Quantized State System Methods for Stochastic Differential Equations

Noelia Pizzi[1,3*], Mariana Bergonzi[1,2], Joaquín Fernandez[1], Damián Marelli[4,1], Ernesto Kofman[1,2*]

[1*]French-Argentine International Center for Information and System Sciences (CIFASIS), CONICET, Argentina.
[2]FCEIA, Universidad Nacional de Rosario, Argentina.
[3]FCByF, Universidad Nacional de Rosario, Argentina.
[4]School of Automation, Guandong University of Technology, Guangzhou, China.

*Corresponding author(s). E-mail(s): pizzi@cifasis-conicet.gov.ar; kofman@cifasis-conicet.gov.ar;
Contributing authors: bergonzi@cifasis-conicet.gov.ar; fernandez@cifasis-conicet.gov.ar; marelli@cifasis-conicet.gov.ar;

Abstract

This work explores the use of Quantized State Systems (QSS) methods for the simulation of Stochastic Differential Equations (SDEs). To that purpose, an extension of these algorithms is proposed wherein the governing Wiener process is sampled at regular intervals, while the states are updated asynchronously when they satisfy the threshold conditions corresponding to the respective QSS method. We show that the resulting schemes produce trajectories that converge to the actual solutions of the SDEs as the sampling interval $\boldsymbol{h}$ and the quantum $\boldsymbol{\Delta Q}$ approach zero. Moreover, we prove that, in stable linear time-invariant cases, the expected norm of the error is globally bounded by a linear function of the quantum $\boldsymbol{\Delta Q}$ and the sampling interval $\boldsymbol{h}$, demonstrating that the proposed scheme preserves practical stability regardless of the choice of these parameters. We also present simulation experiments that illustrate some potential advantages of the scheme.

Keywords: Stochastic Differential Equations, Numerical Integration, Quantized State System Methods

1 Introduction

Quantized State Systems (QSS) methods are a family of numerical integration algorithms for ordinary differential equations (ODEs) that replace time discretization with the quantization of the state variables (Kofman and Junco 2001; Cellier and Kofman 2006). This approximation results in an asynchronous discrete-event simulation scheme that, in some cases, offers advantages over classic numerical ODE solvers.

The basic idea of QSS methods is to update the state variables individually when they undergo a significant change. Thus, in systems that exhibit *heterogeneous activity*, i.e., when some states experience large variations while others remain mostly constant, the QSS algorithm can exploit this feature by performing calculations only on the states that actually change. In addition, the asynchronous nature of these methods simplifies the problem of discontinuity handling.

These characteristics of QSS methods make them particularly suitable for several applications, including the simulation of power electronic converters Migoni et al. (2015), advection-diffusion-reaction equations (Bergero et al. 2016), and spiking neural networks (Bergonzi et al. 2023), among others. In these applications, simulations based on QSS may run more than one order of magnitude faster than the most efficient classic ODE solvers.

While QSS algorithms have been used beyond ODEs to simulate Delay Differential Equations (Castro et al. 2024) and Differential Algebraic Equations (Kofman 2003), they have never been extended for the simulation of Stochastic Differential Equations (SDEs).

In this work, we propose a way of approximating SDEs using QSS algorithms, and we study the convergence, stability, and error bound properties of the resulting approximations. We also introduce a simulation example that illustrates the potential advantage of using QSS over classic algorithms in the presence of white noise.

The article is organized as follows: Section 2 provides an introduction to Quantized State Systems methods and their main properties for the simulation of ODEs. Section 3 introduces the main results, including the proposed QSS algorithms for SDEs and their convergence and stability properties. Then, Section 4 presents and analyzes simulation results, and finally, Section 5 concludes the work.

Notation

Throughout this work, the symbol $\preceq$ denotes the element-wise inequality between two vectors; that is, for $\alpha, \beta \in \mathbb{R}^n$, $\alpha \preceq \beta$ if and only if $\alpha_i \leq \beta_i$ for $i = 1, \ldots, n$. Besides, for a matrix M with components $M_{1,1}, \ldots, M_{n,m}$, $|M|$ indicates a matrix whose components are $|M_{1,1}|, \ldots, |M_{n,m}|$. Also, we use the notation $||.||$ to indicate the norm 2 of a vector or matrix.

2 Background

In this section, we introduce the family of QSS algorithms for the simulation of ODEs and their main properties.

2.1 Quantized State System Methods

Consider a continuous-time system of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector and $\mathbf{u}(t) \in \mathbb{R}^m$ is a known trajectory. The first-order Quantized State System (QSS1) method (Kofman and Junco 2001) solves an approximate ODE called Quantized State System:

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

where $\mathbf{q}(t) \in \mathbb{R}^n$ is the *quantized state* vector and $\mathbf{v}(t) \in \mathbb{R}^m$ is a piecewise constant approximation of $\mathbf{u}(t)$.

Each component of the quantized state $q_i(t)$ follows a piecewise constant trajectory that only changes when its difference with the corresponding state $x_i(t)$ reaches a constant value ΔQ_i called *quantum*. That is, the quantized state trajectory is related to the corresponding state trajectory $x_i(t)$ as follows:

$$q_i(t) = \begin{cases} q_i(t_k) & \text{if } |x_i(t) - q_i(t_k)| < \Delta Q_i \\ x_i(t) & \text{otherwise} \end{cases}$$

for $t_k < t \leq t_{k+1}$, where t_{k+1} is the first time after t_k at which $|x_i(t) - q_i(t_k)| = \Delta Q_i$. In addition, we consider that initially $\mathbf{q}(t_0) = \mathbf{x}(t_0)$.

Since the trajectories of the quantized state $q_i(t)$ and those of $v_j(t)$ are piecewise constant, the state derivatives $\dot{x}_i(t)$ also follow piecewise constant trajectories, and consequently, the states $x_i(t)$ follow piecewise linear trajectories. This piecewise linear nature allows for the exact analytical calculation of the time required for a state x_i to change by a quantum ΔQ_i, avoiding numerical integration errors in the prediction of the next event time. Owing to the particular form of the trajectories, the numerical solution of Eq. (2) is straightforward and can be easily translated into a simple simulation algorithm.

The idea of QSS1 was also extended to higher order methods (QSS2 and QSS3) that share the definition of Eq. (2), but where the quantized states follow piecewise linear and parabolic trajectories, respectively. There are also linearly implicit methods (LIQSS1, LIQSS2, and LIQSS3) that work efficiently for certain classes of stiff systems. A recent summary of the different QSS methods and their main applications can be found in (Castro et al. 2024). Figure 1 shows the trajectories generated using the QSS1 and QSS2 methods.

2.2 Convergence, Error Bounds, and Stability Properties of QSS Methods

The main properties of the QSS approximations are derived from the fact that the difference between a state $x_i(t)$ and the corresponding quantized state $q_i(t)$ in Eq. (2) is bounded by the quantum ΔQ_i.

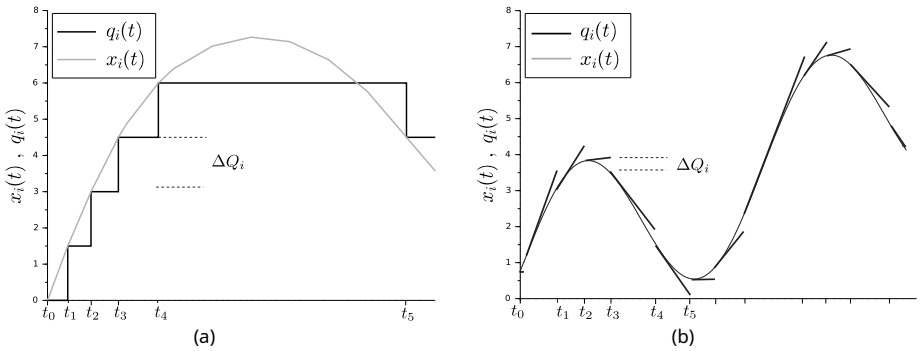


Fig. 1 Typical QSS1 (a) and QSS2 (b) state and quantized state trajectories.

Defining $\Delta\mathbf{x}(t) \triangleq \mathbf{q}(t) - \mathbf{x}(t)$ and $\Delta\mathbf{u}(t) \triangleq \mathbf{v}(t) - \mathbf{u}(t)$, Eq. (2) can be rewritten as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t) + \Delta\mathbf{x}(t), \mathbf{u}(t) + \Delta\mathbf{u}(t)) \tag{3}$$

so the only difference between the original ODE of Eq. (1) and the QSS approximation is the presence of the *disturbances* $\Delta\mathbf{x}(t)$ and $\Delta\mathbf{u}(t)$. As already mentioned, $\Delta\mathbf{x}(t)$ is component-wise bounded by the corresponding quantum $\Delta x_i(t) \leq \Delta Q_i$. Similarly, it is assumed that $\Delta\mathbf{u}(t)$ is bounded by some parameters $\Delta u_j(t) \leq \Delta U_j$.

Consequently, it is relatively easy to prove that the solutions of the QSS approximation converge to the solutions of the original ODE when the quantum and the input approximation bound approach zero (Kofman and Junco 2001). Furthermore, in linear time-invariant (LTI) systems, there exists a global error bound between these solutions that depends linearly on the quantum and the approximation bound of the input. The existence of this error bound also implies that the QSS solutions preserve *practical stability*; i.e., if the solutions of the ODE converge to an equilibrium point as $t \to \infty$, then the solutions of the QSS approximation remain inside a set around that point (Cellier and Kofman 2006).

2.3 Implementation of QSS Methods

The algorithm for QSS simulation keeps track of the values of the states x_i, quantized states q_i, and state derivatives $\dot{x}_i$. For each state x_i, the algorithm also stores the time t_i^{next} at which the quantized state q_i must be updated.

With that information, the algorithm advances the simulation time to $t_k = \min_i(t_i^{\text{next}})$, updates the corresponding state x_k and the quantized state q_k. Then, for all the states x_i whose derivatives depend on q_k, the algorithm recomputes the state derivatives $\dot{x}_i$ and recomputes the times t_i^{next} that have changed due to the change in q_k. Then, it advances the simulation time again according to the next quantized state change.

There is a tool called *Stand-Alone QSS Solver* (Fernández and Kofman 2014) that allows defining the models using a subset of the Modelica language and automatically generates and executes the simulation code for the different QSS methods. This tool

also generates simulation code for classic ODE solvers like DASSL, CVODE, and DOPRI, allowing a fair comparison among the performance of the different algorithms.

2.4 Related Work

To the best of our knowledge, quantization-based integration techniques have never been used to approximate Stochastic Differential Equations. There is an adaptive step-size algorithm based on Euler-Maruyama for discontinuous SDEs proposed in (Malik 2021) that resembles some principles of QSS methods. In addition, QSS methods have been widely used for systems that exhibit stochastic behavior (Assar and Sherman 2013; Pilch et al. 2018; Soto-Francés et al. 2020; Bergonzi et al. 2023), but in all cases, the randomness is related to the occurrence of certain events and not to the presence of a Wiener process.

QSS methods work in an asynchronous way, where every state variable is updated at different times. Thus, they have some features in common with multirate algorithms that have been applied to SDEs (Abdulle and de Souza 2022).

3 Main Results

In this section, we present the methodology developed to obtain approximated solutions of SDEs using QSS methods, and we study the convergence and stability properties of the resulting approximations.

3.1 QSS approximation of SDE models

We consider a continuous-time system given by the following stochastic differential equation:

$$\mathrm{d}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), t)\mathrm{d}t + \mathbf{g}(\mathbf{x}(t), t)\mathrm{d}\mathbf{w}(t); \quad t \in [0, T]; \quad \mathbf{x}(0) = \mathbf{x}_0 \tag{4}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T : [0, T] \to \mathbb{R}^n$ is the unknown continuous stochastic process, $\mathbf{f} : \mathbb{R}^n \times [0, T] \to \mathbb{R}^n$ is the drift coefficient, $\mathbf{g} : \mathbb{R}^n \times [0, T] \to \mathbb{R}^{n \times m}$ is the diffusion coefficient, and $\mathbf{w} = [w_1, w_2, \ldots, w_m]^T : [0, T] \to \mathbb{R}^m$ is an m-dimensional standard Wiener process with incremental covariance $I^{m \times m}\mathrm{d}t$.

The proposed approximation is based on sampling the process $\mathbf{w}(t)$ at times $t_k \triangleq kh$ with a constant sampling period $h = t_{k+1} - t_k$ and computing the random increments as

$$\Delta\mathbf{w}(t_k) \triangleq \mathbf{w}(t_{k+1}) - \mathbf{w}(t_k). \tag{5}$$

Then, the QSS1 approximation for the system of Eq. (4) is given by the following ODE:

$$\dot{\mathbf{x}}(t) = \mathbf{f}^q(\mathbf{q}(t), t) + \mathbf{g}(\mathbf{x}(t_k), t_k)\frac{\Delta\mathbf{w}(t_k)}{h}; \quad t \in [t_k, t_{k+1}); \quad \mathbf{x}(0) = \mathbf{x}_0 \tag{6}$$

for $k = 0, 1, \ldots$. Here, $\mathbf{q}(t)$ is the quantized state computed as in the QSS1 method for ODEs, and the function $\mathbf{f}^q(\mathbf{q}, t)$ is a piecewise constant approximation of $\mathbf{f}(\mathbf{q}, t)$. Note that in the case of multiplicative noise, where the diffusion coefficient $\mathbf{g}$ is a function of

the state $\mathbf{x}$, the term $\mathbf{g}(\mathbf{x}(t_k), t_k)$ is evaluated using the state value at the beginning of the sampling interval and then held constant for the duration of the interval $[t_k, t_{k+1})$.

In practice, the components of the random increment $\Delta\mathbf{w}(t_k)$ can be computed from discrete-time Gaussian processes $z_i(t_k) \sim \mathcal{N}(0,1)$ as

$$\Delta w_i(t_k) = z_i(t_k)\sqrt{h}. \tag{7}$$

The algorithm for simulating an SDE with the QSS1 method works like the one for the simulation of ODEs with an input term $\mathbf{u}(t_k) \triangleq \mathbf{g}(\mathbf{x}(t_k), t_k)\frac{\Delta\mathbf{w}(t_k)}{h}$ that is periodically updated. The following pseudo-code describes it:

Algorithm 1 QSS1 Algorithm for System (4).

```
// inputs: t_i (initial time), t_f (final time), x_0 (initial
    state), ΔQ (quantum vector), h (noise sample period)
// algorithm variables:
// t is the current simulation time
// x is the continuous state vector computed by the algorithm
// q is the quantized state vector
// ẋ is the state derivatives vector
// Δw is the random increment vector
// u is input term g(x(t_k), t_k) Δw(t_k)/h
// t_j^η is the time of the next change of quantized state q_j
// t_j^x is the time of the last change of continuous state x_j
// t_k is the last noise sample time
// Initialization
t = t_i // initial time ti
x = x_0 // initial state
q = x // initial quantized state
Δw = √h · N(0, I^{m×m}) //  vector of random increment
u = g(x, t) · Δw/h // initial input vector
for each j ∈ [1, n]
  ẋ_j = f_j(q, t) + u_j // compute j-th initial state derivative
  t_j^η = t + ΔQ_j/|ẋ_j| // compute the time of the next change in
      the j-th quantized state
  t_j^x = t // time of the last change in the j-th continuous
      state
end for
t_k = t_i // last sample time
//simulation cycle
while(t < t_f) // simulate until final time t_f
  t = min(t_j^η, t_k + h) // advance simulation time.
  if t < t_k + h //quantized state change
    i = argmin(t_j^η) // the i-th quantized state changes first
    x_i = x_i + ẋ_i · (t − t_i^x) // update i-th state value
    q_i = x_i // update i-th quantized state
    t_i^η = t + ΔQ_i/|ẋ_i| // compute the time of the next change in
        q_i
    t_i^x = t // last xi update
    for each j ∈ [1, n] such_that ẋ_j depends_on q_i
      x_j = x_j + ẋ_j · (t − t_j^x) // update j-th state value
      ẋ_j = f_j(q, t) + u_j // recompute j-th state derivative
      t_j^x = t // last xj update
      t_j^η = min(τ > t) subject_to |q_j − x_j(τ)| = ΔQ_j // recompute the
          time of the next change in the j-th quantized state
```

```
    end for
  else //noise sample
    t_k = t //update current sample time
    Δw = √h · N(0, I^{m×m}) // compute random increment vector
    u = g(x,t) · Δw/h// new input vector
    for each j ∈ [1,n] such_that ẋ_j depends_on Δw
      x_j = x_j + ẋ_j · (t − t_j^x) // update j-th state value
      ẋ_j = f_j(q,t) + u_j // recompute j-th state derivative
      t_j^x = t // last xj update
      t_j^η = min(τ > t) subject_to |q_j − x_j(τ)| = ΔQ_j // recompute the
              time of the next change in the j-th quantized state
    end for
  end if
end while
```

The algorithm has two types of steps:

- Quantized state updates: these are identical to those of the QSS1 algorithm for ODEs. A single quantized state q_i changes its value, and all the state derivatives $\dot{x}_j$ depending on q_i are updated, as well as the corresponding times for the next quantized state changes t_j^η.
- Random input updates: these are similar to input changes in the QSS1 algorithm for ODEs. The input vector is updated, taking values from a normally distributed vector according to Eq. (7). Then, the state derivatives $\dot{x}_j$ depending on $\Delta\mathbf{w}$ are updated, as well as the corresponding times for the next quantized state changes t_j^η.

After finishing each step, the simulation cycle starts again, advancing the simulation time until the next change time (which can be a quantized state change or a noise sampling).

Remark 3.1. *The proposed method in Eq. (6) treats the stochastic term as a periodically sampled input to the event-triggered QSS approximation. An alternative approach where the noise is resampled whenever a quantized state q_i is updated might be considered. However its implementation would be impractical since the variance of the random increment $\Delta w(t_k)$ is proportional to the time interval $\Delta t_k = t_{k+1} - t_k$ between samples which cannot be known in advance since the next time of change t_{k+1} in a quantized state depends on the random value of the noise sample $\Delta w(t_k)$ itself.*

The second-order QSS2 method can be identically defined by Eq. (6) with the only difference that the quantized states $\mathbf{q}(t)$ follow piecewise linear trajectories and the approximation $\mathbf{f}^q(\mathbf{q}, t)$ is also piecewise linear. The same idea applies to QSS3, with the quantized trajectories and the function approximations being piecewise parabolic.

Linearly implicit methods LIQSS1, LIQSS2, and LIQSS3 can also be implemented following the same definitions with the appropriate computation of the quantized states.

Algorithm 1 can be generalized for higher order and linearly implicit QSS methods as follows:

Algorithm 2 (LI)QSS(N).

```
1  //inputs: t_i (initial time), t_f (final time), x_0 (initial state
       ), ΔQ (quantum vector), h (step size)).
2  //outputs: x, q (current vectors of polynomials representing
       continuous and quantized state trajectories).
3  //other variables: t (current simulation time), t_n (time of
       the next quantized state change), i (next quantized state
       that changes), t_k (time of last sample), Δw (last noise
       sample), u (last input value).
4  t = t_i
5  t_k = t_i
6  (x, q, t_n, i)=QSS_init(x_0, t, ΔQ)
7  while (t < t_f) //simulation cycle
8    t=min(t_n, t_{k+1})
9    if t_n ≤ t_k + h then //quantized state change
10     (x, q, t_n, i)=QSS_step(x, q, u, t, ΔQ, i)
11   else //noise sample
12     t_k = t_k + h
13     (x, q, t_n, i)=QSS_sample(x, q, t, ΔQ)
14   end if
15 end while
16
17 //functions used
18 QSS_init(x_0, Δw, t, ΔQ)
19   q = constpoly(x_0, t) //initial quantized state polynomial (
       constant)
20   Δw = √h · N(0, I^{m×m})
21   u = g(x, t) · Δw/h// input vector
22   for each j ∈ [1, n]
23     x_j = Statepoly(x_{0,j}, f_j, q, u, t) //initial polynomial of order N
           for the j-th state
24     t_j^η = nTime(x_j, q_j, ΔQ_j) //compute time of next change in q_j
           according to the QSS method used
25   end for
26   t_n = min(t_j^η)
27   i = argmin(t_j^η)
28   return x, q, t_n, i
29
30 QSS_step(x, q, u, t, ΔQ, i)
31   x_i = advpoly(x_i, t) //advance state polynomial to current time
32   q_j = Quantized(x_i, ΔQ_i, t) // compute new quantized state
       polynomial according to the QSS method used
33   t_i^η = nTime(x_i, q_i, ΔQ_i) //time of next change in q_i
34   for each j ∈ [1, n] such that ẋ_j depends_on q_i
35     xj_aux = polyval(x_j, t) //evaluate state polynomial at current
           time
36     x_j = Statepoly(xj_aux, f_j, q, u, t) //compute new state polynomial
           for the j-th state
37     t_j^η = nTime(x_j, q_j, ΔQ_j) //recompute time of next change in q_j
38   end for
39   t_n = min(t_j^η)
40   i = argmin(t_j^η)
41   return x, q, t_n, i
42
43 QSS_sample(x, q, t, ΔQ)
44   Δw = √h · N(0, I^{m×m})
45   u = g(x, t) · Δw/h// input vector
```

```
46    for each j ∈ [1,n] such that ẋ_j depends_on u
47      xj_aux = polyval(x_j,t) //evaluate state polynomial at
            current time
48      x_j = Statepoly(xj_aux, f_j, q, u, t) //compute new state polynomial
            for the j-th state
49      t_j^η = nTime(x_j, q_j, ΔQ_j) //recompute time of next change in q_j
50    end for
51    t_n = min(t_j^η)
52    i = argmin(t_j^η)
53    return x, q, t_n, i
```

In Algorithm 2, function `Statepoly` calculates a polynomial for the state $x_j(t)$ using its current value and its time derivatives (computed from functions f_j an their time derivatives). Similarly, function `Quantized` computes the quantized state polynomial according to the QSS method used while function `nTime` calculates the time for the next change in a quantized state. The way of computing these functions for the different methods can be easily deduced from the definition of the different QSS algorithms.

3.2 Convergence Analysis

We analyze the convergence of the solutions of Eq. (6) to those of Eq. (4) when the step size h and the quantum $\Delta\mathbf{Q}$ approach zero. The result is based on the following assumptions:

Assumption 3.1. *Functions* $\mathbf{f}$, $\mathbf{g}$, *and* $\mathbf{f}^q$ *satisfy the following Lipschitz conditions:*

- $\|\mathbf{f}(\mathbf{x}^q, t^q) - \mathbf{f}(\mathbf{x}, t)\| \leq L_f \left(\|\mathbf{x}^q - \mathbf{x}\| + |t^q - t|\right)$.
- $\|\mathbf{g}(\mathbf{x}^q, t^q) - \mathbf{f}(\mathbf{x}, t)\| \leq L_g \left(\|\mathbf{x}^q - \mathbf{x}\| + |t^q - t|\right)$.

for certain constants L_f *and* L_g. *In addition, we assume that*

- $\mathbf{f}$ *and* $\mathbf{f}^q$ *satisfy* $\|\mathbf{f}^q(\mathbf{x}, t) - \mathbf{f}(\mathbf{x}, t)\| \leq M_f h$.
- *The quantum is chosen such that* $\|\mathbf{q} - \mathbf{x}^q\| \leq \|\Delta\mathbf{Q}\| \leq M_q h$.

for certain constants M_f *and* M_q.

Under these assumptions, the following theorem establishes the convergence of the solutions of the QSS approximation to those of the original SDE:

Theorem 3.1. *Let* $\mathbf{x}(t)$ *and* $\mathbf{x}^q(t)$ *be the solutions of Eqs.* (4) *and* (6), *respectively. Suppose that Assumption 3.1 holds and consider the sampling times* $t_k = t_0 + kh$ *for* $k \geq 0$. *Then, given any* $T > 0$ *and* $c_h > 0$, *there exists a positive constant* c_0 *such that*

$$\sup_{0 \leq k \leq \frac{T}{h}} \mathrm{E}\left[||\mathbf{x}^q(t_k) - \mathbf{x}(t_k)||^2\right] \leq c_0 h \tag{8}$$

for all $h < c_h$.

The proof of this theorem is given in Section A. It is based on bounding the error between the QSS approximation of Eq. (6) and the solution of Euler-Maruyama for Eq. (4).

An immediate consequence of Theorem 3.1 is that when both the sampling period h and the quantum $\Delta\mathbf{Q}$ approach zero, the QSS solution $\mathbf{x}^q(t)$ converges to the SDE solution $\mathbf{x}(t)$ in the mean-square sense.

In the next subsection, we analyze the error bounds for the particular case of stable linear time-invariant systems.

3.3 Stability and Error Bound Analysis

The convergence results provided by Theorem 3.1 show that for a given final time T, the difference between the approximated and the exact solutions at any time $0 \leq t \leq T$ is bounded in expectation by a quantity that depends on the noise sampling period h and also on the quantum $\Delta\mathbf{Q}$ (which is assumed to be proportional to h). Here, we provide a similar bound, but it will be valid for arbitrarily large values of t.

For that purpose, we consider a linear time-invariant system given by

$$\mathrm{d}\mathbf{x}(t) = A\mathbf{x}\mathrm{d}t + B\mathrm{d}\mathbf{w}(t); \quad t \in [0, T]; \quad \mathbf{x}(0) = \mathbf{x}_0 \tag{9}$$

where $\mathbf{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, and the disturbance vector $\mathbf{w}(t) \in \mathbb{R}^m$ is a Wiener process with incremental covariance $I^{m\times m}\mathrm{d}t$. We also assume that A is a Hurwitz matrix.

Taking into account Eq. (6), the QSS approximation of system (9) is given by

$$\dot{\mathbf{x}}(t) = A\mathbf{q}(t) + B\frac{\Delta\mathbf{w}(t_k)}{h}; \quad t \in [t_k, t_{k+1}), \tag{10}$$

which can be regarded as the QSS approximation of the ODE system given by

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\frac{\Delta\mathbf{w}(t_k)}{h}; \qquad t \in [t_k, t_{k+1}). \tag{11}$$

Then, the following theorem establishes an error bound for the solutions of the QSS approximation of Eq. (10) as a function of the sampling period h and the quantum $\Delta\mathbf{Q}$.

Theorem 3.2. *Let $\mathbf{x}(t)$ be the solution of the SDE of Eq. (9) and let $\mathbf{x}^q(t)$ be the solution of its QSS approximation given by Eq. (10) with sampling period h and quantum $\Delta\mathbf{Q}$. Then, for all $t_j = jh$ with $j \in \mathbb{N}$,*

$$\mathrm{E}(||\mathbf{x}^q(t_j) - \mathbf{x}(t_j)||) \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h + c_3 \cdot ||\Delta\mathbf{Q}|| \tag{12}$$

with

$$c_1 \triangleq \frac{1}{2} \cdot v^4 \sqrt{\frac{m \cdot (\max_i |\lambda_i|)^2}{\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}} \cdot ||B||; \quad c_2 \triangleq \sqrt{2 \cdot \min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)} \tag{13}$$

and

$$c_3 = \left\| |V| \cdot |\mathbb{R}\mathrm{e}(\Lambda)^{-1}| \cdot |V^{-1}| \cdot |A| \right\|$$

where $\Lambda = V^{-1}AV$ *is the Jordan canonical form of the Hurwitz matrix* A*, and* $v \triangleq ||V|| \cdot ||V^{-1}||$.

The proof of this theorem, which can be found in Section B of the Appendix, is based on establishing an upper bound for the difference between the solutions of the QSS approximation of Eq. (10) and the solution of the ODE of Eq. (11). Then, we exploit the fact that the latter solution converges to that of the original SDE of Eq. (9).

Theorem 3.2 provides a global bound on the error expectation that contains two separate terms. The first term depends on the sampling period h, and the second one depends on the quantum $\Delta\mathbf{Q}$. The result also shows that the QSS approximation always preserves practical stability since, for any value of h and $\Delta\mathbf{Q}$, the error expectation is bounded for all $t \geq 0$.

Thus, like in ODEs, QSS methods for SDEs have global error bounds and preserve stability of the solutions despite being explicit methods.

Remark 3.2. *(Rate of Convergence).*

Theorem 3.1 and Theorem 3.2 together establish the theoritical rate of convergence for the proposed method.

For the general case, Theorem 3.1 proves that the mean-square error is bounded by a term proportional to the sampling period h*. This establishes a root-mean-square convergence rate of at least* $O(\sqrt{h})$*.*

For the important class of stable linear time invariant (LTI) systems, Theorem 3.2 demonstrates a stronger, first-order convergence result. It shows that the expected error is globally bounded by a function with terms that depend linearly on the sampling period h *and the quantum* ΔQ*. This indicates a convergence rate of* $O(h)$ *with respect to the noise sampling period for this class of systems.*

These results are then verified experimentally in Section 4.3, where a linear dependence of the error on h *is observed for the linear case, in line with Theorem 3.2.*

4 Implementation and Results

In this section, we describe implementation details, and we report numerical experiments aimed at illustrating some features of the simulation of SDEs with QSS methods.

We consider a set of SDEs representing a multi-compartment model of viral replication proposed in (Browning et al. 2024) with the addition of nonlinear terms, defined as follows:

$$\begin{aligned} \mathrm{d}v_1(t) &= -\theta(v_1(t) - \mu)\mathrm{d}t + (a + bv_1(t))\sigma\mathrm{d}w(t), \\ \mathrm{d}v_i(t) &= K(v_{i-1}(t) - v_i(t))\mathrm{d}t + r\frac{v_i(1 - v_i)}{(v_i - 0.5)^2 + m}\mathrm{d}t, \quad i = 2, \ldots, N. \end{aligned} \tag{14}$$

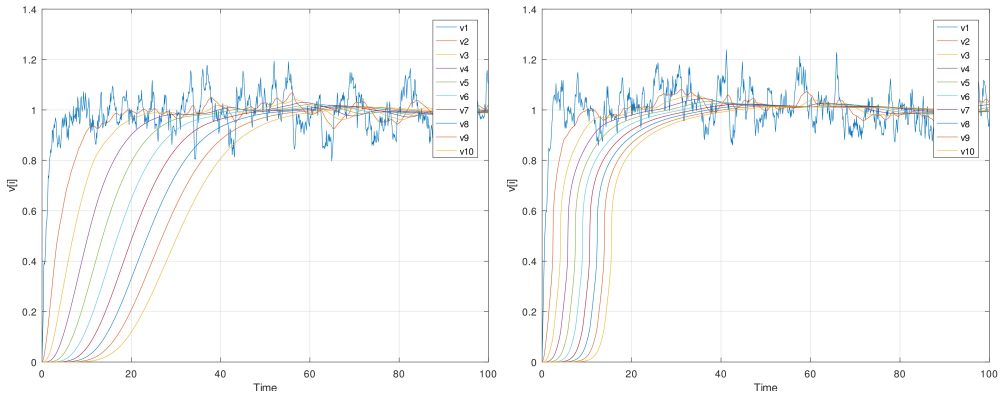
where $w(t) \in \mathbb{R}$ is a scalar Wiener process.

We defined two cases:

- A linear case with $a = 1$, $b = r = 0$. Here, the first state $v_1(t)$ can be thought of as a random input that follows an Ornstein-Uhlenbeck process with mean value μ, noise intensity σ, and reversion strength θ. The remaining states, $v_2, \cdots, v_N$, represent the concentration of material in the compartments, and k is the rate at which material moves between those compartments.
- A nonlinear case with $a = 0$, $b = 1$, $r = 0.01$, and $m = 10^{-6}$, so the SDE contains multiplicative noise and nonlinear dynamics.

In both cases, we simulated the system over a period of 100 seconds, considering the set of parameters: $\theta = \mu = 1$, $K = 0.3$, and $\sigma = 0.1$. We also selected $N = 10$ as the number of compartments.

Figure 2 shows an example of the evolution of the state variables for a particular sample path for $w(t)$ (linear and nonlinear).



(a) Linear Case (b) Non Linear Case

Fig. 2 State Trajectories of Eq. (14).

4.1 Implementation in the Stand-Alone QSS Solver

The algorithm that computes the QSS approximation given by Eq. (6) can be thought of as the QSS approximation of an ODE with a discrete-time input term $\mathbf{u}(t_k) = \mathbf{g}(\mathbf{x}(t_k), t_k)\Delta\mathbf{w}(t_k)$ that is periodically recomputed at event times t_k.

The Stand-Alone QSS solver allows defining models containing discrete variables that are updated under event conditions that might depend on the time or the states. In fact, Algorithm 1 is equivalent to the regular algorithm that executes the QSS solver in the presence of time events. That way, the following Modelica code implements the QSS approximation of the SDE of Eq. (14) in the Stand-Alone QSS Solver:

Algorithm 3 SDE of Eq. (14) in the Stand Alone QSS Solver

```
model virus
  import math;
  constant Integer N=10;
  Real v[N];
  parameter Real K=0.3, th=1, mu=1, sigma=0.1, r=0.01, a=0, b
      =1, m=1e-6;
  discrete Real dw,tk,v1k;
  parameter Real h=0.01;
equation
  der(v[1])=-th*(v[1]-mu)+(a+b*v1k)*sigma*dw/h;
  for i in 2:N loop
    der(v[i])=K*(v[i-1]-v[i])+r*v[i]*(1-v[i])/((v[i]-0.5)^2+m)
        ;
  end for;
algorithm
  when time>tk+h then
     dw:=normal(1)*sqrt(h);
     v1k:=v[1];
     tk:=tk+h;
  end when;
end virus;
```

The translation from Eq. (14) to Algorithm 3 is straightforward. The only detail is the periodic update of the discrete variable `dw` representing the random increment $\Delta\mathbf{w}(t_k)$, which uses the function `normal()` to compute normally distributed pseudo-random samples.

4.2 Numerical Results

To compare solutions and compute errors and mean errors, we generated samples of 30 different Wiener process paths for the signal $w(t)$. Each path was constructed from a pseudo-random sequence of 10^7 normally distributed values representing the random increments in $\Delta w(t_k) = w(t_{k+1}) - w(t_k)$ with $\Delta w \sim \mathcal{N}(0, h)$ and $h = 100/10^7 = 10^{-5}$. From those samples, each sample path was computed by iterating $w(t_{k+1}) = w(t_k) + \Delta w(t_k)$.

Then, the reference solutions were generated from the simulation of the system of Eq. (14) using Euler-Maruyama (EM) algorithm with a step size $h = 10^{-5}$ for each sample path of $w(t)$. This small value of h was chosen to ensure that the reference solution has a much higher fidelity than any of the experimental runs, thus serving as a reliable benchmark for error calculations. Then, we simulated the system varying h and $\mathbf{\Delta Q}$ using different QSS methods (QSS1 to 3). In all the experiments, we used the same quantum ΔQ_i for all the state variables.

The samples of $w(t)$ using different values of h were obtained by down-sampling the sample paths of $w(t)$ (originally sampled with $h = 10^{-5}$) so that the different simulations use different sample rates over the same random signal. In addition to QSS methods, we also simulated the system using the Euler-Maruyama algorithm.

In the different experiments, we computed the errors in two different ways. First, we used a single path for $w(t)$ and compared the complete reference solution with the complete result of each experiment, calculating the error as

$$e_1 = \frac{1}{M}\sum_{k=1}^{M} \|\mathbf{x}(t_k) - \mathbf{x}^{\text{ref}}(t_k)\| \tag{15}$$

where $M = t_f/h$ is the number of sample points of the numerical solution $\mathbf{x}(t_k)$ and $\mathbf{x}^{\text{ref}}(t_k)$ is the reference solution at time t_k.

Then, we ran experiments with 30 different paths for $w(t)$ and compared each reference solution with the trajectories of the solutions obtained using the different algorithms and quantization settings. The error this time was computed as

$$e_2 = \frac{1}{30}\sum_{s=1}^{30} \max_{t_k} \|\mathbf{x}_s(t_k) - \mathbf{x}_s^{\text{ref}}(t_k)\| \tag{16}$$

where $\mathbf{x}_s$ and $\mathbf{x}_s^{\text{ref}}$ are the state trajectories computed using the path $w_s(t)$ obtained from a seed s at the pseudo-random generator.

The errors are reported in Table 1 for the linear case and Table 2 for the nonlinear case.

h	ΔQ_i	QSS1		QSS2		QSS3		EM	
		e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2
1	1E-1	1.30E-1	2.11E-1	1.58E-1	2.18E-1	1.47E-1	2.15E-1	7.37E-2	3.58E-1
	1E-2	2.50E-2	5.80E-2	2.23E-2	5.56E-2	2.26E-2	5.82E-2		
	1E-3	1.62E-2	5.36E-2	1.62E-2	5.34E-2	1.62E-2	5.37E-2		
	1E-4	1.61E-2	5.36E-2	1.61E-2	5.36E-2	1.61E-2	5.37E-2		
	1E-5	1.61E-2	5.36E-2	1.61E-2	5.36E-2	1.61E-2	5.36E-2		
0.1	1E-1	1.47E-1	2.10E-1	1.59E-1	2.15E-1	1.46E-1	2.08E-1	5.84E-3	1.92E-2
	1E-2	1.84E-2	2.85E-2	1.36E-2	2.17E-2	1.45E-2	2.07E-2		
	1E-3	2.64E-3	6.04E-3	2.23E-3	5.71E-3	2.23E-3	5.59E-3		
	1E-4	1.64E-3	5.50E-3	1.64E-3	5.46E-3	1.64E-3	5.48E-3		
	1E-5	1.63E-3	5.47E-3	1.63E-3	5.47E-3	1.63E-3	5.47E-3		
0.01	1E-1	1.47E-1	2.09E-1	1.62E-1	2.22E-1	1.46E-1	2.11E-1	5.77E-4	1.91E-3
	1E-2	1.82E-2	2.81E-2	1.42E-2	2.16E-2	1.43E-2	2.10E-2		
	1E-3	1.97E-3	3.34E-3	1.33E-3	2.08E-3	1.41E-3	2.07E-3		
	1E-4	2.79E-4	6.48E-4	2.43E-4	6.09E-4	2.29E-4	5.96E-4		
	1E-5	1.77E-4	5.82E-4	1.77E-4	5.81E-4	1.77E-4	5.81E-4		
0.001	1E-1	1.46E-1	2.11E-1	1.63E-1	2.23E-1	1.46E-1	2.09E-1	5.51E-5	1.89E-4
	1E-2	1.81E-2	2.78E-2	1.49E-2	2.29E-2	1.45E-2	2.10E-2		
	1E-3	1.96E-3	3.29E-3	1.36E-3	2.11E-3	1.38E-3	1.99E-3		
	1E-4	2.01E-4	3.41E-4	1.49E-4	2.27E-4	1.22E-4	1.95E-4		
	1E-5	2.74E-5	6.28E-5	2.44E-5	5.80E-5	2.19E-5	5.86E-5		

Table 1 Errors for different algorithms and parameters (linear case).

For the different experiments, we also measured the number of simulation steps performed by each algorithm and the CPU[1] time (computed as the mean of 10 simulation runs). The results are reported in Table 3 (linear case) and Table 4 (nonlinear case).

[1] All simulations were run on an Intel Core i5-10400 CPU @ 2.90GHz computer with Kubuntu 24.04 OS. We used in all cases the model of Algorithm 3 on the Stand-Alone QSS Solver. The simulations can be reproduced by downloading the tool from https://github.com/CIFASIS/qss-solver.

h	ΔQ_i	QSS1		QSS2		QSS3		EM	
		e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2
1	1E-1	1.15E-1	5.80E-1	1.91E-1	6.00E-1	1.73E-1	4.56E-1	4.55	72.6
	1E-2	2.58E-2	1.10E-1	3.35E-2	2.59E-1	2.66E-2	1.08E-1		
	1E-3	1.85E-2	8.27E-2	1.79E-2	8.10E-2	1.85E-2	8.34E-2		
	1E-4	1.87E-2	8.33E-2	1.86E-2	8.34E-2	1.87E-2	8.32E-2		
	1E-5	1.87E-2	8.28E-2	1.87E-2	8.23E-2	1.87E-2	8.30E-2		
0.1	1E-1	1.19E-1	5.77E-1	1.96E-1	5.93E-1	1.74E-1	3.78E-1	4.63E-2	79.5
	1E-2	1.97E-2	1.18E-1	3.33E-2	2.62E-1	1.65E-2	1.15E-1		
	1E-3	3.29E-3	1.04E-2	3.81E-3	2.44E-2	3.02E-3	1.00E-2		
	1E-4	2.34E-3	7.70E-3	2.35E-3	7.67E-3	2.34E-3	7.69E-3		
	1E-5	2.34E-3	7.69E-3	2.34E-3	7.68E-3	2.34E-3	7.69E-3		
0.01	1E-1	1.22E-1	5.77E-1	2.25E-1	6.36E-1	1.69E-1	3.87E-1	1.86	7.96
	1E-2	1.96E-2	1.18E-1	2.48E-2	2.55E-1	1.57E-2	9.89E-2		
	1E-3	2.25E-3	1.04E-2	2.59E-3	2.40E-2	1.70E-3	9.56E-3		
	1E-4	5.90E-4	2.03E-3	6.32E-4	3.11E-3	5.79E-4	1.94E-3		
	1E-5	5.46E-4	1.70E-3	5.49E-4	1.70E-3	5.48E-4	1.69E-3		
0.001	1E-1	1.22E-1	5.77E-1	1.94E-1	6.13E-1	1.58E-1	4.39E-1	1.05E-1	1.39
	1E-2	1.95E-2	1.17E-1	2.44E-2	2.50E-1	1.85E-2	1.01E-1		
	1E-3	2.20E-3	1.03E-2	2.48E-3	2.36E-2	1.61E-3	9.93E-3		
	1E-4	2.77E-4	1.48E-3	2.97E-4	3.01E-3	2.24E-4	1.40E-3		
	1E-5	1.47E-4	6.60E-4	1.52E-4	7.65E-4	1.45E-4	6.46E-4		

Table 2 Errors for different algorithms and parameters (nonlinear case).

h	ΔQ_i	QSS1		QSS2		QSS3		EM	
		Steps	Time	Steps	Time	Steps	Time	Steps	Time
1	0.1	321	0.10	510	0.14	638	0.31	100	0.10
	0.01	1951	0.24	630	0.15	874	0.40		
	0.001	18693	1.28	1474	0.25	1105	0.46		
	0.0001	185978	14.09	4590	0.53	1642	0.60		
	0.00001	1859026	142.00	14536	1.47	3205	1.03		
0.1	0.1	1473	0.24	1552	0.28	1807	0.60	1000	0.30
	0.01	4455	0.46	2194	0.33	2416	0.80		
	0.001	37580	2.73	3451	0.46	3671	1.18		
	0.0001	369393	27.86	8206	0.91	4701	1.45		
	0.00001	3687952	315.36	23561	2.55	6992	2.09		
0.01	0.1	12522	1.53	10843	1.46	11114	2.88	10000	2.17
	0.01	17996	2.01	15513	1.86	15939	4.23		
	0.001	98470	8.41	20435	2.21	20732	5.78		
	0.0001	917673	76.47	24157	2.89	24698	7.01		
	0.00001	9112262	764.22	47344	5.00	37482	10.39		
0.001	0.1	122778	16.10	101537	13.34	102080	25.97	100000	39.01
	0.01	130969	16.98	117226	15.72	117686	30.99		
	0.001	342606	36.19	183430	22.13	184311	51.74		
	0.0001	2714325	232.96	202156	24.92	203314	59.36		
	0.00001	26484934	2183.31	214968	25.99	215169	62.76		

Table 3 CPU time (milliseconds) and simulation steps (linear case).

h	ΔQ_i	QSS1		QSS2		QSS3		EM	
		Steps	Time	Steps	Time	Steps	Time	Steps	Time
1	0.1	295	0.11	607	0.17	857	0.48	100	0.09
	0.01	1943	0.26	902	0.22	1186	0.63		
	0.001	18403	1.40	1972	0.36	1674	0.78		
	0.0001	183065	13.93	5876	0.80	2686	1.13		
	0.00001	1829770	152.66	18669	2.20	4860	2.04		
0.1	0.1	1419	0.25	1635	0.32	1863	0.72	1000	0.30
	0.01	4441	0.47	7726	1.01	2760	1.08		
	0.001	37227	2.96	3953	0.59	4219	1.65		
	0.0001	365821	29.38	9492	1.20	5646	2.25		
	0.00001	3652519	299.43	27684	3.35	8649	3.37		
0.01	0.1	12476	1.60	10954	1.53	11161	3.03	10000	4.02
	0.01	17985	1.98	15661	1.96	16000	4.93		
	0.001	97495	7.78	20883	2.55	21271	6.86		
	0.0001	907573	73.23	25485	3.23	25612	8.79		
	0.00001	9010856	733.29	51373	6.27	38990	13.43		
0.001	0.1	122738	15.00	101679	14.73	102047	26.29	100000	44.72
	0.01	130847	15.80	117171	16.60	117673	33.16		
	0.001	339363	34.57	183346	23.49	184330	59.30		
	0.0001	2676996	222.56	203311	27.85	204181	65.90		
	0.00001	26110176	2070.17	219050	27.66	216671	70.24		

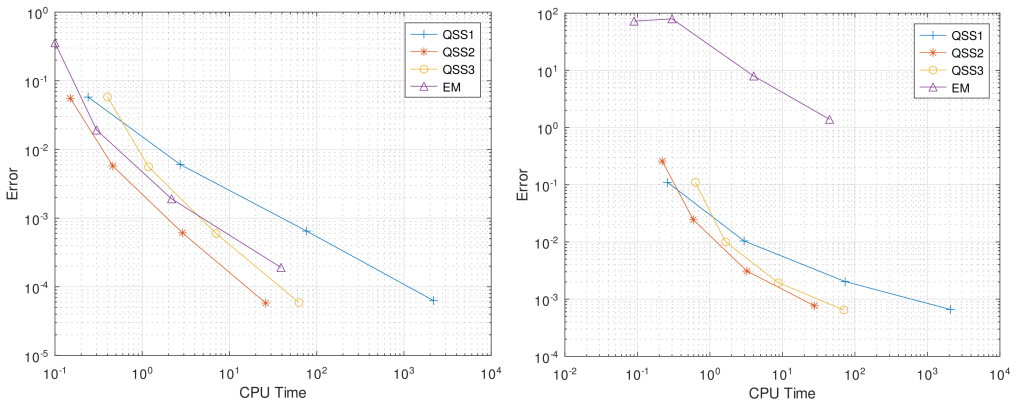
Table 4 CPU time (milliseconds) and simulation steps (nonlinear case).

4.3 Result Analysis

Based on the results reported in Tables 1–4, we can make the following remarks:

- The error of the different QSS methods is very similar using the same quantum and sampling period. This was expected since the error introduced by QSS methods depends on the quantum and not on the order of the algorithm (higher-order methods perform fewer steps).
- As expected, the errors decrease when both the quantum and the sample period decrease. Also, the computational costs increase in that case.
- Looking at the results for $h = 10^{-3}$ (where the contribution of h to the error is small), the linear dependence of the error with the quantum is evident. This was predicted by 3.2, and it corroborates that, just like in ODEs, the quantum $\Delta\mathbf{Q}$ plays the role of the error tolerance.
- Looking at the results for $\Delta Q_i = 10^{-5}$ (where the contribution of the quantization to the error is small), the linear dependence of the error with h predicted by Theorem 3.2 is also evident for the linear case.
- It does not make sense to use a small quantum ΔQ_i when h is large since the overall error will be lower-bounded by the error contribution of the sample period. The same conclusion holds when using a small value for h and a large value for $\Delta\mathbf{Q}$.
- Taking into account the previous remark, the *best* choice for the quantum and the sample time would be such that both terms contribute equally to the error. Observing the results, this situation occurs around $\Delta Q_i = h/100$.

- In that situation, the error for the nonlinear case seems to converge linearly when h goes from 0 to 0.01. At $h = 0.001$, the error decreases slower than linear but faster than $\sqrt{h}$ (which was the upper bound established by Theorem 3.1).
- Regarding computational costs, the number of steps performed by QSS methods can be computed as $M + N$, where M is the number of noise samples, i.e., $M = 100/h$, and N is the total number of updates in the quantized variables. Like in ODEs, N is inversely proportional to the quantum in QSS1, and inversely proportional to $\sqrt{\Delta Q_i}$ in QSS2 and to $\sqrt[3]{\Delta Q_i}$ in QSS3.
- The superior performance of QSS2 over QSS3 in terms of CPU time (Tables 3 and 4) highlights a key trade-off between the number of simulation steps and the computational cost per step. While higher-order methods like QSS3 perform fewer steps for a given quantum (as step count is proportional to $1/\sqrt{\Delta Q_i}$ for QSS2 and $1/\sqrt[3]{\Delta Q_i}$ for QSS3), the overhead of each step is significantly higher. We measured that a QSS3 step is roughly 4 times more computationally expensive than a QSS1 step, while a QSS2 step is only 1.5 times more expensive. For the problems tested, the reduction in step count from QSS3 was not enough to offset its higher per-step cost, making QSS2 the most efficient method overall.
- Under *optimal* settings, i.e., $\Delta Q_i = h/100$, the QSS2 simulations are considerably faster than the EM simulations that achieve similar errors in the linear case. Moreover, QSS3 for small values of h is also faster than EM. Figure 3(a) compares the CPU-time and the error obtained for the different methods, illustrating this observation. For the nonlinear case, the comparison is shown in Figure 3(b), but here EM produces huge errors.



(a) Linear Case (b) Nonlinear Case

Fig. 3 CPU time vs. error (e_2) with different methods.

- The number of steps performed by Euler-Maruyama is smaller than those of QSS methods. However, every EM step involves updating the 10 state variables, while every QSS step only updates 2 states since every state appears on the right-hand side of two equations. Moreover, after each sampling of the random signal, only

one state is recomputed since $w(t)$ only acts on $\dot{x}_1(t)$. Consequently, QSS2 is almost as fast as EM but performs a larger number of steps.

- A closer look at Figure 2 shows that the trajectories of $x_i(t)$ become smoother as i increases. The first state $x_1(t)$ is directly affected by $w(t)$, so its trajectory looks very noisy. The second state $x_2(t)$ is affected only by $x_1(t)$, so it is smoother than $x_1(t)$, and similarly for the remaining states. In this case, the QSS methods automatically exploit this feature, performing more steps to compute $x_1(t)$ (because it performs more quantum crossings) and very few steps to compute $x_{10}(t)$.
- The previous remarks suggest that the most important advantages of the QSS algorithms in the linear case are related to the fact that the system is sparse and that the random signal $w(t)$ only appears in the expression of one state derivative ($\dot{x}_1(t)$). If that signal were present in all the state derivatives, then it would not make much sense to use the QSS approach.
- Another important advantage of the QSS algorithm observed in the nonlinear case is the intrinsic step-size adaptation. States with large derivatives are updated at a faster rate, but only while the derivatives are large. So, at a given instant of time, some states may be being updated with a fast rate, while others are being updated with a slower rate. This fact combines the advantages of step-size adaptation and multi-rate integration.

5 Conclusions and Future Research

We presented a novel method for simulating systems of SDEs based on the use of Quantized State Systems. The algorithm samples the noise process using a constant period h and simulates the SDE as if it were an ODE with a discontinuous input term $\mathbf{u}(t_k) \triangleq \mathbf{g}(\mathbf{x}(t_k), t_k)\frac{\Delta \mathbf{w}(t_k)}{h}$. We proved that the algorithm verifies convergence, error bound, and stability properties equivalent to those of QSS methods for ODEs.

We also presented simulation results that exhibited some features of the proposed methodology. Those results suggest that QSS methods may offer potential advantages over classic approaches since they are able to compute the state trajectories at different rates, performing fewer steps on the trajectories that are smoother.

Anyway, this work is only a first attempt to study the use of QSS methods in SDEs, and it leaves several lines of research open for the future.

As we mentioned earlier, the quantum $\Delta\mathbf{Q}$ plays the role of the error tolerance, so its selection is straightforward. However, given $\Delta\mathbf{Q}$ (or the error tolerance), it is not clear what the correct value for h is so that it contributes to the error in a similar way to $\Delta\mathbf{Q}$ (as occurred in the example when $h = 100\Delta Q_i$). A possible way to find a suitable value for h would be to make it adaptive, following ideas like those of (Hofmann et al. 2000; Lamba 2003; Malik 2021). An even more interesting idea would be applying quantization principles to the signal $\mathbf{w}(t)$, using the concept of *first-passage time* for the resulting process.

Regarding potential applications of the proposed methodology, the main advantage of QSS methods is related to the simulation of discontinuous systems of ODEs. Thus, we can expect that the same occurs in the context of SDEs. Therefore, future work will

also aim to establish the main properties of the QSS approximations of discontinuous stochastic differential equations.

Finally, although we included some comparisons with Euler-Maruyama in the example of Eq. (14), it was not the goal of this work to show that the proposed approach is more efficient than classic algorithms. Future work must compare QSS results with those of more advanced classic SDE integration methods to establish in which situations —if any— the quantization-based algorithms are convenient.

References

Abdulle, A., Souza, G.R.: Explicit stabilized multirate method for stiff stochastic differential equations. SIAM Journal on Scientific Computing **44**(4), 1859–1883 (2022)

Assar, R., Sherman, D.J.: Implementing biological hybrid systems: Allowing composition and avoiding stiffness. Applied Mathematics and Computation **223**, 167–179 (2013)

Bergonzi, M., Fernández, J., Castro, R., Muzy, A., Kofman, E.: Quantization-based simulation of spiking neurons: theoretical properties and performance analysis. Journal of Simulation, 1–24 (2023)

Bergero, F., Fernández, J., Kofman, E., Portapila, M.: Time discretization versus state quantization in the simulation of a one-dimensional advection–diffusion–reaction equation. Simulation **92**(1), 47–61 (2016)

Browning, A.P., Jenner, A.L., Baker, R.E., Maini, P.K.: Smoothing in linear multicompartment biological processes subject to stochastic input. Physical Review E **109**(5), 054405 (2024)

Baccouch, M., Temimi, H., Ben-Romdhane, M.: A discontinuous galerkin method for systems of stochastic differential equations with applications to population biology, finance, and physics. Journal of Computational and Applied Mathematics **388**, 113297 (2021)

Castro, R., Bergonzi, M., Marcosig, E.P., Fernández, J., Kofman, E.: Discrete-event simulation of continuous-time systems:: evolution and state of the art of quantized state system methods. SIMULATION **100**(6), 613–638 (2024)

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

Fernández, J., Kofman, E.: A stand-alone quantized state system solver for continuous system simulation. Simulation **90**(7), 782–799 (2014)

Hofmann, N., Müller-Gronbach, T., Ritter, K.: Optimal approximation of stochastic differential equations by adaptive step-size control. Mathematics of computation **69**(231), 1017–1034 (2000)

Kofman, E., Haimovich, H., Seron, M.: A systematic method to obtain ultimate bounds for perturbed systems. International Journal of Control **80**(2), 167–178 (2007)

Kofman, E., Junco, S.: Quantized-state systems: a devs approach for continuous system simulation. Transactions of the Society for Computer Simulation International **18**(3), 123–132 (2001)

Kofman, E.: Quantization-based simulation of differential algebraic equation systems. Simulation **79**(7), 363–376 (2003)

Lamba, H.: An adaptive timestepping algorithm for stochastic differential equations. Journal of computational and applied mathematics **161**(2), 417–430 (2003)

Malik, A.: Adaptive step size numerical integration for stochastic differential equations with discontinuous drift and diffusion. Numerical Algorithms **87**(2), 849–872 (2021)

Migoni, G., Kofman, E., Bergero, F., Fernández, J.: Quantization-based simulation of switched mode power supplies. Simulation **91**(4), 320–336 (2015)

Pilch, C., Niehage, M., Remke, A.: Hpngs go non-linear: Statistical dependability evaluation of battery-powered systems. In: 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 157–169 (2018). IEEE

Soto-Francés, V.-M., Sarabia-Escrivá, E.-J., Pinazo-Ojer, J.-M., Martínez-Beltrán, P.-J.: Exploring the use of traditional heat transfer functions for energy simulation of buildings using discrete events and quantized-state-based integration. Journal of Building Performance Simulation **13**(3), 247–263 (2020)

A Proof of Theorem 3.1

A.1 Auxiliary Results

The proof of Theorem 3.1 uses some auxiliary results that are given below. These results aim to establish an upper bound for the difference between QSS and Euler-Maruyama (EM) solutions (Theorem A.1). For this purpose, we use the following notation:

- $t_k = t_0 + kh$, with $k = 0, \ldots, N$ are the sampling times where $N \leq T/h$ is the number of samples in the interval $[t_0, T]$.
- $\mathbf{x}(t)$ denotes the solution of the SDE of Eq.(4).
- $\mathbf{x}^q(t)$ denotes the solution of the QSS approximation given by Eq.(6).
- $\mathbf{x}^e(t_k)$ denotes Euler-Maruyama solution of Eq.(4), given by

$$\mathbf{x}^e(t_{k+1}) = \mathbf{x}^e(t_k) + h\mathbf{f}(\mathbf{x}^e(t_k), t_k) + \mathbf{g}(\mathbf{x}^e(t_k), t)\Delta\mathbf{w}_k; \quad \mathbf{x}^e(t_0) = \mathbf{x}_0 \tag{17}$$

where $\Delta\mathbf{w}_k \triangleq \Delta\mathbf{w}(t_k)$.

The proposition below introduces a compact expression for the difference between QSS and EM solutions.

Proposition A.1. *Let $\mathbf{x}^q$ and $\mathbf{x}^e$ denote the QSS and the EM solutions of Eq.(4), given by Eqs.(6) and Eq.(17), respectively. Then, their difference can be written as*

$$\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k) = \sum_{j=0}^{k-1} h\Delta\mathbf{f}_j + \Delta\mathbf{g}_j\Delta\mathbf{w}_j$$

for $k = 0, \ldots, N$, where we defined

$$\Delta\mathbf{f}_j \triangleq \frac{1}{h}\int_{t_j}^{t_{j+1}} (\mathbf{f}^q(\mathbf{q}(t), t) - \mathbf{f}(\mathbf{x}^e(t_j), t_j))\mathrm{d}t, \quad \Delta\mathbf{g}_j \triangleq \mathbf{g}(\mathbf{x}^q(t_j), t_j) - \mathbf{g}(\mathbf{x}^e(t_j), t_j) \tag{18}$$

Proof The solution of Eq.(6) at time t_k is

$$\mathbf{x}^q(t_k) = \mathbf{x}^q(t_0) + \sum_{j=0}^{k-1}\int_{t_j}^{t_{j+1}} (\mathbf{f}^q(\mathbf{q}(t), t) + \mathbf{g}(\mathbf{x}^q(t_j), t_j)\frac{\Delta\mathbf{w}_j}{h})\mathrm{d}t \tag{19}$$

while Euler-Maruyama's solution at time t_k can be computed as

$$\mathbf{x}^e(t_k) = \mathbf{x}^e(t_0) + \sum_{j=0}^{k-1}\int_{t_j}^{t_{j+1}} (\mathbf{f}(\mathbf{x}^e(t_j), t_j) + \mathbf{g}(\mathbf{x}^e(t_j), t_j)\frac{\Delta\mathbf{w}_j}{h})\mathrm{d}t. \tag{20}$$

Subtracting Eq.(20) from Eq.(19), and recalling that $\mathbf{x}^e(t_0) = \mathbf{x}^q(t_0) = \mathbf{x}(t_0)$, we obtain

$$\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k) = \sum_{j=0}^{k-1}\int_{t_j}^{t_{j+1}} (\mathbf{f}^q(\mathbf{q}(t), t) - \mathbf{f}(\mathbf{x}^e(t_j), t_j) + (\mathbf{g}(\mathbf{x}^q(t_j), t_j) - \mathbf{g}(\mathbf{x}^e(t_j), t_j))\frac{\Delta\mathbf{w}_j}{h})\mathrm{d}t$$

and the result follows after replacing with Eq.(18) in the last expression. □

Based on Proposition A.1, the following lemma provides an expression for the sequence of the expected value of the difference between EM and QSS solutions.

Lemma A.1. *Define*

$$e_k \triangleq \mathrm{E}[\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2] \tag{21}$$

for $k = 0, \ldots, N$. Then, the sequence e_k verifies

$$e_{k+1} = e_k + h^2\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_k] + 2h^2\sum_{j=0}^{k-1}\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_j] + \mathrm{E}[\|\Delta\mathbf{g}_k\Delta\mathbf{w}_k\|^2] + 2h\mathrm{E}[\Delta\mathbf{f}_k^T\sum_{j=0}^{k}\Delta\mathbf{g}_j\Delta\mathbf{w}_j] \tag{22}$$

Proof From Proposition A.1 we can write

$$\begin{aligned}\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2 &= \sum_{l=0}^{k-1}(h\Delta\mathbf{f}_l^T + \Delta\mathbf{w}_l^T\Delta\mathbf{g}_l^T)\sum_{j=0}^{k-1}(h\Delta\mathbf{f}_j + \Delta\mathbf{g}_j\Delta\mathbf{w}_j)\\ &= \sum_{l=0}^{k-1}\sum_{j=0}^{k-1}(h\Delta\mathbf{f}_l^T + \Delta\mathbf{w}_l^T\Delta\mathbf{g}_l^T)(h\Delta\mathbf{f}_j + \Delta\mathbf{g}_j\Delta\mathbf{w}_j)\\ &= \sum_{l=0}^{k-1}\sum_{j=0}^{k-1} h^2\Delta\mathbf{f}_l^T\Delta\mathbf{f}_j + \Delta\mathbf{w}_l^T\Delta\mathbf{g}_l^T\Delta\mathbf{g}_j\Delta\mathbf{w}_j + 2h\Delta\mathbf{f}_l^T\Delta\mathbf{g}_j\Delta\mathbf{w}_j\end{aligned}$$

Then, taking expectation at both sides, it results

$$e_k = \sum_{l=0}^{k-1}\sum_{j=0}^{k-1}\left(h^2\mathrm{E}[\Delta\mathbf{f}_l^T\Delta\mathbf{f}_j] + \mathrm{E}[\Delta\mathbf{w}_l^T\Delta\mathbf{g}_l^T\Delta\mathbf{g}_j\Delta\mathbf{w}_j]\right) + \mathrm{E}[\sum_{l=0}^{k-1}2h\Delta\mathbf{f}_l^T\sum_{j=0}^{k-1}\Delta\mathbf{g}_j\Delta\mathbf{w}_j]$$

We know that $\Delta\mathbf{w}_j$ and $\Delta\mathbf{w}_l$ are uncorrelated for $j \neq l$. Also, it can be easily noticed from Eq.(18) that $\Delta\mathbf{g}_j$ and $\Delta\mathbf{g}_l$ are uncorrelated when $j \leq l$. Thus, $\mathrm{E}[\Delta\mathbf{w}_l^T\Delta\mathbf{g}_l^T\Delta\mathbf{g}_j\Delta\mathbf{w}_j] = 0$ for $j \neq l$ and the previous equation becomes

$$e_k = \sum_{l=0}^{k-1}\sum_{j=0}^{k-1} h^2\mathrm{E}[\Delta\mathbf{f}_l^T\Delta\mathbf{f}_j] + \sum_{l=0}^{k-1}\mathrm{E}[\Delta\mathbf{w}_l^T\Delta\mathbf{g}_l^T\Delta\mathbf{g}_l\Delta\mathbf{w}_l] + \mathrm{E}[\sum_{l=0}^{k-1}2h\Delta\mathbf{f}_l^T\sum_{j=0}^{k-1}\Delta\mathbf{g}_j\Delta\mathbf{w}_j].$$

Comparing this expression for e_k and e_{k+1} we obtain

$$\begin{aligned}e_{k+1} &= e_k + \sum_{j=0}^{k} h^2\mathrm{E}[\Delta\mathbf{f}_j^T\Delta\mathbf{f}_k] + \sum_{j=0}^{k-1} h^2\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_j] + \mathrm{E}[\Delta\mathbf{w}_k^T\Delta\mathbf{g}_k^T\Delta\mathbf{g}_k\Delta\mathbf{w}_k] +\\ &+ 2h\mathrm{E}[\Delta\mathbf{f}_k^T\sum_{j=0}^{k}\Delta\mathbf{g}_j\Delta\mathbf{w}_j] + \sum_{j=0}^{k-1}2h\mathrm{E}[\Delta\mathbf{f}_j^T\Delta\mathbf{g}_k\Delta\mathbf{w}_k]\\ &= e_k + h^2\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_k] + 2h^2\sum_{j=0}^{k-1}\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_j] + \mathrm{E}[\|\Delta\mathbf{g}_k\Delta\mathbf{w}_k\|^2] + 2h\mathrm{E}[\Delta\mathbf{f}_k^T\sum_{j=0}^{k}\Delta\mathbf{g}_j\Delta\mathbf{w}_j]\end{aligned}$$

completing the proof. In the last step we used that $\mathrm{E}[\Delta\mathbf{f}_j^T\Delta\mathbf{g}_k\Delta\mathbf{w}_k] = 0$ for $j < k$, since the product $\Delta\mathbf{f}_j^T\Delta\mathbf{g}_k$ is uncorrelated with $\Delta\mathbf{w}_k$. □

The next lemma obtains an upper bound for the last term of the right hand side of Eq. (22).

Lemma A.2. *Let $\Delta\mathbf{f}_k$ and $\Delta\mathbf{g}_j$ be defined by Eq.(18). Then,*

$$|\mathrm{E}[\Delta\mathbf{f}_k^T\sum_{j=0}^{k}\Delta\mathbf{g}_j\Delta\mathbf{w}_j]|^2 \leq \mathrm{E}[\|\Delta\mathbf{f}_k\|^2]\sum_{j=0}^{k}\mathrm{E}[\|\Delta\mathbf{g}_j\Delta\mathbf{w}_j\|^2] \tag{23}$$

for all $0 \leq k \leq N$.

Proof Using Cauchy-Schwarz inequality, it results:

$$
\begin{aligned}
|\mathrm{E}[\Delta\mathbf{f}_k^T \sum_{j=0}^{k} \Delta\mathbf{g}_j \Delta\mathbf{w}_j]|^2 &\leq \mathrm{E}[\|\Delta\mathbf{f}_k^T\|^2]\mathrm{E}[\|\sum_{j=0}^{k} \Delta\mathbf{g}_j \Delta\mathbf{w}_j\|^2] \\
&= \mathrm{E}[\|\Delta\mathbf{f}_k\|^2]\mathrm{E}[(\sum_{j=0}^{k} \Delta\mathbf{g}_j \Delta\mathbf{w}_j)^T (\sum_{i=0}^{k} \Delta\mathbf{g}_i \Delta\mathbf{w}_i)] \\
&= \mathrm{E}[\|\Delta\mathbf{f}_k\|^2]\mathrm{E}[\sum_{j=0}^{k}\sum_{i=0}^{k} \Delta\mathbf{w}_j^T \Delta\mathbf{g}_j^T \Delta\mathbf{g}_i \Delta\mathbf{w}_i] \\
&= \mathrm{E}[\|\Delta\mathbf{f}_k\|^2] \sum_{j=0}^{k}\sum_{i=0}^{k} \mathrm{E}[\Delta\mathbf{w}_j^T \Delta\mathbf{g}_j^T \Delta\mathbf{g}_i \Delta\mathbf{w}_i] \\
&= \mathrm{E}[\|\Delta\mathbf{f}_k\|^2] \sum_{j=0}^{k} \mathrm{E}[\Delta\mathbf{w}_j^T \Delta\mathbf{g}_j^T \Delta\mathbf{g}_j \Delta\mathbf{w}_j]
\end{aligned}
$$

and Eq. (23) follows. In the last step we used that $\mathrm{E}[\Delta\mathbf{w}_j^T \Delta\mathbf{g}_j^T \Delta\mathbf{g}_i \Delta\mathbf{w}_i] = 0$ for $i \neq j$. □

The following lemma establishes an upper bound for the variation of the QSS solution at inter-sample times. This will be latter used to find an upper bound on some terms of the sequence defined in Lemma A.1.

Lemma A.3. *Under Assumption 3.1, given $c_h > 0$ there exist constants a_1 and a_2 such that*

$$
\|\mathbf{x}^q(t) - \mathbf{x}^q(t_k)\| \leq a_1 h + L_f h \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + a_2 \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| + L_f h \|\mathbf{x}^e(t_k)\| \tag{24}
$$

for all $h \leq c_h$, for any $0 \leq k < N$ and for all $t \in [t_k, t_{k+1})$.

Proof From the solution of Eq.(6) it results

$$
\mathbf{x}^q(t) - \mathbf{x}^q(t_k) = \int_{t_k}^{t} (\mathbf{f}^q(\mathbf{q}(\tau), \tau) + \mathbf{g}(\mathbf{x}^q(t_k), t_k) \frac{\Delta\mathbf{w}_k}{h}) \mathrm{d}\tau
$$

Then,

$$
\begin{aligned}
\|\mathbf{x}^q(t) - \mathbf{x}^q(t_k)\| &\leq \int_{t_k}^{t} \|\mathbf{f}^q(\mathbf{q}(\tau), \tau)\| \mathrm{d}\tau + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| \\
&\leq \int_{t_k}^{t} (\|\mathbf{f}^q(\mathbf{q}(\tau), \tau) - \mathbf{f}(q(\tau), \tau)\| + \|\mathbf{f}(q(\tau), \tau)\|) \mathrm{d}\tau + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| \\
&\leq \int_{t_k}^{t} (M_f h + L_f \|\mathbf{q}(\tau) + T\|) \mathrm{d}\tau + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| \\
&\leq \int_{t_k}^{t} L_f \|\mathbf{q}(\tau)\| \mathrm{d}\tau + M_f h^2 + L_f T h + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| \\
&\leq \int_{t_k}^{t} L_f (\|\mathbf{q}(\tau) - \mathbf{x}^q(\tau)\| + \|\mathbf{x}^q(\tau) - \mathbf{x}^q(t_k)\| + \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + \|\mathbf{x}^e(t_k)\|) \mathrm{d}\tau +
\end{aligned}
$$

$$
\begin{aligned}
&+ M_f h^2 + L_f T h + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| \\
&\leq \int_{t_k}^{t} L_f(M_q h + \|\mathbf{x}^q(\tau) - \mathbf{x}^q(t_k)\| + \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + \|\mathbf{x}^e(t_k)\|)\mathrm{d}\tau + \\
&+ M_f h^2 + L_f T h + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| \\
&\leq \int_{t_k}^{t} L_f\|\mathbf{x}^q(\tau) - \mathbf{x}^q(t_k)\|\mathrm{d}\tau + L_f h(M_q h + \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + \|\mathbf{x}^e(t_k)\| + T) + \\
&+ M_f h^2 + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\|
\end{aligned}
$$

Using Gronwall-Bellman inequality, we obtain

$$
\begin{aligned}
&\|\mathbf{x}^q(t) - \mathbf{x}^q(t_k)\| \leq (L_f h(M_q h + \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + \|\mathbf{x}^e(t_k)\| + T) + M_f h^2 + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\|)e^{L_f h} \\
&\quad \leq L_f h\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + L_f h\|\mathbf{x}^e(t_k)\| + (M_f h + M_q L_f h + L_f T)h + \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\|)e^{L_f h}
\end{aligned}
$$

from which the result of Eq.(24) follows with taking $a_1 = M_f + (T + M_q)c_h$ and $a_2 = e^{L_f c_h}$. □

The following lemma provides an upper bound for the term $E[|\Delta\mathbf{f}_k^T \Delta\mathbf{f}_j|]$ that appears in the sequence defined by Lemma A.1.

Lemma A.4. *Let $\Delta\mathbf{f}_j, \Delta\mathbf{f}_k$ be defined by Eq.(18) and e_j, e_k by Eq.(21). Then, under Assumption 3.1 given $c_h > 0$ there exist constants c_1 and c_2 such that*

$$
E[|\Delta\mathbf{f}_k^T \Delta\mathbf{f}_j|] \leq c_1 h + c_2 \max(e_j, e_k) \tag{25}
$$

for all $h \leq c_h$ and for all $k, j \leq N$.

Proof We can bound $\|\Delta\mathbf{f}_k\|$ as follows:

$$
\begin{aligned}
\|\Delta\mathbf{f}_k\| &\leq \frac{1}{h}\int_{t_k}^{t_{k+1}} \|\mathbf{f}^q(\mathbf{q}(t), t) - \mathbf{f}(\mathbf{x}^e(t_k), t_k)\|\mathrm{d}t \\
&\leq \frac{1}{h}\int_{t_k}^{t_{k+1}} (\|\mathbf{f}^q(\mathbf{q}(t), t) - \mathbf{f}(\mathbf{q}(t), t)\| + \|\mathbf{f}(\mathbf{q}(t), t) - \mathbf{f}(\mathbf{x}^e(t_k), t_k)\|)\mathrm{d}t \\
&\leq \frac{1}{h}\int_{t_k}^{t_{k+1}} (M_f h + L_f(\|\mathbf{q}(t) - \mathbf{x}^e(t_k)\| + (t - t_k)))\mathrm{d}t \\
&\leq \frac{1}{h}\int_{t_k}^{t_{k+1}} L_f(\|\mathbf{q}(t) - \mathbf{x}^q(t)\| + \|\mathbf{x}^q(t) - \mathbf{x}^e(t_k)\| + (t - t_k))\mathrm{d}t + M_f h \\
&\leq \frac{1}{h}\int_{t_k}^{t_{k+1}} L_f(M_q h + \|\mathbf{x}^q(t) - \mathbf{x}^q(t_k)\| + \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + h)\mathrm{d}t + M_f h \\
&\leq \frac{1}{h}\int_{t_k}^{t_{k+1}} L_f(M_q h + a_1 h + L_f h\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + a_2\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| + L_f h\|\mathbf{x}^e(t_k)\| + \\
&+ \|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + h)\mathrm{d}t + M_f h
\end{aligned}
$$

where we used the result of Lemma A.3 in the last step. From there, it results

$$
\|\Delta\mathbf{f}_k\| \leq L_f(M_q h + a_1 h + (L_f h + 1)\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + a_2\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| + L_f h\|\mathbf{x}^e(t_k)\| + h) + M_f h
$$

$$\leq (L_f M_q + L_f a_1 + L_f + M_f)h + (L_f^2 h + L_f)\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + L_f a_2 \|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| + L_f^2 h\|\mathbf{x}^e(t_k)\|$$
$$\leq b_1 h + b_2\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + b_3\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| + b_4 h\|\mathbf{x}^e(t_k)\|$$

for $h \leq c_h$ and taking $b_2 = L_f^2 c_h + L_f$.

Then,

$$\Delta\mathbf{f}_k^T \Delta\mathbf{f}_k \leq (b_1 h + b_2\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + b_3\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\Delta\mathbf{w}_k\| + b_4 h\|\mathbf{x}^e(t_k)\|)^2$$
$$\leq 4(b_1^2 h^2 + b_2^2\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2 + b_3^2\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\|^2\|\Delta\mathbf{w}_k\|^2 + b_4^2 h^2\|\mathbf{x}^e(t_k)\|^2).$$

Taking expectation at both sides, we obtain

$$\mathrm{E}[\Delta\mathbf{f}_k^T \Delta\mathbf{f}_k] \leq 4b_1^2 h^2 + 4b_2^2\mathrm{E}[\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2] + 4b_3^2\mathrm{E}[\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\|^2\|\Delta\mathbf{w}_k\|^2] + 4b_4^2 h^2\mathrm{E}[\|\mathbf{x}^e(t_k)\|^2]$$
$$\leq 4b_1^2 h^2 + 4b_2^2 e_k + 4b_3^2\mathrm{E}[\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\|^2]mh + 4b_4^2 h^2 M_x \tag{26}$$

where we used that $\mathrm{E}[\|\Delta\mathbf{w}_k\|^2] \leq m\sqrt{h}$.

From the Lipschitz condition it results

$$\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\| \leq L_g(\|\mathbf{x}^q(t_k)\| + T)$$
$$\leq L_g(\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\| + \|\mathbf{x}^e(t_k)\| + T)$$

then it follows that

$$\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\|^2 \leq 3L_g^2(\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2 + \|\mathbf{x}^e(t_k)\|^2 + T^2)$$

Taking expectation at both sides, we get

$$\mathrm{E}[\|\mathbf{g}(\mathbf{x}^q(t_k), t_k)\|^2] \leq 3L_g^2(\mathrm{E}[\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2] + \mathrm{E}[\|\mathbf{x}^e(t_k)\|^2] + T^2)$$
$$\leq 3L_g^2(e_k + M_x + T^2)$$

where M_x is the upper bound for the expected value of Euler-Maruyama solution $\mathrm{E}[\|\mathbf{x}^e(t_k)\|^2]$ using any step size $h \leq c_h$ and for all $k = 0, \ldots, N$. This upper bound exists due to the convergence of EM under Assumption 3.1.

Then, replacing in Eq.(26), we obtain

$$\mathrm{E}[\Delta\mathbf{f}_k^T \Delta\mathbf{f}_k] \leq 4b_1^2 h^2 + 4b_2^2 e_k + 4b_3^2(3L_g^2(e_k + M_x + T^2))mh + 4b_4^2 h^2 M_x$$
$$\leq (12b_3^2 L_g^2(M_x + T^2) + 4b_1^2 h + 4b_4 M_x h)h + (4b_2^2 + 12b_3^2 L_g^2 mh)e_k$$
$$\leq c_1 h + c_2 e_k$$

taking $c_1 \geq (12b_3^2 L_g^2(M_x + T^2) + 4b_1^2 c_h + 4b_4 M_x c_h)$ and $c_2 \geq (4b_2^2 + 12b_3^2 L_g^2 m c_h)$.

Then, using that

$$|\Delta\mathbf{f}_k^T \Delta\mathbf{f}_j| \leq \|\Delta\mathbf{f}_k\| \cdot \|\Delta\mathbf{f}_j\| \leq \frac{\|\Delta\mathbf{f}_k\|^2 + \|\Delta\mathbf{f}_j\|^2}{2}$$

we arrive to

$$E[|\Delta\mathbf{f}_k^T \Delta\mathbf{f}_j|] \leq E[\frac{\|\Delta\mathbf{f}_k\|^2 + \|\Delta\mathbf{f}_j\|^2}{2}] \leq \frac{c_1 h + c_2 e_k + c_1 h + c_2 e_j}{2} \leq c_1 h + c_2 \max(e_j, e_k)$$

completing the proof. □

In a similar way to the previous result, the following lemma provides an upper bound for the term $E[|\Delta\mathbf{g}_k \Delta\mathbf{w}_k|]$ that appears in the sequence defined by Lemma A.1.

Lemma A.5. *Let $\Delta\mathbf{g}_k$ be defined by Eq.(18) and e_k by Eq.(21). Then, under Assumption 3.1 it results*

$$\mathrm{E}[\|\Delta\mathbf{g}_k\Delta\mathbf{w}_k\|^2] \leq L_g^2 e_k m h \tag{27}$$

for all $k \leq N$.

Proof Notice that

$$\|\Delta\mathbf{g}_k\| = \|\mathbf{g}(\mathbf{x}^q(t_k), t_k) - \mathbf{g}(\mathbf{x}^e(t_k), t_k)\| \leq L_g\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|$$

and then

$$\|\Delta\mathbf{g}_k\|^2 \leq L_g^2\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2$$

Taking expectation at both sides,

$$\mathrm{E}[\|\Delta\mathbf{g}_k\|^2] \leq L_g^2\mathrm{E}[\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2] = L_g^2 e_k$$

Then, using the fact that $\Delta\mathbf{g}_k$ and $\Delta\mathbf{w}_k$ are uncorrelated, it results

$$\begin{aligned}\mathrm{E}[\|\Delta\mathbf{g}_k\Delta\mathbf{w}_k\|^2] &\leq \mathrm{E}[\|\Delta\mathbf{g}_k\|^2 \cdot \|\Delta\mathbf{w}_k\|^2] = \mathrm{E}[\|\Delta\mathbf{g}_k\|^2] \cdot \mathrm{E}[\|\Delta\mathbf{w}_k\|^2] \\ &\leq L_g^2 e_k m h\end{aligned}$$

completing the proof. □

The following theorem, making use of the previous results, establishes an upper bound for the difference between QSS and EM solutions.

Theorem A.1. *Let $\mathbf{x}^q$ and $\mathbf{x}^e$ denote the QSS and the EM solutions of Eq.(4), given by Eqs.(6) and Eq.(17), respectively. Then, under Assumption 3.1 given $c_h > 0$ there exists a constant c_q such that*

$$\mathrm{E}[\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2] \leq c_q h \tag{28}$$

for $k = 0, \ldots, N$.

Proof According to Lemma A.1, the sequence defined by $e_k = \mathrm{E}[\|\mathbf{x}^q(t_k) - \mathbf{x}^e(t_k)\|^2]$ verifies Eq.(22). Using Eq.(25) of Lemma A.4 and Eq.(27) of Lemma A.5 and Eq.(23) of Lemma A.2 in that expression, it results

$$\begin{aligned}e_{k+1} &\leq e_k + h^2\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_k] + 2h^2\sum_{j=0}^{k-1}\mathrm{E}[\Delta\mathbf{f}_k^T\Delta\mathbf{f}_j] + \mathrm{E}[\|\Delta\mathbf{g}_k\Delta\mathbf{w}_k\|^2] + 2h\sqrt{\mathrm{E}[\|\Delta\mathbf{f}_k\|^2]\sum_{j=0}^{k}\mathrm{E}[\|\Delta\mathbf{g}_j\Delta\mathbf{w}_j\|^2]} \\ &\leq e_k + h^2(c_1 h + c_2 e_k) + 2h^2\sum_{j=0}^{k-1}(c_1 h + c_2\max(e_j, e_k)) + c_3 h e_k + 2h\sqrt{(c_1 h + c_2 e_k)\sum_{j=0}^{k} c_3 h e_j}\end{aligned}$$

Let $\bar{e}_k$ be a strictly increasing sequence verifying $\bar{e}_k \geq e_k$. Then, using that $\bar{e}_k \geq \bar{e}_j$ when $k > j$ and using also that $kh \leq T$, it results

$$e_{k+1} \leq \bar{e}_k + h^2(c_1 h + c_2\bar{e}_k) + 2h^2 k(c_1 h + c_2\bar{e}_k) + c_3 h\bar{e}_k + 2h\sqrt{(c_1 h + c_2\bar{e}_k)\sum_{j=0}^{k} c_3 h\bar{e}_j}$$

$$
\begin{aligned}
&\leq \bar{e}_k + h^2(c_1 h + c_2 \bar{e}_k) + 2h^2 k(c_1 h + c_2 \bar{e}_k) + c_3 h \bar{e}_k + 2h\sqrt{(c_1 h + c_2 \bar{e}_k) k c_3 h \bar{e}_k} \\
&\leq \bar{e}_k + h^2(c_1 h + c_2 \bar{e}_k) + 2hT(c_1 h + c_2 \bar{e}_k) + c_3 h \bar{e}_k + 2h\sqrt{(c_1 h + c_2 \bar{e}_k) c_3 T \bar{e}_k} \\
&\leq \bar{e}_k + h^2(c_1 h + c_2 \bar{e}_k) + 2hT(c_1 h + c_2 \bar{e}_k) + c_3 h \bar{e}_k + \sqrt{4c_1 c_3 T h^3 \bar{e}_k} + \sqrt{4c_2 c_3 T h^2 \bar{e}_k^2} \\
&\leq \bar{e}_k(1 + c_2 c_h h + 2hTc_2 + c_3 h + h\sqrt{4c_2 c_3 T}) + (c_1 c_h h^2 + 2Tc_1 h^2) + \sqrt{4c_1 c_3 T h^3 \bar{e}_k} \\
&\leq \bar{e}_k(1 + d_1 h) + d_2 h^{3/2}\sqrt{\bar{e}_k} + d_3 h^2
\end{aligned}
$$

for $h \leq c_h$, after defining $d_1 = c_2 c_h + 2Tc_2 + c_3 + \sqrt{4c_2 c_3 T}$, $d_2 = \sqrt{4c_1 c_3}$, and $d_3 = c_1 c_h + 2Tc_1$.

Then, we can define the sequence $\bar{e}_k$ as

$$
\bar{e}_{k+1} \triangleq \bar{e}_k(1 + d_1 h) + d_2 h^{3/2}\sqrt{\bar{e}_k} + d_3 h^2 \tag{29}
$$

with $\bar{e}_0 = 0$. We shall prove next that

$$
\bar{e}_k \leq (1 + d_1 h)^k r k h^2 \tag{30}
$$

for $k = 0, \ldots, N$ and any constant $r > 0$ verifying

$$
r > d_3 + d_2\sqrt{rT} \tag{31}
$$

Notice that Eq.(30) is verified for $k = 0$. Assume (for induction) that it is verified up to k. Then,

$$
\begin{aligned}
\bar{e}_{k+1} &= \bar{e}_k(1 + d_1 h) + d_2 h^{3/2}\sqrt{\bar{e}_k} + d_3 h^2 \\
&\leq ((1 + d_1 h)^k r k h^2)(1 + d_1 h) + d_2 h^{3/2}\sqrt{(1 + d_1 h)^k r k h^2} + d_3 h^2 \\
&\leq (1 + d_1 h)^{k+1} r k h^2 + d_2 h^2\sqrt{(1 + d_1 h)^k r k h} + d_3 h^2 \\
&\leq (1 + d_1 h)^{k+1} r(k+1)h^2 - (1 + d_1 h)^{k+1} r h^2 + d_2 h^2\sqrt{(1 + d_1 h)^k r T} + d_3 h^2 \\
&\leq (1 + d_1 h)^{k+1} r(k+1)h^2 + (d_2\sqrt{(1 + d_1 h)^k r T} + d_3 - (1 + d_1 h)^{k+1} r)h^2 \\
&\leq (1 + d_1 h)^{k+1} r(k+1)h^2
\end{aligned}
$$

where we used Eq.(31) in the last step. Then, Eq.(30) is verified for $k = 0, \ldots, N$.

Then,

$$
e_k \leq \bar{e}_k \leq (1 + d_1 h)^k r k h^2 \leq (1 + d_1 h)^{\frac{T}{h}} r T h
$$

and taking into account that

$$
(1 + d_1 h)^{\frac{T}{h}} r T < e^{T d_1} r T
$$

the proof is complete by taking $c_q = e^{T d_1} r T$. □

B Proof of Theorem 3.2

B.1 Auxiliary Results

The proof of Theorem 3.2 requires establishing some auxiliary results concerning the solutions of Eq. (11). These results, in turn, require the following proposition:

Proposition B.1. *Let $\sigma \in \mathbb{R}$, with $\sigma > 0$. Then, for any $h > 0$, we have*

$$
r(h) \triangleq \frac{h\mathrm{e}^{-\sigma h}}{1 - \mathrm{e}^{-\sigma h}} \leq \frac{1}{\sigma} \tag{32}
$$

and

$$\frac{h}{1-\mathrm{e}^{-\sigma h}} \leq h + \frac{1}{\sigma}. \tag{33}$$

Proof First, notice that

$$\lim_{h\to 0} r(h) = \frac{\lim_{h\to 0}(\mathrm{e}^{-\sigma h} - h\sigma \mathrm{e}^{-\sigma h})}{\lim_{h\to 0} \sigma \mathrm{e}^{-\sigma h}} = \frac{1}{\sigma}.$$

Also,

$$\lim_{h\to \infty} r(h) = 0 \leq \frac{1}{\sigma}.$$

Thus, Eq. (32) holds for $h \to 0$ and $h \to \infty$. Regarding the extrema, we have

$$r'(h) = \frac{(\mathrm{e}^{-\sigma h} - h\sigma \mathrm{e}^{-\sigma h})(1-\mathrm{e}^{-\sigma h}) - h\mathrm{e}^{-\sigma h}(\sigma \mathrm{e}^{-\sigma h})}{(1-\mathrm{e}^{-\sigma h})^2} = 0$$

from which,

$$r(h) = \frac{h\mathrm{e}^{-\sigma h}}{1-\mathrm{e}^{-\sigma h}} = \frac{(\mathrm{e}^{-\sigma h} - h\sigma \mathrm{e}^{-\sigma h})}{\sigma \mathrm{e}^{-\sigma h}} = \frac{1}{\sigma} - h \leq \frac{1}{\sigma}.$$

Since Eq. (32) holds for $h \to 0$, $h \to \infty$, and when $r'(h) = 0$, it is valid for all $h > 0$. Then, adding h to both sides of Eq. (33), we obtain

$$\frac{h\mathrm{e}^{-\sigma h}}{1-\mathrm{e}^{-\sigma h}} + h \leq \frac{1}{\sigma} + h \implies$$
$$\frac{h\mathrm{e}^{-\sigma h} + h(1-\mathrm{e}^{-\sigma h})}{1-\mathrm{e}^{-\sigma h}} = \frac{h}{1-\mathrm{e}^{-\sigma h}} \leq \frac{1}{\sigma} + h,$$

showing that Eq. (33) also holds, completing the proof. □

Lemma B.1. *Let $\hat{\mathbf{x}}_h(t)$, $\hat{\mathbf{x}}_{2h}(t)$ be the solutions of the system of Eq. (11) for a Wiener process $\mathbf{w}(t) \in \mathbb{R}^m$ with incremental covariance $I^{m\times m}\mathrm{d}t$ using sampling periods h and $2h$, respectively. Then, assuming that A is a Hurwitz matrix, for any $N \in \mathbb{N}$ it results*

$$\mathrm{E}(||\hat{\mathbf{x}}_{2h}(2Nh) - \hat{\mathbf{x}}_h(2Nh)||) \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h \tag{34}$$

with c_1 and c_2 defined in Eq. (13).

Proof Defining $T \triangleq 2Nh$, the values of $\hat{\mathbf{x}}_h(T)$, $\hat{\mathbf{x}}_{2h}(T)$ can be obtained as

$$\hat{\mathbf{x}}_h(T) = \mathrm{e}^{AT}\mathbf{x}(0) + \int_0^T \mathrm{e}^{A(T-\tau)} B \frac{\Delta \mathbf{w}_1(\tau)}{h} \mathrm{d}\tau; \tag{35}$$

and

$$\hat{\mathbf{x}}_{2h}(T) = \mathrm{e}^{AT}\mathbf{x}(0) + \int_0^T \mathrm{e}^{A(T-\tau)} B \frac{\Delta \mathbf{w}_2(\tau)}{2h} \mathrm{d}\tau. \tag{36}$$

where $\Delta\mathbf{w}_1$ and $\Delta\mathbf{w}_2$ are defined as in Eq. (5) with period h and $2h$, respectively, i.e.,

$$\Delta \mathbf{w}_1(t) = \mathbf{w}(t_k + h) - \mathbf{w}(t_k); \quad t \in [t_k; t_k + h)$$

and

$$\Delta \mathbf{w}_2(t) = \mathbf{w}(t_k + 2h) - \mathbf{w}(t_k); \quad t \in [t_k; t_k + h).$$

Subtracting Eq. (35) from Eq. (36), we obtain

$$\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T) = \int_0^T \mathrm{e}^{A(T-\tau)} \frac{B}{h} \left[\frac{\Delta \mathbf{w}_2(\tau)}{2} - \Delta \mathbf{w}_1(\tau) \right] \mathrm{d}\tau.$$

Defining $\mathbf{z}(t) \triangleq \dfrac{\Delta \mathbf{w}_2(t)}{2} - \Delta \mathbf{w}_1(t)$, taking $k = 2j,\ j = 0, 1, \ldots, N$, and taking into account the sampling periods considered we can rewrite the above as

$$\begin{aligned}
\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T) &= \sum_{j=0}^{N-1} \int_{t_{2j}}^{t_{2j+2}} \mathrm{e}^{A(T-\tau)} \frac{B}{h} \mathbf{z}(\tau) \mathrm{d}\tau \\
&= \frac{1}{h} \sum_{j=0}^{N-1} \left[\int_{t_{2j}}^{t_{2j+1}} \mathrm{e}^{A(T-\tau)} B \mathbf{z}(\tau) \mathrm{d}\tau + \int_{t_{2j+1}}^{t_{2j+2}} \mathrm{e}^{A(T-\tau)} B \mathbf{z}(\tau) \mathrm{d}\tau \right] \\
&= \frac{1}{h} \sum_{j=0}^{N-1} \left[\int_{t_{2j}}^{t_{2j+1}} \mathrm{e}^{A(T-\tau)} B \mathbf{z}_{2j} \mathrm{d}\tau - \int_{t_{2j+1}}^{t_{2j+2}} \mathrm{e}^{A(T-\tau)} B \mathbf{z}_{2j} \mathrm{d}\tau \right].
\end{aligned}$$

In the last equality we use the fact that

$$z(t) = \frac{\Delta \mathbf{w}_2(t)}{2} - \Delta \mathbf{w}_1(t) = \begin{cases} \dfrac{\mathbf{w}(t_{k+2})}{2} - \mathbf{w}(t_{k+1}) + \dfrac{\mathbf{w}(t_k)}{2} \triangleq \mathbf{z}_k, & \text{if } t \in [t_k, t_{k+1}) \\ -\dfrac{\mathbf{w}(t_{k+2})}{2} + \mathbf{w}(t_{k+1}) - \dfrac{\mathbf{w}(t_k)}{2} = -\mathbf{z}_k, & \text{if } t \in [t_{k+1}, t_{k+2}). \end{cases}$$

Then, considering the substitution $s = T - \tau$, it results

$$\begin{aligned}
\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T) &= \frac{1}{h} \sum_{j=0}^{N-1} \left[\int_{t_{2j}}^{t_{2j+1}} \mathrm{e}^{A(T-\tau)} \mathrm{d}\tau - \int_{t_{2j+1}}^{t_{2j+2}} \mathrm{e}^{A(T-\tau)} \mathrm{d}\tau \right] B z_{2j} \\
&= \frac{1}{h} \sum_{j=0}^{N-1} \left[\int_{T-t_{2j+2}}^{T-t_{2j+1}} \mathrm{e}^{As} \mathrm{d}s - \int_{T-t_{2j+1}}^{T-t_{2j}} \mathrm{e}^{As} \mathrm{d}s \right] B z_{2j}.
\end{aligned}$$

Now, being $s_k = s_{2j} = T - t_{2j+2} = T - t_k - 2h$, we get

$$\begin{aligned}
\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T) &= \frac{1}{h} \sum_{k=0}^{2N-2} A^{-1} \left[\left(\mathrm{e}^{A(s_k+2h)} - \mathrm{e}^{A(s_k+h)} \right) - \left(\mathrm{e}^{A(s_k+h)} - \mathrm{e}^{As_k} \right) \right] B z_k \\
&= \frac{A^{-1}(\mathrm{e}^{Ah} - I)^2}{h} \sum_{k=0}^{2N-2} \mathrm{e}^{As_k} B z_k. \qquad (37)
\end{aligned}$$

Define

$$\rho \triangleq \sum_{k=0}^{2N-2} \mathrm{e}^{As_k} B z_k,$$

then,

$$\rho^T \rho = \sum_{k=0}^{2N-2} z_k^T B^T e^{A^T s_k} \sum_{j=0}^{2N-2} e^{As_j} B z_j = \sum_{k=0}^{2N-2} \sum_{j=0}^{2N-2} z_k^T B^T e^{A^T s_k} e^{As_j} B z_j.$$

Taking expectation at both sides, it results

$$\mathrm{E}[\rho^T\rho] = \sum_{k=0}^{2N-2}\sum_{j=0}^{2N-2}\mathrm{E}[z_k^T B^T e^{A^T s_k} e^{As_j} B z_j] = \sum_{k=0}^{2N-2}\mathrm{E}[z_k^T B^T e^{A^T s_k} e^{As_k} B z_k]$$
$$= \sum_{k=0}^{2N-2}\mathrm{tr}(B^T e^{A^T s_k} e^{As_k} B\mathrm{E}[z_k z_k^T]) = \sum_{k=0}^{2N-2}\mathrm{tr}(B^T e^{A^T s_k} e^{As_k} B I\frac{h}{2})$$

where we used that $\mathbf{z}_k = \frac{1}{2}(\Delta\mathbf{w}_1(t_{k+1}) - \Delta\mathbf{w}_1(t_k))$ implying that $\mathrm{E}[\mathbf{z}_k^T M\mathbf{z}_j] = 0$ for $j \neq k$. We also used the property of the quadratic form $\mathrm{E}[\mathbf{z}_k^T M\mathbf{z}_k] = \mathrm{tr}(M\mathrm{E}[\mathbf{z}_k\mathbf{z}_k^T])$ and that $\mathrm{E}[\mathbf{z}_k\mathbf{z}_k^T] = I\frac{h}{2}$ since $\mathrm{E}[\Delta\mathbf{w}_1\Delta\mathbf{w}_1^T] = Ih$.

Then, taking into account that $\mathrm{tr}(M) \leq m\cdot||M||$ for $M\in\mathbb{R}^{m\times m}$, it follows that

$$\mathrm{E}[\rho^T\rho] \leq \sum_{k=0}^{2N-2}\frac{mh}{2}\cdot||B^T e^{A^T s_k} e^{As_k} B|| \leq \frac{mh}{2}\cdot(||B||\cdot||V||\cdot||V^{-1}||)^2\sum_{k=0}^{\infty}||e^{\Lambda s_k}||^2$$
$$= \frac{m}{2}(||B||\cdot v)^2\frac{h}{(1-||e^{\Lambda h}||^2)} = \frac{m}{2}(||B||\cdot v)^2\frac{h}{(1-\mathrm{e}^{-2\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)\cdot h})}$$

where we also used that $||\mathrm{e}^{A^T s_k}|| = ||\mathrm{e}^{As_k}|| \leq ||V||\cdot||V^{-1}||\cdot||\mathrm{e}^{\Lambda s_k}|| \triangleq v\cdot||\mathrm{e}^{\Lambda s_k}||$. Here, $\Lambda = V^{-1}AV$ is the Jordan canonical decomposition of matrix A and it results that $||\mathrm{e}^{\Lambda s_k}|| < 1$ (since A is Hurwitz and thus $\mathbb{R}\mathrm{e}(\lambda_i) < 0$).

Using Eq. (33) (Proposition B.1) in the last expression, it results

$$\mathrm{E}[\rho^T\rho] \leq \frac{m}{2}(||B||\cdot v)^2(\frac{1}{2\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)} + h).$$

Then, using Jensen's inequality we obtain

$$\mathrm{E}[||\rho||]^2 \leq \mathrm{E}[||\rho||^2] \leq \frac{m}{2}(||B||\cdot v)^2(\frac{1}{2\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)} + h)$$

and

$$\mathrm{E}[||\rho||] \leq \sqrt{\frac{m}{2}}(||B||\cdot v)\sqrt{\frac{1}{2\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)} + h}$$
$$\leq \sqrt{\frac{m}{2}}(||B||\cdot v)\cdot\left(\sqrt{\frac{1}{2\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}} + \sqrt{h}\right)$$
$$\leq \frac{1}{2}\sqrt{\frac{m}{\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}}(||B||\cdot v)\cdot\left(1 + \sqrt{2h\cdot\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}\right) \tag{38}$$

Using norm at both sides of Eq. (37) we then obtain,

$$||\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T)|| \leq ||\frac{A^{-1}(\mathrm{e}^{Ah} - I)^2}{h}||\cdot||\rho||.$$

and taking expectation at both sides, and replacing with Eq. (38) it results

$$\mathrm{E}[||\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T)||] \leq ||\frac{A^{-1}(\mathrm{e}^{Ah} - I)^2}{h}||\cdot\mathrm{E}[||\rho||]$$
$$\leq ||\frac{A^{-1}(\mathrm{e}^{Ah} - I)^2}{h}||\cdot\frac{1}{2}\sqrt{\frac{m}{\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}}(||B||\cdot v)\cdot(1 + \sqrt{2h\cdot\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}). \tag{39}$$

Using that

$$||A^{-1}(\mathrm{e}^{Ah} - I)|| = ||\int_0^h \mathrm{e}^{A\tau}\mathrm{d}\tau|| \leq \int_0^h ||\mathrm{e}^{A\tau}||\mathrm{d}\tau = \int_0^h ||V\mathrm{e}^{\Lambda\tau}V^{-1}||\mathrm{d}\tau$$
$$\leq ||V|| \cdot ||V^{-1}|| \int_0^h ||\mathrm{e}^{\Lambda\tau}||\mathrm{d}\tau \leq v \cdot h$$

and also

$$||\mathrm{e}^{Ah} - I|| \leq ||A|| \cdot ||A^{-1}(\mathrm{e}^{Ah} - I)|| \leq ||A|| \cdot v \cdot h$$

we arrive to

$$||\frac{A^{-1}(\mathrm{e}^{Ah} - I)^2}{h}|| \leq ||A|| \cdot v^2 h \leq ||\Lambda|| \cdot v^3 h = \max_i |\lambda_i| \cdot v^3 h$$

and replacing with this expression in Eq. (39), we arrive to

$$||\hat{\mathbf{x}}_{2h}(T) - \hat{\mathbf{x}}_h(T)|| \leq \frac{1}{2} \cdot v^4 \sqrt{\frac{m \cdot (\max_i |\lambda_i|)^2}{\min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}} \cdot ||B|| \cdot \left(1 + \sqrt{2h \cdot \min_i(|\mathbb{R}\mathrm{e}(\lambda_i)|)}\right) \cdot h$$
$$= c_1 \cdot (1 + c_2\sqrt{h}) \cdot h$$

completing the proof. □

Replacing h by $h/2$ in Eq. (34) the following corollary is obtained:

Corollary B.1. *Let $\hat{\mathbf{x}}_h(t)$, $\hat{\mathbf{x}}_{h/2}(t)$ be the solutions of the system of Eq. (11) for a Wiener process $\mathbf{w}(t) \in \mathbb{R}^m$ with incremental covariance $I^{m\times m}\mathrm{d}t$ using sampling periods h and $h/2$, respectively. Then, for any $N \in \mathbb{N}$ it results*

$$\mathrm{E}(||\hat{\mathbf{x}}_h(Nh) - \hat{\mathbf{x}}_{h/2}(Nh)||) \leq c_1 \cdot (1 + c_2\sqrt{h/2}) \cdot h/2 \tag{40}$$

with c_1 and c_2 given by Eq. (13).

Using the previous result, the following lemma establishes a bound for the difference between the solutions of the original SDE and the approximated ODE given by Eq. (11).

Lemma B.2. *Let $\mathbf{x}(t)$ be the solution of the SDE of Eq. (9) and let $\hat{\mathbf{x}}(t)$ be the solution of Eq. (11) using a sampling period $h > 0$. Then, for any $t_j = jh$ with $j \in \mathbb{N}$,*

$$\mathrm{E}[||\hat{\mathbf{x}}(t_j) - \mathbf{x}(t_j)||] \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h \tag{41}$$

with c_1 and c_2 given by Eq. (13).

Proof Let $\hat{\mathbf{x}}_k(t)$ be the solution of Eq. (11) using a sampling period $h/2^k$. Thus, from Corollary B.1, it results that

$$\mathrm{E}[||\hat{\mathbf{x}}_{k-1}(t_j) - \hat{\mathbf{x}}_k(t_j)||] \leq c_1 \cdot (1 + c_2\sqrt{\frac{h}{2^k}}) \cdot \frac{h}{2^k} \tag{42}$$

for $k = 1, \ldots$. Then, for any $M \in \mathbb{Z}$, it results

$$\mathrm{E}[||\hat{\mathbf{x}}_0(t_j) - \hat{\mathbf{x}}_M(t_j)||] = \mathrm{E}\left[\left\|\sum_{k=1}^{M} \hat{\mathbf{x}}_{k-1}(t_j) - \hat{\mathbf{x}}_k(t_j)\right\|\right] \leq \sum_{k=1}^{M} \mathrm{E}\left[||\hat{\mathbf{x}}_{k-1}(t_j) - \hat{\mathbf{x}}_k(t_j)||\right].$$

Taking into account that $\hat{\mathbf{x}}(t) = \hat{\mathbf{x}}_{k=0}(t)$ and Eq. (42), it results

$$\mathrm{E}[||\hat{\mathbf{x}}(t_j) - \hat{\mathbf{x}}_M(t_j)||] \leq \sum_{k=1}^{M} c_1 \cdot (1 + c_2\sqrt{\frac{h}{2^k}}) \cdot \frac{h}{2^k} \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h \cdot \sum_{k=1}^{M} \frac{1}{2^k}.$$

Then, since $\sum_{k=1}^{M} \frac{1}{2^k} \leq 1$, it results

$$\mathrm{E}[||\hat{\mathbf{x}}(t_j) - \hat{\mathbf{x}}_M(t_j)||] \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h.$$

This bound also holds for $M \to \infty$, i.e., when the sampling period $h \to 0$.

From the proof of Theorem 3.1 of (Baccouch et al. 2021), it can be verified that for any $t_0 \leq t \leq T$

$$\mathrm{E}(||\hat{\mathbf{x}}(t) - \mathbf{x}(t)||^2) \leq 2m^2 nT\mathrm{e}^{2L^2T^2n}h^2. \tag{43}$$

meaning that the solutions of Eq. (11) converge to those of the SDE (9) as $h \to 0$. Then, for a given t_j, it results

$$\lim_{M\to\infty} \mathrm{E}[||\hat{\mathbf{x}}_M(t_j) - \mathbf{x}(t_j)||] = 0.$$

Then, given $\varepsilon > 0$ there exists M such that

$$\mathrm{E}[||\hat{\mathbf{x}}_M(t_j) - \mathbf{x}(t_j)||] \leq \varepsilon.$$

Then,

$$\mathrm{E}[||\hat{\mathbf{x}}(t_j) - \mathbf{x}(t_j)||] \leq \mathrm{E}[||\hat{\mathbf{x}}(t_j) - \mathbf{x}_M(t_j)||] + \mathrm{E}[||\hat{\mathbf{x}}_M(t_j) - \mathbf{x}(t_j)||] \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h + \varepsilon$$

Since this bound holds for any $\varepsilon > 0$ then

$$\mathrm{E}[||\hat{\mathbf{x}}(t) - \mathbf{x}(t)||] \leq c_1 \cdot (1 + c_2\sqrt{h}) \cdot h$$

completing the proof. □

B.2 Proof of Theorem 3.2

Using the result of Lemma B.2 we can now prove Theorem 3.2.

Proof Let $\hat{x}(t)$ be the solution of the approximate system of Eq. (11) with a sampling period h. Defining $\tilde{\mathbf{x}}(t) \triangleq \mathbf{x}^q(t) - \hat{\mathbf{x}}(t)$, and subtracting Eq. (11) from Eq. (10), we obtain

$$\dot{\tilde{\mathbf{x}}}(t) = A(\tilde{\mathbf{x}}(t) + \Delta\mathbf{x}(t))$$

where $\Delta\mathbf{x}(t) = \mathbf{q}(t) - \mathbf{x}^q(t)$. Then, from Theorem 1 of (Kofman et al. 2007) it results

$$|\tilde{\mathbf{x}}(t)| \preceq |V| \cdot |\mathbb{R}\mathrm{e}(\Lambda)^{-1}| \cdot |V^{-1}| \cdot |A| \cdot |\Delta\mathbf{Q}|.$$

Applying norm at both sides, it follows that

$$||\tilde{\mathbf{x}}(t)|| \leq \left\||V| \cdot |\mathbb{R}\mathrm{e}(\Lambda)^{-1}| \cdot |V^{-1}| \cdot |A|\right\| \cdot ||\Delta\mathbf{Q}|| = c_3 \cdot ||\Delta\mathbf{Q}||.$$

Then, combining this inequality with the result of Lemma B.2, we arrive to

$$\begin{aligned}\mathrm{E}\left[||\mathbf{x}^q(t_j) - \mathbf{x}(t_j)||\right] &\leq \mathrm{E}\left[||\mathbf{x}^q(t_j) - \hat{\mathbf{x}}(t_j)||\right] + \mathrm{E}\left[||\hat{\mathbf{x}}(t_j) - \mathbf{x}(t_j)||\right] \\ &\leq c_3 \cdot ||\Delta\mathbf{Q}|| + c_1 \cdot (1 + c_2\sqrt{h}) \cdot h\end{aligned}$$

completing the proof. □