Discrete-time Solutions to the Continuous-time Differential Lyapunov Equation With Applications to Kalman Filtering

Patrik Axelsson and Fredrik Gustafsson

Patrik Axelsson and Fredrik Gustafsson, Discrete-time Solutions to the Continuous-time Differential Lyapunov Equation With Applications to Kalman Filtering, 2015, IEEE Transactions on Automatic Control, (60), 3, 632-643.

http://dx.doi.org/10.1109/TAC.2014.2353112

Discrete-time Solutions to the Continuous-time Differential Lyapunov Equation With Applications to Kalman Filtering

Patrik Axelsson, and Fredrik Gustafsson, *Fellow, IEEE*

***Abstract*—Prediction and filtering of continuous-time stochastic processes often require a solver of a continuous-time differential Lyapunov equation (CDLE), for example the time update in the Kalman filter. Even though this can be recast into an ordinary differential equation (ODE), where standard solvers can be applied, the dominating approach in Kalman filter applications is to discretize the system and then apply the discrete-time difference Lyapunov equation (DDLE). To avoid problems with stability and poor accuracy, oversampling is often used. This contribution analyzes over-sampling strategies, and proposes a novel low-complexity analytical solution that does not involve oversampling. The results are illustrated on Kalman filtering problems in both linear and nonlinear systems.**

Keywords*—*Continuous time systems, Discrete time systems, Kalman filters, Sampling methods

I. Introduction

NUMERICAL SOLVERS for ordinary differential equations (ODE) is a well studied area [1]. The related area of Kalman filtering (state prediction and state estimation) in continuous-time models was also well studied during the first two decades of the Kalman filter, see for instance [2], while the more recent literature such as the standard reference [3] focuses on discrete time filtering only. A specific example, with many applications in practice, is Kalman filtering based on a continuous-time state space model with discrete-time measurements, known as continuous-discrete filtering. The Kalman filter (KF) here involves a time update that integrates the first and second order moments from one sample time to the next one. The second order moment is a covariance matrix, and it governs a continuous-time differential Lyapunov equation (CDLE). The problem can easily be recast into a vectorized ODE problem and standard solvers can be applied. For linear ODE's, the time update of the linear KF can thus be solved analytically, and for nonlinear ODE's, the time update of the extended KF has a natural approximation in continuous-time. One problem is the large dimension of the resulting ODE. Another possible explanation why the continuous-time update is not used is the common use of discrete-time models in Kalman filter applications, so practitioners often tend to discretize the state space model first to fit the discrete-time Kalman filter time update. Despite a closed form solution exists, this involves approximations that lead to well known problems with accuracy and stability. The *ad-hoc* remedy is to oversample the system, so a large number of small time updates are taken between the sampling times of the observations.

In literature, different methods are proposed to solve the continuous-discrete nonlinear filtering problem using extended Kalman filters (EKF). A common way is to use a first or second order Taylor approximation as well as a Runge-Kutta method in order to integrate the first order moments, see e.g. [4]–[6]. They all have in common that the CDLE is replaced by the discrete-time difference Lyapunov equation (DDLE), used in discrete-time Kalman filters. A more realistic way is to solve the CDLE as is presented in [7], [8], where the first and second order moments are integrated numerically. A comparison between different solutions is presented in [9], where the method proposed by the authors discretizes the stochastic differential equation (SDE) using a Runge-Kutta solver. The other methods in [9] have been proposed in the literature before, e.g. [4], [8]. Related work using different approximations to continuous integration problems in nonlinear filtering also appears in [10], [11] for unscented Kalman filters and [12] for cubature Kalman filters.

This contribution takes a new look at this fundamental problem. First, we review different approaches for solving the CDLE in a coherent mathematical framework. Second, we analyze in detail the stability conditions for oversampling, and based on this we can explain why even simple linear models need a large rate of oversampling. Third, we make a new straightforward derivation of a low-complexity algorithm to compute the solution with arbitrary accuracy. Numerical stability and computational complexity is analyzed for the different approaches. It turns out that the low-complexity algorithm has better numerical properties compared to the other methods, and at the same time a computational complexity in the same order. Fourth, the methods are extended to nonlinear system where the extended Kalman filter (EKF) is used. We illustrate the results on both a simple second order linear spring-damper system, and a nonlinear spring-damper system relevant for mechanical systems, in particular robotics.

II. Mathematical Framework and Background

A. Linear Stochastic Differential Equations

Consider the linear stochastic differential equation (SDE)

$$dx(t) = Ax(t)dt + Gd\beta(t), \tag{1}$$

for $t \geq 0$, where $x(t) \in \mathbb{R}^{n_x}$ is the state vector and $\beta(t) \in \mathbb{R}^{n_\beta}$ is a vector of Wiener processes with $E\left[d\beta(t)d\beta(t)^\mathsf{T}\right] = Qdt$. The matrices $A \in \mathbb{R}^{n_x \times n_x}$ and $G \in \mathbb{R}^{n_x \times n_\beta}$ are here assumed to be constants, but they can also be time varying. It is also possible to include a control signal $u(t)$ in (1) giving a bit more complicated expression for the first order moment.

Given an initial state $\hat{x}(0)$ with covariance $P(0)$, we want to solve the SDE to get $\hat{x}(t)$ and $P(t)$ at an arbitrary time instance. By multiplying both sides with the integrating factor e^{-As} and integrating over the time interval gives

$$x(t) = e^{At}x(0) + \underbrace{\int_0^t e^{A(t-s)}Gd\beta(s)}_{v_d(t)}. \tag{2}$$

The goal is to get a discrete-time update of the mean and covariance, from $\hat{x}\big(kh\big)$ and $P\big(kh\big)$ to $\hat{x}\big((k+1)h\big)$ and $P\big((k+1)h\big)$, respectively. The time interval h may correspond to one sample interval, or be a fraction of it in the case of oversampling. The latter case will be discussed in detail later on. For simplicity, the time interval $[kh, (k+1)h]$ will be denoted as $[0, t]$ below.

The discrete-time equivalent noise $v_d(t)$ has covariance given by

$$Q_d(t) = \int_0^t e^{A(t-s)}GQG^\mathsf{T}e^{A^\mathsf{T}(t-s)}\,\mathrm{d}s = \int_0^t e^{A\tau}GQG^\mathsf{T}e^{A^\mathsf{T}\tau}\,\mathrm{d}\tau, \tag{3}$$

This work was supported by the Vinnova Excellence Center LINK-SIC.

P. Axelsson (e-mail: patrik.axelsson@liu.se, telephone: +46 13 281 365) and F. Gustafsson (e-mail: fredrik.gustafsson@liu.se, telephone: +46 13 282 706) are with the Division of Automatic Control, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden. Fax +46 13 139 282.

Corresponding author: P. Axelsson (e-mail: patrik.axelsson@liu.se).

We immediately get an expression for the first and second order moments of the SDE solution over one time interval as

$$\hat{x}(t) = e^{At}\hat{x}(0), \tag{4a}$$
$$P(t) = e^{At}P(0)e^{A^\mathsf{T}t} + Q_d(t). \tag{4b}$$

From (2) and (3), we can also recover the continuous-time update formulas

$$\dot{\hat{x}}(t) = A\hat{x}(t), \tag{5a}$$
$$\dot{P}(t) = AP(t) + P(t)A^\mathsf{T} + GQG^\mathsf{T}, \tag{5b}$$

by, a bit informally, taking the expectation of (2) and then dividing both sides with t and letting $t \to 0$. A formal proof of (5) can be based on Itô's lemma, see [2]. Equation (5a) is an ordinary ODE and (5b) is the continuous-time differential Lyapunov equation (CDLE).

Thus, there are two conceptually different alternatives. Either, solve the integral (3) defining $Q_d(t)$ and use (4), or solve the ODE and CDLE in (5). These two well-known alternatives are outlined below.

B. Matrix Fraction Decomposition

There are two ways to compute the integral (3) described in literature. Both are based on computing the matrix exponential of the matrix

$$H = \begin{pmatrix} A & GQG^\mathsf{T} \\ 0 & -A^\mathsf{T} \end{pmatrix}. \tag{6}$$

The result is a block matrix in the form

$$e^{Ht} = \begin{pmatrix} M_1(t) & M_2(t) \\ 0 & M_3(t) \end{pmatrix}, \tag{7}$$

where the structure implies that $M_1(t) = e^{At}$ and $M_3(t) = e^{-A^\mathsf{T}t}$. As shown in [13], the solution to (3) can be computed as

$$Q_d(t) = M_2(t)M_1(t)^\mathsf{T} \tag{8}$$

This is immediately verified by taking the time derivative of the definition (3) and the matrix exponential (7), and verifying that the involved Taylor expansions are equal.

Another alternative known as matrix fraction decomposition, which solves a matrix valued ODE, given in [14], [15], is to compute $P(t)$ directly. Using the initial conditions $\begin{pmatrix} P(0) & I \end{pmatrix}^\mathsf{T}$ for the ODE gives

$$P(t) = \underbrace{\left(M_1(t)P(0) + M_2(t)\right)}_{\triangleq C(t)} M_3(t)^{-1}. \tag{9}$$

The two alternatives in (8) and (9) are apparently algebraically the same.

There are also other alternatives described in literature. First, the integral in (3) can of course be solved with numerical methods such as the trapezoidal method or the rectangle method. In [16] the integral is solved analytically in the case that A is diagonalizable. However, not all matrices are diagonalizable, and even in such cases, this method is not numerically stable [17].

C. Vectorization Method

The ODEs for the first and second order moments in (5) can be solved using a method based on vectorization. The vectorization approach for matrix equations is well known and especially for the CDLE, see e.g. [18], [19]. The method uses the fact that (5) can be converted to one single ODE by introducing an extended state vector

$$z(t) = \begin{pmatrix} z_x(t) \\ z_P(t) \end{pmatrix} = \begin{pmatrix} x(t) \\ \operatorname{vech} P(t) \end{pmatrix}, \tag{10}$$
$$\dot{z}(t) = \begin{pmatrix} A & 0 \\ 0 & A_P \end{pmatrix} z(t) + \begin{pmatrix} 0 \\ \operatorname{vech} GQG^\mathsf{T} \end{pmatrix} = A_z z(t) + B_z, \tag{11}$$

where $A_P = D^\dagger \left(I \otimes A + A \otimes I\right) D$. Here, vech denotes the half-vectorisation operator, $\otimes$ is the Kronecker product and D is a duplication matrix, see Appendix A for details.

The solution of the ODE (11) is given by [20]

$$z(t) = e^{A_z t}z(0) + \int_0^t e^{A_z(t-\tau)}\,\mathrm{d}\tau B_z. \tag{12}$$

One potentially prohibitive drawback with the solution in (12) is its computational complexity, in particular for the matrix exponential. The dimension of the extended state z is $n_z = n_x + n_x\left(n_x + 1\right)/2$, giving a computational complexity of $\mathcal{O}(n_x^6)$. Section IV presents a way to rewrite (12) to give a complexity of $\mathcal{O}(n_x^3)$ instead of $\mathcal{O}(n_x^6)$.

D. Discrete-time Recursion of the CDLE

The solution in (4b) using (8), the matrix fraction decomposition in (9), and the ODE (12) evaluated at the discrete-time instances $t = kh$ and $t = (k+1)h$ give the following recursive update formulas

$$P((k+1)h) = e^{Ah}P(kh)e^{A^\mathsf{T}h} + M_2(h)M_1(h)^\mathsf{T} \tag{13}$$
$$P((k+1)h) = \left(M_1(h)P(kh) + M_2(h)\right)M_3(h)^{-1} \tag{14}$$
$$z((k+1)h) = e^{A_z h}z(kh) + \int_0^h e^{A_z\tau}\,\mathrm{d}\tau B_z, \tag{15}$$

which can be used in the Kalman filter time update.

E. The Matrix Exponential

Sections II-A to II-C shows that the matrix exponential function is a working horse to solve the linear SDE. At this stage, numerical routines for the matrix exponential are important to understand. One key approach is based on the following identity and Taylor expansion, [21]

$$\begin{aligned} e^{Ah} &= \left(e^{Ah/m}\right)^m \\ &\approx \left(I + \left(\frac{Ah}{m}\right) + \cdots + \frac{1}{p!}\left(\frac{Ah}{m}\right)^p\right)^m \triangleq e_{p,m}(Ah). \end{aligned} \tag{16}$$

In fact, the Taylor expansion is a special case of a more general Padé approximation of $e^{Ah/m}$ [21], but this does not affect the discussion here.

The eigenvalues of Ah/m are the eigenvalues of A scaled with h/m, and thus they can be arbitrarily small if m is chosen large enough for any given h. Further, the p'th order Taylor expansion converges faster for smaller eigenvalues of Ah/m. Finally, the power function M^m is efficiently implemented by squaring the matrix M in total $\log_2(m)$ times, assuming that m is chosen to be a power of 2. We will denote this approximation with $e_{p,m}(Ah)$.

A good approximation $e_{p,m}(Ah)$ is characterized by the following properties:

- Stability. If A has all its eigenvalues in the left half plane, then $e_{p,m}(Ah)$ should have all its eigenvalues inside the unit circle.
- Accuracy. If p and m are chosen large enough, the error $\left\|e^{Ah} - e_{p,m}(Ah)\right\|$ should be small.

Since the Taylor expansion converges, we have trivially that

$$\lim_{p\to\infty} e_{p,m}(Ah) = \left(e^{Ah/m}\right)^m = e^{Ah}. \tag{17a}$$

From the property $\lim_{x\to\infty}(1 + a/x)^x = e^a$, we also have

$$\lim_{m\to\infty} e_{p,m}(Ah) = e^{Ah}. \tag{17b}$$

Finally, from [17] we have that

$$\left\|e^{Ah} - e_{p,m}(Ah)\right\| \leq \frac{\|A\|^{p+1}\,h^{p+1}}{m^p(p+1)!}e^{\|A\|h}. \tag{17c}$$

However, for any finite p and $m > 1$, then all terms in the binomial expansion of $e_{p,m}(Ah)$ are different from the Taylor expansion of e^{Ah}, except for the first two terms which are always $I + Ah$.

The complexity of the approximation $e_{p,m}(Ah)$, where $A \in \mathbb{R}^{n_x \times n_x}$, is in the order of $(\log_2(m) + p)\, n_x^3$, where pn_x^3 multiplications are required to compute A^p and $\log_2(m) n_x^3$ multiplications are needed for squaring the Taylor expansion $\log_2(m)$ times.

Standard numerical integration routines can be recast into this framework as well. For instance, a standard tuning of the fourth order Runge-Kutta method for a linear ODE results in $e_{4,1}(Ah)$.

F. Solution using Approximate Discretization

We have now outlined three methods to compute the exact time update in the discrete-time Kalman filter. These should be equivalent up to numerical issues, and will be treated as one approach in the sequel.

Another common approach in practice, in particular in Kalman filter applications, is to assume that the noise is piece-wise constant giving the discrete-time system

$$x(k+1) = F_h x(k) + G_h v_h(k), \tag{18a}$$
$$\operatorname{cov}(v_h(k)) = Q_h, \tag{18b}$$

where $F_h = e_{p,m}(Ah)$, $G_h = \int_0^h e^{At} dt G$, and $Q_h = hQ$. The discrete-time Kalman filter update equations

$$\hat{x}(k+1) = F_h \hat{x}(k), \tag{19a}$$
$$P(k+1) = F_h P(k) F_h^{\mathsf{T}} + G_h Q_h G_h^{\mathsf{T}}, \tag{19b}$$

are then used, where (19a) is a difference equation and (19b) is the discrete-time difference Lyapunov equation (DDLE). The update equations (19) are exact for the discrete-time model (18). However, there are several approximations involved in the discretization step:

- First, $F_h = e_{p,m}(Ah)$ is an approximation of the exact solution given by $F_h = e^{Ah}$. It is quite common in practice to use Euler sampling defined by $F_h = I + Ah = e_{1,1}(Ah)$.
- Even without process noise, the update formula for P in (19b) is not equivalent to (5b).
- The discrete-time noise $v_h(t)$ is an aggregation of the total effect of the Wiener process $d\beta(t)$ during the interval $[t, t+h]$, as given in (3). The conceptual drawback is that the Wiener process $d\beta(t)$ is not aware of the sampling time chosen by the user.

One common remedy is to introduce oversampling. This means that (19) is iterated m times using the sampling time h/m. When oversampling is used, the covariance matrix for the discrete-time noise $v_h(k)$ should be scaled as $Q_h = hQ/m$. In this way, the problems listed above will asymptotically vanish as m increases. However, as we will demonstrate, quite large an m can be needed even for some quite simple systems.

G. Summary of Contributions

- Section III gives explicit conditions for an upper bound of the sample time h such that a stable continuous-time model remains stable after discretization. The analysis treats stability of both x and P, for the case of Euler sampling $e_{1,m}(A)$, for the solution of the SDE given by the ODE (11). Results for $p > 1$ are also briefly discussed. See Table I for a summary when the vectorized solution is used.
- Section IV presents a reformulation of the solution to the ODE (11), where the computational complexity has been decreased from $(\log_2(m) + p)\left(n_x^2/2\right)^3$ to $(\log_2(m) + p + 43)\, n_x^3$.

TABLE I. SUMMARY OF APPROXIMATIONS $e_{p,m}(Ah)$ OF e^{Ah}. THE STABILITY REGION ($h < h_{\max}$) IS PARAMETRISED IN λ_i WHICH ARE THE EIGENVALUES TO A. IN THE CASE OF RUNGE-KUTTA, ONLY REAL EIGENVALUES ARE CONSIDERED.

Approach	p	m	Stability region ($h_{\max}$)
Euler sampling	1	1	$-\frac{2\mathfrak{Re}\{\lambda_i\}}{\lvert\lambda_i\rvert^2}$
Oversampled Euler	1	$m > 1$	$-\frac{2m\mathfrak{Re}\{\lambda_i\}}{\lvert\lambda_i\rvert^2}$
Runge-Kutta	4	1	$-\frac{2.7852}{\lambda_i}$, $\lambda_i \in \mathbb{R}$
Oversampled Runge-Kutta	4	$m > 1$	$-\frac{2.7852m}{\lambda_i}$, $\lambda_i \in \mathbb{R}$

- Section V shows how the computational complexity and the numerical properties differs between the different methods.
- Section VI presents a second order spring-damper example to demonstrate the advantages using a continuous-time update.
- Section VII discusses implications for nonlinear systems, and investigates a nonlinear system inspired by applications in robotics.

III. STABILITY ANALYSIS

It is known that the CDLE in (5b) has a unique positive solution $P(t)$ if A is Hurwitz[1], $GQG^{\mathsf{T}} \succeq 0$, the pair $(A, \sqrt{GQG^{\mathsf{T}}})$ is controllable, and $P(0) \succ 0$ [19]. We want to show that a stable continuous-time system results in a stable discrete-time recursion. We therefore assume that the continuous-time ODE describing the state vector $x(t)$ is stable, hence the eigenvalues λ_i, $i = 1, \ldots, n$ to A are assumed to be in the left half plane, i.e., $\mathfrak{Re}\{\lambda_i\} < 0$, $i = 1, \ldots, n_x$. It will also be assumed that the remaining requirements are fulfilled.

For the methods described in Section II-B we have that H in (6) has the eigenvalues $\pm\lambda_i$, $i = 1, \ldots, n_x$, where λ_i are the eigenvalues of A. This follows from the structure of H. Hence, the matrix exponential e^{Ht} will have terms that tend to infinity and zero with the same exponential rate when t increases. However, the case $t = h$ is of most interest, where h is finite. Note that a small/large sample time depends strongly on the system dynamics. Even if the matrix e^{Ht} is ill-conditioned, the product (8) and the ratio (9) can be limited under the assumptions above, for not too large values of t. Note that the solution in (9) is, as a matter of fact, based on the solution of an unstable ODE, see [14], [15], but the ratio $C(t)M_3(t)^{-1}$ can still be bounded. Both of these methods can have numerical problems which will be discussed in Section V-B.

A. Stability for the Vectorisation Method using Euler Sampling

The stability analysis in this section is standard and a similar analysis has been performed in [22]. The difference is that the analysis in [22] investigates which discretization methods that are stable for sufficiently small sample times. The analysis here is about to find an upper bound of the sample time such that a stable continuous-time model remains stable after discretization.

The recursive solution (15) is stable for all h according to Lemma 9 in Appendix B, if the matrix exponential can be calculated exactly. Stability issues arise when $e^{A_z h}$ has to be approximated by $e_{p,m}(A_z h)$. In this section we derive an upper bound on h that gives a stable solution for $e_{1,m}(A_z h)$, i.e., Euler sampling. The Taylor expansion and in particular Euler sampling is chosen due to its simplicity, the same approach is applicable to the Padé approximation as well. Higher orders of approximations using the Taylor expansion will be treated briefly at the end of this section.

From Section II-C we have that the matrix A_z is diagonal, which means that calculation of the matrix exponential $e^{A_z h}$ can

[1] All eigenvalues are in the left half plane.

be separated into e^{Ah} and $e^{A_P h}$. From [23] it is known that the eigenvalues of A_P are given by $\lambda_i + \lambda_j$, $1 \leq i \leq j \leq n_x$, hence the ODE describing the CDLE is stable if A is Hurwitz. In order to keep the discrete-time system stable, the eigenvalues of both $e_{1,m}(Ah)$ and $e_{1,m}(A_P h)$ need to be inside the unit circle. In Theorem 1 an explicit upper bound on the sample time h is given that makes the recursive solution to the continuous-time SDE stable.

Theorem 1: The recursive solution to the SDE (1), in the form of (15), where the matrix exponential $e^{A_z h}$ is approximated by $e_{1,m}(A_z h)$, is stable if

$$h < \min \left\{ -\frac{2m\mathfrak{Re}\left\{\lambda_i + \lambda_j\right\}}{\left|\lambda_i + \lambda_j\right|^2},\, 1 \leq i \leq j \leq n_x \right\}, \tag{20}$$

where λ_i, $i = 1, \ldots, n$, are the eigenvalues to A.

Corollary 2: The bound in Theorem 1 becomes

$$h < -\frac{2m}{\lambda_i}, \quad \lambda_i \in \mathbb{R}, \tag{21}$$

for real eigenvalues.

Proof: Start with the ODE describing the state vector $x(t)$. The eigenvalues to $e_{1,m}(Ah) = (I + Ah/m)^m$ are, according to Lemma 8 in Appendix B, given by $(1 + \lambda_i h/m)^m$. The eigenvalues are inside the unit circle if $|(1 + \lambda_i h/m)^m| < 1$, where

$$\left| \left(1 + \frac{\lambda_i h}{m}\right)^m \right| = \left(\frac{1}{m} \sqrt{m^2 + 2a_i hm + (a_i^2 + b_i^2)h^2} \right)^m. \tag{22}$$

In (22), the parametrization $\lambda_i = a_i + ib_i$ has been used. Solving $|(1 + \lambda_i h/m)^m| < 1$ for h and using the fact $|\lambda_i|^2 = a_i^2 + b_i^2$ give

$$h < -\frac{2ma_i}{|\lambda_i|^2}. \tag{23}$$

Similar calculations for the ODE describing vech $P(t)$ give

$$h < -\frac{2m(a_i + a_j)}{|\lambda_i + \lambda_j|^2}, \quad 1 \leq i \leq j \leq n_x. \tag{24}$$

Using $\lambda_i = \lambda_j$ in (24) gives

$$-\frac{2m(a_i + a_j)}{|\lambda_i + \lambda_j|^2} = -\frac{4ma_i}{|2\lambda_i|^2} = -\frac{ma_i}{|\lambda_i|^2}, \tag{25}$$

which is half as much as the bound in (23), hence the upper bound for h is given by (24). ∎

Theorem 1 shows that the sample time can be decreased if the absolute value of the eigenvalues are increased, but also if the real part approaches zero. The level curves of (20) for $h = c =$ constant in the complex plane are given by

$$-\frac{2a_{ij}m}{a_{ij}^2 + b_{ij}^2} = c \Leftrightarrow \left(a_{ij} + \frac{m}{c}\right)^2 + b_{ij}^2 = \frac{m^2}{c^2} \tag{26}$$

where $a_{ij} = \mathfrak{Re}\left\{\lambda_i + \lambda_j\right\}$ and $b_{ij} = \mathfrak{Im}\left\{\lambda_i + \lambda_j\right\}$. Equation (26) is the description of a circle with radius m/c centered in the point $(-m/c, 0)$. The level curves are shown in Figure 1, where it can be seen how the maximal sample time depends on the magnitude and direction of the eigenvalues.

Stability conditions of $e_{p,m}(A_z h)$ for $p > 1$ can be carried out in the same way as for $p = 1$. For $p = 2$, the calculations can be done analytically and this results again in (20),

$$h < \min \left\{ -\frac{2m\mathfrak{Re}\left\{\lambda_i + \lambda_j\right\}}{\left|\lambda_i + \lambda_j\right|^2},\, 1 \leq i \leq j \leq n_x \right\}. \tag{27}$$

It means that though the accuracy has increased, recall (17c), the stability condition remains the same.

Increasing the order of approximation even more, results in a higher order polynomial inequality that has to be solved. A numerical solution is therefore preferred. The stability bound for h/m will actually increase when $p > 2$ increases. For example, $e_{4,m}(Ah)$, which corresponds to the Runge-Kutta solution for a linear ODE gives, for real eigenvalues,

$$h < -\frac{2.7852m}{\lambda_i}, \quad \lambda_i \in \mathbb{R}. \tag{28}$$

This is less conservative than the corresponding bound in (21).

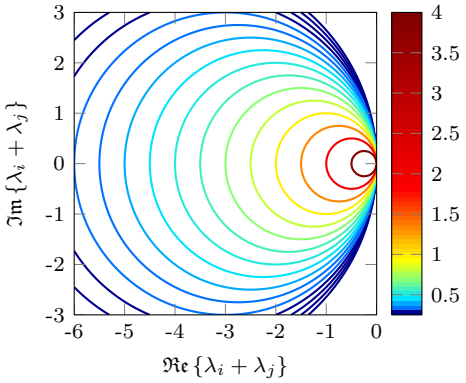

Fig. 1. Level curves of (20), where the colors indicate the values on h.

IV. Reformulation of the Vectorised Solution for the CDLE

The solution to the ODE describing the second order moment given by (12) can be computed efficiently using the following lemma.

Lemma 3: The solution to the vectorized CDLE

$$\text{vech}\, \dot{P}(t) = A_P \text{vech}\, P(t) + \text{vech}\, GQG^{\mathsf{T}}, \tag{29}$$

is given by

$$\text{vech}\, P(t) = F_P(t)\text{vech}\, P(0) + G_P(t)\text{vech}\, GQG^{\mathsf{T}}. \tag{30}$$

where $F_P(t)$ and $G_P(t)$ are given by

$$\exp\left[\begin{pmatrix} A_P & I \\ 0 & 0 \end{pmatrix} t\right] = \begin{pmatrix} F_P(t) & G_P(t) \\ 0 & I \end{pmatrix} \tag{31}$$

Proof: Let $\zeta = \text{vech}\, GQG^{\mathsf{T}}$, then it follows that $\dot{\zeta} = 0$ and $\zeta(0) = \text{vech}\, GQG^{\mathsf{T}}$. The vectorised CDLE (29) can now be written as

$$\frac{d}{dt}\begin{pmatrix} \text{vech}\, P(t) \\ \zeta(t) \end{pmatrix} = \begin{pmatrix} A_P & I \\ 0 & 0 \end{pmatrix}\begin{pmatrix} \text{vech}\, P(t) \\ \zeta(t) \end{pmatrix}. \tag{32}$$

The solution to (32) is

$$\begin{pmatrix} \text{vech}\, P(t) \\ \zeta(t) \end{pmatrix} = \exp\left[\begin{pmatrix} A_P & I \\ 0 & 0 \end{pmatrix} t\right]\begin{pmatrix} \text{vech}\, P(0) \\ \zeta(0) \end{pmatrix}, \tag{33}$$

and the result follows immediately. ∎

The new solution based on Lemma 3 is presented in Theorem 5. The solution requires the matrix exponential and the solution of an algebraic Lyapunov equation for which efficient numerical solvers exist.

Remark 4: In contrast to Lemma 3, the solution to the CDLE in (5b) presented in Theorem 5 actually requires A_P to be non-singular. The eigenvalues to A_P are given by $\lambda_i + \lambda_j$, $1 \leq i \leq j \leq n_x$ [23], so we have that A_P is non-singular when the eigenvalues of A are not mirrored in the imaginary axis. Eigenvalues in the origin is a special case of this.

Theorem 5: Let Q be positive definite and assume that the eigenvalues of A are not mirrored in the imaginary axis. Then the solution of the CDLE (5b) is given by

$$P(t) = e^{At}P(0)e^{A^\mathsf{T}t} + Q_d(t), \tag{34a}$$

$$AQ_d(t) + Q_d(t)A^\mathsf{T} + GQG^\mathsf{T} - e^{At}GQG^\mathsf{T}e^{A^\mathsf{T}t} = 0, \tag{34b}$$

where $Q_d(t)$ is a unique and positive definite solution to (3).

Proof: Taylor expansion of the matrix exponential gives

$$e^{A_P t} = I + A_P t + \frac{A_P^2 t^2}{2!} + \frac{A_P^3 t^3}{3!} + \dots \tag{35}$$

Using (72) in Appendix C, each term in the Taylor expansion can be rewritten according to

$$A_P^k t^k = D^\dagger (I \otimes A + A \otimes I)^k t^k D, \tag{36}$$

hence

$$e^{A_P t} = D^\dagger e^{(I\otimes A + A\otimes I)t} D \overset{(70),(71)}{=} D^\dagger (e^{At} \otimes e^{At}) D. \tag{37}$$

The first term in (30) can now be written as

$$\begin{aligned} e^{A_P t}\operatorname{vech} P(0) &= D^\dagger (e^{At} \otimes e^{At}) D \operatorname{vech} P(0) \\ &= D^\dagger (e^{At} \otimes e^{At}) \operatorname{vec} P(0) \overset{(69)}{=} D^\dagger \operatorname{vec} e^{At} P(0) e^{A^\mathsf{T}t} \\ &= \operatorname{vech} e^{At} P(0) e^{A^\mathsf{T}t}. \end{aligned} \tag{38}$$

Similar calculations give

$$e^{A_P t}\operatorname{vech} GQG^\mathsf{T} = D^\dagger \operatorname{vec} e^{At} GQG^\mathsf{T} e^{A^\mathsf{T}t} \triangleq \operatorname{vech} \widehat{Q}(t). \tag{39}$$

It is easily verified using the Taylor expansion of (31) that $G_P(t) = A_P^{-1}(e^{A_P t} - I)$. The last term in (30) can therefore be rewritten according to

$$\begin{aligned} A_P^{-1}(e^{A_P t} - I)\operatorname{vech} GQG^\mathsf{T} &= A_P^{-1} \operatorname{vech} (\widehat{Q}(t) - GQG^\mathsf{T}) \\ &\triangleq \operatorname{vech} Q_d(t), \end{aligned} \tag{40}$$

where it is assumed that A_P is invertible. Equation (40) can be seen as the solution of the linear system of equations $A_P \operatorname{vech} Q_d(t) = \operatorname{vech} (\widehat{Q}(t) - GQG^\mathsf{T})$. Using the derivation in (64) in Appendix A backwards gives that $Q_d(t)$ is the solution to the algebraic Lyapunov equation

$$AQ_d(t) + Q_d(t)A^\mathsf{T} + GQG^\mathsf{T} - \widehat{Q}(t) = 0. \tag{41}$$

Combining (38) and (40) gives that (30) can be written as

$$P(t) = e^{At}P(0)e^{A^\mathsf{T}t} + Q_d(t), \tag{42}$$

where $Q_d(t)$ is the solution to (41).

It is easily verified that $Q_d(t)$ in (3) satisfies (41) and it is well known that the Lyapunov equation has a unique solution iff the eigenvalues of A are not mirrored in the imaginary axis [19]. Moreover, the assumption that Q is positive definite gives from (3) that $Q_d(t)$ is positive definite, hence the solution to (41) is unique and guaranteed to be positive definite under the assumptions on A. ∎

If Lemma 3 is used directly, a matrix exponential of a matrix of dimension $n_x(n_x+1) \times n_x(n_x+1)$ is required. Now, only the Lyapunov equation (34b) has to be solved, where the dimensions of the matrices are $n_x \times n_x$. The computational complexity for solving the Lyapunov equation is $35n_x^3$ [17]. The total computational complexity for computing the solution of (5b) using Theorem 5 is $(\log_2(m) + p + 43)\, n_x^3$, where $(\log_2(m) + p)\, n_x^3$ comes from the matrix exponential, and $43n_x^3$ comes from solving the Lyapunov equation ($35n_x^3$) and taking four matrix products giving $2n_x^3$ each time.

TABLE II. SIX VARIANTS TO CALCULATE $P(t)$. THE LAST COLUMN SHOWS THE COMPUTATIONAL COMPLEXITY p IN $\mathcal{O}(n_x^p)$ WHICH IS DESCRIBED IN DETAIL IN SECTION V-A.

METHOD	Description	p in $\mathcal{O}(n_x^p)$
I	$P(t)$ from Lemma 3.	6
II	$P(t)$ from Theorem 5.	3
III	$P(t)$ from (4b) with Q_d calculated using equation (8).	3
IV	$P(t)$ from (4b) with Q_d calculated using an eigenvalue decomposition for diagonalising the integrand in (3).	3
V	$P(t)$ from (4b) with Q_d calculated using numerical integration of the integral in (3) using `quadv` in MATLAB.	3
VI	$P(t)$ from the matrix fraction decomposition in (9).	3

The algebraic Lyapunov function (34b) has a unique solution only if the eigenvalues of A are not mirrored in the imaginary axis [19], as a result of the assumption that A_P is non-singular, and this is the main drawback with using Theorem 5 rather than using Lemma 3. In the case of integrators, the method presented in [24] can be used. To be able to calculate $Q_d(t)$, the method transforms the system such that the Lyapunov equation (34b) is used for the subspace without the integrators, and the integral in (3) is used for the subspace containing the integrators.

Discrete-time Recursion: The recursive solution to the differential equations in (5) describing the first and second order moments of the SDE (1) can now be written as

$$\hat{x}((k+1)h) = e^{Ah}\hat{x}(kh), \tag{43a}$$

$$P((k+1)h) = e^{Ah}P(kh)e^{A^\mathsf{T}h} + Q_d(h), \tag{43b}$$

$$AQ_d(h) + Q_d(h)A^\mathsf{T} + GQG^\mathsf{T} - e^{Ah}GQG^\mathsf{T}e^{A^\mathsf{T}h} = 0, \tag{43c}$$

Equations (43b) to (43c) are derived using $t = kh$ and $t = (k+1)h$ in (34).

The method presented in Theorem 5 is derived straightforwardly from Lemma 3. A similar solution that also solves an algebraic Lyapunov function is presented in [18]. The main difference is that Theorem 5 gives a value of the covariance matrix $Q_d(t)$ for the discrete-time noise explicitly, as opposed to the solution in [18]. Moreover, the algebraic Lyapunov function in [18] is independent of time, which is not the case here since $e^{At}GQG^\mathsf{T}e^{A^\mathsf{T}t}$ changes with time. This is not an issue for the recursive time update due to the fact that $e^{Ah}GQG^\mathsf{T}e^{A^\mathsf{T}h}$ is only dependent on h, hence the algebraic Lyapunov equation (43c) has to be solved only once.

V. COMPARISON OF SOLUTIONS FOR THE CDLE

This section will provide rough estimates of the computational complexity of the different approaches to compute the CDLE, by counting the number of flops. Numerical properties are also discussed. Table II summarizes six variants of the methods presented in Section II of how to calculate $P(t)$.

A. Computational Complexity

Rough estimates of the computational complexity can be given by counting the number of operations that are required. From Section IV it is given that the computational complexity for METHOD I is $\mathcal{O}(n_x^6)$ and for METHOD II it is $(\log_2(m) + p + 43)\, n_x^3$. The total computational complexity for METHOD III is roughly $(8(\log_2(m)+p)+6)n_x^3$, where $(\log_2(m) + p)(2n_x)^3$ comes from e^{Ht} and $6n_x^3$ from the remaining three matrix products. Using an eigenvalue decomposition to calculate the integral, i.e., METHOD IV, gives a computational complexity of $\mathcal{O}(n_x^3)$. For numerical integration, i.e., METHOD V,

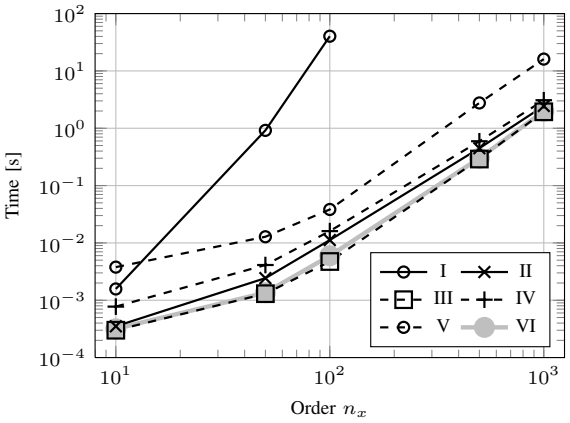

Fig. 2. Mean execution time for calculating $P(t)$ for randomly generated state matrices with order $n_x \times n_x$ over 1000 MC simulations.

TABLE III. STANDARD DEVIATION OF THE EXECUTION TIME FOR CALCULATING $P(t)$.

	10	50	100	500	1000
I	$9.63{\cdot}10^{-5}$	$7.85{\cdot}10^{-2}$	1.38	–	–
II	$1.88{\cdot}10^{-5}$	$1.46{\cdot}10^{-4}$	$8.20{\cdot}10^{-4}$	$4.37{\cdot}10^{-2}$	$1.03{\cdot}10^{-1}$
III	$6.09{\cdot}10^{-6}$	$8.84{\cdot}10^{-5}$	$3.71{\cdot}10^{-4}$	$3.89{\cdot}10^{-2}$	$2.41{\cdot}10^{-1}$
IV	$7.67{\cdot}10^{-5}$	$1.72{\cdot}10^{-4}$	$7.42{\cdot}10^{-4}$	$5.14{\cdot}10^{-2}$	$2.15{\cdot}10^{-1}$
V	$1.26{\cdot}10^{-4}$	$4.96{\cdot}10^{-4}$	$1.68{\cdot}10^{-3}$	$2.12{\cdot}10^{-1}$	1.39
VI	$1.95{\cdot}10^{-5}$	$5.35{\cdot}10^{-5}$	$3.89{\cdot}10^{-4}$	$3.69{\cdot}10^{-2}$	$2.06{\cdot}10^{-1}$

the computational complexity will be $\mathcal{O}(n_x^3)$ due to the matrix exponential and the matrix products. The constant in front of n_x^3 will be larger than for METHOD III and METHOD IV. That is because of element-wise integration of the $n_x \times n_x$ symmetric matrix integrand, which requires $n_x(n_x+1)/2$ number of integrations. For METHOD VI, the same matrix exponential as in METHOD III is calculated which gives $(\log_2(m)+p)8n_x^3$ operations. In addition, $2n_x^3$ operations for the matrix inverse and $4n_x^3$ operations for the two remaining matrix products are required. In total, the computational complexity is $(8(\log_2(m)+p)+6)n_x^3$. The product $C(t)M_3(t)^{-1}$ can of course be calculated without first computing the inverse and then performing the multiplication, but it is a rough estimate presented here.

The computational complexity is also analysed by performing Monte Carlo simulations over 1000 randomly chosen stable systems. The order of the systems are $n_x = 10, 50, 100, 500, 1000$. As expected, the solution using METHOD I takes very long time as can be seen in Figure 2. For METHOD I the size has been restricted to $n_x \leq 100$ since A_P grows too much for larger values. However, the computational time using METHOD I compared to the other methods is clear. The computational time for the other methods is in the same order, which is also expected. As discussed previously, the numerical integration will give a computational complexity that has the same slope but with a higher offset than METHOD II–IV, and METHOD VI, which is seen in Figure 2. It can also be noted that the numerical integration for $n_x = 10$ is slower than for METHOD I.

Note that not only the number of operations of performing e.g. the matrix exponential and the matrix multiplications affects the total time. Also, the time for memory management is included. However, the slope of the lines for large n_x is approximately six for METHOD I and three for the other methods, which agrees with the computational complexity discussed above. The standard deviation for the computation time for the different methods is at least one order of magnitude less than the mean value, see Table III.

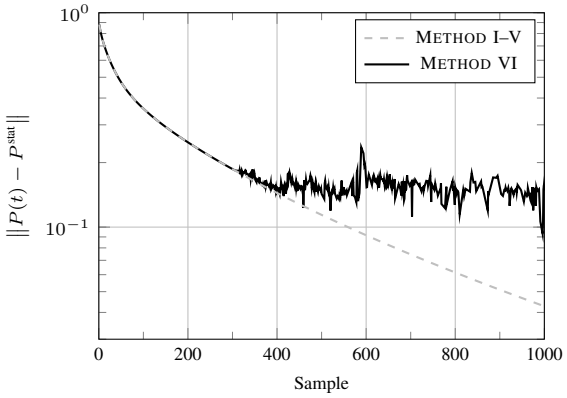

Fig. 3. The mean norm over 100 MC simulations of the error $P(t) - P^{\text{stat}}$ for the recursive updates. METHOD I–V gives the same behaviour whereas METHOD VI diverges

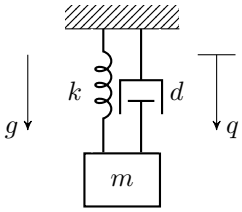

Fig. 4. A mass hanging in a spring and damper.

B. Numerical Properties

Here, the numerical properties will be analysed. First, the solution $P(t)$ should hold for any value of t. It means that a large enough value of t should give that $P(t)$ equals the stationary solution given from the stationary Lyapunov equation

$$AP^{\text{stat}} + P^{\text{stat}}A^{\mathsf{T}} + GQG^{\mathsf{T}} = 0 \tag{44}$$

Second, the recursive updates should approach P^{stat} and then stay there when $k \to \infty$.

Randomly generated stable system matrices, over 100 MC simulations[2], of order $n_x = 2$ will be used with $GQG^{\mathsf{T}} = I$ to show how the methods perform. For the first case the value $t = 100$ has been used and for the second case the sample time $h = 0.01\,\text{s}$ has been used and a total of 10,000 samples.

The stationary matrix P^{stat} is not obtained for METHOD III–IV, and METHOD VI for all MC simulations. However, methods METHOD I–II and METHOD V gives P^{stat} as the solution. The reason that METHOD III and METHOD VI cannot give the correct stationary matrix is that they have to calculate the ill-conditioned matrix e^{Ht}.

For the second case, where the recursive update is used, the difference $\|P(t) - P^{\text{stat}}\|$ for the methods are shown in Figure 3 for the first 1000 samples. It can be seen that METHOD I–V converge to the stationary solution. METHOD VI is not able to converge to the stationary solution when the time goes by, instead numerical problems occur, giving Inf or NaN (Not-a-Number) as solution.

VI. LINEAR SPRING-DAMPER EXAMPLE

The different solutions and approximations described above will be investigated for a linear model of a mass m hanging in a spring and damper, see Figure 4. The equation of motion is

[2]Here, only 100 MC simulations are used to be able to visualise the result, otherwise too many samples with Inf or NaN as solution, which cannot be displayed in the figure.

$$m\ddot{q} + d\dot{q} + kq - mg = 0 \tag{45}$$

where q is the distance from where the spring/damper is unstretched and $g = 9.81$ is the gravity constant. A linear state space model, using $m = 1$, with $x = \begin{pmatrix} q & \dot{q} \end{pmatrix}^{\mathsf{T}}$ is given by

$$\dot{x}(t) = \underbrace{\begin{pmatrix} 0 & 1 \\ -k & -d \end{pmatrix}}_{A} x(t) + \underbrace{\begin{pmatrix} 0 \\ g \end{pmatrix}}_{B}. \tag{46}$$

A. Stability Bound on the Sample Time

The bound on the sample time that makes the solution to (46) stable when $e_{1,m}(Ah)$ is used, can be calculated using Theorem 1. The eigenvalues for A are

$$\lambda_{1,2} = -\frac{d}{2} \pm \frac{1}{2}\sqrt{d^2 - 4k}. \tag{47}$$

If $d^2 - 4k \geq 0$ the system is well damped and the eigenvalues are real, hence

$$\begin{aligned} h &< \min\left\{ \frac{2m}{d + \sqrt{d^2 - 4k}}, \frac{2m}{d - \sqrt{d^2 - 4k}}, \frac{2m}{d} \right\} \\ &= \frac{2m}{d + \sqrt{d^2 - 4k}}, \end{aligned} \tag{48}$$

If instead $d^2 - 4k < 0$, the system is oscillating and the eigenvalues are complex, giving

$$h < \min\left\{ \frac{dm}{2k}, \frac{2m}{d}, \frac{dm}{2k} \right\} = \frac{dm}{2k}, \tag{49}$$

where we have used the fact that $d^2 - 4k < 0$ to get the minimum value.

The values on the parameters have been chosen as $d = 2$ and $k = 10$ giving an oscillating system. The stability bound is therefore $h < 0.1m$ s.

B. Kalman Filtering

We will now focus on Kalman filtering of the spring-damper example.

The continuous-time model (46) is augmented with process noise giving the model

$$dx(t) = Ax(t)dt + B + Gd\beta(t), \tag{50}$$

where A and B are given by (46), $G = \begin{pmatrix} 0 & 1 \end{pmatrix}^{\mathsf{T}}$ and $d\beta(t)$ is a scalar Wiener process with $E\left[d\beta(t)d\beta(t)^{\mathsf{T}}\right] = Qdt$. Here it is used that $Q = 5 \cdot 10^{-3}$. It is assumed that the velocity $\dot{q}$ is measured with a sample rate T_s. The measurement equation can be written as

$$y_k = \begin{pmatrix} 0 & 1 \end{pmatrix} x_k + e_k = Cx_k + e_k, \tag{51}$$

where $e_k \in \mathbb{R}$ is discrete-time normal distributed white noise with zero mean and a standard deviation of $\sigma = 0.05$. Here, $y_k \triangleq y(kT_s)$ has been used for notational convenience. It is easy to show that the system is observable with this measurement. The stability condition for the first order approximation $e_{1,m}(Ah)$ was calculated to be $h < 0.1m$ seconds in Section VI-A. We chose therefore $T_s = h = 0.09$ s.

The simulation represents free motion of the mass when starting at $x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^{\mathsf{T}}$. The ground truth data is obtained by simulating the continuous-time SDE over $t_{\text{max}} = 20$ s with a sample time h_S that is 100 times shorter than T_s. In that case, the Wiener process $d\beta(t)$ can at each sample instance be approximated by a normal distributed zero mean white noise process with covariance matrix Qh_S.

Four Kalman filters are compared where e^{Ah} is approximated either by $e_{1,m}(Ah)$ or by the MATLAB-function `expm`. The function `expm` uses scaling and squaring techniques with a Padé approximation to compute the matrix exponential, see [21], [25]. Moreover, the update of the covariance matrix $P(t)$ is according to the discrete filter (19b) or according to one of the solutions presented in Sections II-A to II-C. Here, the solution to the CDLE given by Theorem 5 has been used, but the other methods would give the same results. Remember though that the matrix fraction method can have numerical problems. In summary, the Kalman filters are:

1) $F_h = e_{1,m}(Ah)$ and $P(k+1) = F_h P(k) F_h^{\mathsf{T}} + G_h Q_h G_h^{\mathsf{T}}$,
2) F_h is given by the MATLAB-function `expm` and $P(k+1) = F_h P(k) F_h^{\mathsf{T}} + G_h Q_h G_h^{\mathsf{T}}$,
3) $F_h = e_{1,m}(Ah)$ and $P(k+1) = F_h P(k) F_h^{\mathsf{T}} + Q_d(h)$,
4) F_h is given by the MATLAB-function `expm` and $P(k+1) = F_h P(k) F_h^{\mathsf{T}} + Q_d(h)$,

where $Q_d(h)$ is the solution to the Lyapunov equation in (43c).

The Kalman filters are initialised with the true x_0, used for ground truth data, plus a normal distributed random term with zero mean and standard deviation 0.1. The state covariance is initialized by $P(0) = I$. The covariance matrix for the measurement noise is the true one, i.e., $R = \sigma^2$. The covariance matrix for the process noise are different for the filters. For filter 1 and 2 the covariance matrix Qh/m is used whereas for filter 3 and 4 the true covariance matrix Q is used.

The filters are compared to each other using $N_{MC} = 1000$ Monte Carlo simulations for different values of m. The oversampling constant m takes the following values:

$$\{1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ 20,\ 30,\ 40,\ 50\} \tag{52}$$

Figure 5 shows the root mean square error (RMSE) defined according to

$$\rho_i = \sqrt{\frac{1}{N}\sum_{t=t_0}^{t_{\text{max}}} \left(\rho_i^{MC}(t)\right)^2}, \tag{53a}$$

where $t_0 = t_{\text{max}}/2$ in order to remove the transients, N is the number of samples in $[t_0, t_{\text{max}}]$, and

$$\rho_i^{MC}(t) = \sqrt{\frac{1}{N_{MC}}\sum_{j=1}^{N_{MC}} \left(x_i^j(t) - \hat{x}_i^j(t)\right)^2}, \tag{53b}$$

where $x_i^j(t)$ is the true ith state and $\hat{x}_i^j(t)$ is the estimated ith state for Monte Carlo simulation number j. The two filters 1 and 3 give almost identical results for the RMSE, therefore only filter 1 is shown in Figure 5, see the solid line. The dashed lines are the RMSE for filter 4 (filter 2 gives the same result). We can see that a factor of $m = 20$ or higher is required to get the same result for Euler sampling as for the continuous-time solution[3]. The execution time is similar for all four filters and increases with the same amount when m increases, hence a large enough oversampling can be difficult to achieve for systems with hard real-time requirements. In that case, the continuous-time solution is to prefer.

Remark 6: The maximum sample time, derived according to Theorem 1, is restricted by the CDLE as is described in the proof. It means that we can use a larger sample time for the ODE describing the states, in this particular case a twice as large sample time. Based on this, we already have oversampling by a factor of at least two, for the ODE describing the states, when the sample time is chosen according to Theorem 1.

In Figure 6 we can see how the norm of the stationary covariance matrix[4] for the estimation error changes when oversampling is used. The four filters converge to the same value when m increases. For the discrete-time update in (19b), i.e., filter 1 and 2, the stationary value

[3] It is wise to choose m to be a power of 2, as explained in Section II-E

[4] The covariance matrix at time t_{max} is used as the stationary covariance matrix, i.e., $P(t_{\text{max}})$.

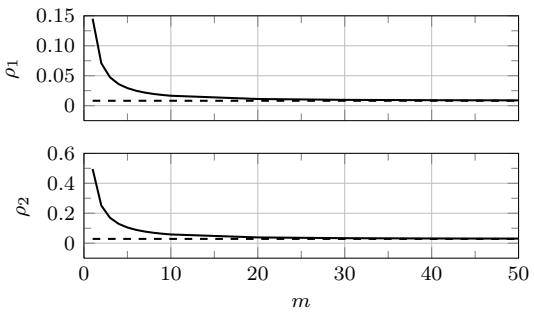

Fig. 5. RMSE according to (53) as a function of the degree of oversampling, where the solid line is filter 1 (filter 3 gives the same) and the dashed line is filter 4 (filter 2 gives the same).

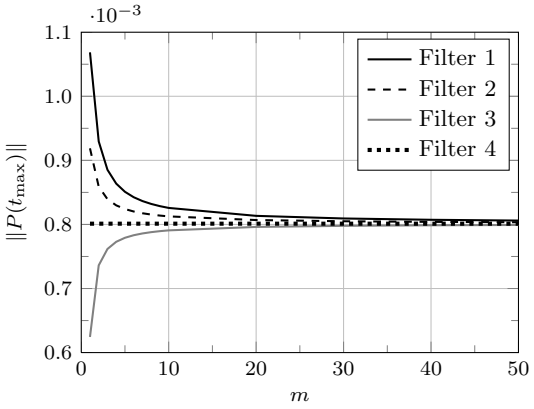

Fig. 6. The norm of the stationary covariance matrix for the estimation error for the four filters, as a function of the degree of oversampling.

is too large for small values of m. For the continuous-time update in Theorem 5, it can be seen that a first order Taylor approximation of the exponential function, i.e., filter 3, gives a too small covariance matrix which increases when m increases.

A too small or too large covariance matrix for the estimation error can be crucial for different applications, such as target tracking, where the covariance matrix is used for data association.

VII. Extensions to Nonlinear Systems

We will in this section adapt the results for linear systems to nonlinear systems. Inevitably, some approximations have to be done, and the most fundamental one is to assume that the state is constant during the small time steps h/m. This approximation becomes better the larger oversampling factor m is chosen.

A. EKF Time Update

Let the dynamics be given by the nonlinear SDE

$$dx(t) = f(x(t))dt + G(x(t))d\beta(t), \tag{54}$$

for $t \geq 0$, where $x(t) \in \mathbb{R}^{n_x}$, $f(x(t)) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$, $G(x(t)) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x \times n_\beta}$, and $d\beta(t) \in \mathbb{R}^{n_\beta}$ is a vector of Wiener processes with $E\left[d\beta(t)d\beta(t)^\mathsf{T}\right] = Qdt$. For simplicity, it is assumed that $G(x(t)) \triangleq G$. The propagation of the first and second order moments for the extended Kalman filter (EKF) can, as in the linear case, be written as [2]

$$\dot{\hat{x}}(t) = f(\hat{x}(t)), \tag{55a}$$
$$\dot{P}(t) = F(\hat{x}(t))P(t) + P(t)F(\hat{x}(t))^\mathsf{T} + GQG^\mathsf{T}, \tag{55b}$$

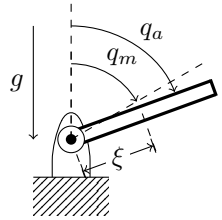

Fig. 7. A single flexible joint.

TABLE IV. Model parameters for the nonlinear model.

J_a	J_m	m	ξ	d	k_1	k_2	f_d	g
1	1	1	1	1	10	100	1	9.81

where $F(\hat{x}(t))$ is the Jacobian of $f(x(t))$ evaluated at $\hat{x}(t)$. The main differences to (5) are that a linear approximation of $f(x)$ is used in the CDLE as well as the CDLE is dependent on the state vector x. Without any assumptions, the two equations in (55) have to be solved simultaneously. The easiest way is to vectorize (55b) similar to what is described in Appendix A and then solve the nonlinear ODE

$$\frac{d}{dt}\begin{pmatrix}\hat{x}(t)\\ \operatorname{vech} P(t)\end{pmatrix} = \begin{pmatrix}f(\hat{x}(t)),\\ A_P(\hat{x}(t))\operatorname{vech} P + \operatorname{vech} GQG^\mathsf{T}\end{pmatrix}, \tag{56}$$

where $A_P(\hat{x}(t)) = D^\dagger(I \otimes F(\hat{x}(t)) + F(\hat{x}(t)) \otimes I)D$. The nonlinear ODE can be solved using a numerical solver such as Runge-Kutta methods [1]. If it is assumed that $\hat{x}(t)$ is constant over an interval of length h/m, then the two ODEs describing $\hat{x}(t)$ and $\operatorname{vech} P(t)$ can be solved separately. The ODE for $\hat{x}(t)$ is solved using a numerical solver and the ODE for $\operatorname{vech} P(t)$ becomes a linear ODE which can be solved using Theorem 5, where $A \triangleq F(\hat{x}(t))$.

Remark 7: When m increases, the length of the interval, where $\hat{x}(t)$ has to be constant, decreases. In that case, the assumption of constant $\hat{x}(t)$ is more valid, hence the two ODEs can be solved separately without introducing too much errors.

Similar extensions for the method using matrix fraction is straightforward to derive. The advantage with the vectorised solution is that it is easy to solve the combined ODE for $x(t)$ and $\operatorname{vech} P(t)$ using a Runge-Kutta solver. This can be compared to the method using matrix fraction, which becomes a coupled differential equation with both vector and matrix variables.

B. Simulations of a Flexible Joint

A nonlinear model for a single flexible joint is investigated in this section, see Figure 7. The equations of motion are given by

$$J_a\ddot{q}_a + G(q_a) + D(\dot{q}_a, \dot{q}_m) + T(q_a, q_m) = 0, \tag{57a}$$
$$J_m\ddot{q}_m + F(\dot{q}_m) - D(\dot{q}_a, \dot{q}_m) - T(q_a, q_m) = u, \tag{57b}$$

where the gravity, damping, spring, and friction torques are modeled as

$$G(q_a) = -gm\xi\sin(q_a), \tag{58a}$$
$$D(\dot{q}_a, \dot{q}_m) = d(\dot{q}_a - \dot{q}_m), \tag{58b}$$
$$T(q_a, q_m) = k_2(q_a - q_m) + k_1(q_a - q_m)^3, \tag{58c}$$
$$F(\dot{q}_m) = f_d\dot{q}_m, \tag{58d}$$

Numerical values of the parameters, used for simulation, are given in Table IV. The parameters are chosen to get a good system without unnecessary large oscillations. With the state vector $x = \begin{pmatrix}q_a & q_m & \dot{q}_a & \dot{q}_m\end{pmatrix}^\mathsf{T}$ a nonlinear system of continuous-time ODEs

can be written as

$$\dot{x} = \underbrace{\begin{pmatrix} x_3 \\ x_4 \\ \frac{1}{J_a}\left(gm\xi\sin(x_1) - d\Delta_{34} - k_2\Delta_{12} - k_1\Delta_{12}^3\right) \\ \frac{1}{J_m}\left(d\Delta_{34} + k_2\Delta_{12} + k_1\Delta_{12}^3 - f_d x_4 + u\right) \end{pmatrix}}_{f(x,u)} \tag{59}$$

where $\Delta_{ij} = x_i - x_j$. The state space model (59) is also augmented with a noise model according to (54) with

$$G = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ J_a^{-1} & 0 \\ 0 & J_m^{-1} \end{pmatrix} \tag{60}$$

For the simulation, the arm is released from rest in the position $q_a = q_m = \pi/2$ and moves freely, i.e., $u(t) = 0$, to the stationary point $x = \begin{pmatrix}\pi & \pi & 0 & 0\end{pmatrix}^{\mathsf{T}}$. The ground truth data are obtained using a standard fourth order Runge-Kutta method with a sample time $h_S = 1 \cdot 10^{-6}$ s, which is much smaller than the sample time T_s for the measurements. In the same way as for the linear example in Section VI, the Wiener process $d\beta(t)$ can be approximated at each discrete-time instant by a zero mean white noise process with a covariance matrix Qh_S, where $Q = 1 \cdot 10^{-3} I_2$. It is assumed that the motor position q_m and velocity $\dot{q}_m$ are measured, with additive zero mean Gaussian measurement noise $e(kT_s) \in \mathbb{R}^2$ with a standard deviation $\sigma = 0.05 I_2$. The sample time for the measurements is chosen to be $T_s = h = 0.1$ s.

Two extended Kalman filters (EKF) are compared. The first filter uses the discrete-time update (19) where Euler sampling

$$x((k+1)h) = x(kh) + hf(x(kh), u(kh)) \triangleq F(x(kh)) \tag{61}$$

has been used for discretisation. The second filter solves the continuous-time ODE (56) using a standard fourth order Runge-Kutta method. The filters are initialised with the true $x(0)$ used for simulating ground truth data plus a random term with zero mean and standard deviation 0.1. The covariance matrix for the estimation error is initialised by $P(0) = 1 \cdot 10^{-4} I_4$. The results are evaluated over 1000 Monte Carlo simulations using the different values of m listed in (52).

Figure 8 shows the RMSE, defined in (53), for the four states. The discrete-time filter using Euler sampling requires an oversampling of approximately $m = 10$ in order to get the same performance as the continuous-time filter, which is not affected by m that much. In Figure 9, the norm of the stationary covariance matrix of the estimation error, i.e., $\|P(t_{\max})\|$, is shown. Increasing m, the value $\|P(t_{\max})\|$ decreases and approaches the corresponding value for the continuous-time filter. The result is in accordance with the linear model described in Section VI-B. The standard deviation for $\|P(t_{\max})\|$ is several orders of magnitude less than the mean value and decreases as m increases with a similar rate as the mean value in Figure 9.

The execution time for the two filters differs a lot. They both increase linearly with m and the continuous-time filter is approximately 4-5 times slower than the discrete-time filter. This is because of that the Runge-Kutta solver evaluates the function $f(x(t))$ four times for each time instant whereas the discrete-time filter evaluates the function $F(x(kh))$ only once. However, the time it takes for the discrete-time filter using $m = 10$ is approximately 1.6 times slower than using $m = 1$ for the continuous-time filter.

VIII. Conclusions

This paper investigates the continuous-discrete filtering problem for Kalman filters and extended Kalman filters. The critical time update

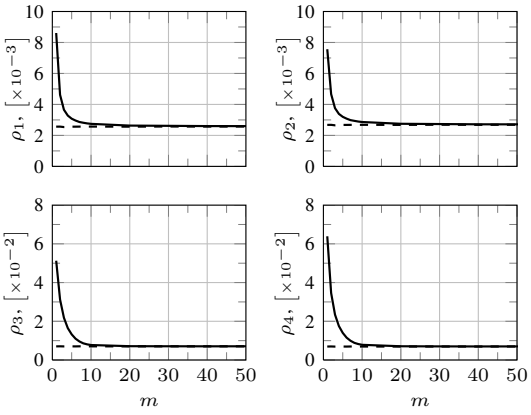

Fig. 8. RMSE according to (53), where the solid line is the discrete-time filter using Euler sampling and the dashed line is the continuous-time filter using a Runge-Kutta solver.

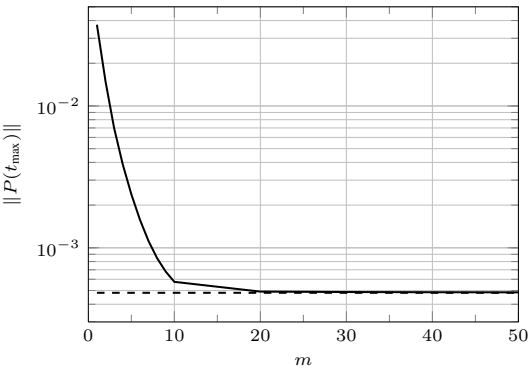

Fig. 9. The norm of the stationary covariance matrix for the estimation error for the EKF using Euler sampling (solid) and a fourth order Runge-Kutta method (dashed).

consists of solving one ODE and one continuous-time differential Lyapunov equation (CDLE). The problem can be rewritten as one ODE by vectorization of the CDLE. The main contributions of the paper are:

1) A survey of different ways to calculate the covariance of a linear SDE is presented. The different methods, presented in Table II, are compared to each other with respect to stability, computational complexity and numerical properties.
2) Stability condition for Euler sampling of the linear ODE which describes the first and second order moments of the SDE. An explicit upper bound on the sample time is derived such that a stable continuous-time system remains stable after discretization. The stability condition for higher order of approximations, such as the Runga-Kutta method, is also briefly investigated.
3) A numerical stable and time efficient solution to the CDLE that does not require any vectorization. The computational complexity for the straightforward solution, using vectorization, of the CDLE is $\mathcal{O}(n_x^6)$, whereas the proposed solution, and the methods proposed in the literature, have a complexity of only $\mathcal{O}(n_x^3)$.

The continuous-discrete filtering problem, using the proposed methods, is evaluated on a linear model describing a mass hanging in a spring-damper pair. It is shown that the standard use of the discrete-time Kalman filter requires a much higher sample rate in order to achieve the same performance as the proposed solution.

The continuous-discrete filtering problem is also extended to nonlinear systems and evaluated on a nonlinear model describing a single flexible joint of an industrial manipulator. The proposed solution requires the solution from a Runge-Kutta method and without any assumptions, vectorization has to be used for the CDLE. Simulations of the nonlinear joint model show also that a much higher sample time is required for the standard discrete-time Kalman filter to be comparable to the proposed solution.

APPENDIX A
VECTORIZATION OF THE CDLE

The matrix valued CDLE

$$\dot{P}(t) = AP(t) + P(t)A^{\mathsf{T}} + GQG^{\mathsf{T}}, \tag{62}$$

can be converted to a vector valued ODE using vectorization of the matrix $P(t)$. $P(t) \in \mathbb{R}^{n_x \times n_x}$ is symmetric so the half-vectorisation is used. The relationship between vectorisation, denoted by vec, and half-vectorisation, denoted by vech, is

$$\operatorname{vec} P(t) = D \operatorname{vech} P(t), \tag{63}$$

where D is a $n_x^2 \times n_x(n_x+1)/2$ duplication matrix. Let $n_P = n_x(n_x+1)/2$ and $\widetilde{Q} = GQG^{\mathsf{T}}$. Vectorisation of (62) gives

$$\begin{aligned}
\operatorname{vech} \dot{P}(t) &= \operatorname{vech}\left(AP(t) + P(t)A^{\mathsf{T}} + \widetilde{Q}\right) \\
&= \operatorname{vech} AP(t) + \operatorname{vech} P(t)A^{\mathsf{T}} + \operatorname{vech} \widetilde{Q} \\
&= D^{\dagger}(\operatorname{vec} AP(t) + \operatorname{vec} P(t)A^{\mathsf{T}}) + \operatorname{vech} \widetilde{Q} \\
&\overset{(68)}{=} D^{\dagger}[(I \otimes A) + (A \otimes I)]D \operatorname{vech} P(t) + \operatorname{vech} \widetilde{Q} \\
&= A_P \operatorname{vech} P(t) + \operatorname{vech} \widetilde{Q}
\end{aligned} \tag{64}$$

where $\otimes$ is the Kronecker product and $D^{\dagger} = (D^{\mathsf{T}}D)^{-1}D^{\mathsf{T}}$ is the pseudo inverse of D. Note that $D^{\dagger}D = I$ and $DD^{\dagger} \neq I$. The solution of the ODE (64) is given by [20]

$$\operatorname{vech} P(t) = e^{A_P t} \operatorname{vech} P(0) + \int_0^t e^{A_P(t-s)} \,\mathrm{d}s \operatorname{vech} \widetilde{Q}. \tag{65}$$

Assume that A_P is invertible, then the integral can be computed as

$$\begin{aligned}
\int_0^t e^{A_P(t-s)} \,\mathrm{d}s &= e^{A_P t} \int_0^t e^{-A_P s} \,\mathrm{d}s \\
&= \Big/ \frac{d}{ds}\left(-A_P^{-1} e^{-A_P s}\right) = e^{-A_P s} \Big/ \\
&= e^{A_P t} A_P^{-1}\left(I - e^{-A_P t}\right) = A_P^{-1}\left(e^{A_P t} - I\right).
\end{aligned} \tag{66}$$

APPENDIX B
EIGENVALUES OF THE APPROXIMATED EXPONENTIAL FUNCTION

The eigenvalues of $e_{p,m}(Ah)$ as a function of the eigenvalues of A is given in Lemma 8 and Lemma 9 presents the result when $p \to \infty$ if A is Hurwitz.

Lemma 8: Let λ_i and v_i be the eigenvalues and eigenvectors, respectively, of $A \in \mathbb{R}^{n \times n}$. Then the p'th order Taylor expansion $e_{p,m}(Ah)$ of e^{Ah} is given by

$$e_{p,m}(Ah) = \left(I + \frac{Ah}{m} + \ldots + \frac{1}{p!}\left(\frac{Ah}{m}\right)^p\right)^m$$

which has the eigenvectors v_i and the eigenvalues

$$\left(1 + \frac{h\lambda_i}{m} + \frac{h^2\lambda_i^2}{2!m^2} + \frac{h^3\lambda_i^3}{3!m^3} + \ldots + \frac{h^p\lambda_i^p}{p!m^p}\right)^m \tag{67}$$

for $i = 1, \ldots, n$.

Proof: The result follows from an eigenvalue decomposition of the matrix A. ∎

Lemma 9: In the limit $p \to \infty$, the eigenvalues of $e_{p,m}(Ah)$ converge to $e^{h\lambda_i}$, $i = 1, \ldots, n$. If A is Hurwitz ($\mathfrak{Re}\{\lambda_i\} < 0$), then the eigenvalues are inside the unit circle.

Proof: When $p \to \infty$ the sum in (67) converges to the exponential function $e^{h\lambda_i}$, $i = 1, \ldots, n$. The exponential function can be written as

$$e^{\lambda_i h} = e^{\mathfrak{Re}\{\lambda_i\}h}(\cos \mathfrak{Im}\{\lambda_i\} + i \sin \mathfrak{Im}\{\lambda_i\})$$

which for $\mathfrak{Re}\{\lambda_i\} < 0$ has an absolute value less than 1, hence $e^{\lambda_i h}$ is inside the unit circle. ∎

APPENDIX C
RULES FOR VECTORISATION AND THE KRONECKER PRODUCT

The rules for vectorisation and the Kronecker product are from [17] and [23].

$$\operatorname{vec} AB = (I \otimes A)\operatorname{vec} B = (B^{\mathsf{T}} \otimes I)\operatorname{vec} A \tag{68}$$

$$(C^{\mathsf{T}} \otimes A)\operatorname{vec} B = \operatorname{vec} ABC \tag{69}$$

$$I \otimes A + B \otimes I = A \oplus B \tag{70}$$

$$e^{A \oplus B} = e^A \otimes e^B \tag{71}$$

$$DD^{\dagger}(I \otimes A + A \otimes I)D = (I \otimes A + A \otimes I)D \tag{72}$$

ACKNOWLEDGMENT

The authors would like to thank Dr. Daniel Petersson, Linköping University, Sweden, for valuable comments regarding matrix algebra. The authors would also like to thank the reviewers for many constructive comments, in particular one reviewer provided extra-ordinary contributions to improve the manuscript.

REFERENCES

[1] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I – Nonstiff Problems*, ser. Springer Series in Computational Mathematics. Berlin, Heidelberg, Germany: Springer-Verlag, 1987.

[2] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. New York, NY, USA: Academic Press, 1970, vol. 64.

[3] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*, ser. Information and System Sciences Series. Upper Saddle River, NJ, USA: Prentice Hall Inc., 2000.

[4] J. LaViola, "A comparison of unscented and extended Kalman filtering for estimating quaternion motion," in *Proceedings of the American Control Conference*, Denver, CO, USA, June 2003, pp. 2435–2440.

[5] B. Rao, S. Xiao, X. Wang, and Y. Li, "Nonlinear Kalman filtering with numerical integration," *Chinese Journal of Electronics*, vol. 20, no. 3, pp. 452–456, July 2011.

[6] M. Mallick, M. Morelande, and L. Mihaylova, "Continuous-discrete filtering using EKF, UKF, and PF," in *Proceedings of the 15th International Conference on Information Fusion*, Singapore, July 2012, pp. 1087–1094.

[7] J. Bagterp Jörgensen, P. Grove Thomsen, H. Madsen, and M. Rode Kristensen, "A computational efficient and robust implementation of the continuous-discrete extended Kalman filter," in *Proceedings of the American Control Conference*, New York City, USA, July 2007, pp. 3706–3712.

[8] T. Mazzoni, "Computational aspects of continuous-discrete extended Kalman-filtering," *Computational Statistics*, vol. 23, no. 4, pp. 519–539, 2008.

[9] P. Frogerais, J.-J. Bellanger, and L. Senhadji, "Various ways to compute the continuous-discrete extended Kalman filter," *IEEE Transactions on Automatic Control*, vol. 57, no. 4, pp. 1000–1004, April 2012.

[10] S. Särkkä, "On unscented Kalman filtering for state estimation of continuous-time nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1631–1641, September 2007.

[11] P. Zhang, J. Gu, E. Milios, and P. Huynh, "Navigation with IMU/ GPS/digital compass with unscented Kalman filter," in *Proceedings of the IEEE International Conference on Mechatronics and Automation*, Niagara Falls, Ontario, Canada, July–August 2005, pp. 1497–1502.

[12] I. Arasaratnam and S. Haykin, "Cubature Kalman filtering: A powerful tool for aerospace applications," in *Proceedings of the International Radar Conference*, Bordeaux, France, October 2009.

[13] C. F. Van Loan, "Computing integrals involving the matrix exponential," *IEEE Transactions on Automatic Control*, vol. 23, no. 3, pp. 395–404, June 1978.

[14] M. S. Grewal and A. P. Andrews, *Kalman Filtering. Theory and Practice Using* MATLAB, 3rd ed. Hoboken, NJ, USA: John Wiley & Sons, 2008.

[15] S. Särkkä, "Recursive Bayesian inference on stochastic differential equations," Ph.D. dissertation, Helsinki University of Technology, Finland, April 2006, ISBN 9-512-28127-9.

[16] H. J. Rome, "A direct solution to the linear variance equation of a time-invariant linear system," *IEEE Transactions on Automatic Control*, vol. 14, no. 5, pp. 592–593, October 1969.

[17] N. J. Higham, *Functions of Matrices – Theory and Computation*. Philadelphia, PA, USA: SIAM, 2008.

[18] E. J. Davison, "The numerical solution of $\dot{X} = A_1X + XA_2 + D$, $X(0) = C$," *IEEE Transactions on Automatic Control*, vol. 20, no. 4, pp. 566–567, August 1975.

[19] Z. Gajić and M. T. J. Qureshi, *Lyapunov Matrix Equation in System Stability and Control*, ser. Mathematics in Science and Engineering. San Diego, CA, USA: Academic Press, 1995, vol. 195.

[20] W. J. Rugh, *Linear System Theory*, 2nd ed., ser. Information and System Sciences Series. Upper Saddle River, NJ, USA: Prentice Hall Inc., 1996.

[21] C. Moler and C. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM Review*, vol. 45, no. 1, pp. 1–46, February 2003.

[22] D. Hinrichsen and A. J. Pritchard, *Mathematical Systems Theory I – Modelling, State Space Analysis, Stability and Robustness*, ser. Texts in Applied Mathematics. Berlin, Heidelberg, Germany: Springer-Verlag, 2005.

[23] H. Lütkepohl, *Handbook of Matrices*. Chichester, West Sussex, England: John Wiley & Sons, 1996.

[24] N. Wahlström, P. Axelsson, and F. Gustafsson, "Discretizing stochastic dynamical systems using Lyapunov equations," in *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014.

[25] N. J. Higham, "The scaling and squaring method for the matrix exponential revisited," *SIAM Journal on Matrix Analysis and Applications*, vol. 26, no. 4, pp. 1179–1193, June 2005.

Patrik Axelsson received the M.Sc. degree in applied physics and electrical engineering in January 2009 and the Ph.D. degree in Automatic Control in May 2014, both from Linköping University, Linköping, Sweden. His research interests include sensor fusion and control applied to industrial manipulators, with focus on nonlinear flexible joint models.

Fredrik Gustafsson is professor in Sensor Informatics at Department of Electrical Engineering, Linköping University, since 2005. He received the M.Sc. degree in electrical engineering 1988 and the Ph.D. degree in Automatic Control, 1992, both from Linköping University. During 1992-1999 he held various positions in automatic control, and 1999-2005 he had a professorship in Communication Systems. His research interests are in stochastic signal processing, adaptive filtering and change detection, with applications to communication, vehicular, airborne, and audio systems. He is a co-founder of the companies NIRA Dynamics (automotive safety systems), Softube (audio effects) and SenionLab (indoor positioning systems).

He was an associate editor for IEEE Transactions of Signal Processing 2000-2006, IEEE Transactions on Aerospace and Electronic Systems 2010-2012, and EURASIP Journal on Applied Signal Processing 2007-2012. He was awarded the Arnberg prize by the Royal Swedish Academy of Science (KVA) 2004, elected member of the Royal Academy of Engineering Sciences (IVA) 2007, elevated to IEEE Fellow 2011 and awarded the Harry Rowe Mimno Award 2011 for the tutorial "Particle Filter Theory and Practice with Positioning Applications", which was published in the AESS Magazine in July 2010.