Improving Quantized-State System Simulation

Mehrdad Niknami

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2020-23
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-23.html

May 1, 2020

Improving Quantized-State System Simulation

Mehrdad Niknami[1]

[1]University of California, Berkeley

Abstract

We discuss and improve prior formulations of Quantized-State System simulation, an alternate class of algorithms to the traditional time-stepping algorithms used for the numerical solution of initial-value problems. We illustrate that the computation of higher-order derivatives in an initial-value problem can be performed in multiple ways, and we present improvements that allow QSS solutions to more closely match analytic solutions. We also show that the problem of *backward* QSS (BQSS) can be ill-posed, and we propose modifications to existing methods to allow for well-defined solutions. Finally, we discuss the implications of these approaches on the efficiency of a solver, and the trade-offs they naturally impose on the result.

1 Introduction

Traditional numerical algorithms for solving initial-value problems use a time-stepping approach: they discretize the time variable and, at each iteration, solve for the system state vector at a later time step considering the state vectors at prior time steps. By contrast, a more recent class of algorithms due to Kofman and Junco [4], known as *quantized-state system* (QSS) simulation, uses a *state*-stepping approach: these algorithms take the system state vector $\vec{x}$, *quantize* it into a vector $\vec{q}$ consisting of low-order Taylor polynomial approximations at the current step, and, at each iteration, solve for the *time* at which the *continuous approximation* x_i of each state component would deviate from the *quantized approximation* q_i of that state component by a threshold called the *quantum*.

This approach is unique in *relaxing* the normally-unchallenged assumption that all components of the state vector are to be updated in lockstep. In QSS, this constraint is relaxed to allow each state component to advance in time at its own natural pace, allowing for potentially improved accuracy, stability, or performance. [1]

This approach turns out to have significant fundamental advantages compared to classical time-stepping methods, some more obvious than others. The most obvious benefit is the fact that merely aggregating together two systems into a single combined (yet decouplable) system no longer automatically forces the ideal rate of simulation to slow down to the minimum of the individual ones, as decoupled state components advance completely independently.

However, this is far from the only advantage of QSS. In fact, QSS has been shown to possess a more extraordinary property: namely, it has been proven (Kofman [2]) that, in the particular case of asymptotically stable LTI systems, QSS's *global* error is bounded by an *constant* that is *independent* of the length of the simulation. This is in stark contrast to (say) an algorithm such

as Euler's method, whose error can grow without bound even when simulating an asymptotically *stable* LTI system. Figure 1 illustrates a simple example of this phenomenon.

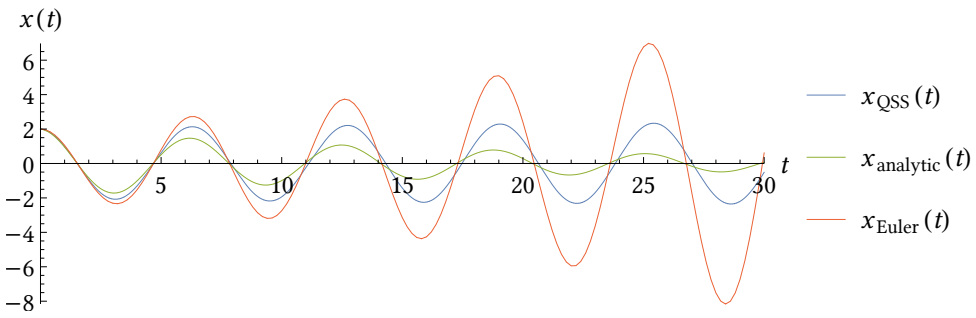


Figure 1: The QSS1, analytic, and Euler solutions for a sample system ($\ddot{x}(t) = -\frac{\dot{x}(t)}{10} - x(t)$, $\dot{x}(0) = -\frac{1}{5}$, $x(0) = 2$) with quantum $Q = \frac{1}{5}$ for QSS and time step $\Delta t = \frac{1}{5}$ for Euler. We observe that the analytic solution converges, and the QSS1 solution maintains bounded global error, in stark contrast with the Euler solution, which diverges.

Just as there is no free lunch, however, the QSS approach is not without its trade-offs. For example, at least in the case of first-order QSS (QSS1), although the error is *upper*-bounded by a function of the quantum, it is also trivially *lower*-bounded by the quantum itself, meaning that the method naturally cannot resolve granularities under one quantum. This issue, however, can be addressed by simply restricting the size of the time-step along with the quantum, effectively forcing steps at least as frequent as those of the Euler method to be taken. Other challenges, however, can be more difficult to overcome, and will thus comprise the foci of this paper.

2 Assumptions

We consider the usual nonlinear initial-value problem given by the ordinary differential equation $\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t))$ describing an autonomous dynamical system over time with known initial state $\vec{x}(0)$. As our goal is to study the simulation of physical systems modeled in this manner, we will implicitly assume $\vec{f}$ is reasonably well-behaved in the relevant portions of the problem domain.

3 Forward Quantized-State System Simulation

As with the forward and backward Euler methods, one can attempt to apply the QSS method in a *forward* or *backward* manner. We visit each of these in turn.

3.1 QSS1

First-order forward *quantized-state system* (QSS1) simulation approximates a dynamical system using a zero-order hold, by assuming that each state variable follows a piecewise-constant *quantized* trajectory q_i that "remembers" the value of the corresponding continuous state x_i until x_i deviates

by a *quantum* Q_i:

$$\dot{\vec{x}}(t) = \vec{f}(\vec{q}(t)) \tag{1}$$

$$q_i(t) = \begin{cases} x_i(t) & \text{if } \left|x_i(t^-) - q_i(t^-)\right| \geq Q_i \\ q_i(t^-) & \text{otherwise} \end{cases} \tag{2}$$

Here, $\vec{q}$ is a vector with the same dimension as $\vec{x}$, q_i and x_i are the i^{th} elements of vectors $\vec{q}$ and $\vec{x}$, and t^- denotes the instant before time t (i.e. $q_i(t^-) = \lim_{\tau \to t^-} q_i(\tau)$). We call $\vec{x}$ the *continuous approximation* to the system (as it is continuous), and we similarly call $\vec{q}$ the *quantized approximation* to the system (it is piecewise-continuous). Whenever x_i deviates from q_i by a quantum Q_i, causing $q_i(t)$ to be reset to $x_i(t)$, we say that a *quantization event* has occurred.

We help visualize the behavior of a QSS solution, we illustrate the QSS1 approximation to a simple system in figure 2.

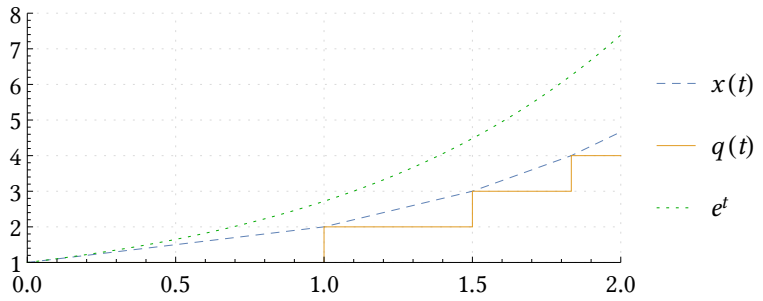
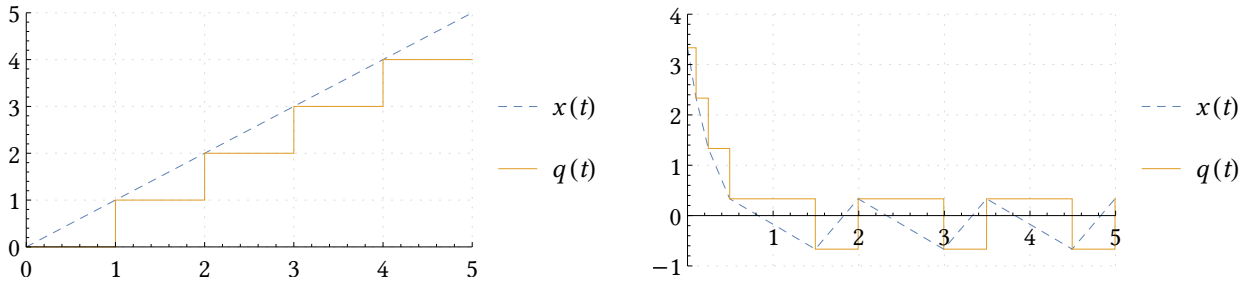


Figure 2: Illustration of the QSS1 solution to the system $\dot{x}(t) = x(t)$ with initial condition $x(0) = 1$ and quantum $Q = 1$. (Note the error bound does not apply to unstable systems.)

As perhaps expected, the QSS1 method is far from perfect. In particular, the QSS solution can be inefficient for some trivial systems. For example:

- $\dot{x}(t) = 1$, which is a simple feed-forward system. A QSS1 solver with $Q = 1$ would iterate through the steps $x(0) = 0$, $x(1) = 1$, $x(2) = 2$, ... despite the fact that the system is completely trivial to pre-solve exactly.
- $\dot{x}(t) = -x$, which is a basic decaying exponential. Because a QSS1 solver cannot take steps smaller than one quantum Q (i.e. the quantum is its natural "resolution limit"), it produces a solution that oscillates back and forth between $x(0) \,\%\, Q$ and $x(0) \,\%\, Q - Q$ indefinitely, unless $x(0)$ happens to be an integer multiple of Q. (Here, % is the modulo, or remainder, operator.)

These efficiency concerns are illustrated in figure 3.



(a) $\dot{x}(t) = 1$, $x(0) = 1$ (b) $\dot{x}(t) = -3\,x(t)$, $x(0) = 3 + \frac{1}{3}$

Figure 3: QSS1 solutions of two simple systems. In both cases, QSS1 continues to process quantization events indefinitely despite the eventually-linear trajectory of the system state, an undesirable source of inefficiency in the model.

One way to attempt to address these issues is to relax the restriction that q be piecewise-constant by generalizing QSS1 to a higher-order "QSS2" method that allows q to be piecewise-*linear*. This would naturally us to represent a linear state trajectory with a *single* quantization event, realizing a significant performance improvement.

It is not immediately clear, however, how exactly to generalize this formulation to higher-order methods such as QSS2 [2] or QSS3 [3]. We still need to obtain $\vec{x}$ by integrating $\dot{\vec{x}}$, which is not a problem for QSS1 since $\vec{q}$ is by definition piecewise-constant and therefore $\vec{x}$ is piecewise-linear. However, if we allow $\vec{q}$ to be even piecewise-linear, then $\vec{x}$ will not be piecewise-quadratic (or anything else noteworthy in general) for nonlinear $\vec{f}$.

Let us make the discrete-event nature of QSS1 explicit to help us generalize QSS1 to a higher-order method. Notice that, if $\frac{\partial \vec{f}}{\partial \vec{q}}$ is the Jacobian of $\vec{f}$, then $\frac{\partial f_i}{\partial q_j} = 0$ (i.e., $\frac{\partial f_i}{\partial q_j}$ is *identically zero*) iff f_i lacks dependence on q_j. With this in mind, we now rewrite QSS1 in a more explicit but equivalent manner:

$$x_i(t) = x_i(\tau_{x_i}(t)) + f_i(\vec{q}(\tau_{x_i}(t))) \cdot (t - \tau_{x_i}(t)) \tag{3}$$

$$q_i(t) = x_i(\tau_{q_i}(t)) \tag{4}$$

$$\tau_{q_i}(t) = \begin{cases} t & \text{if } \left|x_i(t^-) - q_i(t^-)\right| \geq Q_i \\ \tau_{q_i}(t^-) & \text{otherwise} \end{cases} \tag{5}$$

$$\tau_{x_i}(t) = \max_j \left\{ \tau_{q_j}(t) : \frac{\partial f_i}{\partial q_j} \neq 0 \right\} \tag{6}$$

Here, we have defined $\tau_{q_i}(t)$ to be the latest time before t at which q_i was quantized, and $\tau_{x_i}(t)$ to be the latest time before t at which x_i was affected by the quantization of any dependent q_j. This reformulation makes the discrete-event nature of the model clear, since it is now apparent that "events" in the simulation only occur when a component of τ_x or τ_q undergoes a change. (Quantization events, therefore, would refer to changes in a component of τ_q.)

3.2 Higher-Order QSS

We posit that the natural generalization of QSS1 to QSSd would be to calculate when a degree-d Taylor polynomial approximation of x_i deviates from a degree-$d-1$ Taylor polynomial approxima-

tion of q_i by one quantum.

In taking such an approach, Kofman formulates QSS2 as follows [2]:

$$\dot{x}_i(t) = f_i(\vec{q}(t)) \tag{7}$$

$$q_i(t) = \begin{cases} x_i(t) & \text{if } t = 0 \vee \left|q_i(t^-) - x(t^-)\right| \geq Q_i \\ q_i(\tau_{q_i}(t)) + (t - \tau_{q_i}(t))\, \dot{x}_i(\tau_{q_i}(t^-)) & \text{otherwise} \end{cases} \tag{8}$$

$$\dot{x}_i(\tau_{q_i}(0^-)) = 0 \tag{9}$$

This formulation is then used to prove similar theoretical error bounds as for QSS1. Unlike in QSS1—in which the the piecewise-*constancy* of all q_i made the expression $f_i(\vec{q}(t))$ piecewise-constant and thus trivial to integrate—the piecewise-*linearity* of all q_i in QSS2 is insufficient to guarantee anything about the integrability of $f_i(\vec{q}(t))$ for general nonlinear f_i. Therefore, for simulation purposes, Kofman *approximates* nonlinear QSS2 as *linear* QSS2 by first linearizing f_i at each step to make it integrable, resulting in piecewise-*quadratic* trajectories for all x_i. (Hereafter, we use "QSS2" to refer to this simulable approximation of actual QSS2.)

Kofman's formulation of QSS2, however, can exhibit behavior unfaithful to the original system.

Notice that we have $\dot{q}_i(t) = \dot{x}_i(\tau_{q_i}(t^-))$ between quantization events, while at every quantization, we have $q_i(t) = x_i(t)$. This means that the derivative $\dot{x}_i$ from *before* each quantization event to be used along with the value of x_i *after* the quantization, causing the derivative to be obtained from a state that *lags behind the current time*. Moreover, this lag also requires a valid derivative to be specified at the initial time $t = 0$, which in Kofman's formulation has been (somewhat arbitrarily) assumed to be zero. As may be expected, this timing inconsistency can introduce a discrepancy between the simulated and expected solutions, which we will illustrate shortly.

While perhaps seemingly unnecessary at first glance, however, such a lag can be reasonable and difficult to avoid in practical circumstances. For example, it can be difficult to avoid such a lag when the function $\vec{f}$ is not known explicitly. (This can occur when a piece of software is used to compute $\vec{f}$ at each desired value of $\vec{q}$, as in such scenarios it can be impractical to compute associated functions such as $\vec{\nabla}\vec{f}$ efficiently.) However, if $\vec{f}$ is given in a suitable form (such as via an analytic formula), we no longer have this constraint. In that case, we can use the chain rule to directly differentiate the original ODE $\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t))$ with respect to t as needed, evaluating the desired derivatives directly at the current time.

One trade-off of evaluating the derivatives directly, however, is that it can introduce new dependencies into the system, making it potentially less sparse and thus slightly less efficient to evaluate. We will observe an example of this trade-off below in equation 11. However, this decrease in the sparsity can be be quite worthwhile, as it allows us to obtain a simulation result that is inherently a closer match the analytic solution.

Therefore, formally, we put forward the following approximation for QSSd, removing the lag as

mentioned above:

$$x_i(t) = x_i(\tau_{x_i}(t)) + \sum_{k=1}^{d} \hat{x}_i^{(k)}(\tau_{x_i}(t), t) \qquad \text{(continuous trajectory)}$$

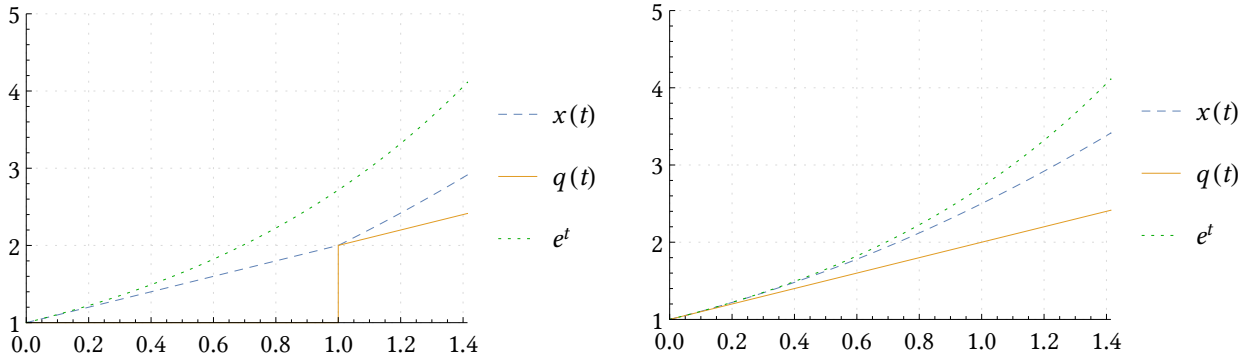
$$q_i(t) = q_i(\tau_{q_i}(t)) + \sum_{k=1}^{d-1} \hat{x}_i^{(k)}(\tau_{q_i}(t), t) \qquad \text{(quantized trajectory)}$$

$$\hat{x}_i^{(k)}(t^*, t) = \frac{(t-t^*)^k}{k!} x_i^{(k)}(t^*) \qquad \text{(Taylor extrapolation from time } t^* \text{ to } t)$$

$$x_i^{(k)}(t) = \frac{d^{k-1}}{dt^{k-1}} f_i(\vec{q}(t)) \qquad \text{(derivatives via eq. 1)}$$

And as a sanity check, we note that, as expected, the summations for $q_i(t)$ vanish when $d = 1$, reducing the system to 3 and 4 for QSS1.

It is not difficult to find a system that demonstrates the differences between these two formulations of QSS2. Indeed, even the trivial scalar system $\dot{x}(t) = x(t)$, $x(0) = 1$ illustrates the effect of the lag and its removal. We show this effect in figure 4.



(a) Kofman QSS2 formulation.
For $t \to 0^+$, $x(t) = t$ and $q(t) = 1 + 0t$.
As $t \to 1^+$, $q(t) = q(1) + \dot{x}(1^-)\, t = 1 + t$.

(b) New QSS2 formulation.
For $t \to 0^+$, $x(t) = 1 + t + t^2/2$, $q(t) = 1 + t$.
As $t \to \sqrt{2}^{\,+}$, $q(t) = -\sqrt{2} + (2 + \sqrt{2})t$.

Figure 4: Comparison of QSS2 formulations for the system $\dot{x}(t) = x(t)$, $x(0) = 1$ with the analytic solution e^t. The new formulation uses *fresher* derivatives, resulting in a better approximation to the true exponential solution than would be obtained via the use of stale (lagging) derivatives.

Notice that the derivative discrepancy begins at the very first moment at $\dot{q}(0) = 0$ and compounds thereafter. Moreover, artificially forcing the initialization of $\dot{q}(0) = 1$ would only resolve the discrepancy at $t = 0$, not afterward: $q(t)$ would still lag behind $\dot{q}(t) = \dot{x}(1^-)$, and the same inconsistency would occur at the subsequent quantization. We can consequently see that eliminating this lag allows the new QSS2 formulation to give a better approximation of the exponential solution than the original formulation. But, of course, it requires that $\vec{f}$ be known and that we be able to analytically differentiate it. When these conditions are not satisfied, we can fall back to the Kofman formulation.

3.3 Quantization vs. Direct Evaluation of State Derivatives

It may be tempting to believe that there is an easier way to resolve the aforementioned discrepancy. After all, the problem we encountered was that $\dot{q}$ was out-of-sync with q. Could we not have resolved this issue by simply updating $\dot{q}$ as soon as q is quantized?

While possibly less obvious, this approach would not resolved the problem. The reason, simply put, is that updating $\dot{q}$ only during quantization events is insufficient to ensure that $\dot{q}$ tracks $\vec{f}(q)$ in general.

We can illustrate this with a concrete counterexample. Consider the initial QSS2 trajectories of the following system starting at $\vec{x}(0) = \vec{0}$ with quantum $Q = 1$:

Differential eq.	Continuous approx. $\vec{x}$	Quantized approx. $\vec{q}$
$\dot{x}_1(t) = 1$	$x_1(t) = t$	$q_1(t) = t$
$\dot{x}_2(t) = 1 + x_1(t)$	$x_2(t) = t + t^2/2$	$q_2(t) = t$
$\dot{x}_3(t) = 1 + x_2(t)$	$x_3(t) = t + t^2/2$	$q_3(t) = t$
$\dot{x}_4(t) = 1 + 2x_3(t)$	$x_4(t) = t + t^2$	$q_4(t) = t$

As usual, the continuous and quantized approximations differ by one Taylor term. Thus, quantizing $\vec{x}$ to get $\vec{q}$, we observe that, since $|x_4 - q_4|$ grows the fastest, the first quantization event is the one that occurs for q_4, at $t = 1$. At this point, we attempt to evaluate $\ddot{x}_4(1)$ via the two approaches, and notice the difference between them:

$$\ddot{x}_4(1) = 0 + 2\dot{q}_3(1) = 2 \qquad \text{(Kofman approach)} \tag{10}$$

$$\neq 0 + 2\underbrace{\big(1 + q_2(1)\big)}_{\dot{x}_3(1)} = 4 \qquad \text{(our approach)} \tag{11}$$

Notice that, as there are no preceding quantization events, the value of $\dot{q}_3(1)$ is simply $\frac{d}{dt}t\Big|_{t=1} = 1$. We can therefore see that, at the cost of introducing a dependency on q_2 (which makes the system less sparse), our approach gains the ability to produce a result closer to the analytic solution[1] by fundamentally utilizing up-to-date values of upstream dependencies instead of attempting to extrapolate them based on outdated information.

4 Backward Quantized-State System Simulation

In a "forward" simulation algorithm, we approximate the evolution of the system state at every step as a function of the system state at the *beginning* of that step. Correspondingly, in a "backward" algorithm, we approximate the evolution of the system state at every step as a function of the system state at the *end* of the step. Given that the classical *forward Euler* simulation method has an analogous *backward Euler* counterpart that is more suitable for stiff systems, one hopes for a *backward QSS* method analogous to *forward QSS* with similar properties.

[1] $\ddot{x}_4(1) = 5$, because $\vec{x}(t) = \left[t \;\; t + \frac{t^2}{2} \;\; t + \frac{t^2}{2} + \frac{t^3}{6} \;\; t + t^2 + \frac{t^3}{3} + \frac{t^4}{12}\right]^\top$.

To address this, *backward QSS*1 (BQSS) has been previously defined by Migoni et al. [5] as follows:

$$\dot{x}_i(t) = \begin{cases} f_i(\vec{q}(t)) & \text{if } f_i(\vec{q}(t)) \cdot (q_i(t) - x_i(t)) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

$$q_i(t) \in \left\{ \overline{q}_i(t), \underline{q}_i(t) \right\} \tag{13}$$
$$\text{such that } f_i(\vec{q}(t)) \cdot (q_i(t) - x_i(t)) > 0$$
$$\text{if uniquely possible, or}$$
$$q_i(t) = q_i(t^-) \text{ otherwise} \tag{14}$$

$$\overline{q}_i(t) = \begin{cases} \overline{q}_i(t^-) + Q_i & \text{if } \overline{q}(t^-) \leq x_i(t) \\ \overline{q}_i(t^-) - Q_i & \text{if } \overline{q}(t^-) \geq x_i(t) + Q_i + \epsilon_i \\ \overline{q}_i(t^-) & \text{otherwise} \end{cases} \tag{15}$$

$$\underline{q}_i(t) = \begin{cases} \underline{q}_i(t^-) - Q_i & \text{if } \underline{q}(t^-) \geq x_i(t) \\ \underline{q}_i(t^-) + Q_i & \text{if } \underline{q}(t^-) \leq x_i(t) + Q_i + \epsilon_i \\ \underline{q}_i(t^-) & \text{otherwise} \end{cases} \tag{16}$$

The idea of this formulation is that, whereas in forward QSS x_i starts at q_i and *diverges* from q_i until the two differ by one quantum, in backward QSS, x_i would be allowed to start one quantum *away* from q_i and subsequently evolve *toward* it. In other words, q_i represents the anticipated *future* value of x_i. In this formulation, $\overline{q}_i(t)$, $\underline{q}_i(t)$, and $q(t^-)$ represent possible values of at the *end* of the current time step for q_i, corresponding to q_i increasing, decreasing, or staying constant, respectively. The constant ϵ_i is called the *hysteresis width* of the i^{th} component, perhaps typically 1% of Q, but is not of significance to our discussion.

Unfortunately, this proposed formulation of BQSS is not without its drawbacks. We explore and attempt to address some of these difficulties below.

4.1 Performance Implications of BQSS

Recall that the fundamental principle of QSS—whether forward or backward—is that the quantized state q_i is assumed to follow a polynomial trajectory (i.e. a constant trajectory for QSS1), until—and unless—the continuous state x_i deviates from it by at least one quantum Q_i.

The aforementioned formulation of BQSS, however, does not quite follow this behavior: q_i can suddenly change long before x_i deviates from it by a quantum. This is a direct consequence of the fact that any change in the sign of $f_i(\vec{q}(t))$ can trigger an instantaneous change in the value of $q_i(t)$ via equation 13. Such premature quantizations can cause the model to undergo far more quantization events than desirable from a backward algorithm, potentially decreasing simulation performance.

Migoni et al. [5], however, do not investigate what would happen if we simply avoided updating $q_i(t)$ in response to sign changes in $f_i(\vec{q}(t))$. Would this resolve the issue?

Unfortunately, this method would not work: imposing such a hysteresis constraint can render the model *illegitimate*—that is, the model would no longer be guaranteed to only complete a finite number of steps in any finite time interval. Intuitively, the illegitimacy is due to the fact that when

x_i changes its trajectory to move *away* from q_i (recall that x_i can change while q_i is held constant until the deviation exceeds one quantum), as soon as the value of $q_i(t)$ is updated, the trajectory of x_i can reverse, causing q_i to immediately revert to its prior value. Figure 5 shows a concrete counterexample illustrating this behavior. As we can see, one seems to have little choice but to update q_i in response to changes in $\dot{x}_i(t)$.

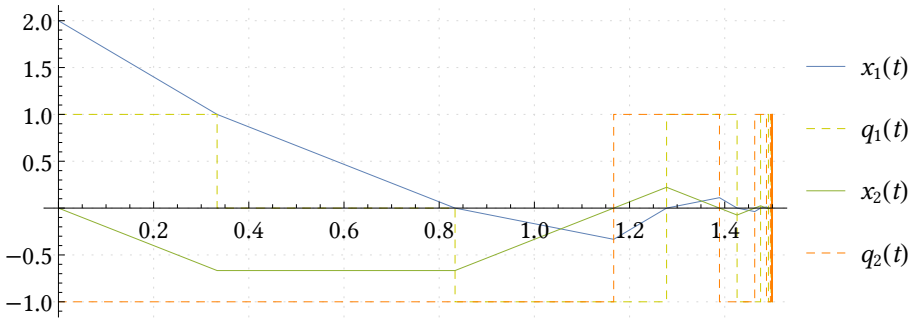


Figure 5: An illegitimate model caused by enforcing hysteresis in the system $\dot{x}_1(t) = -x_1(t)+2\,x_2(t)$, $\dot{x}_2(t) = -2\,x_1(t)$ with the initial condition $x_1(0) = 2$, $x_2(0) = 0$ and quantum $Q = 1$.

4.2 Consistency and Dynamic Quantum Reduction in BQSS

A potentially greater concern than the aforementioned performance issue is the fact that the BQSS formulation can actually introduce causality loops in the evaluation of the model: namely, while $\dot{x}_i$ can trigger a change to q_i, a change in q_i may itself result in reverting change in $\dot{x}_i$. Indeed, even the trivial single-dimensional LTI system

$$\begin{aligned}\dot{x}(t) &= -x(t)\\ x(0) &= Q/2\end{aligned}$$

exhibits a causality loop (recall q is the *future* value of x, i.e. the value *toward* which x evolves):

$$\begin{aligned}
& q(0) \in \{Q/2, 3Q/2\} && \\
\implies\ & \dot{x}(0) \in \{-Q/2, -3Q/2\} && \\
\implies\ & \dot{x}(0) < 0 && \\
\implies\ & q(0) < x(0) && \text{(negative } \dot{x} \text{ means } x \text{ is decreasing)}\\
\implies\ & q(0) \in \{-Q/2\} && \text{(contradiction)}
\end{aligned}$$

Notice, therefore, that this results in an endless update cycle that would never terminate.

Migoni et al. [5] sidestep this problem by providing a greedy algorithm that *forces* $\dot{x}(t) = 0$ while keeping $q(t) = q(t^-)$ whenever there are zero or multiple solutions found for q. This is justified on the grounds that, by the intermediate value theorem, if the derivative is changing sign, then there must exist a nearby value (within Q) of q for which $\dot{x}$ is in fact 0.

The obvious drawback of this approach, is that x would not actually evolve toward q, but would rather be forced to stay constant, making the BQSS approximation inconsistent with (albeit a close approximation to) the actual system.

In an attempt to resolve this inconsistency, we propose, instead, that the quanta $\vec{Q}$ be dynamically reduced by the *maximum* scalar factor $\lambda \leq 1$ that admits a consistent solution, because a sufficiently small choice of λ will necessarily avoid flipping the sign of $\dot{x}_i$ and therefore avoid triggering the inconsistency. (We choose a uniform scaling factor λ across all states rather than allowing for different λ_i, as the latter choice may no longer result in a unique solution.) In effect, this results in a premature quantization, which in general would maintain the solution accuracy.

We can illustrate this on a less trivial system, such as the following:

$$\begin{aligned} \dot{x}_1(t) &= 0 & x_1(0) &= 1 \\ \dot{x}_2(t) &= x_3(t) - 0.5\, x_1(t) & x_2(0) &= 1 \\ \dot{x}_3(t) &= -x_2(t) + 0.5\, x_1(t) & x_3(0) &= 0 \end{aligned}$$

A similar examination as before reveals that this system has no consistent solution vector $\vec{q}(0)$ that $\vec{x}$ can evolve toward for the quantum $Q = 1$. Therefore, we lower the quanta by a factor λ by solving the following optimization problem:

$$\begin{aligned} \vec{q}(0) &= \arg\max_{\vec{q}} \ \lambda \ : \ \vec{q} = \vec{x}(0) + \lambda \,\mathrm{diag}(\mathrm{sgn}(\vec{f}(\vec{q}))\, \vec{Q} \\ &= \arg\max_{\vec{q}} \ \lambda \ : \ \vec{q} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \lambda \,\mathrm{diag}\left(\mathrm{sgn}\left(\begin{bmatrix} 0 & 0 & 0 \\ -0.5 & 0 & +1 \\ +0.5 & -1 & 0 \end{bmatrix} \vec{q} \right) \right) \vec{Q} \end{aligned} \tag{17}$$

$$\therefore \ \vec{q}(0) = \begin{bmatrix} 1 \\ 0.5 \\ 0 \end{bmatrix} \implies \dot{\vec{x}}(0) = \vec{f}(\vec{q}(0)) = \begin{bmatrix} 0 \\ -0.5 \\ 0 \end{bmatrix}$$

In this problem, we attempt to maximize λ subject to the constraint that $\vec{x}$ evolves toward $\vec{q}$, as dictated by the sign of $\dot{\vec{x}}$. This optimization problem can be easily solved for λ via bisection if we are provided an efficient method for computing $\vec{q}(0)$ (and detecting infeasibility) given a choice for λ.

It is not obvious, however, how one may implement such a solver efficiently, due to the exponential number of possibilities (3^m, where m is the number of variables being quantized) that may arise for $\vec{q}(t)$.

To implement such a solver, we exploit the *monotonicity* of the simulator: namely, the fact that, in general, prematurely quantizing a state variable preserves—if not betters—the accuracy of the solution. Therefore, when multiple state variables require a simultaneous backward solution, we can perform a *backward Euler* (BE) step on these states simultaneously, searching (e.g. via bisection) for the maximum time step prior to the next quantization time of any other variable that would limit the deviation to one quantum.

While perhaps seem less than ideal, the use of a BE step here provides us the benefit of being able to progress time efficiently while simultaneously ensuring consistency with the original system. Moreover, and quite crucially, just as in forward QSS, the BE steps are only taken over the state variables that are actually undergoing a quantization, making it far cheaper than a full-sized step in the classical full-fledged BE algorithm.

5 Conclusion and Future Work

We have presented improvements for quantized-state system simulation to improve the quality of the resulting solutions. Specifically, we have shown how to remove a lag in the prior formulation of QSS2 and obtain more accurate simulations at the cost of a small increase in system density. We have also proposed methods to tackle shortcomings in backward QSS, providing a hybrid BQSS algorithm based on the backward Euler method that ensures the solution's consistency.

In spite of these, however, there is considerable room for further work on QSS, both from theoretical and practical standpoints. Many aspect of the simulation may admit significant potential improvements, ranging from parallelization of the discrete-event simulation model to the development of a more intelligent adaptive quantum adjustment algorithm. We hope that our contributions can help pave the way toward a comprehensive development of this elegant class of alternative simulation algorithms.

References

[1] François E Cellier and Ernesto Kofman. 2006. *Continuous system simulation*. Springer Science & Business Media.

[2] Ernesto Kofman. 2002. A second-order approximation for DEVS simulation of continuous systems. *Simulation* 78, 2 (2002), 76–89.

[3] Ernesto Kofman. 2006. A third order discrete event method for continuous system simulation. *Latin American applied research* 36, 2 (2006), 101–108.

[4] Ernesto Kofman and Sergio Junco. 2001. Quantized-state systems: a DEVS Approach for continuous system simulation. *Transactions of The Society for Modeling and Simulation International* 18, 3 (2001), 123–132.

[5] Gustavo Migoni, Ernesto Kofman, and François Cellier. 2012. Quantization-based new integration methods for stiff ordinary differential equations. *Simulation* 88, 4 (2012), 387–407.