# PARAMETERIZED SEQUENTIAL DECISION MAKING PROBLEMS

BY

AMBER SRIVASTAVA

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mechanical Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2021

Urbana, Illinois

Doctoral Committee:

Professor Srinivasa Salapaka, Chair and Director of Research
Professor Placid Ferriera
Associate Professor Matthew West
Professor Olgica Milenkovic
Professor Rayadurgam Srikant

# ABSTRACT

This thesis addresses a class of optimization problems that deals with the two-fold objective of making sequential decisions, and simultaneously determining the unknown problem parameters such that the associated cost function gets minimized. We refer to these problems as *parameterized Sequential Decision Making* (para-SDM) problems. The application areas are plenty; for instance, network design problems, job-shop scheduling, industrial process optimization, last mile delivery, vehicle routing, sensor networks, clustering and classification.

In this work, we develop a combinatorial optimization viewpoint for these problems - where the viewpoint is facilitated by the combinatorially large number of possible *sequences of decisions* - and use Maximum Entropy Principle (MEP) based framework to address them. The optimization problems considered in this thesis have been shown to be NP-hard, accompanied by a non-convex cost function whose surface that is riddled by multiple poor local minima. The combinatorially large number of possible sequences of decisions on top of the above mentioned challenges render para-SDM as a difficult class of optimization problems. Our proposed MEP-based framework is designed to overcome the aforesaid challenges in para-SDM. For instance, we employ annealing from a suitable convex function to the non-convex cost function to avoid getting stuck in a poor local minima. Additionally, we utilize the problem structures (such as the law of optimality of the paths) to represent the combinatorial number of possibilites using much smaller decision variable space. The proposed framework is flexible to incorporating application-specific capacity, inclusion-exclusion, and dynamic constraints. Our framework also extends to the class of problems where the information about the underlying model is lacking, and we develop suitable stochastic iterative updates that interacts with the underlying system to simultaneously learn the sequences and the parameter values.

A peculiar characteristic of the annealing process in our MEP-based frameworks is the *phase transition* phenomenon. In particular, these are the specific instances in the annealing procedure at which the solution undergoes *significant* changes. We demonstrate the utility of these phase transitions in determining certain design hyperparamters in para-SDMs, and in general, in combinatorial optimization problems; for instance, estimating the *true* number of clusters in a data set, or determining the appropriate choice of the sparsity level in sparse linear regression problems.

*To family and friends.*

# ACKNOWLEDGMENTS

The work done in this thesis has been made possible by all those who have guided, motivated, and accompanied me in my years as a graduate student. I am sincerely thankful to my advisor Prof. Srinivasa M. Salapaka for mentoring me on various research projects, supporting me in building up on new ideas, guiding me on appropriate topics, and ultimately helping me grow as an independent researcher. I have also been touched by his humility, and his care and concern for all the students - be it the students conducting research along with him, or the students from his classes.

I would like to thank my committee members - Dr. Placid Ferriera, Dr. Mathew West, Dr. Olgica Milenkovic, Dr. R. Srikant for agreeing to be a part of my thesis committee, and for helping me develop deeper insights into the dissertation work. I would also like to thank my collaborator Dr. Carolyn Beck for providing me the opportunity to work on exciting problems in addition to the problems addressed as a part of my thesis.

The research group at UIUC has been an integral part of my journey as a graduate student. I would like to thank my colleagues - Mayank Baranwal, Ram Sai Gorugantu, Sheikh Mashrafi, Alireza Askarian, Sreenath Sundar, Raj Velicheti, Alisina Bayati and Jaesang Park, for working with me and sharing many lighthearted moments during the course of our graduate studies.

I would like to thank all my friends at Urbana-Champaign who have made my stay here memorable. I would cherish all the time that I have spent with them - playing cricket, travelling, cooking, and the general conversations.

Lastly, but certainly not the least, I would like to thank my family for their boundless love and support in all the endeavours that I have taken up.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## Overview of the work

In this thesis we introduce a class of optimization problems - *parameterized* Sequential Decision Making (para-SDM) problems, and propose a comprehensive framework to address them. Para-SDM problems cover a vast range of network logistics and planning application areas such as facility location with path optimization (FLPO), vehicle routing, sensor network design, manufacturing process parameter optimization, last mile delivery, industrial robot-resource allocation, and data aggregation, classification, and clustering algorithms. These problems include large subclasses of problems such as Markov Decision Processes (MDPs), reinforcement learning (RL), clustering, resource allocation, scheduling, and routing problems. Conceptually these problems require *simultaneously* determining the *shortest path* and allocating *resources* in a network, incorporating application-specific capacity and exclusion constraints, and while respecting the dynamical evolution of the network. Note that Para-SDMs can be viewed as a generalization of MDPs in the following context - just as at the heart of MDPs lies the shortest path or *routing* problems (with inherent system dynamics), the central aspect of Para-SDM is to determine simultaneously the shortest path problems in conjunction with facility allocation. Just as in MDPs, there are many variants to the basic para-SDM formulation - for instance, inclusion of various types of *constraints* on routes and resources such as capacity constraints on nodes or, on the links between nodes in a route, communication or topological (inclusion/exclusion) constraints on the nodes, inclusion of *dynamics* on the nodes where the dynamics may be modeled deterministic or stochastic, or inclusion of uncertainty in the model parameters. This thesis develops a flexible framework that addresses the class of optimization problems that fall under the ambit of para-SDM tasks.

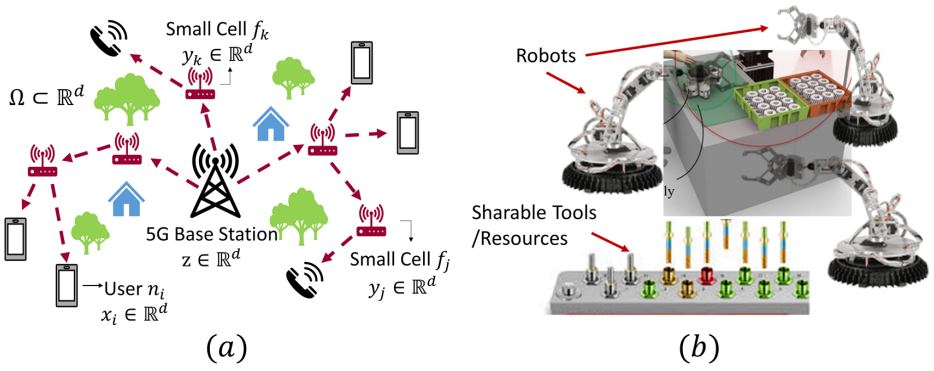


Figure 1.1: (a) 5G small cell networks, (b) Industrial robot-resource allocation

For clear exposition of Para-SDMs, consider an example shown in Figure 1.1(a), which illustrates the design of 5G Small Cell Network where the objective is to simultaneously determine the locations (parameters) of the small cells $\{f_j\}$ and design the communication paths (sequential decisions) between the user nodes $\{n_i\}$ and base station $\delta$ via network of small cells $\{f_j\}$. Here, the constraints can occur in several forms such as the maximum number of hops that an information packet takes in the network, the capacity of the individual small-cells $f_j$, and the capacity of the communication channel between two small cells $f_{j_1}$ and $f_{j_2}$. In the scenarios, where the locations $\{x_i(t)\}$, $z(t)$ of the user nodes and the base station, respectively, are time-varying, the resulting small cell locations and the communication paths, too, need to be time varying to ensure minimum communication cost at each time instant in the network. Figure 1.1(b) illustrates another Para-SDM example - an industrial robot-resource allocation problem; Here the main objective is to determine individual schedules for a group of robots that share resources; more specifically, each robot $R_j \in \mathcal{R}$ has a predefined precedence order in which it uses the available resources $T_i \in \mathcal{T}$, and the objective is to perform scheduling of the resources for the robots such that (a) the cumulative manufacturing time is minimized, and (b) at any given time a maximum of 1 robot is assigned to a particular resource.

In our work, we view the para-SDM task as a *combinatorial optimization* problem - owing to the combinatorially large number of configurations of decision parameters; or equivalently large number of *paths* (or, sequence of decisions) that are possible in these tasks. Combinatorial optimization problems are ubiquitous in science and engineering; for instance, the assignment problem, travelling salesman, job shop scheduling, minimum spanning

tree, the set cover problem, facility location, combinatorial drug discovery, image processing, and graph aggregation problems fall under this category. Interestingly, these problems are inherent in the Nature around us where it meticulously arranges everything - from the atoms in a molecule to the biological systems like cells, or brain. In statistical physics, the selection of a configuration from the large combinatorial configuration space is explained by Free-Energy Principle; according to which, the configuration with minimum Free Energy , which is the difference between the energy of the configuration and irreversible energy (temperature times entropy), is preferred. Maximum Entropy Principle (MEP) [5] is an analogous principle in the area of statistics that provides a way of ascribing probability distribution or weights on the space of possible configurations. In fact, this has motivated the application of MEP in successfully addressing a variety of the above combinatorial optimization problems in engineering [1, 6–9]. Our combinatorial viewpoint of the para-SDM tasks motivates the development of MEP-based frameworks to address them. As illustrated shortly, the resulting algorithms efficiently handle the inherent challenges of the para-SDM problems.

We briefly review the Maximum Entropy Principle (MEP) [5] here since our frameworks rely heavily upon it. MEP states that for a random variable $\mathcal{X}$ with a given prior information, the *most unbiased* probability distribution given prior data is the one that maximizes the Shannon entropy. More specifically, let the known prior information of the random variable $\mathcal{X}$ be given as constraints on the expectation of the functions $f_k : \mathcal{X} \to \mathbb{R}$, $1 \leq k \leq m$. Then, the most unbiased probability distribution $p_{\mathcal{X}}(\cdot)$ solves the optimization problem

$$\begin{aligned} \max_{\{p_{\mathcal{X}}(x_i)\}} \quad & H(\mathcal{X}) = -\sum_{i=1}^{n} p_{\mathcal{X}}(x_i) \ln p_{\mathcal{X}}(x_i) \\ \text{subject to} \quad & \sum_{i=1}^{n} p_{\mathcal{X}}(x_i) f_k(x_i) = F_k \ \ \forall\ 1 \leq k \leq m, \end{aligned} \tag{1.1}$$

where $F_k$, $1 \leq k \leq m$, are known expected values of the functions $f_k$. The above optimization problem results into the Gibbs' distribution [10]

$$p_{\mathcal{X}}(x_i) = \frac{\exp\{-\sum_k \lambda_k f_k(x_i)\}}{\sum_{j=1}^{n} \exp\{-\sum_k \lambda_k f_k(x_j)\}}, \tag{1.2}$$

where $\lambda_k$, $1 \leq k \leq m$, are the Lagrange multipliers corresponding to the constraints in (1.1). The MEP-based frameworks developed in our work start by considering a probability distribution over the space of decision variables; thereafter, we pose the optimization problem similar to (1.1) to determine the most unbiased distribution such that the expected cost function attains a pre-specified value. We solve the optimization problem (1.1) repeatedly at decreasing values of the expected cost function till the distribution over the space of decision variables converges to a particular solution. It is vital to note that in our algorithms we do not explicitly require to know the expected cost function value, albeit, we work with the corresponding Lagrange multiplier $\lambda_k$ that fixes this expected value $F_k$.

There are several challenges associated to the optimization problems addressed in this thesis. To begin with, determining sequential decisions to be taken in finite or infinite horizon and determining the unknown problem parameters *simultaneously* that minimizes the associated cost is a difficult problem. Addressing the two aspects of para-SDM *sequentially* is a way out but it is shown to result into significantly sub-optimal solutions. The decision making aspect alone of para-SDM is a discrete optimization problem that requires determining the optimal solution from a combinatorially (or, even exponentially) large set of possibilities. Here, an exhaustive search is computationally intractable. The problems addressed in this thesis have also been shown to be NP-hard with non-convex cost functions. These cost functions have also been shown to be riddled with multiple poor local minima, where the quality of solution could get dominated by the initialization scheme used in the algorithm. The associated decision variable space in para-SDM problems is large, and require exploiting the system properties to reduce them without loss of generality. Quite often the underlying systems in these problems are time varying, and require the associated solution (parameters, control policy, and the configurations) to evolve dynamically. The straightforward re-runs of the algorithm at each time instant to determine the solution is time-consuming, and thus, require developing methods to determine the time varying solutions without the need to repeatedly solve the associated optimization problem. As illustrated above in the case of 5G small cells, several applications may require various capacity and exclusion constraints to be incorporated in the optimization problem. Many of these applications areas may involve uncertainties at the level of their system

model and require these uncertainties to be methodically incorporated in the problem solution; for instance, the locations $x_i$ of the user node $n_i$ in the Figure 3.1 may not be known with certainty, however, a distribution $p(x_i|n_i)$ encoding the location of $n_i$ may be known. In certain scenarios of para-SDM problems the underlying models of system, such as the state-transition dynamics that determines the subsequent state of the system post a decision and the cost function associated with this transition, may be partially or completely unknown. Such problems require developing algorithms that can exploit their interactions with the *environment* (i.e., the underlying system) to simultaneously *learn* the optimal control policy (governing the sequential decisions), and the unknown parameters.

The main contribution of this thesis is the development of the MEP-based framework and the resulting algorithms that address the above challenges. In our proposed framework, we overcome the computational intractability - due to the associated decision variable space - either by exploiting the inherent structure of the problem (such as *law of optimality* in Facility Location Path Optimization Problems in Chapter 2), or by incorporating suitable constraints on a re-designed decision variable space of much lesser complexity (such as in sparse linear regression in Chapter 7). An essential feature of our framework is *annealing* that comes with multiple benefits to the quality of solution, and as well as provides deeper insights into the nature of the solution. In particular, the process of annealing facilitates a gradual homotopy from a suitable convex function (negative Shannon entropy) to the original non-convex cost function of the problem. We show this is effective in avoiding poor local minima of the original cost function. Additionally, since the algorithm starts by minimizing a convex function it becomes independent of any initialization scheme. Further, we show that several practical network design constraints can be modelled in terms of our decision variables and our framework is flexible to incorporate them in the final solution. Our methodology results into a smooth approximation, referred to as *free energy*, of the original cost function that facilitates sensitivity analysis (which is otherwise difficult in discrete optimization problems) with respect to several problem parameters. Additionally, we show that free-energy is also a good candidate for the choice of a control-Lyapunov function, and thus, helps design control law that governs the dynamical evolution of the solutions to time-varying para-SDM problems.

Another important contribution is determining certain design hyperparameters in para-SDMs. In many problem formulations, these design parameters are assumed to be known, while in practical applications, they are unknown. For instance, data/network classification combinatorial optimization problems deal with categorizing the given elements into a *pre-specified number* of groups. An interesting, though imprecise, question here is to determine what is the *best choice for the number* of these groups? For instance, consider the 5G small cell network illustrated in Figure 1.1(a). Here, in addition to designing the network with a given number of small cells, one may want to determine the appropriate *number* of small cells to be allocated in the network; where allocating too many small cells in the network could be expensive, and too little may result into inefficient communication. Similar questions arise in other combinatorial optimization problems - such as, identifying the *true* number of clusters under a given dataset, the number of partitions $k$ in Multiway $k$-cut problems, and the number of non-zero elements in sparse linear regression. The process of annealing in the MEP-based algorithms indicates specific instances (termed as *phase transitions*) where significant changes to the solutions are observed. We show that quantifying these instances helps us develop a structured methodology that is beneficial to answer the above mentioned *imprecise* aspects of combinatorial optimization.

**Related Work:** Algorithms that address the general class of Para-SDMs are lacking in the literature. There is significantly large literature on one of its main subclasses - the Markov Decision Processes (MDPs). The goal in MDPs is to determine the optimal *control policy* that results in a *path*, a sequence of *actions* and *states*, with minimum cumulative cost; methodologies such as dynamic programming, value and policy iterations [11], and linear programming [12–14] are used to efficiently address the MDPs. The para-SDM problems, however, require simultaneously making sequential decisions and as well as determining the unknown parameters, and hence can not directly be modelled and studied as MDPs. In the case when the parameters are discretized and incorporated in the decision making aspect of the problem, the resulting model will have a large decision variable space that is computationally intractable. Additionally, the discretization may lead to the loss of the *Markov* property as the decision at each time instant may depend upon the exact values of the parameters observed in the past.

Some of the work in recent MDP literature goes beyond just finding an

optimal control policy, and are often referred to as parameterized MDPs [15–24]. For instance the work done in [15] considers the class of MDPs where the state transition probabilities are not known, however, they are bounded by known parameters. This results into an entire *family of MDPs* obtained by taking all possible choices of parameters consistent with these intervals, and the goal is to find a way to measure the quality of a policy for this family of MDPs. The work done in [16] considers the class of problems where the state transition probabilities are parameterized in such a way that observing a few transitions (and inferring the associated parameter) can be helpful in deducing the future transitions, and learning the MDP fully. [17] considers the class of MDPs where the state transition models can defined by a collection of linearly parameterized exponential families. The work in [18] presents action-based parameterization of state space; more precisely, the state space is partitioned, where each partition has a parameters (action) associated to it that affects the associated rewards and transitions. The authors demonstrate its application to service rate control in closed Jackson networks. The work done in [19–24] address the MDPs in the domain of RoboCup soccer where the actions are parameterized in such a way that at each step the agent must select both the discrete action it wishes to execute as well as continuously valued parameters required by that action. On the other hand, the para-SDM problems in this thesis consider the class of problems where the states and actions are parameterized, and the objective is to determine both, the control policy as well as the unknown parameters such that the associated cost is minimized. The parameterized MDPs form a special case of para-SDMs.

When the underlying models for the state dynamics and cost functions are unknown, these parameterized MDPs fall into a new category of parameterized Reinforcement Learning (para-RL) problems. In RL, the algorithms rely upon the instantaneous cost (or reward) generated by the environment to learn the optimal control policy. Some of the popular RL algorithms include Q-learning [25], Double Q-learning [26], Soft Q-learning (entropy regularized Q-learning) [27–33] in discrete state and action spaces, and TRPO [34], and Soft Actor-Critic [35] in continuous spaces. However, these algorithms determine only the control policy for the underlying MDP, and do not deal with simultaneously identifying the unknown parameters in the problem. Further, it is commonly known that for the above algorithms to perform well, all *relevant* states and actions should be explored. Some of the recent works in RL

literature [27–33, 36, 37] rely on the entropy of the stochastic control policy to facilitate this. In particular, the use of entropy of the stochastic control as a regularization term $(-\log \mu(a_t|s_t))$ [27, 28] to the instantaneous cost function $c(s_t, a_t, s_{t+1})$ or maximize the entropy $(-\sum_a \mu(a|s) \log \mu(a|s))$ [29–31] associated *only* to the stochastic policy under constraints on the cost $J$ results into benefits such as better exploration, overcoming the effect of noise $w_t$ in the instantaneous cost $c_t$ and obtaining faster convergence in comparison to Q learning, Double Q-learning [27], and policy gradient methods [29]. However, these stochastic control policies are *not in compliance with MEP* (as illustrated in (1.1)) applied to the distribution over *the paths of MDP*, and thus, are inherently *biased* in their exploration *over these paths*. We demonstrate the downside of these biased policies in terms of convergence rates, performance in noisy environment, and variance in comparison to the unbiased control policies obtained in our MEP-based framework.

## Organization of the thesis and Chapter-wise contributions

The Chapters 2-7 address specific optimization problems in the thesis. At the beginning of each chapter we have included a paragraph that illustrates how they fit into the big picture of the entire thesis. However, we have also tried to keep them as self-contained as possible. Broadly, Chapters 2-4 address the para-SDM (and para-RL) problems, and the Chapters 5-7 demonstrate the phenomenon of phase transitions and their utility in combinatorial optimization problems. The schematic in Figure 1.2 illustrate the dependency of the chapters on each other, and can be used for guidance to navigate through the thesis. Below we illustrate the specific contribution of each chapter.

*Chapters 2 and 3:* Introduces the problem of simultaneous Facility Location and Path Optimization(FLPO). The underlying problem builds-up over the popular Facility Location Problem (FLP), and *additionally* requires determining multi-hop routes (from exponential possibilities) over the allocated facilities such that the transportation cost in the network gets minimized. Here, the two cost functions (facility location and routing) are coupled and thus, need to be optimized simultaneously. This clearly falls under the class of *para-SDM* problems. In our proposed methodology, we develop a *stage-wise viewpoint* of this problem that allows us to exploit the inherent Markov

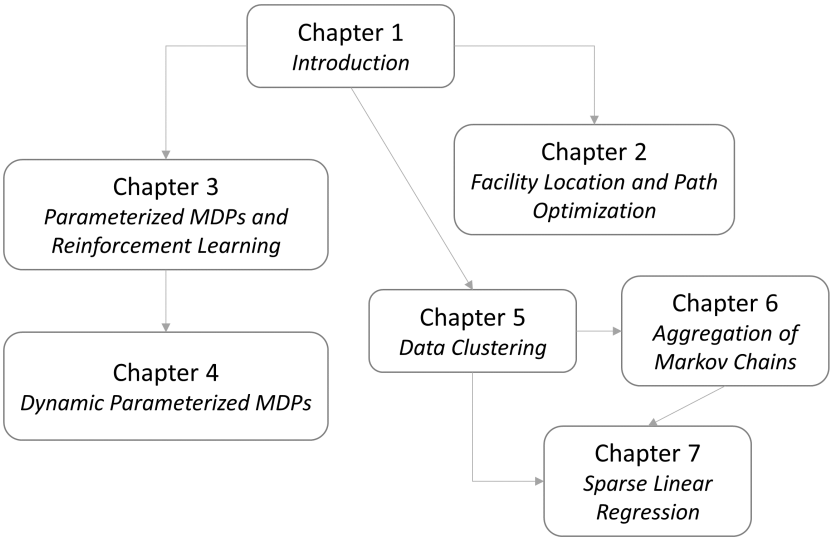


Figure 1.2: Schematic illustrating the dependency of individual chapters.

property of the routes (essential to overcome computational intractability). We demonstrate the capabilities of our framework in incorporating several resource capacity, and network design constraints. In the context of dynamically evolving network nodes (Chapter 3), we design non-linear feedback control law that determines the facility dynamics and the time-varying routes under which asymptotic tracking of local optimal of the FLPO problem is achieved. We also demonstrate that the feedback control law is non-conservative, i.e., if there exists a Lipschitz control law that asymptotically tracks the local optimal of the FLPO problem then the proposed control law is also Lipschitz and bounded.

Interestingly, our stage-wise viewpoint of the FLPO is an abstraction of a larger class of problems - *the parameterized finite-horizon MDPs* - where the associated cost function, states, actions, and the dynamics are (possibly) parameterized. Thus, we can also address the above class of problems using the framework developed in these chapters. The work done here also forms a precursor to the infinite horizon MDPs and reinforcement learning (RL) problems in Chapters 4 and 5.

*Chapters 4 and 5:* In these chapters we consider the class of para-SDM problems over the infinite time horizon and develop a Maximum Entropy Principle (MEP) based framework that simultaneously determines the unknown parameters and the optimal control policy (sequential decisions). Thereafter, we extend it to the case where the underlying cost function and

system dynamics are not known, that is, the case of that requires the algorithm to learn from its interactions with the underlying system (environment). We demonstrate the efficacy of our framework through its utility in the design of 5G small cell networks when the underlying model (cost function and dynamics) is known, as well as unknown. An important aspect of our framework is considering the entropy of the distribution over the *paths* resulting from the sequential decisions, which (in addition to the determining of unknown parameters in our problems) contrasts us from the popular *entropy regularized* methods in literature [27–33,36,37], where the latter consider only the entropy of the control policy. We show the benefit (in terms of final cost and convergence speed) of considering entropy of the distribution over the paths in comparison to the entropy of the control policy. We also obtain sensitivity measures to the SDM problem parameters, and demonstrate robustness of our algorithm to the noisy environment data.

The dynamic para-SDM problems addressed in Chapter 4 determines the time varying control policy, and the parameters such that the resulting cumulative cost gets minimized at each time instant. We develop control theoretic approach to determine this unknown dynamics. In particular, we show that the resulting non-linear feedback control law determines the unknown dynamics for the concerned parameters under which asymptotic tracking of local optimal of the para-SDM problems is achieved. We also demonstrate that the feedback control law is non-conservative, i.e., if there exists a Lipschitz control law that asymptotically tracks the local optimal then the proposed control law is also Lipschitz and bounded.

*Chapter 5, 6, and 7:* In these three chapters we elucidate the benefits of phase transitions, as exhibited by the MEP-based framework, in determining and designing hyper-parameters of the combinatorial optimzation problems. The objective of a combinatorial optimization problem is to determine the appropriate configurations or groupings of elements at a pre-specified *number* of groups (combinations). Here, the choice of number of groups could be ambiguous, and based on the limited prior know-how. In these chapters we provide a structured methodology that helps estimate the *best* choice for the number of groups by developing a novel method that compares the groupings (combinations) obtained at different number of groups. We consider the specific problems of data clustering (Chapter 5), aggregation of Markov chains (Chapter 6), and sparse linear regression (Chapter 7) to illustrate our

proposed method. We additionally note that Chapter 5 and 6 built upon the prior MEP-based frameworks that were developed in [6] and [1]; however, Chapter 7 develops, from scratch, an MEP-based framework to address the sparse linear regression problem which is subsequently analyzed to determine the appropriate sparsity level of its solution.

*Remark:* At the end of each chapter, we have provided a summary of the specific problem addressed in that chapter, the new algorithms developed, and their application to related optimization problems.

# CHAPTER 2

# FACILITY LOCATION AND PATH OPTIMIZATION IN SPATIAL NETWORKS

## 2.1 Introduction

In this chapter we consider the problem of simultaneous Facility Location and Path Optimization(FLPO). The underlying problem builds-up on the popular Facility Location Problem (FLP) in literature [38]. In particular, the FLPO problem requires determining routes (possibly multi-hop) over the allocated facilities such that the transportation cost from one node in the network to another gets minimized. This clearly falls into the category of para-SDM problem, where the facility locations become the unknown parameters and the routes are determined by the sequential decisions. The solution methodology presented in this chapter conveniently generalizes to para-SDM problems in finite horizon.

Many complex systems are modelled as spatial graphs where nodes are embedded into a metric space [39–43]. Areas such as supply chain networks [44], vehicle routing [45], industrial process monitoring and power grids [46], battlefield surveillance [47], disaster management [48, 49], small cell network design in 5G networks [50], wireless networks [51, 52] and last mile delivery [53] come under the purview of spatial networks. Often in these areas a large number of spatially scattered nodes need to transport a commodity (such as information, raw or processed goods) to a given destination (or central processing center). Cost and implementation considerations in these large networks result in nodes that can transport only to nearby locations. This drawback is addressed by overlaying a much smaller network of facilities (special nodes) where each facility has resources to transport commodities to other facilities even if they are far. These facilities also have resources to collect commodities from the nearby nodes. Thus a typical transportation path in such a network would start at a node, go through the network of

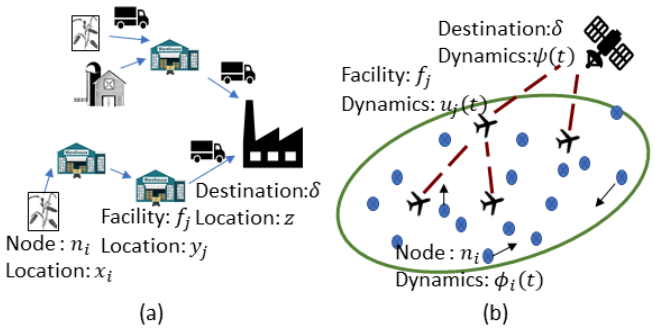


Figure 2.1: (a) An agricultural supply chain which comprises of several farm nodes $n_i$ which transport commodities to the processing unit $\delta$ through a network of warehouses $f_j$. Objective is to determine warehouse locations $\{y_j\}$ and design relevant transportation paths. (b) Battlefield Surveillance - comprises of several nodes $n_i$ (with dynamics $\phi_i(t)$) which communicates with the satellite $\delta$ (with dynamics $\psi(t)$) through a network of facilities $f_j$. Objective is to find the dynamics of the facility locations $\{u_j(t)\}$ and time-varying communication paths.

facilities, and culminate at a destination node (see Fig. 2.1(a)). Therefore designing of such a network requires placement of facilities that *cover* all the nodes and determining the shortest transportation path from each node to the respective destination center.

The problem in the context of overlaying the network of facilities over the network of nodes can be described in terms of the following two objectives - (a) find locations of facilities that cover a large set of underlying nodes and (b) design the shortest transportation path from each node, via the network of facilities, to the destination center such that the total cost of transportation is minimized. For instance, Fig. 2.1(a) illustrates an agricultural supply chain where the farm nodes $n_i$, located at $x_i$, need to transport produce to a food processing center $\delta$ located at $z$ through the network of warehouses $\{f_j\}$. To minimize the cost incurred in the supply chain, the warehouse locations $\{y_j\}$ and the transportation path from each farm node $n_i$ to the processing center $\delta$ needs to be determined. Since all the nodes, destination center and the facilities are static in this problem, we refer to it as simultaneous Facility Location and Path Optimization in static spatial networks, abbreviated as $s$-FLPO.

In many application areas, such as battlefield reconnaissance and disaster management, the nodes and the destination center have associated dynamics. For instance, Fig. 2.1(b) illustrates a scenario of battlefield surveillance where each node $n_i$, with dynamics $\phi_i(t)$, investigates the domain $\Omega$ and

communicates the relevant information in real time to the satellite $\delta$ with dynamics $\psi(t)$. The communication is facilitated through a network of UAVs (Unmanned Aerial Vehicles) $\{f_j\}$ and the objective is to minimize the total cost of communication at each time instant. This task requires determining the appropriate dynamics $u_j(t)$ of each UAV $f_j$ as well as the time-varying optimal communication paths. We refer to this problem as the simultaneous Facility Location and Path Optimization in Dynamic spatial networks, abbreviated as *d*-FLPO.

The goal of the facility location problem in *s*-FLPO is to allocate a set of facilities $\{y_j\}$ to data points $\{x_i\}$ such that the cumulative distance between data points and their nearest facility is minimized. This is an NP-hard problem [54]. Other aspect of *s*-FLPO is the shortest path design problem on a network graph $G(V, E)$ which aims to find a minimum cost path between any two vertices $v_1, v_2 \in V$. This problem is solvable in polynomial order of vertices and edges [55]. The additional objective of determining the shortest path adds to the inherent complexity of the facility location problem thereby resulting into a much more complex *s*-FLPO problem where the cost functions are riddled with multiple poor local minima. A straightforward method to solve the *s*-FLPO problem would be to sequentially solve (a) facility location, and (b) shortest path problems as done in [45] in the context of Multi-depot vehicle routing. However, owing to the fact that the two sub-problems are coupled, such a sequential methodology results in a solution with a much larger cost function value as demonstrated later in our simulations.

There is extensive literature that addresses the facility location problem [1, 7, 56–60] and the shortest path problem [61–63] individually. But there is scant literature that solves the two problems simultaneously. Our previous work in [2] and [3] are, to the best of our knowledge, the only efforts along this direction in the context of spatial networks. However, the framework proposed in [2] does not scale with the number of facilities $M$ and the corresponding algorithm becomes computationally intractable even for small values of $M(\geq 15)$. This is because the framework in [2] views every permutation and combination of the $M$ facilities as a feasible transportation path and requires each such path to be represented by a separate decision variable; resulting into a combinatorially large $\mathcal{O}(\sum_{k=1}^{M} \binom{M}{k} k!)$ decision variable space. In addition to that, the decision variables in [2] fail to provide any *quantitative* insight at the level of the individual facilities $\{f_j\}$ and the

transportation links $\{(f_i, f_j)\}$ such as the usage of a particular facility or a transportation link; due to this the framework in [2] is not flexible to incorporating application-specific capacity based constraints on the facilities and network topology.

On the other hand the framework proposed in [3], though scalable with decison variable space growing polynomially $\mathcal{O}(M^2)$, is applicable to only a restricted class of FLPO problems where all the transportation paths assume a specific structure. As a consequence of this structure, the framework in [3] prohibits all such paths where two distinct facilities $f_{j_1}$ and $f_{j_2}$ establish a transportation link concurrently with the same facility $f_{j_3}$. The set of all feasible paths in [3] consists of only a few *ordered* sequences of facilities where no facility can lie in more than one sequence. Though, enforcing the above path structure in [3] over [2] reduces the decision variable space from combinatorially large $\mathcal{O}(\sum_{k=1}^{M} \binom{M}{k} k!)$ to polynomial size $\mathcal{O}(M^2)$, it results into sub-optimal solutions when used for the general class of FLPO problems that do not necessitate the path structure assumed in [3]. We demonstrate this via simulations in a later section.

In this work we develop a scalable framework for the general class of $s$-FLPO problem to overcome the limitations of [2] and [3]. We achieve this by (a) developing a novel stage-wise viewpoint of the transportation paths (see Fig. 2.2) and (b) exploiting the constraint resulting from the nature of *optimal* transportation paths . More specifically, the stage-wise viewpoint of the paths allows us to impose the structure on the design of decision variables that results from the law of optimality, that is, when any two optimal transportation paths in the network intersect at a particular stage then the subsequent route from the facility at that stage to the destination will be same for both the paths. The stage-wise viewpoint is the main mechanism that allows for substantial reduction in the size of decision variable space from $\mathcal{O}(\sum_{k=1}^{M} \binom{M}{k} k!)$ in [2] to the order $\mathcal{O}(M^3)$ in the current work without enforcing any path structure as in [3].

One of the salient features resulting from our stage-wise illustration of the transportation paths is that it provides quantitative insights into several parameters of the $s$-FLPO problem in terms of the underlying decision variables. For instance, quantities such as the number of transportation paths using a particular facility $f_j$ in a particular stage, fraction of nodes connected directly to a given facility, and number of paths that include a particular transporta-

tion link $(f_i, f_j)$ at a particular stage can be efficiently expressed in terms of the stage-wise decision variables. Note that, once quantifiable in terms of the decision variables it is easier to specify application specific constraints on all such parameters. Additionally, we demonstrate the flexibility of our proposed framework to incorporate various such capacity constraints on facilities, transportation paths and the network topology. We illustrate this using networks where (a) the maximum length of the transportation paths are restricted, (b) the network is partially connected, and/or (c) the facilities and the transportation *links* have maximum associated capacities. It must be noted that incorporating these constraints cause no considerable changes in the algorithm making their implementation straightforward.

In the context of *d*-FLPO problem, we show that the relaxed cost function that appears in the solution framework of the *s*-FLPO problem serves as a good candidate for a control-Lyapunov function. We design a control law for the dynamics $\{u_j(t)\}$ of the facilities which ensures that the time-derivative of this Lyapunov function remains non-positive at all times. The time-varying transportation paths are then determined using the facility locations $\{y_j(t)\}$ at each time instant. The important aspects of our control design that we show are (a) the *asymptotic tracking* of the local minimum to *d*-FLPO problem (Theorem 3), and (b) *non-conservative* property (Theorem 4), i.e., if there exists a Lipschitz control law that asymptotically tracks the local optimal of the *d*-FLPO problem then the proposed control law is also Lipschitz and bounded.

We present extensive simulations for the *s*-FLPO problem in unconstrained and constrained scenarios to demonstrate the efficacy of our algorithm in terms of scalability, time-complexity and the quality of solutions. Also a wide range of constraints on the facilities, transportation path, and network topology are demonstrated in our simulations. We demonstrate the efficacy of our framework on a large scale system with approximately $N = 17000$ nodes and allocate $M = 50$ facilities with the cost function values that are approximately 25% better than the solution obtained using [3]; note that the framework in [2] results in $(\approx)10^{64}$ decision variables for this scenario and hence the corresponding algorithm is severely intractable.

We compare our simulations of the *d*-FLPO problem with the *frame-by-frame* approach where we solve the *s*-FLPO problem at each time instant to estimate the dynamics of the facilities and transportation paths. We show

the considerable benefits of our proposed methodology obtained with respect to algorithmic run times and practicality of the dynamics of the facilities in comparison to the frame-by-frame approach. The computational times are significantly reduced; as much as by 700 times have been demonstrated. Also, the simulations show that the frame-by-frame approach requires some facilities to undergo a considerable spatial change in a very small interval of time thereby resulting into a non-viable dynamics which do not occur when using our methodology.

## 2.2 The *s*-FLPO Problem

The *s*-FLPO problem is characterized by overlaying a network of $M$ facilities on a network of large number $N \gg M$ of nodes and designing a path (single or multi-hop) from each node to the destination via the network of facilities. Let the node $n_i$ be located at $x_i \in \mathbb{R}^d$, $1 \leq i \leq N$ and the destination $\delta$ be located at $z \in \mathbb{R}^d$. Let $\Gamma_0 = \{n_1, \ldots, n_N\}$ denote the set of all nodes. The two-fold objective of the optimization problem is to (a) determine the location $y_j \in \mathbb{R}^d$ of the facilities $f_j$, $1 \leq j \leq M$ and (b) design transportation paths from each node $n_i$ to the destination $\delta$ via the network of facilities such that the total cost of transportation (as quantified later in this section) is minimized. A transportation path

$$n_i \to f_{r_1} \to f_{r_2} \to \ldots \to f_{r_q} \to \delta, \tag{2.1}$$

where $r_j \in \{1, \ldots, M\}$, from the node $n_i \in \Gamma_0$ to the destination $\delta$ is an ordered sequence of $q$ ($\leq M$) distinct facilities. For such a path we say that the path length (or number of hops) is $q$. In our framework we model a transportation path $\gamma$ from a node $n_i \in \Gamma_0$ to the destination $\delta$ as a sequence

$$\gamma = (\gamma_1, \ldots, \gamma_M), \tag{2.2}$$

where $\gamma_k \in \Gamma_k \ \forall \ 1 \leq k \leq M$. Here the stage $\Gamma_k$ is the collection of all the facilities and the destination center, that is, $\Gamma_k = \{f_1, \ldots, f_M, \delta\} \ \forall \ 1 \leq k \leq M$ (see Fig. 2.2). For the transportation path in (2.1) $\gamma_k = f_{r_k} \ \forall \ k \in \{1, \ldots, q\}$ and $\gamma_k = \delta \ \forall \ k \in \{q+1, \ldots, M\}$, i.e. in our representation of a transportation path we pad (2.1) with $M - q$ many $\delta$'s at the end. We

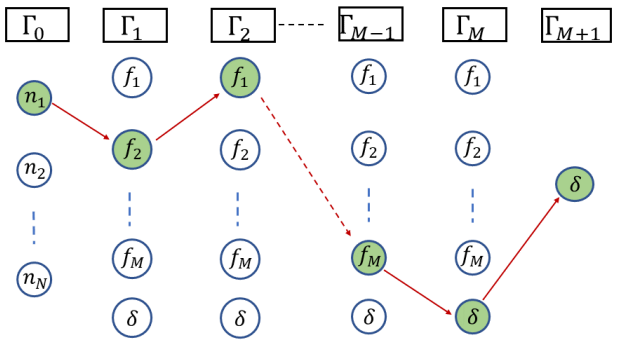


Figure 2.2: Illustrates a transportation path $\gamma$ from the node $n_1 \in \Gamma_0$ to the destination $\delta \in \Gamma_{M+1}$ via the stages $\{\Gamma_k\}_{k=1}^M$.

define $\Gamma_{M+1} := \{\delta\}$ as a singleton set comprising of the destination center and $\mathcal{G} := \{(\gamma_1, \ldots, \gamma_M) : \gamma_k \in \Gamma_k \forall 1 \leq k \leq M\}$ as the set of all possible transportation paths. Please refer to Fig. 2.2 for a pictorial illustration of a transportation path from $n_1 \in \Gamma_0$ to destination $\delta \in \Gamma_{M+1}$ via the stages $\{\Gamma_k\}_{k=1}^M$. The objective of the $s$-FLPO problem is to

$$\min_{\substack{\{y_j\} \\ 1 \leq j \leq M}} \quad D_0 := \sum_{\gamma_0 \in \Gamma_0} \Big[ \rho_{\gamma_0} \sum_{\gamma \in \mathcal{G}} \nu(\gamma|\gamma_0) d(\gamma_0, \gamma) \Big], \tag{2.3}$$

where $\rho_{\gamma_0}$ is a given relative weight of the node $\gamma_0 \in \Gamma_0$,

$$\nu(\gamma|\gamma_0) = \begin{cases} 1, & \text{if } \gamma = \arg\min_{\gamma' \in \mathcal{G}} d(\gamma_0, \gamma') \\ 0, & \text{otherwise} \end{cases} \tag{2.4}$$

and $d(\gamma_0, \gamma) = \sum_{k=0}^M d_k(\gamma_k, \gamma_{k+1})$ is the cost incurred along the path $\gamma = (\gamma_1, \ldots, \gamma_M)$ from the node $\gamma_0$ to the destination $\delta$. Here $d_k(\cdot, \cdot)$ represents the cost of transportation from the stage $\Gamma_k$ to $\Gamma_{k+1}$. For notational simplicity we denote $d_k(\gamma_k, \gamma_{k+1})$ as $d_k$ wherever clear from the context. The framework presented in this paper is applicable to any general cost function $d_k(\gamma_k, \gamma_{k+1})$; however, for the purpose of illustration we assume it to be the squared euclidean distance, i.e. $d_k(\gamma_k, \gamma_{k+1}) = \|y_{r_k} - y_{r_{k+1}}\|^2$, where $\gamma_k = f_{r_k}$, $\gamma_{k+1} = f_{r_{k+1}}$.

## 2.3 Solution to $s$-FLPO problem

A straightforward approach to solving the two-fold objective optimization problem (2.3) is to solve for the two objectives sequentially, that is, first allocate facilities to the nodes using any of the facility location algorithms mentioned in Section 2.1 and then find the shortest transportation path from each node to the destination by solving the shortest path problem on the resulting network graph [64]. However, this approach disregards the fact that the two objectives are coupled and therefore results in a suboptimal solution. Also, the algorithms mentioned in Section I, which solve the facility location problem, are substantially dependent on the initialization step. For instance, in Lloyd's algorithm (or $k$-means algorithm) [59], [65] the initial step consists of randomly choosing facility locations to form the initial facility locations. Since the iterative scheme is such that only the 'proximal' input points determine the facility location and not the 'distant' input points, the $k$-means algorithm has a tendency to get trapped in the local minima.

The algorithm that we propose in this work is motivated from the Deterministic Annealing algorithm for facility location problem [58]. In particular, the proposed algorithm overcomes the local influence of the nodes on the solution to $s$-FLPO problem by associating each node $\gamma_0$ with all the transportation paths through a weighting parameter $p(\gamma|\gamma_0)$. Without loss of generality, we assume that $\sum_{\gamma\in\mathcal{G}} p(\gamma|\gamma_0) = 1 \ \forall \ \gamma_0 \in \Gamma_0$ and the proposed algorithm seeks to minimize the relaxed version of the cost function in (2.3) given by

$$D = \sum_{\gamma_0\in\mathcal{S}} \rho_{\gamma_0} \sum_{\gamma\in\mathcal{G}} p(\gamma|\gamma_0) d(\gamma_0, \gamma). \tag{2.5}$$

It must be noted that the choice of association weights $p(\gamma|\gamma_0)$ determines the trade-off between the local influence (or the initialization of the algorithm) and deviation from the original cost function (2.3), i.e. if the association weights are uniformly distributed $p(\gamma|\gamma_0) = 1/|\mathcal{G}|$, then the two cost functions differ largely from each other, however the minimization of the objective function (2.5) is independent of the initialization of the algorithm as all possible paths from the nodes $\gamma_0$ are given equal weights. For the choice of association weights $p(\gamma|\gamma_0) = \nu(\gamma|\gamma_0)$ the relaxed cost function (2.5) reduces back to the original cost function (2.3).

It follows from the *law of optimality* that along an optimal transportation path, the upcoming facility on the path is decided solely by the current facility and is independent of the prior facilities on that path. We impose this structure on our choice of the association weights $p(\gamma|\gamma_0)$, which translates to a Markov property. Thus the association weight $p(\gamma|\gamma_0)$, which relates an entire transportation path $\gamma = (\gamma_1, \ldots, \gamma_M)$ to the node $\gamma_0$, can be broken down into association weights $\{p_k(\gamma_{k+1}|\gamma_k)\}_{k=1}^{M}$ where $p_k(\gamma_{k+1}|\gamma_k)$ relates the stage $\Gamma_k$ to $\Gamma_{k+1}$. More specifically,

$$p(\gamma|\gamma_0) = \prod_{k=0}^{M} p_k(\gamma_{k+1}|\gamma_k). \tag{2.6}$$

For notational simplicity, we denote $p_k(\gamma_{k+1}|\gamma_k)$ as $p_k$ whenever it is clear from the context. The association weights $p_k(\gamma_{k+1}|\gamma_k)\ \forall\ 0 \leq k \leq M-1$ along with the spatial coordinates $\{y_j\}$, $1 \leq j \leq M$ of the facilities comprises of the decision variable space of our optimization problem. The relaxed cost function in (2.5) is now rewritten as

$$D = \sum_{\gamma_0 \in \Gamma_0} \rho_{\gamma_0} \sum_{\gamma \in \mathcal{G}} \prod_{k=0}^{M} p_k d(\gamma_0, \gamma). \tag{2.7}$$

Observe that the decision variable $p(\gamma|\gamma_0)$ in (2.5) is replaced by the decision variable $p_k(\gamma_{k+1}|\gamma_k)$ in (2.7), thereby making our optimization problem across all possible paths $\gamma$ to an optimization problem across consecutive stages $\Gamma_k$ and $\Gamma_{k+1}$, $1 \leq k \leq M$. This results into the reduction of decison variable space from $O(\sum_{k=1}^{M} \binom{M}{k} k!)$ to $O(M^3)$.

We use the Maximum Entropy Principle (MEP) [66], [67] to design the association weights $p_k(\gamma_{k+1}|\gamma_k)$ such that the cost function (2.7) attains a specified value. More specifically, MEP determines the association weights by solving the following associated optimization problem

$$\max_{\{p_k\}} H := -\sum_{\gamma_0 \in \mathcal{S}} \rho_{\gamma_0} \sum_{\gamma \in \mathcal{G}} \Big( \prod_{k=0}^{M-1} p_k \Big) \log \Big( \prod_{k=0}^{M-1} p_k \Big), \tag{2.8}$$

$$\text{s.t. } D = c_0 \tag{2.9}$$

where $c_0$ is a given value of the cost function. This problem is solved repeatedly at decreasing values of $c_0$, which is described later in Section 2.3. As

the entropy term (2.8) quantifies for the level of randomness, maximizing it at a fixed value $c_0$ of the cost function (2.7) results into association weights $p_k(\gamma_{k+1}|\gamma_k)$ that ensures maximum uncertainty or uncommitted nature of the algorithm towards any particular solution. The Lagrangian corresponding to the optimization problem in (2.8)-(2.9) is given by

$$\bar{F} = (D - c_0) - \frac{1}{\beta} H, \tag{2.10}$$

where $1/\beta$ is the Lagrange multiplier. The Lagrangian $\bar{F}$ is convex in $p_k \; \forall \; k$ and we determine the association weights by setting $\frac{\partial \bar{F}}{\partial p_k} = 0$ which yields

$$p_k = \left(e^{-\beta d_k}\right) \frac{\sum\limits_{\substack{(\sigma_{k+2},\ldots,\sigma_M):\\ \sigma_{k+1}=\gamma_{k+1}}} e^{-\beta \sum_{t=k+1}^{M} d_t(\sigma_t,\sigma_{t+1})}}{\sum\limits_{\substack{(\sigma_{k+1},\ldots,\sigma_M):\\ \sigma_k=\gamma_k}} e^{-\beta \sum_{t=k}^{M} d_t(\sigma_t,\sigma_{t+1})}}. \tag{2.11}$$

In the expression of the unconstrained lagrangian (2.10) we refer to the lagrange parameter $\frac{1}{\beta}$ as the *temperature* and $\bar{F}$ as the *Free energy* because of their close analogies to statistical physics (where Free energy is enthalpy (D) minus the temperature times entropy (TH)). Substituting (2.11) into the expression of *free energy* $\bar{F}$ in (2.10) we obtain

$$F = -\frac{1}{\beta} \sum_{\gamma_0 \in \Gamma_0} \rho_{\gamma_0} \log \sum_{\gamma \in \mathcal{G}} e^{-\beta \sum_{t=0}^{M} d_t(\gamma_t,\gamma_{t+1})}. \tag{2.12}$$

Note that for brevity we ignore the constant term $c_0$ in the above expression of $F$. As illustrated later the Lagrange parameter $\beta$ implicitly decides the value of $c_0$. Additionally, the above $F$ can be viewed as a relaxation of the cost function $D$ in (2.7). In fact, as $\beta \to \infty$ we observe that $F \to D$. We now minimize (locally) $F$ in (2.12) with respect to $y = [y_1^T, \ldots, y_M^T]^T$ to obtain the spatial coordinates of the facilities, i.e. we put $\frac{\partial F}{\partial y} = 0$ to obtain

$$y = (2\hat{A} - \hat{B})^{-1}(\hat{\bar{X}} + \hat{C}). \tag{2.13}$$

where $\hat{A} = I_d \otimes A$, $\hat{B} = I_d \otimes B$, $\hat{\bar{X}} = I_d \otimes \bar{X}$, $\hat{C} = I_d \otimes C$ and $I_d$ is an identity matrix of $d \times d$ dimension. The matrices $A, B \in \mathbb{R}^{M \times M}$, $\bar{X}, C \in \mathbb{R}^{M \times d}$ depend on the association weights $p_k(\gamma_{k+1}\gamma_k)$, the spatial coordinates of the nodes

$\{x_i\}$ and the destination location $z$. Please refer to the Appendix A.1.1 for the definitions of the above matrices and the proof that the matrix $(2\hat{A} - \hat{B})$ is positive definite (i.e., invertible).

The constraint value $c_0$ in (2.9) decides the temperature variable $T = 1/\beta$. It follows from the sensitivity analysis [10] that lower value of $c_0$ corresponds to the higher value of $\beta$. It is clear from (2.10) that for small values of $\beta$ (i.e. high values of $c_0$), we are mainly minimizing the convex function $-H$, which results into uniformly distributed association weights. As $\beta$ increases ($c_0$ decreases), more and more weightage in (2.10) is given to the cost function $D$ i.e. $F$ closely approximates the non-convex cost function $D$. In the limit $\beta \to \infty$ we have that $F = D$ and we obtain hard association weights $p_k(\gamma_{k+1}|\gamma_k) \in \{0, 1\}$. The idea is essentially to find the global minimum of the convex function $-H$ and then track the minimum of $F$ at successively increasing values of $\beta$, until $F \to D$. This is done in our algorithm via iterating between (2.13) and (2.11) at successively increasing value of $\beta$, and using the solution from each $\beta$ iteration as an initialization for the next $\beta$ iteration. The resulting annealing algorithm is as follows.

**Algorithm 1** main($X$, $z$, $\beta_{\min}$, $\beta_{\max}$)

1. Initialize $\beta$ to the small value $\beta_{\min}$.
2. Calculate the association weights $\{p_k(\gamma_{k+1}|\gamma_k)\}$ in (2.11).
3. Calculate the facility locations $y$ in (2.13).
4. Iterate between step 2 and 3 until convergence.
5. Increase $\beta$ by a factor $\kappa > 1$; i.e. $\beta \leftarrow \kappa\beta$.
6. Stop if $\beta \geq \beta_{\max}$. Else go to step 2.

**Theorem 1.** *The iterations in step 4 of the Algorithm 1 are equivalent to iterations of a descent method to solve the implicit equation (2.13). Consequently Algorithm 1 converges.*

Please refer to the Appendix A.1.2 for the proof.

## 2.4 Phase Transition Phenomenon

The above algorithm upon implementation exhibits *phase transitions* very similar to that seen when Deterministic Annealing (DA) [58] is applied to pure facility location problems. In the initial iterations of the algorithm when $\beta$ is very small the cost function is dominated by $-H$; minimizing this gives uniform distributions for the association weights $p_k(\gamma_{k+1}|\gamma_k)$. Also with this uniform distribution of the association weights all the facilities get allocated at the same spatial coordinates given by (2.13) and all the corresponding transportation path are same. Now as $\beta$ is increased, the simulations show that there is no perceptible change on the facility locations till a critical value of $\beta = \beta_{cr1}$ is reached; beyond which the number of *distinct* facility locations and the number of distinct transportation paths increases. Again as $\beta$ is increased further, there is no change in the facility location and transportation paths till the next critical value of $\beta = \beta_{cr2}$ is reached, where the number of distinct facility locations and transportation paths increases again. These *phase transitions* are of interest as they can help control the number of distinct facilities that one may want to allocate to the network of nodes and also help in speeding up the annealing process.

The critical values of $\beta$ $(\beta_{\text{cr1}}, \beta_{\text{cr2}}, \ldots)$ are obtained by tracking the conditions for attaining the minimum of *free-energy* $F$. At $\beta = 0$, the free-energy function is convex, and setting $\frac{\partial F}{\partial y} = 0$ gives the global minimum, also the hessian $\frac{\partial^2 F}{\partial y^2}$ is positive definite. As $\beta$ increases, at a particular $\beta = \beta_{cr1}$, $\frac{\partial F}{\partial y} = 0$ and the Hessian *loses* rank. Here bifurcation occurs leading to an increase in the number of distinct facility locations. Using variational calculus, the necessary condition for $y$ to be a minimum of $F$ requires that for all choices of finite perturbation $\psi$

$$\left.\frac{\partial F_\epsilon}{\partial \epsilon}\right|_{\epsilon=0} = 0, \quad \text{and} \tag{2.14}$$

$$\left.\frac{\partial^2 F_\epsilon}{\partial \epsilon^2}\right|_{\epsilon=0} = \sum_{\gamma\in\mathcal{G}} p(\gamma)\Lambda_\gamma^T\Big(\mathbb{I} - 2\beta\Upsilon_\gamma)\Big)\Lambda_\gamma + 2\beta \sum_{\gamma_0\in\Gamma_0} \rho_{\gamma_0}\Big[\sum_{\gamma\in\mathcal{G}} p(\gamma|\gamma_0)K_\gamma^T\Lambda_\gamma\Big]^2 > 0, \tag{2.15}$$

where $F_\epsilon = F(y + \epsilon\psi)$, $p(\gamma) = \sum_{\gamma_0} \rho(\gamma_0)p(\gamma|\gamma_0)$, $\Upsilon_\gamma = \sum_{\gamma_0} p(\gamma_0|\gamma)K_\gamma K_\gamma^T$ and $\Lambda_\gamma, K_\gamma$ are as defined in the Appendix A.1.3. We characterize phase

transition as below.

**Theorem 2.** *The critical value of $\beta$ at which the hessian (2.15) is no longer positive definite, i.e, it loses rank is given by $\beta_{cr} = \max_\gamma \big(2\lambda_{\max}(\Upsilon_\gamma)\big)^{-1}$ where $\lambda_{\max}(\Upsilon_\gamma)$ is the largest eigenvalue of the matrix $\Upsilon_\gamma := \sum_{\gamma_0} p(\gamma_0|\gamma) K_\gamma K_\gamma^T$.*

See appendix A.1.3 for proof.

## 2.5 Adding Multiple Capabilities and Constraints to the Problem

In various applications involving spatial networks the overall design goal includes efficient utilization of facilities and the transportation paths. This often corresponds to incorporating several application based constraints on the network topology or on the facilities. In this section we elucidate the flexibility of our proposed approach in incorporating such constraints.

*A. Restricted Number of Hops:* In certain applications it is be beneficial to restrict the path length $q$ of the shortest transportation path as any extra-hop for the commodity may involve associated penalties and overheads such as processing energy cost and time delays. Let $L_{\gamma_0}$ be the given maximum allowable path length (or the number of hops) for a commodity originating at a node $\gamma_0 \in \Gamma_0$. Therefore all the transportation paths $\gamma \in \mathcal{G}$ with path length greater than $L_{\gamma_0}$ become invalid for $\gamma_0$. This constraint enforces that on an optimal transportation path the facility $\gamma_{k+1}$ depends on the facility $\gamma_k$ as well as the originating node $\gamma_0$ which results into the dissociation of the association weight $p(\gamma|\gamma_0)$ as $p(\gamma|\gamma_0) = \prod_{k=0}^{M} p_k(\gamma_{k+1}|\gamma_k, \gamma_0)$. For notational simplicity we denote $p_k(\gamma_{k+1}|\gamma_k, \gamma_0)$ by $p_{k,\gamma_0}$ whenever it is clear from the context. Using the Maximum Entropy Principle we obtain the association weights $p_{k,\gamma_0}$ as the Gibbs distribution

$$p_{k,\gamma_0} = \Big(e^{-\beta d_k}\Big) \frac{\sum\limits_{\substack{\sigma_{k+2},\ldots,\sigma_{L_{\gamma_0}} \\ \sigma_{k+1}=\gamma_{k+1}}} e^{-\beta \sum_{t=k+1}^{L_{\gamma_0}} d_t(\sigma_t,\sigma_{t+1})}}{\sum\limits_{\substack{\sigma_{k+1},\ldots,\sigma_{L_{\gamma_0}} \\ \sigma_k=\gamma_k}} e^{-\beta \sum_{t=k}^{L_{\gamma_0}} d_t(\sigma_t,\sigma_{t+1})}}, \tag{2.16}$$

where $\sigma_l \in \Gamma_l \ \forall \ l \in \{k+1, \ldots, L_{\gamma_0}\}$, which when substituted into the expression of $\bar{F}$ in (2.10) results into

$$F_1 = -\frac{1}{\beta} \sum_{\gamma_0} \rho_{\gamma_0} \log \sum_{\gamma_1, \ldots, \gamma_{L_{\gamma_0}}} e^{-\beta \sum_{t=0}^{L_{\gamma_0}} d_t(\gamma_t, \gamma_{t+1})}. \tag{2.17}$$

Minimizing (locally) $F_1$ in (2.17) by setting $\frac{\partial F_1}{\partial y} = 0$, we obtain the facility locations $y$.

*B. Structured (Partially Connected) Network:* A transportation link $(f_i, f_j)$ exists (or is active) if a commodity packet is permitted to hop between the facilities $f_i$ and $f_j$. In a partially connected spatial network, some of the transportation links are absent owing to which the corresponding facility pairs are unable to send across commodities. Fig. 2.3 demonstrates a fully connected and a partially connected network topology. To incorporate the partial connectivity of the network as a constraint in the optimization problem (2.3) we introduce a connectivity parameter $\omega_{f_i, f_j}$ defined as

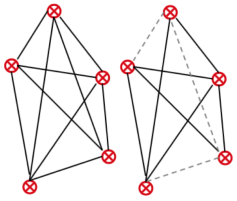

Figure 2.3: A fully-connected and a partially connected network.

$$\omega_{f_i, f_j} = \begin{cases} 1 & \text{if transportation link } (f_i, f_j) \text{ exists} \\ 0 & \text{otherwise} \end{cases}.$$

We incorporate the connectivity parameter $\omega_{f_j, f_k}$ into the expression of the association weights $p_k(\gamma_{k+1}|\gamma_k)$ (2.11) in such a manner which (a) assigns $p_k(f_j|f_i) = 0 \ \forall \ k$ for the non-existent link $(f_i, f_j)$ (b) rules out all transportation paths that consist of the non-existent link from the solution space thus leading to the following expression of the association weights

$$p_k = \omega_{\gamma_k, \gamma_{k+1}} e^{-\beta d_k} \frac{\sum\limits_{\substack{\sigma_{k+2}, \ldots, \sigma_M \\ \sigma_{k+1} = \gamma_{k+1}}} \prod\limits_{t=k+1}^{M} \omega_{\sigma_t \sigma_{t+1}} e^{-\beta d_t}}{\sum\limits_{\substack{\sigma_{k+1}, \ldots, \sigma_M \\ \sigma_k = \gamma_k}} \prod\limits_{t=k}^{M} \omega_{\sigma_t \sigma_{t+1}} e^{-\beta d_t}}. \tag{2.18}$$

We substitute the above association weights into the expression of free energy $\bar{F}$ in (2.10) to obtain $F_2$, which is then minimized (locally) by setting

$\frac{\partial F_2}{\partial y} = 0$ to obtain the facility locations $y$.

*C. Capacity constraint on facilities and path links:* In certain applications the capacity and cost constraints on the facilities result into corresponding constraints on its usage by transportation paths. For instance, the warehouses $f_j$ in Fig. 2.1(a) may have limited storage capacity for agricultural goods collected from the farms. The decision variables in our framework appropriately quantify the usage $C(f_j)$ of each facility $f_j$ as

$$\begin{aligned} C(f_j) = &\sum_{\gamma_0} \rho_{\gamma_0} p_0(f_j|n_0) + \sum_{\gamma_0,\gamma_1} \rho_{\gamma_0} p_0(\gamma_1|\gamma_0) p_1(f_j|\gamma_1) \\ &+ \ldots + \sum_{\gamma_0,\ldots,\gamma_{M-1}} \rho_{\gamma_0} p_0(\gamma_1|\gamma_0) \ldots p_{M-1}(f_j|\gamma_{M-1}), \end{aligned} \tag{2.19}$$

where the first term in the above expression indicates the effective number of transportation paths passing over $f_j$ in the stage $\Gamma_1$, second term indicates the effective number of paths passing over $f_j$ in the stage $\Gamma_2$, so on and so forth till the last term which indicates the effective number of paths passing over $f_j$ in the last stage $\Gamma_M$. We address the facility capacity constraint by requiring the usage of the facility $f_j$ to be given by $C(f_j) = w_j$ where $w_j$ denotes the pre-defined capacity of the $j$-th facility. The corresponding unconstrained Lagrangian is given by

$$\bar{F}_3 = \bar{F} + \sum_{j=1}^{M} \alpha_{f_j} \big( C(f_j) - w_j \big), \tag{2.20}$$

where $\bar{F}$ is given in (2.10). Minimizing $\bar{F}_3$ with respect to the association weights $\{p_k(\gamma_{k+1}|\gamma_k)\}$ we obtain

$$p_k = e^{-\beta(d_k + \alpha_{\gamma_{k+1}})} \frac{\sum_{(\gamma_{k+2},\ldots,\gamma_M)} e^{\sum_{t=k+1}^{M} -\beta(d_t + \alpha_{\gamma_{t+1}})}}{\sum_{(\gamma_{k+1},\ldots,\gamma_M)} e^{\sum_{t=k}^{M} -\beta(d_t + \alpha_{\gamma_{t+1}})}}, \tag{2.21}$$

where $\alpha_{\gamma_k}$ is a Lagrange multiplier in (2.20). We refer to $\zeta_{\gamma_k} := e^{-\beta\alpha_{\gamma_k}} \forall \gamma_k \in \Gamma_k \backslash \{\delta\}$ as the weight parameter. Substituting (2.21) in (2.20) to obtain $F_3$ and minimizing (locally) $F_3$ with respect to $y$ gives the expression for facility locations $y$. To obtain the parameters $\zeta_{f_j}$ we substitute the association

weights (2.21) in the expression (2.19) of $C(f_j)$ and equate the subsequent expression to $w_j$ (i.e. set $C(f_j) = w_j$). This results in the update equation

$$\zeta_{f_j}^{p+1} = \zeta_{f_j}^{p} \frac{w_j}{C(f_j)}, \forall j \in \{1, 2, \ldots, M\}. \tag{2.22}$$

In our algorithm we minimize the *free-energy* $\bar{F}_3$ at successively increasing values of $\beta$ by alternating between the expressions of association weights in (2.21), facility locations $y$ and the update equation for $\zeta_{f_j}$ in (2.22).

Similarly, in certain application areas the amount of traffic $\eta_{f_i f_j}$ that a transportation link $(f_i, f_j)$ is able to handle is known apriori. Using the decision variables in our framework we appropriately quantify the usage $C_{f_i f_j}$ of each transportation link from $f_i$ to $f_j$ as

$$\begin{aligned} C_{f_i f_j} &= p_1(f_j|f_i) \sum_{\gamma_0} \rho_{\gamma_0} p_0(f_i|\gamma_0) \\ &+ p_2(f_j|f_i) \sum_{\gamma_0, \gamma_1} \rho_{\gamma_0} p_0(\gamma_1|\gamma_0) p_1(f_i|\gamma_1) + \ldots \\ &+ p_{M-1}(f_j|f_i) \sum_{\gamma_0, \ldots, \gamma_{M-2}} \rho_{\gamma_0} p_0(\gamma_1|\gamma_0) \cdots p_{M-2}(f_i|\gamma_{M-2}), \end{aligned} \tag{2.23}$$

where the first term in the above expression is the fraction of total paths with node $f_i \in \Gamma_1$ and $f_j \in \Gamma_2$, i.e. the fraction of paths with $(f_i, f_j)$ transportation link occurring in the hop from stage $\Gamma_1$ to stage $\Gamma_2$. Similarly the other subsequent terms count the fraction of the transportation paths with $(f_i, f_j)$ link in the hop from stage $\Gamma_k$ to $\Gamma_{k+1}$ for all $k \in \{2, \ldots, M-1\}$. The unconstrained Lagrangian in this case is given by

$$\bar{F}_4 = \bar{F} + \sum_{i,j:i \neq j} \alpha_{f_i f_j} \big( C_{f_i f_j} - \eta_{f_i f_j} \big), \tag{2.24}$$

where $\bar{F}$ is given by (2.10). The association weights $\{p_k(\gamma_{k+1}|\gamma_k)\}$ that minimize $\bar{F}_4$ are given by

$$p_k = e^{-\beta(d_k + \alpha_{\gamma_k \gamma_{k+1}})} \frac{\sum\limits_{(\gamma_{k+2}, \ldots, \gamma_M)} e^{\sum_{t=k+1}^{M} -\beta(d_t + \alpha_{\gamma_t, \gamma_{t+1}})}}{\sum\limits_{(\gamma_{k+1}, \ldots, \gamma_M)} e^{\sum_{t=k}^{M} -\beta(d_t + \alpha_{\gamma_t, \gamma_{t+1}})}}, \tag{2.25}$$

where $\lambda_{\gamma_k \gamma_{k+1}} := e^{-\beta \alpha_{\gamma_k \gamma_{k+1}}} \forall \gamma_k, \gamma_{k+1}$ is referred to as the weight parameter.

Substituting (2.25) in the expression of *free-energy* $\bar{F}_4$ (2.24) to obtain $F_4$ and setting $\frac{\partial F_4}{\partial y} = 0$ gives the facility locations $y$. To obtain the parameters $\lambda_{f_i f_j}$ we substitute the association weights (2.25) in the expression (2.23) of $C_{f_i f_j}$ and equate the subsequent expression to $\eta_{f_i f_j}$ (i.e. set $C_{f_i f_j} = \eta_{f_i f_j}$). This results in the update equation

$$\lambda_{f_i f_j}^{p+1} = \lambda_{f_i f_j}^{p} \frac{\eta_{f_i f_j}}{C_{f_i f_j}}. \tag{2.26}$$

As before, our algorithm minimizes *free-energy* $\bar{F}_4$ at successively increasing values of $\beta$ by alternating between the expressions of the association weights (2.25), facility locations $y$ and the update equation for $\lambda_{f_i f_j}$ in (2.26).

## 2.6 Extension to Dynamic Spatial Networks

In the case of dynamic spatial networks, the nodes and the destination center have an associated dynamics given by continuously differentiable velocity fields $\phi_i(x_i(t), t) \in \mathbb{R}^d$, $1 \leq i \leq N$ and $\psi(z(t), t) \in \mathbb{R}^d$ respectively. The resulting facility locations and the transportation paths are also time varying and the entire dynamical system is represented as

$$\dot{\zeta} = f(\zeta(t), t) \Longleftrightarrow \begin{cases} \dot{x}(t) &= \Phi(x(t), t) \\ \dot{z}(t) &= \Psi(z(t), t) \\ \dot{y}(t) &= u(x(t), z(t), y(t), t) \end{cases}, \tag{2.27}$$

where $x(t) = [x_1^T(t), \ldots, x_N^T(t)]^T \in \mathbb{R}^{Nd}$, $\Phi = [\phi_1^T(t), \ldots, \phi_N^T(t)]^T \in \mathbb{R}^{Nd}$, $u(t) = [u_1^T(t), \ldots, u_M^T(t)]^T \in \mathbb{R}^{Md}$, and $\zeta(t) = [x(t)^T, z(t)^T, y(t)^T]^T \in \mathbb{R}^{(N+1+M)d}$.

Similar to the static spatial networks, the problem of simultaneous Facility Location and Path Optimization in Dynamic spatial networks ($d$-FLPO) has two fold objectives, (a) allocate facilities $y_j(t)$, $1 \leq j \leq M$ in the domain $\Omega$ and (b) design optimal transportation path from each node $n_i$ to the destination center $\delta$ such that the cost function (2.3) gets minimized at every time instant $t$.

A straightforward approach to solve the $d$-FLPO problem is to solve the $s$-FLPO problem at every time instant to determine the facility locations and the transportation paths. However, it is quite evident that such a method-

ology is computationally expensive. A specific shortcoming of this method is that it does not employ the past knowledge of facility locations and transportation routes to determine the solution at current time instant, which may potentially lead to big changes (jumps) in facility locations over a very small time intervals; and may not be practically achievable as shown in Section 2.7.

We propose a control-based framework to solve the $d$-FLPO problem that builds upon the solution of the $s$-FLPO problem obtained at the initial time instant $t_0$. In our framework, we use the free-energy function $F$ (2.12) as a Lyapunov candidate function for the dynamical system (2.27) and design the control for facility dynamics $\dot{y}(t) = u(t)$ such that $\dot{F} \leq 0 \ \forall \ t \geq 0$. Note that $F$ is a smooth approximation of $D$ in (2.7) which incorporates cost functions for both facility location and path optimization problems. Once the dynamics of the facilities are known, the time-varying transportation paths can be deduced from (2.11). The following theorem justifies the choice of free-energy $F$ as a Lyapunov function.

**Theorem 3.** *Let $F$ be the free-energy function (2.12) corresponding to the dynamical system (2.27) then*
*a) $F(\zeta) + \frac{1}{\beta}\log|\mathcal{G}| > 0, \ \forall \ \zeta = [x^T, y^T, d^T]^T \in \mathbb{R}^{(N+M+1)d}$ where $\mathcal{G} = \{(\gamma_1, \ldots, \gamma_M) : \gamma_k \in \Gamma_k \forall 1 \leq k \leq M\}$ is the set of all possible paths when $M$ facilities are allocated.*
*b) The derivative*

$$\frac{\partial F}{\partial \zeta} = \begin{bmatrix} \hat{P}_{\gamma_0} & -\hat{P}^0(\gamma_1, \gamma_0) & 0 \\ -\hat{P}^0(\gamma_1, \gamma_0)^T & 2\hat{A} - \hat{B} & -\hat{C} \\ 0 & -\hat{C}^T & I \end{bmatrix}, \tag{2.28}$$

*is a symmetric matrix, where $\hat{P}_{\gamma_0} = I_d \otimes P_{\gamma_0}$, $P_{\gamma_0} = diag(\{\rho_{\gamma_0}\})$, $\hat{P}_0(\gamma_1, \gamma_0) = I_d \otimes P_0(\gamma_1, \gamma_0)$, $P_0(\gamma_1, \gamma_0) = [p_{\gamma_0} p_0(\gamma_1|\gamma_0)]$ and $I_d$ is an identity matrix of size $d \times d$. Also note that $\dot{F}(t) = 2\zeta^T \frac{\partial F}{\partial \zeta}\dot{\zeta}$.*
*c) There is no dynamic control authority at the facility locations $y_c(t) = (2\hat{A} - \hat{B})^{-1}(\hat{X} + \hat{C})$ obtained in (2.13), i.e. $\frac{\partial \dot{F}}{\partial u} = 0$ at $y(t) = y_c(t)$.*

Please refer the Appendix A.1.4 for proof of the above theorem. The facilities are at the positions $y_c(t)$ (2.13) only when $\bar{y}(t) := y(t) - y_c(t) = 0$. We transform the coordinates $\zeta = [x^T, y^T, z^T]^T$ to $\bar{\zeta} = [x^T, \bar{y}^T, z^T]^T$, where

$\bar{y} = y - y_c$. In the new coordinates, the dynamics of the facility locations are given by

$$\dot{\bar{y}}(t) = \bar{u}(t) - (2\hat{A} - \hat{B})^{-1}\big[(2\dot{\hat{A}} - \dot{\hat{B}})y_c + \dot{\hat{P}}^0(\gamma_1, \gamma_0)^T x + \dot{\hat{C}}z\big], \tag{2.29}$$

where $\bar{u} = u - (2\hat{A} - \hat{B})^{-1}(\hat{P}^0(\gamma_1|\gamma_0)^T\Phi + \hat{C}\Psi)$ and

$$\dot{F} = (x^T\hat{P}_{\gamma_0} - y_c^T\hat{P}_0(\gamma_1|\gamma_0)^T)\Phi + [z^T - y_c^T\hat{C}]\psi + \bar{y}^T(2\hat{A} - \hat{B})\bar{u}(t). \tag{2.30}$$

We take advantage of the affine dependence of $\dot{F}$ on $\bar{u}(t)$ in (2.30) to determine the choice of $\bar{u}(t)$ such that $\dot{F} \leq 0$ [68], [69], [70]. More particularly, we choose control

$$\bar{u}(\bar{\zeta}) = -\left[K_0 + \frac{\alpha + \sqrt{|\alpha|^2 + (\bar{y}(2\hat{A} - \hat{B})\bar{y})^2}}{\bar{y}^T(2\hat{A} - \hat{B})\bar{y}}\right]\bar{y} \tag{2.31}$$

where $\bar{y} \neq 0$, $K_0 > 0$ and $\alpha = (x^T\hat{P}_{\gamma_0} - y_c^T\hat{P}_0(\gamma_1|\gamma_0)^T)\Phi + [z^T - y_c^T\hat{C}]\psi$. The following two theorems establish that the facility locations $y(t)$ converge asymptotically to $y_c(t)$ and the control effort (2.31) is bounded near $\bar{y} = 0$.

**Theorem 4.** *Asymptotic convergence: For the dynamical system (2.27) the choice of control $\bar{u}(\bar{\zeta})$ in (2.31) results in $\dot{F} \leq 0 \ \forall \ t \geq 0$ and $\bar{y}(t) \to 0$ as $t \to \infty$.*

**Theorem 5.** *Lipschitz continuity: If there exists a control $\hat{\bar{u}} : \mathbb{R}^{(N+M+1)d} \to \mathbb{R}^{Md}$ Lipschitz at $\bar{\zeta} = 0$ such that $\dot{F} \leq 0 \ \forall \ t \geq 0$ for $\bar{u} = \hat{\bar{u}}$, then the choice of control $\bar{u}$ in (2.31) is Lipschitz at $\bar{\zeta} = 0$. That is, $\exists \ \epsilon > 0$ and a constant $c_0$ such that $\|\bar{u}(\bar{\zeta})\| \leq c_o\|\bar{\zeta}\|$ for $\|\bar{\zeta}\| \leq \epsilon$*

Please refer to the Appendix A.1.4 for the proof of the above two theorems. *Remarks:* (i) The above control design methodology can be extended to the $d$-FLPO problems with additional constraints over the network topology and facilities. In fact, our simulations in Section 2.7 demonstrate the $d$-FLPO problem where the spatial network is partially connected. (ii) Theorem 5 emphasizes the non-conversativeness of our solution; i.e., if there exists a Lipschitz (bounded) solution such that $\dot{F} \leq 0$ then Theorem 5 implies that

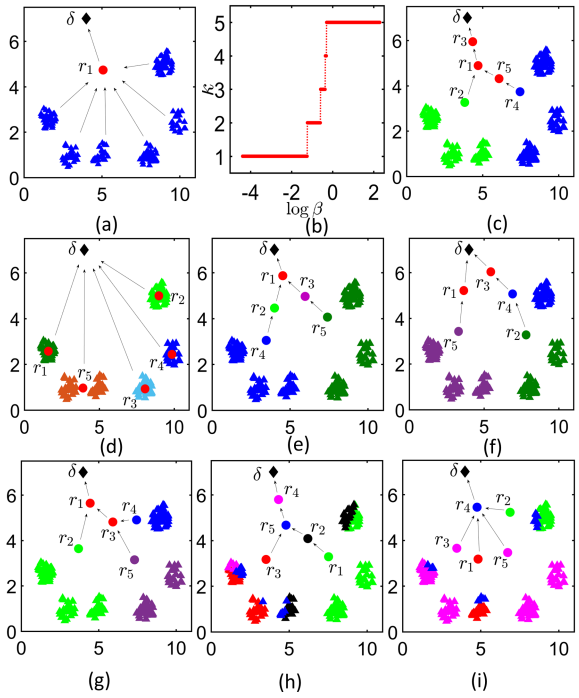


Figure 2.4: The nodes, facilities and the destination center are represented by triangles, circles and diamond, respectively. (a) At $\beta \approx 0$ all facilities are coincident. (b) Phase Transition Phenomenon. (c) Facility locations and paths at $\beta \to \infty$. The first facility to each node is colored identically to that node. The commodity hops to the subsequent facilities from the first facility as denoted by the arrows. (d) Two-step methodology for $s$-RARO. (e) Illustrates $s$-FLPO where $L_{\gamma_0} = 3\ \forall \gamma_0 \in \Gamma_0$. (f) Partially connected network - pairs $(f_1, f_2)$, $(f_4, f_1)$, $(f_2, f_3)$, $(f_4, f_5)$ and $(f_5, f_3)$ do not exist. (g) Facility capacity constraint $w_1 : w_2 : w_3 : w_4 : w_5 = 4 : 2 : 2 : 1 : 1$. (h) Facilities constrained as entry facilities in proportion 4:2:2.5:0.1:1.0 (i) Transportation link capacity constraints $\eta_{f_1 f_4} : \eta_{f_2 f_4} : \eta_{f_3, f_4} : \eta_{f_5, f_4} = 0.3 : 0.7 : 1 : 1$; all other communication links have zero capacities.

our proposed solution is also Lipschitz (bounded). (iii) For the purpose of simulation, we discretize time into $\Delta t$ intervals. At instants $\bar{y} \neq 0$ we determine the dynamics of $y(t)$ using $\bar{u}$ in (2.31). This results into $\dot{F} \leq 0$ at all such time instants. For the time instants when $\bar{y} = 0$ we already have the facilities at the locations $y_c$. At such instants $\dot{F}$ may be positive, negative or zero depending on the dynamics of the node and the destination center.

## 2.7 Simulation and Results

In this section we simulate our proposed algorithms for the $s$-FLPO and $d$-FLPO problems. We first illustrate the $s$-FLPO problem. For the purpose of simulations we randomly distribute 200 nodes around 6 randomly chosen

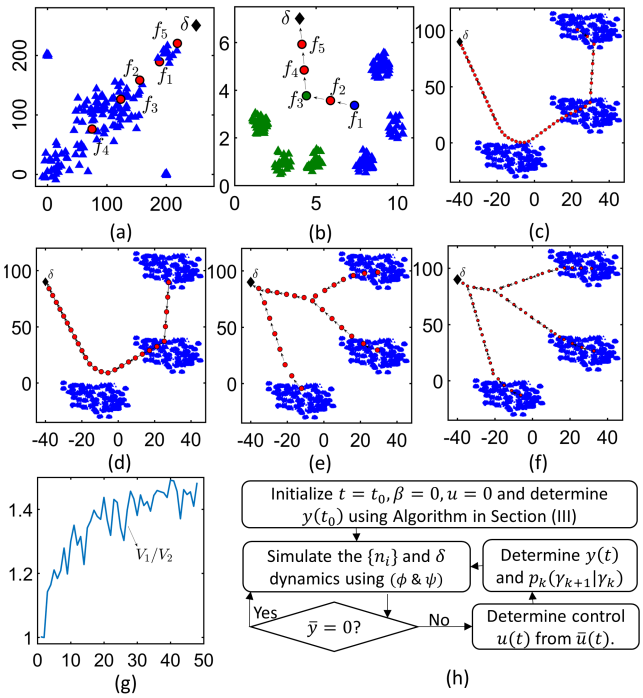


Figure 2.5: (a) For comparing computation times with [2]. Our algorithm - 1.85s, algorithm in [2] - 24.86s. (b) Solution using framework from [3]. (c)-(d) Large scale problem with $N = 17028$, $M = 27$ and $M = 50$ in (c) and (d), respectively. Solved using the algorithm in [3]. (e)-(f) Large scale problem with $N = 17028$, $M = 27$ and $M = 50$ in (e) and (f), respectively. Solved using the algorithm in the current work (g) Comparing the objective function value ($V_1/V_2$) at various number of facilities $M$ as given by algorithm in [3] ($V_1$) and our current work ($V_2$) for the previous large scale setting of nodes and destination. (h) Flowchart of the proposed algorithm in the $d$-FLPO problem.

points in an $11 \times 8$ square unit area. The location of the destination (marked as $\delta$) is randomly chosen to be at $(4, 7)$. For the purpose of illustration we assume the cost function $d_k(\cdot, \cdot)$ to be squared-euclidean distance. Consider the scenario where we allocate $M = 5$ facilities. As stated in section 2.4, at low value of $\beta$, all the facilities $\{f_j\}_{j=1}^{5}$ get allocated at the same spatial coordinates as shown in the Fig. 2.4(a), where the triangles denotes the nodes, circle denotes the facilities and the diamond denotes the destination. As the value of $\beta$ increases the number of distinct facility locations increases (Fig. 2.4(b)), and the final facility locations and transportation paths are obtained as $\beta$ becomes sufficiently large as shown in the Fig. 2.4(c). In the Fig. 2.4(c), a node of a particular color first sends its commodity packet to the facility of the similar color which then reaches the destination center via the path indicated in the figure. Observe that in Fig. 2.4(c) all the nodes $\gamma_0$ either opt for the 4-hop path $f_4 \rightarrow f_5 \rightarrow f_1 \rightarrow f_3 \rightarrow \delta$ or the

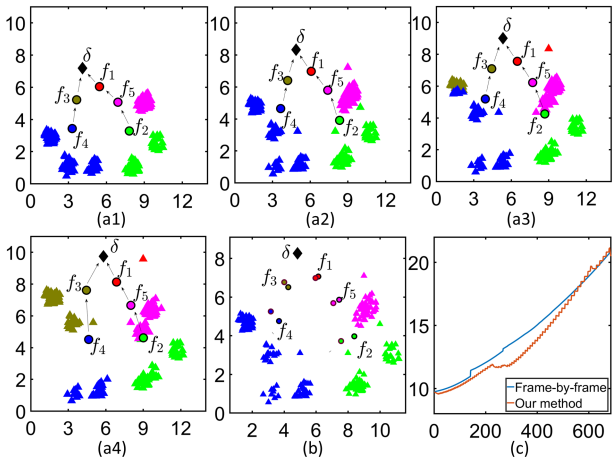


Figure 2.6: (a1)-(a4) Illustrates the solution to the $d$-FLPO problem. Observe the change in spatial coordinates of the nodes, destination center and the facilities. Also, observe the change in the color of the triangles from (a2) to (a3) and (a3) to (a4), indicating the change in their transportation paths. (b) Non-viable dynamics of the facility locations. Observe the considerable change in spatial location of $f_2$ and $f_4$ over a small interval of 0.03 seconds. (c) Comparing distortion from the two approaches.

3-hop path $f_2 \rightarrow f_1 \rightarrow f_3 \rightarrow \delta$. The total cost of transportation incurred is 12.58 units. Fig. 2.4(d) illustrates the solution to the same problem using the sequential approach illustrated in the Section 2.3 which incurs a total transportation cost of 34.16 units, 2.7 times the cost incurred in Fig. 2.4(c). This demonstrates that the sequential approach results into a solution with a much higher cost as compared to the simultaneous approach.

*Restricted Number of Hops:* Fig. 2.4(e) illustrates the solution to the $s$-FLPO problem when the maximum path length is restricted to 3 for all the nodes, i.e. $L_{\gamma_0} = 3 \; \forall \; \gamma_0 \in \Gamma_0$. Observe that now all the nodes opt for either $f_4 \rightarrow f_2 \rightarrow f_1 \rightarrow \delta$ or $f_5 \rightarrow f_3 \rightarrow f_1 \rightarrow \delta$ paths, both of which have path length of 3. The total cost in this case is 12.71 units, which is approximately 1% higher than the scenario in Fig. 2.4(c) where there is no restriction on the maximum path length. We can deduce the effectiveness of restricting hops to 3 which only leads to about 1% cost increase; that is marginal utility of adding more hops is only about 1%.

*Partially Connected Network:* Fig. 2.4(f) simulates a partially connected spatial network. For the purpose of simulation, we assume that the transportation links $(f_1, f_2)$, $(f_2, f_3)$, $(f_4, f_1)$ , $(f_1, f_3)$, $(f_4, f_5)$ and $(f_5, f_3)$ are absent. As shown in the Figure, the algorithm respects the constraint posed by the partially connected network and assigns the facility locations and

paths correspondingly. The total cost of transportation is 12.92 units. Note the difference with the facility location and transportation path assignments in a fully-connected network Fig. 2.4(c).

*Capacity Constraints:* Fig. 2.4(g) demonstrates the capacity constraints on various facilities. Here facility capacities are distributed as $w_1 : w_2 : w_3 : w_4 : w_5 = 4 : 2 : 2 : 1 : 1$. In the figure, the final facility allocation and path design is in such a way that $C(f_1) : C(f_2) : C(f_3) : C(f_4) : C(f_5) = 3.85 : 1.75 : 2.10 : 1 : 1.1$ which is approximately as given in the constraint. The slight mismatch in the values of the usage $C(f_j)$ and capacities $w_j$ could be because of numerical issues in MATLAB and we are currently looking into it. Similarly Fig. 2.4(h) demonstrates the scenario when the facilities are constrained in their capacity to act as an entry facility of a node in the proportion $4 : 2 : 2.5 : 0.5 : 1.0$. The final solution is such that a proportion $3.9 : 1.9 : 2.5 : 0.5 : 1.0$ is achieved for the facilities as the entry point for the nodes in the network. Note the color changes for the nodes in comparison to Fig. 2.4(c), (f) and (g). Fig. 2.4(i) illustrates the scenario when the capacities of the transportation links are known apriori. We assume that all the transportation links except $(f_1, f_3), (f_2, f_3), (f_3, f_4)$ and $(f_5, f_4)$ have zero capacities and we constrain that $\eta_{f_1f_4} : \eta_{f_2f_4} : \eta_{f_3f_4} : \eta_{f_5f_4} = 3 : 7 : 10 : 10$. Upon simulation the facilities are allocated and transportation paths are fixed in such a way that the $C_{f_1f_4} : C_{f_2f_4} : C_{f_3f_4} : C_{f_5f_4} = 3.3 : 6.3 : 9 : 9$, i.e. the solution given by the algorithm complies with the constraint on the communication link capacities. The total cost of communication is 15.8 units.

*Comparison with previous works:* Next we compare our proposed method with [2] and [3]. We begin with comparing the computation time of the algorithm presented in this paper with the one proposed in [2]. The computation time for the problem setting shown in Fig. 2.5(a) is 24.86 seconds, while the algorithm presented in this paper takes 1.85 seconds which is just 8% of the former. Note that the [2] uses MATLAB to run the algorithm on an Intel Core 2 Duo T5470 1.6 GHz processor with 2 GB RAM, while we used MATLAB to code and run our algorithm on i3 2.3 GHz processor with 2 GB RAM. We note that the improvement in the computation time comes from the scalability of the algorithm proposed in this paper, since the configuration of the two machines used are almost similar. Fig. 2.5(b) demonstrates the solution to the *s*-FLPO problem as given by the algorithm in [3]. Here the total cost of transportation comes out to be 14.93, which is 18% (or 2.35

units) more than the solution Fig. 2.4(c) given by the algorithm presented in this paper.

*Large Scale Problems:* Fig. 2.5(c)-(f) demonstrate a large scale *s*-FLPO problem setting with $N = 17028$ nodes as indicated by the blue triangles. In the Fig. 2.5(c) and (e) we allocate $M = 27$ facilities using the algorithm in [3] and our current work, respectively. Note the qualitative difference between the two solutions where the allocated facilities and the transportation paths resulting from the latter are more distributed in the domain as compared to [3]. The objective function value for the solution in Fig. 2.5(c) is 1323.07 units and for the Fig. 2.5(e) is 985.87 units; which is a improvement of 25% over the former. Similarly, in the Fig. 2.5(d) and (f) we allocated $M = 50$ facilities using the approach in [3] and our current algorithm, respectively. The objective value for the solution in Fig. 2.5(d) is 864.35 and for the solution in Fig. 2.5(e) is 582.70; approximately 33% lesser objective function value is obtained using our current framework. The above quantifies the fact that the framework developed in [3] results into suboptimal solutions as compared to our current framework when applied to a general *s*-FLPO problem. Also note that the framework presented in [2] is computationally intractable for both the above cases of $M = 27$ and $M = 50$ facilities as it requires $2.9 \times 10^{28}$ and $8.2 \times 10^{64}$ many decision variables, respectively; the corresponding memory requirements are unthinkable. The Fig. 2.5(g) compares the cost function value of the solutions obtained using [3] ($V_1$) and our current work ($V_2$) for various number of facilities $M$ allocated in the large scale setting of nodes and destination illustrated in Fig. 2.5(c)-(f). Note that the ratio $\eta = V_1/V_2 (> 1)$ increases with number of allocated facilities and reaches close to 1.5 (i.e. 50% increase in the cost function values in [3]) for values of $M \gtrapprox 40$ - clearly indicating sub-optimality of the framework [3] when applied to general *s*-FLPO problem setting.

*d-FLPO in partially connected network:* We consider the scenario of partially connected spatial network where the transportation links $(f_1, f_2)$, $(f_2, f_3)$, $(f_4, f_1)$, $(f_1, f_3)$, $(f_4, f_5)$ and $(f_5, f_3)$ are absent. The main steps of the algorithm are summarized in the Fig. 2.4(q). The sequence of images in Fig. 2.6(a1)-(a4) demonstrate the dynamics of the facilities and the transportation paths for randomly chosen dynamics of the nodes and the destination center. The node and destination center dynamics are simulated for a total duration of 20 seconds and the dynamics of the facilities and transportation

paths are determined using (2.31) after every time interval of $\Delta t = 0.03$ seconds. Observe the change in the *entry facility* of the nodes (marked by the change in their color) and their corresponding transportation path in Fig. 2.6(a1)-(a4).

The *frame-by-frame* approach takes approximately 700 times more computational time than our approach for the example considered above. Fig. 2.6(c) compares the distortion $D_0$ obtained from our control-based approach and the frame-by-frame solution. Apart from that, the simulations using frame-by-frame approach show sudden jumps in positions which may be impractical for scenarios with bounded velocities - for instance in Fig. 2.6(b), note the sudden jump in positions of facilities $f_2$ and $f_4$ in the span of two time instants that are only 0.03 seconds apart.

## 2.8 Analysis and Discussion

**A. Flexibility of the Framework:** The proposed framework is flexible to incorporate various additional constraints on the *s*-FLPO and *d*-FLPO problems in terms of the network topology, facilities, or the transportation paths as illustrated in Section 2.5. Our framework also generalizes to different choices of distance functions $d_k(\gamma_k, \gamma_{k+1})$ as against the squared euclidean function considered in this paper.

**B. Robustness Analysis:** The solutions obtained to *s*-FLPO and *d*-FLPO problems are sensitive to various attributes of the nodes and the destination center (such as spatial locations, dynamics and distance cost functions). This necessitates a study to classify such attributes, that affect the final solution, into various categories of importance. Our framework easily facilitates such a study through the free energy function $F$ which is a smooth approximation of the cost function $D$. For instance, the derivatives $\frac{\partial F}{\partial x_i}$, $\frac{\partial F}{\partial y_j}$ and $\frac{\partial F}{\partial d}$ measure the sensitivity of the final solution to the spatial location of the node $n_i$, facility $f_j$ and destination $\delta$ respectively.

**C. Uncertainty in Parameters:** In certain applications, instead of the exact information about various attributes of the nodes and the destination centre, a partial knowledge in terms of distributions of these attributes may be known. For instance, instead of the exact spatial location $x_i$ of the node $n_i$, distribution $p(x_i|n_i)$ for the spatial location is known. Our proposed frame-

work easily incorporates such uncertainties in parameter values. For example the above uncertainty in the spatial locations of the nodes will result into replacing $d(n_i, f_j)$ with $d'(n_i, f_j) = \sum_{x_i} p(x_i|n_i)d(n_i, f_j)$ and the remainder of the problem solution follows as in Section 2.3.

**D. Application to Parameterized Finite Horizon Markov Decision Processes (MDPs):** The stage-wise framework proposed in Figure 2.2 facilitates a viewpoint where in our MEP-based solution approach for the FLPO problem is easily extendable to the class of sequential decision making problems that are modelled as finite horizon MDPs with *parameterized* state space. In particular, consider the MDP given by $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, c, \mathcal{P}, H \rangle$ where $\mathcal{S}$ is the state space, $\zeta(s)$ for $s \in \mathcal{S}$ represents the unknown parameter (e.g. facility locations in FLPO), $\mathcal{A}$, $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, $\mathcal{P} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \{0, 1\}$, and $H$ respectively denote the set of actions, the cost function, the state transition probability, and maximum number of stages. The underlying FLPO-type objective is to $\min_{\{\zeta(s)\},\{\mu_t(s)\}} J := \sum_{s\in\mathcal{S}} \rho(s)J(s)$ where

$$J(s) = \sum_{\mathcal{X}} \bar{p}_\mu(\mathcal{X}|x_0 = s)\Big[\sum_{t=0}^{H} c(x_t, \mu_t(x_t))\Big], \tag{2.32}$$

$\rho(s)$ denotes the weight of each state $s \in \mathcal{S}$, $\mu_t(\cdot)$ is the policy under which the state $x_t = s \in \mathcal{S}$ is followed by $x_{t+1} = \mu_t(s) \in \mathcal{S}$, $\mathcal{X} := (x_0, x_1, \ldots, x_H)$ denotes a sequences of states and $\bar{p}_\mu(\cdot|s)$ is the distribution over the space of all possible sequences $\mathcal{X}$. Note the resemblance of the objective function here with the cost function $D$ in (2.5) of the FLPO problem where the unknown parameters $\zeta(s)$ are the facility locations $\{y_j\}$ and the policy $\mu_t(\cdot)$ is analogous to the association weights $\{p_k(\cdot|\cdot)\}$. Hence, the solution methodology detailed out in the Section 2.3 also solves the optimization problem posed by the parameterized finite horizon MDPs.

## 2.9 Summary

This chapter addresses the problem of simultaneous Facility Location and Path Optimization(FLPO) in static and dynamic spatial networks. This problem clearly falls into the category of para-SDM tasks, where allocating the facilities is analogous to determining the unknown parameters in para-

SDM and designing the optimal routes is equivalent to making sequential decisions. In our proposed methodology for the FLPO problem, we develop a *stage-wise viewpoint* that allows us to exploit the inherent *law of optimality* followed by the optimal routes; that facilitates in reducing the decision variable space from exponentially large to just polynomially large. We demonstrate the capabilities of our framework in incorporating several resource capacity, and network design constraints. To address the dynamically evolving networks, we design non-linear feedback control law that determines the facility dynamics and the time-varying routes under which asymptotic tracking of local optimal of the FLPO problem is achieved. We also demonstrate that the feedback control law is non-conservative, i.e., if there exists a Lipschitz control law that asymptotically tracks the local optimal of the FLPO problem then the proposed control law is also Lipschitz and bounded.

Interestingly, the stage-wise viewpoint of the FLPO developed in this chapter extends to the para-SDM tasks in finite-horizon, and thus, the solution methodology proposed in this paper easily generalizes to the problems that come under the later class. The work done here also forms a precursor to the infinite horizon MDPs and reinforcement learning (RL) problems in Chapters 3 and 4.

# CHAPTER 3

# MDPs, REINFORCEMENT LEARNING, AND PARAMETERIZED SDM PROBLEMS

## 3.1 Introduction

In the previous chapter we addressed the Facility Location and Path Optimization problem that generalizes to the para-SDM problems in finite horizon. In this chapter we deal with the sequential decision making tasks in infinite horizon. We start with developing an MEP-based method to address the Markov decision processes (that model non-parameterized SDMs), and extend it to the case of model-free Reinforcement Learning (RL). Thereafter, we built on the above framework to address the class of para-SDM optimization problems to determine the unknown parameters and the control policy that minimize the associated cost function; we address this problem in the case when the underlying system model is completely known, as well as not known.

Markov Decision Processes (MDPs) model sequential decision making problems which arise in many application areas such as robotics, automatic control, sensor networks, economics, and manufacturing. These models are characterized by the state-evolution dynamics $s_{t+1} = f(s_t, a_t)$, a control policy $\mu(a_t|s_t)$ that allocates an *action* $a_t$ from a control set to each state $s_t$, and a cost $c(s_t, a_t, s_{t+1})$ associated with the transition from $s_t$ to $s_{t+1}$. The goal in these applications is to determine the optimal control policy that results in a *path*, a sequence of actions and states, with minimum cumulative cost. There are many variants of this problem [71], where the dynamics can be defined over finite or infinite horizons; where the state-dynamics $f$ can be stochastic in which case expected cost functions are minimized; and reinforcement learning (RL) problems where the models for the state dynamics may be partially or completely unknown, and cost function is not known a priori, albeit the cost at each step is revealed at the end of each transition. Some of

the most common methodologies that address MDPs include dynamic programming, value and policy iterations [11], linear programming [12–14], and Q-learning [25].

In this chapter, we view MDPs and their variants as combinatorial optimization problems, and develop a framework based on maximum entropy principle (MEP) [5] to address them. MEP has proved successful in addressing a variety of combinatorial optimization problems such as facility location problems [6], combinatorial drug discovery [7], traveling salesman problem and its variants [6], image processing [8], graph and markov chain aggregation [1], and protein structure alignment [9]. MDPs, too, can be viewed as *combinatorial* optimization problems - due to the combinatorially large number of paths (sequence of consecutive states and actions) that it may take based on the control policy and its inherent stochasticity. In our MEP framework, we determine a probability distribution defined on *the space of paths* [72], such that (a) it is the *fairest* distribution - the one with the maximum Shannon Entropy $H$, and (b) it satisfies the constraint that the expected cumulative cost $J$ attains a prespecified feasible value $J_0$. The framework results in an iterative scheme, an *annealing* scheme, where probability distributions are improved upon by successively lowering the prespecified values $J_0$. In fact, the lagrange multiplier $\beta$ corresponding to the cost constraint ($J = J_0$) in the unconstrained lagrangian is increased from small values (near 0) to large values to effect annealing. Higher values of multipliers correspond to lower values of the expected cost. We show that as multiplier value increases, the corresponding probability distributions become more localized, finally converging to a deterministic policy.

This framework is applicable to all the classes of MDPs and its variants described above. Our MEP based approach inherits the flexibility of algorithms such as deterministic annealing [6] developed in the context of combinatorial resource allocation, which include adding capacity, communication, and dynamic constraints. The added advantage of our approach is that we can draw close parallels to existing algorithms for MDPs and RL (e.g. Q-Learning) – thus enabling us to exploit their algorithmic insights. Below we highlight main contributions and advantages of our approach.

*Exploration and Unbiased Policy:* In the context of model-free RL setting, the algorithms interact with the *environment* via *agents* and rely upon the instantaneous cost (or reward) generated by the environment to learn the

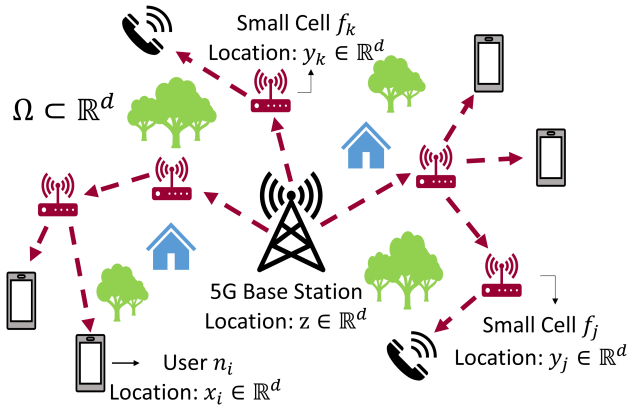


Figure 3.1: Illustrates the 5G Small Cell Network. The 5G Base Station $\delta$ is located at $z \in \mathbb{R}^d$ and the users $\{n_i\}$ are located at $\{x_i \in \mathbb{R}^d\}$. The objective is to determine the small cell $\{f_j\}$ location $\{y_j \in \mathbb{R}^d\}$ and the communication routes (control policy) from the Base Station $\delta$ to each user $\{n_i\}$ via the network of the small cells such that the total cost of communication gets minimized.

optimal policy. Some of the popular algorithms include Q-learning [25], Double Q-learning [26], Soft Q-learning (entropy regularized Q-learning) [27–33] in discrete state and action spaces, and Trust Region Policy Optimization (TRPO) [34], and Soft Actor Critic (SAC) [35] in continuous spaces. It is commonly known that for the above algorithms to perform well, all *relevant* states and actions should be explored. In fact, under the assumption that each state-action pair is visited multiple times during the learning process it is guaranteed that the above discrete space algorithms [25–28] will converge to the optimal policy. Thus, the adequate *exploration* of the state and action spaces becomes incumbent to the success of these algorithms in determining the optimal policy. Often the instantaneous cost is noisy [27] which hinders with the learning process and demands for an enhanced quality exploration.

In our MEP based approach, the Shannon Entropy of the probability distribution over the paths in the MDP explicitly characterizes the *exploration*. The framework results in a *distribution over the paths* that is as *unbiased* as possible under the given cost constraint. The corresponding stochastic policy is maximally noncommittal to any particular path in the MDP that achieves the constraint; this results in better (unbiased) exploration. The policy starts from being entirely explorative, when multiplier value is small ($\beta \approx 0$), and becomes increasingly *exploitative* as the multiplier value increases.

*Parameterized SDM tasks and RL:* These class of optimization problems are not even necessarily MDPs which contributes significantly to their inher-

ent complexities. However, we model them in a specific way to retain the Markov property without any loss of generality, thereby making these problems tractable. Scenarios such as self organizing networks [73], 5G small cell network design [74, 75], supply chain networks, and last mile delivery problems [76] pose a *parameterized* MDP with a two-fold objective of determining simultaneously (a) the optimal control policy for the underlying stochastic process, and (b) the unknown parameters that the state and action variables depend upon such that the cumulative cost is minimized. The latter objective is akin to facility location problem [54, 77, 78], that is shown to be NP-hard [54], and where the associated cost function (non-convex) is riddled with multiple poor local minima. For instance, Figure 3.1 illustrates a 5G small cell network, where the objective is to simultaneously determine the locations of the small cells $\{f_j\}$ and design the communication paths (control policy) between the user nodes $\{n_i\}$ and base station $\delta$ via network of small cells $\{f_j\}$. Here, the state space $\mathcal{S}$ of the underlying stochastic process (para-SDM) is parameterized by unknown locations $\{y_j\}$ of small cells $\{f_j\}$.

*Algebraic structure and Sensitivity Analysis:* In our framework, maximization of Shannon entropy of the distribution over the paths under constraint on the cost function value results in an unconstrained Lagrangian - the *free-energy* function. This function is a *smooth* approximation of the cumulative cost function of the MDP, which enables the use of calculus. We exploit this distinctive feature of our framework to determine the unknown state and action parameters in case of parameterized SDM tasks and perform sensitivity analysis for various problem parameters. Also, the framework easily accommodates stochastic models that describe uncertainties in the instantaneous cost and parameter values.

*Algorithmic Guarantees and Innovation:* For the classes of MDPs that we consider, our MEP-based framework results into non-trivial derivations of the recursive Bellman equation for the associated Lagrangian. We show that these Bellman operators are contraction maps and use their several properties to guarantee the convergence to the optimal policy and as well as to a local minima in the case of para-SDM problems.

In the context of model-free RL, we provide comparison of our proposed framework with the benchmark algorithms Q, Double Q, and entropy regularized G-learning [27] (also referred to as Soft Q-learning). Our algorithms converge at a faster rate (as fast as 1.5 times) than the benchmark algorithms

across various values of discount factor, and even in the case of noisy environments. In the context of para-SDM, we address the small-cell network design problem in 5G communication. Here the parameters are the unknown locations of the small-cells and the control policy determines the routing of the communication packet. Upon comparison with the sequential method of first determining the unknown parameters (small cell locations) and then the control policy (equivalently, the communication paths), we show that our algorithms result into costs that are as low as 65% of the former. The efficacy of our algorithms can be assessed from the fact that the solutions in the model-based and model-free cases are nearly the same (differ by $< 2\%$). We also demonstrate sensitivity analysis, benefits of annealing (11% lesser network design cost) and considering entropy of distribution over the paths (5% lesser network design cost) in our simulations on param-SDM problems. For the ease of reading, please refer to the comprehensive list of symbols presented in the Section **??** of the supplementary material.

## Related Work

*Entropy Regularization:* Some of the previous works in RL literature [27–33, 36, 37] either use entropy as a regularization term $(-\log \mu(a_t|s_t))$ [27, 28] to the instantaneous cost function $c(s_t, a_t, s_{t+1})$ or maximize the entropy $(-\sum_a \mu(a|s) \log \mu(a|s))$ [29–31] associated *only* to the stochastic policy under constraints on the cost $J$. This results into benefits such as better exploration, overcoming the effect of noise $w_t$ in the instantaneous cost $c_t$ and obtaining faster convergence. However, the resulting stochastic policy and soft-max approximation of the value function $J$ is not in compliance with the Maximum Entropy Principle applied to the distribution over the paths of MDP. Thus, the resulting stochastic policy is biased in its exploration *over the paths of the MDP*. Our simulations demonstrate the benefit of *unbiased* exploration (in our framework) in terms of faster convergence and better performance in noisy environment in comparison to the entropy regularized benchmark algorithm.

*Parameterized SDM problems:* The existing solution approaches [11–14] can be extended to the para-SDM problems by discretizing the parameter domain and viewing the parameters as part of the decision making problem

in the MDPs. However, the resulting problem is not necessarily an MDP as every transition from one state to another is dependent on the path (and the parameter values) taken till the current state; thus, losing the Markov property. Other related work in MDPs that determines more than just the control policy are case specific; for instance, [18] presents action-based parameterization of state space with application to service rate control in closed Jackson networks, and [19–24] incorporate parameterized actions that is applicable in the domain of RoboCup soccer where at each step the agent must select both the discrete action it wishes to execute as well as continuously valued parameters required by that action. On the other hand, the class of para-SDM problems that we address in this chapter predominantly originate in network based applications that involves simultaneous routing and resource allocations and pose additional challenges of non-convexity and NP-hardness. We address these problems in both the scenarios, where the underlying model is known as well as unknown.

## 3.2 MDPs with Finite Shannon Entropy

### 3.2.1 Problem Formulation

We consider an infinite horizon discounted MDP that comprises of a *cost-free termination* state $\delta$. We formally define this MDP as a tuple $\langle \mathcal{S}, \mathcal{A}, c, p, \gamma \rangle$ where $\mathcal{S} = \{s_1, \ldots, s_N = \delta\}$, $\mathcal{A} = \{a_1, \ldots, a_M\}$, and $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ respectively denote the state space, action space, and cost function; $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ is the state transition probability function and $0 < \gamma \leq 1$ is the discounting factor. A control policy $\mu : \mathcal{A} \times \mathcal{S} \to \{0, 1\}$ determines the action taken at each state $s \in \mathcal{S}$, where $\mu(a|s) = 1$ implies that action $a \in \mathcal{A}$ is taken when the system is in the state $s \in \mathcal{S}$ and $\mu(a|s) = 0$ indicates otherwise. For every initial state $x_0 = s$, the MDP induces a stochastic process, whose realization is a *path* $\omega$ - an infinite sequence of actions and states, that is

$$\omega = (u_0, x_1, u_1, x_2, u_2, \ldots, x_T, u_T, x_{T+1}, \ldots), \tag{3.1}$$

where $u_t \in \mathcal{A}$, $x_t \in \mathcal{S}$ for all $t \in \mathbb{Z}_{\geq 0}$ and $x_t = \delta$ for all $t \geq k$ if and when the system reaches the termination state $\delta \in \mathcal{S}$ in $k$ steps. The objective is to determine the optimal policy $\mu^*$ that minimizes the state value function

$$J^{\mu}(s) = \mathbb{E}_{p_\mu}\Big[\sum_{t=0}^{\infty} \gamma^t c(x_t, u_t, x_{t+1}) \big| x_0 = s\Big], \quad \forall\ s \in \mathcal{S} \tag{3.2}$$

where the expectation is with respect to the probability distribution $p_\mu(\cdot|s) : \omega \to [0, 1]$ on the space of all possible paths $\omega \in \Omega := \{(u_t, x_{t+1})_{t \in \mathbb{Z}_{\geq 0}} : u_t \in \mathcal{A}, x_t \in \mathcal{S}\}$. For practical problem formulations, here $\sum_{s,a} p(s'|s, a) > 0$ for all $s' \in \mathcal{S}$; that is, we assume that there exist atleast one $(s, a)$ pair such that the probability of reaching $s'$ under the dynamics $p$ is non-zero. In order to ensure that the system reaches the cost-free termination state in finite steps and the optimal state value function $J^{\mu}(s)$ is finite, we make the following assumption throughout this section.

**Assumption 1.** There exists atleast one deterministic proper policy $\bar{\mu}(a|s) \in \{0, 1\}\ \forall\ a \in \mathcal{A}, s \in \mathcal{S}$ such that $\min_{s \in \mathcal{S}} p_{\bar{\mu}}(x_{|\mathcal{S}|} = \delta | x_0 = s) > 0$. In other words, under the policy $\bar{\mu}$ there is a non-zero probability to reach the cost-free termination state $\delta$, when starting from any state $s$.

In our MEP-based framework we consider the following set of stochastic policies $\mu$

$$\Gamma := \{\pi : 0 < \pi(a|s) < 1\ \forall\ a \in \mathcal{A}, s \in \mathcal{S}\}, \tag{3.3}$$

and the following lemma ensures that under the Assumption 1 all the policies $\mu \in \Gamma$ are *proper*; that is,

**Lemma 1.** *For any policy $\mu \in \Gamma$ as defined in (3.3), $\min_{s \in \mathcal{S}} p_\mu(x_{|\mathcal{S}|} = \delta | x_0 = s) > 0$, i.e., under each policy $\mu \in \Gamma$ the probability to reach the termination state $\delta$ in $|\mathcal{S}| = N$ steps beginning from any $s \in \mathcal{S}$, is strictly positive.*

*Proof.* Please refer to the Appendix A.2.1.

We use the Maximum Entropy Principle to determine the policy $\mu \in \Gamma$ such that the Shannon Entropy of the corresponding distribution $p_\mu$ is maximized and the state value function $J^{\mu}(s)$ attains a specified value $J_0$. More

specifically, we pose the following optimization problem

$$\begin{aligned} \max_{\{p_\mu(\cdot|s)\}:\mu\in\Gamma} \quad & H^\mu(s) = -\sum_{\omega\in\Omega} p_\mu(\omega|s)\log p_\mu(\omega|s) \\ \text{subject to} \quad & J^\mu(s) = J_0. \end{aligned} \tag{3.4}$$

*Well posedness:* For the class of proper policy $\mu \in \Gamma$ the maximum entropy $H^\mu(s)\ \forall\ s \in \mathcal{S}$ is finite as shown in [79, 80]. In short, the existence of a cost-free termination state $\delta$ and a non-zero probability to reach it from any state $s \in \mathcal{S}$ ensures that the maximum entropy is finite. Please refer to the Theorem 1 in [79] or Proposition 2 in [80] for further details.

*Remark* 1. Though the optimization problem in (3.4) considers the stochastic policies $\mu \in \Gamma$, our algorithms presented in the later sections are designed in such that the resulting stochastic policy asymptotically converges to a deterministic policy.

### 3.2.2 Problem Solution

The probability $p_\mu(\omega|s)$ of taking the path $\omega$ in (4.1) can be determined from the underlying policy $\mu$ by exploiting the Markov property that dissociates $p_\mu(\omega|s)$ in terms of the policy $\mu$ and the state transition probability $p$ as

$$p_\mu(\omega|x_0) = \prod_{t=0}^{\infty} \mu(u_t|x_t)p(x_{t+1}|x_t,u_t). \tag{3.5}$$

Thus, in our framework we prudently work with the policy $\mu$ which is defined over finite action and state spaces as against the distribution $p_\mu(\omega|s)$ defined over infinitely many paths $\omega \in \Omega$. The Lagrangian corresponding to the above optimization problem in (3.4) is $V_\beta^\mu(s) = J^\mu(s) - \frac{1}{\beta}H^\mu(s) =$

$$\mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t c^{u_t}_{x_t x_{t+1}} + \frac{1}{\beta}(\log \mu_{u_t|x_t} + \log p^{u_t}_{x_t x_{t+1}})\big|x_0 = s\big], \tag{3.6}$$

where $\beta$ is the Lagrange parameter. Here we have not included the constant value $J_0$ in the cost Lagrangian $V_\beta^\mu(s)$ for simplicity. We refer to the above Lagrangian $V_\beta^\mu(s)$ (3.6) as the *free-energy* function and $\frac{1}{\beta}$ as *temperature* due to their close analogies with statistical physics (where free energy is enthalpy

(E) minus the temperature times entropy (TH)). To determine the optimal policy $\mu^*_\beta$ that minimizes the Lagrangian $V^\mu_\beta(s)$ in (3.6), we first derive the Bellman equation corresponding to $V^\mu_\beta(s)$.

**Theorem 6.** *The free-energy function $V^\mu_\beta(s)$ in (3.6) satisfies the following recursive Bellman equation*

$$V^\mu_\beta(s) = \sum_{\substack{a\in\mathcal{A}\\ s'\in\mathcal{S}}} \mu_{a|s} p^a_{ss'} \big(\bar{c}^a_{ss'} + \frac{\gamma}{\beta}\log\mu_{a|s} + \gamma V^\mu_\beta(s')\big) + c_0(s), \tag{3.7}$$

*where $\mu_{a|s} = \mu(a|s)$, $p^a_{ss'} = p(s'|s,a)$, $\bar{c}^a_{ss'} = c(s,a,s') + \frac{\gamma}{\beta}\log p(s'|s,a)$ for simplicity in notation, and function $c_0(s)$ does not depend on the policy $\mu$.*

*Proof.* Please refer to the Appendix A.2.1 for details. It must be noted that this derivation shows and exploits the algebraic structure $\sum_{s'} p^a_{ss'} H^\mu(s') = \sum_{s'} p^a_{ss'}\log p^a_{ss'} + \log\mu_{a|s} + \lambda_s + 1$ as detailed in Lemma 3 in the appendix.

Now the optimal policy satisfies $\frac{\partial V^\mu_\beta(s)}{\partial \mu(a|s)} = 0$, which results into the Gibbs distribution

$$\mu^*_\beta(a|s) = \frac{\exp\big\{-(\beta/\gamma)\Lambda_\beta(s,a)\big\}}{\sum_{a'\in\mathcal{A}}\exp\big\{-(\beta/\gamma)\Lambda_\beta(s,a')\big\}}, \text{ where} \tag{3.8}$$

$$\Lambda_\beta(s,a) = \sum_{s'\in\mathcal{S}} p^a_{ss'}\big(\bar{c}^a_{ss'} + \gamma V^*_\beta(s')\big), \tag{3.9}$$

is the state-action value function, $p^a_{ss'} = p(s'|s,a)$, $c^a_{ss'} = c(s,a,s')$, $\bar{c}^a_{ss'} = c^a_{ss'} + \frac{\gamma}{\beta}\log p^a_{ss'}$ and $V^*_\beta (= V^{\mu^*_\beta}_\beta)$ is the value function corresponding to the policy $\mu^*_\beta$. To avoid notional clutter we use the above notations wherever it is clear from the context. Substituting the policy $\mu^*_\beta$ in (3.8) back into the Bellman equation (3.7) we obtain the *implicit* equation

$$V^*_\beta(s) = -\frac{\gamma}{\beta}\log\Big(\sum_{a\in\mathcal{A}}\exp\Big\{-\frac{\beta}{\gamma}\Lambda_\beta(s,a)\Big\}\Big). \tag{3.10}$$

Without loss of generality, we ignore the term $c_0(s)$ in (3.7) since it does not affect the policy as seen from equation (3.8). To solve for the state-action value function $\Lambda_\beta(s,a)$ and free-energy function $V^*_\beta(s)$ we substitute the expression of $V^*_\beta(s)$ in (3.10) into the expression of $\Lambda_\beta(s,a)$ in (3.9) to

obtain the implicit equation $\Lambda_\beta(s,a) =: [T\Lambda_\beta](s,a)$, where

$$[T\Lambda_\beta](s,a) = \sum_{s'\in\mathcal{S}} p^a_{ss'}\big(c^a_{ss'} + \frac{\gamma}{\beta}\log p^a_{ss'}\big) \\ - \frac{\gamma^2}{\beta}\sum_{s'\in\mathcal{S}} p^a_{ss'}\log\sum_{a'\in\mathcal{A}}\exp\Big\{-\frac{\beta}{\gamma}\Lambda_\beta(s',a')\Big\}. \tag{3.11}$$

To solve the above implicit equation, we show that the map $T$ in (3.11) is a contraction map and therefore $\Lambda_\beta$ can be obtained using fixed point iterations, which guarantee converging to the unique fixed point. Consequently, the global minimum $V^*_\beta$ in (3.10) and the optimal policy $\mu^*_\beta$ in (3.8) can be obtained.

**Theorem 7.** *The map $[T\Lambda_\beta](s,a)$ as defined in (3.11) is a contraction mapping with respect to a weighted maximum norm, i.e. $\exists$ a vector $\xi = (\xi_s) \in \mathbb{R}^{|\mathcal{S}|}$ with $\xi_s > 0\ \forall\ s \in \mathcal{S}$ and a scalar $\alpha < 1$ such that*

$$\|T\Lambda_\beta - T\Lambda'_\beta\|_\xi \le \alpha\|\Lambda_\beta - \Lambda'_\beta\|_\xi \tag{3.12}$$

*where* $\|\Lambda_\beta\|_\xi = \max\limits_{s\in\mathcal{S},a\in\mathcal{A}} \frac{|\Lambda_\beta(s,a)|}{\xi_s}$

*Proof.* Please refer to Appendix A.2.3 for details.

*Remark* 2. It is known from the sensitivity analysis [10] that the value of the Lagrange parameter $\beta$ in (3.6), (3.10) is inversely proportional to the constant $J_0$ in (3.4). It is clear from (3.6) that for small values of $\beta \approx 0$ (equivalently large $J_0$), we are mainly maximizing the Shannon Entropy $H^\mu(s)$ and the resultant policy in (3.8) naturally encourages exploration along the paths of the MDP. As $\beta$ increases ($J_0$ decreases), more and more weight is given to the state value function $J^\mu(s)$ in (3.6) and the policy in (3.8) goes from being exploratory to being exploitative. As $\beta \to \infty$ the exploration is completely eliminated and we converge to a deterministic policy $\to \mu^*$ that minimizes $J^\mu(s)$ in (3.2).

*Remark* 3. We briefly draw readers' attention to the value function $Y(s) = \mathbb{E}[\sum_{t=0}^{\infty}\gamma^t(c^{u_t}_{x_t x_{t+1}} + (1/\beta)\log\mu_{u_t|x_t})]$ considered in the entropy regularized methods [27]. Note that in $Y(s)$ the discounting $\gamma^t$ is multiplied to both the cost term $c^{u_t}_{x_t x_{t+1}}$ as well as the entropy term $(1/\beta)\log\mu_{u_t|x_t}$. However, in our MEP-based method, the Lagrangian $V^\mu_\beta(s)$ in (3.6) comprises of discounting $\gamma^t$ *only* over the cost term $c^{u_t}_{x_t x_{t+1}}$ and not on the entropy terms

$(1/\beta)\log \mu_{u_t|x_t}$ and $(1/\beta)\log p^{u_t}_{x_t x_{t+1}}$. Therefore, the policy in [27] does not satisfy MEP applied over the distribution $p_\mu$; consequently their exploration along the paths is not as unbiased as our algorithm (as demonstrated in Section 3.5).

### 3.2.3 Model-free Reinforcement Learning Problems

In these problems, the cost function $c(s, a, s')$ and the state-transition probability $p(s'|s, a)$ are not known apriori; however, at each discrete time instant $t$ the *agent* takes an action $u_t$ under a policy $\mu$ and the *environment* (underlying stochastic process) results into an instantaneous cost $c^{u_t}_{x_t x_{t+1}}$ and the subsequently moves to the state $x_{t+1} \sim p(\cdot|x_t, u_t)$. Motivated by the iterative updates of Q-learning algorithm [11] we consider the following stochastic updates in our Algorithm 2 to learn the state-action value function in our methodology

$$\begin{aligned}\Psi_{t+1}(x_t, u_t) &= (1 - \nu_t(x_t, u_t))\Psi_t(x_t, u_t) + \\ &\nu_t(x_t, u_t)\Big[c^{u_t}_{x_t x_{t+1}} - \frac{\gamma^2}{\beta}\log \sum_{a' \in \mathcal{A}} \exp\Big\{\frac{-\beta}{\gamma}\Psi_t(x_{t+1}, a')\Big\}\Big],\end{aligned} \tag{3.13}$$

with the stepsize parameter $\nu_t(x_t, u_t) \in (0, 1]$, and show that under appropriate conditions on $\nu_t$ (as illustrated shortly), $\Psi_t$ will converge to the fixed point $\bar{\Lambda}^*_\beta$ of the implicit equation

$$\begin{aligned}\bar{\Lambda}_\beta(s, a) &= \sum_{s' \in \mathcal{S}} p^a_{ss'}\Big(c^a_{ss'} - \frac{\gamma^2}{\beta}\log \sum_{a'} \exp\big(\frac{-\beta}{\gamma}\bar{\Lambda}_\beta(s', a')\big)\Big) \\ &=: [\bar{T}\bar{\Lambda}_\beta](s, a).\end{aligned} \tag{3.14}$$

The above equation comprises of a minor change from the equation $\Lambda_\beta(s, a) = [T\Lambda_\beta](s, a)$ in (3.11). The latter has an additional term $\frac{\gamma}{\beta}\sum_{s'} p^a_{ss'}\log p^a_{ss'}$ which makes it difficult to *learn* its fixed point $\Lambda^*_\beta$ in the absence of the state transition probability $p^a_{ss'}$ itself. Since in this work we do not attempt to determine (or learn) either the distribution $p^a_{ss'}$ (as in [81]) from the interactions of the agent with the environment, we work with the approximate state-action value function $\bar{\Lambda}_\beta$ in (3.14) where $\bar{\Lambda}_\beta \to \Lambda_\beta$ for large $\beta$ values (since $\frac{\gamma}{\beta}(\sum_{s'} p^a_{ss'}\log p^a_{ss'}) \to 0$ as $\beta \to \infty$). The following proposition eluci-

dates the conditions under which the updates $\Psi_t$ in (3.13) converge to the fixed point $\bar{\Lambda}^*_\beta$.

**Proposition 1.** *Consider the class of MDPs illustrated in Section 3.2.1. Given that*

$$\sum_{t=0}^{\infty} \nu_t(s,a) = \infty, \sum_{t=0}^{\infty} \nu_t^2(s,a) < \infty \ \forall \ s \in \mathcal{S}, a \in \mathcal{A},$$

*the update $\Psi_t(s,a)$ in (3.13) converges to the fixed point $\bar{\Lambda}^*_\beta$ of the map $\bar{T}\bar{\Lambda}_\beta \to \bar{\Lambda}_\beta$ in (3.14) with probability 1.*

*Proof.* Please refer to the Appendix A.2.5.

**Algorithm 2** Model-free Reinforcement Learning

Input: $N$, $\nu_t(\cdot,\cdot)$, $\sigma$;
Output: $\mu^*$, $\bar{\Lambda}^*$
Initialize: $t = 0$, $\Psi_0 = 0$, $\mu_0(a|s) = 1/|\mathcal{A}|$.
**for** *episode* $= 1$ to $N$ **do**
    $\beta = \sigma \times$ *epsiode*; reset environment at state $x_t$
    **while** True **do**
        sample $u_t \sim \mu_t(\cdot|x_t)$; obtain cost $c_t$ and $x_{t+1}$
        update $\Psi_t(x_t, u_t)$, $\mu_{t+1}(u_t|x_t)$ in (3.13) and (3.8)
        break if $x_{t+1} = \delta$; $t \leftarrow t + 1$
    **end while**
**end for**

*Remark* 4. Note that the *stochasticity* of the optimal policy $\mu^*_\beta(a|s)$ (3.8) depends on $\gamma$ value which allows it to incorporate for the effect of the discount factor on its exploration strategy. More precisely, in the case of large discount factors the time window $T$, in which instantaneous costs $\gamma^t c(s_t, a_t, s_{t+1})$ is considerable (i.e., $\gamma^t c^{a_t}_{s_t s_{t+1}} > \epsilon \ \forall \ t \leq T$), is large and thus, the stochastic policy (3.8) performs higher exploration along the paths. On the other hand, for small discount factors this time window $T$ is relatively smaller and thus, the stochastic policy (3.8) inherently performs lesser exploration. As illustrated in the simulations, this characteristic of the policy in (3.8) results into even faster convergence rates in comparison to benchmark algorithms as the discount factor $\gamma$ decreases.

## 3.3 MDPs with Infinite Shannon Entropy

Here we consider the MDPs where the Shannon Entropy $H^\mu(s)$ of the distribution $\{p_\mu(\omega|s)\}$ over the paths $\omega \in \Omega$ is not necessarily finite (for instance, due to absence of absorption state). To ensure the finiteness of the objective in (3.4) we consider the *discounted* Shannon Entropy [82, 83]

$$H_d^\mu(s) = -\mathbb{E}\Big[\sum_{t=0}^{\infty} \alpha^t (\log \mu_{u_t|x_t} + \log p^{u_t}_{x_t x_{t+1}}) \big| x_0 = s\Big] \tag{3.15}$$

with a discount factor of $\alpha \in (0,1)$ which we chose to be independent of the discount factor $\gamma$ in the value function $J^\mu(s)$. The free-energy function (or, the Lagrangian) resulting from the optimization problem in (3.4) with the alternate objective function $H_d^\mu(s)$ in (3.15) is given by

$$V^\mu_{\beta,I}(s) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \hat{c}^{u_t}_{x_t x_{t+1}} + \frac{\alpha^t}{\beta} \log \mu(u_t|x_t) \big| x_0 = s\Big], \tag{3.16}$$

where $\hat{c}^{u_t}_{x_t x_{t+1}} = c^{u_t}_{x_t x_{t+1}} + \frac{\gamma^t}{\beta \alpha^t} \log p^{u_t}_{x_t x_{t+1}}$, and the subscript $I$ stands for 'infinite entropy' case. Note that the free-energy functions (3.6) and (3.16) differ only with regards to the discount factor $\alpha$ and thus, our solution methodology in this section is similar to the one in Section 3.2.2.

**Theorem 8.** *The free-energy function $V^\mu_{\beta,I}(s)$ in (3.16) satisfies the recursive Bellman equation*

$$V^\mu_{\beta,I}(s) = \sum_{a,s'} \mu_{a|s} p^a_{ss'} (\check{c}^a_{ss'} + \frac{\gamma}{\alpha\beta} \log \mu_{a|s} + \gamma V^\mu_{\beta,I}(s')) + \bar{c}_0(s) \tag{3.17}$$

*where $\check{c}^a_{ss'} = c^a_{ss'} + \frac{\gamma}{\alpha\beta} \log p^a_{ss'}$ and $\bar{c}_0(s)$ is independent of the policy.*

*Proof.* Please see Appendix A.2.4. The above derivation shows and exploits algebraic structure $\alpha \sum_{s'} p^a_{ss'} H^\mu_d(s') = \sum_{s'} p^a_{ss'} \log p^a_{ss'} + \log \alpha\mu(a|s) + \lambda_s$ (Lemma 5).

The optimal policy satisfies $\frac{\partial V^\mu_{\beta,I}(s)}{\partial \mu(a|s)} = 0$, which results into the Gibbs distribution

$$\mu^*_{\beta,I}(a|s) = \frac{\exp\big\{-\frac{\beta\alpha}{\gamma}\Phi_\beta(s,a)\big\}}{\sum_{a' \in \mathcal{A}} \exp\big\{-\frac{\beta\alpha}{\gamma}\Phi_\beta(s,a')\big\}}, \text{ where} \tag{3.18}$$

$$\Phi_\beta(s,a) = \sum_{s' \in \mathcal{S}} p^a_{ss'}(\breve{c}^a_{ss'} + \gamma V^*_{\beta,I}(s')), \tag{3.19}$$

is the corresponding state-action value function. Substituting the $\mu^*_{\beta,I}$ in (3.18) in the Bellman equation (3.17) results into the following optimal free-energy function $V^*_{\beta,I}(s)(:= V^{\mu^*_{\beta,I}}_{\beta,I}(s))$.

$$V^*_{\beta,I}(s) = -\frac{\gamma}{\alpha\beta} \log \sum_{a' \in \mathcal{A}} \exp\Big(\frac{-\alpha\beta}{\gamma}\Phi_\beta(s,a')\Big) \tag{3.20}$$

*Remark* 5. The subsequent steps to learn the optimal policy $\mu^*_{\beta,I}$ in (3.18) are similar to the steps demonstrated in the Section 3.2.3. We forego the similar analysis here.

*Remark* 6. When $\alpha = \gamma$ the policy $\mu^*_{\beta,I}$ in (3.18), state-action value function $\Phi_\beta$ in (3.19) and the free-energy function $V^*_{\beta,I}$ in (3.20) corresponds to the similar expressions that are obtained in the Entropy regularized methods [27]. However, in this chapter we do not require that $\alpha = \gamma$. On the other hand, we propose that $\alpha$ should take up large values. In fact, our simulations in the Section 3.5 demonstrate better convergence rates that are obtained when $\gamma < \alpha = (1-\epsilon)$ as compared to the scenarios where $\gamma = \alpha < 1$.

## 3.4 Parameterized SDM Problems

### 3.4.1 Problem Formulation

As illustrated in Section 3.1, many application areas such as battlefield surveillance, self organizing networks and 5G small cell networks (see Figure 3.1) pose a parmaterized SDM problem that requires simultaneously determining the sequential decisions and the unknown parameters that minimize the associated cost. We borrow much of the mathematical structure of the MDPs illustrated in Section 3.2 to formally describe the para-SDM problems. Consider the para-SDM task described as a tuple $\langle \mathcal{S}, \mathcal{A}, c, p, \gamma, \zeta, \eta \rangle$ where $\mathcal{S} = \{s_1, \ldots, s_N = \delta\}$, $\mathcal{A} = \{a_1, \ldots, a_M\}$, and $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ respectively denote the state space, action space, and cost function; $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0,1]$ is the state transition probability function and $0 < \gamma \leq 1$ is the discounting factor; $\zeta_1 = \{\zeta_s \in \mathbb{R}^{d_\zeta} : s \in \mathcal{S}_1 \subseteq \mathcal{S}\}$ and $\eta_1 = \{\eta_a \in \mathbb{R}^{d_\eta} : a \in \mathcal{A}_1 \subseteq \mathcal{A}\}$ de-

note the *known* state and action parameters, and $\zeta_2 = \{\zeta_s : s \in \mathcal{S}_2 = \mathcal{S} \backslash \mathcal{S}_1\}$ and $\eta_2 = \{\eta_a : a \in \mathcal{A}_2 = \mathcal{A} \backslash \mathcal{A}_1\}$ denote the *unknown* state and action parameters. A control policy $\mu : \mathcal{A} \times \mathcal{S} \to \{0, 1\}$ determines the action taken at each state $s \in \mathcal{S}$, where $\mu(a|s) = 1$ implies that action $a \in \mathcal{A}$ is taken when the system is in the state $s \in \mathcal{S}$ and $\mu(a|s) = 0$ indicates otherwise. For every initial state $x_0 = s$ and parameter values $\zeta := \zeta_1 \sqcup \zeta_2$, $\eta := \eta_1 \sqcup \eta_2$, the para-SDM induces a stochastic process, whose realization is a *path* $\omega$ - an infinite sequence of actions and states, that is

$$\omega = (u_0, x_1, u_1, x_2, u_2, \ldots, x_T, u_T, x_{T+1}, \ldots), \tag{3.21}$$

where $u_t \in \mathcal{A}$, $x_t \in \mathcal{S}$ for all $t \in \mathbb{Z}_{\geq 0}$ and $x_t = \delta$ for all $t \geq k$ if and when the system reaches the termination state $\delta \in \mathcal{S}$ in $k$ steps; the corresponding cost incurred by the stochastic process is given by

$$J^{\mu}_{\zeta\eta}(s) = \mathbb{E}_{p_\mu}\Big[ \sum_{t=0}^{\infty} \gamma^t c\big(x_t(\zeta), u_t(\eta), x_{t+1}(\zeta)\big) \Big| x_0 = s \Big]. \tag{3.22}$$

Here, $x_t(\zeta)$ denotes the state $x_t \in \mathcal{S}$ with the associated parameter $\zeta_{x_t}$ and $u_t(\eta)$ denotes the action $u_t \in \mathcal{A}$ with the associated action parameter value $\eta_{u_t}$. As in Section 3.2.1, we assume that the parameterized SDM tasks exhibit atleast one deterministic proper policy (Assumption 1) to ensure the finiteness of the associated value function $J^{\mu}_{\zeta\eta}(s)$ and the Shannon Entropy $H^{\mu}(s)$ for all $\mu \in \Gamma$. We further assume that the state-transition probability $\{p^a_{ss'}\}$ is independent of the state and action parameters $\zeta, \eta$.

### 3.4.2 Problem Solution

This problem was solved in Section 3.2.2, where the states and actions were not parameterized, or equivalently can be viewed as if parameters $\zeta$, $\eta$ were known and fixed. For the parameterized SDM case, we apply the same solution methodology, which results in the same optimal policy $\mu^*_{\beta,\zeta\eta}$ as in (3.8) as well as the corresponding free-energy function $V^*_{\beta,\zeta\eta}(s)$ in (3.10) except that now they are characterized by the parameters $\zeta$, $\eta$. To determine the

optimal (local) parameters $\zeta$, $\eta$, we set

$$\sum_{s' \in \mathcal{S}} \frac{\partial V^*_{\beta,\zeta\eta}(s')}{\partial \zeta_s} = 0 \forall\ s \text{ and } \sum_{s' \in \mathcal{S}} \frac{\partial V^*_{\beta,\zeta\eta}(s')}{\partial \eta_a} = 0 \forall\ a, \tag{3.23}$$

which we implement by using the gradient descent steps

$$\zeta_s^+ = \zeta_s - \eta \sum_{s' \in \mathcal{S}} G^\beta_{\zeta_s}(s'),\ \eta_a^+ = \eta_a - \bar{\eta} \sum_{s' \in \mathcal{S}} G^\beta_{\eta_a}(s'). \tag{3.24}$$

Here $G^\beta_{\zeta_s}(s') := \partial V^*_{\beta,\zeta\eta}(s')/\partial \zeta_s$ and $G^\beta_{\eta_a}(s') := \partial V^*_{\beta,\zeta\eta}(s')/\partial \eta_a$. The derivatives $G^\beta_{\zeta_s}$ and $G^\beta_{\eta_a}$ are assumed to be bounded (see constraint (b)-(c) in the Proposition 2). We compute these derivatives as $G^\beta_{\zeta_s}(s') = \sum_{a'} \mu_{a'|s'} K^\beta_{\zeta_s}(s', a')$ and $G^\beta_{\eta_a}(s') = \sum_{a'} \mu_{a'|s'} L^\beta_{\eta_a}(s', a')\ \forall\ s' \in \mathcal{S}$ where $K^\beta_{\zeta_s}(s', a')$ and $L^\beta_{\eta_a}(s', a')$ are the fixed points of their corresponding recursive Bellman equations $K^\beta_{\zeta_s}(s', a') = [T_1 K^\beta_{\zeta_s}](s, a)$ and $L^\beta_{\eta_a}(s', a') = [T_2 L^\beta_{\eta_a}](s', a')$ where

$$\begin{aligned}
[T_1 K^\beta_{\zeta_s}](s', a') &= \sum_{s''} p^{a'}_{s's''} \Big[ \frac{\partial c^{a'}_{s's''}}{\partial \zeta_s} + \gamma G^\beta_{\zeta_s}(s'') \Big], \\
[T_2 L^\beta_{\eta_a}](s', a') &= \sum_{s''} p^{a'}_{s's''} \Big[ \frac{\partial c^{a'}_{s's''}}{\partial \eta_a} + \gamma G^\beta_{\eta_a}(s'') \Big].
\end{aligned} \tag{3.25}$$

Note that in the above equations we have suppressed the dependence of the instantaneous cost function $c^{a'}_{s's''}$ on the parameters $\zeta$ and $\eta$ to avoid notational clutter.

**Theorem 9.** *The operators $[T_1 K^\beta_{\zeta_s}](s', a')$ and $[T_2 L^\beta_{\eta_a}](s', a')$ defined in (3.25) are contraction maps with respect to a weighted maximum norm $\|\cdot\|_\xi$, where $\|X\|_\xi = \max_{s',a'} \frac{X(s',a')}{\xi_{s'}}$ and $\xi \in \mathbb{R}^{|\mathcal{S}|}$ is a vector of positive components $\xi_s$.*

*Proof.* Please refer the Appendix A.2.5 for details.

As previously stated in Section 3.1, the state value function $J^\mu_{\zeta\eta}(\cdot)$ in (4.2) is generally non-convex function of the parameters $\zeta$, $\eta$ and riddled with multiple poor local minima with the resulting optimization problem being possibly NP-hard [54]. In our proposed algorithm for parameterized SDM problems we anneal $\beta$ from $\beta_{\min}$ to $\beta_{\max}$, similar to our approach for MDPs in Section 3.2.2, where the solution from the current $\beta$ iteration is used to initialize the subsequent $\beta$ iteration. However, in addition to facilitating a steady transition from an exploratory policy to an exploitative policy (that is

beneficial in the model-free RL setting), annealing facilitates a gradual homotopy from the convex negative Shannon entropy function to the non-convex state value function $J^{\mu}_{\zeta\eta}$ which prevents our algorithm from getting stuck at a poor local minimum. The underlying idea of our heuristic is to track the optimal as the initial convex function deforms to the actual non-convex cost function. Additionally, minimizing the Lagrangian $V^*_\beta(s)$ at $\beta = \beta_{\min} \approx 0$ determines the global minimum thereby making our algorithm insensitive to initialization. In Algorithm 3 we illustrate the steps to determine the policy and parameter values in a parameterized SDM setting.

**Algorithm 3** Parameterized SDM (model known)

Input: $\beta_{\min}$, $\beta_{\max}$, $\tau$;
Output: $\mu^*$, $\zeta$ and $\eta$.
Initialize: $\beta = \beta_{\min}$, $\mu_{a|s} = \frac{1}{|\mathcal{A}|}$, and $\zeta, \eta, G^{\beta}_{\zeta}, G^{\beta}_{\eta}$, $K^{\beta}_{\zeta}, L^{\beta}_{\eta}\Lambda_{\beta}$ to 0
**while** $\beta \leq \beta_{\max}$ **do**
    **while** True **do**
        **while** until convergence **do**
            update $\Lambda_\beta$,$\mu_\beta$,$G^{\beta}_{\zeta_s}$,$G^{\beta}_{\eta_a}$ in (3.9), (3.8) and (3.25)
        **end while**
        update $\zeta, \eta$ using gradient descent in (3.24)
        **if** $\|G_{\zeta_s}\| < \epsilon$, $\|G_{\eta_a}\| < \epsilon$ **then** break
    **end while**
    $\beta \leftarrow \tau\beta$
**end while**

### 3.4.3 Model-free learning in parameterized SDM

In many applications, formulated as parameterized SDM, the explicit knowledge of the cost function $c^a_{ss'}$, its dependence on the parameters $\zeta$, $\eta$, and the state-transition probabilities $\{p^a_{ss'}\}$ are not known. However, for each action $a$ the environment results into an instantaneous cost based on its current $x_t$, next state $x_{t+1}$ and the parameter $\zeta$, $\eta$ values which can subsequently be used to simultaneously learn the policy $\mu^*_{\beta,\zeta\eta}$ and the unknown state and action parameters $\zeta$, $\eta$ via stochastic iterative updates. At each $\beta$ iteration in our learning algorithm, we employ the stochastic iterative updates in (3.13) to determine the optimal policy $\mu^*_{\beta,\zeta\eta}$ for given $\zeta$, $\eta$ values and subsequently

employ the stochastic iterative updates

$$K_{\zeta_s}^{t+1}(x_t, u_t) = (1 - \nu_t(x_t, u_t))K_{\zeta_s}^{t}(x_t, u_t) + \nu_t(x_t, u_t)\Big[\frac{\partial c_{x_t x_{t+1}}^{u_t}}{\partial \zeta_s} + \gamma G_{\zeta_s}^{t}(x_{t+1})\Big], \tag{3.26}$$

where $G_{\zeta_s}^{t}(x_{t+1}) = \sum_a \mu_{a|x_{t+1}} K_{\zeta_s}^{t}(x_{t+1}, a)$ to learn the derivative $G_{\zeta_s}^{\beta*}(\cdot)$. Similar updates are used to learn $G_{\eta_a}^{\beta*}(\cdot)$. The parameter values $\zeta$, $\eta$ are then updated using the gradient descent step in (3.24). The following proposition formalizes the convergence of the updates in (3.26) to the fixed point $G_{\zeta_s}^{\beta*}$.

**Proposition 2.** *For the class of parameterized MDPs considered in Section 3.4.1 given that*

1. $\sum_{t=0}^{\infty} \nu_t(s, a) = \infty$, $\sum_{t=0}^{\infty} \nu_t^2(s, a) < \infty \ \forall s \in \mathcal{S}$, $a \in \mathcal{A}$,
2. $\exists\ B > 0$ *such that* $\left|\frac{\partial c(s', a', s'')}{\partial \zeta_s}\right| \leq B \ \forall\ s, s', a', s''$,
3. $\exists\ C > 0$ *such that* $\left|\frac{\partial c(s', a', s'')}{\partial \eta_a}\right| \leq C \ \forall\ a, s', a', s''$,

*the updates in (3.26) converge to the unique fixed point $G_{\zeta_s}^{\beta*}(s')$ of the map $T_1 : G_{\zeta_s} \to G_{\zeta_s}$ in (3.25).*

*Proof.* Please refer to the Appendix A.2.5 for details.

*Algorithmic Details:* Please refer to the Algorithm 4 for a complete implementation. Unlike the scenario in Section 3.2.3 where the agent acts upon the environment by taking an action $u_t \in \mathcal{A}$ and learns *only* the policy $\mu^*$, here the agent interacts with the environment by (a) taking an action $u_t \in \mathcal{A}$ and *also* providing (b) estimated parameter $\zeta$, $\eta$ values to the environment; subsequently, the environment results into an instantaneous cost and the next state. In our Algorithm 4, we first learn the policy $\mu_\beta^*$ at a given value of the parameters $\zeta$, $\eta$ using the iterations (3.13) and then learn the fixed points $G_{\zeta_s}^{\beta*}$, $G_{\zeta_a}^{\beta*}$ using the iterations in (3.26) to update the parameters $\zeta$, $\eta$ using (3.24). Note that the iterations (3.26) require the derivatives $\partial c(s', a', s'')/\partial \zeta_s$ and $\partial c(s', a', s'')/\partial \eta_a$ which we determine using the instantaneous costs resulting from two $\epsilon$-*distinct environments* and finite difference method. Here the $\epsilon$-distinct environments essentially represent the same underlying stochastic process but are distinct only in one of the parameter values. However, if two $\epsilon$-distinct environments are not feasible one can

**Algorithm 4** Model-free learning in para-SDM

Input: $\beta_{\min}$, $\beta_{\max}$, $\tau$, $T$, $\nu_t$;
Output: $\mu^*$, $\zeta$, $\eta$
Initialize: $\beta = \beta_{\min}$, $\mu_t = \frac{1}{|\mathcal{A}|}$, and $\zeta, \eta, G_\zeta^\beta, G_\eta^\beta, K_\zeta^\beta, L_\eta^\beta, \bar{\Lambda}_\beta$ to 0.
**while** $\beta \leq \beta_{\max}$ **do**
    Use Algorithm 2 to obtain $\mu^*_{\beta,\zeta\eta}$ at given $\zeta$, $\eta$, $\beta$.
    Consider *env1*($\zeta$,$\eta$), *env2*($\zeta', \eta'$); set $\zeta' = \zeta$, $\eta' = \eta$
    **while** $\{\zeta_s\}$, $\{\eta_a\}$ converge **do**
        **for** $\forall s \in \mathcal{S}$ **do**
            **for** *episode* $= 1$ to $T$ **do**
                reset *env1, env2* at state $x_t$,
                **while** True **do**
                    sample action $u_t \sim \mu^*(\cdot|x_t)$.
                    *env1:* obtain $c_t$, $x_{t+1}$.
                    *env2:* set $\zeta_s' = \zeta_s + \Delta\zeta_s$, get $c_t'$, $x_{t+1}$.
                    update $G_{\zeta_s}^{t+1}(x_t)$ with $\frac{\partial c_{x_t}^{u_t} x_{t+1}}{\partial \zeta_s} \approx \frac{c_t' - c_t}{\Delta \zeta_s}$.
                    break if $x_{t+1} = \delta$; $t \leftarrow t + 1$.
                **end while**
            **end for**
        **end for**
        Use same approach as above to learn $G_{\eta_a}^\beta$
        Update $\{\zeta_s\}$, $\{\eta_a\}$ via gradient descent in (3.24).
    **end while**
    $\beta \leftarrow \tau\beta$
**end while**

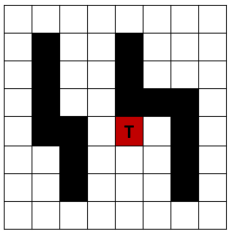


Figure 3.2: A Gridworld environment. State space $\mathcal{S} = \{0, \ldots, 64\}$ - where black grids are invalid states, $\mathcal{A} = \{0, \ldots, 8\}$ - actions correspond to vertical, horizontal, diagonal, and zero movements. Cost (reward) of every step (action) is 1 units. Every step is followed by a (approximate) cumulative probability of 0.3 to slip to neighbouring states (see details in Section 3.5). We consider two variants (a) Finite Entropy - cell **T** denotes terminal state (b) Infinite Entropy - No terminal state.

work with a single environment where the algorithm stores the instantaneous costs and the corresponding parameter values upon each interaction with the environment which can later be used to compute the derivatives using finite difference methods.

*Remark* 7. Parameterized SDM problems with infinite Shannon entropy $H^{\mu}$ can be analogously addressed using above methodology.

## 3.5 Simulations

We broadly classify our simulations into two categories. Firstly, in the model-free RL setting we demonstrate our Algorithm 2 to determine the control policy $\mu^*$ for the finite and infinite Shannon entropy variants of the Gridworld environment in Figure 3.2. Each cell in the Gridworld denotes a state. The cells colored black are invalid states. An agent can choose to move vertically, horizontal, diagonally or stay at the current cell. Each action is followed by a probability to slip in the neighbouring states (probability of 0.05 to slip in each of the vertical and horizontal directions, and probability of 0.025 to slip in each of the diagonal directions - cumulative $p(slip) \approx 0.3$ ). For the finite entropy case - each step incurs a unit cost. The process terminates when the agent reaches the terminal state **T**. For the infinite entropy case - each step incurs a unit reward. Secondly, in the parameterized SDM setting we demonstrate our Algorithms 3 (model-based) and 4 (model-free) in designing a 5G small cell network. This involves simultaneously determining the locations of the small cells in the network as well as the optimal routing path

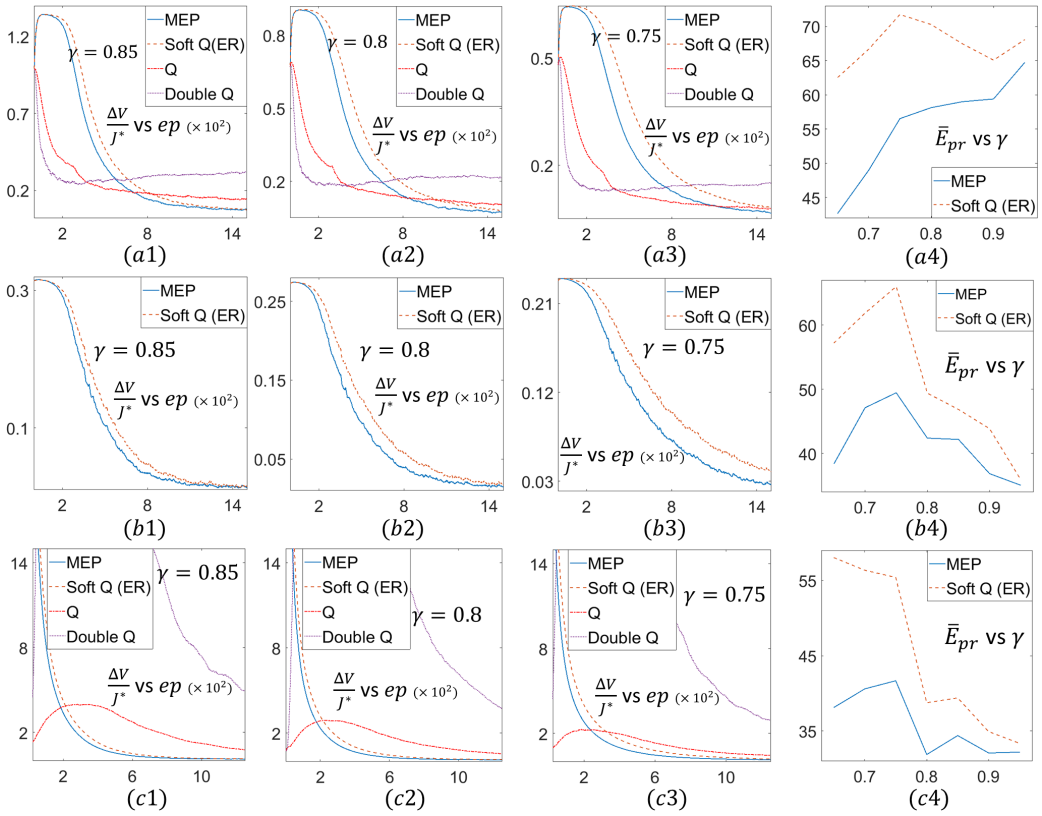


Figure 3.3: Performance of MEP-based algorithm: Illustrations on Gridworld Environment in Figure 3.2. (a1)-(a3) Finite Entropy variant: Illustrates faster convergence of Algorithm 2 (MEP) at different $\gamma$ values. (a4) Demonstrates faster rates of convergence of Algorithm 2 (MEP) for $\gamma$ values ranging from 0.65 to 0.95. (b1)-(b3) *Infinite Entropy Variant* : Demonstrates faster convergence of Algorithm 2 (MEP) to $J^*$. (b4) Illustrates the consistent faster convergence rates of MEP with $\gamma$ ranging from 0.65 to 0.95. (c1)-(c4) Finite Entropy Version (with added Gaussian noise) : Similar observations as in (a1)-(a4) with significantly higher instability with Double Q.

of the communication packets from the base station to the users (see Figure 3.4 for illustration).

We compare our MEP-based Algorithm 2 with the benchmark algorithms Entropy Regularized (ER) G-learning (also referred to as Soft Q-learning) [27], Q-learning [25] and Double Q-learning [26]. Note that our choice of the benchmark algorithm G-learning (or, entropy regularized Soft Q) presented in [27]) is based on its commonality to our framework as discussed in the Section 3.2.2, and the choice of algorithms Q-learning and Double Q-learning is based on their widespread utility in the literature. Also, note that the work done in [27] already establishes the efficacy of the G-learning algorithm over the following algorithms in literature Q-learning, Double Q-learning, $\Psi$-learning [84], Speedy Q-learning [85], and the consistent Bellman Operator $\mathcal{T}_C$ of [86]. Below we highlight features and advantages of our MEP-based Algorithm 2.

*Faster Convergence to Optimal $J^*$:* Figures 3.3(a1)-(a3) (finite entropy variant of Gridworld) and Figures 3.3 (b1)-(b3) (infinite entropy variant of Gridworld) illustrate the faster convergence of our MEP-based Algorithm 2 for different discount factor $\gamma$ values. Here, at each episode the percentage error $\Delta V/J^*$ between the value function $V^{\mu}_{\beta}$ corresponding to *learned* policy $\mu = \mu(ep)$ in the episode $ep$, and the optimal value function $J^*$ is given by

$$\frac{\Delta V(ep)}{J^*} = \frac{1}{N}\sum_{i=1}^{N}\sum_{s\in\mathcal{S}}\frac{\left|V^{\mu(ep)}_{\beta,i}(s) - J^*(s)\right|}{J^*(s)}, \tag{3.27}$$

where $N$ denotes the total experimental runs and $i$ indexes the value function $V^{\mu}_{\beta,i}$ for each run. As observed in Figures 3.3(a1)-(a3), and Figures 3.3(b1)-(b3), our Algorithm 2 converges even faster as the discount factor $\gamma$ decreases. We characterize the faster convergence rates also in terms of the convergence time - more precisely the *percentage* $\bar{E}_{pr}$ of total episodes taken for the learning error $\Delta V/J^*$ to reach within 5% of the best (see Figures 3.3(a4) and 3.3(b4)). As is observed in the figures, the performance of our (MEP-based) algorithm in comparison to entropy regularized G learning is better across all values (0.65 to 0.95) of discount factor $\gamma$. Note that the performance of Algorithm 2 gets even better with decreasing $\gamma$ values where the smaller discount factor values occur in instances such as the context of recommendation systems [87], teaching RL-agents using human-generated

rewards [88, 89].

*Robustness to noise in data:* Figures 3.3(c1)-(c4) demonstrate robustness to noisy environments; here the instantaneous cost $c(s, a, s')$ in the finite horizon variant of Gridworld is noisy. For the purpose of simulations, we add Gaussian noise $\mathcal{N}(0, \sigma^2)$ with $\sigma = 1$ for vertical and horizontal actions, and $\sigma = 0.5$ for diagonal movements. Here, at each episode we compare the percentage error $\Delta V/J^*$ in the *learned* value functions $V_\beta$ (corresponding to the state-action value estimate in (3.13)) of the respective algorithms. Similar to our observations and conclusions in Figures 3.3(a1)-(a3) and Figures 3.3(b1)-(b3) we see faster convergence of our MEP-based algorithm over the benchmark algorithms in Figures 3.3(c1)-(c3) in the case of noisy environment. Also, Figure 3.3(c4) demonstrates that across all discount factor values (0.65 to 0.95), Algorithm 2 converges faster than the entropy regularized Soft Q learning.

*Simultaneously determining the unknown parameters and policy in Parameterized MDPs:* We design the 5G Small Cell Network (see Figure 3.1) both when the underlying model ($c^a_{ss'}$ and $p^a_{ss'}$) is known (using Algorithm 3) and as well as unknown (using Algorithm 4). In our simulations we randomly distribute 46 user nodes $\{n_i\}$ at $\{x_i\}$ and the base station $\delta$ at $z$ in the domain $\Omega \subset \mathbb{R}^2$ as shown in Figure 3.4(a). The objective is to determine the locations $\{y_j\}_{j=1}^5$ (parameters) of the small cells $\{f_j\}_{j=1}^5$ and determine the corresponding communication routes (policy). Here, the underlying state space $\mathcal{S} = \{n_1, \ldots, n_{46}, f_1, \ldots, f_5\}$ where the locations $y_1, \ldots, y_5$ of the small cells are the unknown parameters $\{\zeta_s\}$ of the MDP, the action space is $\mathcal{A} = \{f_1, \ldots, f_5\}$, and the cost function $c(s, a, s') = \|\rho(s) - \rho(s')\|_2^2$ where $\rho(\cdot)$ denotes the spatial location of the respective states. The objective is to simultaneously determine the parameters (unknown small cell locations) and the control policy (communication routes in the 5G network). We consider two cases where (a) $p^a_{ss'}$ is deterministic, i.e. an action $a$ at the state $s$ results into $s' = a$ with probability 1, and (b) $p^a_{ss'}$ is probabilistic such that action $a$ at the state $s$ results into $s' = a$ with probability 0.9 or to the state $s' = f_1$ with probability 0.1. Additionally, due to absence of any prior work in literature on network design problems modeled as parameterized SDM, we compare our results only with the solution resulting from a straightforward sequential methodology (as shown in Figure 3.4(a)) where we first partition the user nodes into 5 distinct clusters to allocate a small cells in each cluster,

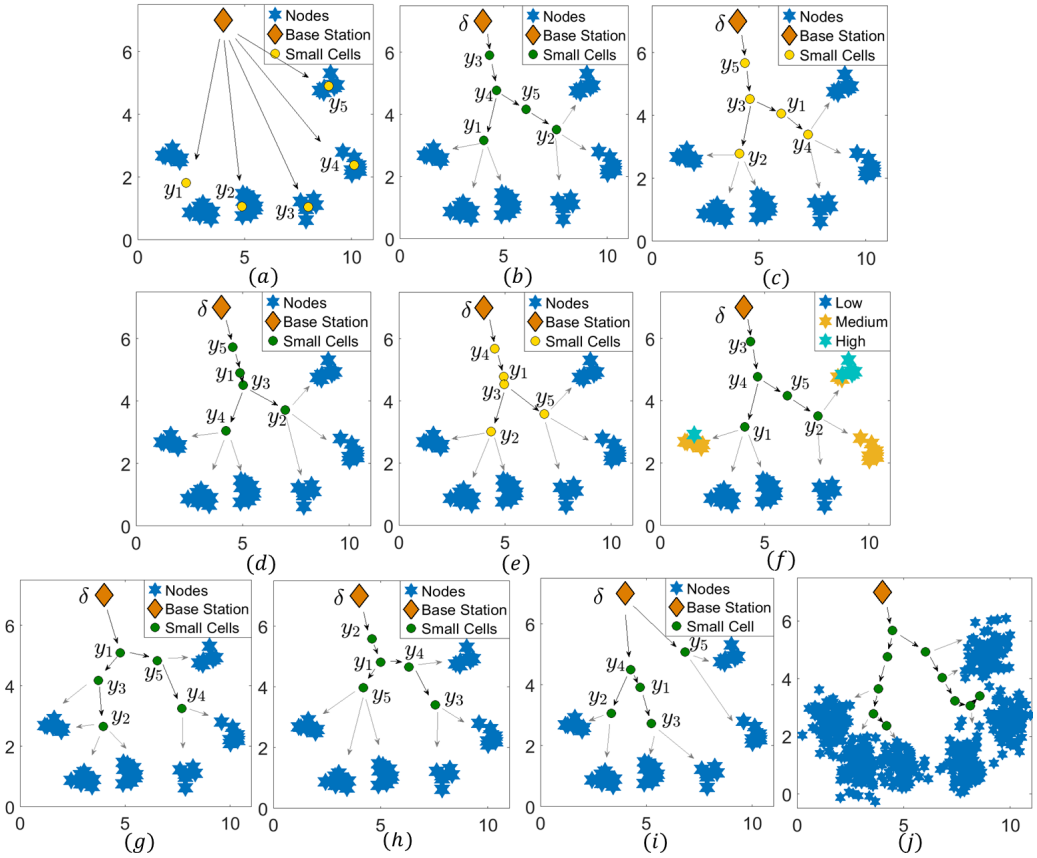


Figure 3.4: Parameterized MDPs and RL - Design of 5G small cell network. State space $\mathcal{S} = \{\{n_i\}, \{f_j\}, \delta\}$ comprises of the user nodes $\{n_i\}$, small cells $\{f_j\}$, and base station $\delta$. The unknown parameters $\zeta_s$ denote the locations $\{y_j\}$ of the small cells. Action space comprises of the small-cells $\mathcal{A} = \{f_j\}$. Based on our modelling of the network there are no unknown action parameters $\{\eta_a\}$/ (a) Illustrates small cell locations $\{y_j\}$ and communication routes determined using a straightforward sequential methodology. (b)-(c) demonstrate small cells at $\{y_j\}$ and communication routes (as illustrated by arrows) resulting from policy obtained from Algorithm 3 (model-based) and Algorithm 4 (model-free), respectively when the $p^a_{ss'}$ is deterministic. (d)-(e) solutions obtained using Algorithm 3 and Algorithm 4, respectively when $p^a_{ss'}$ is probabilistic. (f) sensitivity analysis of the solutions with respect to user node locations $\{x_i\}$. (g)-(h) Network design obtained when considering entropy of the distribution over the control actions and paths, respectively. (i) Network design obtained without annealing in Algorithm 3. (j) Simulation on a larger dataset (user base increased by more than 10 times).

and then determine optimal communication routes in the network.

*Deterministic $p(s, a, s')$:* Figure 3.4(b) illustrates the allocation of small cells and the corresponding communication routes (resulting from optimal policy $\mu^*$) as determined by the Algorithm 3. Here, the network is designed to minimize the cumulative cost of communication from each user node and small cell. As denoted in the Figure, the route $\delta \rightarrow y_3 \rightarrow y_4 \rightarrow y_1 \rightarrow n_i$ carries the communication packet from the base station $\delta$ to the respective user nodes $n_i$ as indicated by the grey arrow from $y_1$. The cost incurred here is approximately 180% lesser than that in Figure 3.4(a) clearly indicating the advantage obtained from simultaneously determining the parameters and policy over a sequential methodology. In the model-free setting where the functions $c(s, a)$, $p^a_{ss'}$, and the locations $\{x_i\}$ of the user nodes $\{n_i\}$ are not known, we employ our Algorithm 4 to determine the small cell locations $\{y_j\}_{j=1}^5$ and as well as the optimal policy $\{\mu^*(a|s)\}$ as demonstrated in the Figure 3.4(c). It is evident from Figures 3.4(b) and (c) that the solutions obtained when the model is completely known and unknown are approximately same. In fact, the solutions obtained differ only by 1.9% in terms of the total cost $\sum_{s\in\mathcal{S}} J^{\mu}_{\zeta\eta}(s)$ (4.2) incurred, clearly indicating the efficacy of our model-free learning Algorithm 4.

*Probabilistic $p(s, a, s')$:* Figure 3.4(d) illustrates the solution as obtained by our Algorithm 3 when the underlying model ($c(s, a)$, $p^a_{ss'}$, and $\{x_i\}$) is known. As before, here the network is designed to minimize the cumulative cost of communication from each user node and small cell. The cost associated to the network design is approximately 127% lesser than in Figure 3.4(a). Figure 3.4(e) illustrates the solution as obtained by the Algorithm 4 for the model-free case ($c(s, a)$, $p^a_{ss'}$, and $\{x_i\}$ are unknown). Similar to the above scenario, the solutions obtained for this case using the Algorithms 3 and 4 are also approximately the same and differ only by 0.3% in terms of the total cost $\sum_s J^{\mu}_{\zeta\eta}(s)$ incurred; thereby, substantiating the efficacy of our proposed model-free learning Algorithm 4.

*Sensitivity Analysis:* Our algorithms enable categorizing the user nodes $\{n_i\}$ in Figure 3.4(b) into the categories of (i) low, (ii) medium, and (iii) high sensitiveness such that the final solution is least susceptible to the user nodes in (i) and most susceptible to the nodes in (iii). Note that the above sensitivity analysis requires to compute the derivative $\sum_{s'} \partial V^{\mu}_{\beta}(s')/\partial\zeta_s$, and we determine it by solving for the fixed point of the Bellman equation in

(3.25). The derivative $\sum_{s'} \partial V_\beta^\mu(s')/\partial \zeta_s$ computed at $\beta \to \infty$ is a measure of sensitivity of the solution to the cost function $\sum_s J_{\zeta\eta}^\mu(s)$ in (4.2) since $V_\beta^\mu$ in (3.10) is a smooth approximation of $J_{\zeta\eta}^\mu(s)$ in (4.2) and $V_\beta^\mu \to J_{\zeta\eta}^\mu(s)$ as $\beta \to \infty$. A similar analysis for Figure 3.4(c)-(e) can be done if the locations $\{x_i\}$ of the user nodes $\{n_i\}$ are known to the *agent*. The sensitivity of the final solution to the locations $\{y_j\}$, $z$ of the small cells and the base station can also be determined in a similar manner.

*Entropy over paths versus entropy of the policy:* We demonstrate the benefit of maximizing the entropy of the distribution $\{p_\mu(\omega|s)\}$ over the paths of the underlying stochastic process as compared to the distribution $\{\mu(a|s)\}$ over the control actions. Figure 3.4(g) demonstrates the 5G network obtained by considering the distribution over the control policy, and the Figure 3.4(h) illustrates the network obtained by considering the distribution over the entire paths. The network cost incurred in the Figure 3.4(h) is 5% less than the cost incurred in Figure 3.4(g). Here, we have considered the above demonstrated probabilistic $p_{ss'}^a$ scenario and minimized the cumulative communication cost incurred only from the user nodes.

*Avoiding poor local minima and large scale setups:* As noted in the Section 3.4.2, annealing $\beta$ from a small value $\beta_{\min}(\approx 0)$ to a large value $\beta_{\max}$ prevents the algorithm from getting stuck at a poor local minima. Figure 3.4(i) demonstrates the network design obtained where the Algorithm 3 does not anneal $\beta$, and iteratively solves the optimization problem at $\beta = \beta_{\max}$. The resulting network incurs a 11% higher cost in comparison to the network obtained in 3.4(h) where the Algorithm 3 anneals $\beta$ from a small to a large value. Figure 3.4(j) demonstrate the 5G network design obtained using Algorithm 3 when the user nodes are increased by around 12 times (610), and the allocated small cells are doubled to 10.

## 3.6 Analysis and Discussion

*A. Mutual Information Minimization:* The optimization problem (3.4) maximizes the Shannon entropy $H^\mu(s)$ under a given constraint on the value function $J^\mu$. We can similarly pose and solve the mutual information minimization problem that requires to determine the distribution $p_{\mu^*}(\mathcal{P}|s)$ (with control policy $\mu^*$) over the paths of the MDP that is close to some given

prior distribution $q(\mathcal{P}|s)$ [28,29]. Here, the objective is to minimize the KL-divergence $D_{KL}(p_\mu \| q))$ under the constraint $J = J_0$ (as in (3.4)).

*B. Non-dependence on choice of $J_0$ in (3.4):* In our framework we do not explicitly determine and work with the value of $J_0$. Instead we work with the Lagrange parameter $\beta$ in the Lagrangian $V^\mu_\beta(s)$ in (3.6) corresponding to the optimization problem (3.4). Since our goal is to minimize $J^\mu(s)$, we aim to solve (3.4) over successive iterations of decreasing values of $J_0$. These iterations are implemented using increasing values of the Lagrange parameter $\beta$. It is known from the sensitivity analysis [5] that the small values of $\beta$ correspond to large values of $J_0$, and large values of $\beta$ correspond to small values of $J_0$. Thus, in our algorithms we solve the optimization problem (3.4) beginning at small values of $\beta = \beta_{\min} \approx 0$ (that corresponds to some feasible large $J_0$), and anneal it to a large value $\beta_{\max}$ (that corresponds to a small $J_0$ value) at which the stochastic policy $\mu$ in (3.8) converges to either 0 or 1. Also at $\beta \approx 0$, the stochastic policy $\mu^*_\beta$ in (3.8) follows a uniform distribution, which implicitly fixes the value of $J_0$. Therefore, the initial value of $J_0$ in the proposed algorithms are fixed and are not required to be pre-specified.

*C. Computational complexity:* Our MEP-based Algorithm 2 performs exactly the same number of computations as the Soft Q-learning algorithm [27] for each *epoch* (or, iteration) within an episode. In comparison to the Q and Double Q learning algorithms, our proposed algorithm, apart from performing the additional minor computations of explicitly determining $\mu^*$ in (3.8), exhibits the similar number of computational steps.

*D. Scheduling $\beta$ and Phase Transition:* In our Algorithm 2, we follow a linear schedule $\beta_k = \sigma k$ ($\sigma > 0$) as suggested in the benchmark algorithm [27] to anneal the parameter $\beta$. In the case of parameterized MDPs (Algorithm 3, and 4) we geometrically anneal $\beta$ (i.e. $\beta_{k+1} = \tau\beta_k$, $\tau > 1$) from a small value $\beta_{\min}$ to a large value $\beta_{\max}$ at which the control policy $\mu^*_\beta$ converges to either 0 or 1. Several other MEP-based algorithms (that address problems akin to parameterized SDM) such as Deterministic Annealing [6], incorporate geometric annealing of $\beta$. The underlying idea in [6] is that the solution undergoes significant changes only at certain *critical* $\beta_{cr}$ (*phase transition*) and shows insignificant changes between two consecutive critical $\beta_{cr}$'s. Thus,

for all practical purposes geometric annealing of $\beta$ works well. Similar to [6] our Algorithms 3 and 4 also undergo the phase transition and we are working on its analytical expression.

*E. Capacity and Exclusion Constraints:* Certain parameterized SDM tasks may pose capacity or dynamical constraints over its parameters. For instance, each small cell $f_j$ allocated in the Figure 3.4 can be constrained in capacity to cater to maximum $c_j$ fraction of user nodes in the network. Our framework allows to model such a constraint as $q_\mu(f_j) := \sum_{a,n_i} \mu(a|n_i)p(f_j|a,n_i) \leq c_j$ where $q_\mu(f_j)$ measures fraction of user nodes $\{n_i\}$ that connect to $f_j$. In another scenario, the locations $\{x_i\}$ of the user nodes could be dynamically varying as $\dot{x}_i = f(x,t)$. Subsequently, the resulting policy $\mu^*_\beta$ and the small cells $\{y_j\}$ will also be time varying. Similar to [90], in our framework we can treat the free-energy function $V^\mu_\beta$ in (3.10) as a *control-Lyapunov* function and determine time varying $\mu^*_\beta$ and $\{y_j\}$ such that $\dot{V}^\mu_\beta \leq 0$.

*F. Uncertainty in Parameters:* Many application areas comprise of states and actions where the associated parameters are uncertain with a known distribution over the set of their possible values. For instance, a user nodes $n_i$ in Figure 3.4 may have an associated uncertainty in its location $x_i$ owing to measurement errors. Our proposed framework easily incorporates such uncertainties in parameter values. For example, the above uncertainty will result into replacing $c(n_i, s', a)$ with $c'(n_i, s', a) = \sum_{x_i \in X_i} p(x_i|n_i)c(n_i, s', a)$ where $p(x_i|n_i)$ is the distribution over the set $X_i$ of location $x_i$. The subsequent solution approach remains the same as in Section 3.4.2.

*G. Choice of annealing parameters $\sigma$, $\tau$:* In Algorithm 2 we follow a linear schedule $\beta_k = \sigma k$ ($\sigma > 0$) in our simulations on model-free RL setting as suggested in the entropy regularized (soft Q) benchmark algorithm in [27]. The idea is to anneal the parameter $\beta$ from a small value (at which the policy $\mu^*_\beta$ in (3.8) is explorative) to a large value (where the policy becomes exploitative). As suggested in [27] the parameter $\sigma$ can be obtained by performing initial runs (for a small number of iteration) with different values of $\sigma$, and picking the one that results into lower value function corresponding to the learned policy. This identified value of $\sigma$ for a particular domain can then be re-used in similar domains without the need of performing any initial

runs.

In Algorithms 3 and 4 the parameter $\tau$ is referred to as the annealing rate and is chosen to be greater than 1. The resulting schedule $\beta_{k+1} = \tau\beta_k$ gemoterically anneals the Lagrange parameter $\beta$ from a small value $\beta_{\min} \approx 0$ to a large value $\beta_{\max}$ at which the control policy $\mu^*_\beta$ in (3.8) converges to 0 or 1. In the case of parameterized SDM, this facilitates a homotopy from the convex function $H^\mu(s)$ to the (possibly) non-convex $J^\mu_{\zeta\eta}(s)$ in (4.2), and prevents from getting stuck in poor local minima. For the purpose of simulations in Figure 3.4 we consider the value $\tau \in (1.01, 1.04)$.

A practical method to determine appropriate $\tau$ can be as follows. Start with a large (for instance $\tau \approx 1.5$) estimate for $\tau$. If the parameter values obtained in the initial iterations of the algorithm *oscillate* a lot then decrease $\tau$. Choose the $\tau$ value where the observed parameter values stop oscillating for the initial iterations of the algorithm. This practical method is rooted in the phenomenon of *phase transition* that our Algorithms 3 and 4 undergo. We illustrate this phenomenon further below.

The Algorithms 3 and 4 address the class of parameterized SDM that simultaneously solve for an unknown parameter and the control policy. These problems are akin to the Facility Location Problem (FLP) addressed in [6]. In particular, [6] develops a Maximum Entropy Principle framework to determine the location of the facilities (parameter), and associate each user node to the closest facility (control policy). In the resulting algorithm (popularly known as Deterministic Annealing (DA)) it is observed that the solution changes significantly only at certain *critical values* of $\beta = \beta_{cr}$ that correspond to the instances of phase transition. At other values of $\beta$, the solution does not change much [90]. It has been observed that a geometric law $\beta_{k+1} = \tau\beta_k$ with $\tau = 1 + \epsilon > 1$ to anneal $\beta$ suffices to capture the changes in the solution that occur during the phase transition. Thus, with reference to the method for choosing a value of $\tau$, if the initial iterations result in high variation in parameter values then possibly some phase transitions are getting skipped and the user needs to anneal slower (i.e., reduce $\tau$) to capture all the phase transitions appropriately.

*H. Extension to Continuous Spaces:* The para-SDM framework proposed in Section 3.4 in this chapter conveniently extends to the application areas with large (or, even continuous) state spaces. For instance, in our simulations

**Algorithm 5** Model-free learning of para-SDM in continuous spaces

Input: $\beta_{\min}$, $\beta_{\max}$, $\tau$, $T$;
Output: $\mu^*$, $\zeta_2$, $\eta_2$
Initialize: $\beta = \beta_{\min}$, $\mu_t = \frac{1}{|\mathcal{A}|}$, and $\zeta_2, \eta_2$ to 0.
**while** $\beta \leq \beta_{\max}$ **do**
    **while** $\zeta_2$ and $\eta$ converge **do**
        **for** $i = 1$ to T **do**
            locally perturb $\zeta_2$ and $\eta$.
            learn the neural net approximator $\hat{\Lambda}_{\beta,\zeta\eta}$.
        **end for**
        Learn $\mathcal{V} := \int_{s\in\mathcal{S}_1} V_{\beta,\zeta\eta}(s)$ as a function of $\zeta_2$, $\eta$
        update $\zeta_2, \eta_2$ that minimizes $\mathcal{V}$ in neighbourhood of current values.
    **end while**
    $\beta \leftarrow \tau\beta$
**end while**

on design of 5G small cell network in Figure 3.4, the user nodes $\{n_i\}$ could possibly be distributed in *continuous* patches in the domain $\Omega$ of the network. The network design objective, however, is still to allocate given number of small cells in the domain and design communication routes in the network to minimize the associated cost. This is equivalent to the para-SDM problems where the states in $\mathcal{S}_1 \subset \mathcal{S}$ are continuously distributed. Note that, the states in $\mathcal{S}_2 \subset \mathcal{S}$ are still discrete and parameterized by the unknown parameters $\zeta_2$. Such para-SDM tasks require to train a function approximator, such as a neural network with parameters $\mathbf{w}$, to estimate the state-action value function $\Lambda_{\beta,\zeta\eta}(s,a)$ in (3.14). We borrow the methodology develop in Deep-Q learning algorithm [91], that extends the classical Q-learning algorithm to the continuous state spaces, and use it to learn the $\Lambda_{\beta,\zeta\eta}(s,a)$-function aspect of our para-SDM algorithms. In particular, we learn the function approximator $\hat{\Lambda}_{\beta,\zeta\eta}(s,a;\mathbf{w})$ by minimizing the following loss at each step $i$ in the interaction with the environment

$$L_i(\mathbf{w}_i) = \mathbb{E}_{s,a,c,s'\sim\rho(\cdot)}\Big[\big(y_i - \hat{\Lambda}_{\beta,\zeta}(s,a;\mathbf{w}_i)\big)^2\Big], \tag{3.28}$$

where $y_i = c(s,a,s') - \frac{\gamma^2}{\beta}\log\sum_{a'}\exp\big(-\frac{\beta}{\gamma}\hat{\Lambda}_{\beta,\zeta\eta}(s',a';\mathbf{w}_{i-1})\big)$ and $\rho(\cdot)$ represents the distributions over the transitions $s,a,c,s'$ observed in the environment. We avoid computing the full expectation in the above loss function by using stochastic gradient descent methods; this is popularly referred to

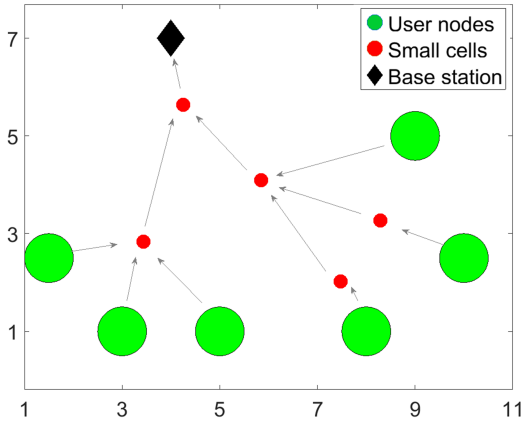


Figure 3.5: Model-free learning in Parameterized SDMs with continuous spaces - Design of 5G small cell network. Here the user nodes $\{n_i\}$ are distributed uniformly in the green patches in the domain, and the objective is to allocate the small cells $\{f_j\}$ in the network and design the communication routes that minimize the associated cost.

as experience replay [91], where $N$ samples from the memory are drawn and the loss function $L_i(\mathbf{w}_i)$ is estimated as $\frac{1}{N}\sum_{j=1}^{N}\left(y_i^j - \hat{\Lambda}_{\beta,\zeta\eta}(s_j, a_j; \mathbf{w}_{i-1})\right)^2$.

The other aspect of identifying the unknown parameters of the para-SDM problems can be efficiently handled by learning state-action value function $\hat{\Lambda}_{\beta,\zeta\eta}(s,a)$ in the neighbourhood of the current $\zeta_2$ parameter values of the states in $\mathcal{S}_2$ and the current $\eta_2$ parameter values of the actions $\mathcal{A}$. Thereafter, updating the $\zeta_2, \eta_2$ parameters such that it minimizes the total state value function $V_{\beta,\zeta\eta}(s)$ in (3.10) (resulting from the above learned state-action value function $\hat{\Lambda}_{\beta,\zeta\eta}(s,a)$) in the above neighbourhood. Please refer to the Algorithm 5 for details. Figure 3.5 illustrates the utility of the Algorithm 5 in designing the 5G small cell network where the user nodes $\{n_i\}$ are distributed in continuous patches across the network domain $\Omega$ instead of being located at discrete places as previously in Figure 3.4. We can use the above methodology to learn the state-action value function $\Lambda_\beta(s,a)$ in (3.14) in the case of model-free Reinforcement Learning problems (as illustrated in Figure 3.2 - without any unknown parameters). The resulting algorithm can be deployed on large datasets as handled by the other Deep-RL learning algorithms; because the only difference between them and our MEP-based RL is only in the expression of the state-action value function which essentially reflects in a single line of the code.

## 3.7 Summary

In this chapter we first propose an MEP-based framework to address the MDPs and extend them to the model-free RL setting. Here, the significant differences from existing literature is in considering the entropy over the paths of the MDP as opposed to consider the entropy of the stochastic control policy [27–33, 36, 37]; and we demonstrate the benefit of the same in terms of convergence speed, robustness to noisy data, and final cost incurred. Thereafter, we define the parameterized SDM problems over infinite horizon and develop an MEP-based framework that simultaneously determines the unknown parameters and the optimal control policy (sequential decisions). Thereafter, we extend it to the case where the underlying cost function and system dynamics are not known, that is, the case of that requires the algorithm to learn from its interactions with the underlying system (environment); i.e., the model-free learning case. We demonstrate the efficacy of our framework through its utility in the design of 5G small cell networks when the underlying model (cost function and dynamics) is known, as well as unknown. We highlight the capability of our framework to obtain sensitivity measures to the para-SDM problem parameters, to incorporate problem specific constraints, and its extension to continuous (large) state spaces.

# CHAPTER 4

# DYNAMIC PARAMETERIZED SDM

## 4.1 Introduction

As illustrated in Chapter 3, scenarios such as self organizing networks [73], 5G small cell network design [74, 75], supply chain networks, and last mile delivery problems [76] pose a *parameterized* SDM taskḢere, the two-fold objective is to simultaneously determine (a) the optimal control policy for the underlying stochastic process, and (b) the unknown parameters that the state and action variables depend upon such that the cumulative cost is minimized. For instance, the Figure 4.1 illustrates a 5G-small cell network where the user nodes $\{n_i\}$ located at $\{x_i \in \mathbb{R}^d\}$ communicate with the base station $\delta$ located at $z \in \mathbb{R}^d$ via a network of small cells $\{f_j\}$. The objective of network design problem is to determine the locations $\{y_j\}$ of the small cells, and the control policy governing the information packet flow such that the cumulative cost (for instance, delay) incurred in the network is minimized. Often such network design problems have dynamics associated to its nodes. For instance, the user nodes $\{n_i\}$ and the base station $\delta$ may evolve dynamically, requiring the small cells and the control policy, too, that result into minimum communication cost to be time-varying. A straightforward approach to address such problems is to solve the associated parameterized SDM at each time instant $t$. However, there are several problems associated with such an approach. Firstly, the computational expense of solving the underlying optimization problem will be immense and thus, it is unfit for application in an online environment. Secondly, under this approach the resulting dynamics of the parameters may not be viable; i.e., the parameters may undergo significant change in values from the time instant $t$ to $t + \Delta t$ owing to the (possible) non-convexity of such problems. The control effort required in implementing such dynamics may be impractical. In this chapter we develop a scalable

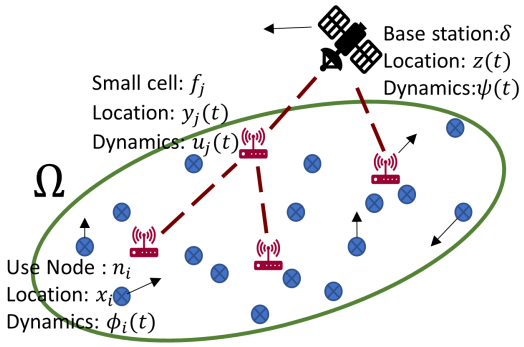


Figure 4.1: Illustrates the time varying 5G Small Cell Network. The location $z(t) \in \mathbb{R}^d$ of the 5G Base Station evolves as per $\psi(t)$, and the locations $\{x_i(t) \in \mathbb{R}^d\}$ of the users $\{n_i\}$ is governed by $\{\phi_(t) \in \mathbb{R}^d\}$ . The objective is to determine the dynamically evolving small cell $\{f_j\}$ locations $\{y_j(t) \in \mathbb{R}^d\}$ and the time varying communication routes (control policy) from the Base Station $\delta$ to each user $\{n_i\}$ via the network of the small cells such that the total cost of communication gets minimized at each time instant.

framework to address the class of dynamic parameterized SDM. We design control laws that govern the unknown dynamics of the concerned parameters in the problem. We show that our control design ensures tracking of the local optima of the associated cost function associated.

## 4.2 Problem Formulation

We consider the definition of the parameterized SDM problems as illustrated in Section 3.4. In particular, we define this parameterized SDM as a tuple $\langle \mathcal{S}, \mathcal{A}, c, p, \gamma, \zeta, \eta \rangle$ where $\mathcal{S} = \{s_1, \ldots, s_N = \delta\}$, $\mathcal{A} = \{a_1, \ldots, a_M\}$, and $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ respectively denote the state space, action space, and cost function; $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ is the state transition probability function and $0 < \gamma \leq 1$ is the discounting factor; $\zeta_1 = \{\zeta_s \in \mathbb{R}^{d_\zeta} : s \in \mathcal{S}_1 \subseteq \mathcal{S}\}$ and $\eta_1 = \{\eta_a \in \mathbb{R}^{d_\eta} : a \in \mathcal{A}_1 \subseteq \mathcal{A}\}$ denote the *known* state and action parameters, and $\zeta_2 = \{\zeta_s : s \in \mathcal{S}_2 = \mathcal{S} \backslash \mathcal{S}_1\}$ and $\eta_2 = \{\eta_a : a \in \mathcal{A}_2 = \mathcal{A} \backslash \mathcal{A}_1\}$ denote the *unknown* state and action parameters. A control policy $\mu : \mathcal{A} \times \mathcal{S} \to \{0, 1\}$ determines the action taken at each state $s \in \mathcal{S}$, where $\mu(a|s) = 1$ implies that action $a \in \mathcal{A}$ is taken when the system is in the state $s \in \mathcal{S}$ and $\mu(a|s) = 0$ indicates otherwise. For every initial state $x_0 = s$ and parameter values $\zeta := \zeta_1 \sqcup \zeta_2$, $\eta := \eta_1 \sqcup \eta_2$, the control policy induces a stochastic process, whose realization is a *path* $\omega$ - an infinite sequence of actions and

states, that is

$$\omega = (u_0, x_1, u_1, x_2, u_2, \ldots, x_T, u_T, x_{T+1}, \ldots), \tag{4.1}$$

where $u_t \in \mathcal{A}$, $x_t \in \mathcal{S}$ for all $t \in \mathbb{Z}_{\geq 0}$ and $x_t = \delta$ for all $t \geq k$ if and when the system reaches the termination state $\delta \in \mathcal{S}$ in $k$ steps; the corresponding cost incurred by the stochastic process is given by (see Chapter 3 for details)

$$J^{\mu}_{\zeta\eta}(s) = \mathbb{E}_{p_\mu}\Big[\sum_{t=0}^{\infty} \gamma^t c\big(x_t(\zeta), u_t(\eta), x_{t+1}(\zeta)\big)\Big| x_0 = s\Big]. \tag{4.2}$$

In the case of *dynamic parameterized SDM* tasks, the parameters associated to the states and actions in the set $\mathcal{S}_1$, and $\mathcal{A}_1$, respectively, have a predetermined dynamics given by

$$\begin{aligned} \dot{\zeta}_1 &= \phi_1(\zeta, \eta, t) \\ \dot{\eta}_1 &= \psi_1(\zeta, \eta, t), \end{aligned} \tag{4.3}$$

where we assume that the dynamics $\phi_1, \psi_1 \in C^0$ are continuous functions. Owing to this time evolution of the known parameters $\zeta_1$, $\eta_1$ in the SDM, the control policy $\mu^*$, and the unknown state and action parameters $\zeta_2$, $\eta_2$ that minimize the cost function (4.2) at each time instant also need to evolve dynamically. We develop a control theoretic framework that determines the dynamics

$$\begin{aligned} \dot{\zeta}_2 &= \phi_2(\zeta_1, \zeta_2, , \eta_1, \eta_2, t) \\ \dot{\eta}_2 &= \psi_2(\zeta_1, \zeta_2, \eta_1, \eta_2, t) \end{aligned} \tag{4.4}$$

that governs the time evolution of the parameters $\zeta_2$, and $\eta_2$, and subsequently evaluates the control policy $\mu^*$ at each time instant $t$ by solving the contraction map (3.11). Though, a straightforward method to approach this problem is to solve the optimization problem (4.2) at each time instant $t$. However, such an approach is computationally expensive, and does not make use of dynamic evolution of the parameters $\zeta_1$ and $\eta_1$ resulting into much computational redundancy.

## 4.3 Problem Solution

In our framework, we view the dynamical systems (4.3) and (4.4) together as a control system

$$\dot{\Upsilon} = f(\Upsilon, t), \tag{4.5}$$

where $\Upsilon = [\zeta_1\ \eta_1\ \zeta_2\ \eta_2] \in \mathbb{R}^{Nd_\zeta + Md_\eta}$, $f = [\phi_1\ \psi_1\ \phi_2\ \psi_2] \in \mathbb{R}^{Nd_\zeta + Md_\eta}$. We design the control field $u(t) := [\phi_2\ \psi_2]$ such that the value function (4.2) is minimized at each time instant. We achieve this by considering the following function as a control-Lyapunov function for the system in (4.5)

$$\mathcal{V}(\Upsilon) = \sum_{s \in \mathcal{S}} V^*_{\beta,\Upsilon}(s), \tag{4.6}$$

where $V^*_{\beta,\Upsilon}(s)$ is the optimal free energy (an approximation of (4.2)) given in (3.20), and $\Upsilon$ denotes its dependence on the state and action parameters. Subsequently, we determine the control field $u(t)$ such that the time derivative $\dot{\mathcal{V}}$ along the trajectory $\Upsilon(t)$ is non-positive. We summarize the properties of $\mathcal{V}$, and its time derivative in the following theorem.

**Theorem 10.** *Let $\mathcal{V}$ be the control-Lyapunov candidate for the parameters $\Upsilon$, whose dynamics are given by (4.5). Then,*

1. *Positive definiteness: There exists a constant $c > 0$ such that $\mathcal{V}(\Upsilon)+c > 0$ for all $\Upsilon \in \mathbb{R}^{Nd_\zeta + Md_\eta}$.*

2. *There is no dynamic control authority only at those time instants when the unknown parameters $\zeta_2, \eta_2$ are at local minima.*

*Proof.* See Appendix A.3.1. □

*Control Design for tracking parameters $\zeta_2, \eta_2$:* The time derivative of the control Lyapunov function $\dot{\mathcal{V}}$ is given by

$$\dot{\mathcal{V}} = G^T \kappa + H^T u, \tag{4.7}$$

where $\kappa^T = [\phi_1^T\ \psi_1^T]$ $G^T = [G_\phi^T\ G_\psi^T] \in \mathbb{R}^{|\mathcal{S}_1|d_\zeta + |\mathcal{A}_1|d_\eta}$, $G_\phi = [g_\phi(s)]_{s \in \mathcal{S}_1}$, $g_\phi(s) = \frac{\partial \mathcal{V}(\Upsilon)}{\partial \zeta_s}$, $G_\psi = [g_\psi(a)]_{a \in \mathcal{A}_1}$, $g_\psi(a) = \frac{\partial \mathcal{V}(\Upsilon)}{\partial \eta_a}$, $H^T = [H_\phi^T\ H_\psi^T] \in \mathbb{R}^{|\mathcal{S}_2|d_\zeta |\mathcal{A}_2|d_\eta}$, $H_\phi = [h_\phi(s)]_{s \in \mathcal{S}_2}$, $h_\phi(s) = \frac{\partial \mathcal{V}(\Upsilon)}{\partial \zeta_s}$, $H_\psi = [h_\psi(a)]_{a \in \mathcal{A}_2}$, $h_\psi(a) = \frac{\partial \mathcal{V}(\Upsilon)}{\partial \eta_a}$. We make

use of the *affine* dependence of $\dot{\mathcal{V}}$ on the control $u(t)$ to make $\dot{\mathcal{V}}$ non-positive analogous to the chosen control design [68–70]. Specifically, we choose $u(t)$ of the form

$$u(\Theta) = \begin{cases} -\left[K_0 + \dfrac{\alpha + \sqrt{\alpha^2 + (H^T H)^2}}{H^T H}\right] H & \text{if } H \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

where $\Theta = [G^T \ H^T] \in \mathbb{R}^{Nd_\zeta + Md_\eta}$, $\alpha = G^T \kappa$, $K_0 > 0$. The following two theorems show that for the above choice of the control $u(\Theta)$, the state and action parameters $\zeta_2(t), \eta_2(t)$ asymptotically tracks the condition $H \to 0$, and also the control in (4.8) is bounded even near $H = 0$.

**Theorem 11.** *Asymptotic convergence: For the dynamical system (4.5) the choice of control $u(\Theta)$ in (4.8) results in $\dot{\mathcal{V}} \leq 0 \ \forall \ t \geq 0$ and the derivatives in $H(t) \to 0$ as $t \to \infty$.*

*Proof.* See Appendix A.3.2. □

**Theorem 12.** *Lipschitz continuity: If there exists a control $\hat{u} : \mathbb{R}^{Nd_\zeta + Md_\eta} \to \mathbb{R}^{|\mathcal{S}_2|d_\zeta + |\mathcal{A}_2|d_\eta}$ Lipschitz at $\Theta = 0$, such that $\dot{\mathcal{V}} \leq 0 \ \forall \ t \geq 0$ for this control, then the choice of control $u(\Theta)$ in (4.8) is Lipschitz at $\Theta = 0$. That is, $\exists$ $\epsilon > 0$ and a constant $c_0$ such that $\|u(\Theta)\| \leq c_0 \|\Theta\|$ for $\|\Theta\| \leq \epsilon$.*

*Proof.* See Appendix A.3.3. □

*Remark* 8. (a) The Theorem 12 emphasizes the non-conversativeness of our solution; i.e., if there exists a Lipschitz (bounded) solution such that $\dot{\mathcal{V}} \leq 0$ then Theorem 5 implies that our proposed solution is also Lipschitz (bounded). (b) For the purpose of simulation, we discretize time into $\Delta t$ intervals. At instants $H \neq 0$ we determine the dynamics of $\zeta_2(t)$ and $\eta_2(t)$ using the control design $u(t)$ in (4.8). This results into $\dot{\mathcal{V}} \leq 0$ at all such time instants. For the time instants when $H = 0$ we already have the parameters $\zeta_2$ and $\eta_2$ at the local minima. At such instants $\dot{\mathcal{V}}$ may be positive, negative or zero depending on the known (predefined) dynamics of the state and action parameters $\zeta_1$ and $\eta_1$.

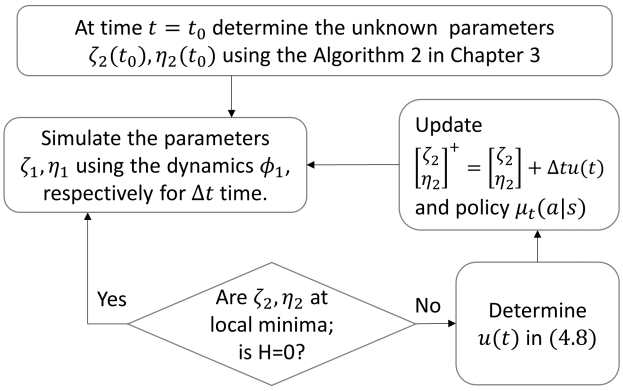


Figure 4.2: Flowchart with the implementation of our proposed algorithm for dynamic parameterized MDPs.

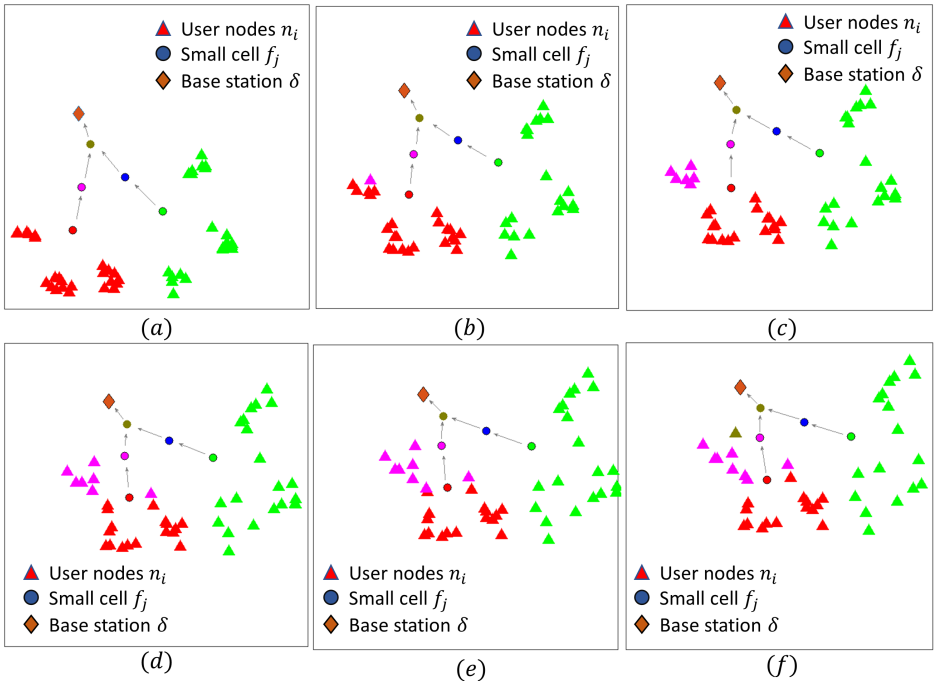


Figure 4.3: (a)-(f) Illustrates the time varying 5G small cell network problem. Observe the change in locations of the user nodes $\{n_i\}$ and the base station $\delta$ with time. Thus, the resulting small cell locations in the network have dynamics governed by $u(t)$ in (4.8). Also, observe the change in the color of the triangles (denoting user nodes) from (a) to (b), (b) to (c) (, and further) indicating the change in communication paths.

## 4.4 Simulation

In this section we provide sample simulations of our proposed control design to determine the time evolution of unknown state and action parameters in dynamic parameterized SDM. We consider the case of 5G small cell networks that are modelled as parameterized SDM (as shown in Chapter 3). In our simulations, we consider that the base station $\delta$ of the small cell network, and the user nodes $\{n_i\}$ (see Figure 3.4) are mobile, and thus, the small cells $f_j$ allocated in the network must also evolve with time so that the cost of network design is minimized at each time instant. We summarize the steps of our algorithm in Figure 4.3.

Figure 4.3(a) illustrates the network design obtained at the initial time instant. Here, a user node $n_i$ of a particular color first sends its information packet to the small cell of the similar color which then reaches the base station via the route indicated in the figure. The dynamics for the user nodes $\{n_i\}$ and the base stations $\delta$ we chose randomly and their evolution can be noted in the Figure 4.3(b)-(e). The algorithm presented in Figure 4.3 updates the locations of the small-cells, and also the corresponding communication routes (i.e., the control policy) via the control design $u(\Theta)$ in (4.8).

Our algorithm in Figure 4.2 incrementally updates the small cell locations and the communication path from each user node using the control design $u(t)$ in (4.8) at every time interval of $\Delta t = 0.03$ seconds. As shown in the Figure 4.2(a), at time $t = t_0$ all the user-nodes are either green or red, where the green user nodes send their communication packet to the green small cell, and similarly for the red user nodes. As time progresses, we observe a change in the user node locations, the base station location, the small cell locations and the optimal communication path from each user node to the base station, which are clearly indicated by the change in color of the triangles representing the user nodes as shown in Figure 4.3.

## 4.5 Analysis and Discussion

1. The above control design methodology can be extended to the dynamic parameterized SDM problems with additional constraints over the state and action parameters. For instance, the capacity and exclusion con-

straints that govern the utility of a state or action are prevalent when network design problems are modelled as parameterized SDM. These constraints reflect in the optimal control policy $\mu(a|s)$ in (3.8), however, the control design methodology illustrated above remains unaffected.

2. For the purpose of simulation, we discretize time into $\Delta t$ intervals. At instants $H \neq 0$ we determine the dynamics of $\zeta_2(t)$ and $\eta_2(t)$ using the control design $u(t)$ in (4.8). This results into $\dot{\mathcal{V}} \leq 0$ at all such time instants. For the time instants when $H = 0$ we already have the parameters $\zeta_2$ and $\eta_2$ at the local minima. At such instants $\dot{\mathcal{V}}$ may be positive, negative or zero depending on the known (predefined) dynamics of the state and action parameters $\zeta_1$ and $\eta_1$.

3. Apart from saving the computational expense, our proposed control design methodology provides viable dynamics for the state and action parameters $\zeta_2$, $\eta_2$. In particular, if the *frame-by-frame* approach, where the optimization problem is solved at each time instant $t$, is used then the resulting time-varying $\zeta_2(t)$ and $\eta_2(t)$ may not necessarily follow a viable dynamics, i.e., they may undergo considerable changes from the time instant $t$ to $t + \Delta t$ based on the solution obtained at each time instant (due to the (possible) non-convexity of the cost function). The control effort required to implement such non-viable dynamics will be vast, and possibly impractical. A similar scenario was observed and illustrated in the case of $d$-FLPO problem illustrated in the Section 2.7.

4. If the state transition probabilities $p(s'|s, a)$ are known apriori then the control design methodology presented here is also extendable to the case of model-free learning problems. This is owing to the fact that the control $u(\Theta)$ in (4.8) depends entirely on the derivatives $\frac{\partial V(s')}{\partial \zeta_s}$, $\frac{\partial V(s')}{\partial \eta_s}$ which are computable using the steps illustrated in Section 3.4. In such instances, the pre-defined dynamics $\phi_1$, $\psi_1$ (required in (4.8)) of the parameters $\zeta_1$ and $\eta_1$ can be estimated using temporal difference methods.

## 4.6 Summary

The dynamic parameterized SDMs addressed here determines the time varying control policy, and the state and action parameters such that the associated cumulative cost function gets minimized at each time instant. In our framework, we view the entire dynamical system (with known and unknown dynamics) as a control system, and design control laws for the unknown dynamics such that the above objective is achieved. Our proposed methodology builts on the MEP-based framework developed to address parameterized SDM in Chapter 3. In particular, the free energy in (3.10) - which is a smooth approximation to the cost function of the SDM - is shown to be suitable a control-Lyapunov function for the underlying control system. We show that the resulting non-linear feedback control law determines the unknown dynamics for the concerned state and action parameters under which asymptotic tracking of local optimal of the parameterized SDM is achieved. We also demonstrate that the feedback control law is non-conservative, i.e., if there exists a Lipschitz control law that asymptotically tracks the local optimal of the parameterized SDM then the proposed control law is also Lipschitz and bounded.

# CHAPTER 5

# ON THE PERSISTENCE OF CLUSTERING SOLUTIONS

## 5.1 Introduction and Related Work

This chapter is the first in the three-part series where we elucidate the benefits of phase transitions, as exhibited by the MEP-based framework, in determining and designing hyper-parameters. In particular, we consider the problem of data clustering, and demonstrate the utility of phase transitions in characterizing the clustering solutions obtained at different number of clusters; and subsequently, in estimating the *true* number of clusters underlying any given dataset. Though, our proposed methodology is rooted in the Maximum Entropy Principle, we also show that it is intuitively understandable in terms of *resolution* scales at which the datasets are being observed.

Many high-impact application areas such as bio-informatics [92], [93], exploratory data mining [94], combinatorial drug discovery [7], data and network aggregation [95], medical imaging [96] and many other information processing fields have fueled significant work on clustering algorithms. Most of these algorithms such as $k$-means [97], $k$-medoids [98], and expectation-maximization [99] require the number of clusters to be prespecified. Despite substantial work on clustering algorithms, there is relatively scant literature on determining the true number of clusters in a dataset. In this context, it should be noted that there is no single agreed-upon notion of *natural* clusters or *true* number of clusters; typically existing algorithms make assumptions on the datasets (e.g. generated from a mixture of Gaussian distributions) and validate their results on datasets that satisfy the assumptions.

There are various measures developed to characterize the clustering solutions resulting from a clustering algorithm with different number ($k$) of clusters. One of the popular methods for determining the number of clusters is based on computing *gap statistic* [100]. It compares the total intracluster

variation for different values of $k$ (number of clusters) with their expected values under null reference distribution of the data. The number of clusters $k$ is ascribed to the case where the gap is largest. However, as remarked in [101], this method works well for finding a small number of clusters, but has difficulty as the true $k$ increases.

Some of the recent methods that determine the number of clusters under some assumptions on datasets include - $X$-means algorithm [102], where clustering using $k$-means is performed for a range of number $k$ of clusters, and the value $k_t := k$ that yields the best Bayesian Information Criterion (BIC) [103] score is chosen as an estimate for the true number of clusters. Other related algorithms use criteria such as Akaike information criteria [104] or minimum description length [105] instead of BIC. The $X$-means algorithm works well for well-separated spherical clusters but tends to overfit in the case of non-spherical clusters [101].

The information-theoretic approach [106] where it estimates the number of true clusters $k_t$ by detecting a significant *jump* in the modified distortion $D^\gamma$ vs $k$ plot; here $D$ is the clustering distortion objective, $k$ is the number of clusters, $\gamma \approx -d/2$, and the data points are in $\mathbb{R}^d$. Although the choice $\gamma \approx -d/2$ works well for certain datasets, one can find examples where this choice fails [106]. $G$-means [107] algorithm identifies the number of clusters in a dataset under the assumption that each cluster is a Gaussian distribution. It is a hierarchical algorithm that increases the number of clusters $k$ until the hypothesis that each cluster comes from a single Gaussian distribution is validated; typically done using the Anderson-Darling statistic test [108] on each cluster after projecting it onto a one-dimensional space. $PG$-means algorithm [101] is an improvement on the $G$-means algorithm, where the number of clusters in a Gaussian mixture model is obtained by applying the Kolmogorov-Smirnov (KS) test to the one-dimensional projection of the *entire* dataset; $PG$-means also works well when the true clusters are overlapping with each other.

Dip-means [109] is another method that assumes the dataset is generated from a mixture of unimodal distributions. Here, Hartigan's dip statistic test [110] is used to verify the unimodal nature of the admissible cluster. The authors also extend the dip-means algorithm to shape clustering problems by using kernel $k$-means [111] for clustering. Other alternative approaches in the literature to estimate the true number of clusters are Bayesian $k$-

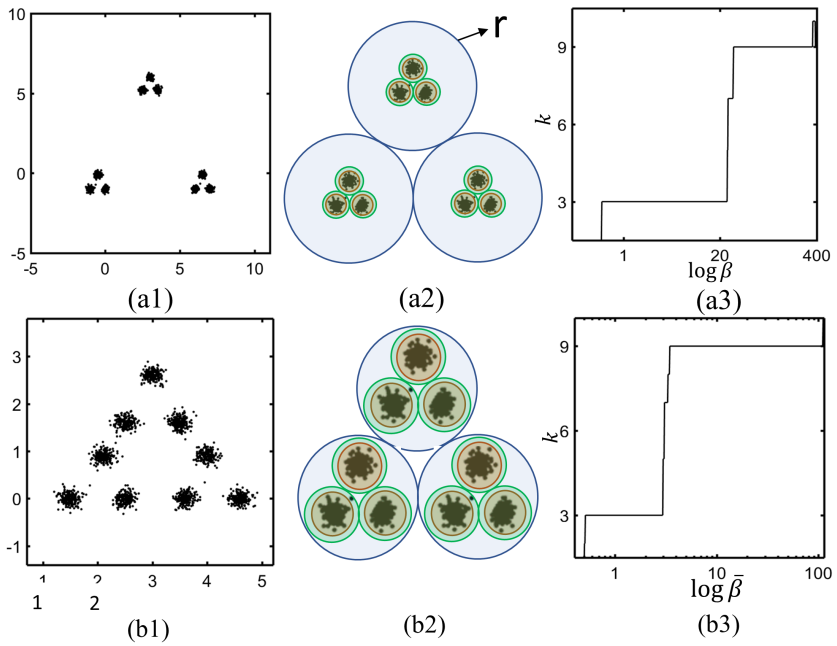


Figure 5.1: Illustration of a mixture of nine Gaussian distributions arranged in groups of three superclusters. In (a1) the three superclusters are well separated from each other while in (b1) they are closer to each other. Observe that in (a2) for a large range in resolution scales (radii $r$) within the blue annulus, each supercluster appears as a single cluster, and only for a small range of resolution scale depicted by green annulus, each Gaussian distribution is identifiable separately. In other words for a large range of resolution the only the three superclusters are distinguishable from one another while for a smaller range of resolution each Gaussian distribution is identified separately. In (b1) since the three superclusters are closer to each other, the range of resolution scales within the blue annulus gets reduced, thereby indicating existence of three natural clusters in (a1) and nine natural clusters in (b1).

means [112] that uses Maximization-Expectation to learn a mixture model, a method based on repairing faults in Gaussian mixture models [113] and various stability-based model validation methods [114], [115], [116]. The main drawbacks in most of the above existing methods stem from the underlying restrictive assumptions on the datasets; accordingly, the algorithms do not perform well when datasets do not meet the assumptions, which is often the case when considering standard non-synthetic datasets. These methods fail to accurately estimate the true number of clusters in most of the standard datasets as illustrated in the experiment section in this chapter.

In this chapter we develop a notion of *persistence* of clustering solutions that enables comparing solutions, which result from a clustering algorithm, with different number of clusters. Here we do not make any assumptions on

the underlying data distribution. Since a clustering solution requires grouping a set of points in such a way that points in the same cluster are more similar to each other than to those in other clusters. We characterize *persistence* of a clustering solution as the *range* of resolution scales for which (a) points within each cluster seem indistinguishable, and (b) points in different clusters are distinguishable. For instance, Figure 5.1(a1) illustrates a dataset containing nine Gaussian clusters which are arranged in groups of three superclusters. If we choose the resolution scale of radius $r$, as shown in Figure 5.1(a2), then the points within each super-cluster is indistinguishable. Therefore, one will conclude that at this resolution level the dataset consists of only three clusters. Also note that in Figure 5.1(a1), the three super-clusters are persistent for a large range of resolution scales as indicated by the thickness of the blue annulus around each of them. On the other hand, the green annulus around each of the nine Gaussian clusters, is relatively thinner indicating that a clustering solution that identifies all the nine Gaussian clusters is relatively less persistent.

In a later section we quantify this notion of persistence of a clustering solution with $k$ distinct clusters. In particular, the persistence is characterized in terms of the maximum over two-norms of all the cluster covariance matrices at two successive values of $k$. We also show analytically how for a clustering solution that identifies natural clusters, this measure correctly estimates the true number of clusters through a simple illustrative example consisting of spherical clusters with uniform distributions. We also provide extensive experimental results on a variety of standard and synthetic datasets in a later section. The results demonstrate that our method outperforms over the existing algorithms described above. In particular, our method correctly estimates the true number of clusters on 13 of the 14 benchmark datasets tested, whereas the next best method could estimate the true number of clusters only on 7 benchmark instances.

## 5.2 Persistence of a Clustering Solution and its Quantification

**Intuitive description :** The notion of a cluster can be related to the resolution scales at which a dataset is viewed. For instance, on one hand the

entire dataset can be considered as a single cluster, while on the other hand each data point can also be considered as a cluster by itself. Thus in Figure 5.1(a1), at low resolution scale (characterized here by a radius greater than the diameter of the entire dataset) no two points of the dataset are distinguishable from each other and the entire dataset is deemed a single cluster. Now consider a clustering solution at a higher resolution scale (for instance resolution characterized by the radius $r$ in Figure 5.1(a2)), the datapoints from the three superclusters become distinct from one another and we are able to identify these as three distinct clusters in the dataset. Upon further increasing the resolution scale (for instance resolution characterized by the radius $r_1$) in Figure 5.1(a2), the datapoints sampled from each of the nine Gaussian distributions become distinct from one another and we are able to identify the nine clusters in the dataset. On further increasing the resolution scale, each point by itself will be regarded as a cluster.

We use resolution to capture the notion of persistence of clustering solution. We propose that the clustering solution that persists for large range of resolution is a good indicator of natural clusters and corresponding number estimates true number of clusters. In fact, after quantifying persistence later in this section, we show that for both the datasets in Figure 5.1(a1) and 5.1(b1), the persistence is larger for clustering solutions with three and nine clusters, while they are relatively small for clustering solutions with other number of clusters. Moreover, we observe that the three super-clusters are more persistent than the nine clusters for the dataset in Figure 5.1(a1), while nine clusters are more persistent than the three super-clusters in Figure 5.1(b1); this is also intuitive from the relative thickness of the blue and green annuli in Figures 5.1(a2) and 5.1(b2). Therefore, this suggests that the three super-clusters are *more* natural in Figure 5.1(a1) while the nine clusters are *more* natural in Figure 5.1(b1). These inferences agree with our intuition from visual inspection of these datasets and are also corroborated by the measure proposed later in this section.

Let $\beta_k$ denote the lowest resolution scale at which $k+1$ clusters are identifiable in a dataset. We define the persistence of a clustering solution with $k$ clusters as $[\log \beta_k - \log \beta_{k-1}]$. The *true number* of clusters can be estimated in terms of persistence of clustering solution. Accordingly, if the clustering into $k$ groups is persistent for a long range of resolution scales, without $k+1$ clusters becoming evident in the dataset, then $k$ is a good estimate for the true

number of clusters. In other words, for a clustering solution with true number $k_t$ of natural clusters it takes a large change in the resolution scale for the data points, originally belonging to the same cluster, to become distinguishable enough so as to belong to different clusters. Our hypothesis is that for a clustering solution with $k_t$ natural clusters $\log \beta_{k_t} - \log \beta_{k_t-1} > \log \beta_k - \log \beta_{k-1}$ for all $k \neq k_t$.

**Quantification of persistence of a clustering solution and resolution scales:** This quantification is substantially motivated by our reinterpretation of the deterministic annealing (DA) algorithm [58]. The DA algorithm mimics the annealing procedure studied in statistical physics literature and gives high quality clustering solutions on linearly separable data. We do not describe the DA algorithm in detail here albeit re-interpret the auxiliary cost function (referred to as *free-energy* in [58]) in DA to discern a possible use of its annealing parameter as a measure of resolution.

The DA algorithm views the problem of clustering a dataset $\mathcal{X} = \{x_i : x_i \in \mathbb{R}^d, 1 \leq i \leq N\}$ consisting of $N$ data points into $k$ groups of nearly similar entities as an equivalent facility location problem (FLP), where the goal is to allocate a set of facilities $\mathcal{Y} = \{\mathbf{y_j} : y_j \in \mathbb{R}^d, 1 \leq j \leq k\}$ to data points $\{\mathbf{x_i}\}$ such that the cumulative distance between data points and their nearest facilities is minimum. Note that the facility locations are indeed the centroids of individual clusters (for linearly separable data and with squared Euclidean metric) and the FLP viewpoint is critical to many commonly used clustering algorithms, such as $k$-means. Thus, a solution to FLP results in clustering of the underlying dataset where the corresponding clusters $\{\pi_j\}$ are defined by *voronoi partitions* $\pi_j = \{\mathbf{x_i} \in \mathcal{X} : \mathbf{y_j} = \arg\min_{\{\mathbf{y_l}\}} d(\mathbf{x_i}, \mathbf{y_l})\}$. More precisely, we consider the following optimization problem for FLP

$$D = \min_{\mathcal{Y}} \sum_{i=1}^{N} p_i \sum_{\{\mathbf{y_j}\}} \min_{1 \leq j \leq k} d(\mathbf{x_i}, \mathbf{y_j}), \tag{5.1}$$

where $p_i$ denotes a known relative weight of vector $\mathbf{x_i}$ (e.g. $p_i = \frac{1}{N}$), and $d(\mathbf{x_i}, \mathbf{y_j})$ is a measure of distance between $\mathbf{x_i}$ and $\mathbf{y_j}$ which is usually considered to be the squared Euclidean distance. In data compression literature $D$ is usually referred to as the distortion function [65]. DA considers the log-sum-exp approximation where it approximates $\min_{1 \leq j \leq k} d(\mathbf{x_i}, \mathbf{y_j})$ by $-\frac{1}{\beta} \log \sum_{j=1}^{k} e^{-\beta d(\mathbf{x_i}, \mathbf{y_j})}$, which results in the following smooth optimization

problem that approximates (5.1)

$$F = \min_{\mathcal{Y}} -\frac{1}{\beta} \sum_{i=1}^{N} p_i \log \sum_{\{\mathbf{y_j}\}} e^{-\beta d(\mathbf{x_i}, \mathbf{y_j})}, \tag{5.2}$$

which is parameterized by $\beta \in \mathbb{R}$. The parameter $\beta$ determines the extent of approximation of $D$ by $F$. At larger values of $\beta \to \infty$ the approximation function $F$ tends to converge at the distortion $D$. On the other hand, at low values of $\beta$ ($\approx 0$) approximation $F$ is considerably distinct from the distortion $D$. Here $F$ is referred to as the *free-energy* function and $\beta$ as an annealing parameter in the DA algorithm. We obtain the minimum (local) of $F$ at a given $\beta$, by setting the partial derivative $\frac{\partial F}{\partial \mathbf{y_j}}$ to zero, which results in the following centroid-like condition for squared Euclidean distances:

$$\mathbf{y_j} = \Big( \sum_{i=1}^{N} p_i p(j|i) \mathbf{x_i} \Big) \Big/ \Big( \sum_{i=1}^{N} p_i p(j|i) \Big), \text{where} \tag{5.3}$$

$$p(j|i) = \Big( e^{-\beta d(\mathbf{x_i}, \mathbf{y_j})} \Big) \Big/ \Big( \sum_{\mathbf{y_j} \in \mathcal{Y}} e^{-\beta d(\mathbf{x_i}, \mathbf{y_j})} \Big). \tag{5.4}$$

We justify the annealing parameter $\beta$ as a measure of resolution as follows. Note that for any two data points $\mathbf{x_1}$ and $\mathbf{x_2}$ in the bounded dataset, the term $e^{-\beta d(\mathbf{x_1}, \mathbf{y_j})} \approx e^{-\beta d(\mathbf{x_2}, \mathbf{y_j})}$ when $\beta$ is small ($\beta \approx 0$), i.e. points $\mathbf{x_1}$ and $\mathbf{x_2}$ are indistinguishable. More precisely, for every $\epsilon > 0$, there exists $\beta > 0$ small enough such that $\left| e^{-\beta d(\mathbf{x_1}, \mathbf{y_j})} - e^{-\beta d(\mathbf{x_2}, \mathbf{y_j})} \right| < \epsilon$. Note that from (5.3), at small values of $\beta$ ($\approx 0$) all the facilities $\{\mathbf{y_j}\}$ are coincident. We can deduce from here that at low $\beta$ values, no two data points are distinguishable (within $\epsilon$), that is, the optimization problem (5.2) cannot differentiate between them, and therefore entire dataset will be deemed as a single cluster. In fact, the DA algorithm associates only one resource $\mathbf{y_1}$ at the centroid of the entire dataset. Now as $\beta$ increases, two distinct points that originally belonged to the same cluster $\pi_j$, become distinguishable for large enough $\beta$, i.e. $e^{-\beta d(\mathbf{x_1}, \mathbf{y_j})}$ no longer approximates $e^{-\beta d(\mathbf{x_2}, \mathbf{y_j})}$; thus the problem (5.2) can differentiate between these data points, and they need not necessarily belong to the same cluster. In the limit $\beta \to \infty$, all the data points are entirely distinct from each other and the optimal solution to (5.2) is to assign a distinct facility to each point since no two data points are similar enough to be put in the same cluster. Thus $\beta$ quantifies our intuitive notion of resolution. Equivalently,

$\log \beta$ quantifies the resolution scale.

To quantify *persistence* of a clustering solution in the FLP setup, we need to determine the range of resolution scales $\log \beta$ over which a clustering solution persists. Note that a clustering solution $\{\pi_j\}$ to the FLP persists till the set of cluster centers $\mathcal{Y} = \{\mathbf{y_j}\}$ cease to be a minima of $F$ as the annealing parameter $\beta$ (resolution) is increased. Therefore in context of the relaxed problem (5.2), we need to compute $\mathcal{Y}$ such that it minimizes $F$ and find the range of values of $\log \beta$ for which it remains a minimum, i.e. at every resolution level in this range the optimal centers $\mathcal{Y}$ must satisfy

$$\frac{d}{d\epsilon} F(\mathcal{Y} + \epsilon\Psi)\big|_{\epsilon=0} = 0, \text{and} \tag{5.5}$$

$$H(\mathcal{Y}, \Psi, \beta) := \frac{d^2}{d\epsilon^2} F(\mathcal{Y} + \epsilon\Psi)\big|_{\epsilon=0} > 0, \tag{5.6}$$

for all finite perturbations $\Psi$. The cluster centers $\mathcal{Y}$ ceases to be a minimum of $F$ for a value of $\beta$ when the Hessian (5.6) is no longer positive definite, that is when there exists a perturbation $\Psi$ such that $H(\mathcal{Y}, \Psi, \beta)$ is no longer positive definite. Now it can be shown that for $d(x_i, y_j)$ as squared Euclidean distance the Hessian is

$$\begin{aligned} H(\mathcal{Y}, \Psi, \beta) = &\sum_{\mathbf{y_j}} p_i p(j|i) \boldsymbol{\psi_j}^T \left[ I - 2\beta\, C^k_{\mathcal{X}|\mathbf{y_j}} \right] \boldsymbol{\psi_j} \\ &+ \sum_{i=1}^{N} p_i \Big[ \sum_{\mathbf{y_j}} p(j|i) (\mathbf{x_i} - \mathbf{y_j})^T \boldsymbol{\psi_j} \Big]^2, \end{aligned} \tag{5.7}$$

$$\text{where } C^k_{\mathcal{X}|\mathbf{y_j}} \triangleq \sum_{i=1}^{N} p(i|j) (\mathbf{x_i} - \mathbf{y_j})(\mathbf{x_i} - \mathbf{y_j})^T \tag{5.8}$$

is the *cluster covariance matrix* of the posterior distribution $p(i|j)$ and $I$ is the identity matrix of appropriate dimensions. From (5.7), it is not difficult to show that $H(\mathcal{Y}, \Psi, \beta)$ loses its positivity only when $\det \left[ I - 2\beta\, C^k_{\mathcal{X}|\mathbf{y_0}} \right] = 0$ at some $\mathbf{y_0} \in \mathcal{Y}$ (please refer to supplementary material for proof)[1]; therefore the critical value of $\beta$ beyond which the clustering center $\mathcal{Y} = \{\mathbf{y_j} \in \mathbb{R}^d : 1 \leq$

[1] arXiv:1811.00102

$j \leq k\}$ is no longer a minimum is given by

$$\beta_k = 1 \Big/ \big(2\lambda_{\max}(C^k_{\mathcal{X}|\mathbf{y_0}})\big), \tag{5.9}$$

where $\lambda_{\max}(C^k_{\mathcal{X}|\mathbf{y_0}})$ is the largest eigenvalue of $C^k_{\mathcal{X}|\mathbf{y_0}}$. In fact $\mathbf{y_0}$ is the centroid of that cluster which has the maximum variance, i.e. $\mathbf{y_0} = \arg\max_{\{\mathbf{y_j}\}} \lambda_{\max}(C^k_{\mathcal{X}|\mathbf{y_j}})$. Therefore beyond the critical value $\beta_k$ in (5.9) the number of identifiable cluster increases by one so as to identify a new minimum of $F$ [58]. This makes intuitive sense, since we would expect the clusters with biggest variance to split before the others. The spread directions are indicated by the associated eigenvectors of the covariance matrix, with the largest spread along the eigenvector corresponding to the largest eigenvalue. Therefore using this analysis we can quantify persistence of a clustering solution with $k$ clusters by $v(k) := \log \beta_k - \log \beta_{k-1}$, where $\beta_k$, is the resolution at which the number of distinct clusters increases from $k-1$ to $k$. The *true* number $k_t$ of clusters can be estimated by $\arg\max_k v(k)$.

**Making the persistence independent of the DA algorithm:** Note that, though our quantification of persistence of a clustering solution is motivated from the DA algorithm, we can easily make it algorithm independent by replacing soft associations in (5.8) with hard associations; thus we re-define the cluster covariance matrix (5.8) for clustering solution with $k$ clusters as

$$\bar{C}^k_{\mathcal{X}|\mathbf{y_j}} = \sum_{i=1}^{N} \nu_{ij} (\mathbf{x_i} - \mathbf{y_j})(\mathbf{x_i} - \mathbf{y_j})^T \tag{5.10}$$

where the posterior distribution term $p(i|j) \in [0,1]$ is replaced by $\nu_{ij} \in \{0,1\}$ that represents hard-associations between the vector $\mathbf{x}_i$ and cluster centroid $\mathbf{y}_j$ such that $\nu_{ij} = 1$ if vector $\mathbf{x}_i$ belongs to the cluster $\pi_j$ and zero otherwise. We formally define the true number of clusters $k_t$ in a dataset as

$$k_t := \arg\max_k \Big[ v(k) := \log \bar{\beta}_k - \log \bar{\beta}_{k-1} \Big], \tag{5.11}$$

$$\text{where } \bar{\beta}_k := \Big[ \max_{1 \leq j \leq k} \big[ 2\lambda_{\max}\big(\bar{C}^k_{\mathcal{X}|\mathbf{y_j}}\big) \big] \Big]^{-1} \tag{5.12}$$

is the minimum resolution level at which $k+1$ distinct clusters centroids should be allocated to the dataset.

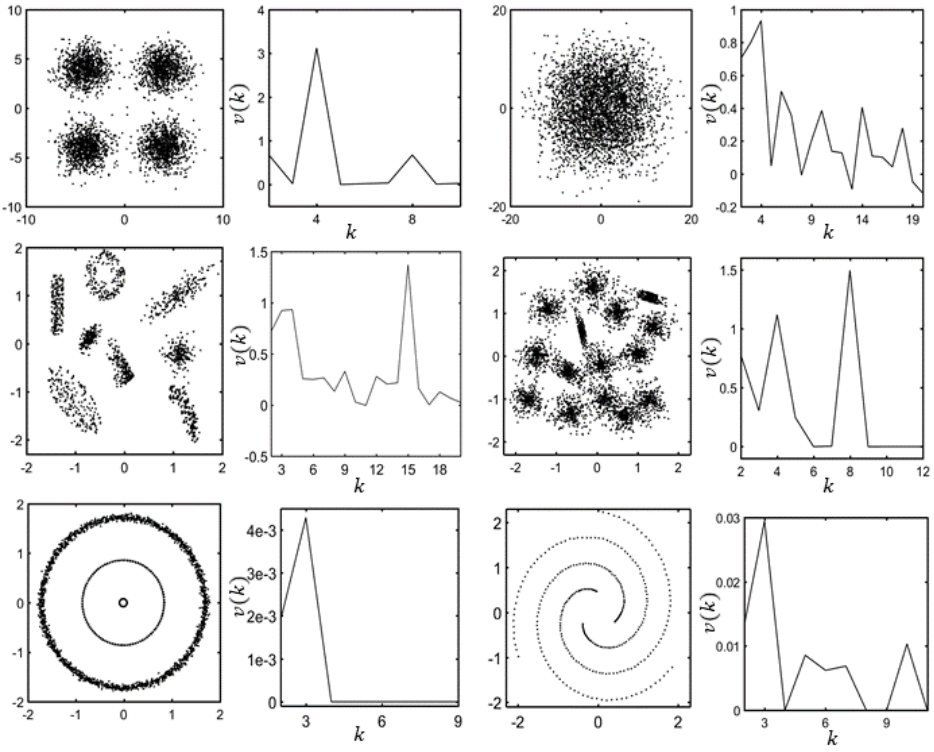


Figure 5.2: Evaluation of our method on a variety of synthetic datasets - (a) Low variance Gaussian, $k_t = 4$, (b) High variance Gaussian, $k_t = 4$, (c) ComboSetting, $k_t = 8$, (d) Synthetic-15 (S15), $k_t = 15$, (e) Concentric rings, $k_t = 3$, and (f) Spirals, $k_t = 3$. Our method predicts the correct number of clusters in each of these scenarios.

**Remark on Scalability and use of** log **in** $v(k)$**:** Note that our proposed method is scalable with respect to size ($N$) of the dataset as it requires computation of the largest eigenvalue of a $d \times d$ covariance matrix $\bar{C}^k_{\mathcal{X}|\mathbf{y_j}}$ which can be computed in $O(d^2)$, where $d$ is the dimension of the feature-space. Our notion of persistence captures, by what factor one should scale resolution to obtain more clusters. This factor is simply expressed as a difference using log for better visualization. Also the range of resolutions is typically very large (each data point is a highest resolution cluster to the entire dataset being the lowest resolution cluster); therefore log function discriminates this range better.

Figures 5.1(a3) and 5.1(b3) illustrates our method for determining number of true clusters on two datasets considered in Figures 5.1(a1) and 5.1(b1) respectively. Observe that the quantities $\log \bar{\beta}_3 - \log \bar{\beta}_2 \gg \log \bar{\beta}_k - \log \bar{\beta}_{k-1}$ and $\log \bar{\beta}_9 - \log \bar{\beta}_8 \gg \log \bar{\beta}_k - \log \bar{\beta}_{k-1}$ for all $k \neq 3$ and $k \neq 9$ in both the figures. Further, we observe in the Figure 5.1(a3) that $\log \bar{\beta}_3 - \log \bar{\beta}_2 >$

**Algorithm 6** main($\mathcal{X}$,$k_{\max}$)

1. Initialize $k = 1$.
2. Run a clustering algorithm on $\mathcal{X}$ with $k$ clusters and compute $\bar{\beta}_k$ using (5.12).
3. $k \longleftarrow k + 1$. Go to step 2. Stop if $k = k_{\max} + 1$.
4. Choose $k_t$ using (5.11).

$\log \bar{\beta}_9 - \log \bar{\beta}_8$ which indicates $k_t = 3$ (using (5.10)) for the dataset in Figure 5.1(a1). Similarly we observe in Figure 5.1(b3) that $\log \bar{\beta}_9 - \log \bar{\beta}_8 > \log \bar{\beta}_3 - \log \bar{\beta}_2$ which indicates $k_t = 9$ for the dataset in Figure 5.1(b1). Again, these inferences agree with our intuition from visual inspection of these datasets.

**Extending to nonlinearly separable data:** The method proposed above works well for linearly separable data. However, many problems such as shape clustering or clustering with pairwise distances often consist of data points that are not separable linearly. We use kernel trick [111] to overcome this issue. Accordingly, we map data points $\mathbf{x_i}$ to an abstract higher-dimensional space (where data-points are linearly separable) through a suitably chosen kernel function $\phi(\cdot)$. While an explicit representation of kernel function $\phi(\cdot)$ is unknown, the inner-products $\phi(\mathbf{x_i})^T\phi(\mathbf{x_j})$ are known as elements of a kernel matrix $\mathcal{K}$ [111]. Thus, in order to identify the clustering solution with the correct number of clusters in a nonlinearly separable dataset, one must evaluate the largest eigenvalue of the kernel data covariance matrix $\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}}$ defined as:

$$\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}} = \sum_{\mathbf{x_i} \in \pi_j} \left(\phi(\mathbf{x_i}) - \mathbf{y_j}\right) \left(\phi(\mathbf{x_i}) - \mathbf{y_j}\right)^T, \tag{5.13}$$

where $\pi_j$ denotes the $j$-th cluster. Computation of eigenvalues of $\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}}$ is not straightforward (since $\phi$ is unknown), however, we make use of the following lemma in order to obtain the spectral values of $\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}}$.

**Lemma 2.** *Let $\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}}$ be the cluster covariance matrix as defined in (5.13). Then $\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}}$ and $A = [A_{kl}]$ share the same non-zero eigenvalues, where $A_{kl} = \left(\phi(\mathbf{x_k}) - \mathbf{y_j}\right)^T \left(\phi(\mathbf{x_l}) - \mathbf{y_j}\right)$.*

**Proof**: Please refer to the supplementary material[1] for proof of the above

Table 5.1: Comparing algorithms on a variety of standard and synthetic datasets

| Algorithm | Low Variance | High Variance | Combo-Setting | Wisconsin | Yeast | Glass | Leaves |
|---|---|---|---|---|---|---|---|
| | $k_t = 4$ | $k_t = 4$ | $k_t = 8$ | $k_t = 2$ | $k_t = 10$ | $k_t = 6$ | $k_t = 100$ |
| gap-statistic | **4** | 1 | 27 | 12 | 49 | 39 | 117 |
| $X$-means | 1 | 1 | 25 | 6 | 47 | 1 | 219 |
| $G$-means | **4** | 68 | 33 | 83 | 56 | 15 | 66 |
| $PG$-means | **4** | 3 | 12 | 6 | 3 | 7 | Error |
| dip-means | **4** | 1 | **8** | 11 | 1 | 1 | 3 |
| Info. Th. | **4** | 3 | **8** | **2** | 2 | **6** | 99 |
| kernel dip-means | - | - | - | - | - | - | - |
| Our Method | **4** | **4** | **8** | **2** | **10** | **6** | **100** |

| Algorithm | Wine | Iris | Banknote | Thyroid | Birch1 | Concentric Rings | Spirals |
|---|---|---|---|---|---|---|---|
| | $k_t = 3$ | $k_t = 3$ | $k_t = 2$ | $k_t = 3$ | $k_t = 100$ | $k_t = 3$ | $k_t = 3$ |
| gap-statistic | **3** | **3** | 58 | 17 | Error | n/a | n/a |
| $X$-means | 40 | 12 | 230 | 1 | 1 | n/a | n/a |
| $G$-means | 2 | 4 | 57 | 7 | 1953 | n/a | n/a |
| $PG$-means | 1 | 2 | 35 | **3** | 32 | n/a | n/a |
| dip-means | 1 | 2 | 4 | 1 | Error | n/a | n/a |
| Info. Th. | **3** | 2 | 5 | **3** | **100** | n/a | n/a |
| kernel dip-means | - | - | - | - | - | **3** | 1 |
| Our method | **3** | 2 | **2** | **3** | **100** | **3** | **3** |

lemma.

Note that from (5.3), the cluster center $\mathbf{y_j} = \frac{1}{|\pi_j|}\sum_{\mathbf{x_i}\in\pi_j}\phi(\mathbf{x_i})$. Thus elements of $A$ are known in terms of the elements of kernel matrix, and hence the spectral values of $\bar{C}^k_{\phi(\mathcal{X})|\mathbf{y_j}}$ can be easily obtained.

We demonstrate the efficacy of our proposed metric using simulations on synthetic and standard datasets in the experiments section. Additionally, we analytically solve for the persistence of clustering solutions for an example problem as shown in Figure 5.3. The Figure illustrates two equally sized circular clusters ($k_t = 2$) with uniform distribution. One can easily compute the cluster covariance matrices for clustering solutions (as given by k-means) at various $k$'s and show that $v(2) > \eta \; \forall \; \eta \in \{v(3), v(4), v(5), v(6)\}$; thereby implying that for the dataset in Figure 5.3, $k = 2$ is a *more* natural choice of the number of clusters than $k = 3, 4, 5$ and 6. Please refer to the supplementary material A.4 for the proof.

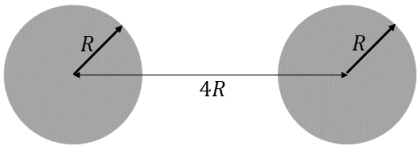


Figure 5.3: Illustrates two circular clusters of radius $R$ with uniformly distributed data-points.

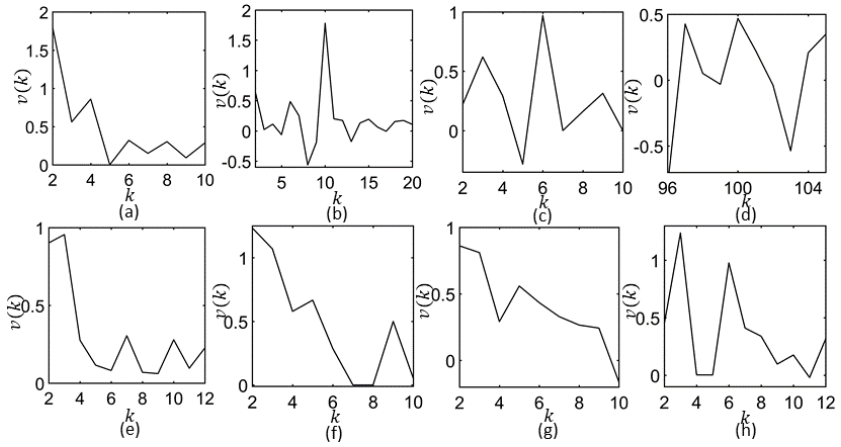


Figure 5.4: Illustration of performance of our method on high-dimensional datasets - (a) Wisconsin ($d = 9$, $N = 681$, $k_t = 2$), (b) Yeast ($d = 8$, $N = 1484$, $k_t = 10$), (c) Glass ($d = 9$, $N = 214$, $k_t = 6$) (d) Leaves ($d = 64$, $N = 1600$, $k_t = 100$) (e) Wine ($d = 13$, $N = 178$, $k_t = 3$) (f) Iris ($d = 4$, $N = 150$, $k_t = 3$) (g) Banknote Authentication ($d = 4$, $N = 1372$, $k_t = 2$) (h) Thyroid ($d = 5$, $N = 215$, $k_t = 3$) . Note that, except for the Iris dataset, $v(k_t)$ is maximum in all the plots.

## 5.3 Experiments

In this section we employ the notion of persistence in estimating the true number of clusters in synthetic as well as standard datasets from the literature [117], [118] and provide comparisons with the gap-statistic method [100], dip-means [109], $X$-means [102], $G$-means [107], $PG$-means [101] algorithms and the information theoretic approach [106]. We observe that our proposed method outperforms the existing methods on various standard datasets as well as on synthetic datasets with acute overlap between two clusters.

As the method proposed in this chapter evaluates a clustering solution for its persistence, any suitable clustering algorithm can be employed to obtain these clustering solutions. Note that irrespective of the criteria or metric used, a stable and good clustering solution at each $k$ is a prerequisite to correctly estimating the true number of clusters in a dataset. Since the $k$-means algorithm is ubiquitous in the data science literature, in all our simulations on linearly separable datasets we use $k$-means algorithm with multiple runs

to determine the clustering solution at each value of $k$. For the purpose of estimating the true number of shape clusters we use the spectral clustering algorithm [119] to obtain the clustering solutions at various $k$'s. Also note that for all the simulations demonstrated in this section we normalize every dataset to mean zero and standard deviation one.

Figure 5.2 demonstrates multiple instances of synthetic datasets, and the corresponding plots of $v(k)$ versus $k$, where $k$ is the choice for the number of clusters. Figure 5.2a illustrates a mixture of four well separated Gaussian distributions. The corresponding plot of $v(k)$ versus $k$ shows a clear peak at $k = 4$. Figure 5.2b illustrates another mixture of four Gaussian distributions with very high variances as compared to the Figure 5.2a. As can be seen, the sampled data is highly overlapping and it seems that the entire dataset is sampled from a single distribution. As seen in the corresponding $v(k)$ versus $k$ plot in the Figure 5.2b, our method exhibits a maximum value of $v(k)$ at $k = 4$. On the other hand, the algorithms such as $G$-means, $PG$-means and dip-means fail to estimate the true number of clusters in this case (as shown in Table 5.1), even though this dataset satisfies the assumptions required by these algorithms. In Figure 5.2c, the dataset is a mixture of well separated eight non-uniform clusters. The corresponding plot of $v(k)$ versus $k$ determines two distinguishable peaks at $k = 4$ and $k = 8$, although the peak is larger at $k = 8$ and denotes the true number of clusters in this dataset.

Similar conclusions are observed for the linearly separable S15 dataset (a mixture of 15 Gaussian distribution [4]) in Figure 5.2(d), and other standard high-dimensional datasets - Wisconsin ($d = 9$, $N = 681$, $k_t = 2$), Yeast ($d = 8$, $N = 1484$, $k_t = 10$), Glass ($d = 9$, $N = 214$, $k_t = 6$), Leaves ($d = 64$, $N = 1600$, $k_t = 100$), Wine ($d = 13$, $N = 178$, $k_t = 3$), Iris ($d = 4$, $N = 150$, $k_t = 3$), Banknote Authentication ($d = 4$, $N = 1372$, $k_t = 2$), Thyroid ($d = 5$, $N = 215$, $k_t = 3$), shown in Figure 5.4, where our proposed method estimates the true number of clusters appropriately. In particular, note that our proposed metric estimates the true number of clusters for a very high-dimensional leaves ($d = 64$, $k_t = 100$) dataset and a large Birch1 ($N = 100,000$, $k_t = 100$) dataset. All the other methods, except Information Theoretic approach on Birch1 dataset, fail to determine the true number of clusters in these two datasets. Table 5.1 compares our method to the gap-statistic, $X$-means, $G$-means, $PG$-means, dip-means, kernel dip-means algorithms and information

theoretic approach of determining the true number of clusters in the dataset. We observe that our method estimates the correct value of $k_t$ even for the high-variance Gaussian distribution Figure 5.2(b), yeast dataset, banknote authentication dataset and a high-dimensional Leaves dataset ($d = 64$) where all the other methods fail to correctly estimate the true number of clusters. As previously noted, the proposed metric is scalable with respect to the size of the datasets and involves eigenvalue computation of a $d \times d$ cluster covariance matrix, where $d$ is the dimension of datapoints in the dataset. For the Iris dataset most of the methods, including ours, estimate the true number of clusters to be 2. This is because the two of three clusters in the Iris dataset have significant overlap with each other. However, the gap-statistic method outperforms all the others and estimates correctly the true number of clusters in the Iris dataset.

As described earlier, our technique extends to non-linearly separable data too. Figure 5.2e and 5.2f illustrate two non-linearly separable datasets. We use the spectral clustering algorithm [119] and determine the similarity graph using the Gaussian similarity function, $s(x_i, x_j) = \exp(-\|x_i - x_j\|^2/(2\sigma^2))$, where the $\sigma$ parameter is set to be 0.01 and 0.08 for (e) concentric rings, and (f) spirals, respectively. Table 5.1 shows comparison with the kernel dip-means [109] algorithm on these two non-linearly separable datasets. Note that the latter fails to correctly estimate the true number for clusters on the spiral dataset.

## 5.4 Summary

This chapter demonstrates the capabilities of the phase transitions in designing hyper-parameters related to the combinatorial optimization problems. In particular, we deal with the data clustering problem in this chapter. Typically clustering algorithms provide clustering solutions with prespecified number of clusters. The lack of a priori knowledge on the true number of underlying clusters in the dataset makes it important to have a metric to compare the clustering solutions with different number of clusters that will be supportive in estimating the number of clusters underlying a dataset. In this chapter quantified a notion of *persistence* of clustering solutions that enables comparing solutions with different number of clusters. The persistence

relates to the range of data-resolution scales (as captured by the analytical conditions on phase transitions) over which a clustering solution persists. Our simulations demonstrate that the datasets where *natural* clusters are a priori known, the clustering solutions that identifies the natural clusters are *most persistent*. Detailed experiments on a variety of standard and synthetic datasets demonstrate that the proposed persistence-based indicator outperformed the existing approaches, such as, gap-statistic method, $X$-means, $G$-means, $PG$-means, dip-means algorithms and information-theoretic method, in accurately identifying the clustering solutions with true number of clusters.

# CHAPTER 6

# ON THE CHOICE OF NUMBER OF SUPERSTATES IN THE AGGREGATION OF MARKOV CHAINS

## 6.1 Introduction

In this chapter we demonstrate the utility of phase transitions in characterizing the aggregated representations of a Markov chain obtained at different number of superstates (i.e., the states of the aggregated chains). We subsequently use this characterization in estimating the *best choice* for the number of superstates to be considered in an aggregated representation. Our proposed methodology is rooted in MEP-based framework in [1] that addresses the problem of Markov chain aggregation. For coherence and understandability, we build analogy with clustering problem addressed in Chapter 5.

Markov chains provide a mathematical model to study many real-world stochastic processes, such as population dynamics, cruise control, transportation systems, and queueing networks [120]. However, Markov chain models for several complex systems such as applications originating in network analysis [121], neuroscience [122], and economics [123] require large number of states where analyzing them is challenging and inefficient; thus, laying foundation for Markov chain aggregation techniques.

The two-fold objective of the aggregation problem is to (a) group *similar* states in the Markov chain and represent them as a single *superstate* in the aggregated model, and (b) determine the state transition probability matrix of the aggregated model. Prior works in literature [124–129] aim at developing an appropriate *dissimilarity* metric to compare the differently-sized state transition probability matrices of the Markov chain and its aggregated model. The algorithms proposed in [1, 129, 130] determine the aggregated models at a *pre-specified* number of superstates such that the dissimilarity is minimized. However, there is scant literature on determining the appropriate number of superstates in the aggregated chain. One of the recent works [131] provides

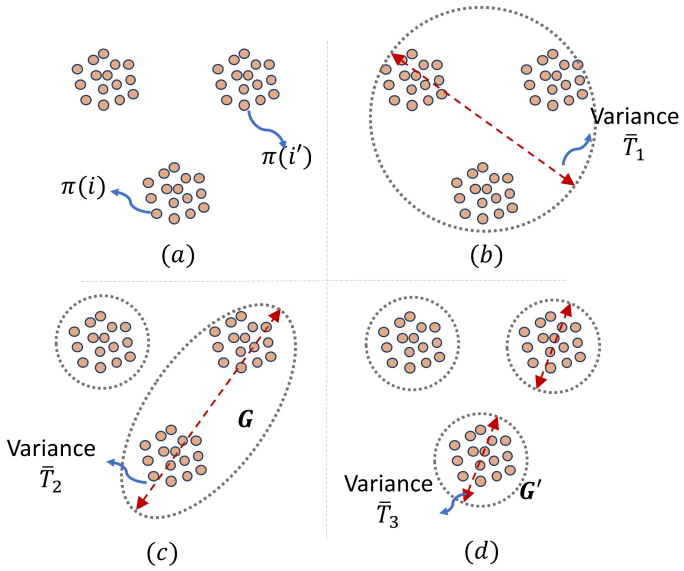


Figure 6.1: (a) Represents the states in the original Markov chain. (b) Aggregated model with 1 superstates, i.e. all states in the same group. (c) Illustrates the 2 superstates, i.e., 2 groups of related states. (c) Illustrates the 3 distinct groups of related states.

an aggregation algorithm for the class of nearly completely decomposable (NCD) [132] Markov chains, that also estimates the number of superstates underlying these particular NCD chains.

In this work we provide a methodology to estimate the number of superstates underlying any general Markov chain. Our proposed approach is independent of the algorithm being used to obtain the aggregated representations of the given Markov chain. We devise a method to quantitatively compare the aggregated models of different sizes that describe the same Markov chain. We demonstrate its utility in determining an appropriate choice for the number of superstates among the *given* aggregated models. More precisely, we develop and quantify a notion of *marginal return* that compares the aggregated models of different sizes. We demonstrate via simulations that the aggregated model with the largest marginal return estimates the superstates *underlying* the original Markov chain. The corresponding size of this aggregated model provides an appropriate choice for the number of superstates.

The primary goal in the aggregation of Markov chain is akin to the data-clustering problem. The former identifies groups of *similar* states, whereas the latter determines the groups (or clusters) of *similar* data points as seen in the previous chapter. Thus, one can view the problem of determining the appropriate number of superstates to be analogous to estimating the *true* (or

natural) number of clusters in a dataset. As already seen, the latter is a well-studied problem in the data clustering literature [100–102, 106, 107, 109, 133]. However, the methods proposed in the context of data clustering are not directly applicable to the aggregation of Markov chains. This is due to the inherent differences in the two problems. The data clustering problem identifies groups (or clusters) of similar data points lying in the euclidean space where the cost function is typically 2-norm. On the other hand, the Markov chain aggregation requires determining the groups of similar states whose transition probabilities lie in the probability space, and the cost is typically in terms of the relative entropy of the transition probabilities. Nonetheless, the methods proposed in the context of data clustering can still be helpful in devising *procedures* to compare aggregated chains of different sizes. These procedures can then be used to estimate the appropriate number of superstates. In fact, this chapter builds upon the previous chapter that develops and quantifies the notion of *persistence* of a clustering solution, and utilizes it to estimate the natural number of clusters in the dataset. Below we provide a qualitative and quantitative descriptions of our notions of marginal return and heterogeneity.

**Qualitative description of marginal return:** As stated above we exploit the work done in Chapter 5 to qualitatively elucidate our notion of marginal return and the heterogeneity of a superstate. For clarity, we first give a brief description of the idea presented in the previous chapter. Thereafter, we abstract it to the aggregation of Markov chains. Consider the dataset $\mathcal{X} = \{x_i\}$ illustrated in the Figure 6.1(a). It is apparent from the figure that the dataset comprises of 3 groups of similar datapoints, i.e., 3 *natural* clusters. Figures 6.1(b), 6.1(c), and 6.1(d) illustrate the corresponding clustering solutions obtained with 1, 2, and 3 number of clusters, respectively. $\bar{T}_1$, $\bar{T}_2$, and $\bar{T}_3$ denote the *variance* (or, spread of data points) within the largest cluster in the respective clustering solutions.

Note that there is a significant reduction in the variance of the largest cluster from $\bar{T}_2$ in Figure 6.1(c) (with 2 clusters) to $\bar{T}_3$ in Figure 6.1(d) (with 3 clusters). Contrastingly, the variance $\bar{T}_1$ in Figure 6.1(b) (with 1 cluster) is comparable to the variance $\bar{T}_2$ in Figure 6.1(c) (with 2 clusters). The significant reduction in the variance from $\bar{T}_2$ to $\bar{T}_3$ coincides with the progression from the clustering solution with 2 clusters (in Figure 6.1(c)) to the clustering solution with 3 clusters (in Figure 6.1(d)) - where 3 is the *natural* number of

clusters in Figure 6.1(a). The work done in Chapter 5 exploits the above observation. It first quantifies variance in terms of the largest eigenvalue of the *cluster covariance matrix* [58]. Subsequently, it proposes that the clustering solution with *true* number of clusters exhibits *a significant drop in the variance of the largest cluster* when compared to the clustering solution with one less cluster. In Chapter 5, we referred this to as the clustering solution with true number of clusters being more *persistent* than other clustering solutions. Our simulations in the previous chapter demonstrate that the above idea significantly outperforms popular benchmark methods [101,102,106,107,109] in literature when applied to diverse synthetic and standard datasets.

We now extend the above idea to the aggregation of Markov chains. Analogous to the *variance* of a cluster in the previous chapter we establish the *heterogeneity* of a superstate in the context of Markov chains; wherein, heterogeneity quantifies the dissimilarity within the group of related states (i.e., states represented by the same superstate). Subsequently, we define the *marginal return* of an aggregated model as the *reduction in the heterogeneity* of the largest superstate in comparison to the aggregated model with one less superstate. We illustrate the idea further as follows. Consider a set of $K$ aggregated models each with different number $k \in \{1, \ldots, K\}$ of superstates. Analogous to the above case of data clustering we propose that *the aggregated model with the largest marginal return estimates the true number $k_t$ of superstates*. The idea is that there is a significant drop in the heterogeneity of the largest superstate in the aggregated models $\bar{M}_{k_t-1}$ with $k_t - 1$ superstate in comparison to the aggregated model $\bar{M}_{k_t}$ with $k_t$ superstates. This could be owing to several reasons. For instance, the largest superstate in $\bar{M}_{k_t-1}$ combines *distinct* groups of *similar* states in the original Markov chain resulting into much larger heterogeneity, whereas the aggregated model $\bar{M}_{k_t}$ possibly identifies each group of *similar* states distinctly thus exhibiting much lesser heterogeneity. As in the case of data-clustering, the simulations on multiple synthetic and real-world Markov chains demonstrate the efficacy of the above proposed methodology.

**Quantifying heterogeneity and marginal return:** The quantification of persistence of clustering solutions and the subsequent characterization of natural number of clusters in the Chapter 5 is motivated from the Deterministic Annealing (DA) algorithm presented in [58]. Deterministic Annealing (DA) is a Maximum Entropy Principle [10] based clustering algo-

rithm. It operates by determining a single cluster at large values of annealing parameter $T$. As the annealing proceeds (i.e., $T$ decreases) the number of distinct clusters increases at specific instances referred to as *phase transitions* [58]. The notion of persistence and the subsequent characterization of natural clusters in Chapter 5 are motivated by the analytical conditions on the annealing parameter $T$ that govern the phase transitions in DA.

The work done in [1] presents a deterministic annealing (DA) approach to the aggregation of Markov chains. Here, the problem formulation and the associated cost functions are fundamentally distinct from the data clustering problem in [58]; however, the aggregation algorithm presented in [1] undergoes similar phase transition phenomenon. In particular, the algorithm begins with determining an aggregated model with a single superstate at large values of annealing parameter $T(\rightarrow \infty)$. As the annealing proceeds (i.e., $T$ decreases), the number of superstates in the model increases at specific critical values $T = T_{\text{cr}}$ resulting into a phase transition. Here, we explicitly determine the analytical conditions on the annealing parameter $T$ at which the phase transition phenomenon occurs in [1]. Thereafter, analogous to the clustering case discussed in the previous chapter, we quantify our notion of marginal return and the heterogeneity of a superstate based on these analytical conditions. Further, our characterization of the best choice for the number of superstates in the aggregated model is also motivated from the phase transition phenomenon. We substantiate on the exact expressions of heterogeneity and marginal return in the later sections.

Our simulations demonstrate that the marginal return value $\nu(k_t)$ is as high as 8 to 60 times the second largest value for Markov chains where the true number $k_t$ is discernible from the heatmap of the associated transition matrices $\Pi$. For Markov chains where true number $k_t$ is not discernible from the heatmaps, i.e. the underlying true number $k_t$ of superstates is not visually apparent, the marginal return value $\nu(k_t)$ is still the largest and as high as 1.2 to 5 times the second largest value. In our simulations on Markov chains that model real-life scenarios - (a) emotion transitions in brain networks, and (b) letter bigram dataset, we observe that marginal return provides meaningful insights. In particular, it captures the distinct types of emotions (positive and negative) in the former, and the different types of English alphabets (vowels and consonants) in latter.

## 6.2 Aggregation of a Markov Chain

In this section we briefly illustrate the Markov chain aggregation framework adopted in [1]. As stated above this framework forms the foundation for our quantification of heterogeneity of a superstate, and the marginal return of an aggregated model. Consider a Markov chain $(X, \Pi)$ with state space $X = \{x_i : 1 \leq i \leq N\}$, and the transition probability matrix $\Pi = (\pi_{ij}) \in \mathbb{R}^{N \times N}$. The objective is to determine an aggregated representative chain $(Y, \Psi, \Phi)$ with $M \ll N$ (super-) states, where $Y = \{y_j : 1 \leq j \leq M\}$ denotes the state space, $\Psi = (\psi_{jk}) \in \mathbb{R}^{M \times M}$ denotes the transition probability matrix, and $\Phi : X \to Y$ is the associated *partition* function such that the state $x_i \in X$ is represented by the superstate $\Phi(x_i) \in Y$.

The framework operates by associating a distribution vector $z(j) = (z_{j1}, \ldots, z_{jN}) \in \mathbb{R}^N$ to each superstate $y_j \in Y$. This distribution vector $z(j)$ is an auxiliary variable that facilitates determining the transition probability matrix $\Psi$ of the aggregated chain. It captures the relation of the superstate $y_j$ to all the $N$ states in $X$, and consequently helps determine $y_j$'s relation (transition probability) with other superstates in $Y$. Please refer to Figure 6.2 for a graphical interpretation of $z(j)$. Without loss of generality $\sum_{k=1}^{N} z_{jk} = 1$ and $z_{jk} \geq 0$ for all $k$. In the original Markov chain $(X, \Pi)$ the transition probability vector $\pi(i) = (\pi_{i1}, \ldots, \pi_{iN})$ captures the relation of the state $x_i$ to all the $N$ states in $X$. Thus, the distance $d(x_i, y_j)$ between the state $x_i$ and the superstate $y_j$ is given by the *relative entropy* between probability vector $\pi(i)$ and distribution vector $z(j)$, i.e.,

$$d(x_i, y_j) = \sum_{k=1}^{N} \pi_{ik} \log \frac{\pi_{ik}}{z_{jk}} \quad \forall \;\; 1 \leq i \leq N, 1 \leq j \leq M. \tag{6.1}$$

Thereafter, the framework determines a set of $M$ distribution vectors $\{z(j)\}_{j=1}^{M}$ such that the cumulative distance of each state $x_i$ to its closest superstate $y_j := \Phi(x_i)$ is minimized, i.e., it solves the optimization problem

$$\begin{aligned} \min_{Z,\Phi} \quad & D(\Pi, Z, \Phi) = \sum_{i=1}^{N} \rho_i \; d(x_i, \Phi(x_i)) \\ \text{subject to} \quad & z_{jk} \geq 0 \; \forall \; j,k \text{ and } z(j)^\top \mathbf{1}_N = 1 \; \forall \; j, \end{aligned} \tag{6.2}$$

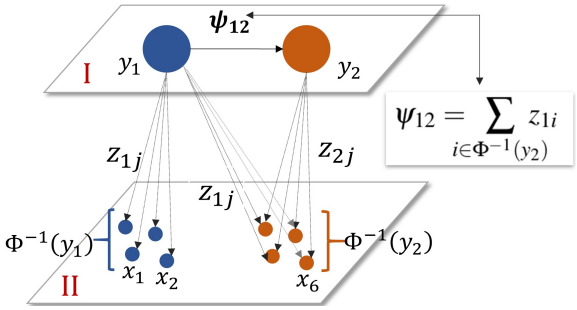


Figure 6.2: The plane I represents the aggregated model for the Markov chain in the plane II. Each superstate $y_j$ in I is associated to each state $x_i$ in II via the weight $z_{ji}$. The set of states represented by superstate $y_j$ is given by $\Phi^{-1}(y_j)$.

where $\rho_i$ is a given relative weight of the state $x_i$ (for instance, $\rho_i = 1/N$). Subsequently, [1] determines the transition matrix $\Psi = (\psi_{jk})$ of the aggregated chain in terms of the distribution vectors $\{z(j)\}_{j=1}^{M}$. In particular, the transition probability $\psi_{jk}$ from the superstate $y_j$ to $y_k$ is the cumulative sum of the weights $z_{ji}$'s from superstate $y_j$ to all states $x_i$ that are represented by the superstate $y_k$ (see Figure 6.2), i.e.,

$$\psi_{jk} = \sum_{x_i \in \Phi^{-1}(y_k)} z_{ji} \ \forall \ 1 \leq j,k \leq M. \tag{6.3}$$

The partition function $\Phi : X \to Y$ in (6.2) associates the state $x_i$ to a *particular* superstate $y_j = \Phi(x_i) \in Y$. In its solution methodology, [1] replaces this partition function by *soft* partition weights $p_{j|i} \in [0,1]$ that determine the association of the state $x_i$ to different superstates $\{y_j \in Y\}$. Without loss of generality $\sum_j p_{j|i} = 1$ and $p_{j|i} \geq 0$ for all $i$, $j$. Thereafter, Maximum Entropy Principle (MEP) is utilized to design the partition weights $\{p_{j|i}\}$. In particular, [1] determines the most *unbiased* partition weights $\{p_{j|i}\}$ and the distribution vectors $\{z(j)\}$ that maximize the corresponding Shannon entropy $H$ such that the expected cost function $\mathbb{E}_p[D]$ attains a pre-specified value $d_0$, i.e, the framework solves

$$\begin{aligned} \max_{\{p_{j|i}\},\{z(j)\}} \quad & H = -\sum_{i=1}^{N} \rho_i \sum_{j=1}^{M} p_{j|i} \log p_{j|i} \\ \text{subject to} \quad & \mathbb{E}_p[D] := \sum_{i=1}^{N} \rho_i \sum_{j=1}^{M} p_{j|i} d(x_i, y_j) = d_0 \\ & z_{jk} \geq 0, \ z(j)\mathbf{1}_N = 1 \ \forall j,k. \end{aligned} \tag{6.4}$$

Minimizing (local) the Lagrangian $\mathcal{L} = (\mathbb{E}_p[D] - d_0) - TH$ (where $T$ is the Lagrange parameter) of above optimization problem with respect to $\{p_{j|i}\}$ and $\{z(j)\}$ results into

$$p_{j|i} = \frac{\exp\{-(1/T)d(x_i, y_j)\}}{\sum_k \exp\{-(1/T)d(x_i, y_k)\}}, \quad Z = P^t \Pi \tag{6.5}$$

$$\text{where } [P]_{ij} = \frac{\rho_i p_{j|i}}{\sum_t \rho_t p_{j|t}} \text{ and } Z = [z(1), \ldots, z(M)]^\top. \tag{6.6}$$

The resulting Lagrangian $\mathcal{L}(Z)$ is given by

$$\mathcal{L}(Z) = -T \sum_{i=1}^{N} \rho_i \log \sum_{j=1}^{M} \exp\Big\{ -\frac{1}{T} \sum_{k=1}^{N} \pi_{ik} \log \frac{\pi_{ik}}{z_{jk}} \Big\}. \tag{6.7}$$

It is known from the sensitivity analysis [10] that the large (small) values of $d_0$ in (6.4) is analogous to large (small) values of the Lagrange parameter $T$.In fact, the optimization problem in (6.4) is repeatedly solved at decreasing values of $T$ (or equivalently, decreasing values of $d_0$). At large values of $T \to \infty$, the Lagrangian $\mathcal{L}$ is dominated by the convex function $-H$, the partition weights $\{p_{j|i}\}$ in (6.5) are uniformly distributed ($p_{j|i} = 1/M$), and all the distributions $\{z(j)\}$ in (6.5) are *co-incident*, i.e., effectively *one* distinct superstate is obtained at large values of $T$. As $T$ decreases further the Lagrangian is more and more dominated by the cost function $\mathbb{E}_p[D]$, the partition weights $\{p_{j|i}\}$ are no longer uniform, and the coincident distributions $\{z(j)\}$ split into distinct groups, i.e., the effective number of distinct superstates increases.

Here we exploit this hierarchical splitting to define the notions of heterogeneity and the marginal return. Insights from the simulations of the algorithm in [1] demonstrate that there exist certain critical temperature $T = T_{\text{cr}}$ values at which the solution undergoes the phenomenon of *phase transition*. As briefly illustrated in Section 6.1, this phenomenon is characterized by the increase in the number of distinct distributions in $\{z(j)\}$; or, equivalently increase in the number of distinct superstates. These critical temperatures $T_{\text{cr}}$'s occur when the critical point $Z^*$ of the Lagrangian $\mathcal{L}$, given by $\frac{\partial \mathcal{L}(Z^*)}{\partial Z} = 0$ is no longer the local minima, that is, when for at least one perturbation

direction $\Psi = [\psi_1, \ldots, \psi_M]^\top \in \mathbb{R}^{M \times N}$, the Hessian $\mathcal{H}(Z^*, P^*, \Psi, T) :=$

$$\begin{aligned}\lim_{\epsilon \to 0} \frac{\partial^2 \mathcal{L}(Z^* + \epsilon\Psi)}{\partial \epsilon^2} =& \sum_{j=1}^{M} q_j \psi_j^\top (\Lambda_T(j) - \frac{1}{T} C_T(j)) \psi_j \\ &+ T \sum_{i=1}^{N} \Big( \sum_{j=1}^{M} p_{j|i} \Big[ \frac{\pi(i).}{z^*(j)} \Big]^\top \psi_j \Big)^2, \end{aligned} \tag{6.8}$$

is no longer positive. Here, $q_j = \sum_{i=1}^{N} \rho_i p_{j|i}$.

$$\begin{aligned}\Lambda_T(j) =& \operatorname{diag}\left\{ \frac{\sum_{i=1}^{N} p_{i|j} \pi(i).}{z^*(j).^2} \right\}, \\ C_T(j) =& \sum_{i=1}^{N} p_{i|j} \Big[ \frac{\pi(i) - z^*(j).}{z^*(j)} \Big] \Big[ \frac{\pi(i) - z^*(j).}{z^*(j)} \Big]^\top, \end{aligned} \tag{6.9}$$

and $p_{i|j} = (\rho_i p_{j|i} / \sum_t \rho_t p_{j|t})$ is the posterior distribution. We derive the expression for critical temperature $T_{cr}$ as below.

**Theorem 13.** *The value of critical temperature $T_{cr}$ at which the the Hessian $\mathcal{H}(Z^*, P^*, \Psi, T)$ in (6.8) is no longer positive for some perturbation direction $\Psi$, and $Z^*$ in (6.5) undergoes phase transition is given by*

$$T_{cr} := \max_{1 \le j \le M} \Big[ T_{cr,j} \Big], \text{ where } T_{cr,j} = \lambda_{\max}\big(\mathcal{C}_T(j)\big), \text{ and} \tag{6.10}$$

$$\mathcal{C}_T(j) = \sum_{i=1}^{N} [P]_{ij} \Big[ \Theta^\top \frac{\pi(i) - z^*(j).}{z^*(j)} \Big] \Big[ \Theta^\top \frac{\pi(i) - z^*(j).}{z^*(j)} \Big]^\top. \tag{6.11}$$

*Here, $\lambda_{\max}(\cdot)$ is the largest eigenvalue, $\frac{a.}{b}$ denotes element-wise division of vectors $a$ and $b$, and $\Theta \in \mathbb{R}^{N \times N-1}$ is a parameter that arises due to the constraint $z(j)^\top \mathbf{1}_N = 1 \ \forall \ 1 \le j \le M$ in (6.2) and (6.4); see A.5 for exact $\Theta$, and further details.*

*Proof.* Please refer to the A.5. □

*Interpreting $T_{cr}$, $T_{cr,j}$, and $\mathcal{C}_T(j)$:* As stated earlier, the annealing temperature $T := T_{cr}$ (computed in (6.10)-(6.11)) marks the increase in the number of distinct superstates in the aggregated chain, i.e., it characterizes the phase transitions in the DA-based algorithm [1]. However, the expressions in (6.10)-(6.11) are also further interpretable beyond the current context of phase transitions. For instance, $\mathcal{C}_T(j)$ in (6.11) is a *soft co-variance matrix*

of the posterior distribution $[P]_{ij}$ corresponding to the superstate $y_j \in Y$. Since, $T_{cr,j}$ in (6.10) is the maximum eigenvalue of $\mathcal{C}_T(j)$ it captures the *maximum variance* within the transition probabilities $\{\pi(i)\}$ of the states $\{x_i\}$ represented by the superstate $y_j$. In other words, $T_{cr,j}$ can be interpreted as a *measure of heterogeneity* within the states represented by the superstate $y_j$, and $T_{cr}$ can be interpreted as the maximum heterogeneity among all the superstates $\{y_j \in Y\}$.

The soft co-variance matrix $\mathcal{C}_T(j)$ in (6.11) depends on the partition weights $[P]_{ij}$ and distribution vectors $Z = P^\top \Pi$ in (6.6) that result from the aggregation algorithm in [1] − making the above quantification $T_{cr,j}$ of heterogeneity dependent on the algorithm. In the following section, we adapt $\mathcal{C}_T(j)$ to incorporate aggregated chains irrespective of the algorithm used to determine them. In particular, for a given aggregated representation $(Y, \Psi, \Phi)$ of a Markov chain $(X, \Pi)$, we replace the soft partitions $[P]_{ij}$ in $\mathcal{C}_T(j)$ with the hard-partitions prescribed by the partition function $\Phi : X \to Y$. Subsequently, we provide the expression for *marginal return*, and the Algorithm 7 that estimates a choice for the number of superstates to be considered in the aggregated chain.

## 6.3 Covariance Matrix and Marginal Return

Consider a Markov chain $(X, \Pi)$ with states $X = \{x_i : 1 \leq i \leq N\}$, and the transition probability matrix $\Pi = (\pi_{ij}) \in \mathbb{R}^{N \times N}$, its aggregated representative chain $(Y, \Psi, \Phi)$ with states $Y = \{y_j : 1 \leq j \leq M\}$, the transition probability matrix $\Psi = (\psi_{jk}) \in \mathbb{R}^{M \times M}$, where $M \ll N$, and the partition function $\Phi : X \to Y$ such that the $x_i \in X$ is represented by the $y_j := \Phi(x_i) \in Y$.

**Definition 1.** *The* co-variance matrix $C_X^\Phi(j)$ *corresponding to the superstate* $y_j \in Y$ *in the aggregated chain is given by*

$$C_X^\Phi(j) := \sum_{i=1}^{N} [Q]_{ij} \Big[\Theta^\top \frac{(\pi(i) - w(j))_\cdot}{w(j)}\Big] \Big[\Theta^\top \frac{(\pi(i) - w(j))_\cdot}{w(j)}\Big]^\top, \tag{6.12}$$

$$\textit{where } [Q]_{ij} = \begin{cases} 1, & \textit{if } x_i \in \Phi^{-1}(y_j) \\ 0, & \textit{otherwise} \end{cases}, \quad Q := [Q]_{ij}, \tag{6.13}$$

$W = [w(1), \ldots, w(M)]^\top = Q^\top \Pi$ *denotes the distribution vector as obtained*

*in (6.5),* $\Theta \in \mathbb{R}^{N\times N-1}$ *corresponds to the constraint* $z(j)^\top \mathbf{1}_N = 1\ \forall\ 1 \leq j \leq M$ *in (6.2),* $\frac{a.}{b}$ *denotes element-wise division of vectors* $a$ *and* $b$*, and* $[\cdot]^\top$ *denotes the transpose.*

**Definition 2.** *The* marginal return $\nu(k)$ *of the aggregated model with* $k$ *number of superstates among the given* $K$ *aggregated representations* $\{(Y_k, \Psi_k, \Phi_k) : |Y_k| = k\}_{k=1}^K$ *of a Markov chain* $(X, \Pi)$ *is given by*

$$\nu(k) := \log \bar{T}_{k-1} - \log \bar{T}_k, \text{ where} \tag{6.14}$$

$$\bar{T}_l := \max_{1\leq j\leq l} \left[\bar{T}_{l,j}\right] \text{ for } l \in \{k-1, k\}, \bar{T}_{l,j} = \lambda_{\max}(C_X^{\Phi_l}(j)) \tag{6.15}$$

*is heterogeneity of superstate* $y_j$*, and* $\lambda_{\max}$ *is largest eigenvalue.*

As illustrated in Section 6.1, the aggregated model with largest marginal return provides an appropriate choice for the number $k_t$ of superstate underlying a Markov chain, i.e.

$$k_t := \arg \max_{1\leq k\leq K} \nu(k), \tag{6.16}$$

The following algorithm computes the marginal return of the $K$ representations $\{(Y_k, \Psi_k, \Phi_k) : |Y_k| = k\}_{k=1}^K$ of a Markov chain $(X, \Pi)$, and estimates the number $k_t$ of superstates.

**Algorithm 7** Marginal return and Number of superstates

Input: $(X, \Pi)$, $\{(Y_k, \Psi_k, \Phi_k) : |Y_k| = k\}_{k=1}^K$;
Output: $\nu(k)$, $k_t$
**for** $k = 1$ to $K$ **do**
  From partition $\Phi_k : X \to Y$ determine $Q$ in (6.13)
  Compute $W := [\cdots w(j) \cdots]^\top = Q^\top \Pi$ and $\bar{T}_k$ using (6.15).
**end for**
$\nu(k)$ in (6.14) $\forall\ 2 \leq k \leq K$, and $k_t := \arg\max_k \nu(k)$.

## 6.4 Simulations

In this section we demonstrate the efficacy of marginal return in comparing different aggregated models, and estimating true number $k_t$ of superstates underlying a given Markov chain. We use Algorithm 7 that takes in the

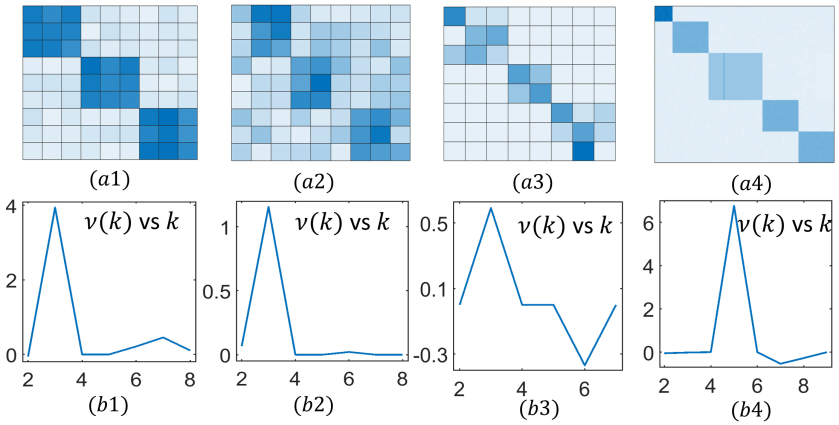


Figure 6.3: Illustrates the efficacy of Algorithm 7 in estimating $k_t$. (a1)-(a4) demonstrate the heatmaps for the transition matrices of NCD Markov chains generated such that $N = 9$, $k_t = 3$ in (a1)-(a2), $N = 8$, $k_t = 3$ in (a3), and $N = 100$, $k_t = 5$ in (a4). (b1)-(b4) are the corresponding persistence plots t hat clearly indicate $\nu(k_t) > \nu(k)\ \forall\ k \neq k_t$.

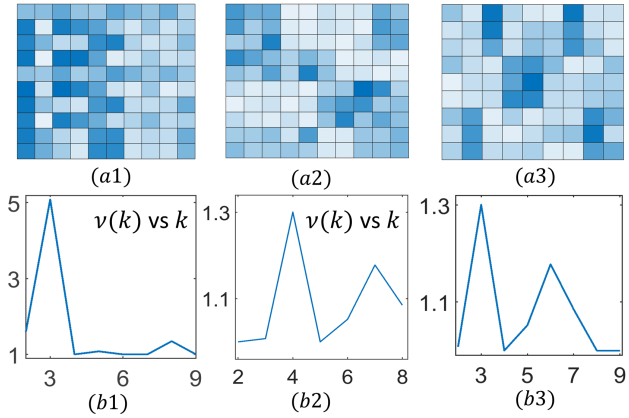


Figure 6.4: Illustrates the efficacy of Algorithm 7 in estimating $k_t$. (a1)-(a3) illustrate Markov chains generated by considering multiple copies of random vectors $\{\xi_i \in \mathbb{R}^N\}_{i=1}^{k_t}$; where $N = 10$, $k_t = 3$ in (a1), $N = 10$, $k_t = 4$ in (a2), and $N = 9$, $k_t = 3$ in (a3). (b1)-b(3) are the corresponding persistence plot that accurately estimate $k_t$.

aggregated models at different number $k$ of superstates as inputs, and outputs the corresponding marginal return and $k_t$. To demonstrate the generality of our method, we use aggregated models obtained from different aggregation algorithms. We use the algorithm presented in [130] on the first four example simulations in Figure 6.4, and [1] on the remaining examples. We would like to remark that both the algorithms perform well and result into meaningful aggregations at different number of superstates.

*NCD Markov Chains:* The transition matrix $\Pi$ for the NCD Markov chains [132] presumes the following structure $\Pi = \Pi^* + \epsilon C$, where $\Pi^*$ is a block diagonal matrix and $C$ adds a perturbation of the range $\epsilon$. Naturally, the number $k_t$ of superstates for such Markov chains can be approximated to

be the number of block diagonals in $\Pi^*$. Figures 6.4(a1)-a(2) illustrate the heatmaps of the transition probability matrix for two such Markov chains obtained at different levels of perturbation to a three block diagonal matrix $\Pi^*$. Figures 6.4(b1)-(b2) illustrate the corresponding persistence $\nu(k)$ versus $k$ plots which correctly identifies the true number $k_t = 3$ of superstates in both cases. Note that, even though the transition matrix $\Pi$ in Figure 6.4(a2) is highly perturbed, our method accurately identifies number of superstates thereby, demonstrating the robustness of the marginal return $\nu(k)$.

Figure 6.4(a3) illustrates a specialized NCD Markov chain with Courtois Transition Matrices [134]. These Markov chains are well studied in literature for their slow convergence to steady state. As is evident from the above heatmap in Figure 6.4(a3), the transition matrix comprises of 3 dominant block diagonal. Thus, the corresponding Markov chain comprises of $k_t = 3$ number of superstates. This is captured in our marginal return analysis in the Figure 6.4(b3). Figure 6.4(a4) demonstrates the state transition matrix of a large ($N = 100$ states) NCD Markov chain constituting 5 underlying block diagonals (one each of size $10 \times 10$ and $30 \times 30$, and three of size $20 \times 20$). Figure 6.4(b4) confirms largest marginal return at $k = 5$ number of superstates, and thus, appropriately estimates $k_t$.

*Randomly generated Markov Chains:* In the following simulations we consider $N$ state Markov chains where the transition matrices $\Pi = [\xi_{i_1}, \ldots, \xi_{i_N}]^\top + \epsilon C$ are generated from $k_t$ random distribution vectors $\{\xi_i\}_{i=1}^{k_t}$, and $\epsilon C$ introduces random perturbations. Naturally, $k_t$ estimates true number of superstates underlying the above Markov chains. Figure 6.4(a5) illustrates one such scenario where $\Pi \in \mathbb{R}^{10\times 10}$ is generated by perturbing multiple copies of $\{\xi_i\}_{i=1}^{3}$, and the marginal return analysis in Figure 6.4(b5) accurately determines $k_t = 3$ as the estimate for the true number of superstates. Similarly, Figures 6.4(a6) and 6.4(a7) illustrate the transition matrices $\Pi \in \mathbb{R}^{10\times 10}$ generated from 4 and 3 random distribution vectors, respectively, and the marginal return plots in Figure 6.4(b6) and 6.4(b7) accurately estimate the underlying true number of superstates. Note that even though the perturbations are significantly large in Figures 6.4(a5)-(a7), marginal return $\nu(k)$ accurately estimates the number $k_t$ of superstates underlying the original Markov chains.

*Emotion Transitions in Brain Network:* We consider the example Markov chain that models the transition between different emotional states of a per-

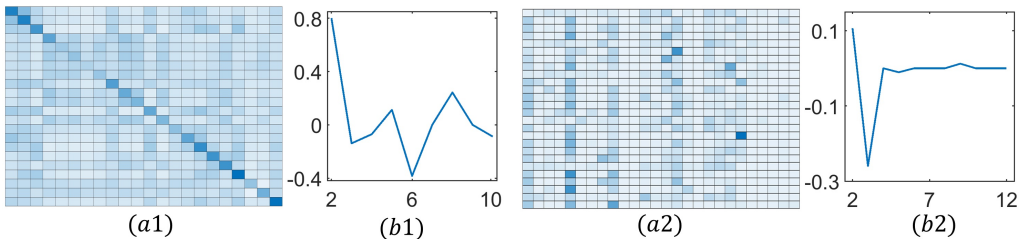


Figure 6.5: Marginal return analysis in realistic Markov chains. (a1) Transition matrix for brain emotion transitions where each state corresponds to an emotion. (b1) Maximum $\nu(k)$ observed at $k_t = 2$ suggesting two types of underlying emotions. (a2) Transition matrix for letter bigram dataset. Each state correspond to an English alphabet. (b2) Largest $\nu(k)$ at $k_t = 2$ indicating two types of alphabets in English language - vowels and consonants.

| $k$ | State Partitioning |
|---|---|
| 2 | $\{aeiouy\}, \{bcdfghjklmnpqrstvwxz\}$ |
| 3 | $\{aeiouy\}, \{cdghknrstvwxz\}, \{bfjlmpq\}$ |
| 4 | $\{aeiouy\}, \{hnrsvxz\}, \{cdgkwt\}, \{bfjlmpq\}$ |
| 5 | $\{aeiouy\}, \{bflmp\}, \{cdgktw\}, \{hnrlvxz\}, \{jq\}$ |

Table 6.1: State aggregations of letter bi-gram data obtained at different number of superstates. Obtained via aggregation algorithm in [1].

son [135]. The Figure 6.5(a1) illustrates the transition matrix $\Pi$ comprising $N = 22$ states - each corresponding to an emotion in the set $X = \{$*anxious, jittery, irritable, vigorous, alert, lively happy, attentive, intense, full-of-pep, excited, distressed, strong, stirred-up, nervous, upset, touchy, bold, temperamental, quiet, talkative, insecure*$\}$. We obtain different aggregated representations of $\Pi$ using the algorithm presented in [1], and observe that the marginal return $\nu(k)$ is the largest for $k_t = 2$ (see Figure 6.5(b1)). This indicates the presence of two superstates underlying the original Markov chain in Figure 6.5(a1). We note that the largest marginal return $\nu(k)$ at $k_t = 2$ is in accordance with the neuro-science literature [135] which also classifies emotions $X = X_p \sqcup X_n$ into positive $X_p$ and negative $X_n$ emotions, where $X_p = \{$vigorous, alert, lively, happy, attentive, intense, full-of-pep, excited, strong, bold, quiet$\}$, and $X_n = \{$anxious, jittery, irritable, vigorous, distressed, stirred-up, nervous, upset, touchy, temperamental, insecure, talkative$\}$. Thus, marginal return captures the number of distinct type of emotions in $X$. Similar analysis can be done on different brain network models [136] to qualify, for instance, the network states responsible for either *sensory-motor systems*, or for *higher-order cognition* (such as language).

*Letter Bigram dataset:* We finally investigate an example from Natural Language Processing. We consider the letter bi-gram dataset [137] that enu-

merates the number of transitions from one alphabet to another based on the Google Corpus Data (collection of $97,565$ distinct words, which were mentioned $743,842,922,321$ times). We represent the entire dataset as a Markov chain where each state represents an individual alphabet. The transition matrix $\Pi \in \mathbb{R}^{26\times 26}$ (see Figure 6.5(a2)) of the Markov chain captures the frequency of transition from one alphabet (state) to another. Our idea is to demonstrate that *marginal return* captures a "meaningful" number of superstates to be considered in the aggregated representation of the above Markov chain. Table 6.1 illustrates aggregated models obtained at different number of superstates. Since the alphabets are of two types, i.e., either vowels or consonants, one may find it reasonable to aggregate the original Markov chain into two superstates - one each for vowels and consonants. The above intuitive argument is reinforced by the largest value of marginal return $\nu(k)$ that is obtained at $k_t = 2$ (see Figure 6.5(b2)). Note that the corresponding aggregated model at $k_t = 2$ (as illustrated in Table 6.1) partitions the alphabets into either vowel sounds $\{aeiouy\}$, or the consonants (where $y$ is a (semi)-vowel). We also emphasize that the $k_t = 2$ superstates is not apparent from the transition matrix $\Pi$ in Figure 6.5(a2), however, the notion of marginal return successfully captures it.

*Remark* 9. The notion of marginal return is useful in evaluating the performance of a proposed aggregation algorithm. This can be done as follows. Consider a Markov chain where the number $k_t$ of superstates are known apriori. Use Algorithm 7 to obtain the marginal return $\nu(k)$ corresponding to the representative chains obtained from the proposed aggregation algorithm. The performance of the aggregation algorithm is assessed based on whether the largest marginal return coincides with $k_t$ or not.

## 6.5 Summary

Many studies involving large Markov chains require determining a smaller representative (aggregated) chain, where each *superstate* in the representative chain represents a *group of related* states in the original Markov chain. Typically, the choice of number of superstates in the aggregated chain is ambiguous, and based on the limited prior know-how. In this chapter we presented a methodology, rooted in phase transitions, to determine the best

candidate for the number of superstates. More precisely, we achieve this by comparing aggregated chains of different sizes. To facilitate this comparison we draw on the analytical conditions for the phase transition phenomenon and develop a new quantity called *heterogeneity* of a superstate, and subsequently use it to establish a notion of *marginal return* of an aggregated chain. In particular, our notion of marginal return captures the decrease in the *heterogeneity* upon a unit increase in the number of superstates in the aggregated chain. Our simulations on synthetic Markov chains, where the number of superstates are known apriori, we show that the aggregated chain with the largest marginal return identifies this number. In case of Markov chains that model real-life scenarios we show that the aggregated model with the largest marginal return identifies an inherent structure unique to the scenario being modelled; thus, substantiating the efficacy of our proposed methodology.

In general, model-reduction techniques usually rely on the pre-specified size of the reduced model. Thus, there arises a need to methodically estimate this size. The ideas and methods presented in this chapter are extendable to several related model-reduction problems. For instance, the graph clustering problem [1], that requires aggregating the nodes of a large graph into *supernodes*, or the co-clustering problem [138], that aggregates a given matrix $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2}$ to a smaller representative matrix $\mathcal{Y} \in \mathbb{R}^{M_1 \times M_2}$. The appropriate size of the representative graph in the former, or the aggregated matrix $Y$ in the latter can be estimated using similar ideas as elucidated in our work.

# CHAPTER 7

# SPARSE LINEAR REGRESSION

## 7.1 Introduction

In line with the theme of the Chapters 5 and 6, in this chapter we demonstrate the utility of phase-transitions in estimating the unknown sparsity in the sparse linear regression problems. In particular, we first develop, from scrtach, an MEP-based framework to address this problem, and demonstrate and derive the analytical conditions governing the phase transition phenomenon. Subsequently, we exploit these conditions to estimate the appropriate choice for the sparsity in the solution.

Sparse linear regression, or in other words, determining a sparse solution to an underdetermined system of linear equations is a prevalent engineering problem, that occurs in several applications such as image processing, machine learning, and compressed sensing [139–142]. Here, the central problem is to estimate a sparse vector $z \in \mathbb{R}^n$ (with $m \ll n$ non-zero entries) from measurements $y \in \mathbb{R}^N$ given by

$$y = Az + \epsilon, \tag{7.1}$$

where $A = [a_1\ a_2\ \dots\ a_n] \in \mathbb{R}^{N \times n}$, and $\epsilon \in \mathbb{R}^N$ denotes external noise. Here, the size of the vector $z$ is much greater than the size of the vector $y$ which is much greater than the sparsity $m$ of $z$, that is, $n \gg N \gg m$. Generally, the underlying optimization problem is posed as

$$\min_{z \in \mathbb{R}^n} \|y - Az\|_2^2 \quad \text{subject to } \|z\|_0 \leq m, \tag{7.2}$$

where $\|z\|_0$, the 0-norm, denotes the number of non-zeros components of $z$. The $\|\cdot\|_0$ norm is non-convex and the optimization problem in (7.2) is NP-hard [143]. Prior works in literature develop several greedy [144–147] and

relaxed [148–150] heuristics to address the optimization problem (7.2) for a pre-specified value of $m$. However, there is scant work that substantiates on the choice of $m$ itself. Recent works [151, 152] address the above problem of unknown sparsity when the optimization problem (7.2) is viewed in the context of compressed sensing (CS). More specifically, the above works propose methodologies to design the matrix $A$ (also referred to as the measurement matrix in CS), and estimate the appropriate sparsity level $m$ based on the resulting measurements $y \in \mathbb{R}^N$ from the above choices of the measurement matrices.

In this work we provide a methodology to estimate the sparsity for any general sparse linear regression problem. More precisely, our proposed approach is applicable to both the scenarios, (a) that allow designing the (measurement) matrix $A$ (such as in CS) and subsequently measure $y$ , and (b) where the vector $y$, and the matrix $A$ are already given and fixed. We achieve this by devising a method to *quantitatively* compare the sparse solutions obtained by solving (7.2) for different values $m$ of sparsity. This quantitative comparison allows to rank the different sparse solutions. Through simulations on synthetic data, where the *true* sparsity $m_t$ is known apriori, we show that the solution with the highest rank identifies this value of sparsity $m_t$. In particular, we develop and quantify a notion of *sparse marginal return* that facilitates the comparison of the solutions to (7.2) obtained at different sparsity values. We demonstrate that the solution with largest sparse marginal return estimates the sparsity $m_t$ underlying the given system of linear equations in (7.1).

**Motivation for the proposed methodology:** The underlying idea in developing and quantifying the notion of sparse marginal return is analogous to the ones presented in the context of data clustering (Chapters 5), and aggregation of Markov chains (Chapter 6). As done in the latter problems, we start by developing a Maximum Entropy Principle (MEP) based framework to address the optimization problem (7.2). Thereafter, we show that the resulting algorithm exhibits *phase transitions* at certain specific instances of the algorithm where the sparsity level $m$ (or, number of non-zero elements $m = \|z\|_0$) increases. In particular, the algorithm operates by determining a sparse solution $z$ with a single non-zero element at large values of the annealing temperature $T$. As the algorithm proceeds, i.e. annealing temperature $T$ decreases, the number of non-zero elements in $z$ increases at specific critical

temperatures $T := T_{\text{cr}}$ resulting into a phase transition. In this work, we explicitly determine the analytical conditions on the annealing parameter $T$ at which the phase transition occurs in our MEP-based algorithm for sparse linear regression. Subsequently, analogous to the case of data clustering and Markov chain aggregation, we quantify our notion of sparse marginal return based on the above obtained analytical conditions on $T$; also, we motivate our best estimate $m_t$ for the sparsity level underlying the linear system (7.1) from the phase transition phenomenon.

## 7.2 Problem Formulation

In our framework, we begin by re-writing the optimization problem (7.2) as

$$\begin{aligned} &\min_{x\in\mathbb{R}^m,\{\eta(V|x)\}} \sum_{V\in\mathcal{V}} \eta(V|x)\|y - AVx\|_2^2, \\ &\text{where } \eta(V|x) = \begin{cases} 1, & \text{if } V = \arg\min\limits_{W\in\mathcal{V}} \|y - AWx\|_2^2 \\ 0, & \text{otherwise} \end{cases}, \end{aligned} \tag{7.3}$$

$\mathcal{V} := \{V = \{v_{ij}\} \in \{0,1\}^{n\times m} : \sum_i v_{ij} = 1 \; \forall \; j\}$, and the constraint $\sum_i v_{ij} = 1$ on the columns of the matrix $V = \{v_{ij}\} \in \mathcal{V}$ ensures that the sparse solution $z = Vx$ contains at most $m$ non-zero elements. Note that in the above representation of the problem (7.2), we have split the sparse vector $z$ in (7.2) with the product $Vx$, where $V \in \mathcal{V}$ captures the location of the $n - m$ zero entries in $z$, and $x \in \mathbb{R}^m$ captures the remaining entries of $z$. Since $|\mathcal{V}| = n^m$ and the resulting decision variable space $\{\eta(V|x)\}$ is exponentially large, we trim down the space of decision variables to polynomial order by abstracting the binary matrix $V \in \mathcal{V}$ in terms of its individual components $v_{ij} \in \{0,1\}$. More precisely, we dissociate the decision variable $\eta(V|x)$ as

$$\eta(V|x) = \prod_{i,j=1}^{n,m} \eta_{ij}(v_{ij}|x), \tag{7.4}$$

$$\text{where } \eta_{ij}(v_{ij}|x) = \begin{cases} 1, & \text{if } v_{ij}\text{is } (i,j)-\text{th entry of } \arg\min_{W\in\mathcal{V}} \|y - AWx\|_2^2 \\ 0, & \text{otherwise} \end{cases},$$

The new decision variable space $\{\{\eta_{ij}(v_{ij}|x)\}, x\}$ is now of the polynomial order $\mathcal{O}(nm)$, and facilitates the computational scalability of our resulting algorithm.

## 7.3 Problem Solution

In our solution methodology, we relax the binary decision variables $\eta_{ij}(v_{ij}|x) \in \{0,1\}$ by soft decision weights $p_{ij}(v_{ij}|x) \in [0,1]$, and design these weights $p_{ij}(v_{ij}|x)$ using Maximum Entropy Principle (MEP). In particular, we determine the most *unbiased* weights $p_{ij}(v_{ij}|x)$, and the vector $x$ that maximize the corresponding Shannon entropy $H$ such that the expected objective attains a pre-specified value $c_0$; i.e., we pose the following optimization problem

$$\begin{aligned} \max_{\{p_{ij}(v_{ij}|x)\}, x\in\mathbb{R}^m} \quad & H := -\sum_{V\in\mathcal{V}} \prod_{i,j=1}^{N,m} p_{ij}(v_{ij}|x) \log p_{ij}(v_{ij}|x) \\ \text{subject to} \quad & D := \sum_{V\in\mathcal{V}} \prod_{i,j=1}^{N,m} p_{ij}(v_{ij}|x) \|y - AVx\|_2^2 = c_0, \\ & (v_{ij}) = V \in \mathcal{V}. \end{aligned} \tag{7.5}$$

We perform some algebraic simplifications to the objective function and the corresponding constraints, that assist in solving (7.5). The resulting optimization problem is given by

$$\begin{aligned} \max_{x\in\mathbb{R}^m, Q\in[0,1]^{N\times m}} \quad & H = 1_N^T [Q \circ \log Q + \bar{Q} \circ \log \bar{Q}] 1_m \\ \text{subject to } D = & \|y - AQx\|_2^2 + [a_1^T a_1 \ \ldots \ a_N^T a_N][Q \circ \bar{Q}][x \circ x] = c_0 \\ & 1_N^T Q = 1_m, \end{aligned} \tag{7.6}$$

where $Q := \{q_{ij}\}$, $q_{ij} := p_{ij}(v_{ij} = 1|x)$, $\bar{Q} = 1_{N\times m} - Q$. Please refer to the Appendix A.6.1 for details. Minimizing (local) the corresponding Lagrangian

$$\begin{aligned} F = & \|y - AQx\|^2 + [a_1^T a_1 \ \ldots \ a_N^T a_N][Q \circ \bar{Q}]x^2 - c_0 \\ & + \mu^T (Q^T 1_N - 1_m) + T 1_N^T [Q \circ \log Q + \bar{Q} \circ \log \bar{Q}] 1_m \end{aligned} \tag{7.7}$$

(where $T$ and $\mu$ are the Lagrange multipliers), by setting $\frac{\partial F}{\partial x} = 0$ and $\frac{\partial F}{\partial Q} = 0$, we obtain

$$x = \Big[Q^T A^T A Q + \underbrace{\operatorname{diag}\big[\lambda^T (Q \circ \bar{Q})\big]}\Big]^{-1} Q^T A^T y, \tag{7.8}$$

$$Q = \frac{\exp(\frac{2}{T} H_m) \circ}{\exp(\frac{2}{T} H_m) + \exp(-\frac{2}{T} H_p)}, \quad \text{where} \tag{7.9}$$

$$H_m = \min \Big\{ A^T (Y - AQx) x^T - \frac{1}{2} \lambda (x \circ x)^T \circ (1 - 2Q) - \frac{1}{2} 1_N \mu^T, \mathbf{0} \Big\},$$

$$H_p = \max \Big\{ A^T (Y - AQx) x^T - \frac{1}{2} \lambda (x \circ x)^T \circ (1 - 2Q) - \frac{1}{2} 1_N \mu^T, \mathbf{0} \Big\},$$

and $\lambda = [a_1^T a_1 \dots a_N^T a_N]$. The Lagrange multiplier $\mu$ is updated using the equality constraint in (7.6). In particular, the constraint results into the following update law for $\mu$

$$\mu = T \log \Big[ \mathbf{1}^T \Big( \frac{1 \circ}{\exp\big(-\bar{\mu}/T\big) + \exp\big(-\bar{D}/T\big)} \Big) \Big], \tag{7.10}$$

where $\bar{\mu} = [\mu \ \mu \ \dots \mu]^T \in \mathbb{R}^{N \times m}$ and $\bar{D} = (A^T (Y - AQx) x^T - 0.5 \lambda (x \circ x)^T \circ (1 - 2Q)$. It is known from sensitivity analysis [10] that the large (small) values of $c_0$ in (7.6) are analogous to large (small) values of the Lagrange parameter $T$. In fact, we solve the optimization problem (7.6) repeatedly at decreasing values of $T$ (or equivalently, at decreasing values of $c_0$). At large values of $T \to \infty$, the Lagrangian $F$ is dominated by the convex function $-H$, the weights $Q$ in (7.9) are uniformly distributed ($q_{ij} = \frac{1}{N}$), and all the entries of $x$ in (7.8) are identical; i.e., the algorithm effectively finds just *one* distinct entry in $x$, or equivalently, the solution is of sparsity $m = 1$ at large values of temperature $T$. As $T$ decreases further the Lagrangian $F$ is more and more dominated by the cost function $D$ in (7.6), the weights $q_{ij}$ in $Q$ are no longer uniform, and the elements of $x$ become distinct from each other, i.e., the sparsity $m > 1$. At $T \to 0$, the weights in $Q$ converge to either 0 or 1, the Lagrangian $F$ converges to $D$, and the algorithm solves the original optimization problem in (7.3). The Algorithm 8 illustrates the steps of the algorithm resulting from our MEP-based framework to determine the sparse solution to (7.3). The following theorem shows that the iterates in the Algorithm 8 mimic a descent step; and thus, guarantee converges to a local minima.

**Algorithm 8** Sparse Linear Regressions Using MEP

Input: $\beta_{\min}$, $\beta_{\max}$, $m_{\max}$, $\tau > 1$;
Output: $Q$ and $x$.
Initialize: $\beta = \beta_{\min}$, $Q = [q_1, \ldots, q_{m_{\max}}]$, $q_j = \frac{1}{N}$
**while** $\beta \leq \beta_{\max}$ **do**
    **while** True **do**
        Iteratively update $Q$ and $\mu$ in (7.9), (7.10).
        break if $x$,$Q$, and $\mu$ converge.
    **end while**
    $\beta \leftarrow \tau\beta$
**end while**

**Theorem 14.** *(a) The implicit equation $Q$ in (7.9) corresponds to gradient descent step in the auxiliary variable $\zeta := -\log \frac{Q_\circ}{1-Q}$, or equivalently, $\zeta = \frac{e^{-Q}\circ}{1+e^{-Q}}$, where the descent step is given by*

$$\zeta^+ = \zeta - \frac{1}{T}\big(\exp(\zeta/2) + \exp(-\zeta/2)\big)^2 \frac{\partial F}{\partial \zeta}, \tag{7.11}$$

*(b) The implicit equation $\mu$ in (7.10) is analogous to the descent step*

$$\mu^+ = \mu - \frac{T}{\bar{k}} \frac{\partial F}{\partial \mu}, \tag{7.12}$$

*where $\bar{k} > 0$ lies between $Q^T 1_N$ and $1_m$.*
*(c) Thus, at each given $\beta$ iteration, the updates $Q, \mu$ given by (7.9), (7.10) respectively, are analogous to the gradient descent step of the form*

$$\begin{bmatrix} [\zeta^+] \\ \mu^+ \end{bmatrix} = \begin{bmatrix} [\zeta] \\ \mu \end{bmatrix} - \begin{bmatrix} \frac{1}{T}\big(\exp(\zeta/2) + \exp(-\zeta/2)\big)^2 & 0 \\ 0 & \frac{T}{\bar{k}} \end{bmatrix} \begin{bmatrix} [\frac{\partial F}{\partial \zeta}] \\ \frac{\partial F}{\partial \mu} \end{bmatrix}. \tag{7.13}$$

*Proof.* Please refer to the Appendix A.6.2 □

In the following section, we further illustrate the gradual increase in the number of distinct elements in $x$, and derive analytical conditions for the same. Subsequently, we motivate and develop our notion of sparse marginal return based on these conditions.

## 7.4 Phase Transition

In this article we exploit this heirarchical increase in the number of *distinct* elements in $x$ to define our notion of sparse marginal return. Insights from the simulations of our algorithm demonstrate that there exist certain critical temperatures $T = T_{cr}$ values at which the solution undergoes the phenomenon of *phase transition*. This phenomenon is characterized by the increase in the number of distinct entries in $x$. These critical termperatures $T_{cr}$'s occur when the critical point $Q$ in (7.9) of the Lagrangian $F$ is no longer the local minima, this is, when for at least one perturbation direction $\Phi = [\phi_1, \ldots, \phi_m]^T \in \mathbb{R}^{N\times m}$, the Hessian $H(Q, \Phi, T) :=$

$$\frac{d^2F(Q+\epsilon\Phi)}{d\epsilon^2}\Big|_{\epsilon=0} = \sum_{l,j=1}^{m} \Phi_j^T C^T \Big[\frac{T}{2}\hat{\Lambda}_l - \Gamma_l\Big] C\Phi_j + \Big\| \sum_{l=1}^{m} E_l C \sum_{j=1}^{m} \Phi_j \Big\|^2 \quad (7.14)$$

is no longer positive. Note that, upon substituting $x$ from (7.8) in the expression (7.7), the resulting Lagrangian $F$ is only a function of $Q$ and thus, the Hessian of $F$ is computed considering it only as function of $Q$ in (7.9). The matrices in (7.14) are given as below

$$\begin{aligned}
-\Gamma_l &\triangleq -\Theta_l^{-1} A^T w w^T A + \Theta_l^{-1} h_l (A^T w L_l^T \Lambda_a + \Lambda_a L_l w^T A) \\
&\quad - \Theta_l^{-1} h_l^2 \Lambda_a L_l L_l^T \Lambda_a - h_l^2 \Lambda_a, \\
E_l &\triangleq \Delta^{-1/2} A\big[h_l I + Q\Theta^{-1} e_l w^T A + h_l Q \Theta^{-1} e_l L_l^T \Lambda_a\big], \\
\Theta &= \mathrm{diag}(\lambda_a^T (Q \circ \bar{Q})), \\
\Theta_l &= e_l^T \mathrm{diag}(\lambda_a^T (Q \circ \bar{Q})) e_l, \\
w &= \Delta^{-\frac{1}{2}} \mu, \quad \Delta = I + AQ\Theta^{-1} Q^T A^T, \\
h_l &= w^T AQ\Theta^{-1} e_l, \quad L_l = 1 - 2q_l \ q_l \text{ is the } l-\text{th column of } Q, \\
\Lambda_a &= \mathrm{diag}[a_1^T a_1 \ldots a_N^T a_N], \hat{\Lambda}_l = \mathrm{diag}\Big[\frac{1}{q_l} - \frac{1}{1-q_l}\Big].
\end{aligned}$$

Please refer to the Appendix A.6.3 for computation involved in (7.14). We derive the expression for the critical temperature $T_{cr}$ as below.

**Theorem 15.** *The value of critical temperature $T_{cr}$ at which the Hessian $H(Q, \Phi, T)$ in (7.14) is no longer positive for some perturbation direction $\Phi$,*

*and $Q$ in (7.9) undergoes phase transition is given by*

$$T_{cr} := 2 \max_{1 \leq l \leq m} [T_{cr,l}], \text{ where } T_{cr,l} = \lambda_{\max}\big(S_T(l)\big). \tag{7.15}$$

*Here, $S_T(l) \in \mathbb{R}^{(N-1)\times(N-1)}$ is referred to as the PT (phase transition) matrix corresponding to the $l$-th column of $Q$ (or, in other words the $l$-th entry in $x$). The PT matrix $S_T(l)$ depends on the matrices $C^T$, and $Q$. Please refer to Appendix A.6.4 for details.*

*Proof.* Please refer to the Appendix A.6.4 □

## 7.5 Sparse Marginal Returns

As illustrated in the Section 7.1, we motivate our methodology to estimate the appropriate sparsity using the prior work done in Data clustering, and the aggregation of Markov chains in Chapters 5 and 6, respectively. In particular, as done in the latter problems, we replace the soft weights $Q \in [0,1]^{N\times m}$ in the PT matrix $S_T(l)$ with the hard (binary) values as prescribed by a given sparse solution $V$ (where $\eta(V|x) = 1$) to (7.3). We denote the modified PT matrix by $\mathbf{S}_T(l)$. Note that this sparse solution $V$ in $\mathbf{S}_T(l)$ can be obtained using any algorithm, i.e., our proposed definition of sparse marginal returns is independent of the algorithm being used to determine the sparse solutions. Subsequently, we define the *sparse marginal return* $\nu(m)$, and provide a structured methodology to determine the appropriate choice of sparsity $m_t$ underlying the linear system in (7.1).

**Definition 3.** *The sparse marginal return $\nu(m)$ of an $m$-sparse solution $z \in \mathbb{R}^n$ among the given $M$ solutions $\{z : 1 \leq \|z\|_0 \leq M\}$ to the linear sparse regression problem in (7.2) is given by*

$$\nu(m) := \log \bar{T}_{m-1} - \log \bar{T}_m, \quad \text{where} \tag{7.16}$$

$$\bar{T}_r = \max_{1 \leq j \leq r} \left[\lambda_{\max}\big(\mathbf{S}_T(j)\big)\right] \text{ for } r \in \{m-1, m\}, \tag{7.17}$$

*where $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue.*

As in the context of data clustering in Chapter 5 - where the clustering solution with largest persistence indicates the estimate for the true number of

clusters - or, as in the case of Markov chain aggregation in Chapter 6 - where the aggregated model with largest marginal return estimates the appropriate choice for the number of superstates - here, we denote the solution with the largest sparse marginal return as a best estimate for the sparsity $m_t$ of the sparse solution to (7.2), i.e.

$$m_t := \arg \max_{1 \leq m \leq M} \nu(m). \tag{7.18}$$

The following algorithm computes the sparse marginal return $\nu(m)$ of the $M$ solutions $\{z : 1 \leq \|z\|_0 \leq M\}$ to the sparse linear regression in (7.2), and estimates the sparsity number $m_t$.

**Algorithm 9** Sparse Marginal return and Estimating $\|z\|_0$

Input: $Y, A, \{z_m : \|z\|_m = m\}_{m=1}^{M}$;
Output: $\nu(m)$, $m_t$
**for** $m = 1$ to $M$ **do**
  represent $z_m$ as $z_m := V_m x_m$ in (7.3) to determine $V_m$.
  Compute $\bar{T}_m$ using (7.17).
**end for**
$\nu(m)$ in (7.16) $\forall\ 2 \leq m \leq M$, and $m_t := \arg\max_m \nu(m)$.

## 7.6 Simulations

In this section we demonstrate the efficacy of our notion of sparse marginal return $\nu(\cdot)$ in (7.16) in comparing the different sparse solutions, and estimating the *true* sparsity $m_t$ underlying the linear equation (7.1). We use the Algorithm 9 that takes in the sparse solutions $\{z_m : \|z\|_m = m\}_{m=1}^{M}$ as inputs, and outputs the corresponding sparse marginal return and $m_t$. To demonstrate the independence of our method to the algorithm being used to obtain the sparse solutions $\{z_m : \|z\|_m = m\}_{m=1}^{M}$, we use the popular OMP (Orthogonal Matching Pursuit) [144] to obtain them.

*Remark* 10. In our simulations, we observe that for $m_t$ (7.18) to estimate the true sparsity underlying the linear system (7.1), the algorithm being used to obtain the sparse solution should result into meaningful (reasonably well, and not a poor local minima) solutions.

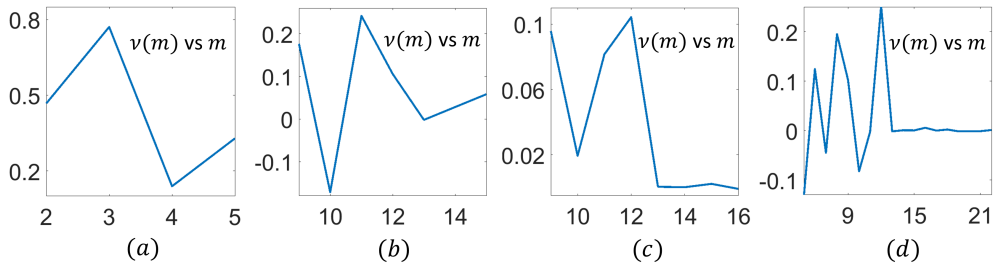


Figure 7.1: Illustrates the efficacy of the Algorithm 9 in estimating $m_t$. (a) $Y \in \mathbb{R}^6$, $A \in \mathbb{R}^{6\times 10}$, $\|z\|_0 = 3$, (b)$Y \in \mathbb{R}^{100}$, $A \in \mathbb{R}^{100\times 1000}$, $\|z\|_0 = 12$, (c)$Y \in \mathbb{R}^{100}$, $A \in \mathbb{R}^{100\times 1000}$, $\|z\|_0 = 12$, (d)$Y \in \mathbb{R}^{100}$, $A \in \mathbb{R}^{100\times 1000}$, $\|z\|_0 = 12$

*Synthetic Data:* In our simulations we randomly generate four pairs of the matrix $A \in \mathbb{R}^{N\times n}$, and the sparse vector $z \in \mathbb{R}^n$ at given sparsity levels $\|z\|_0$. In each case, the subsequent linear measurement $Y$ is given by $Az + \epsilon$, where $\epsilon$ denotes a random Gaussian noise in the setup. Subsequently, for each example, we determine the sparse solutions at different sparsity levels $m$ using the OMP algorithm [144]. Figure 7.1 illustrates the marginal sparse return $\nu(m)$ versus $m$ obtained for each of these four examples. The true underlying true sparsity in the case of Figure 7.1(a),(c), and (d) are $3, 12$, and 12, respectively, and as the same is captured by the corresponding $\nu(m)$ versus $m$; i.e., the Algorithm 9 exactly identifies the underlying sparsity for the examples in the Figure 7.1(a),(c), and (d). In the example presented in Figure 7.1(b), the maximum value of sparse marginal return occurs at $m_t = 11$ which is close to (however, not exactly equal to) the actual underlying sparsity in this case is $\|z\|_0 = 12$. This mismatch could be attributed to reasons such as the presence of random noise $\epsilon$ in the measurement vector $Y$, or the performance of the OMP algorithm on this dataset.

*Standard Data:* We now consider some of the standard datasets available in the literature to demonstrate the performance of sparse marginal return. As there is no explicit knowledge of the true sparsity underlying these standard datasets, we use the magnitude of the coefficients obtained from solving the regression analysis to benchmark the true number of non-zero entries in $z$. In particular, we normalize the matrix $A \in \mathbb{R}^{N\times n}$, and the measurement $Y \in \mathbb{R}^N$, and perform linear regression to obtain $v \in \mathbb{R}^n$ such that $\|Y - Av\|_2^2$ is minimized. Thereafter, we arrange the entires in $v$ in descending order, and consider all the coefficients that are significantly smaller than the largest as zero. The remaining number of non-zero entries are considered as

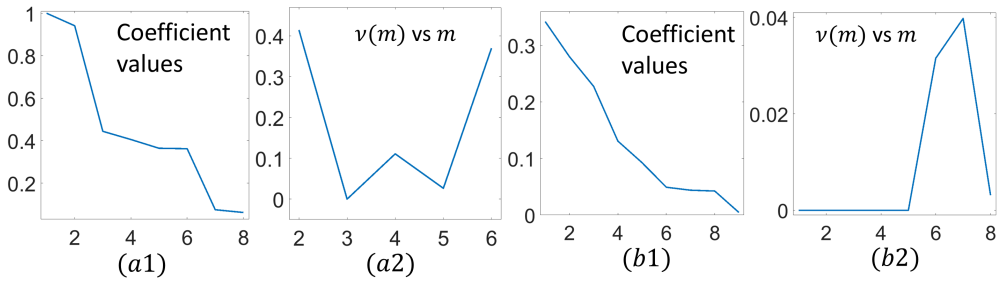


Figure 7.2: Illustrates the efficacy of the Algorithm 9 in estimating $m_t$ on standard datasets. (a1)-(a2) Diabetes data set $Y \in \mathbb{R}^{768}$, $A \in \mathbb{R}^{768\times 8}$, $\|v\|_0 = 2$, (a2) Sparse marginal return maximu at $m_t = 2$ (b1)-(b2) Cancer dataset (b1) $Y \in \mathbb{R}^{97}$, $A \in \mathbb{R}^{97\times 9}$, $\|v\|_0 = 8$, (b2) Sparse marginal return maximum at $m_t = 8$

a benchmark for the true sparsity in $z$. Thereafter, we compare this with the estimate of the sparsity $m_t$ resulting from our Algorithm 9.

We consider the 2 standard datasets, namely the Pima Indian Diabetes dataset [153] and the prostate cancer study dataset in [154]. The diabetes dataset comprises of $N = 768$ examples with $n = 8$ features characterizing the occurrence of diabetes in the patients (this gives the matrix $A \in \mathbb{R}^{N\times n}$. A separate column in the dataset identifies the patient with or without diabetes which forms the vector $Y \in \mathbb{R}^N$. The plot in Figure 7.2(a1) illustrates the magnitudes of the coefficients of the vector $v \in \mathbb{R}^n$ obtained upon performing linear regression. As is evident, only two coefficients in $v$ (consequently, two features in the dataset) are much larger than the other coefficients, and thus, the benchmark sparsity is $\|v\|_0 = 2$. The same is captured by our proposed sparse marginal metric as shown in the Figure 7.2(a2), where the sparse marginal return $\nu(m)$ which is largest for $m = 2$. Similar results are observed in the case of the cancer dataset as shown in Figure 7.2(b1)-(b2), where the regression analysis in Figure 7.2(b1) indicates 8 features (out of 9) in the data set responsible for identifying cancer. Essentially there is no significant cut-off evident from the Figure 7.2(b1) and one of the coefficient from linear regression is close to zero already, so we deduce that $\|v\|_0 = 8$. The same is capture by the sparse marginal return in the Figure 7.2(b2) which attains its maximum values at $m = 8$.

## 7.7 Analysis and Discussion

*A.* Though, this Chapter focuses on bringing out the utility of phase tran-

sitions in estimating the underlying sparsity $m_t$ in the the problem (7.1), the another significant contribution is in developing, from scratch, an MEP-based framework for the sparse linear regression problem. The iterates in the resulting Algorithm 8 are shown to be analogous to descent steps at each value of the annealing parameter $\beta$. Further investigation and benchmarking of the quality of the solutions resulting from the Algorithm 8 is required.

*B.* The notion of sparse marginal return can be useful in evaluating the performance of a proposed algorithm to solve (7.2). This can be done as follows. Consider a Markov chain where the underlying sparsity $\|z\|_0$ is known apriori. Use Algorithm 9 to obtain the sparse marginal return $\nu(m)$ corresponding to the sparse solutions obtained from the proposed algorithm. The performance of the algorithm can be assessed based on whether the largest sparse marginal return corresponds to the known $\|z\|_0$ or not.

*C.* Another future direction to evaluate the performance of sparse marginal return in estimating the true sparsity in standard datasets, could be to run the Algorithm 9 on datasets where the domain experts' knowledge on the number of non-zero entries is available. Such example datasets with expert knowledge can be found in biological studies involving cancer gene data.

## 7.8 Summary

In this chapter we developed a MEP-based framework to address the sparse linear regression problem. We also demonstrate and derive analytical conditions governing the phase transitions in the algorithm resulting from this framework. Thereafter, we develop and quantify a notion of sparse marginal return (in line with the themes in Chapters 5 and 6) that facilitates comparing the sparse solutions obtained at different sparsity level. Subsequently, we demonstrate the efficacy of sparse marginal return in estimate the underlying sparsity in the solution to a linear system of equations.

# CHAPTER 8

# CONCLUSION AND FUTURE DIRECTIONS

In this thesis we have introduced and addressed the class of optimization problems that require making sequential decisions (in finite or infinite horizons) along with determining unknown model parameters such that the associated cost function is minimized. We refer to them as the parameterized Sequential Decision Making (para-SDM) problems. Some of the applications of para-SDM include network design, vehicle routing, industrial process optimization, last mile delivery and facility location with path optimization. We develop a combinatorial viewpoint of the para-SDM problems - facilitated by the combinatorially large number of sequences of decisions possible in these problems. This motivates us to develop Maximum Entropy Principle (MEP) based frameworks to address para-SDM, where the success of the MEP-based frameworks in addressing specific combinatorial optimization problems such as vehicle routing and facility location problems have been demonstrated in the literature. We also bring out the phenomenon of phase transitions exhibited by these MEP-based frameworks, and demonstrate their utility in estimating the various hyperparameters in the combinatorial optimization problems such as identifying the true number of clusters underlying a given dataset. We show the flexibility of the proposed frameworks in incorporating application specific inclusion/exclusion, capacity, and dynamic constraints in the problem.

There are several directions in which the work done in this thesis can be carried forward. One aspect of the future work is to find innovative applications of the proposed frameworks and ideas developed here. For instance, the data clustering (or, analogously the facility allocation) problem is an para-SDM problem where the decisions involve determining the closest facility and the locations of the facilities form the unknown parameters of the problems. A huge benefit of viewing data-clustering as para-SDM is that we can deploy the model-free learning algorithm developed in Section 3.4 to solve the data-

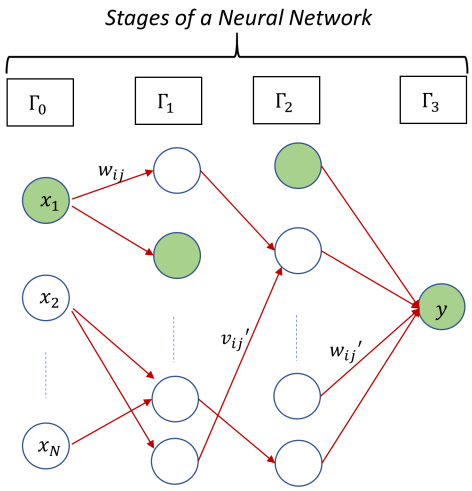


Figure 8.1: Artificial Neural Network as a para-SDM task. The weights of the ANN are the unknown parameters and the connections from one neuron to other are the sequential decisions.

clustering problem. Essentially, this means that we do not need to know the actual cost functions and the system uncertainties to perform clustering, albeit, it can be done through the structured interactions with the underlying environment (system) as demonstrated in Chapter 3.

There are several, seemingly unrelated, applications that can be viewed as para-SDM problems. For instance, the neural network design can be viewed as a para-SDM problem in finite horizon as illustrated in the Figure 8.1. The objective is to design the weights of the neural network such that the associated loss is minimized. We note that the partially connected neural network holds close analogy with the stagewise illustration of the FLPO problem in Figure 2.2. This motivates developing an MEP-based framework for the neural networks. Additionally, the phase transitions, if observed in the resulting framework for ANNs, can be helpful in estimating the number of neurons in the neural network layers.

Another body of future work is in viewing the model predictive control (MPC) as a para-SDM problem. MPC is popular process control technique in industries that requires solving a finite horizon SDM problem repeatedly at each time instant to determine a sequence of control action to be taken for that process. Recently, the use of MPC has also shifted to applications involving fast and non-linear dynamics such as autonomous cars and robotics [155]. The most fundamental challenges facing the MPC here in these systems originate from the need of solving the associated optimization problem repeatedly

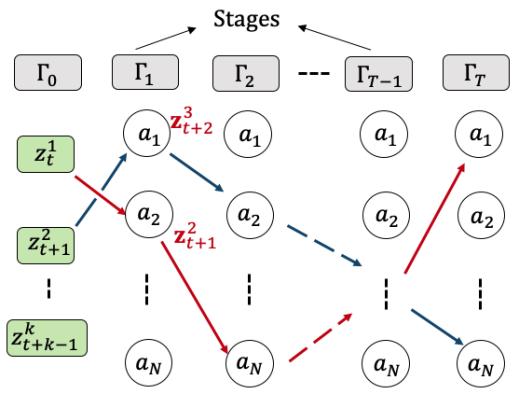


Figure 8.2: Illustrates the schematic of MPC unrolled in $T$ time horizon from the current system state $z_t^1$ and the other estimated $k-1$ states in $\Gamma_0$. The control action space is reduced to $N$ distinct controls in $\{a_1, \ldots a_N\}$ which form the parameters of the para-SDM. The sequence of control action from each state of the system in $\Gamma_0$ forms the sequential decision aspect of the para-SDM.

at each time instant $t$. One way to overcome this challenge is by posing an *alternative* to the above classical MPC scheme wherein it requires to solve the associated optimization problem only once every $k$-time steps along with fewer decision variables in comparison to the above classical MPC (see Figure 8.2). The idea is to consider, along with the current system state $z_t^1$, $k-1$ estimated states $\{z_{t+1}^2, \ldots, z_{t+k-1}^k\}$ as shown in the Figure 8.2, and solve for the optimal sequence of control actions from each of the state where the action space is limited to only $N$ distinct actions (see Figure 8.2). Here, the $N$ distinct actions $\{a_1, \ldots, a_N\}$ will form the parameters of the underlying para-SDM problem and the sequence of control from each state in $\Gamma_0$ will form the decision making part of the para-SDM. The idea is to achieve faster computations by (a)simultaneously optimizing for $k$ consecutive states instead of just solving the underlying optimization problem from the current system state, and (b) trimming the decision variable space by aggregating the control actions, i.e. allowing the states $\{z_t^1, z_{t+1}^2, \ldots, z_{t+k-1}^k\}$ to choose their sequence from $N$ (to be determined) control actions $\{a_1, \ldots, a_N\}$. Further, when addressed as in the para-SDM framework, the non-convexity in the MPC problems arising out of the non-linear dynamics can be taken care of by the annealing aspect of the framework.

The homotopy from a convex function (negative Shannon entropy) to the non-convex cost function is an essential element of MEP-based frameworks that prevents the algorithm to get stuck in a poor local minima. However, it

is observed in the simulations that significant changes occur to the solutions only at the moments of *phase transitions* in the algorithm [60]. For instance, in the Deterministic Annealing (DA) algorithm - MEP-based method for data clustering - these phase transitions are characterized by split of an existing cluster (increase in the cluster count) and re-assigning the data points to their closest cluster. It is observed that this splitting mechanism operates in certain way where the splits occur always in the direction of the largest spread in the clusters. It is possible to abstract a higher-level algorithm for clustering from this observation that bypasses the annealing aspect of the DA algorithm. In particular, the clustering can be performed by allocating a single centroid at the weighted mean, then splitting it in the direction of the largest spread, and continuing this process till the required number of clusters are observed. This is analogous to a careful hierarchical seeding of the cluster centers. Prior works such as in [156, 157], have demonstrated sub-optimality of the clustering algorithms rooted in specific initial seeding of the cluster centers. The above abstraction of the DA algorithm and the work done in [156, 157] forms a basis to perform suboptimality studies on the DA algorithm. This would lay the foundation to quantify the quality of solutions obtained in much larger class of problems, such as para-SDM, that can be addressed using the MEP-based frameworks.

The flexibility to incorporate application-specific constraints at the level of decision making is one of the strengths of MEP-based frameworks. These involve constraints such as capacity constraint on the cluster size, routing and node capacity constraints in a network. There are several aspects that play a crucial role in promoting this capability; for instance, relaxing the binary decision variables (which eventually converges to either 0 or 1 at low annealing temperatures) in the framework and the resulting Gibbs' distribution of these decision variables. We observe that the Lagrange parameters of these constraints reflect in the resulting Gibbs' distributions, and the simple update rules for the Lagrange parameters (along with the solution) becomes analogous to a descent method; thus guaranteeing convergence. Such capabilities of the MEP-based frameworks can be conveniently utilized in power-grid prosumer network design problems, where the capabilities of individual nodes in terms of producing and consuming power is constrained and different from one another, and also the each node could be of different priority (for instance, critical nodes such as a hospital gets a high priority).

# APPENDIX A

## A.1 Facility Location and Path Optimization

### A.1.1 Definitions of Matrices in (2.13)

1. $A = \sum_{i=1}^{M} A_i$, where $A_i \in \mathbb{R}^{M\times M}$ is a diagonal matrix such that $(A_i)_{jj} = \sum_{\gamma_0,\gamma_1,\ldots,\gamma_{i-1}} \rho_{\gamma_0} p_0(\gamma_1|\gamma_0)\ldots p_{i-1}(f_j|\gamma_{i-1})$.

2. $B = \sum_{i=1}^{M-1}(B_i + B_i^T)$ where $B_i \in \mathbb{R}^{M\times M}$ is such that $(B_i)_{mn} = \sum_{\gamma_0,\gamma_1,\ldots,\gamma_{i-1}} \rho_{\gamma_0} p_0(\gamma_1|\gamma_0)\ldots p_{i-1}(f_m|\gamma_{i-1}) p_i(f_n|f_m)$.

3. $\bar{X} \in \mathbb{R}^{M\times n}$, where $\bar{X}_{mn} = \sum_{\gamma_0} \rho_{\gamma_0} p^0(f_m|\gamma_0)(\xi(\gamma_0))_n$, and $(\xi(\gamma_0))_n$ is the n-th component of the spatial coordinate of $\gamma_0$.

4. $C = \bar{B} + \sum_{i=2}^{M-1} \tilde{B}_i + \sum_{j=2}^{M-1} \check{B}_j + \bar{D} \in \mathbb{R}^{M\times n}$, where
   $(\bar{B})_{mn} = \sum_{\gamma_0} \rho_{\gamma_0} p_0(f_m|\gamma_0) p_1(\delta|f_m) z_n$,
   $(\tilde{B}_i)_{mn} = \sum_{\gamma_0,\gamma_1,\ldots,\gamma_{i-1}} \rho_{\gamma_0} p_0 \ldots p_{i-2} p_{i-1}(f_m|\gamma_{i-1}) p_i(\delta|f_m) z_n$,
   $(\check{B}_i)_{mn} = \sum_{\gamma_0,\gamma_1,\ldots,\gamma_{i-1}} \rho_{\gamma_0} p_0 \ldots p_{i-2} p_{i-1}(\delta|\gamma_{i-1}) p_i(f_m|\delta) z_n$,
   $(\bar{D})_{mn} = \sum_{\gamma_0,\ldots,\gamma_{M-1}} \rho_{\gamma_0} p_0 \ldots p_{M-1}(f_m|\gamma_{M-1}) z_n$, and $z_n$ denotes the $n-$th coordinate of the destination $\delta$.

The matrix $C := 2\hat{A} - \hat{B}$ is such that for every $i-$th row in $C$ the sum of absolute value of the off-diagonal entries ($\sum_{j\neq i} |[C]_{ij}|$) is less than the absolute value of the diagonal element ($|[C]_{ii}|$) in that row, i.e. $\sum_{j\neq i} |[C]_{ij}| < |[C]_{ii}|$. Thus, by Gerhgorin's Circle Theorem [158] all the eigenvalues of $C$ are positive and hence $C$ is a positive definite matrix.

### A.1.2 Proof of Theorem 1

As indicated in the algorithm, at each value of the annealing parameter $\beta_t$, where $t$ denotes the $t^{\text{th}}$ iteration, it is required to solve the following implicit equation iteratively $y = (2\hat{A}(y) - \hat{B}(y))^{-1}(\hat{\bar{X}}(y) + \hat{C}(y))$. The corresponding iteration scheme (where $n$ denotes the iterate number) solves for $y_t$ at each value of $\beta_t$

$$y_t(n+1) = \underbrace{(2\hat{A}_t(n) - \hat{B}_t(n))^{-1}(\hat{\bar{X}}_t(n) + \hat{C}_t(n))}_{=:G_t(y_t(n))}$$

where $\hat{A}_t(n), \hat{B}_t(n), \hat{\bar{X}}_t(n)$ and $\hat{C}_t(n)$ are dependent on $y_t(n)$. The *free-energy* at the $n^{\text{th}}$ iteration for the annealing parameter $\beta_t$ is given by $F_t(n) = -\frac{1}{\beta_t}\sum_{\gamma_0 \in \mathcal{S}} \rho_{\gamma_0} \log \sum_{\gamma \in \mathcal{G}} e^{-\beta_t \sum_{t=0}^{M} d_t(\gamma_t(n), \gamma_{t+1}(n))}$ whose gradient with respect to $y$ is $\nabla F_t(n) = 2(2\hat{A}_t(n) - \hat{B}_t(n))(y_t(n) - y_t(n+1))$

$$\Rightarrow y_t(n+1) = y_t(n) - \frac{1}{2}(2\hat{A}_t(n) - \hat{B}_t(n))^{-1}\nabla F_t(n),$$

which is of the form $y_t(n+1) = y_t(n) + \alpha_k \chi_t(n)$, where $\chi_t(n) = -(2(A_t(n) - B_t(n))^{-1}\nabla F_t(n)$. The matrix $C_t(n) = (2A_t(n) - B_t(n)) \in \mathbb{R}^{M \times M}$ for every iteration $n$ is such that, for every $i-$th row in $C_t$ the sum of absolute value of off-diagonal entries $(\sum_{j \neq i} |(C_t)_{ij}|)$ is less than the absolute value of the diagonal element $(|(C_t)_{ii}|)$ in that row, i.e. $\sum_{j \neq i} |(C_t)_{ij}| < |(C_t)_{ii}|$. Thus, by Gerhgorin's Circle Theorem [158] all the eigen values of $C_t(n)$ are positive. Hence $\big(C_t(n)\big)^{-1}$ is positive definite and the descent direction $\chi_t(n)$ is such that $\chi_t(n)^T \nabla F_t(n) \leq 0$, where the equality holds true only for the case when $\nabla F_t(n) = 0$. Therefore $\chi_t(n)$ is the descent direction and the current iteration scheme is a Descent Method which guarantees convergence to a local minimum.

### A.1.3 Proof of Theorem 2

**Definitions of $\Lambda_\gamma$, $K_\gamma$:** $\Lambda_\gamma, K_\gamma \in \mathbb{R}^{M+1}$, $[\Lambda_\gamma]_n = \psi_n - \psi_{n-1}$, $[K_\gamma]_n = \xi(\gamma_n) - \xi(\gamma_{n-1})$, $\mathbb{I} \in \mathbb{R}^{(M+1)\times(M+1)}$ is an identity matrix, $\xi(\gamma_k)$ is the spatial coordinate of $\gamma_k$, $0 \leq k \leq M+1$.

**Proof of Theorem 2:** The solution $y$ to (2.13) no longer implies a (local) minimum to the cost function as soon as the second order condition in (2.15)

fails. There exists a direction $\psi$ along which the cost can decrease, thereby implying that $y$ is not the minimum. In fact, perturbation of $y$ at such critical $\beta$ and re-solving (2.13) results in a new solution $y$. (as done in step 4 of the annealing algorithm), which has more number of distinct locations $\{y_j\}$. To obtain this critical value of $\beta$, we compute $\frac{\partial^2 F}{\partial \epsilon^2}$ as in (2.15).

We claim that the expression of hessian $\frac{\partial^2 F}{\partial \epsilon^2}$ in (2.15) is non-negative for all finite perturbation $\psi$ if and only if the matrix $[\mathbb{I} - 2\beta\Upsilon_\gamma]$ is positive definite. The 'If' part is straightforward since the second term in the expression is non-negative. For the 'only If' part we show that when $[\mathbb{I} - 2\beta\Upsilon_\gamma]$ is not positive definite, there exists a finite perturbation $\psi$ such that the second term becomes zero thereby making the entire expression in (2.15) negative. Let us assume that there exists a transportation path $\gamma \in \mathcal{G}$ with positive probability such that the matrix $[\mathbb{I} - 2\beta\Upsilon_\gamma]$ is not positive definite. In fact, we assume there are several coincident facilities which result into several coincident transportation paths $\gamma \in \mathcal{G}$ such that $[\mathbb{I} - 2\beta\Upsilon_\gamma]$ is not positive definite. Under such circumstances we see that for the finite perturbation $\Lambda_\gamma = 0 \ \ \forall \ \ \gamma \neq \hat{\gamma}$ and $\sum_{\gamma \in \mathcal{G}: \gamma = \hat{\gamma}} \Lambda_\gamma = 0$, the second term in (2.15) is zero. Thus whenever the first term in (2.15) is not positive definite we can construct the above perturbation such that the second term vanishes. Hence the positivity of the expression in (2.15) for all finite perturbations $\psi$ depends solely on the positive definiteness of $[\mathbb{I} - 2\beta\Upsilon_\gamma]$. The phase transition occurs when the matrix $[\mathbb{I} - 2\beta\Upsilon_\gamma]$ loses its positive definiteness; i.e. $\det\big[\mathbb{I} - 2\beta\Upsilon_\gamma\big] = 0 \Rightarrow \beta_{cr}(\gamma) = \frac{1}{2\lambda_{max}(\gamma)}$ where $\lambda_{max}$ is the largest eigenvalue of $\Upsilon_\gamma$. We consider the $\beta_{cr} = \max_\gamma \beta_{cr}(\gamma)$ as we anneal $\beta$ from a large value to zero. The above derivation is analogous to the DA algorithm in [58].

### A.1.4 Proof of Theorems 3, 4, and 5

**A. Theorem 3:** Part (a), we note that $e^{-\beta \sum_{t=0}^{M} d_t(\gamma_t, \gamma_{t+1})} < 1$ since $d_t(\cdot, \cdot) \geq 0$.Therefore $\log \sum_{\mathcal{G}} e^{-\beta \sum_{t=0}^{M} d_t(\gamma_t, \gamma_{t+1})} < \log |\mathcal{G}|$. The result follows since $\sum_{\gamma_0} \rho_{\gamma_0} = 1$. Part (b) of the theorem follow directly from the expression of $F$ in (2.12). Part (c): At the instant when $y(t) = y_c(t)$ we have that $\bar{y}(t) = 0$. Hence the derivative of free-energy $\dot{F}(t)$ is given by $\dot{F} = x^T \hat{P}_{\gamma_0} \Phi + [z^T - y_c^T \hat{C}]\psi$, which is independent of $\bar{u}$. Hence $\frac{\partial \dot{F}}{\partial u} = 0$.

**B. Theorem 4:** Substituting $\bar{u}(t)$ (2.31) in $\dot{F}$ we obtain $\dot{F} = -K_0 \bar{y}^T (2\hat{A} -$

$\hat{B})\bar{y} - (\alpha^2 + (\bar{y}(2\hat{A} - \hat{B})\bar{y})^2)^{1/2}$ where $K_0 > 0$ and $(2\hat{A} - \hat{B})$ positive definite (as shown in Appendix A.1.1). Hence $\dot{F} \leq 0$. We know from Theorem 3 that the free-energy function $F$ is lower bounded and from above we have that for the control $\bar{u}(t)$ in (2.31) $\dot{F} \leq 0$. We conclude from here that $F(t)$ converges (say to $F_\infty$, where $|F_\infty| < \infty$) and $|\dot{F}(t)| \to 0$ as $t \to \infty$. Now since $\dot{F} = -K_0\bar{y}^T(2\hat{A} - \hat{B})\bar{y} - (\alpha^2 + (\bar{y}(2\hat{A} - \hat{B})\bar{y})^2)^{1/2}$, we have that $K_0\bar{y}^T(2\hat{A} - \hat{B})\bar{y} \leq |\dot{F}|$. Thus we conclude that $\bar{y}(t) \to 0$ as $t \to \infty$.

**C. Theorem 5:** Note- The proof here is similar to the proof for Proposition 3.43 in [68]. Since $(2\hat{A} - \hat{B})$ and $\Phi$ are Lipschitz, it is enough to show that $\bar{u}$ is Lipschitz at $\bar{\zeta} = 0$. Since $\underline{\hat{u}} \triangleq \hat{u} - (2\hat{A} - \hat{B})^{-1}(\hat{P}^0(\gamma_1, \gamma_0)^T\Phi + \hat{C}\psi)$ is Lipschitz at $\bar{\zeta} = 0$, there exists a neighborhood $B_\delta \triangleq \{\bar{\zeta} : \|\bar{\zeta}\| \leq \delta\}$ and $\bar{k} > 0$ such that $\|\underline{\hat{u}}(\bar{\zeta})\| \leq \bar{k}\|\bar{\zeta}\|\ \forall \bar{\zeta} \in B_\delta$. Also $\dot{F} = \alpha + \bar{y}^T(2\hat{A} - \hat{B})\underline{\hat{u}}(t) \leq 0$ where $\alpha = [x^T\hat{P}_{\gamma_0} - y_c^T\hat{P}_0(\gamma_1|\gamma_0)^T]\Phi + [z^T - y_c^T\hat{C}]\psi$. If $\alpha > 0$, then $|\alpha| \leq |\bar{y}^T(2\hat{A} - \hat{B})\underline{\hat{u}}| \leq \bar{k}_1\|\bar{y}\|\|\bar{\zeta}\|\ \forall \bar{\zeta} \in B_\delta$ where $\bar{k}_1 = k\lambda_{\max}(2\hat{A} - \hat{B})$. Thus the control design $\bar{u}$ (2.31) can be bounded above as $\|\bar{u}\| \leq (2\bar{k} + K_0 + 1)\|\bar{\zeta}\|$. For the case when $\alpha < 0$, we have that $\|\bar{u}\| \leq (1 + K_0)\|\bar{y}\| \leq (1 + K_0)\|\bar{\zeta}\|$.

## A.2 Parameterized MDPs and Reinforcement Learning

### A.2.1 Proof of Lemma 1 and Theorem 6

**Proof of Lemma 1:** Let $\bar{x}_0 = s$. By Assumption 1 $\exists$ a path $\omega = (\bar{u}_0, \bar{x}_1, \ldots, \bar{x}_N = \delta)$ such that $p_{\bar{\mu}}(\omega|x_0 = s) > 0$ which implies $p(x_{k+1} = \bar{x}_{k+1}|x_k = \bar{x}_k, u_k = \bar{u}_k) > 0$ by (3.5). Then, probability $p_\mu(\omega|x_0 = s)$ of taking path $\omega$ under the stochastic policy $\mu \in \Gamma$ in (3.3) is also positive.

**Proof of Theorem 6:** The following Lemma is needed

**Lemma 3.** *The Shannon Entropy $H^\mu(\cdot)$ corresponding to the MDP illustrated in Section 3.2.1 satisfies the algebraic expression $\sum_{s'} p^a_{ss'}H^\mu(s') = \sum_{s'} p^a_{ss'}\log p^a_{ss'} + \log\mu_{a|s} + \lambda_s + 1$.*

*Proof.* $H^\mu(\cdot)$ in (3.4) satisfies the recursive Bellman equation

$$H^\mu(s') = \sum_{a's''} \mu_{a'|s'}p^{a'}_{s's''}\big[-\log p^{a'}_{s's''} - \log\mu_{a'|s'} + H^\mu(s'')\big]$$

On the right side of the above Bellman equation, we subtract a zero term

$\lambda_{s'}(\sum_a \mu_{a|s'} - 1)$ that accounts for normalization constraint $\sum_a \mu_{a|s'} = 1$ and $\lambda_{s'}$ is a constant. Taking the derivative of the resulting expression with respect to $\mu_{a|s}$ we obtain

$$\frac{\partial H^\mu(s')}{\partial \mu_{a|s}} = \rho(s,a)\delta_{ss'} + \sum_{a',s''} \mu_{a'|s'} p^{a'}_{s's''} \frac{\partial H^\mu(s'')}{\partial \mu_{a|s}} - \lambda_{s'}\delta_{ss'}, \tag{A.1}$$

where $\rho(s,a) = -\sum_{s''} p^a_{ss''}(\log p^a_{ss''} - H^\mu(s'')) - \log \mu_{a|s} - 1$. The subsequent steps in the proof involve algebraic manipulations. Note that $p_\mu(s'|s) := \sum_a \mu(a|s)p(s'|s,a)$. Further, note that $p_\mu(s') = \sum_s p_\mu(s',s)$, where $p_\mu(s',s) = p_\mu(s)p_\mu(s'|s)$. Under the trivial assumption that for each state $s'$ there exists a state-action pair $(s,a)$ such that the probability of the system to enter the state $s'$ upon taking action $a$ in the state $s$ is non-zero ( i.e. $p^a_{ss'} > 0$) we have that $p_\mu(s') > 0$. Now, we multiply equation (A.1) by $p_\mu(s')$ and add over all $s' \in \mathcal{S}$ to obtain

$$\sum_{s'} p_\mu(s') \frac{\partial H^\mu(s')}{\partial \mu_{a|s}} = p_\mu(s')\rho(s,a) + \sum_{s''} p_\mu(s'') \frac{\partial H^\mu(s'')}{\partial \mu_{a|s}} - p_\mu(s)\lambda_s,$$

where $p_\mu(s'') = \sum_{s'} p_\mu(s')p_\mu(s''|s')$. The derivative terms on both sides cancel to give $\rho(s,a) - \lambda_s = 0$ which implies $\sum_{s'} p^a_{ss'} H^\mu(s') = \sum_{s'} p^a_{ss'} \log p^a_{ss'} + \log \mu_{a|s} + \lambda_s + 1$. □

Now consider the free energy function $V^\mu_\beta(s)$ in (3.6) and separate out the $t = 0$ term in its infinite summation to obtain

$$\begin{aligned}
V^\mu_\beta(s) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t c^{u_t}_{x_t x_{t+1}} + \frac{1}{\beta}\log p^{u_t}_{x_t x_{t+1}} + \frac{1}{\beta}\log \mu(u_t|x_t)\Big| x_0 = s\Big] \\
V^\mu_\beta(s) &= \sum_{s',a} \mu(a|s) p^a_{ss'} \big(c^a_{ss'} + \frac{1}{\beta}\log p^a_{ss'} + \frac{1}{\beta}\log \mu(a|s)\big) \\
&\quad + \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t c^{u_t}_{x_t x_{t+1}} + \frac{1}{\beta}\log p^{u_{t'}}_{x_{t'} x_{t'+1}} + \frac{1}{\beta}\log \mu(u_t|x_t)\Big| x_0 = s\Big] \\
&\quad \text{let} \qquad t = t' + 1, \qquad u'_{t'} = u_{t'+1}, \qquad x'_{t'} = x_{t'+1} \\
\Rightarrow V^\mu_\beta(s) &= \sum_{s',a} \mu(a|s) p^a_{ss'} \big(c^a_{ss'} + \frac{1}{\beta}\log p^a_{ss'} + \frac{1}{\beta}\log \mu(a|s)\big) \\
&\quad + \mathbb{E}\Big[\sum_{t'=0}^{\infty} \gamma^{t'+1} c^{u'_{t'}}_{x'_{t'} x'_{t'+1}} + \frac{1}{\beta}\log p^{u'_{t'}}_{x'_{t'} x'_{t'+1}} + \frac{1}{\beta}\log \mu(u'_{t'}|x'_{t'})\Big| x_0 = s\Big] \\
&= \sum_{s',a} \mu(a|s) p^a_{ss'} \big(c^a_{ss'} + \frac{1}{\beta}\log p^a_{ss'} + \frac{1}{\beta}\log \mu(a|s)\big)
\end{aligned}$$

$$
\begin{aligned}
&+\gamma\mathbb{E}\Big[\sum_{t'=0}^{\infty}\gamma^{t'}c^{u'_{t'}}_{x'_{t'}x'_{t'+1}}+\frac{1}{\gamma\beta}\log p^{u'_{t'}}_{x'_{t'}x'_{t'+1}}+\frac{1}{\gamma\beta}\log\mu(u'_{t'}|x'_{t'})\Big|x_0=s\Big]\\
&=\sum_{s',a}\mu(a|s)p^a_{ss'}\big(c^a_{ss'}+\frac{1}{\beta}\log p^a_{ss'}+\frac{1}{\beta}\log\mu(a|s)\big)\\
&+\gamma\mathbb{E}\Big[\mathbb{E}\Big[\sum_{t'=0}^{\infty}\gamma^{t'}c^{u'_{t'}}_{x'_{t'}x'_{t'+1}}+\frac{1}{\gamma\beta}\log p^{a'_{t'}}_{x'_{t'}x'_{t'+1}}\\
&\quad+\frac{1}{\gamma\beta}\log\mu(a'_{t'}|x'_{t'})\Big|x'_0=s'\Big]\Big|x_0=s\Big]\\
&=\sum_{s',a}\mu(a|s)p^a_{ss'}\big(c^a_{ss'}+\frac{1}{\beta}\log p^a_{ss'}+\frac{1}{\beta}\log\mu(a|s)\big)+\gamma\sum_{a,s'}\mu(a|s)p^a_{ss'}V^\mu_{\gamma\beta}(s')\\
\Rightarrow V^\mu_\beta(s)&=\sum_{a,s'}\mu_{a|s}p^a_{ss'}\Big[c^a_{ss'}+\frac{1}{\beta}\log p^a_{ss'}+\frac{1}{\beta}\log\mu_{a|s}+\gamma V^\mu_{\gamma\beta}(s')\Big],
\end{aligned}
\tag{A.2}
$$

where $V^\mu_{\gamma\beta}(s') := J^\mu(s')-\frac{1}{\gamma\beta}H^\mu(s')$. Now we relate $V^\mu_{\gamma\beta}(s')$ with $V^\mu_\beta(s')$ by adding and subtracting $-\frac{1}{\beta}H^\mu(s')$ to $V^\mu_{\gamma\beta}(s')$. We obtain $V^\mu_{\gamma\beta}(s')=V^\mu_\beta(s')-\frac{1-\gamma}{\gamma\beta}H(s')$. Substituting $V_{\gamma\beta}(s')$ and the algebraic expression obtained in Lemma 3 in the above equation (A.2) we obtain

$$
\begin{aligned}
V^\mu_\beta(s)&=\sum_{s',a}\mu(a|s)p^a_{ss'}\big(c^a_{ss'}+\frac{1}{\beta}\log p^a_{ss'}+\frac{1}{\beta}\log\mu(a|s)\big)\\
&\quad+\gamma\sum_{a,s'}\mu(a|s)p^a_{ss'}V^\mu_\beta(s')-\frac{1-\gamma}{\beta}\sum_{a,s'}\mu(a|s)p^a_{ss'}H^\mu(s')\\
\Rightarrow V^\mu_\beta(s)&=\sum_{s',a}\mu(a|s)p^a_{ss'}\big(c^a_{ss'}+\frac{1}{\beta}\log p^a_{ss'}+\frac{1}{\beta}\log\mu(a|s)\big)\\
&\quad+\gamma\sum_{a,s'}\mu(a|s)p^a_{ss'}V^\mu_\beta(s')-\frac{1-\gamma}{\beta}\sum_{a}\mu(a|s)\sum_{s'}p^a_{ss'}H^\mu(s')\\
\Rightarrow V^\mu_\beta(s)&=\sum_{s',a}\mu(a|s)p^a_{ss'}\big(c^a_{ss'}+\frac{1}{\beta}\log p^a_{ss'}+\frac{1}{\beta}\log\mu(a|s)\big)\\
&\quad+\gamma\sum_{a,s'}\mu(a|s)p^a_{ss'}V^\mu_\beta(s')\\
&\quad-\frac{1-\gamma}{\beta}\sum_{a}\mu(a|s)\Big(\sum_{s'}p^a_{ss'}\log p^a_{ss'}+\log\mu(a|s)+\lambda_s+1\Big)\\
\Rightarrow V^\mu_\beta(s)&=\sum_{a,s'}\mu(a|s)p^a_{ss'}\big(c^a_{ss'}+\frac{\gamma}{\beta}\log p^a_{ss'}+\frac{\gamma}{\beta}\log\mu(a|s)\big)\\
&\quad+\gamma\sum_{a,s'}\mu(a|s)p^a_{ss'}V^\mu_\beta(s')+c_0(s)
\end{aligned}
\tag{A.3}
$$

where $c_0(s)=-\frac{1-\gamma}{\beta}(\lambda_s+1)$ does not depend on the policy $\mu$. Therefore, since the control policy $\mu^*(a|s)$ is determined by taking critical points of

$V^{\mu}_{\beta}(s)$, it is independent of $c_0(s)$ as shown in the following subsection.

### A.2.2 Independence of policy on $c_0$

The Bellman equation is given by

$$V^{\mu}_{\beta}(s) = \sum_{a,s'} \mu(a|s) p^a_{ss'} \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} + \frac{\gamma}{\beta} \log \mu(a|s) \big) + \gamma \sum_{a,s'} \mu(a|s) p^a_{ss'} V^{\mu}_{\beta}(s') + c_0(s) \tag{A.4}$$

The optimal control policy $\mu^*_{\beta}(a|s)$ obtained upon taking the derivative of the recursive Bellman equation (and also accounting for $\sum_a \mu(a|s) = 1$) is given by

$$\mu^*_{\beta}(a|s) = \frac{\exp\{-\frac{\beta}{\gamma}\bar{\Lambda}^*_{\beta}(s,a)\}}{\sum_{a'} \exp\{-\frac{\beta}{\gamma}\bar{\Lambda}^*_{\beta}(s,a')\}}, \quad \text{where} \tag{A.5}$$

$$\bar{\Lambda}^*_{\beta}(s,a) = \sum_{s'} p^a_{ss'} \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} + \gamma V^*_{\beta}(s') \big), \tag{A.6}$$

Substituting the optimal policy $\mu^*_{\beta}(a|s)$ (A.5) into the recursive Bellman in (A.4) we obtain

$$V^*_{\beta}(s) = -\frac{\gamma}{\beta} \log \sum_a \exp \Big\{ -\frac{\beta}{\gamma} \bar{\Lambda}^*_{\beta}(s,a) \Big\} + c_0(s). \tag{A.7}$$

Substituting the above equation (A.7) into the state-action value function in (A.6) we obtain the following map

$$\bar{\Lambda}^*_{\beta}(s,a) = \sum_{s'} p^a_{ss'} \Big[ \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} \big) - \frac{\gamma^2}{\beta} \log \sum_a \exp \Big\{ -\frac{\beta}{\gamma} \bar{\Lambda}^*_{\beta}(s,a) \Big\} \Big] + c_0(s) =: [T_1 \bar{\Lambda}_{\beta}](s,a), \tag{A.8}$$

where the proof that the map $T_1 : \bar{\Lambda}_{\beta} \to \bar{\Lambda}_{\beta}$ is a contraction map is analogous to the proof of Theorem 7 in Appendix A.2.2 of the thesis (see the remark below).

The state-action value $\Lambda^*_{\beta}(s,a)$ obtained by disregarding $c_0(s)$ in (A.7)

satisfies the recursive equation

$$\Lambda^*_\beta(s,a) = \sum_{s'} p^a_{ss'} \Big[ \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} \big) - \frac{\gamma^2}{\beta} \log \sum_a \exp \Big\{ - \frac{\beta}{\gamma} \Lambda^*_\beta(s,a) \Big\} \Big]$$
$$=: [T\Lambda_\beta](s,a), \tag{A.9}$$

where the map $T : \Lambda_\beta \to \Lambda_\beta$ has been shown to be a contraction map in the Theorem 7 of the thesis.

*Remark:* Note that $[T_1 x](s,a) = [Tx](s,a) + c_0(s)$. Therefore, $T_1$ is a contraction since $T$ is contraction and has a unique fixed point.

**Claim:** The optimal control policy $\mu^*_\beta(a|s)$ in (A.5) is same if computed using either $\bar{\Lambda}^*_\beta(s,a)$ or $\Lambda^*_\beta(s,a)$. This is so because the fixed points $\bar{\Lambda}^*_\beta(s,a)$ and $\Lambda^*_\beta(s,a)$ of the contraction maps $T_1 : \bar{\Lambda}_\beta \to \bar{\Lambda}_\beta$ and $T : \Lambda_\beta \to \Lambda_\beta$, respectively, differ by $\frac{1}{1-\gamma} c_0(s)$, i.e. $\bar{\Lambda}^*_\beta(s,a) = \Lambda^*_\beta(s,a) + \frac{1}{1-\gamma} c_0(s)$. If we plug the above relation into the control policy in (A.5) we obtain

$$\mu^*_\beta(a|s) = \frac{\exp\{-\frac{\beta}{\gamma}\Lambda^*_\beta(s,a) - \frac{\beta/\gamma}{1-\gamma} c_0(s)\}}{\sum_{a'} \exp\{-\frac{\beta}{\gamma}\Lambda^*_\beta(s,a') - \frac{\beta/\gamma}{1-\gamma} c_0(s)\}}$$
$$\Rightarrow \mu^*_\beta(a|s) = \frac{\exp\{-\frac{\beta}{\gamma}\Lambda^*_\beta(s,a)\}}{\sum_{a'} \exp\{-\frac{\beta}{\gamma}\Lambda^*_\beta(s,a')\}}$$

Thus, indicating that we do not need to take $c_0(s)$ into account while computing the optimal policy.

**Proof for $\bar{\Lambda}^*_\beta(s,a) = \Lambda^*_\beta(s,a) + \frac{1}{1-\gamma} c_0(s)$:** This can be checked by substituting this expression into the definition of the map $[T_1 \bar{\Lambda}_\beta]$ in (A.8); we get

$$T_1 \bar{\Lambda}^*_\beta(s,a) = \sum_{s'} p^a_{ss'} \Big[ \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} \big) - \frac{\gamma^2}{\beta} \log \sum_a \exp \Big\{ - \frac{\beta}{\gamma} \Lambda^*_\beta(s,a) - \frac{\beta/\gamma}{1-\gamma} c_0(s) \Big\} \Big] + c_0(s)$$

$$T_1 \bar{\Lambda}^*_\beta(s,a) = \sum_{s'} p^a_{ss'} \Big[ \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} \big)$$

$$- \frac{\gamma^2}{\beta} \log \sum_a \exp\Big\{ - \frac{\beta}{\gamma} \Lambda^*_\beta(s,a) \Big\} \Big] + \frac{\gamma}{1-\gamma} c_0(s) + c_0(s)$$

$$T_1 \bar{\Lambda}^*_\beta(s,a) = \sum_{s'} p^a_{ss'} \Big[ \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} \big) - \frac{\gamma^2}{\beta} \log \sum_a \exp\Big\{ - \frac{\beta}{\gamma} \Lambda^*_\beta(s,a) \Big\} \Big] + \frac{1}{1-\gamma} c_0(s)$$

$$T_1 \bar{\Lambda}^*_\beta(s,a) = T\Lambda^*_\beta(s,a) + \frac{1}{1-\gamma} c_0(s)$$

$$\bar{\Lambda}^*_\beta(s,a) = \Lambda^*_\beta(s,a) + \frac{1}{1-\gamma} c_0(s)$$

Hence proved.

### A.2.3 Proof of Theorem 7

The following lemma will be used.

**Lemma 4.** *For every policy $\mu \in \Gamma$ defined in (3.3) there exists a vector $\xi = (\xi_s) \in \mathbb{R}^{|\mathcal{S}|}_+$ with positive components and a scalar $\lambda < 1$ such that $\sum_{s'} p^a_{ss'} \xi_{s'} \leq \lambda \xi_s$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.*

*Proof.* Consider a new MDP with state transition probabilities similar to the original MDP and the transition costs $c^a_{ss'} = -1 - \frac{1}{\beta} \log(|\mathcal{A}||\mathcal{S}|)$ except when $s = \delta$. Thus, the free-energy function $V^\mu_\beta(s)$ in (3.6) for the new MDP is less than or equal to $-1$. We define $-\xi_s \triangleq V^*_\beta(s)$ (as given in 3.10)) and use LogSumExp [159] inequality to obtain $-\xi_s \leq \min_a \Lambda_\beta(s,a) \leq \Lambda_\beta(s,a)\ \forall\ a \in \mathcal{A}$ where $\Lambda_\beta(s,a)$ is the state action value function in (3.11). Thus, $-\xi_s \leq \sum_{s'} p^a_{ss'} \big( c^a_{ss'} + \frac{\gamma}{\beta} \log p^a_{ss'} - \gamma \xi_{s'} \big)$ and upon substituting $c^a_{ss'}$ we obtain $-\xi_s \leq -1 - \gamma \sum_{s'} p^a_{ss'} \xi_{s'} \leq -1 - \sum_{s'} p^a_{ss'} \xi_{s'}$.

$$\Rightarrow \sum_{s' \in \mathcal{S}} p^a_{ss'} \xi_{s'} \leq \xi_s - 1 \leq \Big[ \max_s \frac{\xi_s - 1}{\xi_s} \Big] \xi_s =: \lambda \xi_s.$$

Since $V^*_\beta(s) \leq -1 \Rightarrow \xi_s - 1 \geq 0$ and thus $\lambda < 1$. □

Next we show that $T : \Lambda_\beta \to \Lambda_\beta$ in (3.11) is a contraction map. For any $\hat{\Lambda}_\beta$ and $\check{\Lambda}_\beta$ we have that $[T\hat{\Lambda}_\beta - T\check{\Lambda}_\beta](s,a)$

$$= -\frac{\gamma^2}{\beta} \sum_{s' \in \mathcal{S}} p^a_{ss'} \log \frac{\sum_a \exp\big( - \frac{\beta}{\gamma} \hat{\Lambda}_\beta(s',a) \big)}{\sum_{a'} \exp\big( - \frac{\beta}{\gamma} \check{\Lambda}_\beta(s',a') \big)}$$

$$\geq \gamma \sum_{s',a'} p^a_{ss'} \hat{\mu}_{a'|s'} (\hat{\Lambda}_\beta(s',a') - \check{\Lambda}_\beta(s',a')) =: \gamma \Delta_{\hat{\mu}}, \tag{A.10}$$

where we use the Log sum inequality to obtain (A.10), and $\hat{\mu}_{a|s}$ is the stochastic policy in (3.8) corresponding to $\hat{\Lambda}_\beta(s,a)$. Similarly, we obtain $[T\check{\Lambda}_\beta - T\hat{\Lambda}_\beta](s,a) \geq -\gamma \sum_{s',a'} p^a_{ss'} \check{\mu}_{a'|s'} (\hat{\Lambda}_\beta(s',a') - \check{\Lambda}_\beta(s',a')) =: -\gamma \Delta_{\check{\mu}}$ where $\check{\mu}_{a|s}$ is the policy in (3.8) corresponding to $\check{\Lambda}_\beta(s,a)$. Now from $\gamma\Delta_{\hat{\mu}} \leq [T\hat{\Lambda}_\beta - T\check{\Lambda}_\beta](s,a) \leq \gamma\Delta_{\check{\mu}}$ we conclude that $|[T\hat{\Lambda}_\beta - T\check{\Lambda}_\beta](s,a)| \leq \gamma \Delta_{\bar{\mu}}(s,a)$ where $\Delta_{\bar{\mu}}(s,a) = \max\{|\Delta_{\hat{\mu}}(s,a)|, |\Delta_{\check{\mu}}(s,a)|\}$ and we have $|[T\hat{\Lambda}_\beta - T\check{\Lambda}_\beta](s,a)|$

$$\leq \gamma \sum_{s',a'} p^a_{ss'} \bar{\mu}_{a'|s'} |\hat{\Lambda}_\beta(s',a') - \check{\Lambda}_\beta(s',a')| \tag{A.11}$$

$$\leq \gamma \sum_{s',a'} p^a_{ss'} \xi_{s'} \bar{\mu}_{a'|s'} \|\hat{\Lambda}_\beta - \check{\Lambda}_\beta\|_\xi \tag{A.12}$$

where $\|\Lambda_\beta\|_\xi = \max_{s,a} \frac{\Lambda_\beta(s,a)}{\xi_s}$ and $\xi \in \mathbb{R}^{\mathcal{S}}$ is as given in Lemma 4. Further, from the same Lemma we obtain

$$|[T\hat{\Lambda}_\beta - T\check{\Lambda}_\beta](s,a)| \leq \gamma\lambda\xi_s \sum_{a' \in \mathcal{A}} \bar{\mu}_{a'|s'} \|\hat{\Lambda}_\beta - \check{\Lambda}_\beta\|_\xi \tag{A.13}$$

$$\Rightarrow \|T\hat{\Lambda}_\beta - T\check{\Lambda}_\beta\|_\xi \leq \gamma\lambda \|\hat{\Lambda}_\beta - \check{\Lambda}_\beta\|_\xi \text{ with } \gamma\lambda < 1 \tag{A.14}$$

### A.2.4 Proof of Theorem 8

The proof follows the similar idea as the proof for Theorem 6 in Appendix A.2.1 and thus, we do not explain it in detail to avoid redundancy except the following Lemma that illustrates the algebraic structure of the discounted Shannon entropy $H^\mu_d(\cdot)$ in (3.15) which is different from that in Lemma 3 and also required in our proof of the said theorem.

**Lemma 5.** *The discounted Shannon entropy $H^\mu_d(\cdot)$ corresponding to the MDP in Section 3.3 satisfies the algebraic term*

$$\alpha \sum_{s'} p^a_{ss'} H^\mu_d(s') = \sum_{s'} p^a_{ss'} \log p^a_{ss'} + \log \alpha\mu(a|s) + \lambda_s + 1.$$

*Proof.* Define a new MDP that augments the action and state spaces $(\mathcal{A}, \mathcal{S})$ of the original MDP with an additional action $a_e$ and state $s_e$, respectively, and derives its state-transition probability $\{q^a_{ss'}\}$ and policy $\{\zeta_{a|s}\}$ from original

MDP as

$$q^a_{ss'} = \begin{cases} p^a_{ss'} & \forall s, s' \in \mathcal{S}, a \in \mathcal{A} \\ 1 & \text{if } s', a = s_e, a_e \\ 1 & \text{if } s' = s = s_e \\ 0 & \text{otherwise} \end{cases} \quad \zeta_{a|s} = \begin{cases} \alpha\mu_{a|s} & \forall (s,a) \in (\mathcal{S}, \mathcal{A}) \\ 1-\alpha & \text{if } a = a_e, s \in \mathcal{S} \\ 0 & \text{if } a \in \mathcal{A}, s = s_e \\ 1 & \text{if } a = a_e, s = s_e \end{cases} \tag{A.15}$$

Next, we define $T^\mu := \alpha H^\mu_d$ that satisfies

$$T^\mu(s') = \sum_{a's''} \eta_{a'|s'} p^{a'}_{s's''} \big[ -\log p^a_{s's''} - \log \eta_{a'|s'} + T^\mu(s'') \big]$$

derived using (3.15) where $\eta_{a'|s'} = \alpha\mu_{a'|s'}$. The subsequent steps of the proof are same as the proof of Lemma 3. □

### A.2.5 Proof of Proposition 1, 2 and Theorem 9

*Proof of Proposition 1:* The proof in this section is analogous to the proof of Proposition 5.5 in [11]. Let $\bar{T}$ be the map in (3.14). The stochastic iterative updates in (3.13) can be re-written as $\bar{\Psi}_{t+1}(x_t,u_t) = (1-\nu_t(x_t,u_t))\bar{\Psi}_t(x_t,u_t) + \nu_t(x_t,u_t)\big([\bar{T}\bar{\Psi}_t](x_t,u_t) + w_t(x_t,u_t)\big)$ where

$$w_t(x_t,u_t) = c^{u_t}_{x_t x_{t+1}} - \frac{\gamma^2}{\beta}\log\sum_a \exp(-\frac{\beta}{\gamma}\bar{\Psi}_t(s_{t+1},a)) - \bar{T}\bar{\Psi}_t(x_t,u_t).$$

Let $\mathcal{F}_t$ represent the history of the stochastic updates, i.e.,

$$\mathcal{F}_t = \{\bar{\Psi}_0, \ldots, \bar{\Psi}_t, w_0, \ldots, w_{t-1}, \nu_0, \ldots, \nu_t\},$$

then $\mathbb{E}[w_t(x_t,u_t)|\mathcal{F}_t] = 0$ and $\mathbb{E}[w_t^2(x_t,u_t)|\mathcal{F}_t] \le K(1+\max_{s,a}\bar{\Psi}_t^2(s,a))$, where $K$ is a constant. These expressions satisfy the conditions on the expected value and the variance of $w_t(x_t,u_t)$ that along with the contraction property of $\bar{T}$ guarantees the convergence of the stochastic updates (3.13) as illustrated in the Proposition 4.4 in [11].

*Proof of Theorem 9:* We show that the map $T_1$ in (3.25) is a contraction map. For any $K^\beta_{\zeta_s}$ and $\bar{K}^\beta_{\zeta_s}$ we obtain that $|[T_1K^\beta_{\zeta_s} - T_1\bar{K}^\beta_{\zeta_s}](s')| \le \gamma\sum_{a,s''} p^a_{s's''}\mu_{a|s'}|K^\beta_{\zeta_s}(s'',a) - \bar{K}^\beta_{\zeta_s}(s'',a)|$. Note that this inequality is similar

to the one in (A.11); thus, we follow the exact same steps from (A.11) to (A.14) to show that $\|T_1 K^{\beta}_{\zeta_s} - T_1 \bar{K}^{\beta}_{\zeta_s}\|_{\xi} \leq \gamma\lambda \|K^{\beta}_{\zeta_s} - \bar{K}^{\beta}_{\zeta_s}\|_{\xi}$ and $\gamma\lambda < 1$.
*Proof of Proposition 2:* The proof in this section is similar to the proof of Proposition 1 in Appendix A.2.5. Additional conditions on the boundedness of the derivatives $\left|\frac{\partial c^a_{ss'}}{\partial \zeta_l}\right|$ and $\left|\frac{\partial c^a_{ss'}}{\partial \eta_k}\right|$ are required to bound the variance $\mathbb{E}[w_t^2|\mathcal{F}_t]$.

## A.3 Dynamic Parameterized MDPs

### A.3.1 Proof of Theorem 10

(a) By definition $\mathcal{V}(\Upsilon) = \sum_{s\in\mathcal{S}} V^*_{\beta,\Upsilon}(s) = \sum_{s\in\mathcal{S}} \min_{\mu\in\Gamma} V^{\mu}_{\beta,\Upsilon}(s)$, where

$$V^{\mu}_{\beta,\Upsilon}(s) := J^{\mu}_{\Upsilon}(s) - \frac{1}{\beta} H^{\mu}(s)$$

is the Lagrangian corresponding to the optimization problem (3.4). Thus, $\mathcal{V}(\Upsilon) \geq \sum_{s\in\mathcal{S}} \left( \min_{\mu\in\Gamma} J^{\mu}_{\Upsilon}(s) - \frac{1}{\beta} \max_{\mu\in\Gamma} H^{\mu}(s)\right)$. By definition of the set $\Gamma$, the maximum entropy $\max_{\mu\in\Gamma} H^{\mu}(s)$ is finite, and $min_{\mu\in\Gamma} J^{\mu}_{\Upsilon}(s) > 0$; thus, there exists a constant $c > 0$ such that $\mathcal{V} \geq -c$.
(b)When the parameters $\zeta_2$ and $\eta_2$ are at the local minima, the corresponding derivatives in $H = 0$; thus, it follows from the time-derivative $\dot{\mathcal{V}}$ in (4.7) of the control-Lyapunov function that there is no dynamic control authority at such instants.

### A.3.2 Proof of Theorem 11

Substituting $u(t)$ (4.8) in $\dot{\mathcal{V}}$ we obtain $\dot{\mathcal{V}} = -K_0 H^T H - \sqrt{(G^T\kappa)^2 + (H^T H)^2}$ where $K_0 > 0$. Hence $\dot{\mathcal{V}} \leq 0$. We know from Theorem 10 that the function $\mathcal{V}$ is lower bounded; from above we have that for the control $u(\Theta)$ in (4.8) $\dot{\mathcal{V}} \leq 0$. We conclude from here that $\mathcal{V}(t)$ converges (say to $\mathcal{V}_{\infty}$, where $|\mathcal{V}_{\infty}| < \infty$) and $|\dot{\mathcal{V}}(t)| \to 0$ as $t \to \infty$. Now since $\dot{\mathcal{V}} = -K_0 H^T H - \sqrt{(G^T\kappa)^2 + (H^T H)^2}$, we have that $K_0 H^T H \leq |\dot{\mathcal{V}}|$. Thus we conclude that $H(t) \to 0$ as $t \to \infty$.

### A.3.3 Proof of Theorem 12

Note - the proof here is similar to the proof for the Proposition 3.43 in [68]. Since $\hat{u}$ is Lipschitz at $\Theta = 0$, there exists a neighbourhood $B_r = \{\Theta : \|\Theta\| \leq r\}$ and $\bar{k} > 0$ such that $\|\hat{u}\| \leq \bar{k}\|\Theta\|$ for all $\Theta \in B_r$. Also, $\dot{\mathcal{V}} = G^T\kappa + H^T\hat{u} \leq 0$; if $G^T\kappa > 0$, then $|G^T\kappa| \leq |H^T\hat{u}| \leq \|H\|\|\hat{u}\| \leq \bar{k}\|H\|\|\Theta\|$ $\forall\ \Theta \in B_r$. Thus, the control design $u(\Theta)$ in (4.8) can be bounded above as $\|u\| \leq (K_0 + 2\bar{k} + 1)\|\Theta\|$. For the case when $G^T\kappa < 0$, we have that $\|u(\Theta)\| \leq (1 + K_0)\|H\| \leq (1 + K_0)\|\Theta\|$.

## A.4 Persistence of Data Clustering Solutions

### A.4.1 Additional Experimental Results

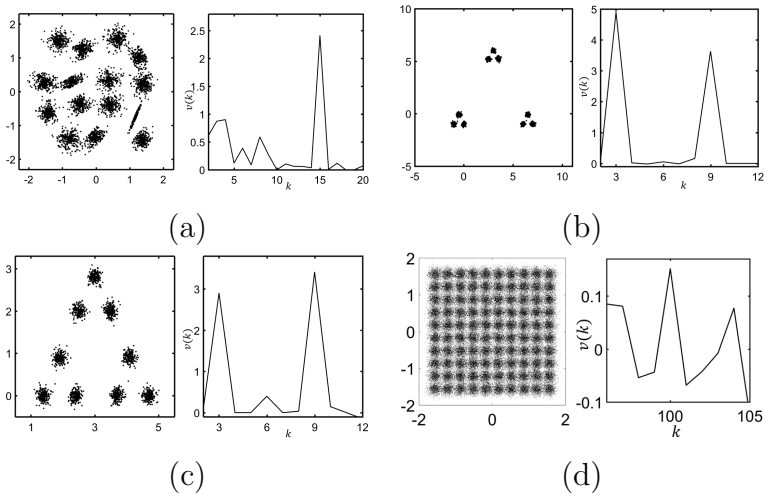


Figure A.1: (a) Synthetic 2-d data with $N = 5000$ vectors and $k_t = 15$ Gaussian clusters [4] with different degree of overlapping. The corresponding $v(k)$ versus $k$ plot for both the cases show a peak at $k = 15$. (b) The $v(k)$ versus $k$ plot indicates 3 to be true number of clusters. (c) The $v(k)$ versus $k$ plot indicates 9 to be true number of clusters. (d) The $v(k)$ versus $k$ plot indicates $k_t = 100$ for Birch1 dataset.

### A.4.2 Proof of Lemma 1

**Proof**: Let $(\lambda, \mathbf{u})$ be any eigenvector-eigenvalue pair of $A$. Here

$$\mathbf{u} \triangleq [u_1, u_2, \ldots, u_N]$$

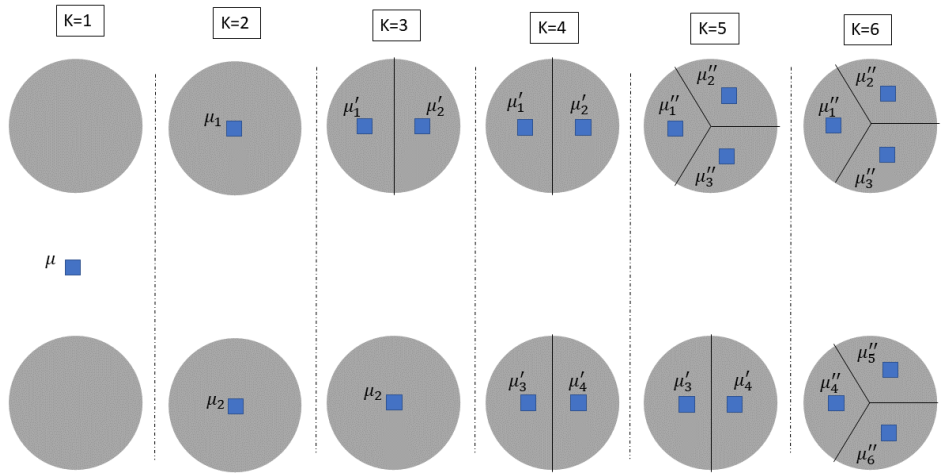


Figure A.2: The Figure illustrates the optimal clustering at each value of $k \in \{1, 2, 3, 4, 5, 6\}$ for the dataset in the Figure 5.3

. Then, by definition:

$$\sum_{l=1}^{N} A_{kl} u_l = \lambda u_k$$

$$\Rightarrow (\phi(\mathbf{x_k}) - \mathbf{y})^T \sum_{l=1}^{N} (\phi(\mathbf{x_l}) - \mathbf{y})\, u_l = \lambda u_k \tag{A.16}$$

Multiplying (A.16) by $(\phi(\mathbf{x_k}) - \mathbf{y})$ and then summing over $k$ yields:

$$\sum_{k=1}^{N} (\phi(\mathbf{x_k}) - \mathbf{y})\, (\phi(\mathbf{x_k}) - \mathbf{y})^T \underbrace{\sum_{l=1}^{N} (\phi(\mathbf{x_l}) - \mathbf{y})\, u_l}_{\tilde{u}} = \lambda \underbrace{\sum_{k=1}^{N} (\phi(\mathbf{x_k}) - \mathbf{y})\, u_k}_{\tilde{u}},$$

$$\Rightarrow C^k_{\phi(\mathcal{X})|\mathbf{y_j}} \tilde{u} = \lambda \tilde{u} \tag{A.17}$$

### A.4.3 Proof for $H(\mathcal{Y}, \Psi, \beta) = 0$ only when $\det(I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}) = 0$

We claim that $H(\mathcal{Y}, \Psi, \beta)$ is non-negative for all finite perturbation $\Psi$ if and only if the matrix $[I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}]$ is positive definite. The 'if' part is straightforward since the second term in the expression is non-negative. For the 'only if' part we show that when $[I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}]$ is negative semi-definite

definite, there exists a finite perturbation $\Psi$ such that the second term in $H(\mathcal{Y}, \Psi, \beta)$ becomes zero thereby making the entire term negative. Let us assume that there exists a $\mathbf{y_0} \in \mathcal{Y}$ with positive probability such that the matrix $[I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}]$ is negative semi-definite. The perturbation $\Psi$ be such that $\Psi_{\mathbf{y}} = 0 \ \forall \ \mathbf{y} \neq \mathbf{y_0}$ and $\sum_{\mathbf{y} \in \mathcal{Y}: \mathbf{y} = \mathbf{y_0}} \Psi_{\mathbf{y}} = 0$. Then the second term in $H(\mathcal{Y}, \Psi, \beta)$ becomes zero. Thus whenever the first term in $H(\mathcal{Y}, \Psi, \beta)$ is non-positive we can construct a perturbation such that the second term vanishes. Hence the positivity of the $H(\mathcal{Y}, \Psi, \beta)$ for all perturbations $\Psi$ depends solely on the positive definiteness of $[I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}]$. The $\mathcal{Y}$ is no longer a minimum of $F$ when $H(\mathcal{Y}, \Psi, \beta) = 0$ which happens when the matrix $[I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}]$ loses its positive definiteness; i.e.

$$\det[I - 2\beta C^k_{\mathcal{X}|\mathbf{y_0}}] = 0$$

The proof is as given in [58].

### A.4.4 Proof for $v(2) > v(k) \ \forall \ k \in \{3, 4, 5, 6\}$

Figure A.2 illustrates an optimal clustering solution at various values of $k$. We assume that each circle contains $N$ datapoints. We have rotated the original dataset in Figure 5.3 by 90 degrees for the ease of presentation; however, this will not change the corresponding problem and its solution. Note that Figure A.2 shows one of the optimal clusterings at each value of $k$; even if some other optimal clustering is chosen, due to symmetry, the cluster covariance matrix will remain unchanged. To evaluate $\bar{\beta}_k \ \forall \ k \in \{2, 3, 4, 5, 6\}$, we compute the relevant covariance matrices $\bar{C}^1_{\mathcal{X}|\mu}$, $\bar{C}^2_{\mathcal{X}|\mu_1}$, $\bar{C}^3_{\mathcal{X}|\mu_2}$, $\bar{C}^4_{\mathcal{X}|\mu'_1}$, $\bar{C}^5_{\mathcal{X}|\mu'_3}$ and $\bar{C}^6_{\mathcal{X}|\mu''_1}$.

- Computing $\bar{C}^2_{\mathcal{X}|\mu_1}$.

$$\begin{aligned}
\bar{C}^2_{\mathcal{X}|\mu_1} &= \sum_{x \in C_{\mu_1}} (x - \mu_1)(x - \mu_1)^T \\
&\approx \int_{r,\theta} (x - \mu_1)(x - \mu_1)^T \rho r dr d\theta, \\
&\text{where} \quad x = [r\cos\theta \quad r\sin\theta]^T \\
&= \int_0^{2\pi} \int_0^R \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} \begin{bmatrix} r\cos\theta & r\sin\theta \end{bmatrix} \rho r dr d\theta
\end{aligned}$$

we assume $\mu_1 = [0 \quad 0]^T$ without loss of generality

$$= \int_0^{2\pi} \int_0^R \begin{bmatrix} r^2\cos^2\theta & r^2\cos\theta\sin\theta \\ r^2\sin\theta\cos\theta & r^2\sin^2\theta \end{bmatrix} \rho r dr d\theta$$

$$= \begin{bmatrix} (\rho\pi R^2)R^2/4 & 0 \\ 0 & (\rho\pi R^2)R^2/4 \end{bmatrix}$$

$(\rho\pi R^2)$ denotes the number of data points $N$

$$= \frac{NR^2}{4} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow \lambda_{\max}(\bar{C}^2_{\mathcal{X}|\mu_1}) = NR^2/4$$

- Computing $\bar{C}^3_{\mathcal{X}|\mu_2}$. Due to symmetry between the cluster $\pi_{\mu_1}$ and $\pi_{\mu_2}$ we can see that $\bar{C}^2_{\mathcal{X}|\mu_1} = \bar{C}^3_{\mathcal{X}|\mu_2}$. Hence

$$\Rightarrow \lambda_{\max}(\bar{C}^3_{\mathcal{X}|\mu_2}) = NR^2/4$$

  In Figure A.2, we assume WLOG that $\pi_{\mu_1}$ splits and $\pi_{\mu_2}$ remains intact between $k = 2$ and $k = 3$.

- Computing $\bar{C}^1_{\mathcal{X}|\mu}$

$$\begin{aligned}
\bar{C}^1_{\mathcal{X}|\mu} &= \sum_{x \in C_\mu} (x-\mu)(x-\mu)^T \\
&= \sum_{x \in C_{\mu_1}} (x-\mu)(x-\mu)^T + \sum_{x \in C_{\mu_2}} (x-\mu)(x-\mu)^T \\
&= \sum_{x \in C_{\mu_1}} (x-\mu_1+(\mu_1-\mu))(x-\mu_1+(\mu_1-\mu))^T \\
&+ \sum_{x \in C_{\mu_2}} (x-\mu_2+(\mu_2-\mu))(x-\mu_2+(\mu_2-\mu))^T \\
&= \bar{C}^2_{\mathcal{X}|\mu_1} + \bar{C}^2_{\mathcal{X}|\mu_2} + 2N\big(\frac{\mu_1-\mu_2}{2}\big)\big(\frac{\mu_1-\mu_2}{2}\big)^T \\
&\text{using the definition of } \bar{C}^2_{\mathcal{X}|\mu_1} \text{ and } \bar{C}^2_{\mathcal{X}|\mu_2} \\
&\text{and } \mu = \frac{\mu_1+\mu_2}{2} \\
&\text{note that } \mu_1 - \mu_2 = [0 \quad 4R]^T \\
&= \begin{bmatrix} NR^2/2 & 0 \\ 0 & NR^2/2 + 8NR^2 \end{bmatrix}
\end{aligned}$$

$$= NR^2 \begin{bmatrix} 1 & 0 \\ 0 & 8.5 \end{bmatrix}$$

$$\Rightarrow \lambda_{\max}((\bar{C}^1_{\mathcal{X}|\mu})) = 8.5NR^2$$

- Computing $\bar{C}^4_{\mathcal{X}|\mu'_1}$. Since $\pi_{\mu'_1}$ is a semi-circle, we can easily locate the centroid $\mu'_1$ in it. Assuming that the origin $(0, 0)$ is at the center of the full circle corresponding to this semi-circle, hence we have that $\mu'_1 = [0 \quad \frac{4R}{3\pi}]^T$.

$$\begin{aligned} \bar{C}^4_{\mathcal{X}|\mu'_1} &= \sum_{x \in C_{\mu'_1}} (x - \mu'_1)(x - \mu'_1)^T \\ &\approx \int_0^R \int_0^\pi \begin{bmatrix} r\cos\theta \\ r\sin\theta - \frac{4R}{3\pi} \end{bmatrix} \begin{bmatrix} r\cos\theta \\ r\sin\theta - \frac{4R}{3\pi} \end{bmatrix}^T \rho r dr d\theta \\ &= NR^2 \begin{bmatrix} \frac{1}{8} & 0 \\ 0 & \frac{1}{8} + \frac{8}{9\pi^2} - \frac{16}{9\pi^2} \end{bmatrix} \end{aligned}$$

$$\Rightarrow \lambda_{\max}(\bar{C}^4_{\mathcal{X}|\mu'_1}) = 0.125NR^2$$

- Computing $\bar{C}^5_{\mathcal{X}|\mu'_3}$. Note that due to symmetry $\bar{C}^4_{\mathcal{X}|\mu'_1} = \bar{C}^5_{\mathcal{X}|\mu'_3}$.

$$\Rightarrow \lambda_{\max}(\bar{C}^5_{\mathcal{X}|\mu'_3}) = 0.125NR^2$$

- Computing $\bar{C}^6_{\mathcal{X}|\mu''_1}$. Can be calculated as done above. We get

$$\bar{C}^6_{\mathcal{X}|\mu''_1} = NR^2 \begin{bmatrix} 0.0165 & 0 \\ 0 & 0.049 \end{bmatrix}$$

$$\Rightarrow \lambda_{\max}(\bar{C}^6_{\mathcal{X}|\mu''_1}) = 0.049NR^2$$

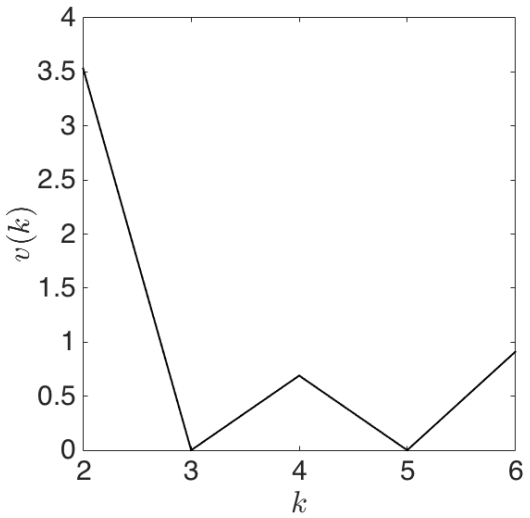


Figure A.3: Illustrates the $v(k)$ vs $k$ plot for the analytically calculated $v(k)$ at various $k$'s for the dataset in Figure 5.3. As shown, $k = 2$ is a more natural number of cluster than $k \in \{3, 4, 5, 6\}$.

Hence we have that $\bar{\beta}_1 = \frac{1}{17NR^2}$, $\bar{\beta}_2 = \frac{1}{2NR^2/4}$, $\bar{\beta}_3 = \frac{1}{2NR^2/4}$, $\bar{\beta}_4 = \frac{1}{0.125NR^2}$, $\bar{\beta}_5 = \frac{1}{0.125NR^2}$, $\bar{\beta}_6 = \frac{\pi}{0.98NR^2}$

$$
\begin{aligned}
v(2) &= \log \bar{\beta}_2 - \log \bar{\beta}_1 = 3.53 \\
v(3) &= \log \bar{\beta}_3 - \log \bar{\beta}_2 = 0 \\
v(4) &= \log \bar{\beta}_4 - \log \bar{\beta}_3 = 0.69 \\
v(5) &= \log \bar{\beta}_5 - \log \bar{\beta}_4 = 0 \\
v(6) &= \log \bar{\beta}_6 - \log \bar{\beta}_5 = 0.91
\end{aligned}
$$

Hence we have that $v(2) > v(k) \; \forall \; k \in \{3, 4, 5, 6\}$ as summarized in the Figure A.3

## A.5 Markov Chain Aggregation

Perturbation $\Psi = [\psi_1, \ldots, \psi_M] \in \mathbb{R}^{M \times N}$. Let $\psi_j = \Sigma K_j$, where $\Sigma \in \mathbb{R}^{N \times N}$ and $\Phi \mathbf{1}_N = 0$ for admissible perturbation of $z(j)$ (i.e. $(z(j) + \epsilon\psi_j)\mathbf{1}_N = 1$ in (6.2)).

**Lemma 6.** *The Hessian $\mathcal{H}(Z^*, P^*, \Psi, T)$ in (6.8) loses rank when*

$$
det\big[\Sigma^\top \big(\Lambda_T(j) - \frac{1}{T} C_T(j)\big)\Sigma\big] = 0
$$

*for some $1 \leq j \leq M$, where $\Lambda_T(j)$ and $C_T(j)$ are as defined in (6.9).*

*Proof.* Motivated from [58]. $\mathcal{H}(Z^*, P^*, \Psi, T)$ is positive for all perturbation $\Psi$ if and only if its first part (see (6.8)) is positive. 'If' part is straightforward. For the 'only if' part we show that $\exists$ a $\Psi$ such that second term in $\mathcal{H}$ vanishes. Let $J = \{j_1, \ldots, j_r\}$ denote the superstates with distribution $z_{j_0}$. Select $\Psi$ such that $\psi_j = 0 \; \forall \; j \notin J$ and $\sum_{j \in J} \psi_j = 0$. For this perturbation the second term vanishes and $\mathcal{H}$ depends only on its first term. Thus, we establish the 'only if' part. Replacing the $\psi_j$ in (6.8) with $\Sigma K_j$ we obtain the condition stated in the Lemma. □

**Lemma 7.** *The condition $det\big[\Sigma^\top \big(\Lambda_T(j) - \frac{1}{T} C_T(j)\big)\Sigma\big] = 0$ for some $1 \leq j \leq M$ is attained at temperature value $T = T_{cr} := \lambda_{\max}\big(\mathcal{C}_T(j)\big)$, where $\mathcal{C}_T(j)$ is given in (6.11).*

*Proof.* Rank($\Sigma$)= $N-1$. Let $\Upsilon \in \mathbb{R}^{N \times N-1}$ such that Range($\Upsilon$) = Range($\Sigma$). Let $W_j \in \mathbb{R}^{N-1}$ so that $\Upsilon W_j = \Sigma K_j$. From Lemma 6 $\exists \; W_j : W_j \Upsilon^\top \big(\Lambda_T(j) - \frac{1}{T} C_T(j)\big) \Upsilon W_j = 0$. Let $H_0 = \Upsilon^\top \Lambda_T(j) \Upsilon$, $H_1 = \Upsilon^\top C_T(j) \Upsilon$. From Theorem 12.19 in [160] $\exists \; G \in \mathbb{R}^{N \times N-1}$ s.t. $G^\top H_0 G = I$ and $G^\top H_1 G = \mathcal{C}_T(j)$ where $G = L^{-\top} P$, $LL^\top = H_0$, $P^\top [L^{-1} H_1 L^{-\top}] P = \mathcal{C}_T(j)$. Let $W_j = G\omega_j$ for some $\omega_j \in \mathbb{R}^{N-1}$, $\Rightarrow \omega_j^\top (G^\top H_0 G - \frac{1}{T} G^\top H_1 G) \omega_j = 0 \Rightarrow \det\big(I - \frac{1}{T} \mathcal{C}_T(j)\big) = 0$ $\Rightarrow T_{cr} = \lambda_{\max}\big(\mathcal{C}_T(j)\big)$. Since $G^\top H_1 G = \mathcal{C}_T(j)$ we have that $\Theta = \Upsilon G$. □

## A.6 Sparse Linear Regression

### A.6.1 Algebraic Simplification

Below we simplify the objective function (7.5) in terms of $Q := [q_{ij}]$ where $q_{ij} := p_{ij}(v_{ij} = 1|x)$.

$$\begin{aligned}
\sum_{V \in \mathcal{V}} p(V|x) \|y - AVx\|_2^2 &= \sum_{V \in \mathcal{V}} p(V|x)(y - AVx)^T (y - AVx) \\
&= y^T y - 2 \sum_{k,j=1}^{N,m} y^T a_k x_j \sum_{V \in \mathcal{V}} p(V|x) v_{kj} + \sum_{k,j,l,s=1}^{N,N,m,m} a_k^T a_l x_j x_s \sum_{V \in \mathcal{V}} p(V|x) v_{kj} v_{ls} \\
&= y^T y - 2 \sum_{k,j=1}^{N,m} y^T a_k x_j q_{kj} + \sum_{k,j=1}^{N,m} q_{kj} x_j a_k^T \sum_{l,s=1}^{N,m} q_{ls} x_s a_l
\end{aligned}$$

$$+ \sum_{k,j=1}^{N,m} (q_{kj} - q_{kj}^2) a_k^T a_k x_k^2, \tag{A.18}$$

where $q_{kj} := \eta_{kj}(v_{kj} = 1|x)$. Equivalently, the objective in (A.18) can be re-written in matrix notation as

$$\|y - AQx\|_2^2 + [a_1^T a_1 \ \dots \ a_N^T a_N][Q \circ \bar{Q}][x \circ x], \tag{A.19}$$

where $Q = (q_{ij}) \in [0,1]^{N \times m}$, and $\circ$ denotes the Hadamard product. The feasibility constraints $\sum_i v_{ij} = 1$ on the columns of $V \in \mathcal{V}$ can ingeniously be represented in terms of the decision variables $q_{kj}$. Since $q_{kj} := \eta_{kj}(v_{kj} = 1|x)$, the constraint $\sum_{k=1}^N q_{kj} = 1$ is equivalent to the constraint $\sum_{k=1}^N v_{kj} = 1$. The optimization problem in (7.3) is thus re-written as

$$\begin{aligned} &\min_{\substack{x \in \mathbb{R}^M \\ Q \in \{0,1\}^{N \times m}}} \quad \|y - AQx\|_2^2 + [a_1^T a_1 \ \dots \ a_N^T a_N][Q \circ \bar{Q}][x \circ x] \\ &\text{subject to} \quad {1_N}^T Q = 1_m \end{aligned} . \tag{A.20}$$

### A.6.2 Convergence of the Algorithm 9

We have that

$$\begin{aligned} \frac{\partial F}{\partial Q} &= T \log \Big[\frac{Q\circ}{1-Q}\Big] \\ &\quad + 1_N \mu + 2A^T(y - AQx)x^T - \lambda(x^2)^T \circ (1 - 2Q), \end{aligned} \tag{A.21}$$

where $\circ$ denotes the element-wise operation. Let $\zeta := -\log \frac{Q\circ}{1-Q}$ which implies $Q = \frac{\exp(-\zeta)}{1+\exp(-\zeta)}$. Setting $\frac{\partial F}{\partial Q} = 0$ we obtain

$$-T\zeta^+ + 1_N \mu + R = 0 \tag{A.22}$$

$$\Rightarrow \zeta + = \frac{1}{T}\big[1_N \mu^T + R\big], \tag{A.23}$$

where $R = 2A^T(y - AQx)x^T - \lambda(x^2)^T \circ (1 - 2Q)$. Note that

$$\frac{\partial F}{\partial Q} = \frac{\partial F}{\partial \zeta} \circ \Big(\frac{\partial \zeta}{\partial Q}\Big) = \frac{\partial F}{\partial \zeta}\Big[-\frac{1\circ}{Q} - \frac{1\circ}{1-Q}\Big]$$

$$= -\frac{\partial F}{\partial \zeta}\Big[\exp(\zeta) + 2 + \exp(-\zeta)\Big] = -\frac{\partial F}{\partial \zeta}\Big[\exp(\zeta/2) + \exp(-\zeta/2)\Big]^2$$
$$\Rightarrow \frac{\partial F}{\partial Q} = -T\zeta + T\zeta^+ \Rightarrow \zeta^+ = \zeta - \frac{1}{T}\big(\exp(\zeta/2) + \exp(-\zeta/2)\big)^2 \frac{\partial F}{\partial \zeta} \quad \text{(A.24)}$$

Similarly, for $\mu$ using (7.10) we directly arrive at

$$\mu^+ = \mu - T\log(Q^T 1_N) \quad \text{(A.25)}$$
$$= \mu - T\Big[\frac{\log(Q^T 1_N)\circ}{(Q^T 1_N - 1_m)}\Big] \circ (Q^T 1_N) \quad \text{(A.26)}$$
$$= \mu - T\Big[\frac{\log(Q^T 1_N)\circ}{(Q^T 1_N - 1_m)}\Big] \circ \frac{\partial F}{\partial \mu} \quad \text{(A.27)}$$

Since $f(x) = \log(x)$ satisfies conditions for mean value theorem between $(Q^T 1_N, 1_m)$, there exists $\bar{k} \in (Q^T 1_N, 1_m)$ such that $\frac{1}{\bar{k}} = \frac{\log(Q^T 1_N)\circ}{(Q^T 1_N - 1_m)}$. Since $Q^T 1_N > 0$ and $1_m > 0$, $\bar{k} > 0$.

### A.6.3 Computation of the Hessian

$$F = \|y - AQx\|^2 + \underbrace{[a_1^T a_1 \ \ldots \ a_N^T a_N]}_{\triangleq \lambda^T}[Q \circ \bar{Q}]x^2 - c_0$$
$$+ \mu^T(Q^T 1_N - 1_m) + T 1_N^T [Q \circ \log Q + \bar{Q} \circ \log \bar{Q}] 1_m$$
$$\frac{\partial F}{\partial x} = -2(y - AQx)^T AQ + 2x^T \mathrm{diag}[\lambda^T (Q \circ \bar{Q})]$$
$$\Rightarrow x = \Big[Q^T A^T AQ + \underbrace{\mathrm{diag}\big[\lambda^T (Q \circ \bar{Q})\big]}_{\triangleq \Theta}\Big]^{-1} Q^T A^T y$$

Note that $\lambda^T[Q \circ \bar{Q}]x^2 = x^T \Theta x$. Consider $\|y - AQx\|^2 + x^T \Theta x$ with $x = \big(Q^T A^T AQ + \Theta\big)^{-1} Q^T A^T y \triangleq \Gamma^{-1} Q^T A^T y$

$$\begin{aligned}
\|y - AQx\|^2 + x^T \Theta x &= y^T y - 2y^T AQx + x^T (Q^T A^T AQ + \Theta)x \\
&= y^T y - 2y^T AQ\Gamma^{-1} Q^T A^T y + y^T AQ\Gamma^{-1}\Gamma\Gamma^{-1} Q^T A^T y \\
&= y^T [I - AQ\Gamma^{-1} Q^T A^T] y \\
&= y^T [I + AQ(\Gamma - Q^T A^T AQ)^{-1} Q^T A^T]^{-1} y \\
&= y^T [I + AQ\Theta^{-1} Q^T A^T]^{-1} y
\end{aligned}$$

Let $F := F_1 + F_2$, where $F_1 = \|y - AQx\|^2 + x^T\Theta x$

$$F_1(Q_\epsilon) = y^T\Delta_\epsilon^{-1}y \quad \text{where } \Delta_\epsilon = I + AQ_\epsilon\Theta_\epsilon^{-1}Q_\epsilon^TA^T,$$

$Q_\epsilon = Q + \epsilon\Psi$, where $\Psi$ is the permissible perturbation matrix such that its column sum is zero. Let $J_\epsilon^T = AQ_\epsilon$; $H^T = A\Psi \Rightarrow \frac{dJ_\epsilon^T}{d\epsilon} = H^T$ and $\Delta_\epsilon = I + J_\epsilon^T\Theta_\epsilon^{-1}J_\epsilon$.

$$\begin{aligned}\Rightarrow \frac{d\Delta_\epsilon}{d\epsilon} &= H^T\Theta_\epsilon^{-1}J_\epsilon + J_\epsilon^T\Theta_\epsilon^{-1}H - J_\epsilon^T\Theta_\epsilon^{-1}\frac{d\Theta_\epsilon}{d\epsilon}\Theta_\epsilon^{-1}J_\epsilon \\ &= H^T\Theta_\epsilon^{-1}J_\epsilon + J_\epsilon^T\Theta_\epsilon^{-1}H - J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon\end{aligned}$$

where $\Lambda_{\alpha\epsilon} = \frac{d\Theta}{d\epsilon} = \frac{d\ \mathrm{diag}\left(\lambda^T(Q_\epsilon\circ\bar{Q}_\epsilon)\right)}{d\epsilon} = \mathrm{diag}\big(\lambda^T(\Psi\circ(\bar{Q}_\epsilon - Q_\epsilon)\big)$ and $\frac{d^2\Lambda_{\alpha\epsilon}}{d\epsilon^2} = -2\mathrm{diag}\big[\lambda^T\big(\Psi\circ\Psi\big)\big] \triangleq -2D_\epsilon$.

$$\begin{aligned}\frac{d^2\Delta_\epsilon}{d\epsilon^2} &= -H^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon + H^T\Theta_\epsilon^{-1}H + H^T\Theta_\epsilon^{-1}H - J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}H \\ &\quad - H^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon + 2J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon \\ &\quad + 2J_\epsilon^T\Theta_\epsilon^{-1}D_\epsilon\Theta_\epsilon^{-1}J_\epsilon - J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}H\end{aligned}$$

Now,

$$\begin{aligned}\frac{dF_1(Q_\epsilon)}{d\epsilon} &= \frac{d}{d\epsilon}\big(y^T\Delta_\epsilon^{-1}y\big) = -y^T\Delta_\epsilon^{-1}\frac{d\Delta_\epsilon}{d\epsilon}\Delta_\epsilon^{-1}y \\ \frac{d^2F_1(Q_\epsilon)}{d\epsilon^2} &= 2y^T\Delta_\epsilon^{-1}\frac{d\Delta_\epsilon}{d\epsilon}\Delta_\epsilon^{-1}\frac{d\Delta_\epsilon}{d\epsilon}\Delta_\epsilon^{-1}y - y^T\Delta_\epsilon^{-1}\frac{d^2\Delta_\epsilon}{d\epsilon^2}\Delta_\epsilon^{-1}y \\ &= 2y^T\Delta_\epsilon^{-1}\Big[\frac{d\Delta_\epsilon}{d\epsilon}\Delta_\epsilon^{-1}\frac{d\Delta_\epsilon}{d\epsilon} - \frac{1}{2}\frac{d^2\Delta_\epsilon}{d\epsilon^2}\Big]\Delta_\epsilon^{-1}y\end{aligned}$$

$$\begin{aligned}\frac{d^2F_1}{d\epsilon^2} &= -2y^T\Delta_\epsilon^{-1}\Big[-2H^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon \\ &\quad + H^T\Theta_\epsilon^{-1}H + J_\epsilon^T\Theta_\epsilon^{-1}D_\epsilon\Theta_\epsilon^{-1}J_\epsilon + J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon\Big]\Delta_\epsilon^{-1}y \\ &\quad + 2y^T\Delta_\epsilon^{-1}\Big[H^T\Theta_\epsilon^{-1}J_\epsilon + J_\epsilon^T\Theta_\epsilon^{-1}H - J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon\Big] \\ &\quad \Delta_\epsilon^{-1}\Big[H^T\Theta_\epsilon^{-1}J_\epsilon + J_\epsilon^T\Theta_\epsilon^{-1}H - J_\epsilon^T\Theta_\epsilon^{-1}\Lambda_{\alpha\epsilon}\Theta_\epsilon^{-1}J_\epsilon\Big]\Delta_\epsilon^{-1}y\end{aligned}$$

Let $\Delta_\epsilon^{-1/2}H^T\Theta_\epsilon^{-1/2} = \Delta_\epsilon^{-1/2}A\Psi\Theta_\epsilon^{-1/2} \triangleq Z_\epsilon^T$;
$\Delta_\epsilon^{-1/2}J_\epsilon^T\Theta_\epsilon^{-1/2} = \Delta_\epsilon^{-1/2}AQ_\epsilon\Theta_\epsilon^{-1/2} \triangleq X^T$;

$\mu = \Delta_\epsilon^{-1/2} y$; $\bar{\Lambda} = \Theta^{-1/2}\Lambda_{\alpha\epsilon}\Theta^{-1/2}$; $\bar{D} = \Theta^{-1/2} D \Theta^{-1/2}$.

$$\Rightarrow \frac{d^2 F_1}{d\epsilon^2}\Big|_{\epsilon=0} = -2\mu^T\big[-2Z^T\bar{\Lambda}X + Z^TZ + X^T\bar{D}X + X^T\bar{\Lambda}^2X\big]\mu + 2\mu^T\big[Z^TX + X^TZ - X^T\bar{\Lambda}X\big]\big[Z^TX + X^TZ - X^T\bar{\Lambda}X\big]\mu$$

where $\mu = \Delta^{-1/2} y$; $\Delta = I + J^T\Theta^{-1}J = I + AQ\Theta^{-1}Q^TA^T$; $\Theta = \text{diag}(\lambda^T(Q \circ \bar{Q}))$; $\bar{\Lambda} = \Theta^{-1/2}\text{diag}(\lambda^T(\Psi \circ (\bar{Q} - Q))\Theta^{-1/2}$; $\bar{D} = 2\Theta^{-1/2}\text{diag}(\lambda^T(\Psi \circ \Psi))\Theta^{-1/2}$; $X = \Theta^{-1/2}Q^TA^T\Delta^{-1/2}$; $Z = \Theta^{-1/2}\Psi_T A^T\Delta^{-1/2}$. We know that $F_\epsilon = F_{1\epsilon} + F_{2\epsilon}$, where

$$F_{2\epsilon} = \frac{1}{\beta} I_N^T\Big[Q_\epsilon \circ \log Q_\epsilon + \bar{Q}_\epsilon \circ \log \bar{Q}_\epsilon\Big] I_m$$
$$\Rightarrow \frac{dF_{2\epsilon}}{d\epsilon} = \frac{1}{\beta} I_N^T\Big[\big(\Psi\big) \circ \big[\log Q_\epsilon - \log \bar{Q}_\epsilon\big]\Big] I_m$$
$$\Rightarrow \frac{d^2F_{2\epsilon}}{d\epsilon^2} = \frac{1}{\beta} I_N^T\Big[\big(\Psi\big) \circ \big(\frac{1}{Q_\epsilon} + \frac{1}{\bar{Q}_\epsilon}\big) \circ \big(\Psi\big)\Big] I_m$$

$$F_\epsilon = \underbrace{\|y - AQ_\epsilon x\|^2 + \lambda^T x^T \Theta_\epsilon x}_{F_{1\epsilon}} + \underbrace{\frac{1}{\beta} I_N^T\big[Q_\epsilon \circ \log Q_\epsilon + \bar{Q}_\epsilon \circ log\bar{Q}_\epsilon\big] I_m}_{F_{2\epsilon}}$$

$$\Rightarrow \frac{d^2F_\epsilon}{d\epsilon^2} = -2\mu^T[\bar{\Lambda}X - Z]^T[\bar{\Lambda}X - Z]\mu + 2\mu^T[Z^TX + X^TZ - X^T\Lambda X]^T[Z^TX + X^TZ - X^T\Lambda X]\mu - 2\mu^TX^T\bar{D}X\mu + \frac{1}{\beta} I_N^T\Big[(\Psi) \circ \big(\frac{1}{Q} + \frac{1}{\bar{Q}}\big) \circ \Psi\Big] I_m$$

$$\begin{aligned}\Rightarrow \frac{1}{2}\frac{d^2F}{d\epsilon^2}\Big|_{\epsilon=0} &= -\mu^TZ^TZ\mu + 2\mu^TZ^T\bar{\Lambda}X\mu - \mu^TX^T\bar{\Lambda}^2X\mu \\ &\quad + \mu^TX^TZZ^TX\mu + 2\mu^TX^TZX^TZ\mu \\ &\quad - 2\mu^TX^TZX^T\bar{\Lambda}X\mu + \mu^TZ^TXX^TZ\mu \\ &\quad - 2\mu^TZ^TXX^T\bar{\Lambda}X\mu + \mu^TX^T\bar{\Lambda}XX^T\bar{\Lambda}X\mu - \mu^TX^T\bar{D}X\mu \\ &\quad + \frac{1}{2\beta} I_N^T\Big[\big(\Psi\big) \circ \Big(\frac{1}{Q} + \frac{1}{\bar{Q}}\Big) \circ \big(\Psi\big)\Big] I_m \\ &= -\mu^TZ^TZ\mu + 2\mu^TZ^T\bar{\Lambda}X\mu - \mu^TX^T\bar{\Lambda}^2X\mu\end{aligned}$$

$$+\frac{1}{2\beta}I_N^T\Big[(\Psi)\circ\Big(\frac{1}{Q}+\frac{1}{\bar{Q}}\Big)\circ(\Psi)\Big]I_m$$
$$+\mu^T(X^TZ+Z^TX-X^T\bar{\Lambda}X)(X^TZ+Z^TX-X^T\bar{\Lambda}X)\mu$$

1. Let $w=\Delta^{-1/2}\mu$

$$\begin{aligned}\mu^TZ^TZ\mu &= \mu^T\Delta^{-1/2}A\Psi\Theta^{-1/2}\Theta^{-1/2}\Psi^TA^T\Delta^{-1/2}\mu\\ &= w^TA\sum_j\Psi_je_j^T\Theta^{-1}\sum_k e_k\Psi_k^TA^Tw\\ &= \sum_{j,k}w^TA\Psi_j\underbrace{e_j^T\Theta^{-1}e_k}_{\Theta_j^{-1}\delta_{jk}}\Psi_k^TA^Tw\\ &= \sum_j\Psi_j^TA^Tww^TA\Psi_j\Theta_j^{-1}\end{aligned}$$

2. Note that

$$\begin{aligned}\bar{\Lambda} &= \Theta^{-1/2}\mathrm{diag}\big(\lambda^T\big((\Psi)\circ(\bar{Q}-Q)\big)\big)\Theta^{-1/2}\\ &= \Theta^{-1/2}\sum_k\lambda^T\Lambda_{l_k}\Psi_ke_ke_k^T\Theta^{-1/2}\qquad \Lambda_{l_k}=\mathrm{diag}(\bar{q}_k-q_k)\\ &= \Theta^{-1/2}\sum_k L_k^T\Lambda_a\Psi_ke_ke_k^T\Theta^{-1/2},\qquad \text{where } L_k=\bar{q}_k-q_k\end{aligned}$$

Now consider $\mu^TZ^T\bar{\Lambda}X\mu$.

$$\begin{aligned}&\mu^TZ^T\bar{\Lambda}X\mu=\mu^T\Delta^{-1/2}A\Psi\Theta^{-1/2}\bar{\Lambda}X\mu\\ &= w^TA\sum_j\Psi_je_j^T\Theta^{-1/2}\Theta^{-1/2}\big(\sum_k\lambda^T\Lambda_{L_k}\Psi_ke_ke_k^T\big)\\ &\quad\Theta^{-1/2}\Theta^{-1/2}Q^TA^T\Delta^{-1/2}\mu\\ &= \sum_{j,k}w^TA\Psi_je_j^T\Theta^{-1}\lambda^T\Lambda_{L_k}\Psi_ke_ke_k^T\Theta^{-1}Q^TA^Tw\\ &= \sum_{j,k}\Psi_j^TA^Tw\lambda^T\Lambda_{L_k}\Psi_ke_j^T\Theta^{-1}e_ke_k^T\Theta^{-1}Q^TA^Tw\\ &= \frac{1}{2}\sum_j\Psi_j^T\Big[A^Tw\lambda^T\Lambda_{L_k}+\Lambda_{L_k}\lambda w^TA\Big]\Psi_j\Theta_j^{-1}e_j^T\Theta^{-1}Q^TA^Tw\end{aligned}$$

3. Consider $\mu^T X^T \bar{\Lambda} X \mu$

$$\begin{aligned}
\mu^T X^T \bar{\Lambda} X \mu &= \mu^T \Delta^{-1/2} A Q \Theta^{-1/2} \Theta^{-1/2} \big( \sum_k \lambda^T \Lambda_{L_k} \Psi_k e_k e_k^T \big) \Theta^{-1/2} \\
&\quad \Theta^{-1/2} \big( \sum_l \lambda^T \Lambda_{L_l} \Psi_l e_l e_l^T \big) \Theta^{-1/2} \Theta^{-1/2} Q^T A^T \Delta^{-1/2} \mu \\
&= \sum_{k,l} w^T A Q \Theta^{-1} \lambda^T \Lambda_{L_k} \Psi_k e_k e_k^T \Theta^{-1} \lambda^T \Lambda_{L_l} \Psi_l e_l e_l^T \Theta^{-1} Q^T A^T w \\
&= \sum_{k,l} \Psi_k \Lambda_{L_k} \lambda \lambda^T \Lambda_{L_l} \Psi_l w^T A Q \Theta^{-1} e_k \underbrace{e_k^T \Theta^{-1} e_l}_{\Theta_k^{-1} \delta_{kl}} e_l^T \Theta^{-1} Q^T A^T w \\
&= \sum_j \Psi_j \Lambda_{L_j} \lambda \lambda^T \Lambda_{L_j} \Psi_j \Theta_j^{-1} (w^T A Q \Theta^{-1} e_j)^2
\end{aligned}$$

4. Consider $\mu^T X^T \bar{D} X \mu$.

$$\begin{aligned}
&\mu^T X^T \bar{D} X \mu = \\
&\mu^T \Delta^{-1/2} A Q \Theta^{-1/2} \Theta^{-1/2} \underbrace{\mathrm{diag}(\lambda^T (\Psi \circ \Psi))}_{D} \Theta^{-1/2} \Theta^{-1/2} Q^T A^T \Delta^{-1/2} \mu \\
&= w^T A Q \Theta^{-1} D \Theta^{-1} Q^T A^T w \\
&\text{where } D = \sum_k \lambda^T (\Psi_k)^2 e_k e_k^T = \sum_k \Psi_k^T \Lambda_a \Psi_k e_k e_k^T \\
&= w^T A Q \Theta^{-1} \sum_k \Psi_k \Lambda_a \Psi_k e_k e_k^T \Theta^{-1} Q^T A^T w \\
&= \sum_k \Psi_k^T \Lambda_a \Psi_k (w^T A Q \Theta^{-1} e_k)^2
\end{aligned}$$

5. Consider $\mu^T X^T Z Z^T X \mu$.

$$\begin{aligned}
\mu^T X^T Z Z^T X \mu &= \mu^T \Delta^{-1/2} A Q \Theta^{-1/2} \Theta^{-1/2} \Psi^T A^T \Delta^{-1/2} \\
&\quad \Delta^{-1/2} A \Psi \Theta^{-1/2} \Theta^{-1/2} Q^T A^T \Delta^{-1/2} \mu \\
&= \sum_{j,k} w^T A Q \Theta^{-1} e_j \Psi_j A^T \Delta^{-1} A \Psi_k e_k^T \Theta^{-1} Q^T A^T w \\
&= \sum_{j,k} \Psi_j A^T \Delta^{-1} A \Psi_k (w^T A Q \Theta^{-1} e_j)(w^T A Q \Theta^{-1} e_k)
\end{aligned}$$

6. Consider $Z^T X \mu$.

$$Z^T X \mu = \Delta^{-1/2} A \Psi \Theta^{-1/2} \Theta^{-1/2} Q^T A^T \Delta^{-1/2} \mu$$

$$
\begin{aligned}
&= \Delta^{-1/2} A \Psi \Theta^{-1} Q^T A^T w \\
&= \sum_j \Delta^{-1/2} A \Psi_j e_j^T \Theta^{-1} Q^T A^T w \\
&= \sum_j \big(e_j^T \Theta^{-1} Q^T A^T w\big) \Delta^{-1/2} A \Psi_j
\end{aligned}
$$

7. Consider $X^T Z \mu$.

$$
\begin{aligned}
X^T Z \mu &= \Delta^{-1/2} A Q \Theta^{-1/2} \Theta^{-1/2} \Psi^T A^T \Delta^{-1/2} \mu \\
&= \sum_j \Delta^{-1/2} A Q \Theta^{-1} e_j \Psi_j A^T \Delta^{-1/2} \mu \\
&= \sum_j \Delta^{-1/2} A Q \Theta^{-1} e_j w^T A \Psi_j
\end{aligned}
$$

8. Consider $X^T \bar{\Lambda} X \mu$.

$$
\begin{aligned}
X^T \bar{\Lambda} X \mu = \Delta^{-1/2} A Q \Theta^{-1/2} & \\
\Theta^{-1/2} \sum_j L_j^T \Lambda_a \Psi_j e_j e_j^T \Theta^{-1/2} \Theta^{-1/2} Q^T A^T \Delta^{-1/2} \mu & \\
= \sum_j \big(e_j^T \Theta^{-1} Q^T A^T w\big) \Delta^{-1/2} A Q \Theta^{-1} e_j L_j^T \Lambda_a \Psi_j &
\end{aligned}
$$

Thus adding $6 + 7 + 8$ we obtain,

$$
\begin{aligned}
= \Delta^{-1/2} A \sum_j \Big[ & \big(e_j^T \Theta^{-1} Q^T A^T w\big) I + Q \Theta^{-1} e_j w^T A \\
& + \big(e_j^T \Theta^{-1} Q^T A^T w\big) Q \Theta^{-1} e_j L_j^T \Lambda_a \Big] \Psi_j
\end{aligned}
$$

Let $h_j = w^T A Q \Theta^{-1} e_j$, $\Theta_j^{-1} = e_j^T \Theta^{-1} e_j$. Then,

$$
\begin{aligned}
\frac{1}{2} \frac{d^2 F}{d\epsilon^2}\Big|_{\epsilon=0} = \sum_j \Psi_j \Big[ & - \Theta_j^{-1} A^T w w^T A + 2 \Theta_j^{-1} h_j A^T w L_j \Lambda_a \\
& - \Theta_j^{-1} h_j^2 \Lambda_a L_j L_j^T \Lambda_a - h_j^2 \Lambda_a + \frac{1}{2\beta} \Lambda_{b_j} \Big] \Psi_j \\
& + \Big\| \Delta^{-1/2} A \Big[ \sum_j h_j I + Q \Theta^{-1} e_j w^T A + h_j Q \Theta^{-1} e_j L_j^T \Lambda_a \Big] \Psi_j \Big\|^2
\end{aligned}
$$

Note that the $Q$ matrix for very small $\beta$ has identical columns. As annealing progresses this matrix has multiple vectors that are identical. Let $\{q_{\eta_1}, \ldots, q_{\eta_r}\}$ be distinct vectors such that $\eta_1 + \eta_2 + \ldots + \eta_r = m$. Let

$$\tilde{I}_l = \{j \text{ s.t. } q_j = q_{\eta_l},\ 1 \leq j \leq m\} \quad \text{for} \quad 1 \leq l \leq r$$

Note that for each $l$ and $j \in \tilde{I}_l$ we have that

$$\begin{aligned}
&\Theta_j^{-1} = e_j^T \Theta^{-1} e_j = e_j^T \mathrm{diag}(\lambda^T (Q \circ \bar{Q})) e_j = q_j^T \Lambda_a \bar{q}_j = q_{\eta_l} \Lambda_a \bar{q}_{\eta_l} = \Theta_{\eta_l}^{-1} \\
&Q\Theta^{-1} e_j = Q\Theta_{\eta_l}^{-1} e_j = q_{\eta_l} \Theta_{\eta_l}^{-1} = Q\Theta^{-1} e_{\eta_l} \\
&h_j = w^T A Q \Theta^{-1} e_j = w^T A Q \Theta^{-1} e_{\eta_l} = h_{\eta_l} \\
&L_j = q_j - \bar{q}_j = q_{\eta_l} - \bar{q}_{\eta_l} = L_{\eta_l} \\
&\Lambda_{b_j} = \mathrm{diag}\big[\frac{1}{q_j} + \frac{1}{\bar{q}_j}\big] = \mathrm{diag}\big[\frac{1}{q_{\eta_l}} + \frac{1}{\bar{q}_{\eta_l}}\big] = \Lambda_{b_{\eta_l}}
\end{aligned}$$

Thus, we have that

$$\begin{aligned}
\frac{d^2 F}{d\epsilon^2}\Big|_{\epsilon=0} &= \sum_j \Psi_j \big[ -\Theta_j^{-1} A^T w w^T A + 2\Theta_j^{-1} h_j A^T w L_j^T \Lambda_a \\
&\quad - \Theta_j h_j^2 \Lambda_a L_j L_j^T \Lambda_a - h_j^2 \Lambda_a + \frac{1}{2\beta} \Lambda_{b_j} \big] \Psi_j \\
&\quad + \| \Delta^{-1/2} A \big[ \sum_j h_j I + Q\Theta^{-1} e_j w^T A + h_j Q \Theta^{-1} e_j L_j^T \Lambda_a \big] \Psi_j \|^2 \\
&= \sum_l \sum_{j \in \tilde{I}_l} \Psi_j \big[ -\Theta_{\eta_l}^{-1} A^T w w^T A + 2\Theta_{\eta_l}^{-1} h_{\eta_l} A^T w L_{\eta_l}^T \Lambda_a \\
&\quad - \Theta_{\eta_l}^{-1} h_{\eta_l}^2 \Lambda_a L_{\eta_l} L_{\eta_l}^T \Lambda_a - h_{\eta_l}^2 \Lambda_a + \frac{1}{2\beta} \Lambda_{b_{\eta_l}} \big] \Psi_j \\
&\quad + \| \Delta^{-1/2} A \big[ \sum_l \sum_{j \in \tilde{I}_l} h_{\eta_l} I + Q\Theta^{-1} e_{\eta_l} w^T A + h_{\eta_l} Q \Theta^{-1} e_{\eta_l} L_{\eta_l}^T \Lambda_a \big] \Psi_j \|^2
\end{aligned}$$

Let $-\Gamma_l \triangleq -\Theta_{\eta_l}^{-1} A^T w w^T A + \Theta_{\eta_l}^{-1} h_{\eta_l} (A^T w L_{\eta_l}^T \Lambda_a + \Lambda_a L_{\eta_l} w^T A) - \Theta_{\eta_l}^{-1} h_{\eta_l}^2 \Lambda_a L_{\eta_l} L_{\eta_l}^T \Lambda_a - h_{\eta_l}^2 \Lambda_a$ and $E_l \triangleq \Delta^{-1/2} A \big[ h_{\eta_l} I + Q\Theta^{-1} e_{\eta_l} w^T A + h_{\eta_l} Q \Theta^{-1} e_{\eta_l} L_{\eta_l}^T \Lambda_a \big]$. Thus we have that

$$\frac{d^2 F}{d\epsilon^2}\Big|_{\epsilon=0} = \sum_l \sum_{j \in \tilde{I}_l} \Psi_j \Big[ \frac{1}{2\beta} \Lambda_{b_{\eta_l}} - \Gamma_l \Big] \Psi_j + \Big\| \sum_l E_l C \sum_{j \in \tilde{I}_l} \Phi_j \Big\|^2$$

### A.6.4 Proof of Theorem 15

Note that we are interested in all $\Phi_j$ such that $\frac{\Phi_j}{\|\Phi_j\|} \neq 1_N$; since then $Q_\epsilon = Q + \epsilon\Psi = Q$. Since $C1_N = 0$; rank(C)=N-1. Let $\bar{C} \in \mathbb{R}^{N\times N-1}$ be such that Range($\bar{C}$)=Range($C$). Let $\bar{\Phi}_j \in \mathbb{R}^{N-1}$ be such that $\bar{C}\bar{\Phi}_j = \Psi_j$. Then

$$\frac{d^2F}{d\epsilon^2}\Big|_{\epsilon=0} = \underbrace{\sum_l \sum_{j\in\tilde{I}_l} \bar{\Phi}_j \bar{C}^T \Big[\frac{1}{2\beta}\Lambda_{b_{\eta_l}} - \Gamma_l\Big]\bar{C}\bar{\Phi}_j}_{\text{I}} + \underbrace{\Big\|\sum_l E_l \bar{C} \sum_{j\in\tilde{I}_l} \bar{\Phi}_j\Big\|^2}_{\text{II}}$$

We now claim that $\frac{d^2F}{d\epsilon^2}\Big|_{\epsilon=0} > 0$ if and only if I$> 0$.
Note that the "if" direction is obvious since the second term II is non-negative. For the "only if" direction we will show that when I$= 0$, we can determine $\{\bar{\Phi}_j\}$, not all zero, such that II$= 0$. Note that I$= 0 \Rightarrow \beta$ is such that $\Big|\bar{C}^T\frac{\Lambda_{b_{\eta_l}}}{2\beta}\bar{C} - \bar{C}^T\Gamma_l\bar{C}\Big| = 0$; Let $l$ be chosen such that the above condition happens for smallest value of $\beta$. Let $V_l$ be such that $V_l^T\Big(\bar{C}^T\frac{\Lambda_{b_{\eta_l}}}{2\beta}\bar{C} - \bar{C}^T\Gamma_l\bar{C}\Big)V_l = 0$. Let

$$\Phi_j = \begin{cases} 0, & \text{if } j \notin \tilde{I}_l \\ \alpha_j V_l, & \text{if } j \in \tilde{I}_l \text{ s.t. } \sum_j \alpha_j = 0, \alpha_1 \neq 0 \end{cases}$$

In this case II$= \Big\|\sum_l E_l\bar{C}\sum_{j\in\tilde{I}_l}\alpha_j V_l\Big\|^2 = \Big\|\sum_l E_l\bar{C}V_l\sum_{j\in\tilde{I}_l}\alpha_j\Big\|^2 = 0$.

$\frac{d^2F}{d\epsilon^2} = 0$ when $\beta$, $V_l$ satisfies $V_l^T\bar{C}^T\Big[\frac{1}{2\beta}\Lambda_{b_{\eta_l}} - \Gamma_l\Big]\bar{C}V_l = 0$; this is possible only when

$$\lambda_{\min}\Big[\frac{1}{2\beta}\bar{C}^T\Lambda_{b_{\eta_l}}\bar{C} - \bar{C}^T\Gamma_l\bar{C}\Big] = 0$$

Note that $H_0 \triangleq \bar{C}^T\Lambda_{b_{\eta_l}}\bar{C} > 0$ and $H_1 \triangleq \bar{C}^T\Gamma_l\bar{C} = H_1^T$. From Theorem 12.19 (Simultaneous Reduction to Diagonal Form) in Laub's book, we have - there exists a non-singular matrix $G$ such that $G^TH_0G = I$; $G^TH_1G = D_\Delta$ where $D_\Delta$ is a diagonal matrix. Here $G = L_\Delta^{-T}P_\Delta$ where $H_0 = L_\Delta L_\Delta^T$ (Choleski factorization of $H_0$) and $P_\Delta$ is such that $P_\Delta^T[L_\Delta^{-1}H_1L_\Delta^{-T}]P_\Delta = D_\Delta$ is a diagonal matrix. Such a $P_\Delta$ always exists since $H = H^T$ [SVD of $L_\Delta^{-1}H_1L_\Delta^{-T}$]. Let $V_l = G\bar{V}_l$

$$\therefore V_l^T\big[\frac{1}{2\beta}\bar{C}^T\Lambda_{b_{\eta_l}}\bar{C} - \bar{C}^T\Gamma_l\bar{C}\big]V_l \geq 0$$

$$\Rightarrow V_l^T\big[\frac{1}{2\beta}G^T H_0 G - G^T H_1 G\big]V_l \geq 0$$

$$\Rightarrow V_l^T\big[\frac{1}{2\beta}I - D_\Delta\big]V_l \geq 0$$

$$\Rightarrow \frac{1}{2\beta}I - D_{\Delta_{\min}} \geq 0$$

$$\Rightarrow \beta \leq \frac{1}{2D_{\Delta_{\min}}}$$

# REFERENCES


[1] Y. Xu, S. M. Salapaka, and C. L. Beck, "Aggregation of graph models and markov chains by deterministic annealing," *IEEE Transactions on Automatic Control*, vol. 59, no. 10, pp. 2807–2812, 2014.

[2] N. V. Kale and S. M. Salapaka, "Maximum entropy principle-based algorithm for simultaneous resource location and multihop routing in multiagent networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 4, pp. 591–602, 2012.

[3] A. Srivastava and S. M. Salapaka, "Combined resource allocation and route optimization in multiagent networks: A scalable approach," in *American Control Conference (ACC), 2017*. IEEE, 2017, pp. 3956–3961.

[4] P. Fränti and O. Virmajoki, "Iterative shrinking method for clustering problems," *Pattern Recognition*, vol. 39, no. 5, pp. 761–765, 2006. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2005.09.012

[5] E. T. Jaynes, "Information theory and statistical mechanics," *Physical review*, vol. 106, no. 4, p. 620, 1957.

[6] K. Rose, "Deterministic annealing, clustering, and optimization," Ph.D. dissertation, California Institute of Technology, 1991.

[7] P. Sharma, S. Salapaka, and C. Beck, "A scalable approach to combinatorial library design for drug discovery," *Journal of chemical information and modeling*, vol. 48, no. 1, pp. 27–41, 2008.

[8] J.-G. Yu, J. Zhao, J. Tian, and Y. Tan, "Maximal entropy random walk for region-based visual saliency," *IEEE transactions on cybernetics*, vol. 44, no. 9, pp. 1661–1672, 2013.

[9] L. Chen, T. Zhou, and Y. Tang, "Protein structure alignment by deterministic annealing," *Bioinformatics*, vol. 21, no. 1, pp. 51–62, 2005.

[10] E. T. Jaynes, *Probability theory: The logic of science*. Cambridge university press, 2003.

[11] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic programming*. Athena Scientific Belmont, MA, 1996, vol. 5.

[12] A. Hordijk and L. Kallenberg, "Linear programming and markov decision chains," *Management Science*, vol. 25, no. 4, pp. 352–362, 1979.

[13] Y. Abbasi-Yadkori, P. L. Bartlett, and A. Malek, "Linear programming for large-scale markov decision problems," in *JMLR Workshop and Conference Proceedings*, no. 32. MIT Press, 2014, pp. 496–504.

[14] D. P. De Farias and B. Van Roy, "The linear programming approach to approximate dynamic programming," *Operations research*, vol. 51, no. 6, pp. 850–865, 2003.

[15] A. Tewari and P. L. Bartlett, "Bounded parameter markov decision processes with average reward criterion," in *International Conference on Computational Learning Theory*. Springer, 2007, pp. 263–277.

[16] A. Gopalan and S. Mannor, "Thompson sampling for learning parameterized markov decision processes," in *Conference on Learning Theory*. PMLR, 2015, pp. 861–898.

[17] S. R. Chowdhury, A. Gopalan, and O.-A. Maillard, "Reinforcement learning in parametric mdps with exponential families," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 1855–1863.

[18] L. Xia and Q.-S. Jia, "Parameterized markov decision process and its application to service rate control," *Automatica*, vol. 54, pp. 29–35, 2015.

[19] M. Hausknecht and P. Stone, "Deep reinforcement learning in parameterized action space," *arXiv preprint arXiv:1511.04143*, 2015.

[20] W. Masson, P. Ranchod, and G. Konidaris, "Reinforcement learning with parameterized actions," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[21] E. Wei, D. Wicke, and S. Luke, "Hierarchical approaches for reinforcement learning in parameterized action space," in *2018 AAAI Spring Symposium Series*, 2018.

[22] J. Xiong, Q. Wang, Z. Yang, P. Sun, L. Han, Y. Zheng, H. Fu, T. Zhang, J. Liu, and H. Liu, "Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space," *arXiv preprint arXiv:1810.06394*, 2018.

[23] V. Narayanan and S. Jagannathan, "Event-triggered distributed control of nonlinear interconnected systems using online reinforcement learning with exploration," *IEEE transactions on cybernetics*, vol. 48, no. 9, pp. 2510–2519, 2017.

[24] E. Çilden and F. Polat, "Toward generalization of automated temporal abstraction to partially observable reinforcement learning," *IEEE transactions on cybernetics*, vol. 45, no. 8, pp. 1414–1425, 2014.

[25] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[26] H. V. Hasselt, "Double q-learning," in *Advances in Neural Information Processing Systems*, 2010, pp. 2613–2621.

[27] R. Fox, A. Pakman, and N. Tishby, "Taming the noise in reinforcement learning via soft updates," *arXiv preprint arXiv:1512.08562*, 2015.

[28] J. Grau-Moya, F. Leibfried, and P. Vrancx, "Soft q-learning with mutual-information regularization," 2018.

[29] J. Peters, K. Mulling, and Y. Altun, "Relative entropy policy search," in *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[30] G. Neu, A. Jonsson, and V. Gómez, "A unified view of entropy-regularized markov decision processes," *arXiv preprint arXiv:1705.07798*, 2017.

[31] K. Asadi and M. L. Littman, "An alternative softmax operator for reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 243–252.

[32] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans, "Bridging the gap between value and policy based reinforcement learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 2775–2785.

[33] B. Dai, A. Shaw, L. Li, L. Xiao, N. He, Z. Liu, J. Chen, and L. Song, "Sbeed: Convergent reinforcement learning with nonlinear function approximation," *arXiv preprint arXiv:1712.10285*, 2017.

[34] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, 2015, pp. 1889–1897.

[35] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.

[36] W. Shi, S. Song, and C. Wu, "Soft policy gradient method for maximum entropy deep reinforcement learning," *arXiv preprint arXiv:1909.03198*, 2019.

[37] G. Xiang and J. Su, "Task-oriented deep reinforcement learning for robotic skill acquisition and control," *IEEE transactions on cybernetics*, 2019.

[38] G. Cornuéjols, G. Nemhauser, and L. Wolsey, "The uncapicitated facility location problem," Cornell University Operations Research and Industrial Engineering, Tech. Rep., 1983.

[39] M. Barthélemy, "Spatial networks," *Physics Reports*, vol. 499, no. 1-3, pp. 1–101, 2011.

[40] D. Li, B. Fu, Y. Wang, G. Lu, Y. Berezin, H. E. Stanley, and S. Havlin, "Percolation transition in dynamical traffic network with evolving critical bottlenecks," *Proceedings of the National Academy of Sciences*, vol. 112, no. 3, pp. 669–672, 2015.

[41] D. Vaknin, M. M. Danziger, and S. Havlin, "Spreading of localized attacks in spatial multiplex networks," *New Journal of Physics*, vol. 19, no. 7, p. 073037, 2017.

[42] Y. Berezin, A. Bashan, M. M. Danziger, D. Li, and S. Havlin, "Localized attacks on spatially embedded networks with dependencies," *Scientific reports*, vol. 5, p. 8934, 2015.

[43] M. Barthelemy, *Morphogenesis of spatial networks*. Springer, 2018.

[44] M. Watson, *Supply chain network design: applying optimization and analytics to the global supply chain*. Pearson Education, 2013.

[45] D. Hwang, P. Jaillet, and Z. Zhou, "Distributed multi-depot routing without communications," *ArXiv*, vol. abs/1412.2123, 2014.

[46] I. Bistritz, A. Ward, Z. Zhou, and N. Bambos, "Smart greedy distributed allocation in microgrids," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[47] R. Takei, R. Tsai, Z. Zhou, and Y. Landa, "An efficient algorithm for a visibility-based surveillance-evasion game," TEXAS UNIV AT AUSTIN INST FOR COMPUTATIONAL ENGINEERING AND SCIENCES, Tech. Rep., 2012.

[48] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[49] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[50] D. R. d. M. Faria, R. A. d. Santos, K. M. G. Santos, and D. H. Spadoti, "A system to improve the management of 5g and iot networks by determining the mobile position," *Journal of Microwaves, Optoelectronics and Electromagnetic Applications*, vol. 18, no. 2, pp. 293–305, 2019.

[51] A. Moustakas, P. Mertikopoulos, Z. Zhou, and N. Bambos, "Least action routing: Identifying the optimal path in a wireless relay network," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2017, pp. 1–5.

[52] F. Kuhn, R. Wattenhofer, and A. Zollinger, "An algorithmic approach to geographic routing in ad hoc and sensor networks," *IEEE/ACM Transactions On Networking*, vol. 16, no. 1, pp. 51–62, 2008.

[53] L. Zhou, X. Wang, L. Ni, and Y. Lin, "Location-routing problem with simultaneous home delivery and customer's pickup for city distribution of online shopping purchases," *Sustainability*, vol. 8, no. 8, p. 828, 2016.

[54] M. Mahajan, P. Nimbhorkar, and K. Varadarajan, "The planar k-means problem is np-hard," in *International Workshop on Algorithms and Computation*. Springer, 2009, pp. 274–285.

[55] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A search meets graph theory," in *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2005, pp. 156–165.

[56] Z. Drezner and H. W. Hamacher, *Facility location: applications and theory*. Springer Science & Business Media, 2001.

[57] M. Baranwal, "Entropy-based framework for combinatorial optimization problems and enabling the grid of the future," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2018.

[58] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2210–2239, 1998.

[59] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[60] P. Sharma, S. M. Salapaka, and C. L. Beck, "Entropy-based framework for dynamic coverage and clustering problems," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 135–150, 2012.

[61] Y. Wang, X. Li, and R. Ruiz, "A fast algorithm for finding the bi-objective shortest path in complicated networks," in *2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD))*. IEEE, 2018, pp. 104–109.

[62] S. Okada and T. Soper, "A shortest path problem on a network with fuzzy arc lengths," *Fuzzy sets and systems*, vol. 109, no. 1, pp. 129–140, 2000.

[63] L. Guo, Y. Deng, K. Liao, Q. He, T. Sellis, and Z. Hu, "A fast algorithm for optimally finding partially disjoint shortest paths." in *IJCAI*, 2018, pp. 1456–1462.

[64] R. K. Ahuja, *Network flows: theory, algorithms, and applications*. Pearson Education, 2017.

[65] A. Gersho and R. M. Gray, *Vector quantization and signal compression*. Springer Science & Business Media, 2012, vol. 159.

[66] C. E. Shannon and W. Weaver, "The mathematical theory of communication," *Urbana: University of Illinois Press*, 1949.

[67] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[68] R. Sepulchre, M. Jankovic, and P. V. Kokotovic, *Constructive nonlinear control*. Springer Science & Business Media, 2012.

[69] E. D. Sontag, "A lyapunov-like characterization of asymptotic controllability," *SIAM Journal on Control and Optimization*, vol. 21, no. 3, pp. 462–471, 1983.

[70] E. D. Sontag, "A 'universal'construction of artstein's theorem on nonlinear stabilization," *Systems & control letters*, vol. 13, no. 2, pp. 117–123, 1989.

[71] E. A. Feinberg and A. Shwartz, *Handbook of Markov decision processes: methods and applications*. Springer Science & Business Media, 2012, vol. 40.

[72] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[73] A. Aguilar-Garcia, S. Fortes, M. Molina-García, J. Calle-Sánchez, J. I. Alonso, A. Garrido, A. Fernández-Durán, and R. Barco, "Location-aware self-organizing methods in femtocell networks," *Computer Networks*, vol. 93, pp. 125–140, 2015.

[74] U. Siddique, H. Tabassum, E. Hossain, and D. I. Kim, "Wireless backhauling of 5g small cells: Challenges and solution approaches," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 22–31, 2015.

[75] G. Manganini, M. Pirotta, M. Restelli, L. Piroddi, and M. Prandini, "Policy search for the optimal control of markov decision processes: A novel particle-based iterative scheme," *IEEE transactions on cybernetics*, vol. 46, no. 11, pp. 2643–2655, 2015.

[76] A. Srivastava and S. M. Salapaka, "Simultaneous facility location and path optimization in static and dynamic networks," *IEEE Transactions on Control of Network Systems*, pp. 1–1, 2020.

[77] A. A. Abin, "Querying beneficial constraints before clustering using facility location analysis," *IEEE transactions on cybernetics*, vol. 48, no. 1, pp. 312–323, 2016.

[78] D. Huang, C.-D. Wang, and J.-H. Lai, "Locally weighted ensemble clustering," *IEEE transactions on cybernetics*, vol. 48, no. 5, pp. 1460–1473, 2017.

[79] F. Biondi, A. Legay, B. F. Nielsen, and A. Wkasowski, "Maximizing entropy over markov processes," *Journal of Logical and Algebraic Methods in Programming*, vol. 83, no. 5-6, pp. 384–399, 2014.

[80] Y. Savas, M. Ornik, M. Cubuktepe, and U. Topcu, "Entropy maximization for constrained markov decision processes," in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2018, pp. 911–918.

[81] S. Ross, J. Pineau, B. Chaib-draa, and P. Kreitmann, "A bayesian approach for learning and planning in partially observable markov decision processes," *Journal of Machine Learning Research*, vol. 12, no. May, pp. 1729–1770, 2011.

[82] L. P. Hansen, T. J. Sargent, G. Turmuhambetova, and N. Williams, "Robust control and model misspecification," *Journal of Economic Theory*, vol. 128, no. 1, pp. 45–90, 2006.

[83] Z. Zhou, M. Bloem, and N. Bambos, "Infinite time horizon maximum causal entropy inverse reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 2787–2802, 2017.

[84] K. Rawlik, M. Toussaint, and S. Vijayakumar, "Approximate inference and stochastic optimal control," *arXiv preprint arXiv:1009.3958*, 2010.

[85] M. Ghavamzadeh, H. J. Kappen, M. G. Azar, and R. Munos, "Speedy q-learning," in *Advances in neural information processing systems*, 2011, pp. 2411–2419.

[86] M. G. Bellemare, G. Ostrovski, A. Guez, P. S. Thomas, and R. Munos, "Increasing the action gap: New operators for reinforcement learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[87] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li, "Drn: A deep reinforcement learning framework for news recommendation," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 167–176.

[88] W. B. Knox and P. Stone, "Reinforcement learning from human reward: Discounting in episodic tasks," in *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2012, pp. 878–885.

[89] A. L. Thomaz and C. Breazeal, "Teachable robots: Understanding human teaching behavior to build more effective robot learners," *Artificial Intelligence*, vol. 172, no. 6-7, pp. 716–737, 2008.

[90] P. Sharma, S. M. Salapaka, and C. L. Beck, "Entropy-based framework for dynamic coverage and clustering problems," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 135–150, 2011.

[91] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski et al., "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[92] B. Andreopoulos, A. An, X. Wang, and M. Schroeder, "A roadmap of clustering algorithms: finding a match for a biomedical application," *Briefings in Bioinformatics*, vol. 10, no. 3, pp. 297–314, 2009.

[93] A. G. Di Nuovo and V. Catania, "An evolutionary fuzzy c-means approach for clustering of bio-informatics databases," in *Fuzzy Systems, 2008. FUZZ-IEEE 2008.(IEEE World Congress on Computational Intelligence). IEEE International Conference on*. IEEE, 2008, pp. 2077–2082.

[94] D. T. Larose and C. D. Larose, *Discovering knowledge in data: an introduction to data mining*. John Wiley & Sons, 2014.

[95] F. Yuan, Y. Zhan, and Y. Wang, "Data density correlation degree clustering method for data aggregation in wsn," *IEEE Sensors Journal*, vol. 14, no. 4, pp. 1089–1098, 2014.

[96] C.-W. Huang, K.-P. Lin, M.-C. Wu, K.-C. Hung, G.-S. Liu, and C.-H. Jen, "Intuitionistic fuzzy $c$-means clustering algorithm with neighborhood attraction in segmenting medical image," *Soft Computing*, vol. 19, no. 2, pp. 459–470, 2015.

[97] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[98] H.-S. Park and C.-H. Jun, "A simple and fast algorithm for k-medoids clustering," *Expert systems with applications*, vol. 36, no. 2, pp. 3336–3341, 2009.

[99] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[100] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001.

[101] Y. Feng and G. Hamerly, "Pg-means: learning the number of clusters in data," in *Advances in neural information processing systems*, 2007, pp. 393–400.

[102] D. Pelleg, A. W. Moore et al., "X-means: Extending k-means with efficient estimation of the number of clusters." in *Icml*, vol. 1, 2000, pp. 727–734.

[103] R. E. Kass and L. Wasserman, "A reference bayesian test for nested hypotheses and its relationship to the schwarz criterion," *Journal of the american statistical association*, vol. 90, no. 431, pp. 928–934, 1995.

[104] H. Akaike, "Akaike's information criterion," in *International encyclopedia of statistical science*. Springer, 2011, pp. 25–25.

[105] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[106] C. A. Sugar and G. M. James, "Finding the number of clusters in a dataset: An information-theoretic approach," *Journal of the American Statistical Association*, vol. 98, no. 463, pp. 750–763, 2003.

[107] G. Hamerly and C. Elkan, "Learning the k in k-means," in *Advances in neural information processing systems*, 2004, pp. 281–288.

[108] M. A. Stephens, "Edf statistics for goodness of fit and some comparisons," *Journal of the American statistical Association*, vol. 69, no. 347, pp. 730–737, 1974.

[109] A. Kalogeratos and A. Likas, "Dip-means: an incremental clustering method for estimating the number of clusters," in *Advances in neural information processing systems*, 2012, pp. 2393–2401.

[110] J. A. Hartigan and P. M. Hartigan, "The dip test of unimodality," *The Annals of Statistics*, pp. 70–84, 1985.

[111] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 551–556.

[112] K. Kurihara and M. Welling, "Bayesian k-means as a "maximization-expectation" algorithm," *Neural computation*, vol. 21, no. 4, pp. 1145–1172, 2009.

[113] P. Sand and A. W. Moore, "Repairing faulty mixture models using density estimation," in *ICML*. Citeseer, 2001, pp. 457–464.

[114] T. Lange, M. L. Braun, V. Roth, and J. M. Buhmann, "Stability-based model selection," in *Advances in neural information processing systems*, 2003, pp. 633–642.

[115] R. Tibshirani and G. Walther, "Cluster validation by prediction strength," *Journal of Computational and Graphical Statistics*, vol. 14, no. 3, pp. 511–528, 2005.

[116] E. Levine and E. Domany, "Resampling method for unsupervised estimation of cluster validity," *Neural computation*, vol. 13, no. 11, pp. 2573–2593, 2001.

[117] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[118] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," pp. 1–17, 2018. [Online]. Available: http://cs.uef.fi/sipu/datasets/

[119] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, 2002, pp. 849–856.

[120] P. A. Gagniuc, *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017.

[121] R. Srikant, *The mathematics of Internet congestion control*. Springer Science & Business Media, 2004.

[122] C. J. Quinn, T. P. Coleman, N. Kiyavash, and N. G. Hatsopoulos, "Estimating the directed information to infer causal relationships in ensemble neural spike train recordings," *Journal of computational neuroscience*, vol. 30, no. 1, pp. 17–44, 2011.

[123] Q. Zhang and G. Yin, "Nearly-optimal asset allocation in hybrid stock investment models," *Journal of Optimization theory and Applications*, vol. 121, no. 2, pp. 419–444, 2004.

[124] R. W. Aldhaheri and H. K. Khalil, "Aggregation of the policy iteration method for nearly completely decomposable markov chains," *IEEE Transactions on Automatic Control*, vol. 36, no. 2, pp. 178–187, 1991.

[125] Z. Rached, F. Alajaji, and L. L. Campbell, "The kullback-leibler divergence rate between markov sources," *IEEE Transactions on Information Theory*, vol. 50, no. 5, pp. 917–921, 2004.

[126] C. L. Beck, S. Lall, T. Liang, and M. West, "Model reduction, optimal prediction, and the mori-zwanzig representation of markov chains," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*. IEEE, 2009, pp. 3282–3287.

[127] M. Vidyasagar, "A metric between probability distributions on finite sets of different cardinalities and applications to order reduction," *IEEE Transactions on Automatic Control*, vol. 57, no. 10, pp. 2464–2477, 2012.

[128] Y. Xu, S. M. Salapaka, and C. L. Beck, "A distance metric between directed weighted graphs," in *52nd IEEE Conference on Decision and Control*. IEEE, 2013, pp. 6359–6364.

[129] K. Deng, P. G. Mehta, and S. P. Meyn, "Optimal kullback-leibler aggregation via spectral theory of markov chains," *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2793–2808, 2011.

[130] B. C. Geiger, "Markov state space aggregation via the information bottleneck method," *Schedae Informaticae*, vol. 23, 2014.

[131] I. J. Sledge and J. C. Pri'ncipe, "An information-theoretic approach for automatically determining the number of state groups when aggregating markov chains," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3612–3616.

[132] A. Ando and F. M. Fisher, "Near-decomposability, partition and aggregation, and the relevance of stability discussions," *International Economic Review*, vol. 4, no. 1, pp. 53–67, 1963.

[133] A. Srivastava, M. Baranwal, and S. Salapaka, "On the persistence of clustering solutions and true number of clusters in a dataset," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5000–5007.

[134] A. L. Elsayad, "Numerical solution of markov chains," 2002.

[135] M. A. Thornton and D. I. Tamir, "Mental models accurately predict emotion transitions," *Proceedings of the National Academy of Sciences*, vol. 114, no. 23, pp. 5982–5987, 2017.

[136] D. Vidaurre, S. M. Smith, and M. W. Woolrich, "Brain network dynamics are hierarchically organized in time," *Proceedings of the National Academy of Sciences*, vol. 114, no. 48, pp. 12 827–12 832, 2017.

[137] P. Norvig, "English letter frequency counts: Mayzner revisited or etaoin srhldcu," *Dostopno na http://www. norvig. com/mayzner. html.[obiskano 2016-08-07]*, 2013.

[138] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information-theoretic coclustering," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp. 89–98.

[139] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[140] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on information theory*, vol. 52, no. 2, pp. 489–509, 2006.

[141] D. Bertsimas and A. King, "Or forum—an algorithmic approach to linear regression," *Operations Research*, vol. 64, no. 1, pp. 2–16, 2016.

[142] J. Zeng, S. Lin, and Z. Xu, "Sparse solution of underdetermined linear equations via adaptively iterative thresholding," *Signal processing*, vol. 97, pp. 152–161, 2014.

[143] W. J. Welch, "Algorithmic complexity: three np-hard problems in computational statistics," *Journal of Statistical Computation and Simulation*, vol. 15, no. 1, pp. 17–25, 1982.

[144] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of 27th Asilomar conference on signals, systems and computers*. IEEE, 1993, pp. 40–44.

[145] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on information theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[146] D. Needell and R. Vershynin, "Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit," *IEEE Journal of selected topics in signal processing*, vol. 4, no. 2, pp. 310–316, 2010.

[147] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE transactions on Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.

[148] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[149] R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707–710, 2007.

[150] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted $l_1$ minimization," *Journal of Fourier analysis and applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[151] M. Lopes, "Estimating unknown sparsity in compressed sensing," in *International Conference on Machine Learning*. PMLR, 2013, pp. 217–225.

[152] C. Ravazzi, S. Fosson, T. Bianchi, and E. Magli, "Sparsity estimation from compressive projections via sparse random matrices," *EURASIP journal on advances in signal processing*, vol. 2018, no. 1, pp. 1–18, 2018.

[153] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[154] T. A. Stamey, J. N. Kabalin, J. E. McNeal, I. M. Johnstone, F. Freiha, E. A. Redwine, and N. Yang, "Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate. ii. radical prostatectomy treated patients," *The Journal of urology*, vol. 141, no. 5, pp. 1076–1083, 1989.

[155] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 269–296, 2020.

[156] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," Stanford, Tech. Rep., 2006.

[157] R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy, "The effectiveness of lloyd-type methods for the k-means problem," *Journal of the ACM (JACM)*, vol. 59, no. 6, pp. 1–22, 2013.

[158] R. S. Varga, *Geršgorin and his circles*. Springer Science & Business Media, 2010, vol. 36.

[159] K. Kobayashi et al., *Mathematics of information and coding*. American Mathematical Soc., 2007, vol. 203.

[160] A. J. Laub, *Matrix analysis for scientists and engineers*. Siam, 2005, vol. 91.