Safe Metropolis-Hastings Algorithm and Its Application to Swarm Control

Mahmoud El Chamie\textsuperscript{a}, Behçet Açıkmese\textsuperscript{b}

\textsuperscript{a}United Technologies Research Center (UTRC), Systems Department, East Hartford, CT 06108, USA, email: elchamm@utrc.utc.com
\textsuperscript{b}University of Washington, William E. Boeing Department of Aeronautics and Astronautics, Seattle, WA 98195, USA, email: behcet@uw.edu

Abstract

This paper presents a new method to synthesize safe reversible Markov chains via extending the classical Metropolis-Hastings (M-H) algorithm. The classical M-H algorithm does not impose safety upper bound constraints on the probability vector, discrete probability density function, that evolves with the resulting Markov chain. This paper presents a new M-H algorithm for Markov chain synthesis that ensures such safety constraints together with reversibility and convergence to a desired stationary (steady-state) distribution. Specifically, we provide a convex synthesis method that incorporates the safety constraints via designing the proposal matrix for the M-H algorithm. It is shown that the M-H algorithm with this proposal matrix, safe M-H algorithm, ensures safety for a well-characterized convex set of stationary probability distributions, i.e., it is robustly safe with respect to this set of stationary distributions. The size of the safe set is then incorporated in the design problem to further enhance the robustness of the synthesized M-H proposal matrix. Numerical simulations are provided to demonstrate that multi-agent systems, swarms, can utilize the safe M-H algorithm to control the swarm density distribution. The controlled swarm density tracks time-varying desired distributions, while satisfying the safety constraints. Numerical simulations suggest that there is insignificant trade-off between the speed of convergence and the robustness.

Keywords: Markov chains; Metropolis-Hastings; Safety Constraints; Convex Optimization; Swarm Control

1. Introduction

The Metropolis-Hastings (M-H) algorithm [2, 3, 4] is a method for obtaining random samples from a probability distribution. The M-H algorithm builds on the theory of Markov processes and Markov-Chain-Monte-Carlo (MCMC) sampling methods [5, 6, 7, 8] to synthesize reversible Markov chains that guarantee the desired stationary distributions. Recent research has focused on synthesizing fast mixing Markov chains with desired stationary distributions that incorporate constraints on transition probabilities [9, 10] by using tools from graph theory [11], Lyapunov stability analysis [12], and convex optimization [13].

The M-H algorithm is very useful when online Markov chain synthesis is needed because it can be implemented easily and executed very efficiently. Given a matrix \( K \) (called proposal matrix), the M-H algorithm can be used to construct a stochastic transition matrix \( M \) of a Markov chain to satisfy some specifications, e.g., a prescribed stationary distribution and constraints on transitions. The matrix \( M \) inherits the key properties of the proposal matrix \( K \).
such as speed of convergence, while satisfying the prescribed specifications. Thus the choice of matrix $K$ is critical for the performance of the M-H algorithm [14], in order to speed up the warm up phase, i.e., the transient regime. However, currently, the M-H algorithm cannot impose hard constraints on the probability distribution vector during the warm up phase, such as upper bound constraints on the probability distribution. These hard constraints on the probability vector of the Markov chain are often referred to as safety constraints [15, 16].

Safety constraints are critical in applications where the violations during transients can cause a failure of the system. The primary focus of this paper is on swarm control [17, 18, 19], where overcrowded regions can increase the risk of collisions in space. Having safe transients is also critical for systems where an exogenous process can push the system out of the stationary regime. Some examples of exogenous processes in swarm control are: addition or removal of agents in and out of the swarm and disaster incidents that cause a group of agents in a region to fail. These incidents push the system out of the stationary regime into a new transient regime, so a safe convergence back to the stationary distribution is necessary and can be provided by the safe M-H algorithm proposed in this paper.

In [17, 16], we synthesize Markov matrices with safety constraints via convex optimization methods. The approach in [17, 16] requires solving an LMI optimization for the construction of these matrices and it is suited for constant offline synthesis of Markov matrices. On the other hand, in this paper, we obtain a computationally efficient M-H algorithm to synthesize a time-varying Markov matrix. This allows the system to quickly adapt to time-varying desired distribution specifications without recomputing the proposal matrix, which is a very useful property for the swarm control application where this adaptation must happen in real-time. In addition, we present a robust version, which handles a larger set of stationary distributions, and a fast version, which optimizes the rate of convergence of the Markov chain.

In summary, the main contributions of this paper are: (i) Incorporating safety into the M-H algorithm for a set of stationary probability distributions; (ii) Providing a new linear-programming-based method to synthesize the proposal matrix of the safe M-H algorithm with some robustness properties; (iii) Studying the speed of convergence of the resulting transition matrix of the robust and safe M-H algorithm; (iv) Applying the robust and safe M-H algorithm to a swarm density control problem, with time-varying desired density specifications.

2. Problem Formulation

The probability distribution over the states of a Markov chain can be expressed as a probability vector $x(t) \in \mathbb{R}^m$ with the relation

$$x(t + 1) = M(t)x(t) \quad t = 0, 1, 2, \ldots$$

where $M(t)$ is a column stochastic matrix for all time, i.e., $1^T M(t) = 1^T$ and $M(t) \geq 0$, with $\geq$ being the component-wise inequality. In many applications, it is desired to design the column stochastic matrix $M(t)$ to satisfy some specifications. For example, in swarm control, also referred to as Randomized Motion Planning (RMP) [17, 20], $x(t)$ describes the probability of an agent (e.g., vehicle) to be in a given region and $M(t)$ determines the probability distributions for possible transitions between these regions. In later sections, we will apply our theoretical findings on the RMP problem. In gossiping and wireless sensor networks [21], Eq. (1) describes the dynamics for the evolution of an estimate of a relevant physical quantity as temperature, pressure, etc. In voting models [22], $x(t)$ determines the preference of a group of people towards a given object of interest (e.g., application, leader, etc.). In consensus protocols, the transition matrix (also called the weight matrix) is designed for the fastest convergence of the consensus among a group of networked agents [23, 24].

The probability vector $x(t)$ characterizes the behavior of the underlying Markov chain, governed by Eq. (1), both during the transient (warm-up) and steady-state phases. During the warm-up phase, the M-H algorithm samples from $x(t)$, which is biased by the initial distribution $x(0)$,\(^1\) where $x(t)$

\(^1\)The M-H algorithm eventually samples from the limiting distribution.
satisfies some constraints naturally due to the column stochasticity of $M(t)$, such as $x(t) \geq 0$ and $1^T x(t) = 1$ for all $t = 0, 1,...$. There can also be additional constraints characterized by hard safety upper bounds on the probability vector, i.e.,

$$x(t) \leq d \text{ for all } t \geq 0,$$

where $d \in \mathbb{R}^m_+$ is a constant non-negative vector. The classical M-H algorithm does not impose the above safety constraints (2) on the probability vector of the resulting Markov chain. This paper presents a new M-H algorithm for Markov chain synthesis method that handles safety constraints while ensuring other specifications such as reversibility, desired stationary (steady-state) distribution, and transitional constraints. The synthesis of the proposed M-H algorithm is based on a numerically tractable linear programming formulation.

3. Formulation of Reversible Markov Chain Synthesis with Safety Constraints

3.1. Notation

In this paper, small bold letters are used for vectors (e.g., $x$ whose elements are indicated as $x_1, x_2, \ldots$), and capital letters are used for matrices (e.g., $X$ whose $i$-th row $j$-th column element is denoted by $X_{ij}$). A graph is denoted by $G = (V, E)$ where $V = \{1, \ldots, m\}$ is the set of vertices and $E \subseteq V \times V$ is the set of edges. We use the pair $(j, i)$ to denote the edge from vertex $j$ to vertex $i$. We assume that $(i, i) \in E$ for all $i \in V$, i.e., $G$ contains all the self loops. The adjacency matrix $A$ of $G = (V, E)$ is: $A_{ij} = 1$ if $(j, i) \in E$ and $A_{ij} = 0$ otherwise. A summary of the notation is given by Table 1. We will consider the following assumption on the graph $G$:

**Assumption 1.** $G$ is undirected and connected.

In an undirected graph, $(j, i) \in E$ if and only if $(i, j) \in E$. A graph is connected if for any pair of vertices $i$ and $j$, there is a path from $i$ to $j$, i.e., a sequence of edges $(i, i_1), (i_1, i_2), \ldots, (i_p, j)$ starting from vertex $i$ and ending at vertex $j$. With Assumption 1, the adjacency matrix $A$ is symmetric and irreducible.

3.2. Markov Chain Specifications

We consider a Markov chain describing the evolution of a discrete probability vector $x(t) \in \mathbb{R}^m$ given by (1), where $M(t)$ is a column stochastic matrix for all time $t$. Motivated by Markov chain terminology, $M(t)$ is referred to as the transition matrix where $M_{ij}(t)$ is the probability of transition from a state $j$ to state $i$ at time $t$.

We first consider the case with a constant transition matrix $M(t) = M$ for all $t$, and then discuss the time-varying case after introducing the M-H algorithm for the time-invariant desired distributions. Specifically, our objective is to synthesize a transition matrix $M$ such that the resulting Markov chain (1) has the following properties:

1. The desired probability distribution $v \in \mathbb{R}^m_+$ is the stationary distribution: 
   $$\lim_{t \to \infty} x(t) = v, \forall x(0) \in \mathbb{R}^m_+.$$
2. Reversibility: $v_i M_{ji} = v_j M_{ij}$, for all $i, j = 1, \ldots, m$.
3. Transition constraints: $M_{ij} = 0$ when $(i, j) \notin E$, and $M_{ij} > 0$ when $(i, j) \in E$ (the set of feasible transitions).
4. Safety constraints: $x(t) \leq d$ for all $t \geq 0$, and for a given vector $0 \leq d \leq 1$.

The transition constraints are described by an adjacency matrix characterizing the set of feasible state transitions. The safety constraints bound the probability distribution during both the transients and at the steady-state.

3.3. Convex Formulations of the Specifications

This section summarizes our results on convex formulations of the above specifications for a time-invariant Markov matrix $M$. These convex representations are equivalent to the original specifications and they facilitate the formulation of Linear Matrix Inequality (LMI) problems for the synthesis of reversible Markov chains for given steady-state distributions.
Transition, Stochasticity, and Reversibility Constraints. First of all, the transition constraints can be expressed using the adjacency matrix \( A \) of the graph \( G \). The set \( C_G := \{ X \in \mathbb{R}^{m \times m} : X_{ij} = 0 \text{ if } (i, j) \notin \mathcal{E} \text{ and } X_{ij} > 0 \text{ if } (i, j) \notin \mathcal{E} \} \) represents all feasible transition matrices, and hence

\[
M \in C_G \iff (11^T - A) \odot M = 0 \quad \exists \epsilon > 0 \text{ s.t. } M \geq \epsilon A ,
\tag{3}
\]

which imposes linear constraints on \( M \), where \( \odot \) is the component-wise, Hadamard, product. In implementation, \( \epsilon \) can be chosen to be a sufficiently small positive scalar.

The Markov matrix must satisfy the column stochasticity constraints described as

\[
1^T M = 1^T ,
\tag{4}
\]

which is also a linear equality constraint on \( M \).

Reversible chains have favorable properties (e.g., detailed balance condition) and are commonly utilized in applications of Markov Chain Monte Carlo (MCMC) methods, e.g., in birth death processes, M/M/1 queues, and symmetric random walks on graphs [25]. A Markov matrix \( M \) is reversible with a stationary distribution \( \mathbf{v} \) if and only if

\[
M \text{diag}(\mathbf{v}) = \text{diag}(\mathbf{v}) M^T .
\tag{5}
\]

The above equation implies that \( \mathbf{v} \) is a steady-state distribution for the resulting Markov chain, which can be obtained by simply multiplying both sides of the equality by \( \mathbf{1} \), \( M\mathbf{v} = \mathbf{v} \). The set of all admissible Markov matrices for a given steady-state distribution, \( \mathbf{v} \), and adjacency matrix, \( A \), is:

\[
\mathcal{M}_G(\mathbf{v}) := \{ \text{Matrices satisfying (3), (4), and (5)} \}.
\tag{6}
\]

The results in this paper hold (with minor modifications) when \( \mathbf{v} \) has some zero entries. Without loss of generality, we assume in what follows that \( \mathbf{v} > 0 \).

We can parameterize all reversible matrices having \( \mathbf{v} \) as the steady-state distribution. Let \( \Delta \) be an \( m \times m \) matrix defined by the stationary distribution \( \mathbf{v} \) as follows:

\[
\Delta_{ij}(\mathbf{v}) := \begin{cases} v_i/v_j & \text{if } i < j \\ 1 & \text{else} \end{cases}
\tag{7}
\]

Next we give a useful parameterization of all reversible Markov matrices for a given \( \mathbf{v} \).

Lemma 1. \( M \in \mathcal{M}_G(\mathbf{v}) \) if and only if there exists an \( m \times m \) matrix \( Y \) such that

\[
M = \Delta(\mathbf{v}) \odot Y + I - \text{diag}(1^T(\Delta(\mathbf{v}) \odot Y)) ,
\tag{8}
\]

\[
M \geq \delta I , \quad Y = Y^T , \quad Y \in C_G ,
\]

for some scalar \( \delta > 0 \).

Proof. Let \( Y \) be a matrix such that (8) is satisfied, then

\[
1^T M = 1^T(\Delta(\mathbf{v}) \odot Y) + 1^T - 1^T(\Delta(\mathbf{v}) \odot Y) = 1^T.
\]

Table 1: Summary of the notation.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( G = (V, \mathcal{E}) )</td>
<td>Undirected and connected graph</td>
</tr>
<tr>
<td>( V = {1, \ldots, m} )</td>
<td>Set of vertices</td>
</tr>
<tr>
<td>( A \subseteq V \times V )</td>
<td>Proposal matrix of the M-H algorithm</td>
</tr>
<tr>
<td>( \mathcal{C}_G )</td>
<td>Set of matrices ( \mathcal{C}<em>G := { X \in \mathbb{R}^{m \times m} : X</em>{ij} = 0 \text{ if } (i, j) \notin \mathcal{E} \text{ and } X_{ij} &gt; 0 \text{ if } (i, j) \notin \mathcal{E} } )</td>
</tr>
<tr>
<td>( \text{diag}(\mathbf{x}) )</td>
<td>Adjacency matrix of ( G ): ( A_{ij} = 1 ) if ((j, i) \notin \mathcal{E} ) and ( A_{ij} = 0 ) otherwise</td>
</tr>
<tr>
<td>( K )</td>
<td>Diagonal matrix whose diagonal entries are ( x_1, x_2, \ldots ) of vector ( \mathbf{x} )</td>
</tr>
<tr>
<td>( M )</td>
<td>Stationary distribution</td>
</tr>
<tr>
<td>( \odot )</td>
<td>Component-wise, Hadamard, product</td>
</tr>
<tr>
<td>( 0, 1, \text{ and } I )</td>
<td>Vector/matrix of all zeros, vector of all ones, and the identity matrix respectively</td>
</tr>
<tr>
<td>( X \leq Y )</td>
<td>Set of probability vectors: ( x \in \mathbb{P}^m ) if ( x \geq 0 ) and ( 1^T x = 1 )</td>
</tr>
<tr>
<td>( X \preceq Y )</td>
<td>Set of column stochastic matrices: ( M \in \mathbb{P}^m ) if ( M \geq 0 ) and ( 1^T M = 1^T )</td>
</tr>
</tbody>
</table>

Table 1: Summary of the notation.
Since $Y \in \mathcal{C}_G$, $Y \geq \beta A$ for some $\beta > 0$ and $v_i/v_j > 0$ for all $i, j$, and hence there exists some $\kappa > 0$ such that $M_{ij} \geq \kappa$ for all $i \neq j$ if $(i, j) \in \mathcal{E}$, and also $M_{ij} = 0$ for all $i \neq j$ if $(i, j) \notin \mathcal{E}$. Since $M \geq \delta I$, we have $M \geq \epsilon A$ with $\epsilon = \min(\delta, \kappa)$, and hence (4) and (3) are satisfied. For $i < j$, we have $M_{ij} = \Delta_{ij}(v)Y_{ij} = (v_i/v_j)Y_{ij}$, and for $i > j$, we have $M_{ij} = \Delta_{ij}(v)Y_{ji} = Y_{ij}$. Then, using the fact that $Y$ is a symmetric matrix, we get $M_{ij} = (v_i/v_j)Y_{ij} = (v_i/v_j)M_{ji}$, which implies that (5) is satisfied. Therefore $M \in \mathcal{M}_G(v)$.

To show the other direction of the lemma, now suppose that $M \in \mathcal{M}_G(v)$, it is sufficient to construct $Y$ from $M$ such that $Y$ satisfies (8). It is indeed the case by constructing $Y$ as follows: $Y_{ij} = M_{ij}$ if $i > j$, $Y_{ij} = Y_{ji}$ if $i < j$, $Y_{ii} = A_{ii}$. Hence $Y = Y^T$, and it is straightforward to verify that the off-diagonal terms of $M_{ij}$ satisfy the first equation in (8). Since $1^TM = 1^T$ in (8), the diagonal terms of $M$ also satisfy the equation. This completes the proof.

Lemma 1 shows that any matrix $M$ that generates a reversible chain can be parameterized by a set of linear constraints.

Convergence to Steady-State Distribution. The following well-known result (see for example [26]) shows that asymptotic convergence to $v$ is ensured by matrix $M$ when the spectral radius condition is satisfied,

$$\rho(M - v1^T) < 1.$$  \hfill (9)

**Lemma 2.** Consider a column stochastic matrix, $M$, such that $Mv = v$, and the dynamical system (1). Then, $\lim_{t \to \infty} x(t) = v$ holds for any initial probability vector $x(0)$ if and only if (9) is satisfied.

The following lemma characterizes the convergence of the system using conditions on the graph $G$,

**Lemma 3.** If $\mathcal{G} = (V, \mathcal{E})$ is undirected and connected (Assumption 1), then $\rho(M - v1^T) < 1$ for all $M \in \mathcal{M}_G(v)$.

**Proof.** When $\mathcal{G}$ is connected, then for any pair of vertices $i$ and $j$, we can find a path $i u_1 u_2 \ldots u_{i-j}$, and thus $(M^t)_{ij} \geq M_{i u_1} M_{u_1 u_2} \ldots M_{u_{i-j}} > 0$ (the last inequality is due to (3)). Since $l \leq m$, then $M$ is irreducible as for every pair $i, j$ of its index set, there exists a positive integer $l \equiv l(i, j)$ such that $(M^l)_{ij} > 0$ [27, p. 18]. Moreover, since self loops are included in $G$, then there exists $i$ such that $M_{ii} > 0$ and thus $M$ is primitive having a unique stationary distribution $v$, so $\lim_{t \to \infty}(M - v1^T)^t = 0$ and the lemma follows.

Note that $\mathcal{M}_G(v)$ is a convex set because (3), (4), and (5) are linear (in-)equalities, which facilitates the convex synthesis.

**Safety.** We consider the following safety constraints [15, 16]

$$x(t) \leq d, \quad t = 1, 2, \ldots, \quad \text{if } x(0) \leq d.$$ \hfill (10)

which require the probability vector (density distribution) to be bounded by the vector $d$ for all time instances. These constraints can be useful in swarm control for collision/conflict mitigation by limiting the densities, hence eliminating overcrowding during transitions. The following theorem provides convex conditions (linear (in-)equalities) on the matrix $M$ for the safety property to be satisfied.

**Theorem 1.** Consider the Markov chain described by (1) with a constant $M \in \mathbb{R}^{m \times m}$ matrix. Given a scalar $\gamma \in [0, 1]$, for any $x(0) \leq d$, the following holds

$$x(t) \leq (1 - \gamma)d \quad \forall t = 1, 2, \ldots$$

if and only if there exists a variable $S \in \mathbb{R}^{m \times m}$ such that:

$$S \geq 0, \quad M \leq S - Sd1^T + (1 - \gamma)d1^T.$$ \hfill (11)

**Proof.** An equivalent set of linear inequalities was proved in [16] for $\gamma = 0$. When $\gamma \in [0, 1]$, we will give here a new, simpler, proof for the sufficiency part of the theorem. For the necessary part, the proof given in [16, Theorem 1] using duality theory of linear programming can be refined in a straightforward way.

Suppose there exist $M$ and $S$ such that (11) is satisfied. Then we need to show that for all vectors
where the first inequality uses (11) and the fact that $\mathbf{x} \geq 0$, the equality in the second line uses the fact that $\mathbf{x}$ is a probability vector and must sum to one, and the last inequality is due to the fact that $S \geq 0$ and $\mathbf{x} - \mathbf{d} \leq 0$. Then by mathematical induction, for any $\mathbf{x}(0) \leq \mathbf{d}$ we have $\mathbf{x}(t) \leq (1 - \gamma)\mathbf{d} \ \forall t = 1, 2, ...$ because $\mathbf{x}(t+1) = M \mathbf{x}(t)$.

4. Safe Metropolis-Hastings (M-H) Algorithm

This section presents a new M-H synthesis method for the $M(t)$ matrix in (1) as a function of a time-varying desired probability distribution $\mathbf{v}(t)$. If the desired distribution does not change with time, then the algorithm will generate a constant Markov matrix. However, if the desired distribution $\mathbf{v}$ is time-varying, the Markov matrix will be time-varying as well. Our main result adapts the well-known M-H algorithm to handle the safety constraints for a time-varying stationary distribution. In particular, we show that the M-H algorithm can be applied for a well-characterized set of stationary distributions by properly designing the proposal matrix to ensure convergent and safe Markov chains. Beyond having a new M-H result to generate safe Markov chains, the resulting algorithm presents a computationally inexpensive method, which makes it easily adaptable for online computations once a proper proposal matrix is computed offline via using convex optimization techniques.

4.1. M-H Algorithm

The M-H algorithm [28, 3] given by Algorithm 1 is a Markov Chain Monte Carlo (MCMC) method for obtaining a sequence of random samples from a given probability distribution.

Note that if $K_{ij} = 0$, then $M_{ij} = M_{ji} = 0$. Similarly, $K_{ij} > 0$ and $K_{ji} > 0$ imply that $M_{ij} > 0$ and $M_{ji} > 0$. Consequently, when $\mathbf{v} > 0$ and the matrix $F$ is chosen by (13), we can impose transition constraints, given by (3), on the proposal matrix, $K$, to guarantee that $M$ also satisfies the transition constraints.

Lemma 4. Consider the M-H algorithm, for some $\mathbf{v} > 0$. If the proposal matrix $K$ satisfies $K \in \mathcal{M}_G(\mathbf{u})$ for some probability vector $\mathbf{u}$, then the resulting Markov matrix given by (12) satisfies $M \in \mathcal{M}_G(\mathbf{v})$.

Proof. The matrix $M$ is by the M-H algorithm's construction a column stochastic matrix satisfying the reversibility assumption. Since $K \in \mathcal{M}_G(\mathbf{u})$, then $K \in \mathcal{C}_G$, and thus $M \in \mathcal{C}_G$. As a result, $M \in \mathcal{M}_G(\mathbf{v})$ and this completes the proof.

The M-H can transform any proper proposal matrix into a Markov matrix with a desired stationary probability distribution $\mathbf{v}$. However incorporating safety constraints into the resulting transition matrix $M$ for a time-varying distribution $\mathbf{v}(t)$ is not straightforward. Therefore, it becomes natural to investigate the conditions under which the M-H algorithm can produce safe Markov chains. It turns out that if the proposal matrix for a nominal distribution $\hat{\mathbf{v}}$ is safe and has some robustness properties, then the M-H procedure can also be safe as long as $\mathbf{v}$ is in a well-characterized neighborhood of $\hat{\mathbf{v}}$ as discussed next.

Algorithm 1 M-H Algorithm

1: Input: Probability vector $\mathbf{v}$ and proposal matrix $K$.
2: The M-H algorithm produces an $M$ matrix given by:

\[
M_{ij} = \begin{cases} 
K_{ij}F_{ij} & \text{if } i \neq j \\
K_{jj} + \sum_{k \neq j}(1 - F_{kj})K_{kj} & \text{if } i = j
\end{cases}
\]

(12)

where $K$ is a proposal matrix that satisfies $K \geq 0$ and $1^T K = 1^T$; $\mathbf{v} > 0$; and $F$ is an acceptance matrix, which satisfies for $i \neq j$,

\[
F_{ij} = \min\left(1, \frac{\mathbf{v}_i K_{ij}}{\mathbf{v}_j K_{ji}}\right) \quad i, j = 1, \ldots, m. \quad (13)
\]


6
4.2. Robust Proposal for Safe M-H Algorithm

By construction, the M-H algorithm can ensure reversible Markov chains with the desired steady-state distribution and transition constraints. However, the safety specification is not necessarily guaranteed. In this section, we study the effect of the proposal matrix $K$ on the safety of the M-H algorithm. For this purpose, we introduce the following definitions.

**Definition 1 (S-Safe).** The M-H algorithm with a given proposal matrix $K$ is called S-Safe if the resulting matrix $M$ leads to Markov chains satisfying the safety constraints (10) for all steady-state distributions $v \in S$.

**Definition 2.** Given a matrix $K \geq 0$ and $\gamma \in [0,1], \mathcal{V}_\gamma(K) \subseteq \mathbb{P}^m$ is defined as follows: $\mathcal{V}_\gamma(K) = \{v \in \mathbb{P}^m : v$ satisfies (14)$\}$.

$$\sum_{k=1}^{m} \max\{0, v_iK_{ki} - v_kK_{ik}\} \leq \gamma v_i, \text{ for } i = 1, \ldots, m.$$  \hspace{1cm} \text{(14)}

Note that the set $\mathcal{V}_\gamma(K)$ is parameterized by $\gamma$, which can be considered as a robustness parameter as discussed next: the safe set for the M-H algorithm gets larger as $\gamma$ increases.

The following proposition gives some properties of the set $\mathcal{V}_\gamma(K)$.

**Proposition 1.** Consider a proposal matrix $K \in \mathcal{M}(\hat{\mathcal{P}})$ used in the M-H algorithm to generate a reversible Markov chain. Then, for any $\gamma \in [0,1]$: (i) $\mathcal{V}_\gamma(K)$ is a nonempty convex set; (ii) $V_0(K) = \{\hat{v}\}$; (iii) $V_1(K) = \mathbb{P}^m$; (iv) $\mathcal{V}_\gamma(K) \subseteq \mathcal{V}_{\gamma_1}(K)$ if $\gamma_1 \leq \gamma_2$.

**Proof.** $\mathcal{V}_\gamma(K)$ is nonempty because by considering $\hat{v}$, the stationary distribution of $K$, the following equation holds $\hat{v}_iK_{ki} = \hat{v}_kK_{ik}$ for all $i$ and $k$. Therefore the left hand side of (14) is zero and the equations are satisfied for any $\gamma \in [0,1]$. In particular, if $\gamma = 0$, then $\hat{v}$ is the unique value of $v$ that satisfies the inequalities and in this case $\mathcal{V}_\gamma$ is a singleton. If $\gamma = 1$, then $\mathcal{V}_\gamma$ is the set of all probability vectors ($\mathcal{V}_\gamma = \mathbb{P}^m$) since $\sum_{k=1}^{m} \max\{0, v_iK_{ki} - v_kK_{ik}\} \leq \sum_{k=1}^{m} v_iK_{ki} = v_i$ for any $v$.

$\mathcal{V}_\gamma$ is a convex set of probability vectors $v$ because of the following argument: given a matrix $K$, then $v_iK_{ki} - v_kK_{ik}$ is a convex function of $v$ (a linear function of $v$), $\max\{0, v_iK_{ki} - v_kK_{ik}\}$ is convex (the maximum of convex functions), $\sum_{k=1}^{m} \max\{0, v_iK_{ki} - v_kK_{ik}\} - \gamma v_i$ is convex (sum of convex functions), the set defined by $\sum_{k=1}^{m} \max\{0, v_iK_{ki} - v_kK_{ik}\} - \gamma v_i \leq 0$ is a convex set (sub-level set of a convex function), and finally $\mathcal{V}_\gamma$ is a convex set (intersection of convex sets). Moreover, $\mathcal{V}_\gamma$ defines a neighborhood around $\hat{v}$ because for any $\gamma > 0$, a small enough perturbation around $\hat{v}$ would satisfy equations (14).

The last relationship follows directly: If any $v$ satisfies (14) for $\gamma_1 \in [0,1]$, then it should also satisfy it for $\gamma_2 \geq \gamma_1$ since it further relaxes the inequality.

We can now give the main technical result of this paper.

**Theorem 2.** Suppose that there exist variables $Y \in \mathbb{R}^{m \times m}$, $S \in \mathbb{R}^{m \times m}$, and $\epsilon > 0$ such that the proposal matrix $K$ satisfies the following conditions

$$K = \Delta(\hat{v}) \odot Y + I - \text{diag}(1^T(\Delta(\hat{v}) \odot Y)), \quad K \geq \epsilon I, \quad Y = YT, \quad Y \in C$$

$$S \geq 0, \quad K \leq S - Sd1^T + (1 - \gamma)d1^T,$$

where $\Delta(\hat{v})$ is given by (7) and $\gamma \in [0,1]$. Then the M-H algorithm with the proposal matrix $K$ is $\mathcal{V}_\gamma$-Safe, that is, the resulting Markov chain is safe for all $v \in \mathcal{V}_\gamma$.

**Proof.** The proof is provided in the appendix.

4.3. Maximizing the Safe Set

Note that Theorem 2 results in a Markov matrix $K$ with reversible chains and the set $\mathcal{V}_\gamma$ is a convex set containing $\hat{v}$ (via Proposition 1). Fig. 1 illustrates the size of the set $\mathcal{V}_\gamma$ as a function of the robustness parameter $\gamma$.

2It demonstrates that, as $\gamma$ approaches 1, the size of the set becomes larger and approaches the simplex defining the probability distributions. This implies that maximizing $\gamma$ maximizes the size of the set $\mathcal{V}_\gamma$, which is the basis of the following LP-based synthesis for the proposal matrix $K$:

---

3The figure is generated using Monte Carlo simulations where random feasible distribution vectors are generated. The size of $\mathcal{V}_\gamma$ represents the percentage of simulation runs where the random generated vector $v$ satisfies (14).
4.4. Observations on Robustness and Convergence Speed

In optimization problem (16) for synthesis of \( K \), the speed of convergence to the stationary distribution \( \hat{\nu} \) is not constrained. Hence, though the convergence is guaranteed (due to Lemma 3), the speed at which the Markov chain converges can be arbitrarily slow. It is desirable to generate proposal matrices having both fast speed of convergence and good robustness properties. It is well-known that the speed of convergence of a Markov matrix is determined by the second largest eigenvalue’s magnitude \(|\lambda_2|\). Since \( K \) is searched over Markov matrices for reversible chains, the fastest converging proposal matrix can be computed via solving the following Semi-Definite Programming (SDP) problem for a prescribed robustness measure \( \gamma \):

\[
\begin{align*}
\text{minimize} & \quad \lambda \\
\text{subject to} & \quad -\lambda I \preceq Q^{-1}KQ - \mathbf{qq}^T \preceq \lambda I \\
& \text{Eq. (15)}
\end{align*}
\]

\( Eq. \ (17) \)

where \( \mathbf{q} = \hat{\nu}^{1/2} \) is the component-wise square root of \( \hat{\nu} \) and \( Q = \text{diag}(\mathbf{q}) \). The first constraint in (17) is a linear matrix inequality. Note that \( \gamma \) is not a solution variable in this formulation and it is prescribed. The above SDP has a feasible solution for \( [0, \gamma_{\text{max}}] \) where \( \gamma_{\text{max}} \) is computed via (16). Then a measure of convergence speed can also be described by the spectral gap \( s_g = 1 - \lambda^* \) where \( \lambda^* \) is the minimum value of \( \lambda \) computed by solving the SDP (17). Since any feasible solution from (15) for a given \( \gamma \) is also a feasible solution for \( \gamma \) where \( \gamma_2 \leq \gamma_1 \), we expect the convergence speed to be a non-increasing function of \( \gamma \) as \( \gamma \) varies from 0 to \( \gamma_{\text{max}} \). It is also observed that the speed of convergence has low sensitivity to the robustness parameter. In other words, we can have both fast convergence and robustness at the same time. Fig. 2 demonstrates this observation via a simulation example with 20 states, i.e., \( x(t) \in \mathbb{B}^{20} \). The parameters (the safety upper bound \( \mathbf{d} \) and the graph \( G \)) are selected at random initially, and then they are fixed throughout the simulation.\(^3\) Given these initial parameters, \( \gamma_{\text{max}} \) is deterministic and is computed via the linear program in Eq. (16). Each simulation point in the plot corresponds to solving the SDP in Eq. (17) for a fixed value of \( \gamma \in (0, \gamma_{\text{max}}) \). The figure shows that the spectral gap \( 1 - \lambda^* \) only decreases slightly as \( \gamma \) approaches \( \gamma_{\text{max}} \). Hence we can obtain both fast and robust Markov matrices.

\(^3\)We experimented with various initial conditions, and the results are similar to the ones shown in Fig. 2.

Figure 1: The size of the neighborhood as function of the parameter \( \gamma \). (One instance of a connected random network of 10 states, each element in the safety vector is chosen uniformly at random from \([0, 0.4, 1]\), nominal stationary distribution is chosen uniformly at random from distributions satisfying the bound.)
Figure 2: The spectral gap of the fastest proposal matrix has low sensitivity to the robustness parameter $\gamma$ (the spectral gap only decreased slightly as $\gamma$ approached $\gamma_{\text{max}}$). Therefore, we can have matrices that are both: robust and have fast convergence. (The connectivity network $G=(V,E)$ is constructed using a Random Geometric Graph (RGG) model with $n=20$ states and $m=55$ edges connecting the states, $\gamma_{\text{max}} = 0.4$, uniformly random safety bound where $d_i \in [0.4,1]$, and random nominal distribution $\mathbf{v}$. The parameters safety upper bound $d$ and the graph $G$ are selected at random initially, and then they are fixed throughout the simulation.)

5. Application of the Robust M-H for Safe Randomized Swarm Motion Planning

5.1. Background Summary

Swarm control objective is to synthesize desired motion commands for autonomous vehicles in the swarm to achieve prescribed mission objectives, which are described in terms of swarm density distribution over an operational domain. Randomized Motion Planning (RMP) generates the motion commands for vehicles probabilistically by specifying the probability distribution over the set of feasible actions as a function of the vehicle’s state. This section describes an RMP synthesis method for a swarm, statistically large number, of multi-vehicle system by using the safe M-H algorithm. The approach to RMP is to synthesize a Markov chain [26, 20] that governs the evolution of the swarm density such that the swarm converges to a desired probability distribution. Having a large number of agents is the motivation behind the probabilistic interpretation of the swarm distribution [17]. We also consider constraints for the RMP, such as the desired distribution, motion-transition and safety constraints (as in Section 3.2).

5.2. RMP Problem Formulation

This section summarizes the key components of the RMP formulation introduced in [26, 20]. The physical domain over which the agents are distributed is denoted by $R$. It is assumed that region $R$ is partitioned as the union of $m$ disjoint subregions $b_i$, $i = 1, \ldots, m$, (referred to as bins) such that $R = \bigcup_{i=1}^{m} b_i$, and $b_i \cap b_j = \emptyset$ for $i \neq j$. Let $r_k(t)$ be the position of an agent $k$ at a discrete time index $t$. Let $\mathbf{x}(t)$ be a probability vector such that $x_i(t)$ is the probability that an agent $k$ (chosen uniformly at random) is in bin $b_i$ at time $t$,

$$x_i(t) := \mathcal{P}(r_k(t) \in b_i). \quad (18)$$

Then the RMP problem is defined as follows: Given any initial probability distribution vector, $\mathbf{x}(0)$, the agents must be guided to a specified steady-state distribution, $\mathbf{v}$, that is:

$$\lim_{t \to \infty} x_i(t) = v_i \quad \text{for} \quad i = 1, \ldots, m, \quad (19)$$

subject to motion constraints (i.e., allowable transitions between bins), specified by an adjacency matrix, $A$, as:

$$A_{ij} = 0 \Rightarrow r_k(t+1) \notin b_i \quad \text{when} \quad r_k(t) \in b_j, \quad \forall t. \quad (20)$$

In an actual implementation, the subregions can be treated as either discrete points or point-sets. If a target region $b_j$ is a point-set, an additional randomization over $b_j$ can be applied by an agent to determine its target position in $b_j$.

Next, consider a matrix function of time, $M(t)$, that contains the transition probabilities between bins: For $t \in \mathbb{N}_+$ and $i, j = 1, \ldots, m$,

$$M_{ij}(t) := \mathbb{P}(r_k(t+1) \in b_i | r_k(t) \in b_j). \quad (21)$$

i.e., an agent $k$ in bin $j$ transitions to bin $i$ with probability $M_{ij}(t)$ at time $t$. The matrix valued function, $M$, determines the time evolution of the probability vector [26, 20], $\mathbf{x}(t)$, as

$$\mathbf{x}(t+1) = M(t)\mathbf{x}(t) \quad t = 0, 1, 2, \ldots \quad (22)$$
$M(t)$ is a column stochastic matrix, hence the probability vector $x(t)$ stays normalized as $1^T x(t) = 1$ for all $t \geq 0$.

Now the RMP problem becomes one of designing a specific Markov process (1) for $x$ that converges to a desired distribution, $\nu$. The distribution of agents, $x$, has the following interpretation: there are $m$ bins in the region, and $x_i(t)$ represents the probability of finding an agent in the $i$'th bin at time $t$. If there are $N$ agents, then $N x_i(t)$ describes the expected number of agents in the $i$'th bin. Let $n = [n_1(t), ..., n_m(t)]^T$ denote the actual number of agents in each bin. Then, $n_i(t)$ is generally different from $N x_i(t)$, although it follows from the independent and identically distributed agent realizations that

$$x(t) = \frac{E[n(t)]}{N},$$

where $E[.]$ is the expectation. From the law of large numbers [17], $n(t)/N \rightarrow x(t)$ as $N \rightarrow \infty$ for all $t$. The idea behind RMP is to control the propagation of the probability vector, $x$, rather than individual agent positions $\{r_k(t)\}_{k=1}^N$. While, in general, $n(t)/N \neq x(t)$, it will always be equal to $x$ on average, and can be made arbitrarily close to $x$ by using a sufficiently large number of agents.

### 5.3. The Robust Offline-Online Randomized Motion Planning Algorithm

In this section, we present the robust M-H based RMP algorithm used by agents. The RMP algorithm is implemented by computing offline an $M$ matrix for agents based on a nominal desired density specification $\hat{\nu}$. Then each agent propagates its position as an independent realization of the corresponding Markov chain.

In the first step of Algorithm 2, the agents receive the offline designed proposal matrix that guarantees the satisfaction of the motion specifications. In the second step each agent determines its current bin and online generates the Markov matrix based on the current desired density $\nu(t)$ using the M-H algorithm. Lines 7 and 8 in the algorithm describe how each agent samples from the discrete probability distribution given by the column of $M(t)$ that corresponds to the agent’s bin.

### Algorithm 2 Randomized Motion Planning Algorithm (RMPA) [20, 17]

1: **Input:** A robust $K$ matrix is designed offline by a centralized unit to satisfy constraints and tailored for a nominal desired density $\nu$ using the LP (16) and the SDP (17).
2: **Start of Online Procedure:**
3: $t = 1$
4: **while** True **do**
5: Each agent determines its current bin, e.g., $r(t) \in b_i$
6: Each agent computes $M(t)$ online using the M-H algorithm (Algorithm 1 introduced in Section 4.1) for the current desired density $\nu(t)$
7: Each agent generates a random number $z$ that is uniformly distributed in $[0, 1]$
8: The agent moves to bin $j$, $r(t + 1) \in b_j$, if $\sum_{k=1}^{t-1} M_{ki}(t) \leq z < \sum_{k=1}^{t} M_{ki}(t)$
9: $t \leftarrow t + 1$
10: **end while**

### 5.4. Numerical Simulations

The simulation example considers an operational area defined by a $3 \times 3$ grid, that is, the area is divided into 9 bins: $b_1, \ldots, b_9$ (i.e., the Markov chain has $m = 9$ states), see Fig. 3. Transitions are feasible between neighboring (having a common edge) bins. The graph $\mathcal{G}$ is then constructed by having each bin as a vertex in $\mathcal{G}$, and any two neighboring bins are connected by an edge. A swarm of $N = 2000$ agents is simulated, where each agent executes the M-H algorithm using Algorithm 2, with a pre-designed proposal matrix $K$. Each bin in the area has capacity limits, i.e., there are upper bound constraints on the number of agents that are allowed to be in a bin, see Fig. 3. We compare three different approaches used in this paper for designing the proposal matrix:

- Safe proposal matrix for the M-H algorithm: computed by solving a feasibility problem satisfying the linear equations (11) of Theorem 1.
- Robust and safe proposal matrix for the M-H algorithm: computed by solving the linear program (16) to guarantee safety within a neigh-
Figure 3: The randomized motion planning initial configuration. The 3 by 3 grid space is divided into nine bins. Within each bin we provide the number of agents in the initial distribution, the number of agents in the desired distribution, and the maximum number of agents allowed by the safety upper bound constraints.

- Fast, robust, and safe proposal matrix for the M-H algorithm: computed via the semi-definite program (17).

Three different cases are simulated. In the first case, a fixed desired distribution is specified and the proposal matrix is designed offline to guarantee safe convergence to this desired distribution. In the second case, after the convergence to the desired distribution of case 1, the desired distribution is changed and the M-H algorithm is used to generate a Markov matrix that guarantees convergence to the new distribution safely via the same algorithm, Algorithm 2. For the third case, we compare the violations of the safety bounds when the three approaches for designing the proposal matrix are used in a scenario with time-varying distributions, i.e., the desired distributions change after every 100 time steps.

In the first case, the classical M-H algorithm does not guarantee the safety upperbound constraints (during the transients) as illustrated by Fig. 4a. The proposal matrix $K$ in the classical M-H is designed by using the local degree transition matrix, i.e., $K = AD^{-1}$ where $A$ is the adjacency matrix, and $D$ is the diagonal matrix such that $D_{ii}$ is the degree of state $i$ in the graph $G$. The proposal matrix for the safe M-H is designed using an LP with the linear constraints given in (15). It is worth mentioning that both algorithms in Fig. 4a result in convergence to the steady-state distribution that is safe, then the violations due to the classical M-H occur during the transients, i.e., there is an iteration after which the classical algorithm would be safe. The safe M-H, on the other hand, guarantees that violations will not occur at all times, not even during the transients.

In the second case, after the convergence, the desired distribution is changed. The M-H algorithm (Algorithm 1) can adapt dynamically to such a change and it allows agents to tune online the transition probabilities to converge to the new desired stationary distribution. This procedure however can cause violations of the safety bound (because the transition matrix is changed) and Fig. 4b shows such violations for different proposal matrices. The figure shows that using the robustly safe proposal matrix reduces the number of violations (even though they exist because the new stationary distribution does not belong to the safe set $V_{\gamma}$) relative to a feasible safe proposal matrix. The additional optimization of the speed of convergence for the “fast” robust safe proposal matrix does not seem to impact the number of violations, while providing increased convergence speed, and the number of collisions is similar to that of the robust proposal matrix computed using the LP (16).

The third case has a time-varying desired distribution. The time-varying desired distribution here is not changed at every single iteration, but after sufficient time for the system to settle within a prescribed neighborhood of the current desired distribution. At every 100 times steps, a random desired distribution $\hat{v}$ is selected from the safe set satisfying the upper bound constraint. Thus the transition matrix in Algorithm 2 changes after every 100 iterations while the proposal matrix is designed offline and is always the same. Fig. 5 shows the cumulative violations (the total number of violations from $t = 0$ till the current iteration) due to the three different approaches for designing the proposal matrix. The figure shows that both, the LP and the SDP, designed proposal matrices provide less number of violations than just
The total number of violations at an iteration $k$ is given by the following formula:

$$
\sum_{i=1}^{n} \max(0, N_i(k) - N_{d_i})
$$

where $N_i(k)$ is the number of agents in state $i$ at iteration $k$, and $N_{d_i}$ is the upperbound constraint for the number of agents that are allowed to be in state $i$ shown in Fig. 3. The total number of agents is $N = 2000$. The statistics shown here are collected after running each algorithm a total of 2000 simulation runs, each simulation run is independent and corresponds to an agent in the system.

6. Conclusion

A safe M-H algorithm is introduced that takes into consideration safety upper-bound constraints. The proposal matrix of the M-H algorithm is designed offline using an LP formulation to maximize the set of robust desired stationary distributions that the M-H algorithm can handle safely. This LP formulation is further extended to an SDP to achieve fast convergence to the desired steady-state distributions. Our numerical results indicate that both the speed of convergence and the robustness can be achieved simultaneously. A further analytical investigation of this observation can be a useful as a next research step. An illustrative example of swarm density control is given as an application for the methods developed in this paper. This method guides the swarm to follow a time-varying prescribed desired probability distribution. The main idea is to have each agent follow an independent realization of a Markov chain generated online using the safe M-H algorithm. The
desired distribution emerges asymptotically for the ensemble as each agent in the swarm controls its motion by using the synthesized Markov controls. Hence the swarm achieves a desired steady-state probability distribution, i.e., the desired emergent behavior, while satisfying the safety upper-bound constraints that are aimed at mitigating conflicts/collisions between agents.

Appendix A. Proof of Theorem 2

The first two conditions in Eq. (15) ensure that $K \in \mathcal{M}(v)$ by Lemma 1. The last condition in Eq. (15) ensures that $K$ satisfy the conditions of Theorem 1 and thus $K$ is safe. Moreover, using Lemma 4, the resulting transition matrix $M$ from the M-H algorithm belongs to $\mathcal{M}_G(v)$. It remains to prove that $M$ is safe when $v \in \mathcal{V}_\gamma$. Note that the last condition in (15) is equivalent to the following expression (Theorem 1).

\[
Kx \leq (1 - \gamma)d \quad \text{for all} \quad 0 \leq x \leq \mathbf{d}, \quad x^T \mathbf{1} = 1.
\]  

(A.1)

To show that $M$ is safe, we need to show that for any $0 \leq x \leq \mathbf{d}$, $x^T \mathbf{1} = 1$, we have that $Mx \leq \mathbf{d}$, or equivalently

\[
\sum_k M_{ik}x_k \leq d_i, \quad \text{for } i = 1, \ldots, m.
\]

As a result, for $i = 1, \ldots, m$

\[
\sum_k M_{ik}x_k = M_{ii}x_i + \sum_{k \in N_i} M_{ik}x_k
\]

\[
= \left(1 - \sum_{k \in N_i} \frac{v_k}{v_i} K_{ik} - \sum_{k \in N_i^2} K_{ki}\right)x_i
\]

\[
+ \sum_{k \in N_i} K_{ik}x_k + \sum_{k \in N_i^2} \frac{v_i}{v_k} K_{ki}x_k
\]

\[
= \sum_{k \in N_i} K_{ik}x_k + \left(\sum_{k \in N_i^1} (K_{ki} - \frac{v_k}{v_i} K_{ik}) x_i\right)_{\leq (1 - \gamma)d_i, \quad : = u_i}
\]

\[
+ \sum_{k \in N_i^2} \left(\frac{v_i}{v_k} K_{ki} - K_{ik}\right) x_k_{\leq 0}
\]

\[
\leq (1 - \gamma)d_i + u_i d_i.
\]  

(A.2)

Note that, the first inequality in (A.2) is due to $K$ satisfying (A.1), the last inequality follows directly from the M-H algorithm (i.e., because for any $k \in N_i^2$, $C \geq 1$). It remains to show that $u_i \leq \gamma$,

\[
u_i = \sum_{k \in N_i^1} (K_{ki} - \frac{v_k}{v_i} K_{ik}) = \frac{1}{v_i} \sum_{k \in N_i^1} (v_i K_{ik} - v_k K_{ik})
\]

\[
= \frac{1}{v_i} \sum_{k=1}^{m} \max\{0, v_i K_{ik} - v_k K_{ik}\}
\]

\[
\leq \frac{1}{v_i} \gamma v_i
\]

\[
= \gamma,
\]  

(A.4)

(A.5)

(A.6)

where (A.4) follows from the definition of $N_i^1$ and the inequality (A.5) follows from the definition of $\mathcal{V}_\gamma$. Combining (A.3) and (A.6) shows that $M$ is safe (i.e., $Mx \leq \mathbf{d}$ for all $x \leq \mathbf{d}$), which concludes the proof.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful and constructive comments that greatly contributed to improving the final version of the paper. This research was supported in part by the ONR Grant No. N00014-15-IP-00052 and the NSF Grant No. CNS-1624328.
References


