

Common numerical integrators like Adams and BDF are based on polynomial interpolation. A different approach has been taken by exponential time differencing (ETD) methods that use the exact solution for the linear part of the ODE. ETD methods have several advantages over conventional integrators, including the possibility of using large stepsize, even when integrating so-called stiff ODEs. ETD methods consider ODEs composed by a linear part and a non-linear part in the form of:

$$y' = cy + F(x(t), y), y(0) = y_0$$

where F is a non-linear function. The *exact* solution for $t \in [0, T]$ is given in (Cox and Matthews 2002):

$$y(t) = y(0)e^{ct} + e^{ct} \int_0^t e^{-c\tau} F(x(\tau), y(\tau)) d\tau$$

For numerical integration an approximation F of is required. Considering an integrator with a fixed stepsize T , the solution for a zero-order approximation of F is given by Cox and Matthews (2002):

$$y_{n+1} = y_n e^{cT} + \frac{e^{cT} - 1}{c} F(x_n, y_n)$$

The HYFLOW representation for a zero-order ETD that works at a fixed stepsize T is given by:

$$M_x = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda, \lambda_d)$$

where

$$X = \mathbb{R} \times \mathbb{R}; Y = \mathbb{R} \times \{\phi\}$$

$$P = \{(\alpha, x, y) | \alpha, x, y \in \mathbb{R}\}$$

$$P_0 = \{(0, 0, y_0) \in P\}$$

$$\rho(\alpha, x, y) = \alpha; \omega(\alpha, x, y) = \infty$$

$$\delta((0, x, y), h), (x_c, x_d) = (T, x_c, y)$$

$$\delta((\alpha, x, y), h), (x_c, x_d) = (\alpha, x_c, ye^{c\bar{h}} + \frac{e^{c\bar{h}} - 1}{c} F(x, y))$$

$$\Lambda_c((\alpha, x, y), h) = ye^{c\bar{h}} + \frac{e^{c\bar{h}} - 1}{c} F(x, y)$$

$$\lambda_d(\alpha, x, y) = \phi$$

with $\bar{h} = h_{std}$

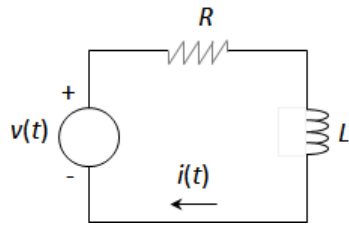


Figure 2: RL circuit.

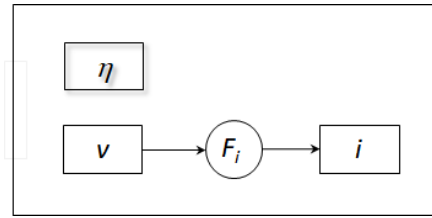


Figure 3: HYFLOW RL circuit network.

The model specifies an initial value $\alpha = 0$ so it samples the input value at simulation start. After the beginning, the sampling period is set to T . The sampling interval can be modified by discrete flows that usually signal an input discontinuity. ETD continuous flow output is an exponential function described by the continuous output function Λ_c . Likewise the previous models, the ETD integrator can be sampled at an asynchronous rate, making composition with other HYFLOW models a simple task. The next example describes an electrical RL circuit solved by an ETD integrator.

3.1 Example: RL Circuit

We consider the electrical circuit of Figure 2 composed by a voltage generator $v(t)$, a resistor R and an inductance L . The circuit current is described by:

$$\frac{di}{dt} = v - \frac{R}{L}i$$

The HYFLOW network representation is depicted in Figure 3. Considering $v(t)$ as a square wave generator, the ETD integrator described above provides an exact solution. The HYFLOW network is defined by:

$$C = \{v, i\}, I_\eta = I_v = \{\}, I_i = \{v\}$$

$$F_i(x_c, x_d) = (x_c, x_d)$$

We obtained simulation results using HYFLOW++, an implementation of HYFLOW in the C++17 language. The network of Figure 3 is described by the HYFLOW++ model given in Figure 4, which defines the following circuit parameters: $R = 0.5 \Omega$, and $L = 0.01$ H. The square wave is defined by voltages -40 V, 40 V and by a period of 0.2 s (time up = time down = 0.1 s). Results for a simulation time of 1 s are depicted in Figure 5. As mentioned before, these results are exact since the input signal for the ETD integrator is piecewise constant. Given that the integration is exact, it requires no sampling, the computation being entirely driven by the discrete flows describing the square wave. HYFLOW support for hybrid models can, thus, increase simulation efficiency.

```

class CPS_RL: public Executive {
    const double R = 0.5; //Ohm
    const double L = 1.0e-2; //H
public:
    CPS_RL(const string& name): Executive(name) {
        add(new SquareWave("v", 40, -40, 0.1, 0.1));
        add(new ETD("i", -R/L, 0, 1)); //ETD constant = -R/L, initial current = 0, sampling time = 1s
        influencers("i", {"v"});
    }
};

```

Figure 4: HYFLOW++ RL circuit network topology.

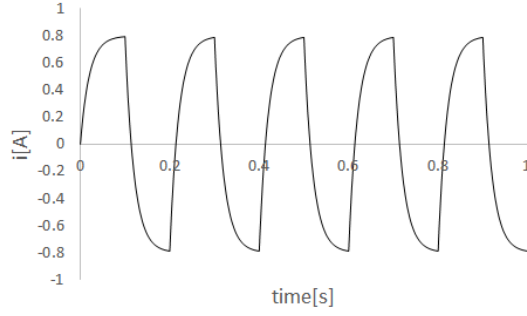


Figure 5: RL circuit current for a square wave voltage.

4 GEOMETRIC INTEGRATOR

The common rule in modeling and simulation tools is to map a set of arbitrary order ODEs into a system of first-order ODEs (Fritzon 2015). This approach is not acceptable for simulating long time periods in many systems, like energy conserving systems, where solutions have properties that can only be observable after long simulations. For example, in celestial mechanics, properties like the precession of planets may require the simulation of hundreds of years to be characterized. The traditional methods based on first-order decomposition are not acceptable in these cases, since they do not preserve system energy. To overcome these limitations, direct representations of second-order ODEs have been developed (Hairer et al. 2005). We introduce now the HYFLOW representation for second-order geometric ODEs integrators. Given the second-order ODE

$$y'' = f(x(t), y) \quad \text{with } y(0) = y_0, y'(0) = v_0$$

and using the variable $v = y'$, a fixed stepsize T , second-order ODE, second-degree polynomial approximation, geometric integrator is described by the equations (Swope et al. 1982):

$$y_{n+1} = y_n + hv_n + \frac{1}{2}h^2 f_n$$

$$v_{n+1} = v_n + \frac{h}{2}(f_n + f_{n+1})$$

The HYFLOW corresponding model is given by:

$$M_\Gamma = (X, Y, P, P_0, \rho, \omega, \delta, \Lambda_c, \lambda_d)$$

where

$$X = \mathbb{R} \times \mathbb{R}; Y = \mathbb{R} \times \phi$$

$$P = \{(\alpha, y_n, v_n, f_n) | y_n, v_n, f_n \in \mathbb{R}\}$$

$$P_0 = \{(0, y_0, v_0, 0) \in P\}$$

$$\rho(\alpha, y_n, v_n, f_n) = \alpha$$

$$\omega(\alpha, y_n, v_n, f_n) = \infty$$

$$\delta((0, y_n, v_n, f_n), e), (x_c, x_d) = (T, y_n, v_n, f(x_c, y_n))$$

$$\delta((\alpha, y_n, v_n, f_n), e), (x_c, x_d) = (\alpha, y_{n+1}, v_n + \frac{1}{2}e_{std}(f_n + f_{n+1}), f_{n+1})$$

$$\text{with } y_{n+1} = y_n + e_{std}v_n + \frac{1}{2}e_{std}^2 f_n \text{ and } f_{n+1} = f(x_c, y_{n+1})$$

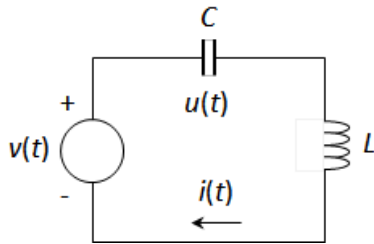


Figure 6: LC circuit.

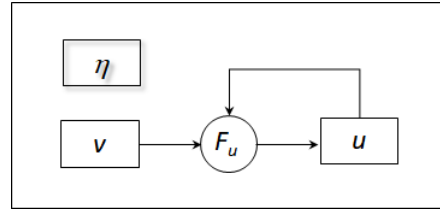


Figure 7: HYFLOW LC circuit network.

$$\Lambda_c((\alpha, y_n, v_n, f_n), e) = y_n + e_{std}v_n + \frac{1}{2}e_{std}^2f_n$$

$$\lambda_d(\alpha, y_n, v_n, f_n) = \phi$$

The recurrences are computed by the transition function δ , and the sampling interval T is specified by function ρ . The output flow is given by the second-degree polynomial provided by function Λ_c . Although sampling time is constant (T), the solver can accept discrete flows that trigger the transition function and provide asynchronous integration behavior. Asynchronous sampling becomes important for performing the integration of discontinuous functions and it is used in the next example for handling discontinuities in the input signal.

4.1 Example: LC Circuit

As an example for the application of the geometric integrator we consider the LC circuit with inductance L and capacity C connected to a voltage source $v(t)$ as depicted in Figure 6. The capacitor voltage $u(t)$ is described by:

$$u'' = \frac{v - u}{LC}$$

The circuit current $i(t)$ is given by:

$$i = Cu'$$

Since the electric circuit has no dissipative elements (resistors), it becomes a second-order energy conservative system. A good solution is provided by the geometric integrator described above. The HYFLOW network representation of the circuit is depicted in Figure 7.

For simulation, we consider a circuit with $C = 1\text{mF}$ and an inductance $L = 1\text{mH}$. The square input voltage is defined by $\{-5, 5\}$ V and has a half-period of 0.002s. Capacitor voltage and current are represented in Figure 8. The plot of u vs. i is depicted in Figure 9. Results were obtained with a stepsize of $2 \cdot 10^{-5}\text{s}$ and a simulation time of 0.1s.

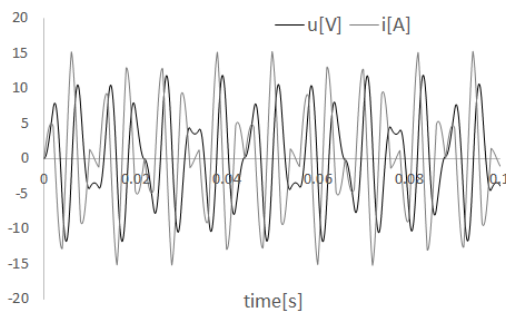


Figure 8: Capacitor voltage and current.

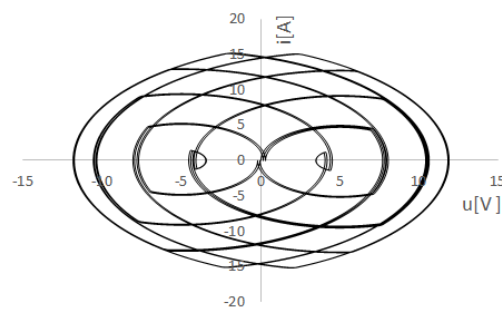


Figure 9: Capacitor $u \times i$ for a square input voltage.

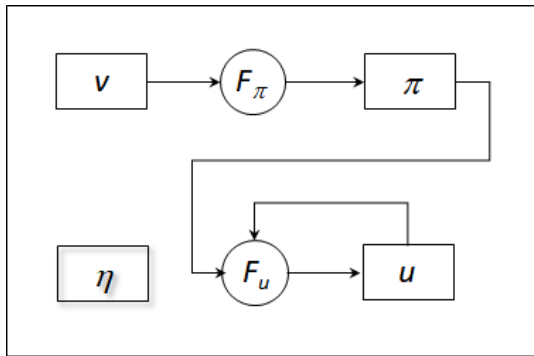


Figure 10: LC circuit with the additional pulse integrator π .

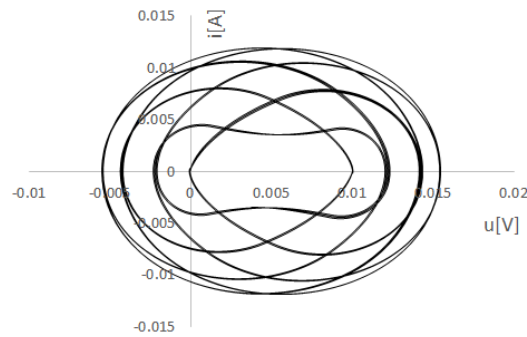


Figure 11: Capacitor $u \times i$ for a triangular input voltage.

To show HYFLOW's ability to seamlessly integrate additional models, we consider the pulse integrator defined in Section 2.2, now combined with the square wave generator, transforming the original signal into a triangular wave. The LC circuit is now described by Figure 10, where π is the pulse integrator. Simulation results considering the pulse integrator are depicted in Figure 11. The remaining parameters are kept unchanged.

While the geometric integrator is a sample-based model, both the square wave generator and the pulse integrator are event-based models. HYFLOW, however, enables the seamless integration of these models without making any assumption about their internal behavior, showing formalism composition and interface adaptation capabilities. HYFLOW can, thus, support the co-simulation of a large variety of models (Barros 2016a).

5 CONCLUSION

Co-simulation requires the ability to compose models known only by their interfaces. The HYFLOW modular co-simulators keep model modularity during simulation, requiring only the external information published by the HYFLOW models, being fully compatible with the data-hiding requirements of co-simulation. In this paper, we have shown the ability of HYFLOW to compose models based on geometric, exponential, and pulse integrators. HYFLOW's dense output enables the use of arbitrary interpolation functions including, for example, polynomials and exponentials. The communication of HYFLOW models is achieved through sampling, which enables seamless model composition. HYFLOW's support for discrete flows enables also the representation of hybrid systems, described by both continuous and discrete event signals, making it an effective framework for describing discontinuities. In future work, we plan to develop a representation for differential algebraic equations (Ascher and Petzold 1988), using the HYFLOW formalism.

REFERENCES

- Ascher, U., and L. Petzold. 1988. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Philadelphia: SIAM.
- Barros, F. 2000. "A Framework for Representing Numerical Multirate Integration Methods". In *AI, Simulation and Planning in High Autonomy Systems*, edited by H. Sarjoughian et al., March 6th–8th, Tucson, AZ, USA.
- Barros, F. 2002. "Towards a Theory of Continuous Flow Models". *International Journal of General Systems* 31(1):29–39.
- Barros, F. 2003. "Dynamic Structure Multiparadigm Modeling and Simulation". *ACM Transactions on Modeling and Computer Simulation* 13(3):259–275.

- Barros, F. 2008. “Semantics of Discrete Event Systems”. In *Distributed Event-Based Systems*, edited by R. Baldoni et al., 252–258. July 1st–4th, Rome, Italy.
- Barros, F. 2015. “Asynchronous, Polynomial ODE Solvers based on Error Estimation”. In *Symposium on Theory of Modeling and Simulation*, edited by F. Barros et al., 115–121. April 12th–15th, Alexandria, VA, USA.
- Barros, F. 2016a. “A Modular Representation of Asynchronous, Geometric Solvers”. In *Symposium on Theory of Modeling and Simulation*, edited by F. Barros et al., April 3rd–5th, Pasadena, CA, USA, article 27.
- Barros, F. 2016b. “On the Representation of Time in Modeling & Simulation”. In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. M. K. Roeder et al., 1571–1582. Piscataway, NJ: IEEE.
- Calvo, M., and J. Sanz-Serna. 1993. “The Development of Variable-step Symplectic Integrators with Application to the Two-Body Problem”. *SIAM Journal Science Computing* 14(4):936–952.
- Cox, S. M., and P. C. Matthews. 2002. “Exponential Time Differencing for Stiff Systems”. *Journal of Computational Physics* (176):430–455.
- Fritzson, P. 2015. *Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach*. 2nd ed. Hoboken, NJ: IEEE Press, Wiley-Interscience.
- Goldblatt, R. 1998. *Lectures on the Hyperreals: An Introduction to Nonstandard Analysis*. Number 188 in Graduate Texts in Mathematics. New York: Springer.
- Hairer, E., C. Lubich, and G. Wanner. 2005. *Geometrical Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. 2nd ed. Berlin: Springer.
- Hairer, E., S. Nørsett, and G. Wanner. 2000. *Solving Ordinary Differential Equations I: Nonstiff Problems*. 2nd ed. Number 14 in Springer Series in Computational Mathematics. Berlin: Springer-Verlag.
- Henzinger, T. 1996. “The Theory of Hybrid Automata”. In *11th Annual IEEE Symposium on Logic in Computer Science*, 278–292. July 27th–30th, New Brunswick, NJ, USA.
- Hochbruck, M., and A. Osterman. 2011. “Exponential Multistep Methods of Adams-type”. *BIT Numerical Mathematics* 51(4):889–908.
- Migoni, G., M. Bortolotto, E. Kofman, and F. Cellier. 2013. “Linearly Implicit Quantization-based Integration Methods for Stiff Ordinary Differential Equations”. *Simulation Modelling Practice and Theory* 35:118–136.
- Praehofer, H. 1991. *System Theoretic Foundations for Combined Discrete-Continuous System Simulation*. Ph. D. thesis, University of Linz.
- Swope, W., H. Andersen, P. Berens, and K. Wilson. 1982. “A Computer Simulation Method for the Calculation of Equilibrium Constants for the Formation of Physical Clusters of Molecules: Application to Small Water Clusters”. *Journal of Chemical Physics* 76(1):637–649.
- Zeigler, B. 1984. *Multifaceted Modelling and Discrete Event Simulation*. San Diego: Academic Press.
- Zeigler, B., and J. Lee. 1998. “Theory of Quantized Systems: Formal Basis for DEVS/HLA Distributed Simulation Environment”. In *Enabling Technology for Simulation Science II*, edited by A. Sisti, Volume 3369 of *SPIE*, 49–58. April 13th–17th, Orlando, FL, USA.

AUTHOR BIOGRAPHY

FERNANDO J. BARROS is a professor at the University of Coimbra. His research interests include theory of modeling and simulation, hybrid systems, and dynamic topology models. He has authored more than 60 scientific publications and he has been responsible for the organization of several M&S conferences. Fernando Barros is a member of the editorial board *International Journal of Simulation and Process Modelling* and associate editor of the *International Journal of Agent Technologies and Systems*. His e-mail address is barros@dei.uc.pt.