Technical report from Automatic Control at Linköpings universitet

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

Patrik Axelsson, Fredrik Gustafsson

Division of Automatic Control

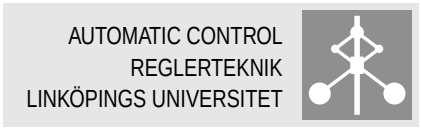
E-mail: axelsson@isy.liu.se, fredrik@isy.liu.se

17th December 2012

Report no.: LiTH-ISY-R-3055
Submitted to IEEE Transactions on Automatic Control

Address:
Department of Electrical Engineering
Linköpings universitet
SE-581 83 Linköping, Sweden

WWW: http://www.control.isy.liu.se



Abstract

Prediction and filtering of continuous-time stochastic processes require a solver of a continuous-time differential Lyapunov equation (CDLE). 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 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

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 require a solver of a continuous-time differential Lyapunov equation (CDLE). 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 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]. Despite this fact, the related area of filtering (state prediction and state estimation) in continuous-time stochastic models is much less studied in literature. 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 an 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. However, we are not aware of any publication where this analytical approach is used. One problem is the large dimension of the resulting ODE. Another possible explanation 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. This involves approximations, though, 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. [2]–[4]. 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 [5], [6], where the first and second order moments are integrated numerically. A comparison between different solutions is presented in [7], where the method proposed by the authors discretize the stochastic differential equation (SDE) using a Runge-Kutta solver. The other methods in [7] have been proposed in the literature before, e.g. [2], [6]. Related work using different approximations to continuous integration problems in nonlinear filtering also appears in [8], [9] for unscented Kalman filters and [10] for cubature Kalman filters.

This contribution takes a new look at this fundamental problem. First, we review the mathematical framework and different approaches for solving the CDLE. 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 analytical solution with arbitrary accuracy. We illustrate the results on both a simple second order linear spring-damper system, and a non-linear spring-damper system relevant for mechanical systems, in particular robotics.

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).

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 independent Wiener processes with $E\left[d\beta(t)d\beta^\mathsf{T}\right] = Qdt$. 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) but that is omitted here for brevity.

The first and second order moments, $\hat{x}(t)$ and $P(t)$ respectively, of the stochastic variable $x(t)$ are propagated by [11]

$$\dot{\hat{x}}(t) = A\hat{x}(t), \tag{2a}$$
$$\dot{P}(t) = AP(t) + P(t)A^\mathsf{T} + GQG^\mathsf{T}, \tag{2b}$$

where (2a) is an ordinary ODE and (2b) is the continuous-time differential Lyapunov equation (CDLE). Equation (2) can be converted to one single ODE by introducing an extended state vector

$$z(t) = \begin{pmatrix} z_x(t) \\ z_P(t) \end{pmatrix} = \begin{pmatrix} x(t) \\ \operatorname{vech} P(t) \end{pmatrix}, \tag{3}$$
$$\dot{z}(t) = \begin{pmatrix} Az_x(t) \\ A_P z_P(t) + \operatorname{vech} GQG^\mathsf{T} \end{pmatrix} = A_z z(t) + B_z, \tag{4}$$

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 to an ODE of the kind

$$\dot{x}(t) = Ax(t) + B \tag{5}$$

is given by [12]

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}\,\mathrm{d}\tau B \tag{6}$$

Using (6) at the discrete-time instants $t = kh$ and $t = (k+1)h$ gives the following recursive scheme

$$x((k+1)h) = e^{Ah}x(kh) + \int_0^h e^{A\tau}\,\mathrm{d}\tau B \tag{7}$$

This contribution examines different ways to solve (2a) and (2b) for discrete-time instants, such as is required in Kalman filters.

For implementation in software, the expression in (6) can be calculated according to

$$x(t) = \begin{pmatrix} I & 0 \end{pmatrix} \exp\left[\begin{pmatrix} A & I \\ 0 & 0 \end{pmatrix} t\right] \begin{pmatrix} x(0) \\ B \end{pmatrix} \tag{8}$$

which is easily verified by using the Taylor expansion definition of the matrix exponential $\exp(A) = I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \dots$.

B. The Matrix Exponential

The ODE solutions (6) and (7) show that the matrix exponential function is a working horse for all ODE solvers. 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, [13]

$$\begin{aligned} e^{Ah} &= \left(e^{Ah/m}\right)^m \\ &\approx \left(I + \left(\frac{Ah}{m}\right) + \cdots + \frac{1}{p!}\left(\frac{Ah}{m}\right)^p\right)^m \triangleq e_{p,m}(Ah). \end{aligned} \tag{9}$$

In fact, the Taylor expansion is a special case of a more general Padé approximation of $e^{Ah/m}$ [13], 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

$$\lim_{p\to\infty} e_{p,m}(Ah) = \left(e^{Ah/m}\right)^m = e^{Ah}. \tag{10a}$$

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

$$\lim_{m\to\infty} e_{p,m}(Ah) = e^{Ah}. \tag{10b}$$

Finally, from [14] we have that

$$\left\|e^{Ah} - e_{p,m}(Ah)\right\| \leq \frac{\|A\|^{p+1}\,h^{p+1}}{m^p(p+1)!}e^{\|A\|h}. \tag{10c}$$

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 results in $e_{4,1}(Ah)$.

C. Analytical Solution to the SDE

The analytical solution to the SDE, expressed in the form of the ODE (4), using (7) is given by

$$z((k+1)h) = e^{A_z h}z(kh) + \int_0^h e^{A_z\tau}\,\mathrm{d}\tau B_z \tag{11a}$$

Thus, the solution to (2a) and (2b) can be solved using existing solvers for the matrix exponential.

One potentially prohibitive drawback with the analytical solution is its computational complexity, in particular for the matrix exponential. The dimension of the extended state z is $n_z = n_x + n_x\,(n_x + 1)\,/2$, giving a computational complexity of $(\log_2(m) + p)\left(n_x^2/2\right)^3$.

D. Approximate Solution using the Discrete-time Difference Lyapunov Equation

A common approach in practice, in particular in Kalman filter applications, is to first discretize the system to

$$x(k+1) = F_h x(k) + G_h v_h(k), \tag{12a}$$
$$\mathrm{cov}(v_h(t)) = Q_h. \tag{12b}$$

and then use

$$\hat{x}(k+1) = F_h\hat{x}(k), \tag{13a}$$
$$P(k+1) = F_h P(k) F_h^{\mathsf{T}} + G_h Q_h G_h^{\mathsf{T}}, \tag{13b}$$

where (13a) is a difference equation and (13b) is the discrete-time difference Lyapunov equation (DDLE). This solution is exact for the discrete time model (12). However, there are several approximations involved here:

- First, F_h is an approximation $e_{p,m}(Ah)$ 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 (13b) is not equivalent to (2b).
- The discrete-time noise $v_h(t)$ is an aggregation of the total effect of the Wiener process $d\beta(t)$ during the interval $[t, t+h]$. Most severly, (13b) is a coarse approximation of the solution to (2b). One standard approach is to assume that the noise process is piece-wise constant, in which case $G_h = \int_0^h e^{At}dtG$. 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 (13) is iterated m times using the sampling time h/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.

E. Summary of Contributions

- Section III gives explicit conditions for stability of both x and P, for the case of Euler sampling $e_{1,m}(A)$. See Table I for a summary.
- Section IV extends the stability conditions from $p = 1$ to $p = 2, 3, 4$, including the standard fourth order Runge-Kutta schema. Conditions for the Runge-Kutta schema are summarized in Table I.
- Section V shows how the computational complexity in the analytical solution can be decreased from $(\log_2(m) + p)\left(n_x^2/2\right)^3$ to $(\log_2(m) + p + 43)\, n_x^3$.
- 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.

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	$-\frac{2\Re\mathrm{e}\{\lambda_i\}}{\lvert\lambda_i\rvert^2}$
Oversampled Euler	1	$m>1$	$-\frac{2m\Re\mathrm{e}\{\lambda_i\}}{\lvert\lambda_i\rvert^2}$
Runge-Kutta	4	1	$-\frac{2.7852}{\lambda_i}$, $\lambda_i \in \mathbb{R}$
Oversampled Runge-Kutta	4	$m>1$	$-\frac{2.7852m}{\lambda_i}$, $\lambda_i \in \mathbb{R}$

III. STABILITY ANALYSIS FOR EULER SAMPLING, $p = 1$

The recursive solution of the SDE (1), in the form of the ODE

$$\dot{z} = A_z z(t) + B_z \tag{14}$$

is given by

$$z((k+1)h) = e^{A_z h} z(kh) + \int_0^h e^{A_z \tau}\, \mathrm{d}\tau B_z, \tag{15}$$

as presented in Section II-C. The solution is stable for all h according to Lemma 8 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.

First, the structure of the ODE in (14) is exploited. From Section II-A we have that

$$\begin{aligned}\dot{z}(t) &= \begin{pmatrix} A z_x(t) \\ D^\dagger \left(I \otimes A + A \otimes I\right) D z_P(t) + \mathrm{vec}\left(GQG^\mathsf{T}\right)\end{pmatrix} \\ &= \begin{pmatrix} A & 0 \\ 0 & A_P \end{pmatrix} z(t) + \begin{pmatrix} 0 \\ \mathrm{vec}\left(GQG^\mathsf{T}\right)\end{pmatrix},\end{aligned} \tag{16}$$

where $A_P = D^\dagger \left(I \otimes A + A \otimes I\right) D$, hence the matrix A_z is diagonal which means that calculation of the matrix exponential $e^{A_z h}$ can be separated into e^{Ah} and $e^{A_P h}$. 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 is in the left half plane, i.e., $\Re\mathrm{e}\{\lambda_i\} < 0$, $i = 1, \ldots, n_x$. From [15] it is known that the eigenvalues of A_P are given by $\lambda_i + \lambda_j$, $1 \le i \le j \le n_x$, hence the ODE describing the CDLE is also stable. In order to keeping 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

$$h < \min\left\{ -\frac{2m\Re\mathrm{e}\{\lambda_i + \lambda_j\}}{\lvert\lambda_i + \lambda_j\rvert^2},\ 1 \le i \le j \le n_x \right\}, \tag{17}$$

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

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 7 in Appendix B, given by $(1 + \lambda_i h/m)^m$. The eigenvalues are inside the unit circle if $\lvert(1 + \lambda_i h/m)^m\rvert < 1$, where

$$\left\lvert \left(1 + \frac{\lambda_i h}{m}\right)^m \right\rvert = \left(\frac{1}{m} \sqrt{m^2 + 2a_i h m + (a_i^2 + b_i^2)h^2} \right)^m. \tag{18}$$

In (18), the parametrization $\lambda_i = a_i + i b_i$ has been used. Solving $\lvert(1 + \lambda_i h/m)^m\rvert < 1$ for h and using the fact $\lvert\lambda_i\rvert^2 = a_i^2 + b_i^2$ give

$$h < -\frac{2ma_i}{\lvert\lambda_i\rvert^2}. \tag{19}$$

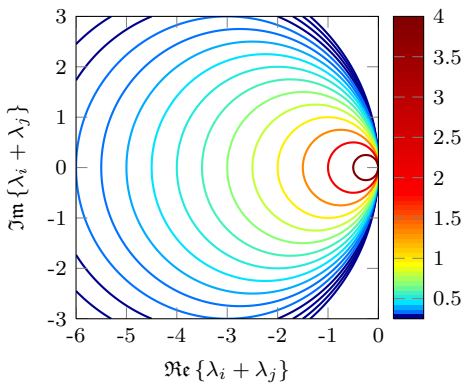


Fig. 1. Level curves of (17), where the colors indicate the values on h.

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

$$h < -\frac{2m(a_i + a_j)}{\lvert\lambda_i + \lambda_j\rvert^2}, \quad 1 \le i \le j \le n_x. \tag{20}$$

Using $\lambda_i = \lambda_j$ in (20) gives

$$-\frac{2m(a_i + a_j)}{\lvert\lambda_i + \lambda_j\rvert^2} = -\frac{4ma_i}{\lvert 2\lambda_i\rvert^2} = -\frac{ma_i}{\lvert\lambda_i\rvert^2}, \tag{21}$$

which is half as much as the bound in (19), hence the upper bound for h is given by (20). ∎

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 (17) for $h = c =$ constant in the complex plane are given by

$$\begin{aligned} -\frac{2a_{ij}m}{a_{ij}^2 + b_{ij}^2} = c &\Leftrightarrow c a_{ij}^2 + c b_{ij}^2 + 2a_{ij}m \\ &= c\left(\left(a_{ij} + \frac{m}{c}\right)^2 - \frac{m^2}{c^2}\right) + c b_{ij}^2 = 0 \\ &\Leftrightarrow \left(a_{ij} + \frac{m}{c}\right)^2 + b_{ij}^2 = \frac{m^2}{c^2} \end{aligned} \tag{22}$$

where $a_{ij} = \Re\mathrm{e}\{\lambda_i + \lambda_j\}$ and $b_{ij} = \Im\mathrm{m}\{\lambda_i + \lambda_j\}$. Equation (22) 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.

IV. STABILITY ANALYSIS FOR HIGHER ORDERS OF APPROXIMATION, $p > 1$

Stability conditions for higher orders of approximations of e^{Ah} will be derived in this section. We restrict the discussion to real eigenvalues for simplicity, and also to the ODE $\dot{x}(t) = Ax(t)$. The ODE for vech $P(t)$ gives the same result but with the eigenvalues for A_P instead of the eigenvalues for A.

Starting with the second order approximation

$$e_{2,m}(Ah) = \left(I + \frac{Ah}{m} + \frac{1}{2}\left(\frac{Ah}{m}\right)^2\right)^m \tag{23}$$

which has the eigenvalues

$$\left(1 + \frac{h\lambda_i}{m} + \frac{1}{2}\left(\frac{h\lambda_i}{m}\right)^2\right)^m \tag{24}$$

gives that

$$\left\lvert \left(1 + \frac{h\lambda_i}{m} + \frac{1}{2}\left(\frac{h\lambda_i}{m}\right)^2\right)^m \right\rvert < 1. \tag{25}$$

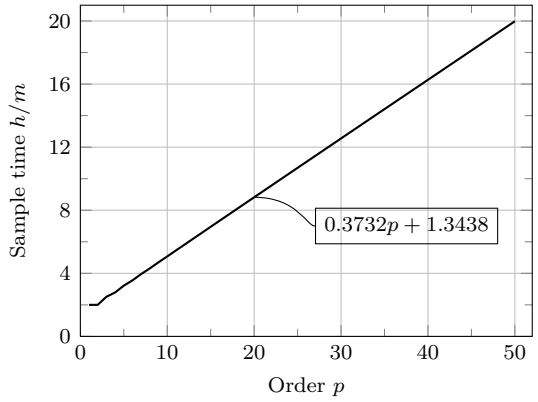


Fig. 2. Upper bound of the sample time h/m for different orders of the Taylor expansion of e^{Ah}, i.e., $e_{p,m}(Ah)$.

Inequality (25) is satisfied if

$$h < -\frac{2m}{\lambda_i}, \tag{26}$$

where it has been used that the real solutions to $|x^m| < 1 \Leftrightarrow -1 < x^m < 1$ have to satisfy $x < 1$, $x < -1$, $x > -1$, hence $-1 < x < 1$.

The stability condition for the second order approximation is consequently the same as the condition for the first order approximation. It means that the accuracy has increased, recall (10c), but the stability condition remains the same.

Increasing the order of approximation even more, results in a third or higher order polynomial inequality that has to be solved. To manage to solve these equations in the case of a third and fourth order approximation, a computer program for symbolic calculations can be used, e.g. Maple or Mathematica. The result is

$$h < -\frac{2.5127m}{\lambda_i}, \quad h < -\frac{2.7852m}{\lambda_i} \tag{27}$$

for the third and fourth order of approximation, respectively, where we only consider real eigenvalues. For complex eigenvalues, numerical solutions are to prefer. We can see that the stability bound increases when the order of approximation increases. In fact, we can show numerical that the stability bound increases linearly when the order of approximation is increased. We are interested in the constant in the numerator of the expressions describing the bound, e.g. 2.5127 for the third order of approximation. Therefore, the eigenvalue λ_i is fixed to -1. The result is shown in Figure 2 where it can be seen that the bound increases linearly according to $h/m = 0.3732p + 1.3438$. It is reasonably to say that the linear increase in the upper bound of h/m also holds for other real values of λ_i.

V. Decreased Computational Complexity for the CDLE

The analytical solution of the SDE (1) given by (11a) can be separated into two parts as is explained in Section III. The solution to the ODE describing the second order moment is given in the following lemma, where $\widetilde{Q} \triangleq GQG^{\mathsf{T}}$ has been introduced.

Lemma 2: The solution to the vectorized CDLE

$$\operatorname{vech} \dot{P}(t) = A_P \operatorname{vech} P(t) + \operatorname{vech} \widetilde{Q}, \tag{28}$$

is given by

$$\operatorname{vech} P(t) = e^{A_P t} \operatorname{vech} P(0) + A_P^{-1}(e^{A_P t} - I) \operatorname{vech} \widetilde{Q}. \tag{29}$$

Proof: See Appendix A. ∎

Note that the factor $A_P^{-1}(e^{A_P t} - I)$ can be computed using a Taylor series expansion, hence A_P^{-1} does not have to be computed explicit. The solution still suffers from the curse of dimensionality, since the size of the matrix A_P is quadratic in the number of states. In this section the CDLE (2b) will be solved without blowing up the dimensions of the problem, i.e., keep the dimensions to the same size as the state vector. The analytical solution to the matrix valued CDLE (2b) is given by [16]

$$P(t) = e^{At} P(0) e^{A^{\mathsf{T}} t} + \int_0^t e^{A(t-s)} \widetilde{Q} e^{A^{\mathsf{T}}(t-s)} \, \mathrm{d}s \tag{30}$$

As we can see, the matrix exponential is still used but now with the matrix A instead of A_P. The integral can be solved using numerical integration, e.g. the trapezoidal method or the rectangle method. In [17] the integral is solved analytically when A is diagonalizable. A way to diagonalize a matrix is to compute the eigenvalue decomposition. However, all matrices cannot be diagonalizable using real matrices, which gives rise to complex matrices.

Remark 3: In theory, the eigenvalue decomposition will work. However, the eigenvalue decomposition is not numerically stable [14].

In Theorem 4, a new solution based on Lemma 2 is presented. The solution contains the matrix exponential and the solution of an algebraic Lyapunov equation for which efficient numerical solvers exist.

Theorem 4: The solution of the CDLE (2b) is given by

$$P(t) = e^{At} P(0) e^{A^{\mathsf{T}} t} + \bar{P}, \tag{31a}$$

$$A\bar{P} + \bar{P}A^{\mathsf{T}} + \widetilde{Q} - \widehat{Q} = 0, \tag{31b}$$

$$\widehat{Q} = e^{At} \widetilde{Q} e^{A^{\mathsf{T}} t}. \tag{31c}$$

Proof: Taylor expansion of the matrix exponential gives

$$e^{A_P t} = I + A_P t + \frac{A_P^2 t^2}{2!} + \frac{A_P^3 t^3}{3!} + \dots \tag{32}$$

Using (70) in Appendix C, each term in the Taylor expansion can be rewritten according to

$$A_P^k t^k = D^{\dagger}(I \otimes A + A \otimes I)^k t^k D, \tag{33}$$

hence

$$e^{A_P t} = D^{\dagger} e^{(I \otimes A + A \otimes I)t} D \overset{(68),(69)}{=} D^{\dagger}(e^{At} \otimes e^{At}) D. \tag{34}$$

The first term in (29) can now be written as

$$\begin{aligned} e^{A_P t} \operatorname{vech} P(0) &= D^{\dagger}(e^{At} \otimes e^{At}) D \operatorname{vech} P(0) \\ &= D^{\dagger}(e^{At} \otimes e^{At}) \operatorname{vec} P(0) \overset{(67)}{=} D^{\dagger} \operatorname{vec} e^{At} P(0) e^{A^{\mathsf{T}} t} \\ &= \operatorname{vech} e^{At} P(0) e^{A^{\mathsf{T}} t}. \end{aligned} \tag{35}$$

Similar calculations give

$$e^{A_P t} \operatorname{vech} \widetilde{Q} = D^{\dagger} \operatorname{vec} e^{At} \widetilde{Q} e^{A^{\mathsf{T}} t} = \operatorname{vech} \widehat{Q}. \tag{36}$$

The last term in (29) can be rewritten according to

$$A_P^{-1}(e^{A_P t} - I) \operatorname{vech} \widetilde{Q} = A_P^{-1} \operatorname{vech}(\widehat{Q} - \widetilde{Q}) \triangleq \operatorname{vech} \bar{P}. \tag{37}$$

Equation (37) can be seen as the solution of the linear system of equations $A_P \operatorname{vech} \bar{P} = \operatorname{vech}(\widehat{Q} - \widetilde{Q})$. Using the derivation in (62) in Appendix A backwards gives that $\bar{P}$ is the solution to the algebraic Lyapunov equation

$$A\bar{P} + \bar{P}A^{\mathsf{T}} + \widetilde{Q} - \widehat{Q} = 0. \tag{38}$$

Combining (35) and (37) gives that (29) can be written as

$$\operatorname{vech} P(t) = \operatorname{vech} e^{At} P(0) e^{A^{\mathsf{T}} t} + \operatorname{vech} \bar{P} = \tag{39}$$

$$\Leftrightarrow$$

$$P(t) = e^{At} P(0) e^{A^{\mathsf{T}} t} + \bar{P}, \tag{40}$$

where $\bar{P}$ is the solution to (38). ∎

A. Discrete-time Recursion

The recursive solution to the differential equations in (2) describing the first and second order moments of the SDE (1) can now be written as

$$\hat{x}((k+1)h) = e^{Ah}\hat{x}(kh), \tag{41a}$$

$$P((k+1)h) = e^{Ah}P(kh)e^{A^{\mathsf{T}}h} + \bar{P}, \tag{41b}$$

where

$$A\bar{P} + \bar{P}A^{\mathsf{T}} + \widetilde{Q} - \widehat{Q} = 0, \tag{41c}$$

$$\widehat{Q} = e^{Ah}\widetilde{Q}e^{A^{\mathsf{T}}h}. \tag{41d}$$

Equations (41b) to (41d) are derived using $t = kh$ and $t = (k+1)h$ in (31).

The method presented in Theorem 4 is derived straightforwardly from Lemma 2. A similar solution that also solves an algebraic Lyapunov function is presented in [18]. The main difference is that the algebraic Lyapunov function in [18] is independent of time, which is not the case here since $\widehat{Q}$ changes with time. This is not an issue for the recursive time update due to the fact that $\widehat{Q}$ is only dependent on h, hence the algebraic Lyapunov equation (41c) has to be solved only once.

B. Evaluation of Computational Complexity for Solving the CDLE

The time it takes to calculate $P(t)$ using Theorem 4 and Lemma 2 is considered here. We also compare with the time it takes using the solution presented in [17], where the integral in (30) is calculated using an eigenvalue decomposition of A.

Using Lemma 2 to solve (2b) involves inversion of the $n_P \times n_P$ matrix A_P, where $n_P = n_x(n_x+1)/2$, thus the computational complexity for inverting A_P is $2n_P^3 = 2(n_x^2/2)^3$. Using the Taylor expansion of $A_P^{-1}(e^{A_P t} - I)$ will also be in the order of $(n_x^2/2)^3$. If instead Theorem 4 is used the inversion of A_P has been reduced to solving the Lyapunov equation (31b) where the dimensions of the matrices are $n_x \times n_x$. The computational complexity for solving the Lyapunov equation is $35n_x^3$ [14]. The total computational complexity for computing the solution of (2b) using Theorem 4 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.

Monte Carlo simulations over 100 randomly chosen stable systems are performed in order to compare the three methods. The order of the systems are $n_x = 10, 50, 100, 500, 1000$. As expected, the solution using Lemma 2 takes very long time as can be seen in Figure 3. We can also see that no solution is obtained for $n \geq 500$ because of too large matrices to be handled in MATLAB. The computational time for Theorem 4 and (30) are in the same order, which is also expected. The difference can be explained by the eigenvalue decomposition, more matrix multiplications and more memory management for (30). The slope of the lines for large n_x is approximately 6 for (29) and 3 for (30) and (31), which agree with the computational complexity discussed above.

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

$$m\ddot{q} + d\dot{q} + kq - mg = 0 \tag{42}$$

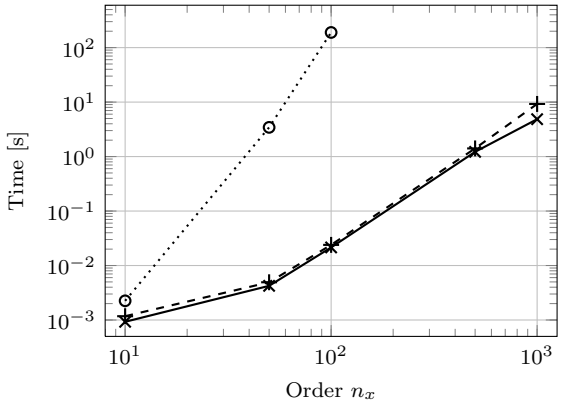


Fig. 3. Mean execution time for calculating $P(t)$ using (29) (dotted), (30) (dashed) and (31) (solid).

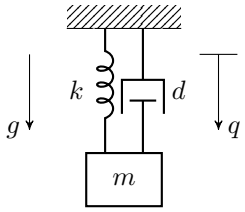


Fig. 4. A mass hanging in a spring and damper.

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

$$\dot{x}(t) = \underbrace{\begin{pmatrix} 0 & 1 \\ -k & -d \end{pmatrix}}_{A} x(t) + \underbrace{\begin{pmatrix} 0 \\ g \end{pmatrix}}_{B}. \tag{43}$$

A. Stability Bound on the Sample Time

The bound on the sample time that makes the solution to (43) stable when $e_{1,m}(Ah)$ is used, can be calculated using Theorem 1. The eigenvalues for A are

$$\lambda_{1,2} = -\frac{d}{2} \pm \frac{1}{2}\sqrt{d^2 - 4k}. \tag{44}$$

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

$$\begin{aligned} h &< \min\left\{ \frac{2m}{d + \sqrt{d^2 - 4k}}, \frac{2m}{d - \sqrt{d^2 - 4k}}, \frac{2m}{d} \right\} \\ &= \frac{2m}{d + \sqrt{d^2 - 4k}}, \end{aligned} \tag{45}$$

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

$$h < \min\left\{ \frac{dm}{2k}, \frac{2m}{d}, \frac{dm}{2k} \right\} = \frac{dm}{2k}, \tag{46}$$

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 (43) is augmented with process noise giving the model

$$dx(t) = Ax(t)dt + B + Gd\beta(t), \tag{47}$$

where A and B are given by (43), $G = \begin{pmatrix} 0 & 1 \end{pmatrix}^{\mathsf{T}}$ and $d\beta(t)$ is a scalar Wiener process with $E\left[d\beta(t)d\beta^{\mathsf{T}}\right] = Qdt$. 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

$$y_k = \begin{pmatrix} 0 & 1 \end{pmatrix} x_k + e_k = Cx_k + e_k, \tag{48}$$

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, $y_k \triangleq y(kh)$ 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 $T_s = h = 0.09\,\text{s}$.

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 $t_{\max} = 20\,\text{s}$ 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`. Moreover, the update of the covariance matrix $P(t)$ is according to the discrete filter (13b) or according to the solution to the CDLE given by Theorem 4. In summary, the Kalman filters are:

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

where $\bar{P}$ is the solution to the Lyapunov equation in (41c).

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:

$$\{1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ 20,\ 30,\ 40,\ 50\} \tag{49}$$

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

$$\rho_i = \sqrt{\frac{1}{N}\sum_{t=t_0}^{t_{\max}} \left(\rho_i^{MC}(t)\right)^2} \tag{50a}$$

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

$$\rho_i^{MC}(t) = \sqrt{\frac{1}{N_{MC}}\sum_{j=1}^{N_{MC}} \left(x_i^j(t) - \hat{x}_i^j(t)\right)^2}, \tag{50b}$$

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 analytical solution[1]. 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 analytical solution is to prefer.

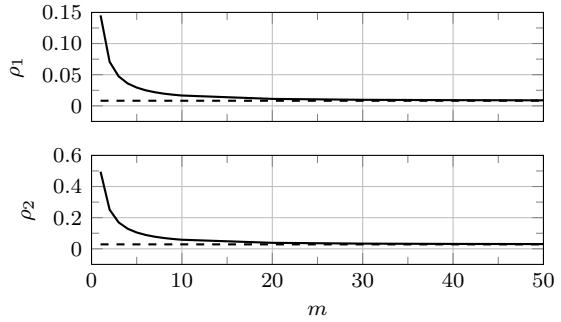


Fig. 5. RMSE according to (50) as a function of the degree of oversampling, where the solid line is filter 1 (filter 3 gives the same) and the dashed line is filter 4 (filter 2 gives the same).

Remark 5: 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[2] 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 (13b), i.e., filter 1 and 2, the stationary value is too large for small values of m. For the continuous-time update in Theorem 4, 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

$$dx(t) = f(x(t))dt + G(x(t))d\beta(t), \tag{51}$$

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 independent Wiener processes with $E\left[d\beta(t)d\beta^{\mathsf{T}}\right] = Qdt$. For simplicity, it is assumed that $G(x(t)) \triangleq G$. The propagation of the first and second order

[1] It is wisely to choose m to be a power of 2, as explained in Section II-B

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

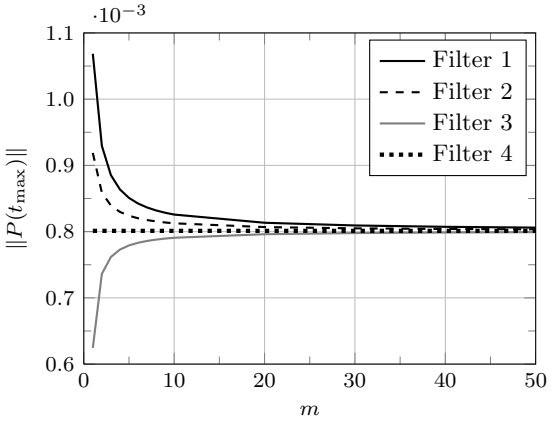


Fig. 6. The norm of the stationary covariance matrix for the estimation error for the four filters, as a function of the degree of oversampling.

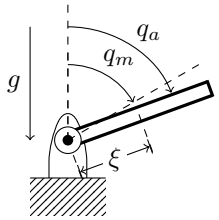


Fig. 7. A single flexible joint.

moments for an extended Kalman filter (EKF) can, as in the linear case, be written as

$$\dot{\hat{x}}(t) = f(\hat{x}(t)), \tag{52a}$$
$$\dot{P}(t) = F(\hat{x}(t))P(t) + P(t)F(\hat{x}(t))^{\mathsf{T}} + GQG^{\mathsf{T}}, \tag{52b}$$

where $F(\hat{x}(t))$ is the Jacobian of $f(x(t))$ evaluated at $\hat{x}(t)$. The main differences to (2) 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 (52) have to be solved simultaneously. The easiest way is to vectorize (52b) similar to what is described in Appendix A and then solve the nonlinear ODE

$$\frac{d}{dt}\begin{pmatrix}\hat{x}(t)\\ \operatorname{vech} P(t)\end{pmatrix} = \begin{pmatrix} f(\hat{x}(t)),\\ A_P(\hat{x}(t))\operatorname{vech} P + \operatorname{vech} GQG^{\mathsf{T}}\end{pmatrix}, \tag{53}$$

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 4, where $A \triangleq F(\hat{x}(t))$.

Remark 6: 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.

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

$$J_a\ddot{q}_a + G(q_a) + D(\dot{q}_a, \dot{q}_m) + T(q_a, q_m) = 0, \tag{54a}$$
$$J_m\ddot{q}_m + F(\dot{q}_m) - D(\dot{q}_a, \dot{q}_m) - T(q_a, q_m) = u, \tag{54b}$$

TABLE II. 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 the gravity, damping, spring, and friction torques are modeled as

$$G(q_a) = -gm\xi\sin(q_a), \tag{55a}$$
$$D(\dot{q}_a, \dot{q}_m) = d(\dot{q}_a - \dot{q}_m), \tag{55b}$$
$$T(q_a, q_m) = k_2(q_a - q_m) + k_1(q_a - q_m)^3, \tag{55c}$$
$$F(\dot{q}_m) = f_d\dot{q}_m, \tag{55d}$$

Numerical values of the parameters, used for simulation, are given in Table II. 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

$$\dot{x} = \underbrace{\begin{pmatrix} x_3 \\ x_4 \\ \frac{1}{J_a}\left(gm\xi\sin(x_1) - d\Delta_{34} - k_2\Delta_{12} - k_1\Delta_{12}^3\right) \\ \frac{1}{J_m}\left(d\Delta_{34} + k_2\Delta_{12} + k_1\Delta_{12}^3 - f_dx_4 + u\right)\end{pmatrix}}_{f(x,u)} \tag{56}$$

where $\Delta_{ij} = x_i - x_j$. The state space model (56) is also augmented with a noise model according to (51) with

$$G = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ J_a^{-1} & 0 \\ 0 & J_m^{-1}\end{pmatrix} \tag{57}$$

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 fourth order Runge-Kutta 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, i.e.,

$$y(kh) = \begin{pmatrix}0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}x(kh) + e(kh), \tag{58}$$

where $e(kh) \in \mathbb{R}^2$ is discrete-time zero mean Gaussian measurement noise with a standard deviation $\sigma = 0.05I_2$. The sample time for the measurements is chosen to be $T_s = 0.1$ s.

Two extended Kalman filters (EKF) are compared. The first filter uses the discrete-time update (13) where Euler sampling

$$x((k+1)h) = \underbrace{x(kh) + hf(x(kh), u(kh))}_{F(x(kh))} \tag{59}$$

has been used for discretisation. The second filter solves the continuous-time ODE (53) 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 100 Monte Carlo simulations using the different values of m listed in (49).

Figure 8 shows the RMSE, defined in (50), 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.

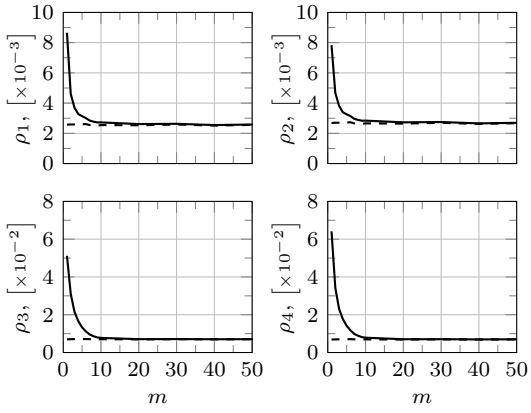


Fig. 8. RMSE according to (50), where the solid line is the discrete-time filter using Euler sampling and the dashed line is the continuous-time filter using a Runge-Kutta solver.

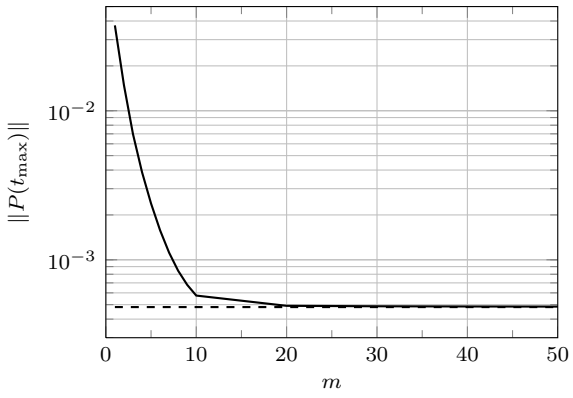


Fig. 9. The norm of the stationary covariance matrix for the estimation error for the two filters.

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 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 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) Stability condition for Euler sampling of the linear ODE. An explicit upper bound on the sample time is derived such that the discrete-time system has all its eigenvalues inside the unit circle, i.e., a stable system. The stability condition for higher order of approximations is also briefly investigated.
2) 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 has 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

$$\dot{P}(t) = AP(t) + P(t)A^{\mathsf{T}} + GQG^{\mathsf{T}}, \tag{60}$$

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

$$\operatorname{vec} P(t) = D \operatorname{vech} P(t), \tag{61}$$

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 (60) gives

$$\begin{aligned}
\operatorname{vech} \dot{P}(t) &= \operatorname{vech}\left(AP(t) + P(t)A^{\mathsf{T}} + \widetilde{Q}\right) \\
&= \operatorname{vech} AP(t) + \operatorname{vech} P(t)A^{\mathsf{T}} + \operatorname{vech} \widetilde{Q} \\
&= D^{\dagger}(\operatorname{vec} AP(t) + \operatorname{vec} P(t)A^{\mathsf{T}}) + \operatorname{vech} \widetilde{Q} \\
&\overset{(66)}{=} D^{\dagger}[(I \otimes A) + (A \otimes I)]D \operatorname{vech} P(t) + \operatorname{vech} \widetilde{Q} \\
&= A_P \operatorname{vech} P(t) + \operatorname{vech} \widetilde{Q}
\end{aligned} \tag{62}$$

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 (62) is given by

$$\begin{aligned}
\operatorname{vech} P(t) &= e^{A_P t} \operatorname{vech} P(0) + \int_0^t e^{A_P(t-s)} \,\mathrm{d}s \operatorname{vech} \widetilde{Q} \\
&= e^{A_P t} \operatorname{vech} P(0) + A_P^{-1}(e^{A_P t} - I) \operatorname{vech} \widetilde{Q},
\end{aligned} \tag{63}$$

where it has been used that

$$\begin{aligned}
\int_0^t e^{A_P(t-s)} \,\mathrm{d}s &= e^{A_P t} \int_0^t e^{-A_P s} \,\mathrm{d}s \\
&= \Big/ \frac{d}{dt}\left(-A_P^{-1} e^{-A_P s}\right) = e^{-A_P s} \Big/ \\
&= e^{A_P t} A_P^{-1}\left(I - e^{-A_P t}\right) = A_P^{-1}\left(e^{A_P t} - I\right).
\end{aligned} \tag{64}$$

The complexity for solving (62) using (63) is $\mathcal{O}(n_P^3) = \mathcal{O}(n_x^6)$.

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 7 and Lemma 8 presents the result when $p \to \infty$ if A is Hurwitz.

Lemma 7: 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

$$\left(1 + \frac{h\lambda_i}{m} + \frac{h^2\lambda_i^2}{2!m^2} + \frac{h^3\lambda_i^3}{3!m^3} + \ldots + \frac{h^p\lambda_i^p}{p!m^p}\right)^m \tag{65}$$

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

Proof: Let the eigenvalue decomposition $A = V\Lambda V^{-1}$ be given, where the columns in V are the eigenvectors v_i and Λ is a diagonal matrix with the eigenvalues λ_i. Using $A = V\Lambda V^{-1}$ in the pth Taylor expansion of e^{Ah} gives

$$\begin{aligned} e_{p,m}(Ah) =& \left(I + \frac{Ah}{m} + \ldots + \frac{1}{p!}\left(\frac{Ah}{m}\right)^p\right)^m \\ =& V\left(I + \frac{\Lambda h}{m} + \ldots + \frac{1}{p!}\left(\frac{\Lambda h}{m}\right)^p\right)^m V^{-1}. \end{aligned}$$

The eigenvectors to $e_{p,m}(Ah)$ are now given by v_i and the eigenvalues by

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

for $i = 1, \ldots, n$. ∎

Lemma 8: 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 (65), that describes the eigenvalues of the pth order Taylor approximation, converges to the exponential function, hence the eigenvalues converge to $e^{h\lambda_i}$, $i = 1, \ldots, n$. The exponential function can be written as

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

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 [14] and [15].

$$\text{vec}\, AB = (I \otimes A)\text{vec}\, B = (B^\mathsf{T} \otimes I)\text{vec}\, A \tag{66}$$

$$(C^\mathsf{T} \otimes A)\text{vec}\, B = \text{vec}\, ABC \tag{67}$$

$$I \otimes A + B \otimes I = A \oplus B \tag{68}$$

$$e^{A\oplus B} = e^A \otimes e^B \tag{69}$$

$$DD^\dagger(I \otimes A + A \otimes I)D = (I \otimes A + A \otimes I)D \tag{70}$$

ACKNOWLEDGMENT

The authors would like to thank D. Petersson for valuable comments regarding matrix algebra.

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] 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.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*, ser. Mathematics in Science and Engineering. New York, NY, USA: Academic Press, 1970, vol. 64.

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

[13] 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.

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

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

[16] 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.

[17] 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.

[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.

Avdelning, Institution
Division, Department

Division of Automatic Control
Department of Electrical Engineering

Datum
Date

2012-12-17

Språk
Language

☐ Svenska/Swedish
☒ Engelska/English

☐ _____

Rapporttyp
Report category

☐ Licentiatavhandling
☐ Examensarbete
☐ C-uppsats
☐ D-uppsats
☒ Övrig rapport
☐ _____

ISBN
—

ISRN
—

Serietitel och serienummer
Title of series, numbering

ISSN
1400-3902

LiTH-ISY-R-3055

URL för elektronisk version

http://www.control.isy.liu.se

Titel
Title

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

Författare
Author

Patrik Axelsson, Fredrik Gustafsson

Sammanfattning
Abstract

Prediction and filtering of continuous-time stochastic processes require a solver of a continuous-time differential Lyapunov equation (CDLE). 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 low-complexity analytical solution that does not involve oversampling. The results are illustrated on Kalman filtering problems in both linear and nonlinear systems.

Nyckelord
Keywords

Continuous time systems, Discrete time systems, Kalman filters, Sampling methods