LINEARLY IMPLICIT DISCRETE EVENT METHODS FOR STIFF ODE'S.

G. MIGONI† and E. KOFMAN†

†*CIFASIS–CONICET. Laboratorio de Sistemas Dinámicos FCEIA - UNR. Riobamba 245 bis - (2000) Rosario.*

***Abstract*— This paper introduces two new numerical methods for integration of stiff ordinary differential equations. Following the idea of quantization based integration, i.e., replacing the time discretization by state quantization, the new methods perform first and second order backward approximations allowing to simulate stiff systems. It is shown that the new algorithms satisfy the same theoretical properties of previous quantization–based integration methods. The translation of the new algorithms into a discrete event (DEVS) specification and its implementation in a DEVS simulation tool is discussed. The efficience of the methods is illustrated comparing the simulation of two examples with the classic methods implemented by Matlab/Simulink.**

***Keywords*— Stiff System Simulation, Quantization Based Integration, DEVS**

I. INTRODUCTION

The use of traditional methods (Hairer *et al.*, 1993; Hairer and Wanner, 1991; Cellier and Kofman, 2006) based on time discretization to integrate stiff systems require the use of implicit algorithms since the required step size used by explicit methods is limited by the stability region and the resulting step size becomes inadmissibly small (Cellier and Kofman, 2006).

In fact, numerical integration methods that include in their numerically stable region the entire left half ($\lambda \cdot h$) plane (or at least a large portion of it) are necessary for stiff systems integration (Cellier and Kofman, 2006). Only some implicit methods have this type of stability region. Explicit algorithms showing that feature do not exist.

The problem with implicit methods is that they are computationally expensive because in each step they need to use iterative algorithms to determine the next value (usually with the Newton Iteration). The problem becomes critical in applications related to real time simulation, where in many cases performing iterations becomes unacceptable.

An alternative approach to classic time slicing started to develop since the end of the 90's, where time discretization is replaced by state variables quantization. As a result, the simulation models are not discrete time but discrete event systems. The origin of this idea can be found in the definition of Quantized Systems (Zeigler *et al.*, 2000).

This idea was then reformulated with the addition of hysteresis –to avoid the appearance of infinitely fast oscillations– and formalized as the Quantized State Systems (QSS) method for ODE integration in (Kofman and Junco, 2001). This was followed by the definition of the second order QSS2 method (Kofman, 2002), the third order QSS3 method (Kofman, 2006).

The QSS methods showed some important advantages with respect to classic discrete time methods in the integration of discontinuous ODEs (Kofman, 2004), sparsity exploitation (Kofman, 2002), the property of absolute stability, and the existence of a global error bound (Cellier and Kofman, 2006).

In spite of these properties, QSS, QSS2 and QSS3 fail when applied to stiff systems due to the appearance of fast oscillations. They remain numerically stable at the expense of utilizing excruciatingly small step sizes, as any explicit algorithm is expected to require in this situation.

To solve this problem, a first order backward QSS method (called BQSS, after Backward QSS) was proposed in (Migoni *et al.*, 2007). The method was able to efficiently integrate many stiff systems. The basic idea of the BQSS method is to use a *future* value of the states to obtain the quantized value. Yet, whereas the algorithm is implicit, it still can be implemented without Newton iterations. The reason is that each state has only two possible future values. If a state variable currently assumes a value of q_j, the next value of that state variable must be either $q_j + \Delta q_j$, or $q_j - \Delta q_j$. Hence all possible next values can be *enumerated* without an open-ended search. In other words BQSS was the first explicit method for stiff ODEs.

The main drawback of BQSS is that it performs only a first order approximation and accurate results cannot be obtained. Another problem is that BQSS introduced an extra perturbation term that increases the error bound and, in some nonlinear systems, might provoke the appearance of spurious equilibrium points.

This paper presents first a new method that combines the idea of BQSS and linearly implicit integration.

The first order accurate linearly implicit QSS

(LIQSS) method follows the idea of BQSS, but avoids the presence of the mentioned perturbation term and the appearance of spurious equilibrium point using a linearly implicit idea to find the state values where some derivatives cross by zero. Then, a second order accurate LIQSS method (called LIQSS2) is defined combining the principles of LIQSS and the second order accurate QSS method (QSS2). This new method solves, up to certain limits, the problem of accuracy.

The work is organized in the following way: Section II.recalls the principles of Quantization–Based Integration and introduces the problems of QSS methods to deal with stiff systems. Then, Section IIIintroduces the new methods, namely, LIQSS and LIQSS2 and discuss the main implementation issues of them. Section IV.studies the properties of the new methods (legitimacy and error bound). Finally, Section V. introduces some simulation examples and Section VI. presents conclusions and future work.

II. QUANTIZATION-BASED INTEGRATION

This section introduces the principles of QSS methods, shows the problems they have in the simulation of stiff systems, and presents a first approximation to solve these problems (the BQSS method). Finally, a brief introduction to the DEVS formalism is provided and the DEVS implementation of QSS methods is discussed.

A. QSS Method

Consider a time invariant ODE in its State Equation System (SES) representation:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the state vector and $u(t) \in \mathbb{R}^m$ is an input vector, which is a known piecewise constant function.

The QSS method simulates an approximate system, which is called Quantized State System:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{q}(t), \mathbf{u}(t)) \tag{2}$$

where $\mathbf{q}(t)$ is a vector of quantized variables which are quantized versions of the state variables $x(t)$. Each component of $\mathbf{q}(t)$ follows a piecewise constant trajectory, related with the corresponding component of $\mathbf{x}(t)$ by a hysteric quantization function so that:

$$q_j(t) = \begin{cases} x_j(t) & \text{if } |q_j(t^-) - x_j(t)| = \Delta Q_j \\ q_j(t^-) & \text{otherwise} \end{cases} \tag{3}$$

and $q_j(t_0) = x_j(t_0)$. Thus, $q_j(t)$ only changes when it differs from $x_j(t)$ in $\pm\Delta Q_j$. The magnitude ΔQ_j is called quantum.

B. QSS and Stiff Systems

The system

$$\begin{aligned} \dot{x}_1(t) &= 0.01\, x_2(t) \\ \dot{x}_2(t) &= -100\, x_1(t) - 100\, x_2(t) + 2020 \end{aligned} \tag{4}$$

has eigenvalues $\lambda_1 \approx -0.01$ and $\lambda_2 \approx -99.99$ which means that the system is stiff.

The QSS method approximates this system as

$$\begin{aligned} \dot{x}_1(t) &= 0.01\, q_2(t) \\ \dot{x}_2(t) &= -100\, q_1(t) - 100\, q_2(t) + 2020 \end{aligned} \tag{5}$$

Considering initial conditions $x_1(0) = 0$, $x_2(0) = 20$, and quanta $\Delta Q_1 = \Delta Q_2 = 1$, the QSS integration performs the following steps:

In $t = 0$ we set $q_1(0) = 0$ and $q_2(0) = 20$. Then, $\dot{x}_1(0) = 0.2$ and $\dot{x}_2(0) = 20$. This situation remains until $|q_i - x_i| = \Delta Q_i = 1$.

The next change in q_1 is then scheduled at $t = 1/0.2 = 5$ while the change in q_2 is scheduled at $t = 1/20 = 0.05$.

Thus, a new step is performed in $t = 0.05$. After this step it results $q_1(0.05) = 0$, $q_2(0.05) = 21$, $x_1(0.05) = 0.01$, $x_2(0.05) = 21$. The derivatives are $\dot{x}_1(0.05) = 0.21$ and $\dot{x}_2(0.05) = -80$.

The next change in q_1 is rescheduled at $0.05 + (1 - 0.01)/0.21 = 4.764$ while the next change in q_2 is scheduled at $0.05 + 1/80 = 0.0625$. Hence, the next step is performed at $t = 0.0625$.

In $t = 0.0625$ it results $q_1(0.0625) = 0$, $q_2(0.0625) = x_2(0.0625) = 20$, $x_1(0.0625) \approx 0.0126$ and the derivatives coincide with those of $t = 0$.

This is cyclically repeated until a change in q_1 occurs. That change occurs at $t \approx 4.95$, after 158 changes in q_2 (which oscillates between 20 and 21).

The simulation continues in the same way. Fig.1 shows the evolution of $q_1(t)$ and $q_2(t)$ through 500 units of simulated time.

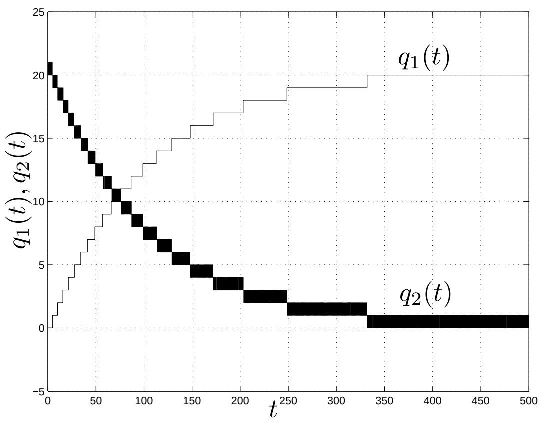

Figure 1: QSS Simulation

The fast oscillations of q_2 provoke a total of 15995 transitions in that variable, while q_1 only changes 21 times. Consequently, the total number of steps to complete the simulation is greater than 16000 (this number is of the order of the 25000 steps needed by Forward Euler method to obtain a stable result).

Evidently, the QSS method is unable to efficiently integrate System (4).

C. QSS2 Method

The second order QBI method uses first order quantization. As it is shown in Figure 2, a first order quantizer produces a piecewise linear output trajectory. Each section of that trajectory starts with the value and slope of the input and finishes when it differs from the input in ΔQ_i. A formal definition of a first order quantization function can be found in (Kofman, 2002).

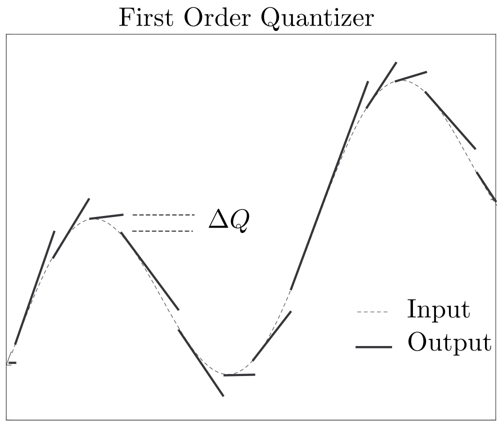

Figure 2: Trajectories of a first–order quantizer.

The QSS2 method then approximates a system like (1) by (2) but now, the quantized variables $q_i(t)$ follow piecewise linear trajectories and the state variables $x_i(t)$ are piecewise parabolic functions of the time.

Like the first order QSS, QSS2 can be represented by a DEVS model. The advantage of QSS2 is that it permits using a small quantum –i.e., setting a small error tolerance– without increasing considerably the number of calculations. In QSS, the number of steps is inversely proportional with the quantum, while in QSS2 it is only inversely proportional with its square root.

QSS2 also exhibits the problem of fast oscillations in the simulation of stiff systems. For instance, if we use the QSS2 method to simulate System (4), it performs 65467 steps (19 changes in q_1 and 65448 changes in q_2).

D. BQSS Method

The BQSS method is similar to QSS, but q_i is always chosen so that $x_i(t)$ goes to $q_i(t)$.

Basically, given a state variable $x_j(t)$, BQSS uses two hysteretic quantization functions: one from below ($\underline{q}_j(t) \leq x_j(t)$) and the other from above ($\overline{q}_j(t) \geq x_j(t)$). Both quantization functions are defined so that they never differ from x_j in more than $2\Delta Q_j$.

The quantized variable q_j is chosen equal to either $\underline{q}_j$ or $\overline{q}_j$, according to the direction of $\dot{x}_j(t)$. When $\dot{x}_j > 0$ we use $q_j = \overline{q}_j$, and viceversa.

When $\dot{x}_j = f_j(\mathbf{q}, \mathbf{u})$ depends on q_j, it could happen that the sign of the derivative changes when we evaluate f_j using each possibility, i.e., $f_j|_{\underline{q}_j} > 0$ and $f_j|_{\overline{q}_j} < 0$. Thus, we cannot find a correct value for q_j.

However, in that situation, continuity in f_j ensures that a value $\hat{q}_j$ exists, with $\underline{q}_j < \hat{q}_j < \overline{q}_j$, such that $f_j|_{\hat{q}_j} = 0$.

Thus, the BQSS method sets either $q_j = \overline{q}_j$ or $q_j = \underline{q}_j$ and enforces the derivative $\dot{x}_j = 0$ adding an extra perturbation term $\Delta f_j = f_j|_{q_j}$.

Then, given the system Eq.(1), instead of simulating a system like Eq.(2), BQSS simulates a system of the form:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{q}(t), \mathbf{u}(t)) + \Delta\mathbf{f}(t) \tag{6}$$

where $\Delta\mathbf{f}(t)$ is normally zero, except when the situation described above is found (i.e., when we cannot find a correct value for q_j).

BQSS works fine with most stiff systems. Anyway, the presence of the perturbation term $\Delta\mathbf{f}(t)$ increases the error and can cause the appearence of spurious equilibrium points in some nonlinear systems.

Another limitation of BQSS is that it is only first order accurate. We could not find, based on that idea, a second order accurate method.

E. DEVS Formalism

A DEVS model (Zeigler *et al.*, 2000) processes an input event trajectory, and, based on that trajectory and the initial state, produces an output event trajectory.

The behavior of an atomic DEVS model is formally defined by the structure:

$$M = (X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta) \tag{7}$$

where

- X is input event set, i.e., the set of all possible input event values.
- Y is the output event set.
- S is the state value set.
- δ_{int}, δ_{ext}, λ and ta are the functions that define the model dynamics.

Each possible state s ($s \in S$) has an associated time advance given by the *time advance function* ($ta(s) \rightarrow \mathbb{R}_0^+$). $ta(s)$ is a non–negative number indicating how long the system remains in a given state in absence of input events.

Then, if at time t_1 the state takes the value s_1, after $ta(s_1)$ time units (i.e., at time $t_1 + ta(s_1)$) the system performs an internal transition going to a new state $s_2 = \delta_{\text{int}}(s_1)$. Function δ_{int} ($\delta_{\text{int}} : S \rightarrow S$) is called *internal transition function.*

When the state goes from s_1 to s_2 an output event is produced, with value $y_1 = \lambda(s_1)$. Function λ ($\lambda : S \rightarrow Y$) is called *output function.* Functions ta, δ_{int} and λ defined the autonomous behavior of a DEVS model.

When an input event arrives, the state changes instantaneously. The new state depends not only on the input event value but also on the previous state and the elapsed time since the last transition. If the model arrives to state s_3 at t_3 and an input event arrives at time $t_3 + e$ with a value x_1, the new state is calculated as $s_4 = \delta_{\rm ext}(s_3, e, x_1)$ (notice that $e < ta(s_3)$). In this case, we say that the system performs an external transition. Function $\delta_{\rm ext}$ ($\delta_{\rm ext} : S \times \Re_0^+ \times X \to S$) is called *external transition function.* During an external transition no output event is produced.

DEVS models can be coupled and the result of the coupling defines an equivalent atomic DEVS model.

F. DEVS and QSS

Each component of Eq.(2) can be thought as the coupling of two elementary subsystems. A static one,

$$d_j(t) = f_j(q_1, \cdots, q_n, u_1, \cdots, u_m) \tag{8}$$

and a dynamical one

$$q_j(t) = Q_j(x_j(\cdot)) = Q_j(\int d_j(\tau)d\tau) \tag{9}$$

where Q_j is the hysteretic quantization function (it is not a function of the instantaneous value $x_j(t)$, but a functional of the trajectory $x_j(\cdot)$).

Since the components $u_j(t)$ and $q_j(t)$ are piecewise constant, the output of Subsystem (8), i.e., $d_j(t)$, will be piecewise constant. In this way, both subsystems have input and output piecewise constant trajectories that can be represented by sequences of events.

Then, Subsystems (8) and (9) define a relation between their input and output sequences of events. Consequently, equivalent DEVS models can be found for these systems, called *static functions* and *quantized integrators* respectively (Kofman and Junco, 2001).

The second order accurate QSS2 method can be implemented in the same way of QSS. However, the trajectories are now piecewise linear instead of piecewise constant. Thus, the events carry two numbers that indicate the initial value and the slope of each segment. Also, the static functions and quantized integrators are modified with respect to those of QSS so they can take into account the slopes.

III. LINEARLY IMPLICIT QSS METHODS

In this section, we introduce the new Linearly Implicit QSS (LIQSS) methods of first and second order and we discusses the implementation of them as DEVS models.

A. First Order LIQSS Method

The idea of LIQSS is very similar to BQSS. However, when a value q_j so that x_j goes to it cannot be found, LIQSS tries to find the value $\hat{q}_j$ for which $\dot{x}_j = 0$ instead of adding the perturbation term Δf_j to enforce that situation.

In order to illustrate this idea, we shall simulate System (4) from the same initial conditions and quantum than before.

In $t = 0$, we can choose $q_2 = 19$ or $q_2 = 21$ according to the sign of $\dot{x}_2(t)$. In both cases, $\dot{x}_1(0) > 0$ so the quantized future value of x_1 will be $q_1(0) = 1$. On the other hand, if we choose $q_2(0) = 21$ then $\dot{x}_2(0) = -180 < 0$ and if we choose $q_2(0) = 19$, it results $\dot{x}_2(0) = 20 > 0$ so there exists a point $19 < \hat{q}_2(0) < 21$ in which $\dot{x}_2(0) = 0$. The value for $\hat{q}_2(0)$ can be calculated (exploiting the linear dependence of $\dot{x}_2$ with q_2) as

$$\hat{q}_2(0) = 21 - \frac{-180}{-100} = 19.2$$

Then, the state derivatives result: $\dot{x}_1(0) = 0.192$, $\dot{x}_2(0) = 0$.

The next change in q_1 is scheduled for $t = 1/0.192 \approx 5.2083$ while the corresponding in q_2 is scheduled for $t = \infty$

Then, the next step takes place in $t = 1/0.192 \approx 5.2083$. At this time, $x_1 = 1$ and $x_2 = 0$. After that, $q_1(5.2083) = 2$ (because $\dot{x}_1(5.2083) > 0$). If we reevaluate $\dot{x}_2$ for $q_2 = 19$ and $q_2 = 21$ it results lower than zero in both cases, so the correct value is $q_2(5.2083) = 19$ because in this way x_2 goes to q_2.

With these values of q_1 and q_2 the following state derivatives are obtained: $\dot{x}_1(5.2083) = 0.19$ and $\dot{x}_2(5.2083) = -80$. The next change in q_1 is scheduled to $t = 1/0.192 + 1/0.19 \approx 10.47149$, while the one in q_2 is scheduled to $t = 1/0.192 + 1/80 \approx 5.22083$. So, the next step is given in $t = 5.22083$, when x_2 reaches q_2.

Calculations follow in this way. Fig.3 show the evolution of $q_1(t)$ and $q_2(t)$ through 500 units of simulated time. As it can be seen, in this method the fast oscil-

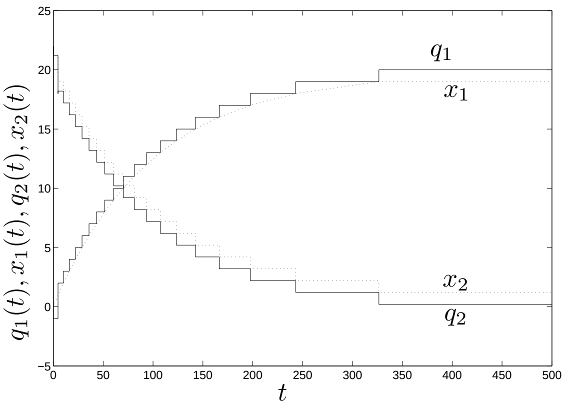

Figure 3: LIQSS Simulation

lations of q_2 are not present. In this way, q_1 changes 21 times and q_2 changes 25 times, which totalizes 46 steps (this constitutes a rather decent result for a stiff system).

B. LIQSS Definition

Given System (1), the LIQSS method approximates it by Eq.(2), where each q_j is defined by the following function

$$q_j(t) = \begin{cases} \underline{q}_j(t) \text{ if } f_j(\mathbf{q}(t),\mathbf{u}(t))(\underline{q}_j(t) - x_j(t)) \geq 0 \\ \overline{q}_j(t) \text{ if } f_j(\mathbf{q}(t),\mathbf{u}(t))(\overline{q}_j(t) - x_j(t)) \geq 0 \\ \quad \wedge f_j(\mathbf{q}(t),\mathbf{u}(t))(\underline{q}_j(t) - x_j(t)) < 0 \\ \widetilde{q}_j(t) \text{ otherwise} \end{cases} \tag{10}$$

with

$$\underline{q}_j(t) = \begin{cases} \underline{q}_j(t^-) - \Delta Q_j \\ \quad \text{if } x_j(t) - \underline{q}_j(t^-) \leq 0 \\ \underline{q}_j(t^-) + \Delta Q_j \\ \quad \text{if } x_j(t) - \underline{q}_j(t^-) \geq 2 \cdot \Delta Q_j \\ \underline{q}_j(t^-) \text{ otherwise} \end{cases} \tag{11}$$

$$\overline{q}_j(t) = \underline{q}_j(t) + 2\Delta Q_j \tag{12}$$

$$\widetilde{q}_j(t) = \begin{cases} \overline{q}_j(t) - \frac{1}{A_{jj}} \cdot f_j(\overline{\mathbf{q}}^j(t), \mathbf{u}(t)) \;\; \text{if } A_{jj} \neq 0 \\ q_j(t^-) \quad \text{otherwise} \end{cases} \tag{13}$$

where $\overline{\mathbf{q}}^j(t)$ is equal to $\mathbf{q}(t^-)$ except for the j–th component, where it is equal to $\overline{q}_j$ and $A_{j,j}$ is the j,j component of the Jacobian matrix evaluated in $\overline{\mathbf{q}}^j$, i.e.,

$$A_{jj} = \left. \frac{\partial f_j}{\partial x_j} \right|_{\overline{\mathbf{q}}^j, u(t^-)} \tag{14}$$

As we shall see now, when $A_{j,j} \neq 0$, setting $q_j = \widetilde{q}_j$ provokes (in the linear case) the situation $\dot{x}_j = 0$.

C. Calculation of $\widetilde{q}_j$

We take $\underline{\mathbf{q}}^j(t)$ equal to $\overline{\mathbf{q}}^j(t)$, except that the j–th component is $\underline{q}_j$. Notice that we use $\widetilde{q}_j$ when f_j changes the sign between $\underline{q}_j$ and $\overline{q}_j$. Thus, an intermediate point $\hat{q}_j$ exists where $f_j = 0$.

We take $\hat{\mathbf{q}}^j(t)$ equal to $\overline{\mathbf{q}}^j(t)$, except that the j–th component is $\hat{q}_j$.

Defining

$$A_j = \left. \frac{\partial f_j}{\partial x} \right|_{\overline{\mathbf{q}}^j, u(t^-)}$$

and the residual

$$g(\mathbf{x},\mathbf{u}) = f_j(\mathbf{x},\mathbf{u}) - A_j \mathbf{x},$$

calling $\hat{q}_j$ the point where $f_j = 0$, we can write

$$\begin{aligned} f_j(\overline{\mathbf{q}}^j, \mathbf{u}) &= A_j \overline{\mathbf{q}}^j - A_{j,j}\overline{q}_j + A_{j,j}\overline{q}_j + g(\overline{\mathbf{q}}^j, \mathbf{u}) \\ f_j(\underline{\mathbf{q}}^j, \mathbf{u}) &= A_j \underline{\mathbf{q}}^j - A_{j,j}\underline{q}_j + A_{j,j}\underline{q}_j + g(\underline{\mathbf{q}}^j, \mathbf{u}) \\ f_j(\hat{\mathbf{q}}^j, \mathbf{u}) &= A_j \hat{\mathbf{q}}^j - A_{j,j}\hat{q}_j + A_{j,j}\hat{q}_j + g(\hat{\mathbf{q}}^j, \mathbf{u}) \end{aligned}$$

Taking into account that $A_j\overline{\mathbf{q}}^j - A_{j,j}\overline{q}_j = A_j\hat{\mathbf{q}}^j - A_{j,j}\hat{q}_j$ (because $\overline{\mathbf{q}}^j$ and $\hat{\mathbf{q}}^j$ only differ in the j–th component) and considering that $f_j(\hat{\mathbf{q}}^j, \mathbf{u}) = 0$, we can solve the previous equations for $\hat{\mathbf{q}}^j$, obtaining

$$\hat{q}_j = \overline{q}_j - \frac{f_j(\overline{\mathbf{q}}^j, \mathbf{u})}{A_{j,j}} + \frac{g(\overline{\mathbf{q}}^j, \mathbf{u}) - g(\hat{\mathbf{q}}^j, \mathbf{u})}{A_{j,j}} \tag{15}$$

In order to estimate $A_{j,j}$ we can use the expression

$$A_{j,j} \approx \frac{f_j(\overline{\mathbf{q}}^j, \mathbf{u}) - f_j(\underline{\mathbf{q}}^j, \mathbf{u})}{\overline{q}_j - \underline{q}_j} \tag{16}$$

If f_j depends linearly on q_j, Eq.(16) gives the exact value of the Jacobian. Moreover, in that case the last term in Eq.(15) results zero and $\widetilde{q}_j = \hat{q}_j$, i.e., the value of $\widetilde{q}_j$ provokes $\dot{x}_j = 0$.

In a nonlinear case, we will have $\widetilde{q}_j \approx \hat{q}_j$, and $\dot{x}_j \approx 0$. Although the oscillations will not disappear in this case, they will have a low frequency.

This is analogous to what linearly implicit discrete time methods do by solving the implicit equation only for the linear part of the problem, and this is the reason for calling LIQSS to this method.

D. DEVS Implementation of LIQSS

The main difference between LIQSS and QSS is the way in which $\mathbf{q}$ is obtained from $\mathbf{x}$. Eq.(10) says that q_j not only depends on x_j but also on $\mathbf{q}$.

In QSS, the changes in q_j are produced when x_j differes from it in ΔQ_j. Similarly, in LIQSS q_j changes when x_j reaches q_j. However, changes in q_j can trigger changes in other quantized variables because of Eq.(10). In the same way, changes in some component of u_j can also change some quantized variable.

Following the idea of the DEVS implementation of QSS, i.e., by the coupling of *quantized integrators* and *static functions*, but taking into account the mentioned differences, we can obtain a DEVS model for the LIQSS algorithm.

The structure of this implementation is shown in Fig.4.

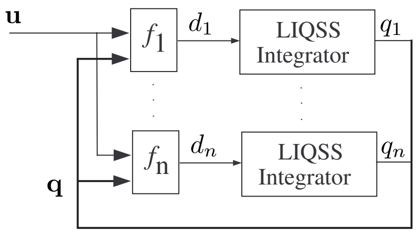

Figure 4: Block Diagram of LIQSS

Since the trajectories of $u_j(t)$ and $q_j(t)$ are piecewise contant, the static funcions are the same than those of QSS.

Then, we only need to define the LIQSS integrators so that they calculate q_j according to the definition of LIQSS.

In order to build the DEVS model of the LIQSS quantized integrator, we shall first analyze its behavior.

Let us suppose that in time t the state x_j reaches the value of q_j with a positive slope ($\dot{x}_j(t^-) > 0$). Then, the upper and lower quantization functions $\overline{q}_j$ and $\underline{q}_j$

must be updated (increasing them by ΔQ_j). In this situation we first try with an output value $q_j(t) = \overline{q}_j = q_j(t^-) + \Delta Q_j$.

If, due to the feedback, we receive an input event with $d_j = \dot{x}_j(t) < 0$, then we are in the situation where we need to calculate the value $\hat{q}_j$ that provokes $\dot{x}_j = 0$, i.e., we estimate $\tilde{q}_j$ using Eq.(13), that in this case takes the form

$$\widetilde{q}_j = \overline{q}_j - \frac{1}{A_{jj}} \cdot d_j$$

and we set $q_j(t) = \tilde{q}_j$ (supposing that $A_{j,j} \neq 0$, otherwise we just use $q_j = \overline{q}_j$).

The value of $A_{j,j}$ can be easily estimated using the values of $\dot{x}_j(t^-)$, $\dot{x}_j(t)$, $q_j(t^-)$ and $\overline{q}_j(t)$.

It could happen that the integrator then receives (also at time t) another event with a nonzero value (because of the error in the calculation of $\hat{q}_j$). In this case we shall not calculate any further value for q_j at time t.

In the opposite case (when x_j reaches q_j from above) we proceed in an analogous way.

The other case in which q_j changes and the quantized integrator must provoke output events is when a change in the sign of the derivative is received. Suppose that x_j was going up towards $q_j = \overline{q}_j$ and at time t (due to the change in some quantized variable or input), an event with $d_j = \dot{x}_j(t) < 0$ is received.

Then, the integrator must send the new output value $q_j(t) = \underline{q}_j$ so that x_j goes towards q_j. In this case, it can also happen that, due to the feedback, a new event with $\dot{x}_j < 0$ is received at time t and we are again in the situation where we need to calculate $\hat{q}_j$. In this case, we proceed exactly as before, calculating $\widetilde{q}_j$ and ignoring any further change of sign at time t.

As before, the case where x_j is initially going down to q_j is completely analogous to the one described above.

In any situation, after calculating q_j, it results easy to schedule the next output event time:

$$\sigma_j = \begin{cases} (\overline{q}_j(t) - x_j(t))/d_j & \text{if } d_j > 0 \\ (x_j - \underline{q}_j(t))/d_j & \text{if } d_j < 0 \\ \infty & \text{otherwise} \end{cases}$$

The behavior described for the quantized integrator can be easily translated into a DEVS model.

E. Second Order LIQSS: Basic Idea

The second order linearly implicit method, called LIQSS2, was developed combining the ideas of QSS2 and LIQSS.

The quantized variables of this new method are piecewise linear instead of being piecewise constant and they are chosen in order to verify $\ddot{x}_j \cdot (q_j - x_j) > 0$, i.e., so that x_j goes towards q_j.

As an example, in Figure 5 a general trajectory is shown following this idea.

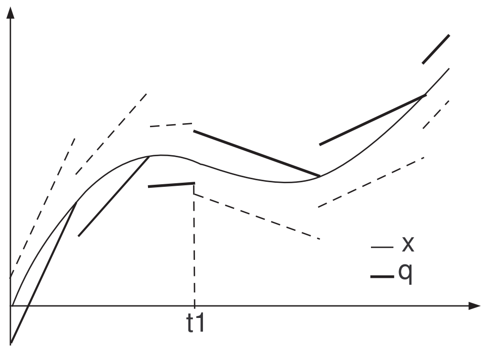

Figure 5: LIQSS2 Trajectories

Analogously to the case of LIQSS and BQSS, it can happen that the sign of $\ddot{x}_j$ changes when we start a new segment of $q_j(t)$. Then, an intermediate slope m_j exists that makes $\ddot{x}_j = 0$. In this case, we can also select the initial value q_j of the new segment so that $\dot{x}_j = m_j$, i.e., we can make the state trajectory to run parallel to the quantized trajectory so that no events are generated. Both values, q_j and m_j, can be easily obtained when $\dot{x}_j$ depends linearly on q_j.

If we simulate System (4) from the same initial conditions but with quanta $\Delta q_1 = 0.1$ and $\Delta q_2 = 0.1$ (10 times smaller than before), the LIQSS2 method only performs 59 steps (20 changes in q_1 and 39 in q_2). The simulation results can be seen in Figure 6

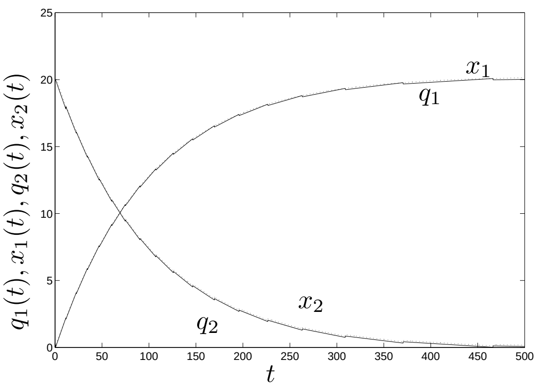

Figure 6: LIQSS2 Simulation

F. LIQSS2 Definition

Given the system (1), the LIQSS2 method approximates it by (2), where each component q_j is defined as:

$$q_j = \begin{cases} \overline{q}_j(t) & \text{if } \ddot{x}_j(t) > 0 \vee \\ & (\ddot{x}_j(t) = 0 \wedge \dot{x}_j(t) > m_j) \\ \underline{q}_j(t) & \text{if } \ddot{x}_j(t) < 0 \vee \\ & (\ddot{x}_j(t) = 0 \wedge \dot{x}_j(t) <= m_j) \\ \widetilde{q}_j(t) & \text{otherwise} \end{cases} \qquad (17)$$

with

$$\underline{q}_j(t) = \begin{cases} x_j(t_0) - \Delta Q_j \quad \text{if } t = t_0 \\ \underline{q}_i(t^-) + \Delta Q_j \\ \quad \text{if } (x_j(t) = \underline{q}_j(t^-) + 2\Delta Q_j \\ \underline{q}_i(t^-) - \Delta Q_j \\ \quad \text{if } (x_j(t) = \underline{q}_j(t^-) \\ \underline{q}_j(t_j) + m_j \cdot (t - t_j) \;\; \text{otherwise} \end{cases} \tag{18}$$

$$\overline{q}_j(t) = \underline{q}_j(t) + 2 \cdot \Delta Q_j \tag{19}$$

$$\widetilde{q}_j(t) = \begin{cases} \dfrac{m_j(t) - \ddot{x}(t^-)}{A_{j,j}} + q_j(t^-) \\ \quad \text{if } A_{j,j} \neq 0 \\ q_j(t^-) \quad \text{otherwise} \end{cases} \tag{20}$$

and

$$m_j = \begin{cases} m_j(t^-) \text{ if } q_j(t^-) = q_j(t) \\ \dot{x}_j(t^-) \text{ if } (\ddot{x}_j(t) \cdot \ddot{x}_j(t^-) > 0 \vee A_{j,j} = 0) \\ \quad \wedge q_j(t^-) \neq q_j(t) \\ m_j(t^-) - \dfrac{\ddot{x}_j(t^-)}{A_{j,j}} \text{ otherwise} \end{cases} \tag{21}$$

Note that in Eq.(21), the condition $\ddot{x}_j(t) \cdot \ddot{x}_j(t^-) < 0$ means that an intermediate value m_j exists so that $\ddot{x}_j = 0$. In a linear system this value can be calculated analogously to LIQSS with the expression $m_j(t) = m_j(t^-) - \ddot{x}_j(t^-)/A_{j,j}$. In a nonlinear case, we shall also obtain an approximate value.

G. DEVS Implementation of LIQSS2

The simulation scheme for LIQSS2 is the same than before (Fig.4), but now the trajectories are piecewise linear an parabolic.

Since the quantized variable trajectories of LIQSS2 are, as in QSS2, piecewise linear, the static functions are the same of QSS2.

The main difference between QSS2 and LIQSS2 is the way in which the quantized state variable trajectories are calculated in the integrator.

Each segment of the quantized variable trajectory can be characterized by an initial point q_j and a slope mq_j. In the same way, the state derivative can be characterized by the pair (d_j, md_j). Thus, each input and output event of the quantized integrator will carry two numbers.

Let us then analyze the behavior of the resulting DEVS model. Suppose that at time t the state x_j reaches either $\overline{q}_j$ or $\underline{q}_j$ with $\ddot{x}_j(t^-) > 0$. Using the fact that x_j is piecewise parabolic, we know both $\dot{x}_j(t^-)$ and $\ddot{x}_j(t^-)$. Thus, we set the new segments of $\overline{q}_j$ and $\underline{q}_j$ with slope $m_q = \dot{x}_j(t^-)$ and initial values $x_j + \Delta Q_j$ and $x_j - \Delta Q_j$ respectively. Then, as $\ddot{x}_j(t^-) > 0$ we select $q_j(t) = \overline{q}_j(t)$.

It could happen that, due to the feedback, at time t we then receive an event with slope $md_j(t) = \ddot{x}_j(t) < 0$. Thus, an intermediate output slope $\hat{m}q_j$ between the old and the new one exists that provokes the situation $md_j = \ddot{x}_j = 0$. If $A_{j,j} \neq 0$, this slope can be estimated as

$$\hat{m}q_j(t) \approx \widetilde{m}q_j(t) = mq_j(t^-) - \frac{\ddot{x}(t^-)}{A_{j,j}} \tag{22}$$

We also calculate the value $\hat{q}_j(t)$ that makes $md_j = \dot{x}_j(t) = \hat{m}q_j(t)$:

$$\hat{q}_j(t) \approx \widetilde{q}_j(t) = \frac{\widetilde{m}q_j(t) - \ddot{x}(t^-)}{A_{j,j}} + x_j(t^-)$$

Thus, we update the slopes of $\underline{q}_j$ and $\overline{q}_j$ to the value $\widetilde{m}q_j$ and we output an event with the pair $(\widetilde{q}_j, \widetilde{m}q_j$.

Like the case of LIQSS, if we receive another event at time t we do not produce any further output event.

Another situation in which q_j changes is when an event with a change in the sign of the second derivative is received. Suppose that x_j was moving towards q_j with $\ddot{x}_j > 0$ and at time t an event is received with $md_j = \ddot{x}_j < 0$ (due to the change is some other quantized or input variable).

Thus, we first try with $q_j = \underline{q}_j(t)$ and we update the slope $mq_j = \dot{x}_j(t^-)$. If, due to feedback, we receive another event at time t with $md_j > 0$, we must calculate the value $\hat{m}q_j$ that makes $md_j = 0$, and we repeat what we did from Eq.(22).

Once q_j is calculated, we must schedule the next event time. The time to the next event is given by the first crossing of x_j with either $\overline{q}_j$ or $\underline{q}_j$. This can be calculated as the minimum positive solution σ_j of the following equations

$$\begin{aligned} x_j(t) + \dot{x}_j(t) \cdot \sigma_j + \frac{1}{2}\ddot{x}_j(t)\sigma_j^2 &= \overline{q}_j \\ x_j(t) + \dot{x}_j(t) \cdot \sigma_j + \frac{1}{2}\ddot{x}_j(t)\sigma_j^2 &= \underline{q}_j \end{aligned}$$

Similarly to the case of LIQSS, $A_{j,j}$ can be estimated as:

$$A_{j,j}(t) = \frac{md_j(t^-) - md_j}{mq_j(t^-) - mq_j(t)}$$

All these ideas can be easily translated into the DEVS model of the LIQSS2 quantized integrator.

H. PowerDEVS Implementation

PowerDEVS (Pagliero *et al.*, 2003) is a software tool for DEVS simulation. It has a graphical editor that permits building block diagrams of DEVS models. PowerDEVS libraries contain all the blocks needed to implement the QSS methods, including quantized integrators, static functions, source terms and blocks for discontinuity handling.

The DEVS models corresponding to both LIQSS methods were added to the generic quantized integrator that implementes the QSS methods. Now, this block permits selecting among the following methods: QSS, QSS2, QSS3, BQSS, CQSS, LIQSS and LIQSS2.

This block also permits selecting the quantum and the initial state value.

Section V.illustrates the use of these new methods in PowerDEVS.

IV. THEORETICAL PROPERTIES

We shall treat here the most important properties of the LIQSS methods. We shall show first that the methods perform a finite number of steps in a finite interval of time (this guarantees that the simulation time will always advance). Then we shall analyze the stability and accuracy properties.

A. Trajectories and Legitimacy of LIQSS

A crucial requirement of QSS methods is the legitimacy condition, which ensures that a finite number of events occurs in any finite interval of time. The following theorem proves this property for the first order LIQSS method.

Theorem 1. *Suppose that function* $\mathbf{f}$ *in* (1) *is bounded in a domain* $D \times D_u$*, where* $D \subset \Re^n$*,* $D_u \subset \Re^m$ *and assume that the trajectory* $\mathbf{u}(t) \in D_u$ *is piecewise constant. Then,*

1. *Any solution* $\mathbf{x}(t)$ *of* (2) *is continuous while* $\mathbf{q}(t)$ *remains in* D.
2. *The trajectory* $\mathbf{q}(t)$ *is piecewise constant while it remains in* D.

Proof. The proof of (1) is straightforward since, according to (2), the derivative of $\mathbf{x}$ is bounded.

For the item (2), in order to prove that $\mathbf{q}$ is piecewise constant it is necessary to ensure that it only experiences a finite number of changes in any finite interval of time.

Let (t_1, t_2) be an arbitrary interval of time in which $\mathbf{q}(t)$ remains in D. We shall prove that, within this interval, $\mathbf{q}(t)$ has a finite number of changes.

The assumptions of the theorem ensure that $\mathbf{f}(\mathbf{q}, \mathbf{u})$ is bounded and, taking into account the relation between x_j and q_j, positive constants $\bar{f}_j$ exist so that, for $t \in (t_1, t_2)$

$$|\dot{x}_j(t)| \leq \bar{f}_j; \text{ for } j = 1, \ldots, n.$$

Let $t_c \in (t_1, t_2)$ and suppose that $\overline{q}_j(t_c^-) \neq \overline{q}_j(t_c^+)$. According to (12), this situation cannot be repeated until $|x_j(t) - x_j(t_c)| \geq \Delta Q_j$. Thus, the minimum time interval between two discontinuities in $\overline{q}_j(t)$ is

$$t_j = \frac{\Delta Q_j}{\bar{f}_j}$$

Then, calling $\overline{n}_j$ the number of changes of $\overline{q}_j(t)$ in the interval (t_1, t_2), it results that

$$\overline{n}_j \leq (t_2 - t_1)\frac{\bar{f}_j}{\Delta Q_j}$$

Since $\mathbf{u}(t)$ is piecewise constant, it will perform a finite number of changes n_u in the interval (t_1, t_2).

The definition of q_j ensures that it can only change when $\overline{q}_j(t)$ changes or when there is a change in some other quantized or input variable ($q_i(t)$ or $u_i(t)$) that inverts the sign of $\dot{x}_j$.

In conclusion, changes in $q_j(t)$ are linked to changes in some $\overline{q}_i(t)$ or $u_i(t)$. Thus, the total number of changes will be equal or less than the sum of all the changes in those variables, i.e.,

$$n_j \leq n_u + (t_2 - t_1)\sum_{i=1}^{n}\frac{\bar{f}_i}{\Delta Q_i}$$

which is a finite number. □

Although this theorem is only valid for the first order LIQSS method, an analog result for LIQSS2 can be obtained combining this proof with that of QSS2 (Kofman, 2002).

B. Perturbed representation

The theorical properties of the QSS methods are based in a perturbed representation of the original system (1) that is equivalent to the approximation Eq.(2). Defining $\Delta\mathbf{x}(t) = \mathbf{q}(t) - \mathbf{x}(t)$ each row of System (2) can be rewritten as:

$$\dot{x}_i = f_i(\mathbf{x}(t) + \mathbf{\Delta x}(t), \mathbf{u}(t)) \qquad (23)$$

From (10), (11), (12) and (13) in the definition, it can be ensured that each component $\Delta x_i(t)$ of $\mathbf{\Delta x}(t)$ is bounded by[1]

$$|\Delta x_i(t)| \leq 2 \cdot \Delta Q_i \qquad (24)$$

where ΔQ_i is the quantization adopted for $x_j(t)$. Thus, the LIQSS methods simulate an approximate system which only differ from the original SES(1) due to the presence of the bounded state perturbation.

C. Global Error Bound and Stability

Given the LTI system

$$\dot{\mathbf{x}}_a(t) = A\mathbf{x}_a(t) + B\mathbf{u}(t) \qquad (25)$$

were A is a Hurwitz matrix with Jordan canonical form $\Lambda = V^{-1}AV$, the LIQSS(2) approximation simulates the system

$$\dot{\mathbf{x}} = A(\mathbf{x}(t) + \Delta\mathbf{x}(t)) + B\mathbf{u}(t) \qquad (26)$$

Defining the error as $e(t) = x(t) - x_a(t)$, and following the procedure of (Kofman, 2002) and (Cellier and Kofman, 2006), it results that

$$|e(t)| \leq |V||\mathbb{R}\text{e}(\Lambda)^{-1}\Lambda||V^{-1}|2\Delta Q \qquad (27)$$

where ΔQ is the vector of quantum adopted.

The error bound is twice the error bound of QSS, QSS2 and QSS3.

[1]The symbols $|\cdot|$ and "$\leq$" denote the componentwise module and inequallity, respectively.

D. Equilibrium Points

One of the drawbacks of BQSS was the appearence of spurious equilibrium points. Due to the term $\Delta\mathbf{f}$, Eq.(6) admits equilibrium points also when $\mathbf{f}(\mathbf{q},\mathbf{u}) \neq 0$.

However, the only possibility in which LIQSS or LIQSS2 arrive to an equilibrium point is when $\mathbf{f}(\mathbf{q},\mathbf{u}) = 0$, i.e., when the quantized variables reach an equilibrium point.

V. EXAMPLES

This section is aimed to introduce some examples which show the advantages of the LIQSS methods.

A. Second Order Linear System

In Section II the following system was presented:

$$\begin{aligned} \dot{x}_1 &= 0.01x_2 \\ \dot{x}_2 &= -100x_1 - 100x_2 + 2020 \end{aligned} \qquad (28)$$

with initial conditions $x_1(0) = 0$ and $x_2(0) = 20$

The system was first simulated using LIQSS and LIQSS2 methods with quantum $\Delta Q_i = 1$. Then, the simulations were repeated decreasing the quantum 10, 100 and 1000 times. The following table shows the number of steps performed for each method using the mentioned quantization:

ΔQ_i	LIQSS1			LIQSS2		
	N°x_1	N°x_2	Total	N°x_1	N°x_2	Total
1	21	25	46	8	17	24
0.1	201	203	404	20	39	59
0.01	2006	2026	4032	60	126	186
0.001	20064	28174	48238	186	391	577

This table shows that the number of steps performed by LIQSS linearly varies with the quantization, while the number of steps in LIQSS2 grows approximately with the square root of the quantum reduction.

Figure 7 shows the simulation results using Simulink ODE15s with tolerance 10^{-3} and PowerDEVS with $\Delta Q_i = 0.001$ (the difference between both methods cannot be appreciated with the naked eye).

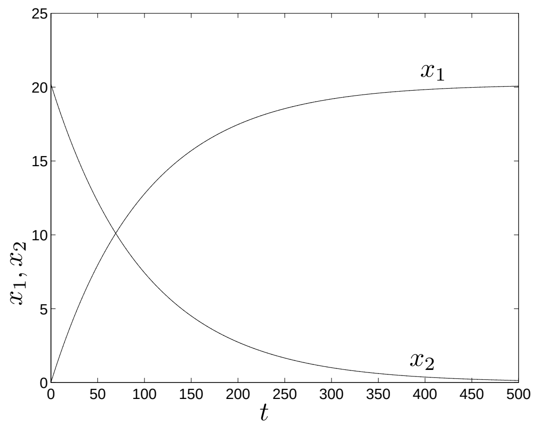

Figure 7: Linear Stiff System Simulation

The simulation time could not be evaluated under PowerDEVS using $\Delta Q_i = 0.001$ (it was too small in order to be accurately measure). Thus, we compared the simulation times using quantum $\Delta Q_i = 0.0001$. The simulation in Simulink takes 0.078 seconds (ODE15s, in accelerated mode, with tolerance 10^{-3}), while in PowerDEVS it takes 0.015 seconds.

The global error bound –Eq.(27)– ensures that the error in the LIQSS2 simulation is always less than $2 \cdot 10^{-4}$ in x_1 and $6 \cdot 10^{-4}$ in x_2.

Figure (8) shows the simulation error in PowerDEVS with quanta $\Delta Q_i = 0.0001$. Comparing it with the theoretical bound, we see that the last one is a bit conservative.

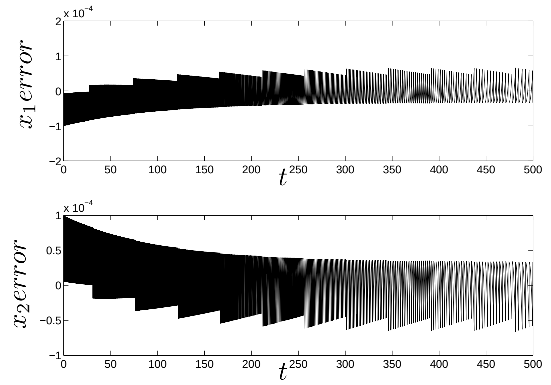

Figure 8: Error

B. Van der Pol Oscillator

The problem consists of a second order differential equation proposed by B. van der Pol in the 1920's, that describes the behavior of nonlinear vacuum tube circuits. It has two periodic solutions, the constant solution, $z(t) = 0$, which is unstable, and a nontrivial periodic solution that corresponds to an attractive limit cycle. The equation depends on a parameter that weights the importance of the nonlinear part of the equation. The corresponding state equations are:

$$\begin{aligned} \dot{x}_1(t) &= x_2 \\ \dot{x}_2(t) &= (1 - x_1^2) \cdot \mu - x_1 \end{aligned} \qquad (29)$$

We fixed the parameter $\mu = 1000$, what gives rise to a stiff problem that is often used as a test problem for stiff ODEs solvers (Enright and Pryce, 1987).

The model was then built in PowerDEVS (Fig.9)

For the simulation, we used initial conditions $x_1(0) = 2$, $x_2(0) = 0$ and quantization $\Delta Q_1 = 0.001$, $\Delta Q_2 = 1$. We simulated the system with LIQSS2 until $t_f = 4000$. The results are shown in Figure 10. The total number of steps was 2159 (838 in x_1 and 1321 in x_2). The simulation takes 0.031 seconds.

The same system was simulated using different Matlab/Simulink methods. The best results were obtained with ODE15s. In order to obtain similar results to the ones given by PowerDEVS the tolerance error must be set to 10^{-14}. Using larger tolerances the results were qualitatively similar but with a significant phase error. In this case, the number of steps was 2697 and the simulation takes 0.056 seconds.

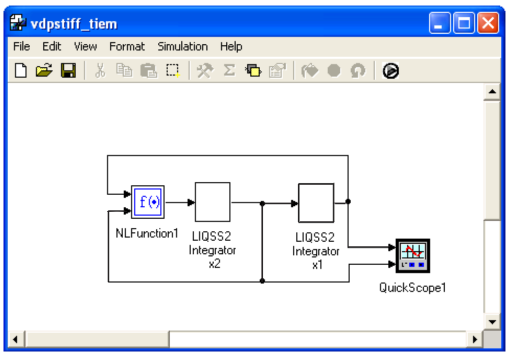

Figure 9: PowerDEVS model of the Van der Pol oscillator

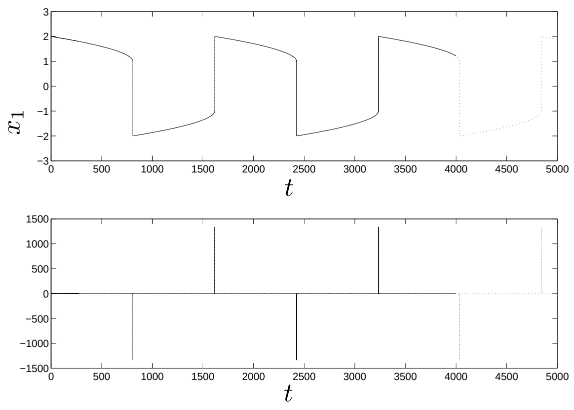

Figure 10: Van der Pol oscillator simulation

It must be taken into account as we compare the results, that the order of LIQSS2 is smaller that the one of ODE15s.

Finally, the system was simulated again using LIQSS2 but with quanta $\Delta Q_1 = 0.0001$ and $\Delta Q_2 = 0.1$ (ten times smaller). The number of steps performed result 4148(1975 in x_1 and 2173 in x_2) and the simulation takes 0.047 seconds. Comparing this result with the previous one, it can be seen that the number of steps was increased just in a factor of 2 while the quantized levels were reduced in a factor of ten. This put in evidence the second order nature of the LIQS2 method

VI. CONCLUSIONS

We presented two novel QBI methods that are able to integrate stiff systems based on linearly implicit principles. The fact that the methods do not call for iterations permits achieving an important reduction of computational costs when compared with traditional implicit discrete time methods.

We showed that these methods, satisfy the global error bound property of QSS method.

We showed that these methods, satisfy the global error bound property of QSS method.

The methods improve the performance of BQSS by solving the problem of the spurious equilibrium points and by increasing to 2 the order of accuracy.

Future work must be done in order to develop higher order methods (following the idea of QSS3 for instance). It is also important to establish which kind of stiff system can be efficiently simulated with LIQSS methods.

REFERENCES

Cellier, F. and E. Kofman, *Continuous System Simulation*, Springer, New York (2006).

Enright, W. H. and J. D. Pryce, “Two (fortran) packages for assessing initial value methods,” *(ACM) Transactions on Mathematical Software,* **13**(1), 1–27 (1987).

Hairer, E., S. Norsett, and G. Wanner, *Solving Ordinary Differential Equations I. Nonstiff Problems*, Springer, 2nd edition (1993).

Hairer, E. and G. Wanner, *Solving Ordinary Differential Equations II. Stiff and Differential–Algebraic Problems*, Springer, 1st edition (1991).

Kofman, E., “A Second Order Approximation for DEVS Simulation of Continuous Systems,” *Simulation,* **78**(2), 76–89 (2002).

Kofman, E., “Discrete Event Simulation of Hybrid Systems,” *SIAM Journal on Scientific Computing,* **25**(5), 1771–1797 (2004).

Kofman, E., “A Third Order Discrete Event Simulation Method for Continuous System Simulation,” *Latin American Applied Research,* **36**(2), 101–108 (2006).

Kofman, E. and S. Junco, “Quantized State Systems. A DEVS Approach for Continuous System Simulation,” *Transactions of SCS,* **18**(3), 123–132 (2001).

Migoni, G., E. Kofman, and F. Cellier, “Integración por Cuantificación de Sistemas Stiff,” *Revista Iberoam. de Autom. e Inf. Industrial,* **4**(3), 97–106 (2007).

Pagliero, E., M. Lapadula, and E. Kofman, “PowerDEVS. Una Herramienta Integrada de Simulación por Eventos Discretos,” *Proceedings of RPIC’03,*, San Nicolas, Argentina **1**, 316–321 (2003).

Zeigler, B., T. Kim, and H. Praehofer, *Theory of Modeling and Simulation*, Academic Press, New York, 2nd edition (2000).