SoftwareX 34 (2026) 102666

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

StoLyaBMI: Software to compute Lyapunov functions for linear SDEs with constant coefficients

Eyleifur Bjarkason*, Sigurdur Hafstein, Ingvar Thoroddsson

Science Institute, University of Iceland, Dunhagi 3, 107 Reykjavík, Iceland

ARTICLE INFO

Keywords:
Stochastic differential equations
Lyapunov function
Global asymptotic stability in probability

ABSTRACT

We present an algorithm implemented in MATLAB, that formulates the problem of computing Lyapunov functions for a linear stochastic differential equations (SDEs) with constant coefficients as a bilinear matrix inequality (BMI) problem. By solving such BMIs our software generates Lyapunov functions that assert the global asymptotic stability in probability (GASiP) of the origin for the system at hand. Our software enables, for the first time, the verification of GASiP for systems where exponential mean-square stability fails.

Metadata

Code metadata: General information about this code

Current code version	v1
Permanent link to code/repository used for this code version	https://github.com/Eyleifur/StoLyaBMI
Legal Code License	none
Code versioning system used	none
Software code languages, tools, and services used	MATLAB
Compilation requirements, operating environments & dependencies	CVX: Matlab Software for Disciplined Convex Programming Symbolic Math Toolbox
Link to developer documentation/manual	none
Support email for questions	eyleifur@gmail.com

1. Motivation and significance

A linear deterministic system $\dot{\mathbf{x}} = A\mathbf{x}$, $A \in \mathbb{R}^{n\times n}$, has a globally asymptotically stable (GAS) equilibrium, if and only if A is Hurwitz, that is, the real-parts of all eigenvalues of A are strictly negative [14]. In this case, one can compute a quadratic Lyapunov function $V(\mathbf{x}) = \mathbf{x}^T Q\mathbf{x}$ by solving a particular type of the Sylvester equation [21], the so-called continuous-time Lyapunov equation,

$$Q^T A + AQ = -P,$$

for any given symmetric and positive definite $P \in \mathbb{R}^{n\times n}$. In the spirit of linear matrix inequalities (LMIs) this is sometimes written:

$$\text{compute } Q \succ 0, \text{ such that } Q^T A + AQ \prec 0,$$

where $A \succ B$ ($A \succeq B$) means that $A - B$ is a symmetric and positive (semi-)definite matrix.

The stability theory of stochastic differential equations (SDEs) is considerably more involved; see e.g., [5,15,18], and even the linear, constant coefficients case is quite difficult. There have been some recent approaches to compute Lyapunov functions for SDEs; see e.g., [3,

* Corresponding author.
Email addresses: eyleifur+softwarex@gmail.com; eib15+softwarex@hi.is (E. Bjarkason).

https://doi.org/10.1016/j.softx.2026.102666
Received 29 January 2026; Received in revised form 7 April 2026; Accepted 16 April 2026

6,8,10], and just as in the deterministic case, it is often advantageous to compute a local Lyapunov function for the linearised system around the equilibrium in question, together with a nonlocal Lyapunov function further away from the equilibrium [4,11]. This makes the study of linear SDEs with constant coefficients particularly important. We will only sketch the theory here. For a full discussion see [1], where our software was used to compute Lyapunov functions for numerous different SDEs.

Our software deals with SDEs of the form

$$\mathrm{d}\boldsymbol{X}(t) = F\boldsymbol{X}(t)\mathrm{d}t + \sum_{u=1}^{U} G^u \boldsymbol{X}(t)\mathrm{d}W_u(t), \tag{1}$$

where $\boldsymbol{X}(t)$ is a stochastic process that solves the SDE interpreted in the Itô sense, $U \in \mathbb{N}_+ := \{1,2,\dots\}$, $F, G^u \in \mathbb{R}^{n\times n}$, and the W_u are independent and identically distributed one-dimensional Wiener processes.

The stability concept for SDEs that corresponds to GAS for deterministic systems, is global asymptotic stability in probability (GASiP) [15], defined as $\mathbb{P}\left\{\lim_{t\to\infty} \boldsymbol{X}(t) = \boldsymbol{0}\right\} = 1$ for every initial-value $\boldsymbol{X}(0) \in \mathbb{R}^n$, and for every $r > 0$ and $0 < \varepsilon < 1$ there exists a $\delta > 0$ such that $\|\boldsymbol{X}(0)\| < \delta$ implies $\mathbb{P}\left\{\sup_{t\geq 0} \|\boldsymbol{X}(t)\| \leq r\right\} \geq 1-\varepsilon$. Here $\mathbb{P}\{\cdot\}$ denotes the probability measure of the set $\{\cdot\}$ and $\|\cdot\|$ denotes the Euclidean norm of a vector.

The aim of our software is to compute a Lyapunov function of the form

$$V(\boldsymbol{x}) = \|\boldsymbol{x}\|_Q^p := \left(\sqrt{\boldsymbol{x}^T Q \boldsymbol{x}}\right)^p = (\boldsymbol{x}^T Q \boldsymbol{x})^{p/2}, \tag{2}$$

where $Q \succ 0$ and $p > 0$. The origin is GASiP for the system (1), if and only if there exists a Lyapunov function $V \in C(\mathbb{R}^n) \cap C^2(\mathbb{R}^n \setminus \{\boldsymbol{0}\})$, i.e., continuous and two-times continuously differentiable at every $\boldsymbol{x} \neq \boldsymbol{0}$, which fulfills the conditions of Theorem 3.2 in [12]. For any $p > 0$ a function of the form (2) automatically fulfills all the conditions of the theorem, except for possibly the condition

$$LV(\boldsymbol{x}) \leq -\widetilde{c}\|\boldsymbol{x}\|_Q^p \tag{3}$$

for all $\boldsymbol{x} \in \mathbb{R}^n \setminus \{\boldsymbol{0}\}$, where $\widetilde{c} > 0$ is an arbitrary constant and $LV(\boldsymbol{x})$ is the generator of the system (1). The generator can be written as

$$LV(\boldsymbol{x}) = -\frac{p}{2}\|\boldsymbol{x}\|_Q^{p-4} H(\boldsymbol{x}), \tag{4}$$

where

$$H(\boldsymbol{x}) = -\boldsymbol{x}^T \left(F^T Q + QF + \sum_{u=1}^{U} (G^u)^T Q G^u\right)\boldsymbol{x}\|\boldsymbol{x}\|_Q^2 + \frac{2-p}{4}\sum_{u=1}^{U}\left(\boldsymbol{x}^T (QG^u + (G^u)^T Q)\boldsymbol{x}\right)^2. \tag{5}$$

Using (4) and (5), it is easy to see that the condition (3) is equivalent to

$$P_c(\boldsymbol{x}) := H(\boldsymbol{x}) - c\|\boldsymbol{x}\|^4 \geq 0 \tag{6}$$

for some $c > 0$ and that P_c is a homogeneous polynomial of order four. The objective of our algorithm is therefore, for a given $p > 0$, to find a matrix $Q \succ 0$ and a constant $c > 0$ such that the polynomial $P_c(\boldsymbol{x})$ is non-negative for all $\boldsymbol{x} \in \mathbb{R}^n$. If this is possible, then $V(\boldsymbol{x}) = \|\boldsymbol{x}\|_Q^p$ is a Lyapunov function for the system. To compute such $Q \succ 0$ and $c > 0$ we attempt to write P_c as the sum of squared polynomials. This is the celebrated SOS relaxation that is often used when computing Lyapunov functions for deterministic systems, see e.g., [7,19,20].

Let $\boldsymbol{z}$ be a vector whose elements are all the different second order monomials in $\boldsymbol{x}$, e.g.,

$$\boldsymbol{z} = \begin{pmatrix} x_1^2 & x_1x_2 & \cdots & x_1x_n & x_2^2 & x_2x_3 & \cdots & x_n^2 \end{pmatrix}^T,$$

where $\boldsymbol{z}$ has dimension $m = \frac{n(n+1)}{2}$ by basic combinatorics. If there exists a symmetric, positive semi-definite matrix $P \in \mathbb{R}^{m\times m}$ such that for all $\boldsymbol{x} \in \mathbb{R}^n$ we have

$$P_c(\boldsymbol{x}) = \boldsymbol{z}^T P \boldsymbol{z} =: P_{\boldsymbol{z}}(\boldsymbol{x}), \tag{7}$$

then $P_c(\boldsymbol{x})$ is the sum of squared polynomials and therefore positive for all $\boldsymbol{x} \in \mathbb{R}^n$.

To achieve the equality $P_c(\boldsymbol{x}) = P_{\boldsymbol{z}}(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^n$, our software equates all fourth-order partial derivatives of P_c and $P_{\boldsymbol{z}}$. This gives us a relationship between each of the corresponding coefficients of the two polynomials. This delivers the following set of conditions, that have to be fulfilled in order for $Q \in \mathbb{R}^{n\times n}$ to be a solution to our original problem:

$$c > 0, \quad Q \succ 0, \quad P \succeq 0,$$
$$\frac{\partial^4 P_c}{\partial x_{i_1}\partial x_{i_2}\partial x_{i_3}\partial x_{i_4}} = \frac{\partial^4 P_{\boldsymbol{z}}}{\partial x_{i_1}\partial x_{i_2}\partial x_{i_3}\partial x_{i_4}}, \quad i_1, i_2, i_3, i_4 \in \{1,2,\dots,n\}.$$

Our algorithm encodes all of these constraints into a single bilinear matrix inequality (BMI) feasibility problem

$$M(\mathbf{q}) := \sum_{i,j=1}^{K} q_i q_j A_{ij} + \sum_{i=1}^{K} q_i B_i + C \succeq 0, \tag{8}$$

with variables $\mathbf{q} \in \mathbb{R}^K$ and where $A_{ij}, B_i, C \in \mathbb{R}^{N\times N}$ are symmetric matrices, $i, j = 1,2,\dots,K$. For discussion on how to fix the constants $K, N \in \mathbb{N}_+$ we refer to [1]. The BMI can then be solved using a number of methods—our software uses the heuristics from [16,17].

In two important cases the BMI (8) simplifies to an LMI, which can be algorithmically solved in polynomial time. The first case is when one verifies the Lyapunov function condition (6) for a given fixed symmetric matrix $Q \in \mathbb{R}^{n\times n}$ in (2), this was studied in [13]. The other case is when $p = 2$, because then the second term in the formula (5) for $H(\boldsymbol{x})$ vanishes and the Lyapunov function condition (3) simplifies to

$$\boldsymbol{x}^T \left(F^T Q + QF + \sum_{u=1}^{U} (G^u)^T Q G^u\right)\boldsymbol{x} \leq -\widetilde{c}\|\boldsymbol{x}\|_Q^2,$$

which is equivalent to

$$F^T Q + QF + \sum_{u=1}^{U} (G^u)^T Q G^u \prec 0 \tag{9}$$

because $Q \succ 0$ and $\widetilde{c} > 0$ is arbitrary. The $p = 2$ case is how the Lyapunov theory for SDEs is usually applied in the literature, but it has serious drawbacks. First, for the corresponding deterministic system to (1), i.e., the G^us are all zero, the origin must be GAS, for otherwise (9) cannot hold true. Indeed, the term $\sum_{u=1}^{U} (G^u)^T Q G^u$ is positive semi-definite. Second, if (9) is fulfilled, then the origin is exponentially mean-square stable, which is a much stricter stability concept than GASiP. As shown in [1] for numerous examples, very often one does not recognise stability of the zero solution when this concept is used. In particular, the concept of mean-square exponential stability is not able to recognize stabilization by noise. However, our software has functions dedicated to both these important LMI cases.

2. Algorithms

Our software algorithmically constructs the BMI problem (8) for the SDE (1). For a given dimension n and a given number U of Wiener processes in (1), a general BMI problem is constructed using symbolic variables, of which the particular matrices F and G^u, $u = 1,2,\dots,U$, are parameters. This general BMI problem is stored in a *.mat* file. To write and solve the BMI problem for a particular SDE (1), the matrices F and G^u and the constant $p > 0$ are supplied to this general problem. The algorithm is quite involved, but is described in detail in [1,2] with worked out examples. The heuristic used to solve the resulting BMI problem in our software is described in [16,17]. A thorough investigation of which heuristics are most suited to solve our BMI problems has not been conducted, but is a matter of future research.

3. Software description

Our software is implemented in MATLAB and requires the CVX package, which is a MATLAB-based modeling system for convex optimization [9]. The semi-definite solver we use is SDPT3 [22], which is included with CVX. Additionally, the Symbolic Math Toolbox in MATLAB is needed for our software to run. We have tested our software with the versions R2021b and R2024b of MATLAB.

3.1. Software architecture

The code consists of several functions, of which five are intended to be called by the applicant:

- `function SaveBMIMatrices(n, numGs, NameFile)` writes, using symbolic variables, the general BMI problem (8) for the SDE (1) for a given dimension `n` and a given number of Wiener processes `numGs` $=U$. Optionally, one can supply a name for the file where the general BMI problem is saved; the default is *BMI system.mat*. As an example, `SaveBMIMatrices(2,3)` writes the file *3G 2x2 BMI system.mat*, which contains the general BMI problem for (1) with $n = 2$ and $U = 3$. The function that actually constructs the BMI matrices is `BMIconstruction`.
- `function [retQ, retC, retP, retEq] = BMISolution(F, G, p0, eps, maxIter, NumberTrials, eta, seed, Q_init)` generates the BMI problem for a particular system (1) by substituting the symbolic variables in the *.mat* file generated by `SaveBMIMatrices` with the values `F,G,p0` and then attempts to solve it using the heuristic from [16,17]. The inputs are the matrix `F` and the matrices G^u, stored in a 3-dimensional array `G` such that `G(:,:,u)` is G^u, `u=1:numGs`. Additionally, `p0` must be delivered, which is the $p > 0$ in (2). From (5) and (6) it is obvious, that it is easier to obtain a solution for smaller $p > 0$.
Additionally, one can deliver some optional arguments: `eps` is used to model $c > 0$ and $Q \succ 0$ through $c \geq \varepsilon$ and $Q \succeq \varepsilon I$ with ε =`eps`. The default value for `eps` is 0.01. `maxIter`, `NumberTrials`, `eta`, and `seed` are parameters used to control the heuristic from [16,17], which starts with a random vector and then iterates in an attempt to construct a feasible matrix Q. `maxIter` fixes the maximum number of iterations in each trial, where the heuristics tries to compute a feasible Q and `NumberTrials` fixes how many trials are attempted before the heuristic gives up. The default values are `maxIter`= 100, and `NumberTrials`= 10. `eta` is an internal parameter for the heuristic with a recommended value between 50 and 100; the default value is `eta`= 50. For reproducibility `seed`, which initialises the random generator, has the default value 1. Finally, one can supply an initial matrix Q_{init} for Q, which is then used instead of a random one. This can be very useful if one already has a solution Q for a slightly different system.
The outputs of the function are `retQ`, which is the computed matrix Q in the formula (2) for the Lyapunov function and is usually the only output of interest. For analysing the solution one might also want to have a look at `retC`, which is the value of c in (6) and is usually slightly larger than ε =`eps`, the matrix `retP`, which is the matrix P in (7) and should be positive semi-definite, and the vector `retEq`, which contains the equality relations in the BMI (8), explained in detail in [1,2], and should be very close to the zero vector.
- `function retQ = LMISolutionQ(F, G, p0, eps, Q_guess)` is similar to `BMISolution` above, but with the difference that the matrix Q in the BMI problem (8) is supplied in `Q_guess`. Verifying whether the BMI can be fulfilled with this choice of Q is an LMI problem as discussed at the end of Section 1.
- `function retQ = LMISolutionP2(F, G, p0, eps)` is similar to `BMISolution` above, but with $p = 2$ so the BMI problem (8) becomes an LMI problem as discussed at the end of Section 1. Here `eps`=: ε, in addition to modelling $Q \succ 0$ through $Q \succeq \varepsilon I$, is also used to model $\prec 0$ in (9) through $\preceq -\varepsilon I$.
- `function VerifyH(F, G, Q, p, NrChecks, BreakOnFail)` is a simple Monte Carlo test routine that checks the condition $H(\boldsymbol{x}) \geq 0$, with $H(\boldsymbol{x})$ defined in (5), for `NrChecks` random vectors.

3.2. Software functionalities

Our software delivers a Lyapunov function for linear SDEs with constant coefficients, of which the origin is GASiP. The existence of such a Lyapunov function, inter alia, asserts the GASiP property of the origin.

4. Illustrative example

We show how to use our software for the concrete example of the damped harmonic oscillator with white noise in the damping. This system was studied in detail in [15, Ex. 6.6] and [13, Section 5.1]. The ODE for the damped harmonic oscillator is given by

$$\ddot{x} + k\dot{x} + \omega^2 x = 0.$$

By setting $x_1 = \omega x$ and $x_2 = \dot{x}$ we get the linear system of equations

$$\dot{\boldsymbol{x}} = F\boldsymbol{x}, \quad \text{where} \quad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad \text{and} \quad F = \begin{pmatrix} 0 & \omega \\ -\omega & -k \end{pmatrix}.$$

By adding white noise to the damping we get the SDE

$$\mathrm{d}\boldsymbol{X}(t) = F\boldsymbol{X}(t)\mathrm{d}t + G\boldsymbol{X}(t)\mathrm{d}W(t), \quad \boldsymbol{X} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \quad \text{and} \quad G = \begin{pmatrix} 0 & 0 \\ 0 & -\sigma \end{pmatrix},$$

where W is a one-dimensional Wiener-process and $\sigma \geq 0$ quantifies the noise.

In Listing 1 we have set up the system in Matlab and do some illustrative experiments. This example is included in the software in the file `ExampleSoftwareX.m`. Additionally, all the examples computed in [1] are included in the software.

In [13] it is shown that the origin is exponentially mean-square stable, i.e., $p = 2$, if and only if $k > \sigma^2/2$. Hence, if we set $\omega = 2.75$, $k = 2.5$, and $\sigma = 2$, we obtain a solution from `LMISolutionP2(F,G,eps)`, where we use `eps`= 0.1 (line 9). However, if we change k to 2 we do not get a solution (line 12). In [13] it is shown that even for $k = 0.9$, one can set $Q_{\text{guess}} = \begin{pmatrix} 3 & 1/3 \\ 1/3 & 3 \end{pmatrix}$ and the LMI resulting from the BMI (8) by plugging in the values from Q_{guess} for Q, has a solution with $p = 0.1$. This is done in lines 14–16. In line 17 we show how to perform a Monte Carlo test with 100,000 random vectors to check whether $H(\boldsymbol{x}) \geq 0$ when using these parameters; $H(\boldsymbol{x})$ is the function in (5). Finally, if ω is changed to 2.5, then $V(\boldsymbol{x}) = \|\boldsymbol{x}\|_{Q_{\text{quess}}}^{0.1}$ is not a Lyapunov function for the system

```
1  n = 2; numGs = 1; p = 0.1; eps = 0.01;
2  w = 2.75; k = 2.5; s = 2;
3
4  F = [0, w; -w, -k];
5  G = [0, 0; 0, -s];
6
7  Q = LMISolutionP2(F, G,  eps); pause
8
9  k = 2; F = [0, w; -w, -k];
10 Q = LMISolutionP2(F, G,  eps); pause
11
12 k = 0.9; F = [0, w; -w, -k];
13 Q_guess = [3, 1/3; 1/3, 3];
14 Q = LMISolutionQ(F, G, p, eps, Q_guess); pause
15 VerifyH(F, G, Q, p, 100000); pause
16
17 w = 2.5; F = [0, w; -w, -k];
18 Q = LMISolutionQ(F, G, p, eps, Q_guess); pause
19
20 Q = BMISolution(F, G, p, eps);
```

Listing 1. Illustrative example.

[13]. However, if we use the full algorithm `BMISolution(F,G,p,eps)` we obtain a matrix $Q \succ 0$ such that $V(\boldsymbol{x}) = \|\boldsymbol{x}\|_Q^{0.1}$ is a Lyapunov function for the system that assures GASiP of the origin. This is done in line 22.

5. How to use

First make sure that CVX and Symbolic Math Toolbox are installed. Then study the illustrative startup example in Listing 1 and/or the additional examples included with the software to see how to use it in practice to compute Lyapunov functions.

6. Impact

Our software computes Lyapunov functions for linear SDEs with constant coefficients and, hence, delivers a certificate for GASiP of the origin for the SDE. In the literature the concept of exponential mean-square stability is almost always used, because algorithms to certify weaker, but more sensible types of stability, have not been available. Our software changes this and the stability of the origin for many more SDEs of the form (1) can be asserted. This is shown in [1] for numerous different examples.

7. Conclusions

We have published MATLAB functions that construct a bilinear matrix inequality (BMI) problem for a stochastic differential equation (SDE) of the form (1), i.e., linear and with constant coefficients. A feasible solution to the BMI problem with parameter $p > 0$ delivers a symmetric, positive definite matrix Q such that $V(\boldsymbol{x}) = \|\boldsymbol{x}\|_Q^p$ is a Lyapunov function for the system. Such a Lyapunov function asserts the global asymptotic stability in probability (GASiP) of the origin. For $p = 2$ the BMI problem reduces to a linear matrix inequality (LMI) and a feasible solution asserts exponential mean-square stability of the origin, which is a much stricter property than GASiP. Hence, our software enables the assertion of stability for many more systems than previously possible, in particular for systems stabilized by noise. It remains to be studied which heuristics are most adequate to solve the BMI problems generated.

CRediT authorship contribution statement

Eyleifur Bjarkason: Writing – review & editing, Software. **Sigurdur Hafstein:** Writing – review & editing, Writing – original draft, Software. **Ingvar Thoroddsson:** Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Sigurdur Hafstein reports that financial support was provided by The Icelandic Centre for Research. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was partially supported by the Icelandic Research Fund in grant *Lyapunov Methods and Stochastic Stability* number 152429–051.

Data availability

The software to the manuscript *StoLyaBMI: Software to compute Lyapunov functions for linear SDEs with constant coefficients* can be accessed at https://github.com/Eyleifur/StoLyaBMI.

References

[1] Barrera G, Bjarkason E, Hafstein S. The stability of the multivariate geometric brownian motion as a bilinear matrix inequality problem. SIAM J Appl Dyn Syst 2025;24(2):1251–88.
[2] Bjarkason E. Lyapunov functions for linear stochastic differential equations by bilinear matrix inequalities–theory and implementation [M.Sc. thesis]. University of Iceland; 2022.
[3] Björnsson H. Lyapunov functions for stochastic systems: theory and numerics [Ph.D. thesis]. University of Iceland; 2023.
[4] Bjornsson H, Giesl P, Gudmundsson S, Hafstein S. Local lyapunov functions for nonlinear stochastic differential equations by linearization. In: Proceedings of the 15th international conference on informatics in control, Automation and Robotics (ICINCO); 2018. p. 579–86.
[5] Bjornsson H, Hafstein S, Dynamical systems in theoretical perspective, volume 248 of springer proceedings in Mathematics and statistics. In: Chapter lyapunov functions for almost sure exponential stability. Springer; 2018. p. 51–61.
[6] Bjornsson H, Hafstein S, Giesl P, Scalas E, Gudmundsson S. Computation of the stochastic basin of attraction by rigorous construction of a lyapunov function. Discrete Contin. Dyn. Syst Ser. B 2019;24(8):4247–69.
[7] Chesi G. Domain of attraction: analysis and control via SOS programming. Lecture Notes in Control and Information Sciences, Vol. 415, Springer; 2011.
[8] Giesl P, Hafstein S, Pokkakkillath S. Piecewise quadratic lyapunov functions for stochastic differential equations by linear programming. Discrete Contin. Dyn. Syst. Ser. B 2025;30(6):2027–50.
[9] Grant M, Boyd S. CVX: matlab software for disciplined convex programming, version 2.1 Mar, 2014. http://cvxr.com/cvx.
[10] Grüne L, Camilli F. Characterizing attraction probabilities via the stochastic zubov equation. Discrete Contin. Dyn. Syst. Ser. B 2003;3(3):457–68.
[11] Gudmundsson S, Hafstein S. Probabilistic basin of attraction and its estimation using two lyapunov functions. Complexity 2018:2895658.
[12] Hafstein S. Lyapunov functions for linear stochastic differential equations: bmi formulation of the conditions. In: Proceedings of the 16th international conference on informatics in control, Automation and Robotics (ICINCO); 2019. p. 147–55.
[13] Hafstein S, Gudmundsson S, Giesl P, Scalas E. Lyapunov function computation for autonomous linear stochastic differential equations using sum-of-squares programming. Discrete Contin. Dyn. Syst Ser. B 2018;23(2): 939–56.
[14] Khalil H. Nonlinear systems. 3. ed. Pearson; 2002.
[15] Khasminskii R. Stochastic stability of differential equations. 2nd ed. Springer; 2012.
[16] Kheirandishfard M, Zohrizadeh F, Adil M, Madani R. Convex relaxation of bilinear matrix inequalities part II: applications to optimal control synthesis. In: Proceedings of 57rd IEEE conference on decision and control (CDC); 2018. p. 75–82.
[17] Kheirandishfard M, Zohrizadeh F, Madani R. Convex relaxation of bilinear matrix inequalities Part I: Theoretical Results. In: Proceedings of 57rd IEEE conference on decision and control (CDC); 2018. p. 67–74.
[18] Mao X. Stochastic differential equations and applications. 2nd ed. Woodhead Publishing; 2008.
[19] Parrilo P. Structured semidefinite programs and semialgebraic geometry methods in robustness and optimiziation [Ph.D. thesis]. California Institute of Technology Pasadena, California; 2000.
[20] Peet M, Papachristodoulou A. A converse sum of squares lyapunov result with a degree bound. IEEE Trans Autom Control 2012;57(9):2281–93.
[21] Sylvester J. Sur l'equation EN matrices px = xq. C. R. Acad. Sci. Paris. 1884;99(2):115–6.
[22] Toh KC, Todd MJ, Tütüncü RH. Sdpt3 - a matlab software package for semidefinite programming, version 1.3. Optim Methods Softw 1999;11(1–4): 545–81.