TMR4515 - Advanced Model-Based Design and Testing of Marine Control Systems

Maiken Berthelsen, Toni Klausen, Signe Moltu

Abstract: The first part of the project consists of designing a Kalman filter to be implemented on a benchmark example, consisting of two motors driving two loads through flexible couplings. The continuous plant needs to be discretized, to implement a discrete time Kalman filter. The second part of the project consists of designing dynamic hypothesis testing. Two parameters of the plant are uncertain, and can be assumed to lie within one of three given subsets. Three hypotheses are created by assuming an approximate mean value for the uncertain parameters. For each hypothesis the Kalman gains are calculated using a discrete, steady-state Kalman filter. The probability of each hypothesis being the correct one is then calculated. It is shown that the system converges to employing the correct hypothesis when the uncertain parameters are changed between the values from the different subsets.

INTRODUCTION

In the final project we aim to build a controller that uses Machine Learning techniques to control an uncertain system. The structure of the controller is of the multi-controller kind and the benchmark example of the system is a multi-input multi-output plant with uncertainties.

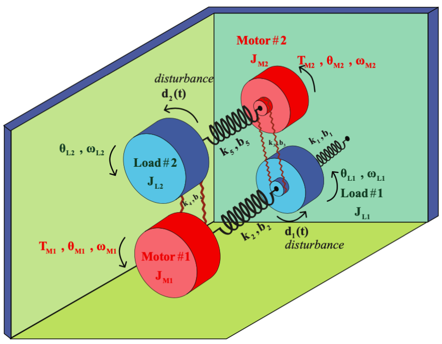

Fig. 1. The system motors and loads with elastic transmission

The continuous state space representation of the system, with process noise $\boldsymbol{w} = [w_1\ w_2]^\top$ and measurement noise $\boldsymbol{v} = [v_1\ v_2]^\top$, is given as

$$\begin{array}{l} \dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t) + \mathbf{G}\boldsymbol{w}(t) \\ \boldsymbol{y}(t) = \mathbf{C}\boldsymbol{x}(t) + \boldsymbol{v}(t) \end{array} . \tag{1}$$

The system matrices are given in (2) and (3), and the state vector is given in (3).

$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{J_{M1}} & 0 \\ 0 & \frac{1}{J_{M1}} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{G} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}, \tag{2}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The disturbances are given as

$$\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} \frac{0.2}{s+0.2} & 0 \\ 0 & \frac{0.2}{s+0.2} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}.$$

Both the process and measurement noise are modeled as independent zero mean white noise. The covariance matrices $\mathbf{Q}$ and $\mathbf{R}$, corresponding to the process and measurement noise, respectively, are given below.

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 11 \end{bmatrix}, \qquad \mathbf{R} = \begin{bmatrix} 10^{-6} & 0 \\ 0 & 10^{-6} \end{bmatrix}.$$

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2+k_3}{J_{L1}} & \frac{k_2}{J_{L1}} & \frac{k_3}{J_{L1}} & 0 & -\frac{b_1+b_2+b_3}{J_{L1}} & \frac{b_2}{J_{L1}} & \frac{b_3}{J_{L1}} & 0 & \frac{1}{J_{L1}} & 0 \\ \frac{k_2}{J_{M1}} & -\frac{k_2+k_4}{J_{M1}} & 0 & \frac{k_4}{J_{M1}} & \frac{b_2}{J_{M1}} & -\frac{b_2+b_4}{J_{M1}} & 0 & \frac{b_4}{J_{M1}} & 0 & 0 \\ \frac{k_3}{J_{M2}} & 0 & -\frac{k_3+k_5}{J_{M2}} & \frac{k_5}{J_{M2}} & \frac{b_3}{J_{M2}} & 0 & -\frac{b_3+b_5}{J_{M2}} & \frac{b_5}{J_{M2}} & 0 & 0 \\ 0 & \frac{k_4}{J_{L2}} & \frac{k_5}{J_{L2}} & -\frac{k_4+k_5}{J_{L2}} & 0 & \frac{b_4}{J_{L2}} & \frac{b_5}{J_{L2}} & -\frac{b_4+b_5}{J_{L2}} & 0 & \frac{1}{J_{L2}} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.2 \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} \theta_{L1} \\ \theta_{M1} \\ \theta_{M2} \\ \theta_{L2} \\ \omega_{L1} \\ \omega_{M1} \\ \omega_{M2} \\ \omega_{L2} \\ d_1 \\ d_2 \end{bmatrix} \tag{3}$$

1. PART A - KALMAN FILTER DESIGN

In the first part of the project, a discrete Kalman filter was to be designed. To do this, the system in (1) must be discretized. By using the MATLAB function `c2d`, one can obtain the discrete-time system matrices $\mathbf{A_d}$ and $\mathbf{B_d}$ with the command `[A_d, B_d] = c2d(A, B, Ts)`, where `Ts` is the sampling time. `c2d` discretizes the system using the zero-order hold approximation. This is acceptable with regards to the input $\boldsymbol{u}$ since it is provided by a controller operating with the same sampling time `Ts`. Hence we can take $\boldsymbol{u}$ to be constant over each timestep. However, due to the continuous nature of the process noise, we cannot apply the same procedure to discretize $\mathbf{G}$. Instead, van Loan's method (van Loan, 1978) is used as follows

$$\mathbf{M} = \begin{bmatrix} -\mathbf{A} & \mathbf{G}\mathbf{Q}\mathbf{G}^T \\ \mathbf{0} & \mathbf{A}^T \end{bmatrix},$$
$$\mathbf{N} = \text{expm}(\mathbf{M}) = \begin{bmatrix} \ldots & \mathbf{A}_d^{-1}\mathbf{Q}_d \\ \mathbf{0} & \mathbf{A}_d^T \end{bmatrix}.$$

Solving this, the discrete-time process noise covariance matrix $\mathbf{Q_d}$ can be obtained by extracting $\mathbf{Q_d}$ from $\mathbf{N}$. The discrete-time system becomes

$$\begin{aligned} \boldsymbol{x}[k+1] &= \mathbf{A_d}x[k] + \mathbf{B_d}\boldsymbol{u}[k] + \boldsymbol{w}_d[k] \\ \boldsymbol{y}[k] &= \mathbf{C}\boldsymbol{x}[k] + \boldsymbol{v}[k] \end{aligned}.$$

Note that the $\mathbf{G}$-matrix is replaced by the identity matrix in the discrete-time state-space system. The information in the continuous time $\mathbf{G}$-matrix is now contained within the covariance matrix, which is now 10×10. Correspondingly, $\boldsymbol{w}_d$ is 10×1. For the measurement noise, the covariance matrix is $\mathbf{R_d} = \mathbf{R} \cdot$`Ts`.

Table 1. Equations of the Kalman filter. Adapted from Table 11.1 in (Fossen, 2011).

Initial state:	$\bar{\boldsymbol{x}}[0] = \boldsymbol{x}_0$
Initial covariance error:	$\bar{\mathbf{P}}[0] = \mathbf{P_0}$
Kalman gain:	$\mathbf{K}[k] = \bar{\mathbf{P}}[k]\mathbf{C}^\top[k] \cdot \left(\mathbf{C}[k]\bar{\mathbf{P}}[k]\mathbf{C}[k]^\top + \mathbf{R}_d[k]\right)^{-1}$
State estimate update:	$\hat{\boldsymbol{x}}[k] = \bar{\boldsymbol{x}}[k] + \mathbf{K}[k]\left(\boldsymbol{y}[k] - \mathbf{C}[k]\bar{\boldsymbol{x}}[k]\right)$
Error covariance update:	$\hat{\mathbf{P}}[k] = \mathbf{K}[k]\mathbf{R}_d[k]\mathbf{K}^\top[k] + \left(\mathbf{I} - \mathbf{K}[k]\mathbf{C}[k]\right)\bar{\mathbf{P}}[k]\left(\mathbf{I} - \mathbf{K}[k]\mathbf{C}[k]\right)^\top$
State estimate propagation:	$\bar{\boldsymbol{x}}[k+1] = \mathbf{A}_d\hat{\boldsymbol{x}}[k] + \mathbf{B}_d\boldsymbol{u}[k]$
Error covariance propagation:	$\bar{\mathbf{P}}[k+1] = \mathbf{A}_d\hat{\mathbf{P}}[k]\mathbf{A}_d^\top + \mathbf{Q}_d$

Having found the discrete-time system matrices and noise covariances, implementation of the Kalman filter is straightforward. In Simulink we create a subsystem with a MATLAB function block that takes as input the system and covariance matrices, as well as the noisy measurement $\boldsymbol{y}$ and the control $\boldsymbol{u}$. In the function block, the equations defining the Kalman filter (see table 1) are used to calculate an estimate of $\boldsymbol{y}$. The function block has a sampling time of `Ts`. Fig. 2 shows this subsystem.

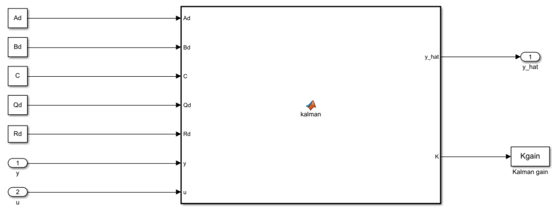

Fig. 2. Kalman filter implemented as a Simulink subsystem with MATLAB function block.

A plot of the measured and real y is provided in Fig. 3. Performance of the Kalman filter described above is indicated in Fig. 4. From this figure we note that the filter performs exceedingly well; only by magnifying the plot significantly can any deviation from the true value be clearly discerned.

An alternative approach to the design of the Kalman filter is to use the `kalmd` function in MATLAB. This function takes the continous-time state-space model, along with $\mathbf{Q}$ and $\mathbf{R}$ as input arguments. It returns a discrete state-space

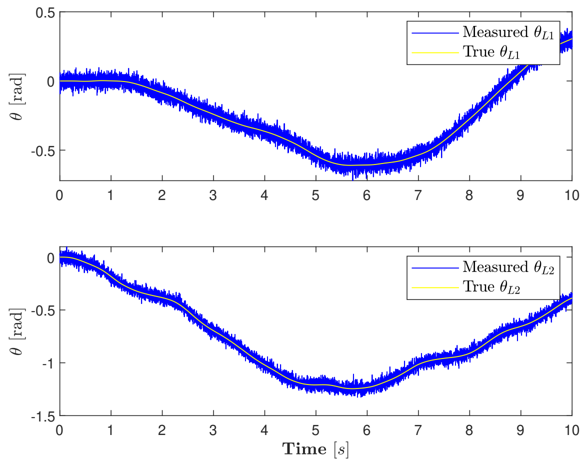

Fig. 3. Plot of measured quantities and true values.

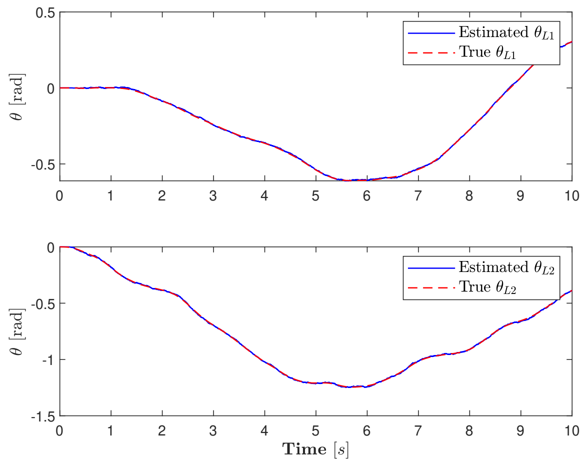

Fig. 4. Plot of true quantities and estimated quantities from the Kalman filter.

model of the Kalman filter and the steady-state Kalman gain matrix $\mathbf{K}_\infty$.

Using the step-by-step approach as described in table 1, leads to a time-varying gain $\mathbf{K}$ that converges to $\mathbf{K}_\infty$. To illustrate this convergence, we present a plot of the 2-norm of both $\mathbf{K}$ and $\mathbf{K}_\infty$ against time, seen in Fig. 5. From this we see that $\mathbf{K}$ converges in less than 1.5 seconds. Hence, one can use $\mathbf{K}_\infty$ for all time-steps in the Kalman filter when simulating. This means that fewer calculations need to be done at each iteration. In the remainder of the project, we use the built-in `kalmd` function in MATLAB for the Kalman filter. That is, we use the steady-state Kalman gain found from this function in our implementation of the filter. Additionally, `kalmd` returns the steady-state covariance matrix $\mathbf{P}$, which we use later in the dynamic hypothesis testing.

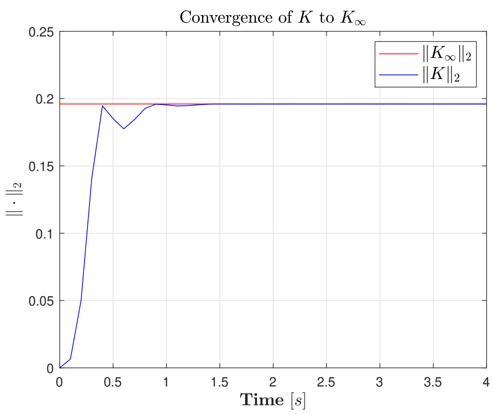

Fig. 5. Convergence of $\mathbf{K}$ to the steady-state gain matrix $\mathbf{K}_\infty$, as indicated by the time evolution of $||\mathbf{K}||_2$.

2. PART B - DESIGNING DYNAMIC HYPOTHESIS TESTING

For this part of the project we are to design a Dynamic Hypothesis Testing (DHT) in order to determine the true value of the parameters k_2 and k_5, based on three different hypotheses. In Fig. 6 we see three different subsets for the two parameters. The three hypotheses, which we denote by $\mathcal{H}_1$, $\mathcal{H}_2$ and $\mathcal{H}_3$, included in this test are:

- $\mathcal{H}_1$: The uncertain parameters lie in subset 1. Assume $k_2 = 2$ and $k_5 = 2$.
- $\mathcal{H}_2$: The uncertain parameters lie in subset 2. Assume $k_2 = 1$ and $k_5 = 1.75$.
- $\mathcal{H}_2$: The uncertain parameters lie in subset 3. Assume $k_2 = 2$ and $k_5 = 1.25$.

For each of the hypotheses, the discrete matrices and the corresponding steady state Kalman gains are calculated using the `kalmd` function in MATLAB. This is used to construct three distinct Kalman filters; one for each hypothesis. The estimated output for each sampling time, $\hat{y}[k]$, is then calculated using the three different Kalman filters. For each time step, we look at the conditional probability of each hypothesis based on the observation vector $Z(t)$,

$$h_i(t) = \Pr\{\mathcal{H} = \mathcal{H}_i | Z(t)\}.$$

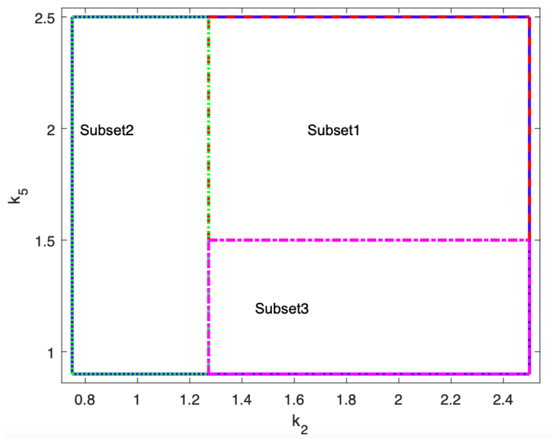

Fig. 6. The partition of the parametric uncertainty sets.

Using Bayes rule, the conditional probability theorem and the total probability theorem, we get that the probability $\Pr\{\mathcal{H} = \mathcal{H}_i | Z(t+1)\}$ can be written as

$$h_i(t+1) = \frac{\Pr\{y(t+1)|\mathcal{H} = \mathcal{H}_i, Z(t)\}}{\sum_{k=1}^{N} \Pr\{y(t+1)|Z(t), \mathcal{H} = \mathcal{H}_k\} h_k(t)} h_i(t).$$

The denominator turns into a sum since we have a limited number of hypotheses. Since all the noises and disturbances are Gaussian, we can simplify this into

$$h_i(t+1) = \frac{\frac{e^{-\frac{1}{2}\tilde{y}^T_{\mathcal{H}}(t+1)S^{-1}_{\mathcal{H}_i}\tilde{y}_{\mathcal{H}_i}(t+1)}}{\sqrt{(2\pi)^2|S_{\mathcal{H}_i}|}}}{\sum_{k=1}^{N} h_k(t)\frac{e^{-\frac{1}{2}\tilde{y}^T_{\mathcal{H}_k}(t+1)S^{-1}_{\mathcal{H}_k}\tilde{y}_{\mathcal{H}_k}(t+1)}}{\sqrt{(2\pi)^2|S_{\mathcal{H}_k}|}}} h_i(t). \quad (4)$$

The covariance matrix S_{H_k} of y_{H_k} is calculated as in (5), and the symbols presented in (4) and (5) are explained in table 2.

$$S_{\mathcal{H}_k} = CPC^T + \Theta_d \quad (5)$$

Table 2. Symbol explanation

$\tilde{y}_{\mathcal{H}_k}(t+1)$:	Output error (residual) from the k^{th} hypothesis
$\left\|S_{\mathcal{H}_k}\right\|$:	Determinant of the covariance matrix of $y_{\mathcal{H}_k}$
P:	Steady-state covariance matrix
Θ_d:	Discrete equivalent of continous measurement noise covariance matrix

Another more practical consideration to the implementation of DHT is the avoidance of probability values very close to zero, as this will cause the hypothesis in question to become frozen and unresponsive to changes in the system parameters. This is handled by letting $\Pr(\mathcal{H}_i) = \varepsilon$ whenever $\Pr(\mathcal{H}_i) < \varepsilon$, and then normalizing the probability vector. In our testing, we used $\varepsilon = 5 \cdot 10^{-5}$. To do the hypothesis testing, the parameters k_2 and k_5 were varied in such a way that hypothesis 1 through 3 was true for 100 seconds each in the simulation. The values of k_2 and k_5 are presented in table 3.

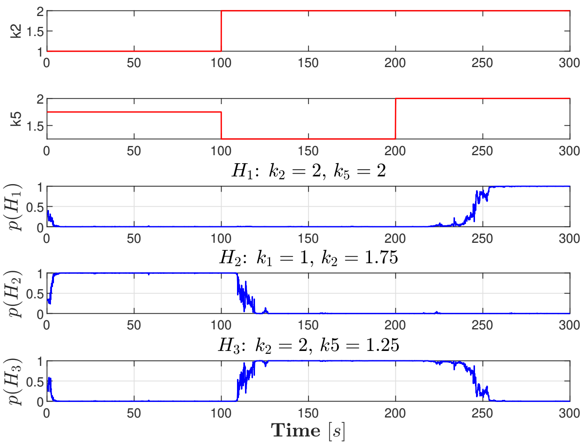

Fig. 7. Dynamic hypothesis testing with time-varying k_2 and k_5, and feedback from true y (uncorrupted by measurement noise).

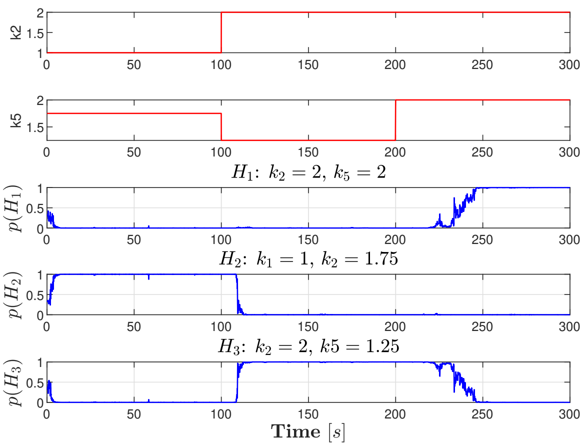

Fig. 8. Dynamic hypothesis testing with time-varying k_2 and k_5, and feedback from the best estimate of y.

Table 3. Parameter values

Time interval [s]	k_2	k_5
$[0 - 100)$	1	1.75
$[100 - 200)$	2	1
$[200 - 300]$	2	2

Furthermore, the feedback to the controller was taken from the true $\boldsymbol{y}$. Results are seen in Fig. 7. We see that in all cases, there is a convergence to the true hypothesis. However, we note that there is a longer time to reach convergence as the simulation proceeds. Initially, the hypothesis testing finds the true hypothesis in only a few seconds, but gets increasingly slower. For the last change in signal, approximately 50 seconds passes before the true hypothesis is clearly identified. We attempt to explain this in the following. Initially, all probabilities have equal value, but only one hypothesis is correct for both k_2 and k_5. For that reason, we would expect a rapid identification of the true hypothesis. For the first change in parameters at $t = 100$s, both k_2 and k_5 changes. Now, both $\mathcal{H}_1$ and $\mathcal{H}_3$ have the correct value for k_2, while the previously true hypothesis is wrong for both parameters. Hence, it is reasonable that the time to convergence will take a bit longer now, because the true hypothesis has a very low probability that it must recover from. For the second change at $t = 200$s, only k_5 changes. Since the true hypothesis $\mathcal{H}_1$ has a very low probability at this instant, and it is competing with the similar $\mathcal{H}_3$ that currently has a probability of approximately 1, it is expected that convergence will take even longer time here.

Since we do not have access to the real measurement in practical applications, another simulation was run where we at each iteration use feedback from the "best estimate" - the estimate from the Kalman filter whose associated hypothesis has the highest probability of being true. This case is presented in Fig. 8. Here, we see a similar performance to the previous case, but convergence is slightly faster and more direct.

Another comment on the time-varying nature of the parameters, and hence the system, used in the simulation is warranted. Strictly speaking, the mathematical theory underlying DHT is only valid for time-invariant systems. However, the times at which the parameters change are chosen such that the system rate of change is much slower than the convergence rate of the probabilities. In other words, the system can be regarded as time-invariant within the time interval to convergence. So, when we change the parameters at e.g $t = 100$s, it would be similar to starting a simulation for a time-invariant system with the appropriate parameters, with $\Pr(\mathcal{H}_1)$ and $\Pr(\mathcal{H}_3)$ approximately equal to 0, and $\Pr(\mathcal{H}_2)$ approximately 1.

3. CONCLUSION

The systems implemented have proven to work in a satisfactory manner. The Kalman filter performs very well at estimating the true y from a noisy measurement. Additionally, the gain of the time varying Kalman filter have been shown to converge to the steady state Kalman gain. In the dynamic hypothesis testing the true hypothesis is found, but the convergence is a little slow when changing between different hypotheses. This is considered to be due to similarity in the parameter values of the hypotheses, combined with a very low probability of the true hypothesis when making the switch in parameters.

REFERENCES

Fossen, T.I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley Sons, Ltd, Chichester, UK.

van Loan, C. (1978). Computing integrals involving the matrix exponential. *IEEE Transactions on Automatic Control*, 23(3), 395–404. doi: 10.1109/TAC.1978.1101743.