A Rate of Convergence for Learning Theory with Consensus

Jessica Gray Gregory

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Andrew Kurdila, Chair
Javid Bayandor, Co-Chair
John Burns
Alexander Leonessa

December 08, 2014
Blacksburg, Virginia

Keywords: Learning theory, infinite dimensional estimation, convergence rate, consensus, communication network

A Rate of Convergence for Learning Theory with Consensus

Jessica Gray Gregory

(ABSTRACT)

This thesis poses and solves a distribution free learning problem with consensus that arises in the study of estimation and control strategies for distributed sensor networks. Each node i for $i = 1, \ldots, n$ of the sensor network collects independent and identically distributed local measurements $\{\mathbf{z}^i\} := \{z_j^i\}_{j\in\mathbb{N}} := \{(x_j^i, y_j^i)\}_{j\in\mathbb{N}} \subseteq X \times Y := Z$ that are generated by the probability measure ρ^i on Z. Each node i for $i = 1, \ldots, n$ of the network constructs a sequence of estimates $\{f_k^i\}_{k\in\mathbb{N}}$ from its local measurements $\{\mathbf{z}^i\}$ and from information functionals whose values are exchanged with other nodes as specified by the communication graph G for the network. The optimal estimate of the distribution free learning problem with consensus is cast as a saddle point problem which characterizes the consensus-constrained optimal estimate. This thesis introduces a two stage learning dynamic wherein local estimation is carried out via local least square approximations based on wavelet constructions and information exchange is associated with the Lagrange multipliers of the saddle point problem. Rates of convergence for the two stage learning dynamic are derived based on certain recent probabilistic bounds derived for wavelet approximation of regressor functions.

Acknowledgements

I would like to acknowledge the strong support, dedication, and guidance that I received from Andrew Kurdila throughout the entire process of developing and writing this thesis. The depth and breadth of his knowledge in this field and his patience in working with me on these difficult subjects is what made this thesis possible.

Contents

List of Figures **vi**

1 Introduction **1**

1.1 Motivation 1

1.2 Literature Review 2

1.3 Problem Statement 4

1.4 Contribution Summary 7

1.5 Outline 8

2 Learning Theory with Consensus: A Formulation **9**

2.1 Classical Learning Theory 9

2.1.1 The Learning Problem 12

2.1.2 Empirical Risk Minimization 15

2.1.3 Structural Risk Minimization 16

2.2 Graphs and Communication 18

2.3 Learning Theory with Consensus 23

2.3.1 Problem Statement in Infinite Dimensions 23

2.3.2 Formulation as a Saddle Point Problem 26

2.3.3 Norm Equivalence 30

2.3.4 Finite Dimensional Approximation 32

2.3.5 Finite Dimensional Approximation in Common Basis 37

3 The Learning Dynamic **42**

3.1 General Definitions 42

3.2 Inexact Uzawa Algorithm 44

3.3 Approximation on Uniform Partitions 47

3.4 The Solution and Convergence Rate 50

4 Numerical Studies **55**

4.1 Function Approximation 57

4.2 Simulations 61

5 Conclusion and Future Work **71**

Appendix A Functional Analysis **73**

Appendix B Probability **82**

Appendix C Graph Theory **84**

Appendix D Learning Theory **86**

Appendix E Constructions **88**

References **89**

List of Figures

2.1.1 Perceptron Neuron Model 11

2.1.2 Space $X = \mathbb{R}^n$ Divided by a Perceptron 11

2.1.3 Block Diagram of Learning Algorithm. The learning algorithm is given an input $\mathbf{x}$ from the training data and implements a function $\hat{f} = f(\mathbf{x}, \hat{\alpha})$ from the function space H. It outputs $\hat{\mathbf{y}}$ which is sent, along with the true output $\mathbf{y}$, to the loss function. The loss function establishes a criterion for choosing $\hat{f}$ and feeds that to the learning algorithm. The learning algorithm eventually arrives at the best approximation $f^* = f(\mathbf{x}, \alpha^*)$. 14

2.2.1 Simple Graph and Directed Graph 18

2.2.2 Unconnected Graph with Connected Components 19

2.3.1 Probability Space 27

2.3.2 Disjoint Support of Measures ρ_X and m 31

2.3.3 Commutative Diagram of Norm Equivalence 37

3.2.1 Dual Operator Form of Inexact Uzawa Algorithm 45

3.2.2 Primal Operator Form of Inexact Uzawa Algorithm 45

3.2.3 Coefficient Form of Inexact Uzawa Algorithm 46

3.3.1 Approximation Spaces. As the smoothness of the functions increase the size of their space decreases 48

3.3.2 Interpretation of Probability Measure in Theorem 3.3.1 50

4.0.1 Division of Set X into Subsets with Local Support 55

4.1.1 Two-Scale Basis Relationship in 1D 58

4.1.2 Basis Functions Refined from Grid $l = 1$ to $l = 2$ 58

4.1.3 Mapping of Coefficients Between Grid Levels 61
4.2.1 Sampling Densities for Two Agents 61
4.2.2 Coefficients of the True Function $f(x, y)$ 62
4.2.3 Refinement of Approximation of Function Coefficients for Agent One 63
4.2.4 Refinement of Approximation of Function Coefficients for Agent Two 64
4.2.5 Log of the Collective Error Squared Calculated at Each Level 65
4.2.6 Refinement of Approximation of Function Coefficients for Agent One 66
4.2.7 Refinement of Approximation of Function Coefficients for Agent Two 67
4.2.8 Refinement of Approximation of Function Coefficients for Agent Three . . . 68
4.2.9 Refinement of Approximation of Function Coefficients for Agent Four 69

A.0.1Graphical Representation of Relationship Between Topological Spaces 75
A.0.2Riesz Map $\mathcal{R}_u$ 76
A.0.3Commutative Diagram of Riesz Representation Theorem 78
A.0.4Commutative Diagram Showing Relation Between B^* and $B^\top$ 81

Nomenclature

$\mathcal{O}$	The agreement space. Defined as the space spanned by the vector span $\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^\top$. This space is orthogonal to the constraint space $\mathcal{Q}$.
α	Learning gain
ψ	Approximate inverse operator
ϕ^{H^i}	Basis vector of the hypothesis space H^i
ϕ^{Q}	Basis vector of the hypothesis space Q
$\boldsymbol{\ell}$	Stacked vector $\boldsymbol{\ell} = \begin{bmatrix}\ell^1 & \cdots & \ell^n\end{bmatrix}^\top$ of size $n \times 1$ whose i^{th} component is the i^{th} agent's regressor $f_{\rho_X^i} = \ell^i$
$\mathbf{f}$	Stacked vector $\mathbf{f} = \begin{bmatrix}f^1 & \cdots & f^n\end{bmatrix}^\top$ of size $n \times 1$ whose i^{th} component is the i^{th} agent's estimate f^i
$\mathbf{p}$	Stacked vector $\mathbf{p} = \begin{bmatrix}p^1 & \cdots & p^{n-1}\end{bmatrix}^\top$ of size $(n-1) \times 1$ whose j^{th} component is p^j
$\mathbf{S}$	Matrix representation of the Schur complement operator S
δ	Approximate solution tolerance
f^i	Function estimate of unknown external field that agent i has created
$f_{\rho_X^i} = \ell^i$	Regressor for agent i
$\Gamma^s_{\rho^i}$	Approximation space with measure ρ^i of functions with smoothness coefficient s
$\mathbf{M}^{\mathcal{H}}$	Diagonal matrix whose ii^{th} entries are the $\mathbf{M}^H$ over the hypothesis space H^i
$\mathbf{M}^H$	Gramian of the basis vectors over the hypothesis space H^i
$\mathbf{M}^{Q^n}$	Diagonal matrix whose ii^{th} entries are the Gramians over the hypothesis space Q
$\mathcal{H}$	Cross product space $H^1 \times H^2 \times \cdots \times H^n$

H^i	Hypothesis space for agent i
$\mathcal{Q}$	Cross product space $Q \times Q \times \cdots \times Q$
Q	Constraint space
p^i	Lagrange multipliers for agent i. These represent the information exchanged between agents.
Λ_l	Partition of domain X at resolution l into dyadic cells
$\mathbf{M}^B$	The constraint matrix. Constructed from the Laplacian matrix $\mathbf{L} = \mathbf{A} - \mathbf{D}$, where $\mathbf{A}$ and $\mathbf{D}$ are the adjacency and degree matrices respectively
B	Constraint operator $Q \to H$ which is induced via multiplication by the constraint matrix
B^*	Adjoint of the constraint operator B and maps $Q \to H$
ρ_X	The marginal probability measure on the domain space X.
$\mathcal{R}$	The Riesz map
$\underset{\sim}{\mathbf{F}}$	Stacked vector $\underset{\sim}{\mathbf{F}} = \begin{bmatrix}\mathbf{F}^{1,\top} & \cdots & \mathbf{F}^{n,\top}\end{bmatrix}^\top$ of size $n \times 1$ whose i^{th} component is the i^{th} agent's coefficient representation of it's estimate f^i
$\underset{\sim}{\mathbf{P}}$	Stacked vector $\underset{\sim}{\mathbf{P}} = \begin{bmatrix}\mathbf{P}^{1,\top} & \cdots & \mathbf{P}^{n-1,\top}\end{bmatrix}^\top$ of size $(n-1) \times 1$ whose j^{th} component is the coefficient representation of the Lagrange multiplier p^j
A	Operator acting from $H^i \to Q$
a	Number of children produced by each cell when going from grid level l to grid level $l+1$
l	Grid/resolution level
m_k	Number of samples taken during k^{th} learning epoch
n	Number of agents
S	Schur complement operator
X	Domain set
Y	Range/measurement set

Chapter 1

Introduction

1.1 Motivation

Following decades of research into control and estimation problems for single vehicles, a well-documented trend to study such problems for multi-vehicle collectives emerged in the 1990's. Notable examples of such work includes [1] and [2]. For the most part, these efforts are similar in that they consider estimates that evolve in finite dimensional spaces. The estimates usually represent the vehicular state as real variables that describe its orientation and location. It has become increasingly common, however, that a team of sensor vehicles make measurements of some external field that must be approximated by the collective. In other words, the unknown that must be approximated by the collective is not some state in $\mathbb{R}^n$ but rather some unknown function that resides in some abstract space of functions. This thesis is motivated by the previous finite dimensional work and looks to solve a learning problem where the estimates evolve in infinite dimensional space. Its objective is to integrate results from statistical learning theory, approximation theory, and consensus estimation.

The study of statistical learning theory began in the 1930's with psychologists working to understand the mechanism behind human inference; the act of drawing a conclusion based upon some known facts that are not necessarily related. By the 1960's it reached a well-known milestone when the first mechanical replication of human inference via a machine learning algorithm, the perceptron, was implemented. It was in the 1990's that statistical learning theory was codified in the seminal work by Vapnik [3]. The work by Vapnik and others was then further advanced by approximation theorists in [4], [5], who created sharper rates of convergence for the single observer estimates to unknown functions. Simultaneously, but independently the 1990's saw the development of research in the problem of consensus estimation amongst multiple vehicles. Control engineers were the first to the scene. Their initial goal was to design controllers that would maintain formations of multiple moving unmanned vehicles. The goal led naturally to the notion of defining agreement, or consensus,

of vehicle networks. The field has since expanded to include graph theoretic models of sensors communicating with one another or some central command that reflect the underlying network topology. In the preponderance of research in this field, the goal has been for the multiple observers to agree on some estimate that resides in $\mathbb{R}^n$, where n is usually of low order. *It is in combining the results of consensus estimation with the evolution of estimates in infinite dimensional space from statistical learning and approximation theory that this thesis makes a unique contribution to the current available research.*

1.2 Literature Review

Much of the literature relevant to this thesis can be broken into three main areas: consensus estimation and control theory; optimization theory; approximation and learning theory. In the area of consensus estimation and control, there are many efforts to develop new or modify known algorithms to achieve consensus. In contrast to this thesis, these works almost always treat measurement processes that evolve in some finite dimensional space. Examples of such work can be found in [6–11].

The research area of decentralized optimization covers problems from a broad range of applications including supply-chain management, scheduling/decision-making, energy supply, cash balances for decentralized firms, chemical processes, and many other disparate fields. In principle, decentralized optimization treats quite general optimization problems, and consensus estimation can be viewed as one highly structured example where communications are precisely defined. Examples of such work can be found in [12–16]. Most of the studies of decentralized optimization seek to minimize a global cost function by simultaneously seeking to minimize local cost functions. Again, nearly all of the relevant work treats minimization of real-valued functions of several real variables. A particular example of this problem is decentralized optimization over multi-agent sensor networks. Examples of such work can be found in [17–22].

Approximation theory is mostly concerned with finding sets of simple functions to approximate some unknown function and to determine the fidelity of such approximation. Distribution-free learning is a particular problem initially studied in the realm of statistical learning theory that has more recently come under the scrutiny of approximation theorists. One of the foundational texts on statistical learning can be found in [3]. In distribution-free learning, the goal is to construct an algorithm that selects a function from some hypothesis space that best represents the data. An approximation theoretic framework for the distribution-free learning problem was introduced in 2002 by Cucker and Smale in [4] and has formed the basis for efforts in [23–27]. All of these efforts treat measurement processes by a single observer. However, the optimization problem is expressed in terms of a real-valued functional taking as arguments elements of infinite dimensional space.

We have noted that this thesis focuses on consensus learning problems wherein the esti-

mates evolve in infinite dimensional spaces, and have observed that most publications on consensus estimation treat estimates in finite dimensional spaces. However, a few examples of work where the estimates evolve in infinite dimensional spaces has been considered by Predd, Kulkarni, and Poor and can be found in [28–31]. Reference [28] provides a general survey of the distributed learning problem over wireless sensor networks (WSN). The focus is in solving the problem using reproducing kernel Hilbert space (RKHS) methods. They classify distributed learning approaches into two classes: those with a fusion center and those with ad-hoc structures where the network topology may evolve with time. For the fusion center class one approach employs clustering and may filter the data before communicating with the fusion center. In the other method each sensor communicates with the fusion center. However, in a WSN network it is not guaranteed that each sensor is always able to communicate with the fusion center when requested. Reference [29] studies the statistical limits of this method and considers an ad-hoc WSN scenario where the presence or absence of a fusion center is not assumed and the individual sensors can do some processing. Implicit in the consideration of the WSN is the assumption that the possible topologies of the network, which describes which sensors are communicating, are constricted by the physical obstructions blocking communication between sensors. Communication is constrained by energy and bandwidth consideration. These constraints result in local communication being preferred to global communication where all raw data from each node is sent to a central or fusion node. Predd, Kulkarni, and Poor draw upon classical supervised learning models, such as nonparametric least squares regression and regularized kernel methods, to develop a distributed learning model for the WSN. In their model the agents have limited access to a training database. The agents go through a two-stage learning process: 1) first they build their local estimate of the functional dependency, 2) they subsequently update the training data set predictions provided by their local estimate. An update of the agent's estimate is obtained from the solution to a relaxation of the classical kernel-linear, least squares regression optimization problem. This creates a learning dynamic where the agents only have to agree on their common data points rather than agree globally. In addition to not directly communicating with their neighbors, the agent in this formulation pass real-valued information during the update stage. They do not communicate the details or structure of the estimate, only its functional values.

Perez-Cruz and Kulkarni in [32] derive a variation of the message passing algorithm given in [30]. In [30] two cases of the message passing algorithm are considered, binary classification and regression estimation. The work in [32] treats the problem of nonparametric estimation. The approach induces a communication burden independent of the number of neighbors and allows for asynchronous updating. This is another paper where the minimization problem is stated in terms of a RKHS.

An alternative strategy defines an inference problem on graphical models. Zheng, Kulkarni, and Poor apply this approach in [33] and [34] to parameterized, low-dimensional cases and nonparameterized, high-dimensional cases. The network is represented via a graph where sensors are nodes and adjacent nodes are required to have similar estimators. The infor-

mation passed between nodes describes distributions of functions or sample functions. The algorithm performs local training followed by global inference. The goal is to compute the marginal distribution based on estimated weights for each edge in the graph and estimate distributions for each node. In the high-dimensional case a sampling algorithm is used to search for optimal estimators. The problem of finding the optimal estimator is converted into a maximum a posteriori(MAP) probability problem. Again, however, the estimates evolve in a fixed, in finite dimensional space.

Another distributed learning problem that is treated by Zheng, Kulkarni, and Poor, distributes data amongst the agents in the network based on some set of attributes. The authors examine this problem in [35] making no assumption on the presence or absence of a fusion center. Since each agent has access to different attributes of the training data, they will thus construct estimators having different forms. This fact makes the evaluation of agreement among their respective estimators more difficult. The problem is cast as a two-stage optimization problem and solved using a gradient-based algorithm. The authors further investigate this scenario in [36] and [37]. In [36] the two-stage learning dynamic is set up for a regression problem that employs a fusion center coordinating all the agents. They compare the performance of a greedy algorithm, the Simple Iterative Projection algorithm, and a parallel algorithm. Convergence and uniqueness of the estimator is established assuming that the unknown function that the agents are attempting to estimate is a polynomial. In [37] a prototypical residual-refitting algorithm is used as a test case to evaluate the greedy algorithm and an Intelligent Agent Selection algorithm. Both have a fusion center that filters agent participation, while the test case treats agents the same. In both cases the training error converges faster than the test case but each demands a higher bandwidth for data transmission.

1.3 Problem Statement

Our problem considers a set of agents or nodes whose communication is defined in terms of a communication network. This network is a connected undirected graph as discussed more fully in Section 2.2. This means that communication between a node and its neighbors flows in both directions and that there are no isolated nodes or islands of nodes. Together these agents travel through physical space X taking measurements of some unknown external field whose values are in Y. In this thesis we will always assume that X is a compact normed vector space and Y is a compact subset of a normed vector space. More specifically, $X \subseteq \mathbb{R}^n$ and $Y \subseteq \mathbb{R}$ are compact. While the agents move through space X, the underlying communication topology of the connected graph they create is assumed not to change in this thesis. In other words, each agent collects its own set of measurements of the external field taken at various times as it moves through the domain. After collecting a predetermined number of measurements, the agents each estimate, or infer, the functional dependency between a location in space and the value of the external field. Roughly, we can think of the external

field as some unknown function $f : X \to Y$ that we must identify from our measurements. An estimate f^i of the function f by agent i is also a function, $f^i : X \to Y$. After each agent creates its own estimate, f^i, where i represents agent i, it exchanges information about f^i with its neighbors and vice-versa. Each agent i then repeats the process: it collects measurements and then improves f^i by using the new data points, its previous estimate f^i, and the information about the estimates of its neighbors $f^{n(i)}$, where $n(i)$ represents the neighbors of the i^{th} agent. Our goal is to define a strategy that guarantees that all of the agents converge to the same estimate, f^*, called the consensus estimate, and to find a rate of convergence.

The process and theory of inferring a functional dependency from a set of data points is one of the fundamental problems of learning theory. The requirement that there be consensus motivates us to refer to the estimation problem described above as one of *learning theory with consensus*. An important feature of our problem is that we are interested in learning theory with consensus where we allow the functions f^i to exist and evolve in some infinite dimensional space.

To create a mathematical framework for our problem, we start with a single agent. The agent moves along some path in the compact set $X \subseteq \mathbb{R}^N$ taking measurements, $y_j \in Y \subseteq \mathbb{R}$ of some external field at point $x_j \in X$ for $j \in \mathbb{N}$. Note that $y_j \in \mathbb{R}$, is the j^{th} measurement taken at the point $x_j = (x_{j1}, x_{j2}, \ldots, x_{jn}) \in \mathbb{R}^N$. It is convenient in the discussion of the algorithms defined below to group measurements into epochs or cycles which are sets with cardinality m and are indexed by another integer $k \in \mathbb{N}$. A single agent, therefore, collects during the k^{th} epoch the set of points

$$\{(x_j, y_j)\}_{j=n_k:m_k} = \{z_j\}_{j=n_k:m_k} = \mathbf{z}_{n_k:m_k}$$

where $m_k = m \cdot k$ and $n_k = m(k-1)+1$, for $k = 1, 2, \ldots$. The observations $\{z_j\}_{j\in\mathbb{N}}$ are assumed to be independent and identically distributed by some unknown probability measure ρ on $X \times Y = Z$. This sampling is often described by a marginal probability measure ρ_X and a conditional probability measure $\rho(\cdot|x)$. With this interpretation in mind, $\{x_k\}_{k\in\mathbb{N}}$ are generated by the marginal measure ρ_X where $\rho_X(A) = \rho(A \times Y)$ for all $A \subseteq X$. The $\{y_k\}_{k\in\mathbb{N}}$ are generated by $\rho(y|x)$.

From each sequence of observations the single agent thus constructs a sequence of local estimates, $\{f_k\}_{k\in\mathbb{N}}$, of the functional dependency f between the data in the domain X and range Y. The function f_k is the k^{th} sequential local approximation based on the set of local data the agent collects in the k^{th} epoch, $\{x_j, y_j\}_{j=n_k:m_k}$.

To measure the fidelity of our estimates $\{f_k\}_{k\in\mathbb{N}}$ to the true unknown function f, we use a technique familiar in classical learning theory where a loss functional is introduced

$$\mathcal{J}(f) := \int_Z L(f(x), y)\rho(dz)$$

that is defined in terms of some kernel $L(\cdot,\cdot) : Y \times Y \to \mathbb{R}$ and the unknown probability measure ρ on $Z = X \times Y$. Therefore, the problem statement for a single agent is that we want to construct a sequence of estimates $\{f_k\}_{k\in\mathbb{N}}$ that converge to the minimizing or optimal function f^* in the sense that

$$f^* := \arg\min_{f\in H} \mathcal{J}(f)$$

where H is the set of admissible functions in the hypothesis space.

In the multiple agent case we assume that each agent i creates its own estimate f_k^i. The cardinality m^i of the k^{th} data set collected by agent i, $\{(x_j^i, y_j^i)\}_{j=n_k^i:m_k^i}$, is not necessarily identical to that of the other agent's k^{th} data set. We want to define an algorithm that guarantees that the sequence of estimates $\{f_k^i\}_{k\in\mathbb{N}}$ for each agent $i = 1,\ldots,n$ converges to the single function, f^*, i.e.

$$f_k^i \to f^*$$

in some suitable topology. To accomplish this we introduce a two stage learning dynamic that the agents will employ. The first part of the learning dynamic is referred to as the local learning stage. During the local learning stage k each agent i updates its local estimates using the nonlinear mapping, T_k^i,

$$f_k^i = T_k^i(\mathbf{z}^i_{n_k^i:m_k^i}, f_{k-1}^i, p_{k-1})$$

In this recursion the vector p_{k-1} represents the information that is exchanged between an agent and its neighbors at the end of epoch $k-1$. Once the agents construct their new estimates, they move into the second stage of the learning dynamic, the information exchange. We assume that the information shared depends only on the estimates made during the k^{th} epoch of local learning,

$$p_k = p_k(f_k^1, f_k^2, \ldots, f_k^n)$$

In practice, the information exchange corresponds to collecting information functionals that are shared among agents and distributing the information based on the connectivity graph. This process is summarized in the pseudocode below.

Algorithm 1.3.1: TWO STAGE LEARNING PROCESS(f_0, p_0)

Choose the initial estimates f_0 and information functionals p_0
$f_0 = (f_0^1, f_0^2, \ldots, f_0^n)$
$p_0 = (p_0^1, p_0^2, \ldots, p_0^{(n-1)})$
repeat
For each epoch $k \in \mathbb{N}$

do Update the local estimate f_k^i
$f_k^i = T_k^i(\mathbf{z}_{n_k:m_k}, f_{k-1}^i, p_{k-1})$

do Calculate information to be exchanged
$p_k = p_k(f_k^1, f_k^2, \ldots, f_k^n)$
Transmit new information
$(p_k)_i \longrightarrow n(i)$ **comment:** i^{th} entry goes to $n(i)$, the neighbors of agent i

until Convergence

As in the case of a single agent, we want to make precise how a sequence of estimates $\{f_k^i\}_{k\in\mathbb{N}}$ converges to an optimal estimate in some sense. In the multiple agent case, we choose the loss functional (where for the sake of notational simplicity we continue using $\mathcal{J}(\cdot)$ and $L(\cdot,\cdot)$)

$$\mathcal{J}(\mathbf{f}) := \int_{Z^n} L(\mathbf{f}(\mathbf{x}), \mathbf{y})\rho^1(dz_1) \times \rho^2(dz_2) \times \cdots \times \rho^n(dz_n)$$

where $\mathbf{f} = \begin{bmatrix} f^1 & f^2 & \cdots & f^n \end{bmatrix}^\top \in H^1 \times H^2 \times \cdots \times H^n = \mathcal{H}$. Therefore, the problem statement for the multiple agent case is to construct a sequence of estimates $\{\mathbf{f}_k\}_{k\in\mathbb{N}}$ that converge to $\mathbf{f}^*$ defined as

$$\mathbf{f}^* = \underset{\mathbf{f}\in\mathcal{H}\cap\mathcal{O}}{\arg\min}\, \mathcal{J}(\mathbf{f})$$

where $\mathcal{O}$ is the collection of $\mathbf{f}$ that are in agreement among the agents. The definition of the set $\mathcal{O}$, that establishes what constitutes agreement, is discussed in Section 2.3. Note that the limiting solution of the sequence of estimates is expressed as a constrained optimization problem.

1.4 Contribution Summary

The contribution of this thesis to the current literature is proving a convergence rate in space for consensus in infinite dimensional space. As stated in the literature review the majority of work has been in finding consensus and a convergence rate in time in finite dimensional space. In the cases where infinite dimensional space was considered it was with the restriction that

the space be a reproducing kernel Hilbert space (RKHS). Not only does this thesis provide a convergence rate in space but also a constructive algorithm for practical application with infinite dimensional consensus problems.

1.5 Outline

This thesis considers three main topics: the formulation of the learning with consensus problem, the finite dimension construction of approximations to our infinite dimensional problem, and the numerical results. We begin in Chapter 2 with an introduction to classical learning theory. Section 2.1 and Subsection 2.1.1 explain the origin of the learning problem and discuss how the current question can be cast as that of finding the functional dependency of a set of data points. Then a discussion follows in Subsections 2.1.2 and 2.1.3 of current methods and theory for assessing the accuracy of the estimates. We derive the conditions necessary for solving the problem of finding a function f^* that minimizes the cost functional $\mathcal{J}(f)$ when the probability measure ρ over $Z = X \times Y$ is unknown, called the distribution free learning problem.

Section 2.2 provides a short introduction to graph theory. Some of the more basic terms are left to the appendix for the interested reader. This section focuses on some of the matrices that can be constructed from a graph. It makes clear how we can use these to enforce the constraint on our problem that all nodes in our graph must arrive at the same approximation, f^*.

Section 2.3 gives the formulation of the learning problem within the confines of consensus. Subsections 2.3.1 and 2.3.2 discuss how the optimization problem for learning with consensus can be cast in a tractable form. In these subsections we recast the original problem statement into an equivalent saddle point problem. From this formulation we are able to create approximants that have finite dimension. Subsections 2.3.3 and 2.3.4 provide the background and process for this formulation.

In particular, Section 2.3.4 discusses the creation of a sequence of functions evolving in infinite dimension, $\{f_k^i\}_{k\in\mathbb{N}}$, to functions that are finite dimensional and provide a good approximation of the true function. The inexact Uzawa algorithm is used to create a sequence of convergent approximations to the solution of the saddle point problem that characterizes the solution of the constrained optimization problem. The various forms of the Uzawa algorithm are described and practical algorithms are discussed in Section 3.2. This section provides a more in-depth description of the two stage learning dynamic and how it is interpreted as an Uzawa algorithm.

Finally, this thesis discusses numerical results in Chapter 4. Matlab code has been written with which to test the hypothesis that the agents will all converge, at the proposed rate, to the same consensus estimate.

Chapter 2

Learning Theory with Consensus: A Formulation

2.1 Classical Learning Theory

In this thesis classical learning theory refers to the theoretical foundations and computational algorithms that have been derived over the past few decades to solve problems of statistical learning. Perhaps the best way to describe statistical learning is to state its goal. In his authoritative text on statistical learning theory [3] Vladimir Vapnik refers to statistical inference as the attempt to find reliable methods for the following problem:

> "Given a collection of empirical data originating from some functional dependency, infer this dependency."

The first systematic approach to this problem was formulated in the 1920's. In these nascent years of study two alternate approaches were identified, parametric and nonparametric inference. In parametric inference one assumes that the physical laws that are generating the stochastic data are well known, and hence the functional form is known up to a finite number of unknown parameters. The goal of this approach is, therefore, to estimate these unknown parameters. In general or nonparametric inference one does not have *a priori* knowledge of the structural form of the unknown function of statistical laws underlying the problem. Hence in this case the learning problem seeks to approximate not just the parameters but the whole function.

The parametric approach was studied extensively during the 1930's-1960's. It dominated much of the research in statistical inference during this period. The goal was to"create simple statistical methods of inference for solving real-life problems" [3]. This approach is dependent upon three assumptions being true:

1. It is possible to define a set of functions, linear in their parameters, that are good approximations of the optimal solution.

2. The probability distribution shared by all the data points in $S = \{(x_j, y_j)\}_{j\in\mathbb{N}}$ is the normal law.

3. The maximum likelihood method is the most appropriate tool for estimating the unknown parameters.

However, at the start of the 1960's when large computers with greater processing power came online, it was shown that these assumptions were often not true for real-life problems. In particular, R. Bellman showed in 1961 that increasing the number of factors that must be considered, in other words, increasing the dimensionality of the sample space, leads to sparser data and hence requires an exponentially increasing number of observations in order to maintain the same accuracy of the estimated unknown parameters. This translates into exponentially increasing the amount of processing space needed which quickly becomes untenable. Bellman referred to this phenomenon as the "curse of dimensionality."

In the late 1950's the focus shifted to the nonparametric approach. In 1958 F. Rosenblatt, a physiologist, proposed a computer program, a learning machine, he called the perceptron to "illustrate some of the fundamental properties of intelligent systems in general" [38]. He demonstrated that his perceptron was capable of learning to recognize patterns. In the simplest pattern recognition problem this involves categorizing a set of points, $X \subseteq \mathbb{R}^n$ into two different classes.

Rosenblatt's perceptron was a modification of the McCulloch-Pitts neuron model. In the McCulloch-Pitts model each neuron takes a binary input $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in X \subseteq \mathbb{R}^n$ and produces a binary output $y = \{0, 1\}$. This is constructed by assigning a threshold θ to the neuron(node), summing the inputs, and outputting a 1 if the threshold is met and 0 otherwise. Different threshold functions, such as NAND or NOR, can be replicated by adjusting the threshold but others, such as XOR[1] cannot. Additionally, it does not allow for weightings of the inputs. This is quite limiting, particularly since psychologist Donald Hebb proposed that neurons that are repeatedly activated can become organized into a functional unit, essentially weighting their inputs. Therefore, the perceptron altered the McCulloch-Pitts model to allow for this phenomenon and thus allowed for weights w_i to be placed on inputs x_i.

The arithmetic test performed by the perceptron at each neuron is $\mathbf{w} \bullet \mathbf{x} - \theta > 0$, where $\mathbf{x}$ is the input vector, $\mathbf{w}$ is the weight vector, and θ is the threshold. This gives the following

[1] NAND : $\mathbb{F}_2^n \to \mathbb{F}_2$, $\mathrm{NAND}(\mathbf{x}) = 1 + x_1 + x_2 + \cdots + x_n$; NOR : $\mathbb{F}_2^n \to \mathbb{F}_2$, $\mathrm{NOR}(\mathbf{x}) = (1 + x_1)(1 + x_2)\cdots(1 + x_n)$; $\mathrm{XOR} : \mathbb{F}_2^n \to \mathbb{F}_2$, $\mathrm{XOR}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathrm{sum}(x_i)_i = 1 \\ 0 & \text{otherwise} \end{cases}$

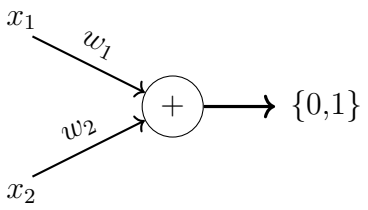


Figure 2.1.1: Perceptron Neuron Model

functional dependency for the perceptron between its input $\mathbf{x}$ and its output $y = f(\mathbf{x})$,

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \bullet \mathbf{x} - \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$

where θ is normally set to zero. Rosenblatt's learning algorithm successfully divides the space X into the equivalence classes $[x]_0$ and $[x]_1$, each separated by the linear hyperplane $\mathbf{w} \bullet \mathbf{x} - \theta = 0$. The influence of θ is to simply shift the hyperplane from the origin.

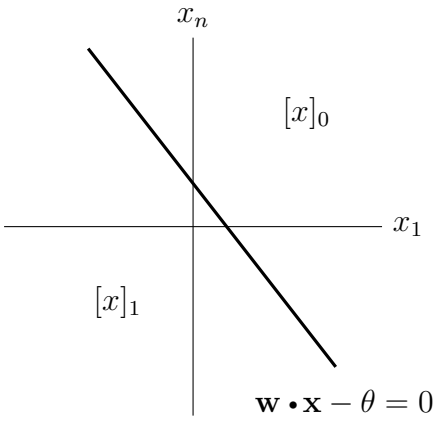


Figure 2.1.2: Space $X = \mathbb{R}^n$ Divided by a Perceptron

Rosenblatt demonstrates that the perceptron can learn from a finite training data set the coefficient vector $\mathbf{w}$. Given an unseen data set it can accurately classify it. That is, the perceptron can generalize. After the perceptron other learning machines were proposed that were equally capable. According to Vapnik [3], this lead to the question,

"Does there exist a general principle of inductive inference that they implement?"

In other words, is there an underlying general principle that all successful learning machines employ? A.B. J. Novikoff blazed a trail to solving this problem when he proved convergence rates for the perceptron. He connected the ability to generalize to the principle of minimizing the error from the training set.

2.1.1 The Learning Problem

The perceptron was proposed by a physiologist interested in understanding neural networks and the process of making decisions. It was not an attempt to answer the original question of statistical inference stated at the beginning of this section. But from the perceptron arises a new field with new problems to solve, machine learning. The field studies the question of how you create a machine that learns. Analysis and development of theory in machine learning then leads to a recasting of the problems in statistical inference. Research studying the newly cast problems in statistical inference in turn spawns new answers and helps to build the theoretical foundations in machine learning and the learning problem.

In the modern formulation of the learning problem, we have three key ingredients: the training set, $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2) \cdots (\mathbf{x}_n, \mathbf{y}_n)\}$, the hypothesis set H of admissible functions, $f(\mathbf{x}, \alpha) \in H$ where $\alpha \in \Lambda$[2] is the set of parameters, and the learning algorithm or machine. We can think of the learning machine as a mapping from the training set S to a function $f(\mathbf{x}, \alpha) \in H$. We can also think of it as a step-by-step procedure for processing measurements so that an approximate answer evolves and becomes more accurate as more and more data from S is processed. The problem is how to get a machine to learn: from a limited number of data points $(\mathbf{x}_i, \mathbf{y}_i)$, $i = 1, 2, \ldots N$, learn either the unknown functional dependency $f(\mathbf{x}_i) = \mathbf{y}_i$ or how to recognize a pattern to the data points.

We see that this type of learning problem breaks machine learning into two categories: 1) finding a functional dependency or 2) finding a pattern to the data. Formally speaking, machine learning itself can further be classified into supervised learning and unsupervised learning. In supervised learning we are given the correct answer to our input $\mathbf{x}_i \in X$ for all measurements on some training set. We therefore want to find the function that best approximates the output $\mathbf{y}_i \in Y$ over all the elements in the training set. In unsupervised learning we do not necessarily know the exact value of the output and thus want the learning algorithm to recognize a pattern in the data points in S.

With supervised learning we are brought back to the original goal: how do we infer dependency? Thus, the statistical learning problem is now couched in terms of a learning problem, and we can solve it using this structure and theoretical foundation. Below is a general outline of the learning problem. More specialized versions can be derived from it, based on properties of the data, and the approach taken to achieve the goal.

Outline of the Learning Problem

Goal: Choose from the given set of functions $f(\mathbf{x}, \alpha) \in H$, $\alpha \in \Lambda$, the one that best approximates the set of data $\{(\mathbf{x}_i, \mathbf{y}_i)\}$.

[2]Note that Λ is an arbitrary set and hence α may be a scalar, a vector, or an abstract element.

Learning Model: The learning model consists of an input space, an output space, and a learning algorithm.

1. The input space $X \subseteq \mathbb{R}^n$ contains the random vectors $\mathbf{x} \in X$ that are independently and identically distributed (i.i.d.) according to a fixed but unknown probability distribution ρ_X.[a]
2. The output space $Y \subseteq \mathbb{R}^n$ contains the range measurements $\mathbf{y} \in Y$ associated to every input $\mathbf{x}$, which are distributed according to the conditional distribution function, $\rho(\cdot|\mathbf{x})$.
3. The learning machine(algorithm) chooses a function from H and evaluates it over the input space X.

Learning Process: The learning machine is provided samples from the training set $Z = \{\mathbf{z}_i\}_{i=1}^n$, where each $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)$, is drawn randomly and independently according to the unknown joint distribution function ρ on Z with $\rho(d\mathbf{z}) = \rho_X(d\mathbf{x})\rho(d\mathbf{y}|\mathbf{x})$.[b] Using the entire training set the learning machine chooses a function from the set of admissible functions H.

[a] Use of the term probability distribution can lead to confusion since not all authors define the term consistently. In this thesis the term probability distribution refers to the *distribution* of a random variable/vector. See Appendix B for the details.

[b] In particular, this means $\int f(\mathbf{z})\rho(d\mathbf{z}) = \int f(\mathbf{x}, \mathbf{y})\rho_X(d\mathbf{x})\rho(d\mathbf{y}|\mathbf{x})$

In supervised learning the underlying algorithm uses a process called learning from example whereby the algorithm calculates the discrepancy between $\mathbf{y}$ and $f(\mathbf{x}, \alpha)$ via a loss function. The learning process then modifies $f(\mathbf{x}, \alpha)$ until the discrepancy converges to a minimum value over the training set. The algorithm modifies $f(\mathbf{x}, \alpha)$ by varying the parameter value $\alpha \in \Lambda$. When approximating the outputs $\mathbf{y} \in Y$, the learning machine can either attempt to predict the output, in other words, achieve the best result in prediction, or it can identify the output, in other words, find a function that is actually close to $\mathbf{y}$ as measured by some given metric. These different views dictate different approaches to formulating the goal of the learning problem. In this thesis we want to identify the output $\mathbf{y}$ and will use risk minimization to do so.

In risk minimization we introduce a function $L : Y \times H \to \mathbb{R}$ that measures the fidelity or error in representing the data by some function $f \in H$ called the loss function. Depending on the learning problem we are trying to solve, the loss function will vary. The three main learning application problems are pattern recognition, regression estimation, and density estimation. Pattern recognition is the original problem addressed by the perceptron. Each of these problems can be cast in terms of the general framework discussed in this section. Density estimation is concerned with constructing an estimate of the underlying probability distribution that generates the training set. Regression estimation seeks some $f \in H$ that best approximates the output, as measured by the loss function. We will use regression

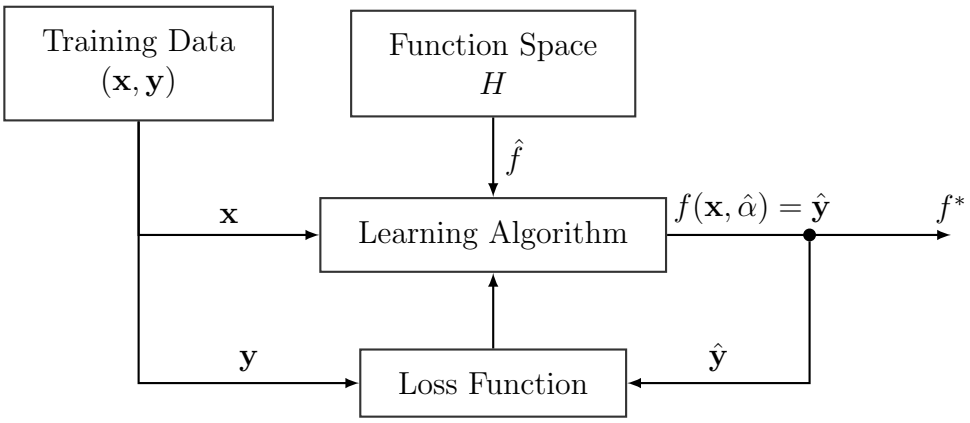


Figure 2.1.3: Block Diagram of Learning Algorithm. The learning algorithm is given an input $\mathbf{x}$ from the training data and implements a function $\hat{f} = f(\mathbf{x}, \hat{\alpha})$ from the function space H. It outputs $\hat{\mathbf{y}}$ which is sent, along with the true output $\mathbf{y}$, to the loss function. The loss function establishes a criterion for choosing $\hat{f}$ and feeds that to the learning algorithm. The learning algorithm eventually arrives at the best approximation $f^* = f(\mathbf{x}, \alpha^*)$.

estimation.

We thus have a *supervised* learning problem where we seek to *identify* the output $\mathbf{y}$ by minimizing a loss functional associated with *regression estimation*. The calculus of variations gives us the rigorous formulation for this problem. We seek to find the function f^*, that minimizes the loss or risk functional

$$\mathcal{J}(f) = \int L(\mathbf{y}, f(\mathbf{x}))\rho(d\mathbf{x}, d\mathbf{y}) \tag{2.1.1}$$

A first step to solving Equation 2.1.1 is to select the class of admissible functions H. Techniques from calculus of variations and optimization theory can be easily employed if the hypothesis space H is selected and measure ρ is known. That is not the case with the learning problem. If we are working with parameterized functions so that the set $H = \{f(\mathbf{x}, \alpha)\}$,then a specific $\alpha = \hat{\alpha}$ in Λ defines a specific function $\hat{f}(\mathbf{x}) = f(\mathbf{x}, \hat{\alpha})$. Let $\rho(\mathbf{x}, y) \equiv \rho(\mathbf{z})$ be defined on X, then from our parametric notation Equation 2.1.1 can be rewritten as

$$\mathcal{J}(\alpha) = \int P(\mathbf{z}, \alpha)\rho(d\mathbf{z}), \quad \alpha \in \Lambda \tag{2.1.2}$$

where

$$P(\mathbf{z}, \alpha) = L(\mathbf{y}, f(\mathbf{x}, \alpha)).$$

The risk functional can be interpreted as the expected value of the loss. Our goal is to find the function $f^*(\mathbf{x}) = f(\mathbf{x}, \alpha^*)$ that minimizes $\mathcal{J}(\alpha)$ given that the joint probability distribution function $\rho(d\mathbf{z}) = \rho(d\mathbf{x}, d\mathbf{y})$ is unknown and the only information available is the samples in the training set. This is the sample space, not the training set $\{(\mathbf{x}_i, \mathbf{y}_i)\}$.

The question arises, though, of how can one minimize $\mathcal{J}(\alpha)$ if the probability distribution is unknown.

2.1.2 Empirical Risk Minimization

Because the probability measure ρ is unknown, it is not possible to use Equation 2.1.1 to compute the risk in algorithms. In practice, an approximation risk is introduced that can be evaluated from the available data or measurements. This process is called empirical risk minimization(ERM). ERM replaces Equation 2.1.2 with the empirical risk functional,

$$\mathcal{J}_{\text{emp},\ell}(\alpha) = \frac{1}{\ell}\sum_{i=1}^{\ell} P(\mathbf{z}_i, \alpha), \quad \alpha \in \Lambda \tag{2.1.3}$$

which is constructed using the training set Z. Note that the expression in Equation 2.1.3 can always be computed using the available set of samples, in contrast to Equation 2.1.2 which depends on the unknown measure ρ. The learning process approximates the function that minimizes the true risk in Equation 2.1.2 by a function $P(\mathbf{z}, \alpha^*)$ that minimizes the empirical risk in Equation 2.1.3. This is a general method and can be applied to any of the three main learning problems defined earlier. When we employ ERM using the functional defined in Equation 2.1.2, we introduce a family of minimization problems for $\ell = 1, 2, \ldots$

$$\alpha_\ell^* = \arg\min_{\alpha} \mathcal{J}_{\text{emp},\ell}(\alpha)$$

It is hoped that the sequence of empirical optima converges to the optimal risk

$$\mathcal{J}_{\text{emp},\ell}(\alpha_\ell^*) \longrightarrow \mathcal{J}(\alpha^*) \text{ in } \mathbb{R}$$

such that the optimizing functions $\{f(\cdot, \alpha_\ell^*)\}_{\ell\in\mathbb{N}}$ also converge

$$f(\cdot, \alpha_\ell^*) \longrightarrow f^* \text{ in } H$$

in some suitable topology on the hypothesis space.

This problem, and similar considerations, have been studied for some time. Vapnik and Chervonenkis in [3] define consistency - the property that as the size of the sample set increases then there will exist a sequence $\{\ell_i\}_{i\in\mathbb{N}}$ for which $\{\mathcal{J}_{\text{emp},\ell_i}(\alpha_{\ell_i}^*)\}_{i\in\mathbb{N}}$ converge to the minimum - and derive the necessary and sufficient conditions for when ERM is consistent. Along with consistency one must also consider the rate of convergence. In estimating the rate of convergence we are interested in ascertaining the generalization ability of the learning machine. For a given set of functions, we are looking for estimates of "the value of achieved risk for the function minimizing the empirical risk" [3], and bounds on the difference between the value of achieved risk and that of minimal possible risk. It is these bounds that determine the learning machine's generalization ability. Note that we are still referring to learning machines that employ the ERM inductive principle. Consideration of the rate of convergence leads to the development of a new inductive principle, that of structural risk minimization (SRM).

2.1.3 Structural Risk Minimization

ERM works best with large sample sizes. When the sample size becomes small, however, ERM does not guarantee that a small empirical risk $\mathcal{J}_{\text{emp},\ell}(\alpha)$ guarantees a small expected risk $\mathcal{J}(\alpha)$. Therefore, we need an inductive principle that will work when the sample size is small. One technique to address this problem is the technique of structural risk minimization. Before we discuss this method, we must first introduce the VC dimension. The VC dimension is a measure of the"richness" of the hypothesis space. Since the underlying probability distribution of the data points in our set $S = X \times Y$ is unknown, we would like to have some measure for determining whether our inductive method will work. We get some insight into this from the VC dimension which provides an upper bound on the true expected risk. Vapnik and Chervonenkis have found that if the VC dimension of the hypothesis space H is finite, then as the number of data points increases the empirical error converges uniformly, with a probability approaching 1, to the minimum value of the expected error. More precisely, if ℓ is the number of points in our sample space S and h is the VC dimension of our function space $H = \{f(\mathbf{x}, \alpha)\}$, then with probability at least $1 - \delta$, δ, $0 \leq \delta \leq 1$

$$\mathcal{J}(\alpha) \leq \mathcal{J}_{\text{emp},\ell}(\alpha) + 2\sqrt{2\frac{h(\log \frac{2\ell}{h} + 1) + \log \frac{\delta}{4}}{\ell}}$$

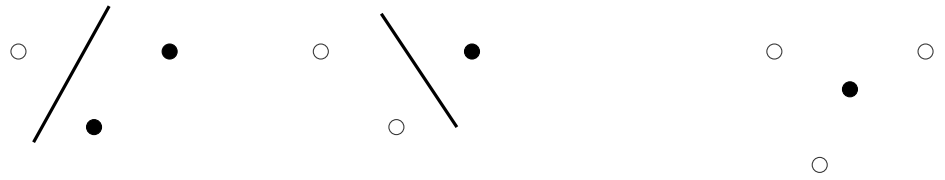
What this tells us is that the difference between the true and the empirical risk is at most

$$\sqrt{\frac{h \log \ell}{\ell}}$$

The question we must now answers is how do we calculate the VC dimension of a space. For the calculation of the VC dimension we define it to be a measure of the ability of the hypothesis space to *shatter* some set of n points in our data set X. To shatter a set means that we take n points and give them any possible labeling, for example, assign a value from $\{0, 1\}$. If for any arrangement of these points in X there exists a function from H that will partition X into two disjoint subsets, then H is said to have infinite VC dimension of *at least* n. If, however, arbitrarily large finite sets of X can be shattered by H then it is said to have VC dimension infinity.

Example 1. *The clearest example of this is to take a hypothesis space of linear functions and consider points $x \in \mathbb{R}^2$. We take n points in X, which are non-colinear, and to this fixed arrangement of points we assign a value from $Y = \{0, 1\}$ to each point. In this case let filled circles equal 0 and empty circles equal 1. Each possible assigned arrangements is a learning problem. There are a total of 2^n possible learning problems. If for any of these learning problems we can find a hypothesis $h \in H$, in our case a line, that separates the points labelled 0 from those labelled 1, then we say that H shatters n points. In the figure below we see two examples for the case of $n = 3$ and one for the case of $n = 4$. In both cases we have a fixed arrangement and choose arbitrarily from Y values for the*

points. In the case of $n = 3$ we can imagine from these two examples that no matter the values assigned we will always be able to separate the white points from the black ones. However, in the case of four points we are able to produce an example where no linear function can separate them. Therefore, the function space H of linear functions has a VC dimension of 3.



The name structural risk minimization is suitably named since the first step in this principle assigns a"complexity" to the elements of the set H of admissible functions. It is assumed that the set H of functions $P(\mathbf{z}, \alpha)$, $\alpha \in \Lambda$, is such that there exists subsets $H_k = \{P(\mathbf{z}, \alpha), \alpha \in \Lambda_k\}$, $H_1 \subset H_2 \subset \cdots \subset H_n \cdots$ with the following properties

1. The VC dimension h_k of each subset $H_k \subseteq H$ is finite and hence $h_1 \leq h_2 \leq \cdots \leq h_n \cdots$
2. Each subset $H_k \subseteq H$ contains either

   a set of bounded functions,

   $$0 \leq P(\mathbf{z}, \alpha) \leq B_k, \quad B_k = \sup_{\alpha, \mathbf{z}} P(\mathbf{z}, \alpha), \quad \alpha \in \Lambda_k$$

   or a set that contains unbounded functions such that the functions satisfy

   $$\sup_{\alpha \in \Lambda_k} \frac{(\int P^p(\mathbf{z}, \alpha) d\rho(\mathbf{z}))^{1/p}}{\int P(\mathbf{z}, \alpha) d\rho(\mathbf{z})} \leq \tau_k < \infty, \quad p > 2$$

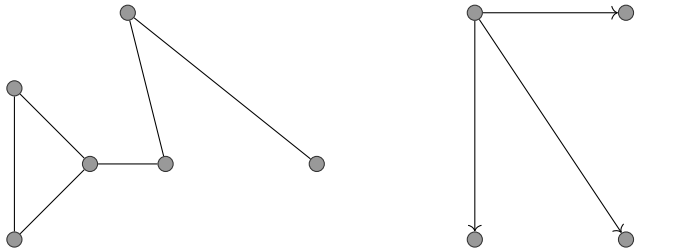
Since the sets H_k are nested, then from condition (2) we get that $B_1 \leq B_2 \leq \cdots \leq B_n \cdots$. With the supremum condition we are requiring that the tails of the distribution of the random variable $\mathbf{z}$ not be "heavy". In other words, that the large values of $\mathbf{z}$ do not necessarily have large probabilities.

The SRM principle chooses a function $P(\mathbf{z}, \alpha_k^*)$ that minimizes the empirical risk over the subset H_k. In essence the SRM principle looks to find the function $P(\mathbf{z}, \alpha)$, $\alpha \in \Lambda$, that simultaneously achieves a small empirical risk while being the least "complex".

To summarize, ERM minimizes the empirical risk of a cost functional to approximate the true risk. With SRM the goal is to find an optimal function that balances the sometimes conflicting requirements to limit the quantity of empirical data collected to achieve the highest quality of the approximation, and to limit the complexity measure of the set of functions from which the approximating function is drawn. In this thesis we use a structural risk minimization inference principle.

2.2 Graphs and Communication

This thesis casts the consensus estimation problem as one of distribution-free learning theory. Since much of the literature on estimation with consensus makes reference to elements of graph theory, this section will review the basics of graph theory. By definition a graph G consists of a set of vertices, $V(G) = \{v_1, v_2, \ldots, v_i \mid i \in \mathbb{N}\}$, a set of edges, $E(G) = \{e_1, e_2, \ldots, e_j \mid j \in \mathbb{N}\}$, and a relation that associates each edge with a pair of vertices. We denote the cardinality of the graph by $|V(G)|$. In our applications the vertices will represent vehicles or sensing agents and the edges correspond to communication channels. Graphs thereby describe agent communication and connectivity. Two examples of a graph are shown below, and they illustrate examples of the variability in communication topology that must be modeled.



(a) Simple Graph (b) Directed Graph

Figure 2.2.1: Simple Graph and Directed Graph

Graphs are represented as $G = (V, E)$ where the edges are defined by the pair of vertices, (i, j) they join. Generally, the vertices are numbered $1, \ldots, n$, and if the graph is a directed graph, as shown in Figure 2.2.1b, then (i, j) is interpreted as an ordered pair. In this case the first vertex is at the tail of the arrow and the second at the tip. If a graph is undirected and has no loops (an example of a loop is shown below), then it is called a simple graph. We are primarily interested in simple graphs.

Connectivity is an important characteristic of a graph, one that can have profound effect on consensus estimation. We want to know when a graph is connected. A graph is considered connected if for any two vertices $u, v \in V(G)$ there exists a path between them. In the case of a directed graph, it can be either weakly connected or strongly connected.

Definition 1. *A directed graph is* ***weakly connected*** *if it's underlying graph is connected.*

Definition 2. *A directed graph is* ***stronly connected*** *if for each ordered pair (u, v) of vertices, there is a path from u to v.*

For the graphs in Figure 2.2.1, the first is connected and the second weakly connected. The graph below, however, is not connected since there is no path between u, v, and w. Instead, this is a disconnected graph that consists of three connected components.

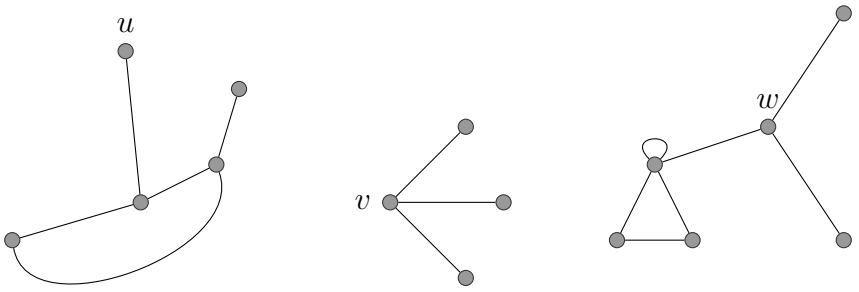


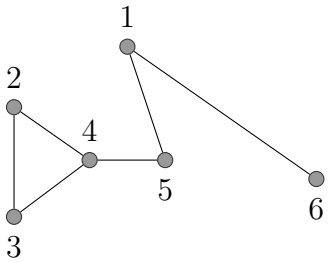
Figure 2.2.2: Unconnected Graph with Connected Components

A graph is completely determined by either its adjacencies or its incidences. The adjacencies can be represented in matrix form by the adjacency matrix. For the graph G with cardinality $|V(G)| = n$, this is the $n \times n$ matrix in which each entry a_{ij} is the number of edges between vertices i and j. The graphs we study will not have multiple edges between vertices and hence the adjacency matrix will be of the form

$$a_{ij} = \begin{cases} 1 & \text{if vertices } i \text{ and } j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

In the case of the simple graph, this matrix is real, symmetric, and has zeros on the diagonal. Its eigenvalues are non-negative, $\lambda_i \geq 0$.

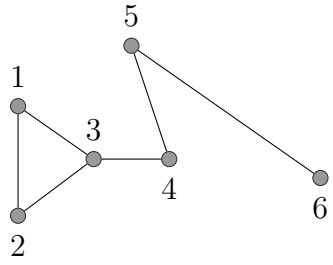
Example 2. *For the undirected graph in Figure 2.2.1 the adjacency matrix is given as shown below.*



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

It should be noted that if the labeling of the vertices is changed then a different adjacency matrix $\mathbf{A}'$ is generated. This, however, is not a problem since there exists some permutation matrix $\mathbf{P}$ such that $\mathbf{A} = \mathbf{P}^{-1}\mathbf{A}'\mathbf{P}$.

Example 3. *Switching the vertex labels on the graph above, we can calculate the new adjacency matrix,* $\mathbf{A}'$



$$\mathbf{A}' = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

We can construct the permutation matrix $\mathbf{P}$ *by associating to each row of the matrix a vertex* $1, 2, \ldots, 6$ *and then placing a solitary* 1 *into each of these row in the position where the column number corresponds to the number of the vertex that it was permuted to.*

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 4 & 5 & 1 & 6 \end{pmatrix} \quad \mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Then using the adjacency matrix from the original graph G*, and recalling that permutation matrices are orthogonal, in other words,* $\mathbf{P}^{-1} = \mathbf{P}^T$*,*

$$\mathbf{P}^T\mathbf{A}\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \mathbf{A}'$$

Many of the properties of a graph can be discerned from a study of its adjacency matrix. For example, a graph is connected if and only if there is no labeling of vertices such that the adjacency matrix can be reduced to the form

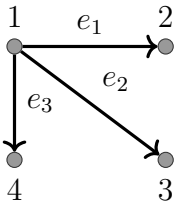
$$\mathbf{A} = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}$$

As stated above, the incidences also completely determine a graph G. This is clear by considering the edge incidence matrix. For G with $|V(G)| = n$ and $|E(G)| = m$ this is an

$m \times n$ matrix with entries

$$B_{ij} = \begin{cases} +1 & \text{if the vertex is at the tail of the arrow} \\ -1 & \text{if the vertex is at the tip of the arrow} \\ 0 & \text{otherwise} \end{cases}$$

Example 4. *From the directed graph in Figure 2.2.1 the incidence matrix would be*



$$\mathbf{B} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}$$

Looking at this matrix we see that each row has exactly one $+1$ entry and one -1 entry. If we calculate $\sum_k B_{ki}B_{jk}$, then for $i \neq j$ the only non-zero terms will occur when both B_{ki} and B_{jk} are non-zero. This occurs exactly when there is an edge between vertices i and j. This non-zero term will be -1. If $i = j$, then $\sum_k B_{ki}^2 = \deg(v_i)$. Therefore, $\sum_k B_{ki}B_{kj} = \mathbf{B}^T\mathbf{B}$, an $n \times n$ matrix, will have the degrees of the vertices along the diagonal, $b_{ij} = -1$ wherever there is an edge that join vertices i and j, and zero everywhere else. For the example above we would have,

$$\mathbf{B}^T\mathbf{B} = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}$$

Upon observation though, one can see that $\mathbf{B}^T\mathbf{B} = \mathbf{D} - \mathbf{A} = \mathbf{L}$ where $\mathbf{D}$ is the degree matrix of $d_{ii} = \deg(v_i)$ and zero everywhere else and $\mathbf{A}$ is our adjacency matrix. While $\mathbf{A}$ and $\mathbf{L}$ will be used later to represent completely different objects it should be clear from context to which we are referring.

We call $\mathbf{L}$ the Laplacian and it has many interesting properties that will be used for establishing consensus between the agents of a network. First notice that if we take a normalized eigenvector $\mathbf{v}_i$ of $\mathbf{L}$, then

$$(\mathbf{B}\mathbf{v}_i, \mathbf{B}\mathbf{v}_i) = \mathbf{v}_i^T\mathbf{L}\mathbf{v}_i = \lambda_i\mathbf{v}_i^T\mathbf{v}_i = \lambda_i$$

Since the inner product of a vector with itself is always positive, we have that $\lambda_i \geq 0$ for $i = 1, 2, \ldots, n$. Next, since for each row of $\mathbf{L}$ we have $l_{ii} = \deg(v)_i$ and $l_{ij} = -1$ whenever

vertex j is a neighbor of i, then it is clear that summing along the rows gives $\sum_{j}^{n} l_{ij} = 0$ for $i = 1, 2, \ldots, n$. We thus have $\mathbf{L} \bullet \mathbf{1} = 0 \cdot \mathbf{1}$, where $\mathbf{1} = (1, 1, \ldots, 1)^T$. In other words, $\mathbf{1}$ is an eigenvector of $\mathbf{L}$ with eigenvalue $\lambda = 0$. The Laplacian, therefore, is a rank deficient semi-positive definite matrix.

Now consider the real spectrum of $\mathbf{L}$, $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{|G|}$, listed with multiplicities. We can determine if the graph is connected by the value of the second eigenvalue. Take the case of the graph in Figure 2.2.2 without the loop. Then we could write a Laplacian for each connected component, and for the entire graph the Laplacian will have block the diagonal form,

$$\mathbf{L} = \begin{bmatrix} [L_1] & 0 & \cdots \\ 0 & [L_2] & \\ \vdots & & [L_3] \end{bmatrix}$$

We can now see that an eigenvector with $\lambda = 0$ can be constructed as $v_1 = (1, 1, \ldots, 1, 0, 0, \ldots)^T$ where $v_{1i} = 1$ for $i = 1, 2, \ldots, |L_1|$. This process can be repeated for each block, giving multiple eigenvectors with $\lambda = 0$. Therefore, $\lambda_2 = 0$ if the graph is unconnected and $\lambda_2 > 0$ if it is connected. Conversely, the multiplicity of $\lambda = 0$ reflects the number of connected components in G.

We next discuss how graph theoretic properties can be critical in describing consensus in agent networks. Our discussion considers a typical result, and the reader should consult [41] for general analysis. We can use the Laplacian to help define and determine whether there is agreement, in other words consensus, amongst the vertices. Assume that each vertex has a dynamic associated with it, given by

$$\dot{x}_i(t) = \sum_{j \in N(i)} (x_j(t) - x_i(t)) \tag{2.2.1}$$

where $j \in N(i)$ symbolizes that j is a neighbor of i. A plot of $\dot{x}(t)$ shows to what degree each vertex agrees with all its neighbors as t approaches infinity. The collection of agents can be said to reach consensus provided the vertex values (x_i, x_j) all have the identical value. If we want the network to achieve consensus, then we ideally want to see $\dot{\mathbf{x}}(t)$ approach zero. Looking closely at Equation 2.2.1, we can see that this equation can be rewritten in terms of the Laplacian as

$$\dot{\mathbf{x}}(t) = -\mathbf{L}\mathbf{x}(t) \tag{2.2.2}$$

We know from above that for a connected graph $\mathcal{N}(\mathbf{L}) \subseteq \mathbb{R}^n$, the nullspace of $\mathbf{L}$, is the subspace span$\{\mathbf{1}\}$. Therefore, if our trajectory $\mathbf{x}(t)$ converges to $\mathbf{1}$ then we will have agreement. Relabeling $\mathcal{N}(\mathbf{L})$ as the agreement set $\mathcal{O}$, where we have chosen the symbol $\mathcal{O}$ to emphasize that this space is the orthogonal complement of im$(\mathbf{L})$, we have the following,

Proposition 1. *([42]) The Laplacian dynamics in Equation (2.2.2) converges to the agreement subspace $\mathcal{O}$ from an arbitrary initial condition if and only if the underlying graph is connected.*

2.3 Learning Theory with Consensus

Again, in our problem we have n agents forming a connected graph moving in the domain X. Each agent i collects a set of data points $\{z_j^i\}_{j\in\mathbb{N}} = \{(x_j^i, y_j^i)\}_{j\in\mathbb{N}}$ with the distribution given by the probability measure, ρ^i. The fidelity of the estimator f^i generated by agent i is measured by the local cost functional.

$$\mathcal{J}^i(f^i) = \int_Z L(f^i(x), y)\rho^i(dz)$$

The overarching claim in this section is that Equation 2.3.1 is equivalent to Equation 2.3.6. In order to get there we must first make an assumption about the estimators f^i, $i = 1, \ldots, n$. Currently each f^i is picked from a unique hypothesis space H^i. For consensus, however, it is easiest if we assume that all the function spaces H^i, $i = 1, \ldots, n$ are based on the same set of possible functions. If we make this assumption then Equation 2.3.6 will easily follow from the discussion below. Later we will discuss how the problem statement differs when this assumption is not made.

2.3.1 Problem Statement in Infinite Dimensions

We collect the individual estimates into the vector $\mathbf{f} = \begin{bmatrix} f^1 & f^2 & \cdots & f^n \end{bmatrix}^\top \in H^1 \times H^2 \times \cdots \times H^n \equiv \mathcal{H}$ and restate our goal as that of finding the $\mathbf{f}^*$ that minimizes

$$\mathbf{f}^* = \underset{\mathbf{f}=\begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix} f}{\arg\min} = \mathcal{J}(\mathbf{f}) = \sum_{i=1}^{n} \mathcal{J}^i(f^i) \tag{2.3.1}$$

where $\mathbf{f}^* = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^\top \cdot f^* \in \mathcal{H}$ is the consensus estimate. More specifically, each agent collects a set of data points $\left\{(x_j^i, y_j^i)\right\}_{j=1:m_k}$ during stage or epoch k after which it builds its local estimate of the functional dependency. Each agent will perform this for k epochs, building a sequence of local approximations, $\{f_k^i\}_{k\in\mathbb{N}}$. The index m_k is the number of data points collected before the k^{th} approximation f_k^i is built by agent i. However, we want consensus amongst the agents. Therefore, not only do we require that the estimates give good approximations of local measurements, but also that the agents converge to a common estimate $\mathbf{f}^*$. We want

$$f_k^i \to f^*$$

as $j \to \infty$. Therefore, finding $\mathbf{f}^*$ that minimizes $\mathcal{J}(\cdot)$ with consensus is equivalent to,

$$\mathbf{f}^* = \underset{\substack{\mathbf{f}\in\mathcal{H} \\ \mathbf{f}=\begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix}}}{\arg\min}\, \mathcal{J}(\mathbf{f}), \quad \mathbf{f}^* = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} f^* \tag{2.3.2}$$

We next deduce some alternative representations of the above constrained minimization problem that is amenable to decentralized solution procedures. To address consensus we consider the topology of the communication graph of our agents. The agents make their own approximations f^i for $i = 1, 2, \ldots, n$ of an external field. After some period of time each updates their neighbors with information regarding their local approximation f^i. In order to solve our minimization problem with consensus we need that the approximations of the neighbors of the i^{th} agent, $f^{n(i)}$, approaches f^i. Recall, that we already saw this constraint in the discussion of the Laplacian of a graph. Generally we have $\text{span}\{\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^\top\} \subseteq \ker(\mathbf{L})$. For a connected graph $\text{span}\{\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^\top\} \equiv \ker(\mathbf{L})$. Therefore, to enforce the consensus condition on our agents we construct a constraint matrix $\mathbf{M}^B$ from the Laplacian of the communication graph G.

As described in the graph theory section of this thesis, the Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$. It is also true that the kernel of $\mathbf{L}$ is the span $\left\{\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^{\mathrm{T}}\right\}$. Hence from the rank-nullity theorem we know that $\mathbf{L}$ is rank deficient with one redundant row (recalling that row rank and column rank are equivalent). We will see later in this thesis that we need to have the constraint matrix be full rank. Therefore, we want to identify the redundant row from $\mathbf{L}$, which we will do via singular value decomposition (SVD).

Recalling from linear algebra, the SVD decomposes a matrix into the product $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}} \in \mathbb{R}^{m\times n}$, where $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\Sigma \in \mathbb{R}^{m\times n}$ is a diagonal matrix. The matrices $\mathbf{U}$ and $\mathbf{V}$ can be partitioned into columns in the form,

$$\mathbf{U} = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_r & \mathbf{u}_{r+1} & \cdots & \mathbf{u}_m\end{bmatrix} = \begin{bmatrix}U_1|U_2\end{bmatrix} \tag{2.3.3}$$

$$\mathbf{V} = \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_r & \mathbf{v}_{r+1} & \cdots & \mathbf{v}_n\end{bmatrix} = \begin{bmatrix}V_1|V_2\end{bmatrix} \tag{2.3.4}$$

For a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with rank r, the first r columns of Σ are nonzero and the remaining $m - r$ columns are zero. From this decomposition a great deal can be deduced about the matrix $\mathbf{A}$. We have that $\text{im}(\mathbf{A}) = \text{span}\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r\} = \text{span}\{U_1\}$ and $\ker(\mathbf{A}) = \text{span}\{\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n\} = \text{span}\{V_2\}$. In particular, the column vectors V_1 of $\mathbf{V}$ span the row space of $\mathbf{A}$, $\text{im}(\mathbf{A}^{\mathrm{T}}) = \text{span}\{V_1\}$. We can use this technique to select the independent row vectors of $\mathbf{L}$, giving the full rank matrix we require.

We can now state the conditions that the constraint matrix must satisfy:

1. $\mathbf{M}^B \in \mathbb{R}^{n-1} \times \mathbb{R}^n$
2. $\mathbf{M}^B$ must be full rank: $n - 1$ rows must be linearly independent
3. $\mathbf{M}^B \cdot \begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} = \mathbf{0}$

And since $\mathbf{f}^* = \begin{bmatrix}1 & 1 & \dots & 1\end{bmatrix}^\top \cdot f^*$, our third requirement states that $\mathbf{M}^B\mathbf{f}^* = \mathbf{0}$.

Example 5. *Consider the simple communication graph below of a set of agents numbered* $1-5$*, left to right*

We may require that agent 1 *agrees with* 2*,* 2 *with* 3*, etc. For this simple example we can solve this using either simple observation or the SVD method. We will demonstrate both.*

For the SVD we first construct our Laplacian as,

$$\underbrace{\begin{bmatrix}1&0&0&0&0\\0&2&0&0&0\\0&0&2&0&0\\0&0&0&2&0\\0&0&0&0&1\end{bmatrix}}_{\mathbf{D}} - \underbrace{\begin{bmatrix}0&1&0&0&0\\1&0&1&0&0\\0&1&0&1&0\\0&0&1&0&1\\0&0&0&1&0\end{bmatrix}}_{\mathbf{A}} = \underbrace{\begin{bmatrix}1&-1&0&0&0\\-1&2&-1&0&0\\0&-1&2&-1&0\\0&0&-1&2&-1\\0&0&0&-1&1\end{bmatrix}}_{\mathbf{L}}$$

We are interested in the independent rows of $\mathbf{L}$*, in other words the* $\mathrm{im}(\mathbf{L}^\top)$*. From SVD of* $\mathbf{L}$ *we take the first* r *rows of* $V^\top$ *where we can determine* r *from the rank of* Σ*. We construct* V *by finding the orthonormal vectors of* $\mathbf{L}^\top\mathbf{L}$ *and concatenating them into* V *as shown in Equation 2.3.4. Using the Matlab function* `svd` *we get,*

$$\mathrm{rank}(\Sigma) = 4$$

$$\mathbf{V} = \begin{bmatrix}-0.1954 & -0.3717 & -0.5117 & -0.6015 & -0.4472\\ 0.5117 & 0.6015 & 0.1954 & -0.3717 & -0.4472\\ -0.6325 & 0.0000 & 0.6325 & 0 & -0.4472\\ 0.5117 & -0.6015 & 0.1954 & 0.3717 & -0.4472\\ -0.1954 & 0.3717 & -0.5117 & 0.6015 & -0.4472\end{bmatrix}$$

Taking the first four columns of V *we transpose them to construct a full-rank constraint matrix* $\mathbf{M}^B \in \mathbb{R}^4 \times \mathbb{R}^5$ *that clearly satisfies condition 3 above,*

$$\mathbf{M}^B\begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}-0.1954 & 0.5117 & -0.6325 & 0.5117 & -0.1954\\ -0.3717 & 0.6015 & 0.0000 & -0.6015 & 0.3717\\ -0.5117 & 0.1954 & 0.6325 & 0.1954 & -0.5117\\ -0.6015 & -0.3717 & 0.0000 & 0.3717 & 0.6015\end{bmatrix}\begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}$$

For the second method, we can construct a possible constraint matrix from the simple observation that if agent i *is to agree with agent* $i+1$ *then we must have* $x_i - x_{i+1} = 0$*.*

From this we can easily construct $\mathbf{M}^B \in \mathbb{R}^4 \times \mathbb{R}^5$ *so that it satisfies all our conditions,*

$$\mathbf{M}^B \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

With more complicated graph topologies we cannot easily construct the constraint matrix from observation. Instead we will have to construct the graph Laplacian and then use SVD.

Incorporating the constraint matrix into our problem statement we can refine it a little more. We now state that we want to find $\mathbf{f}^*$ such that,

$$\mathbf{f}^* = \underset{\substack{\mathbf{f} \in \mathcal{H} \\ \mathbf{M}^B \mathbf{f} = \mathbf{0}}}{\arg\min} \ \mathcal{J}(\mathbf{f}) \tag{2.3.5}$$

where $\mathcal{H} = H^1 \times H^2 \times \cdots \times H^n$ and H^i is the space of admissible functions for the estimate f^i of agent i. This formulation can be refined further by noting that the matrix $\mathbf{M}^B$ induces via multiplication a constraint operator $B : \mathcal{H} \to \mathcal{Q}$, where $\mathcal{Q} := \{\mathbf{g} = \mathbf{M}^B \cdot \mathbf{f} : \mathbf{g} \text{ is } (n-1) \times 1 \text{ vector }, \mathbf{f} \in \mathcal{H}\}$ and $\mathcal{Q} = \underbrace{Q \times Q \times \cdots \times Q}_{(n-1)}$. And the constraint is now written as

$$(B\mathbf{f})(x) = \mathbf{M}^B \mathbf{f}(x)$$

For the operator form of our problem statement we have that we want to find $\mathbf{f}^*$ such that

$$\mathbf{f}^* = \underset{\substack{\mathbf{f} \in \mathcal{H} \\ B\mathbf{f} = \mathbf{0}}}{\arg\min} \ \mathcal{J}(\mathbf{f}) \tag{2.3.6}$$

We now have the most general form of the problem statement which we will show later makes sense even if the function spaces H^i are different. So we see that the agreement space $\mathcal{O}$ mentioned in Section 1.3 can be characterized as either $\mathcal{N}(B)$ or as Range $\left(\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} f \right)$ for $f \in H$.

2.3.2 Formulation as a Saddle Point Problem

Up until now the specific form of the loss function has remained undefined. That is we have not selected the kernel $J(\cdot, \cdot)$. In the learning problem section we briefly described the three main learning applications. Each one selects a different form for the functional $\mathcal{J}$ and the

kernel J. Recall that we are looking at the regression application and we choose to take the structural risk minimization approach. In SRM the quadratic loss function is most commonly used. This is not by accident. It is a well studied function for which there exists a large body of theory. Luenberger's classical text [43] on optimization details the theory behind recasting this constrained optimization problem as an equivalent saddle point problem. Once we have our problem in this form we can then apply the well-known Babuska-Brezzi theory for constructing approximations. To begin this process, we start by to looking at the case for a single agent.

The quadratic loss functional for the single agent is

$$\mathcal{J}(f) = \frac{1}{2}\int_Z (f(x) - y)^2 \underbrace{\rho(dz)}_{\rho(dy|x)\rho_X(dx)}$$

where $\rho_X(\cdot)$ is the marginal measure on X and $\rho(\cdot|x)$ is the conditional probability measure on Y given $x \in X$.

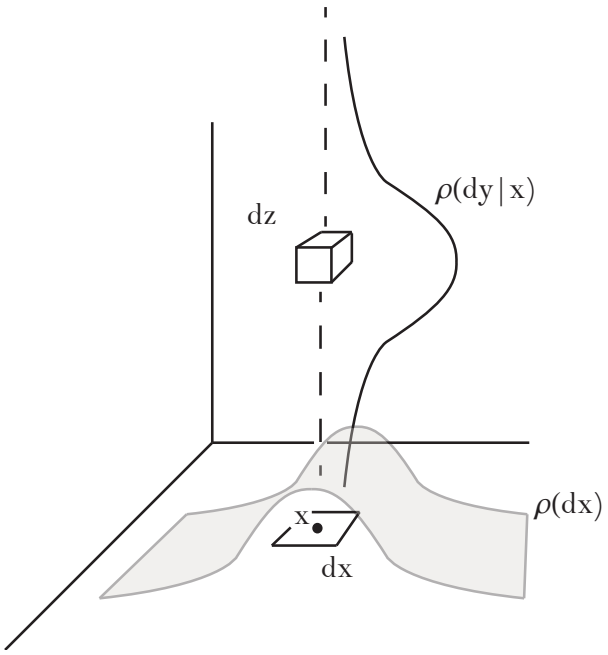


Figure 2.3.1: Probability Space

Rewriting our functional with this decomposition of the measure ρ, we have,

$$\begin{aligned}
\mathcal{J}(f) &= \frac{1}{2}\int_Z (f^2(x) - 2yf(x) + y^2)\rho(dy|x)\rho_X(dx) \\
&= \frac{1}{2}\Bigg\{ \int_X f^2(x)\cancel{\Big(\int_Y \rho(dy|x)\Big)}^{1}\rho_X(dx) - 2\int_X f(x)\underbrace{\Big(\int_Y y\rho(dy|x)\Big)}_{f_\rho(x)}\rho_X(dx) \\
&\quad + \int_X\int_Y y^2\rho(dy|x)\rho_X(dx)\Bigg\} \\
&= \frac{1}{2}\Bigg\{ \int_X f^2(x)\rho_X(dx) - 2\int_X f(x)f_\rho(x)\rho_X(dx) + \int_X\int_Y y^2\rho(dy|x)\rho_X(dx)\Bigg\}
\end{aligned}$$

The function $f_\rho(x)$ is called the regressor function. By adding and subtracting $\frac{1}{2}\int_X f_\rho^2(x)\rho_X(dx)$ to the above, we get,

$$\mathcal{J}(f) = \frac{1}{2}\int_X (f(x) - f_\rho(x))^2\rho_X(dx) + \underbrace{\frac{1}{2}\Bigg\{ -\int_X f_\rho^2(x)\rho_X(dx) + \int_X\int_Y y^2\rho(dy|x)\rho_X(dx)\Bigg\}}_{\mathcal{J}(f_\rho)}$$

$$\mathcal{J}(f) = \frac{1}{2}\int_X (f(x) - f_\rho(x))^2\rho_X(dx) + \mathcal{J}(f_\rho) \tag{2.3.7}$$

It is now clear that minimizing the functional $\mathcal{J}(\cdot)$ is equivalent to finding a good approximation of the regressor f_ρ.

If we take our function space $H = \mathsf{L}^2_{\rho_X}$, then Equation 2.3.7 can be put into the following quadratic form,

$$\begin{aligned}
\frac{1}{2}\parallel f(x) - f_\rho(x)\parallel^2_{\mathsf{L}^2_{\rho_X}} + \mathcal{J}(f_\rho) &= \frac{1}{2}(f - f_\rho, f - f_\rho)_H + \mathcal{J}(f_\rho) \\
&= \frac{1}{2}(f, f)_H - (f_\rho, f)_H + c
\end{aligned} \tag{2.3.8}$$

where $c = \mathcal{J}(f_\rho) + \frac{1}{2}||f_\rho||^2_H$ and $(\cdot,\cdot)_H$ is the inner product in the hypothesis space H.

We now return to our case where we have multiple agents. To accommodate the minimization process occurring at each node of the connected graph, we define a function space, $H^i = \mathsf{L}^2(\rho^i_X)$ for each agent. Returning to Equation 2.3.1 we use the fact that $(\mathbf{a}, \mathbf{b}) = \sum_i a_1 b_1 +$

$a_2b_2 + \cdots + a_nb_n$ and write the loss functional $\mathcal{J}(\mathbf{f})$ in the following quadratic form,

$$\begin{aligned}\mathcal{J}(\mathbf{f}) = \sum_{i=1}^{n} \mathcal{J}^i(f)^i &= \frac{1}{2}\left(\begin{bmatrix} I_{H^1} & & & \\ & I_{H^2} & & \\ & & \ddots & \\ & & & I_{H^n} \end{bmatrix} \begin{bmatrix} f^1 \\ f^2 \\ \vdots \\ f^n \end{bmatrix}, \begin{bmatrix} f^1 \\ f^2 \\ \vdots \\ f^n \end{bmatrix} \right)_{\mathcal{H}} - \left(\begin{bmatrix} f_{\rho^1} \\ f_{\rho^2} \\ \vdots \\ f_{\rho^n} \end{bmatrix}, \begin{bmatrix} f^1 \\ f^2 \\ \vdots \\ f^n \end{bmatrix} \right)_{\mathcal{H}} + \tilde{c} \\ &= \frac{1}{2}\big(\mathbf{I}_{\mathcal{H}}\mathbf{f}, \mathbf{f}\big)_{\mathcal{H}} - \big(\mathbf{f}_\rho, \mathbf{f}\big)_{\mathcal{H}} + \tilde{c}\end{aligned}$$

where $\tilde{c}$ is the constant that incorporates $c^i = \mathcal{J}^i(f_{\rho^i}) + \frac{1}{2}||f_{\rho^i}||^2_{H^i}$, $\tilde{c} = \sum_{i=1} c^i$, and I_{H^i} is the identity operator on H^i. It will be convenient to view this quadratic form as a special case of a more general expression. Define the linear operator $A : \mathcal{H} \to \mathcal{H}$ as

$$A = \begin{bmatrix} A^1 & 0 & 0 & \cdots & 0 \\ 0 & A^2 & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & A^n \end{bmatrix} := [A_{ik}]\,. \tag{2.3.9}$$

Hence the general form of the quadratic representation of our problem is,

$$\mathcal{J}(\mathbf{f}) = (A\mathbf{f}, \mathbf{f})_{\mathcal{H}} - (\boldsymbol{\ell}, \mathbf{f})_{\mathcal{H}} + \tilde{c} \tag{2.3.10}$$

where $\boldsymbol{\ell} = \begin{bmatrix} \ell^1 & \ell^2 & \cdots & \ell^{n-1} \end{bmatrix}$, $\ell^i = f_{\rho^i}$, the unknown regressor. It is evident that if we choose $A^i := I_{H^i}$, the general quadratic form reduces to our specific problem.

It is well-known that if certain convexity conditions hold for the functional $\mathcal{J}$ then the constrained optimization problem

$$\mathbf{f}^* = \underset{\substack{\mathbf{f}\in\mathcal{H} \\ B\mathbf{f}=0}}{\arg\min}\, \mathcal{J}(\mathbf{f})$$

can be equivalently written as a saddle point problem by introducing the *Lagranian functional* $\mathcal{L}$.

$$\mathcal{L} : \mathcal{H} \times \mathcal{Q} \to \mathbb{R}$$

$$\mathcal{L}(\mathbf{f}, \mathbf{p}) := \mathcal{J}(\mathbf{f}) + (\mathbf{p}, B\mathbf{f})_{\mathcal{Q}}$$

See Luenberger in [43], for example, for a detailed analysis.

The functional $\mathcal{L}(\cdot, \cdot)$ is used to define the saddle point problem that seeks the pair $(\mathbf{f}^*, \mathbf{p}^*) \in \mathcal{H} \times \mathcal{Q}$ that solve the minmax problem,

$$\mathcal{L}(\mathbf{f}^*, \mathbf{p}^*) = \inf_{\mathbf{p}\in\mathcal{Q}} \sup_{\mathbf{f}\in\mathcal{H}} \mathcal{L}(\mathbf{f}, \mathbf{p}) \tag{2.3.11}$$

The solution of the saddle point problem is characterized by the saddle point equations. In these equations we seek $(\mathbf{f}^*, \mathbf{p}^*) \in \mathcal{H} \times \mathcal{Q}$ such that

$$\begin{bmatrix} A & B^* \\ B & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_\rho \\ \mathbf{0} \end{bmatrix} \tag{2.3.12}$$

Specifically, for our case, we have

$$\begin{bmatrix} \begin{bmatrix} I_{H^1} & & \\ & \ddots & \\ & & I_{H^n} \end{bmatrix} & \left[\mathbf{M}^B\right]^T \\ \left[\mathbf{M}^B\right] & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_\rho \\ \mathbf{0} \end{bmatrix}$$

Note that I_{H^i} is the infinite dimensional identity operator on H^i.

In the construction above we choose the hypothesis spaces for each agent as $H^i = \mathsf{L}^2(\rho_X^i)$.

2.3.3 Norm Equivalence

The discussion above was predicated on each space containing the same set of functions. In our formulation of the problem statement we make an assumption that is essential to guarantee existence and uniqueness of solutions. We assume that the norm $\|\cdot\|_{H^i}$ that we put on our space H^i must be equivalent to the norm $\|\cdot\|_Q$ that we put on the constraint space Q. This assumption has two consequences on our derivation of our algorithms presented in Section 3.2. First, it puts restrictions on the sampling measures that we can use, and second, it allows us to use a common basis on the inner product spaces $\mathcal{H}$ and $\mathcal{Q}$.

We begin with the definition of norm equivalence.

Definition 3. *Let $\|\cdot\|_a$ and $\|\cdot\|_b$ be norms defined on the vector space X. These norms are equivalent provided that there exist real numbers $0 < \underline{c} \leq \bar{c}$ such that for all $x \in X$ the following inequality holds,*

$$\underline{c}\|x\|_b \leq \|x\|_a \leq \bar{c}\|x\|_b$$

For our infinite dimensional functional spaces H^i and Q we have chosen these to be weighted Lebesgue space with the standard 2-norm. Therefore, the requirement of norm equivalence implies that for any f the measures m and ρ we put on our functional spaces must satisfies the following condition,

$$\underline{c} \int |f|^2 \rho_X(dx) \leq \int |f|^2 m(dx) \leq \bar{c} \int |f|^2 \rho(dx)$$

One can view this condition as requiring that the sampling measure is spread throughout our space. Imagine the situation in Figure 2.3.2 where the box represents the domain over

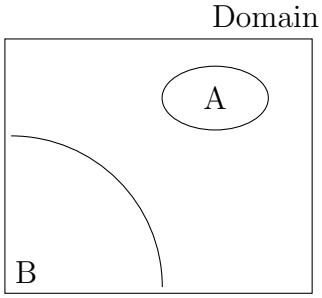


Figure 2.3.2: Disjoint Support of Measures ρ_X and m

which functions are defined and the areas A and B represent regions over which measure m and the measure ρ_X are supported, respectively.

The condition of norm equivalence makes this situation impossible. Choose any positive function f that is supported only on the set A. If m is the support over region A and ρ over region B, then by our norm equivalence condition we would have

$$\underline{c} \underbrace{\int |f|^2 \rho_X(dx)}_{=0} \leq \underbrace{\int |f|^2 m(dx)}_{\neq 0} \leq \bar{c} \underbrace{\int |f|^2 \rho_X(dx)}_{=0}$$

which is impossible. Therefore, it must be that the different measures we apply do not have disjoint supports.

Example 6. *Consider the case where we define our measure,*

$$\rho_X^i(dx) = p^i(x) m(dx) \tag{2.3.13}$$

where $p^i(x)$ is a density function over the space X, $m(dx)$ is the Lebesgue measure, and for each i,

$$0 < \underline{c}^i \leq p^i(x) \leq \bar{c}^i \tag{2.3.14}$$

We can easily show Equations 2.3.13 and 2.3.14 imply norm equivalence. Using these equations and the definition of the 2-norm on the space $\mathsf{L}^2(\rho_X^i)$ we have that

$$\int |f|^2 \rho_X^i(dx) = \int |f|^2 p^i(x) m(dx) \leq \bar{c}^i \int |f|^2 m(dx)$$

and from below,

$$\int |f|^2 \rho_X^i(dx) = \int |f|^2 p^i(x) m(dx) \geq \underline{c}^i \int |f|^2 m(dx)$$

We thus have the condition of norm equivalence,

$$\underline{c}^i \int |f|^2 m(dx) \leq \int |f|^2 \rho_X^i(dx) \leq \bar{c}^i \int |f|^2 m(dx)$$

It is this norm equivalence requirement that restricts the sampling measures that are admissible for our formulation of consensus.

2.3.4 Finite Dimensional Approximation

In order to move from the infinite dimensional operator equations that describe the saddle point formulation to a finite dimensional form we need to find the matrix representations. We begin by defining Equation 2.3.12 in terms of specific bases. We assume that we are given a basis for the hypothesis space of each agent, H^i, $\{\phi_j^{H^i}\}_{j\in\mathbb{N}}$, and one for Q, $\{\phi_j^Q\}_{j\in\mathbb{N}}$. We know that we can represent each element $f^i \in H^i$ and $p^i \in Q$ as

$$f^i = \sum_j \phi_j^{H^i} F_j^i \tag{2.3.15a}$$

$$p^i = \sum_j \phi_j^Q P_j^i \tag{2.3.15b}$$

where F_j and P_j are the weights (coefficients) that multiply the j^{th} basis element. From Equation 2.3.12 let $\mathbf{g} = A\mathbf{f}$, where $\mathbf{g}, \mathbf{f} \in \mathcal{H}$ and A is an operator defined as in Equation 2.3.9. Then for agent i, we can write its expansion as

$$\begin{aligned} g^i = \sum_j \phi_j^{H^i} G_j^i = \sum_k A_{ik} f^i = \sum_k A_{ik} \sum_j \phi_j^{H^i} F_j^i \\ \sum_j \phi_j^{H^i} G_j^i = \sum_k \sum_j A_{ik} \phi_j^{H^i} F_j^i \end{aligned} \tag{2.3.16}$$

where we define the vectors of coefficients as $\mathbf{G}^i := \begin{bmatrix} G_1^i & G_2^i & \cdots \end{bmatrix}^\top$ and $\mathbf{F}^i := \begin{bmatrix} F_1^i & F_2^i & \cdots \end{bmatrix}^\top$ associated with the i^{th} agent. For our basis of H^i we could assume that it is orthonormal. If our basis was not orthonormal we could orthonormalize it by use of the Gram-Schmidt process. However, in the distribution free learning problem we do not know the measure ρ^i. An orthonormal basis for $H^i = \mathsf{L}^2(\rho_X^i)$ is useful only if we know ρ^i and can compute the inner product on H^i. In this thesis we simply choose a basis without enforcing orthogonality. Picking an arbitrary basis element $\phi_r^{H^i}$ from H^i we take the inner product between it and Equation 2.3.16

$$\left(\phi_r^{H^i}, \sum_j \phi_j^{H^i} G_j^i \right)_{H^i} = \left(\phi_r^{H^i}, \sum_{j,k} A_{ik} \phi_j^{H^k} F_j^k \right)_{H^i}$$

Using norm properties,

$$\sum_j \left(\phi_r^{H^i}, \phi_j^{H^i}\right)_{H^i} G_j^i = \sum_{j,k} \left(\phi_r^{H^i}, A_{ik}\phi_j^{H^k}\right)_{H^i} F_j^k \tag{2.3.17}$$

The inner products on both sides of the equation induce a simple multiplication of a real-valued matrix by a real valued vector. To see this fix each r and then sum over j. On the left-hand side this gives the matrix $\mathbf{M}^{H^i}$ times the vector $\mathbf{G}^i$, where $\mathbf{G}^i = \begin{bmatrix} G_1^i & \cdots & G_j^i & \cdots \end{bmatrix}^\top$. On the right-hand side this gives the matrix $\mathbf{A}_{ik(rj)}$ where we define its entries

$$\mathbf{A}_{ik(rj)} := \left(\phi_r^{H^i}, A_{ik}\phi_j^{H^k}\right)_{H^i}$$

Summing over j leaves us with

$$\mathbf{M}^{H^i}\mathbf{G}^i = \sum_k \mathbf{A}_{ik(rj)}\mathbf{F}^k$$

If we had chosen an orthonormal basis $\mathbf{M}^{H^i}$ would simply be the identity matrix. Our matrix $\mathbf{M}^{H^i}$ is known as the Gramian matrix, or simply Gramian. We construct a Gramian by taking a set of vectors, $\{v_1, v_2, \ldots, v_k\}$, and concatenating them as column vectors forming the matrix $V = \begin{bmatrix} v_1 & v_2 & \cdots & v_k \end{bmatrix}$. Then the Gramian is equal to the integral of the product $V^\top V$. From linear algebra we know that $\det(V^\top V) = (\det(V))^2 \neq 0$ if the vectors in the set $\{v_1, v_2, \ldots, v_k\}$ are linearly independent. Since our vectors are basis vectors we know the condition is satisfied hence $\mathbf{M}^{H^i}$ is invertible. Finally, we premultiply by the inverse of the Gramian and for each fixed i, we sum over k giving,

$$\underset{\sim}{\mathbf{G}} = (\mathbf{M}^{\mathcal{H}})^{-1}\mathbf{A}\underset{\sim}{\mathbf{F}} \tag{2.3.18}$$

where $\underset{\sim}{\mathbf{G}}$ and $\underset{\sim}{\mathbf{F}}$ are constructed by stacking the vectors $\mathbf{G}^i$ and $\mathbf{F}^i$, respectively. More explicitly,

$$\underset{\sim}{\mathbf{G}} = \begin{bmatrix} \mathbf{M}^{H^1} & 0 & \cdots & 0 \\ 0 & \mathbf{M}^{H^2} & \cdots & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{M}^{H^n} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \cdots & \mathbf{A}_{1n} \\ \mathbf{A}_{21} & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{n1} & \cdots & \cdots & \mathbf{A}_{nm} \end{bmatrix} \underset{\sim}{\mathbf{F}} = (\mathbf{M}^{\mathcal{H}})^{-1}\mathbf{A}\underset{\sim}{\mathbf{F}}$$

Now suppose that $\mathbf{p} = B\mathbf{f}$ where $\mathbf{p} \in \mathcal{Q}$ is an $(n-1) \times 1$ vector of functions and B is an operator. We can infer the matrix representation of this equation as we did with $\mathbf{g} = A\mathbf{f}$. For some i, $p^i = B_{i1}f^1 + B_{i2}f^2 + \cdots + B_{in}f^n$ and in the basis representation,

$$p^i = \sum_j \phi_j^Q P_j^i = \sum_m B_{im}f^m = \sum_{j,m} B_{im}\phi_j^{H^m} F_j^m = \sum_m \mathbf{M}_{jm}^B f^m$$

We take the inner product of this equation as we did with Equation 2.3.16 but we do so in Q with a basis function ϕ_r^Q.

$$\left(\phi_r^Q, \sum_j \phi_j^Q P_j^i\right)_Q = \left(\phi_r^Q, \sum_{j,m} B_{im}\phi_j^{H^m} F_j^m\right)_Q$$
$$\sum_j \left(\phi_r^Q, \phi_j^Q\right)_Q P_j^i = \sum_{j,m} \left(\phi_r^Q, B_{im}\phi_j^{H^m}\right)_Q F_j^m$$
$$\mathbf{P}^i = \sum_m \mathbf{B}_{im(rj)}\mathbf{F}_m$$

We have defined the matrix $\mathbf{B}_{im(rj)}$ as,

$$\mathbf{B}_{im(rj)} := \left(\phi_r^Q, B_{im}\phi_j^{H^m}\right)_Q \tag{2.3.19}$$

We now create our matrix representation of $\mathbf{p} = B\mathbf{f}$ by defining the matrix $\mathbf{B}$ in terms of submatrices $\mathbf{B}_{im(rj)}$. Holding i fixed and summing over m, we get the representation,

$$\underset{\sim}{\mathbf{P}} = (\mathbf{M}^{Q^n})^{-1}\mathbf{B}\underset{\sim}{\mathbf{F}}, \tag{2.3.20}$$

where again the under tilde on the bold letters represents a vector whose components are the vectors $\mathbf{P}^i$ and $\mathbf{F}^i$. More explicitly,

$$\underset{\sim}{\mathbf{P}} = \begin{bmatrix} \mathbf{M}^{Q^n 1} & 0 & \cdots & 0 \\ 0 & \mathbf{M}^{Q^n 2} & \cdots & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{M}^{Q^n n} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \cdots & \mathbf{B}_{1n} \\ \mathbf{B}_{21} & \ddots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ \mathbf{B}_{(n-1)1} & \cdots & \cdots & \mathbf{B}_{(n-1)n} \end{bmatrix} \underset{\sim}{\mathbf{F}} = (\mathbf{M}^{Q^n})^{-1}\mathbf{B}\underset{\sim}{\mathbf{F}}$$

Next, we expand $\mathbf{f} = B^*\mathbf{p}$ into a matrix representation. We follow the same procedure as above. For agent i,

$$f^i = \sum_j \phi_j^{H^i} F_j^i = \sum_k B_{ik}^* p^k = \sum_k B_{ik}^* \sum_j \phi_j^Q P_j^k$$
$$\sum_j \phi_j^{H^i} F_j^i = \sum_{j,k} B_{ik}^* \phi_j^Q P_j^k \tag{2.3.21}$$

Take the inner product over H^i with an arbitrary basis function $\phi_r^{H^i}$,

$$\sum_j \left(\phi_r^{H^i}, \phi_j^{H^i}\right)_{H^i} F_j^i = \sum_{j,k} \left(\phi_r^{H^i}, B_{ik}^*\phi_j^Q\right)_{H^i} P_j^k \tag{2.3.22}$$

For a fixed r and summation over j, we define the following matrix

$$\mathbf{B^*}_{ik(rj)} := \left(\phi_r^{H^i}, B_{ik}^*\phi_j^Q\right)_{H^i}$$

A note about $\mathbf{B}^*{}_{ik(rj)}$, this is a matrix, not an adjoint operator. Again, we denote operators with plain upper case letters and matrices with bold upper case letters. The asterisk is placed on this matrix as a reminder that within the entries of the matrix the adjoint operator B^*_{ik} appears versus the operator B_{ik} in the matrix Equation 2.3.19. In fact, we have a simple relationship between $\mathbf{B}^*{}_{ik(rj)}$ and $\mathbf{B}_{ik(rj)}$. If we define $\mathbf{C} = \mathbf{B}_{ik(rj)}$ and $\mathbf{D} = \mathbf{B}^*{}_{ki(jr)}$, then from the definition of the adjoint operator given in Appendix A,

$$[C_{rj}] = (\phi_r, B_{ik}\phi_j) = ((B_{ik})^*\phi_r, \phi_j) = (B^*_{ki}\phi_r, \phi_j) = [D_{jr}]$$
$$[C_{rj}] = [D_{jr}]$$

Since this holds for all r, j, then,

$$\mathbf{C}^{\mathrm{T}} = \mathbf{D}$$
$$\mathbf{B}^*_{ki(jr)} = (\mathbf{B}_{ik(rj)})^\top = \mathbf{B}^\top_{ik(jr)}$$

Returning to Equation 2.3.22 and using the identity above we have,

$$\mathbf{M}^{H^i}\mathbf{F}^i = \sum_k \mathbf{B}^\top_{ik(jr)}\mathbf{P}^k$$

Fixing i and summing over k, and taking note that $\mathbf{M}^{H^i}$ is another Gramian matrix we get,

$$\mathbf{M}^{\mathcal{H}}\underset{\sim}{\mathbf{F}} = \mathbf{B}^\top\underset{\sim}{\mathbf{P}}$$
$$\underset{\sim}{\mathbf{F}} = (\mathbf{M}^{\mathcal{H}})^{-1}\mathbf{B}^\top\underset{\sim}{\mathbf{P}} \tag{2.3.23}$$

or

$$\underset{\approx}{\mathbf{F}} = \begin{bmatrix} \mathbf{M}^{H^1} & 0 & \cdots & 0 \\ 0 & \mathbf{M}^{H^2} & \cdots & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{M}^{H^n} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}^\top_{11} & \mathbf{B}^\top_{12} & \cdots & \mathbf{B}^\top_{1(n-1)} \\ \mathbf{B}^\top_{21} & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}^\top_{n1} & \cdots & \cdots & \mathbf{B}^\top_{n(n-1)} \end{bmatrix} \underset{\approx}{\mathbf{P}} = (\mathbf{M}^{\mathcal{H}})^{-1}\mathbf{B}^\top\underset{\approx}{\mathbf{P}}$$

The last part of creating a complete matrix representation involves the regressors that we represent by $\boldsymbol{\ell} = \begin{bmatrix} \ell^1 & \ell^2 & \cdots & \ell^{n-1} \end{bmatrix}$, $\ell^i = f_{\rho^i}$. Recall by its form that the regressors reside in the hypothesis space H and hence,

$$\ell^i := f_{\rho^i} = \sum_j \phi^{H^i}_j \hat{L}^i_j \tag{2.3.24}$$

As before we take the inner product of our expansion with an arbitrary basis function $\phi^{H^i}_r$.

$$\left(\phi^{H^i}_r, f_{\rho^i}\right)_{H^i} = \left(\phi^{H^i}_r, \sum_j \phi^{H^i}_j \hat{L}^i_j\right)_{H^i} = \sum_j \left(\phi^{H^i}_r, \phi^{H^i}_j\right)_{H^i} \hat{L}^i_j$$

We define the coefficients of our regressor,

$$\mathbf{L}^i := \left(\phi_r^{H^i}, f_{\rho^i}\right)_{H^i}$$

and define $\mathbf{M}^{H^i}$ as previously. The matrix form of Equation 2.3.24 is,

$$\hat{\mathbf{L}}^i = (\mathbf{M}^{H^i})^{-1}\mathbf{L}^i$$

or

$$\underset{\sim}{\hat{\mathbf{L}}} = (\mathbf{M}^{\mathcal{H}})^{-1}\underset{\sim}{\mathbf{L}}$$

where $\underset{\sim}{\hat{\mathbf{L}}} = \begin{bmatrix}\hat{\mathbf{L}}^{1,\top} & \hat{\mathbf{L}}^{2,\top} & \cdots & \hat{\mathbf{L}}^{n,\top}\end{bmatrix}^\top$ and $\underset{\sim}{\mathbf{L}} = \begin{bmatrix}\mathbf{L}^{1,\top} & \mathbf{L}^{2,\top} & \cdots & \mathbf{L}^{n,\top}\end{bmatrix}^\top$. We can now construct our matrix representation of the saddle point formulation by putting together the pieces $\underset{\sim}{\mathbf{G}}, \underset{\sim}{\mathbf{P}}, \underset{\sim}{\mathbf{F}}, \underset{\sim}{\hat{\mathbf{L}}}$. Recall that our original saddle point equation is the set of equations,

$$\begin{aligned} A\mathbf{f} + B^*\mathbf{p} &= \boldsymbol{\ell} \\ B\mathbf{f} &= \mathbf{0} \end{aligned}$$

Substituting in the appropriate pieces gives,

$$\begin{aligned} \underset{\sim}{\mathbf{G}} + \underset{\sim}{\mathbf{P}} &= \underset{\sim}{\hat{\mathbf{L}}} \\ \underset{\sim}{\mathbf{F}} &= \mathbf{0} \end{aligned}$$

or more explicitly,

$$\begin{bmatrix}(\mathbf{M}^{\mathcal{H}})^{-1}\mathbf{A} & (\mathbf{M}^{\mathcal{H}})^{-1}\mathbf{B}^\top \\ (\mathbf{M}^{Q^n})^{-1}\mathbf{B} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\underset{\sim}{\mathbf{F}} \\ \underset{\sim}{\mathbf{P}}\end{bmatrix} = \begin{bmatrix}(\mathbf{M}^{\mathcal{H}})^{-1}\underset{\sim}{\mathbf{L}} \\ \mathbf{0}\end{bmatrix}$$

and by multiplying the top partition by $\mathbf{M}^{\mathcal{H}}$ and the lower partition by $\mathbf{M}^{Q^n}$, we get the matrix representation we need,

$$\begin{bmatrix}\mathbf{A} & \mathbf{B}^\top \\ \mathbf{B} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\underset{\sim}{\mathbf{F}} \\ \underset{\sim}{\mathbf{P}}\end{bmatrix} = \begin{bmatrix}\underset{\sim}{\mathbf{L}} \\ \mathbf{0}\end{bmatrix} \tag{2.3.25}$$

Up until now we have been using the most general form for the operator $\mathbf{A}$ but recall that for our problem this matrix has the very simple form of

$$A_{ik} = \begin{cases} I_{H^i} & \text{if } i = k \\ 0 & \text{otherwise} \end{cases}$$

From the construction of $\mathbf{A}$, this implies that each of its block submatrix $(\mathbf{M}^{H^i})^{-1}\mathbf{A}_{ik}$ has the simple form

$$(\mathbf{M}^{H^i})^{-1}\mathbf{A}_{ik} = \begin{cases} \mathbf{I} = (\mathbf{M}^{H^i})^{-1}\mathbf{M}^{H^i} & \text{if } i = k \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbf{I}$ is the infinite identity matrix. For a general choice of bases $\{\phi_j^{H^i}\}_{j\in\mathbb{N}}$, the matrix $\mathbf{A}$ is full rank, but its entries depend upon the unknown measure ρ^i, $i = 1, \ldots, n$. This is easily seen by observing the matrix construction laid out in appendix E. The other matrix representation in Equation 2.3.33 is $\mathbf{B}$. Recall that $\mathbf{B}$ is the matrix representation of the constraint operator B. B is the operator defined in terms of the constraint matrix $\mathbf{M}^B$. Understanding the structure of $\mathbf{M}^B$ is not as straight-forward as with $\mathbf{A}$. Additionally, it too depends on the unknown measure ρ^i, $i = 1, \ldots, n$. We can greatly simplify our problems with both $\mathbf{A}$ and $\mathbf{M}^B$ by our choice of bases.

2.3.5 Finite Dimensional Approximation in Common Basis

Another consequence of norm equivalence discussed in Section 2.3.3 is that a basis for Q is also a basis for H^i. In particular it means that the commutative diagram in Figure 2.3.3 holds, where m represents a Lebesgue measure on the space $Q := \mathsf{L}^2$. We can easily show

$$\begin{array}{ccc} \|\cdot\|_{H^i} & \Longleftrightarrow & \|\cdot\|_Q \\ \Updownarrow & & \Updownarrow \\ \mathsf{L}^2(\rho_X^i) & \Longleftrightarrow & \mathsf{L}^2(m) \end{array}$$

Figure 2.3.3: Commutative Diagram of Norm Equivalence

that,

$$\{\phi_i^Q\}_{i\in\mathbb{N}} \text{ is a basis for } Q \iff \{\phi_k^Q\}_{k\in\mathbb{N}} \text{ is a basis for } H^i \tag{2.3.26}$$

provided that the norm equivalence condition holds.

Proof. In order to show that Equation 2.3.26 is true, for each $f \in H^i$ it must be shown that there exists $\{a_k\}_{k\in\mathbb{N}}$ such that,

$$\left\| f - \sum_k a_k \, \phi_k^Q \right\|_{H^i} = 0$$

From the definition of a basis on a normed space, we know that if $f \in Q$ there exists a sequence of real numbers $\{a_k\}_{k\in\mathbb{N}}$ such that

$$\left\| f - \sum_k a_k \, \phi_k^Q \right\|_{Q} = 0$$

From the norm equivalence requirement we additionally know that,

$$\underline{c}^i \left\| f - \sum_k a_k \, \phi_k^Q \right\|_{H^i} \leq \underbrace{\left\| f - \sum_k a_k \, \phi_k^Q \right\|_{Q}}_{=0}$$

We conclude that,

$$\underline{c}^i \left\| f - \sum_k a_k \, \phi_k^Q \right\|_{H^i} = 0,$$

and thus $\{\phi_k^Q\}_{k\in\mathbb{N}}$ is also a basis for H^i. □

We can now derive the finite dimensional approximation as in Section 2.3.4 but just using the inner product on the constraint space Q. We start with the basis $\{\phi_j^Q\}_{j\in\mathbb{N}}$ for H^i and Q. We represent each element $f^i \in H^i$ and $p^i \in Q$ as

$$f^i = \sum_j \phi_j^Q F_j^i \tag{2.3.27a}$$

$$p^i = \sum_j \phi_j^Q P_j^i \tag{2.3.27b}$$

Again, recall that for agent i we define $\mathbf{g} = A\mathbf{f}$, where $\mathbf{g}, \mathbf{f} \in \mathcal{H}$. However, to make use of the inner product on Q we must be able to interpret A as an operator on Q^n, where $Q^n = Q \times Q \times \cdots \times Q$ n times. Strictly speaking the operators A and B that appear in the saddle point equations in Section 2.3.2, shown again below,

$$\begin{bmatrix} A & B^* \\ B & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_\rho \\ \mathbf{0} \end{bmatrix}$$

are defined as mappings,

$$A : \mathcal{H} \to \mathcal{H}$$
$$B : \mathcal{H} \to \mathcal{Q}$$

However, by virtue of the norm equivalence, any operator $A : \mathcal{H} \to \mathcal{H}$ also acts on Q^n. It is possible to be more rigorous in defining the induced operator acting on Q^n by introducing an injection operator,

$$\imath_i : Q \to H^i \quad \text{for } i = 1, \ldots, n$$

The injection operator associates to each element $q \in Q$ the same element in H^i denoted $\imath_i(q)$. With this injection, the norm equivalence is rewritten as,

$$\underline{c}_i \|\imath_i(q)\|_{H^i} \leq \|f\|_Q \leq \bar{c}_i \|\imath_i(q)\|_{H^i}$$

for each $i = 1, \ldots, n$. It is straightforward to show from this form of the norm equivalence that $\imath_i$ is boundedly invertible from Q to H^i for each $i = 1, \ldots, n$ and consequently $\boldsymbol{\imath}$ is boundedly invertible from Q^n to $\mathcal{H}$.

The action of the operator A on $\mathcal{H}$ induces an operator on Q^n via the identities,

$$\imath_i^{-1} A_{ii} \imath_i : Q \to Q$$

In fact in our case, since $A_{ii} := I_{H^i}$, the induced operator is simply the identity on Q,

$$\imath_i^{-1} A_{ii} \imath_i = \imath_i^{-1} I_{H^i} \imath_i = \imath_i^{-1} \imath_i = I_Q$$

With this interpretation in mind, we simply use A^i or A_{ik} to denote the operator on H^i or the induced operator on Q in the following discussions - we suppress the notation of the injection operator $\imath_i : Q \to H^i$.

Substituting in f^i and p^i from Equations 2.3.27a and 2.3.27b we get

$$g^i = \sum_j \phi_j^Q G_j^i = \sum_k \sum_j A_{ik} \phi_j^Q F_j^i \tag{2.3.28}$$

and pull out the coefficients by taking the inner product over the constraint space Q of Equation 2.3.28 with an arbitrary basis element ϕ_r^Q.

$$\left(\phi_r^Q, \sum_j \phi_j^Q G_j^i \right)_Q = \left(\phi_r^Q, \sum_{j,k} A_{ik} \phi_j^Q F_j^k \right)_Q$$

Using norm properties,

$$\sum_j \left(\phi_r^Q, \phi_j^Q \right)_Q G_j^i = \sum_{j,k} \left(\phi_r^Q, A_{ik} \phi_j^Q \right)_Q F_j^k \tag{2.3.29}$$

The inner products on both sides of equation induce a simple multiplication of a real-valued matrix by a real-valued vector. We fix each r and then sum over j. On the left-hand side this gives the matrix $\mathbf{M}^{Q^n}$ times the vector $\mathbf{G}^i$, where $\mathbf{G}^i = \begin{bmatrix} G_1^i & \cdots & G_j^i & \cdots \end{bmatrix}^\top$. On the right-hand side, we define the matrix $\mathbf{A}_{ik(rj)}$ by its entries

$$\mathbf{A}_{ik(rj)} := \left(\phi_r^Q, A_{ik} \phi_j^Q \right)_Q$$

Summing over j leaves us with

$$\mathbf{M}^{Q^n} \mathbf{G}^i = \sum_k \mathbf{A}_{ik(rj)} \mathbf{F}^k$$

We premultiply by the inverse of the Gramian $\mathbf{M}^{Q^n}$ and for each fixed i we sum over k giving,

$$\underset{\sim}{\mathbf{G}} = (\mathbf{M}^{Q^n})^{-1} \mathbf{A} \underset{\sim}{\mathbf{F}} \tag{2.3.30}$$

where $\underset{\sim}{\mathbf{G}}$ and $\underset{\sim}{\mathbf{F}}$ are constructed by stacking the vectors $\mathbf{G}^i$ and $\mathbf{F}^i$, respectively. More explicitly,

$$\underset{\sim}{\mathbf{G}} = \begin{bmatrix} \mathbf{M}^{Q^n 1} & 0 & \cdots & 0 \\ 0 & \mathbf{M}^{Q^n 2} & \cdots & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{M}^{Q^n n} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \cdots & \mathbf{A}_{1n} \\ \mathbf{A}_{21} & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{n1} & \cdots & \cdots & \mathbf{A}_{nm} \end{bmatrix} \underset{\sim}{\mathbf{F}} = (\mathbf{M}^{Q^n})^{-1} \mathbf{A} \underset{\sim}{\mathbf{F}}$$

We do not need to revisit $\mathbf{p} = B\mathbf{f}$, where $\mathbf{p} \in \mathcal{Q}$ is an $(n-1) \times 1$ vector of functions and B is an operator. This derivation was originally done in the constraint space Q and we can simply refer to Equation 2.3.20.

Next we re-examine $\mathbf{f} = B^*\mathbf{p}$. Now, for agent i we have,

$$f^i = \sum_j \phi_j^Q F_j^i = \sum_{j,k} B_{ik}^* \phi_j^Q P_j^k$$

We take the inner product over Q with an arbitrary basis function ϕ_r^Q,

$$\sum_j \left(\phi_r^Q, \phi_j^Q\right)_Q F_j^i = \sum_{j,k} \left(\phi_r^Q, B_{ik}^* \phi_j^Q\right)_Q P_j^k$$

Though $B_{ik}^* \phi_j^Q = h \in H^i$ it still makes sense to take the inner product in Q. By norm equivalence if h is an element of H^i then it is also an element of Q.

For a fixed r and summation over j we define the following matrix,

$$\mathbf{B^*}_{ik(rj)} := \left(\phi_r^Q, B_{ik}^* \phi_j^Q\right)_Q$$

Using the identity derived in Section 2.3.4 $\mathbf{B^*}_{ki(jr)} = \mathbf{B}_{ik(jr)}^\top$ we can now write,

$$\mathbf{M}^{Q^n} \mathbf{F}^i = \sum_k \mathbf{B}_{ik(jr)}^\top \mathbf{P}^k$$

Fixing i and summing over k, and taking note that $\mathbf{M}^{H^i}$ is another Gramian matrix we get,

$$\begin{aligned} \mathbf{M}^{Q^n} \underset{\sim}{\mathbf{F}} &= \mathbf{B}^\top \underset{\sim}{\mathbf{P}} \\ \underset{\sim}{\mathbf{F}} &= (\mathbf{M}^{Q^n})^{-1} \mathbf{B}^\top \underset{\sim}{\mathbf{P}} \end{aligned} \tag{2.3.31}$$

or

$$\underset{\sim}{\mathbf{F}} = \begin{bmatrix} \mathbf{M}^{H^1} & 0 & \cdots & 0 \\ 0 & \mathbf{M}^{H^2} & \cdots & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{M}^{H^n} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}_{11}^\top & \mathbf{B}_{12}^\top & \cdots & \mathbf{B}_{1(n-1)}^\top \\ \mathbf{B}_{21}^\top & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{n1}^\top & \cdots & \cdots & \mathbf{B}_{n(n-1)}^\top \end{bmatrix} \underset{\sim}{\mathbf{P}} = (\mathbf{M}^{Q^n})^{-1} \mathbf{B}^\top \underset{\sim}{\mathbf{P}}$$

Lastly, we rewrite the representation of the regressors, $\boldsymbol{\ell} = \begin{bmatrix} \ell^1 & \ell^2 & \cdots & \ell^{n-1} \end{bmatrix}$, $\ell^i = f_{\rho^i}$.

$$\ell^i := f_{\rho^i} = \sum_j \phi_j^Q \hat{L}_j^i \tag{2.3.32}$$

As before we take the inner product of our expansion with an arbitrary basis function ϕ_r^Q.

$$\mathbf{L}^i := \left(\phi_r^Q, f_{\rho^i}\right)_Q = \sum_j \left(\phi_r^Q, \phi_j^Q\right)_Q \hat{L}_j^i$$

We can know write the matrix form of Equation 2.3.32

$$\hat{\mathbf{L}}^i = (\mathbf{M}^{Q^n})^{-1}\mathbf{L}^i$$

or

$$\underset{\sim}{\hat{\mathbf{L}}} = (\mathbf{M}^{Q^n})^{-1}\underset{\sim}{\mathbf{L}}$$

where $\underset{\sim}{\hat{\mathbf{L}}} = \begin{bmatrix}\hat{\mathbf{L}}^{1,\top} & \hat{\mathbf{L}}^{2,\top} & \cdots & \hat{\mathbf{L}}^{n,\top}\end{bmatrix}^\top$ and $\underset{\sim}{\mathbf{L}} = \begin{bmatrix}\mathbf{L}^{1,\top} & \mathbf{L}^{2,\top} & \cdots & \mathbf{L}^{n,\top}\end{bmatrix}^\top$. Using the same procedure as at the end of in Section 2.3.4 we arrive at,

$$\begin{bmatrix}(\mathbf{M}^{Q^n})^{-1}\mathbf{A} & (\mathbf{M}^{Q^n})^{-1}\mathbf{B}^\top \\ (\mathbf{M}^{Q^n})^{-1}\mathbf{B} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\underset{\sim}{\mathbf{F}} \\ \underset{\sim}{\mathbf{P}}\end{bmatrix} = \begin{bmatrix}(\mathbf{M}^{Q^n})^{-1}\underset{\sim}{\mathbf{L}} \\ \mathbf{0}\end{bmatrix}$$

Simple multiplication by $\mathbf{M}^{Q^n}$ gives us our final form,

$$\begin{bmatrix}\mathbf{A} & \mathbf{B}^\top \\ \mathbf{B} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\underset{\sim}{\mathbf{F}} \\ \underset{\sim}{\mathbf{P}}\end{bmatrix} = \begin{bmatrix}\underset{\sim}{\mathbf{L}} \\ \mathbf{0}\end{bmatrix} \tag{2.3.33}$$

To clarify the coefficients $\mathbf{F}^i$ and $\mathbf{P}^i$ are not the same as in Equation 2.3.33. So as to minimize the number of symbols being used we chose to use the same symbol for the coefficients weighing each basis element ϕ_j^Q as we used with $\phi_j^{H^i}$.

Chapter 3

The Learning Dynamic

3.1 General Definitions

The learning dynamic discussed in Chapter 2 is studied in this chapter as a probabilistic implementation of the Uzawa algorithm described in [44]. Before stating the Uzawa algorithm and how we employ it, some mathematical background needs to be reviewed. The Uzawa algorithm has been studied in great detail over the years, and a general framework has been derived to study the method. This framework exploits the equivalence of certain linear operators between real Hilbert spaces to bilinear forms acting on those Hilbert spaces. A real-valued bilinear form is a mapping $a(\cdot,\cdot) : H \times H \to \mathbb{R}$ such that $a(\alpha u + \beta v, w) = \alpha a(u,w) + \beta a(v,w)$ for all scalars $\alpha, \beta \in \mathbb{R}$ and all $u, v, w \in H$. If $A : H \to H$ is a linear operator, it is possible to define a bilinear form via the identity

$$(Au, v)_H = a(u,v) \quad \forall v \in H \tag{3.1.1}$$

The use of bilinear forms for the study of saddle point problems derives, historically speaking, from the study of partial differential equations subject to linear or convex constraints. See Glowinski [44] for a detailed discussion.

The bilinear framework introduces many equivalent forms of the saddle point problem. One of these forms, the primal form, makes use of the operator A acting from $H \to H$. Another equivalent form, the dual form, makes use of the operator $\mathcal{A}$ that acts from $H \to H^*$. The operators, A and $\mathcal{A}$ are closely related. If $a(\cdot,\cdot)$ is a bounded bilinear form, it is always possible to define a bounded linear operator $\mathcal{A} : H \to H^*$ such that,

$$\langle \mathcal{A}u, v \rangle_{H^* \times H} = a(u,v) \quad \forall v \in H$$

where $\langle \cdot, \cdot \rangle$ denotes the duality pairing. Combining Equation 3.1.1 and the above, it follows that the operators A and $\mathcal{A}$ satisfy

$$\langle \mathcal{A}u, v \rangle = (Au, v)_H \qquad \forall u, v \in H$$

The choice of whether to express the underlying saddle point equation in terms of either the operator $\mathcal{A}$ or A gives rise to alternative forms which are discussed in detail next.

In the original saddle point equation we seek the pair $(\mathbf{f}, \mathbf{p}) \in \mathcal{H} \times \mathcal{Q}$ that satisfy

$$\begin{bmatrix} A & B^* \\ B & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\ell} \\ \mathbf{0} \end{bmatrix} \tag{3.1.2}$$

We refer to these equations as the saddle point equations in the primal space. We want to reformulate the saddle point problem into the dual space. We start by constructing a version of the governing equations expressed in terms of bilinear forms. Since $\mathcal{H} \times \mathcal{Q}$ is a product of inner product spaces, we define the usual inner product $(\cdot, \cdot)_{\mathcal{H}\times\mathcal{Q}}$ on the space $\mathcal{H} \times \mathcal{Q}$. Equation 3.1.2 is equivalent to requiring that

$$\left(\begin{bmatrix} A & B^* \\ B & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{p} \end{bmatrix} - \begin{bmatrix} \boldsymbol{\ell} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{h} \\ \mathbf{q} \end{bmatrix} \right)_{\mathcal{H}\times\mathcal{Q}} = 0$$

for all $(\mathbf{h}, \mathbf{q}) \in \mathcal{H} \times \mathcal{Q}$. By the properties of the inner product, Equation 3.1.2 simplifies to the requirement that

$$\begin{aligned} (A\mathbf{f}, \mathbf{h})_{\mathcal{H}} + (B^*\mathbf{p}, \mathbf{h})_{\mathcal{H}} - (\boldsymbol{\ell}, \mathbf{h})_{\mathcal{H}} &= 0 \qquad \text{(Primal)} \\ (B\mathbf{f}, \mathbf{q})_{\mathcal{Q}} &= 0 \end{aligned} \tag{3.1.3}$$

for all $(\mathbf{h}, \mathbf{q}) \in \mathcal{H}\times\mathcal{Q}$. Equations 3.1.3 are used to define the bilinear forms that appear in the saddle point problem. We use the identity from Equation 3.1.1 and define $b(\cdot, \cdot) : \mathcal{H}\times\mathcal{Q} \to \mathbb{R}$ to obtain,

$$\begin{aligned} a(\mathbf{f}, \mathbf{h}) + b(\mathbf{p}, \mathbf{h}) &= (\boldsymbol{\ell}, \mathbf{h})_{\mathcal{H}} \\ b(\mathbf{f}, \mathbf{q}) &= 0 \end{aligned} \tag{3.1.4}$$

for all $(\mathbf{h}, \mathbf{q}) \in \mathcal{H} \times \mathcal{Q}$. Equations 3.1.4 can also be written in terms of the operators $\mathcal{A} : \mathcal{H} \to \mathcal{H}^*$ and $\mathcal{B} : \mathcal{H} \to \mathcal{Q}^*$. They hold if and only if

$$\begin{aligned} \langle \mathcal{A}\mathbf{f}, \mathbf{h} \rangle_{\mathcal{H}^*\times\mathcal{H}} + \langle \mathcal{B}^\top \mathbf{p}, \mathbf{h} \rangle_{\mathcal{H}^*\times\mathcal{H}} &= \langle \mathcal{R}_{\mathcal{H}}^{-1} \boldsymbol{\ell}, \mathbf{h} \rangle_{\mathcal{H}^*\times\mathcal{H}} \\ \langle \mathcal{B}\mathbf{f}, \mathbf{q} \rangle_{\mathcal{Q}^*\times\mathcal{Q}} &= 0 \end{aligned} \tag{3.1.5}$$

for all $(\mathbf{h}, \mathbf{q}) \in \mathcal{H} \times \mathcal{Q}$. In this equation $\mathcal{B}^\top$ is the transpose or dual of the operator $\mathcal{B}$. Additionally, $\mathcal{R}_{\mathcal{H}} : \mathcal{H}^* \to \mathcal{H}$ represents the Riesz map from the dual of the hypothesis space $\mathcal{H}^*$ to the Hilbert spaces $\mathcal{H}$. Similarly, $\mathcal{R}_{\mathcal{Q}} : \mathcal{Q}^* \to \mathcal{Q}$ is defined as the Riesz map from the dual of the constraint space $\mathcal{Q}$ to the constraint space $\mathcal{Q}$. Since the Riesz map is an isometric isomorphism, the inverses of the operators exist. We represent them as $\mathcal{R}_{\mathcal{H}}^{-1} : \mathcal{H} \to \mathcal{H}^*$ and $\mathcal{R}_{\mathcal{Q}}^{-1} : \mathcal{Q} \to \mathcal{Q}^*$. By definition, the Riesz maps satisfy

$$\begin{aligned} \langle \mathcal{R}_{\mathcal{H}}^{-1}\mathbf{f}, \mathbf{g} \rangle_{\mathcal{H}^*\times\mathcal{H}} &= (\mathbf{f}, \mathbf{g})_{\mathcal{H}} && \forall \mathbf{f}, \mathbf{g} \in \mathcal{H} \\ \langle \mathcal{R}_{\mathcal{Q}}^{-1}\mathbf{p}, \mathbf{q} \rangle_{\mathcal{Q}^*\times\mathcal{Q}} &= (\mathbf{p}, \mathbf{g})_{\mathcal{Q}} && \forall \mathbf{p}, \mathbf{q} \in \mathcal{Q} \end{aligned}$$

Equation 3.1.3 can be re-written using the identities above as,

$$\begin{aligned}\left\langle \mathcal{R}_{\mathcal{H}}^{-1} A\mathbf{f}, \mathbf{h} \right\rangle_{\mathcal{H}^* \times \mathcal{H}} + \left\langle \mathcal{R}_{\mathcal{H}}^{-1} B^* \mathbf{p}, \mathbf{h} \right\rangle_{\mathcal{H}^* \times \mathcal{H}} &= \left\langle \mathcal{R}_{\mathcal{H}}^{-1} \boldsymbol{\ell}, \mathbf{h} \right\rangle_{\mathcal{H}^* \times \mathcal{H}} \\ \left\langle \mathcal{R}_{\mathcal{Q}}^{-1} B\mathbf{f}, \mathbf{q} \right\rangle_{\mathcal{Q}^* \times \mathcal{Q}} &= 0 \end{aligned} \tag{3.1.6}$$

for all $(\mathbf{f}, \mathbf{q}) \in \mathcal{H} \times \mathcal{Q}$. Comparing Equations 3.1.5 and 3.1.6, we see that it must be true that

$$\begin{aligned}\mathcal{A} &= \mathcal{R}_{\mathcal{H}}^{-1} A \\ \mathcal{B}^\top &= \mathcal{R}_{\mathcal{H}}^{-1} B^* \\ \mathcal{B} &= \mathcal{R}_{\mathcal{Q}}^{-1} B \end{aligned}$$

Now the dual form of the saddle point problem seeks to find $(\mathbf{f}, \mathbf{p}) \in \mathcal{H} \times \mathcal{Q}$ such that

$$\begin{bmatrix} \mathcal{A} & \mathcal{B}^\top \\ \mathcal{B} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathcal{R}^{-1} \boldsymbol{\ell} \\ \mathbf{0} \end{bmatrix} \in \mathcal{H}^* \times \mathcal{Q}^* \tag{3.1.7}$$

3.2 Inexact Uzawa Algorithm

As was noted at the end of Chapter 2, the saddle point formulation for our learning problem with consensus cannot be solved directly due to the unknown distributions ρ^i, $i = 1, \ldots, n$, associated with each Hilbert space H^i. In Section 2.3.5 we simplified $\mathbf{A}$ and $\mathbf{B}$ by choosing the bases $\{\phi_j^{H^i}\}_{j \in \mathbb{N}} \equiv \{\phi_j^Q\}_{j \in \mathbb{N}}$ so that they are now computable. However, the vector $\underset{\sim}{\mathbf{L}}$ is still dependent upon ρ^i, $i = 1, \ldots, n$. We use the inexact Uzawa algorithm to generate a sequence of convergent approximations to our saddle point problem in terms of an approximation $\mathbf{L}_a$ of $\mathbf{L}$. We base our analysis on the theory discussed in [45]. In this paper Bacuta introduces the operator $\psi : V^* \subset \mathcal{H}^* \to \mathcal{H}$ as the approximate inverse operator that returns the approximate solution ξ to the equation $A\xi = \gamma$, for $\gamma \in V^*$. For our saddle point problem we select $\psi : \mathcal{H}^* \to \mathcal{H}$ to create an approximation $\mathbf{g}_a \in \mathcal{H}$, $\mathbf{g}_a = \psi(\mathcal{R}_H^{-1}\boldsymbol{\ell} - \mathcal{A}\mathbf{f}_{k-1} - \mathcal{B}^\top \mathbf{p}_{k-1})$, of the solution $\mathbf{g}$ of the equation

$$\mathcal{A}\mathbf{g} = \mathcal{R}_H^{-1}\boldsymbol{\ell} - \mathcal{A}\mathbf{f}_{k-1} - \mathcal{B}^\top \mathbf{p}_{k-1}$$

The regressor $\boldsymbol{\ell}$ is not indexed by k because $\boldsymbol{\ell}$ represents the regressor which is based on the unknown probability distribution ρ^i. We summarize the algorithm as it applies to our specific problem in Figure 3.2.1. This, however, is not a form that we can execute in a program. The unknowns in this algorithm evolve in infinite dimensional spaces. We have already demonstrated in Section 2.3.4 that from the primal space we can rewrite the saddle point problem in matrix form where f^i is given in terms of a vector of coefficients $\mathbf{F}^i$ and p^i is expressed via to the vector of coefficients $\mathbf{P}^i$. By the linearity and isometry properties of the Riesz map, we easily transform the dual form of our algorithm into the primal formulation. In this form our approximation operator is now $\hat{\psi} = \psi \mathcal{R}_H^{-1} : H \to H$. It generates an

Algorithm 3.2.1: TWO STAGE LEARNING PROCESS$(\mathbf{f}_k, \mathbf{p}_k)$

Choose initial estimates $\mathbf{f}_0, \mathbf{p}_0 \in \mathcal{H} \times \mathcal{Q}$
repeat
for $k = 1, 2, \ldots$
Update the estimate of the agents, $\mathbf{f}_k$

$$\mathbf{f}_k = \mathbf{f}_{k-1} + \psi(\mathcal{R}_H^{-1}\boldsymbol{\ell} - \mathcal{A}\mathbf{f}_{k-1} - \mathcal{B}^\top \mathbf{p}_{k-1})$$

Update the estimates of the multipliers, $\mathbf{p}_k$

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \alpha \mathcal{R}_Q \mathcal{B}\mathbf{f}_k$$

until Convergence

Figure 3.2.1: Dual Operator Form of Inexact Uzawa Algorithm

Algorithm 3.2.2: TWO STAGE LEARNING PROCESS$(\mathbf{f}_k, \mathbf{p}_k)$

Choose initial estimates $\mathbf{f}_0, \mathbf{p}_0 \in \mathcal{H} \times \mathcal{Q}$
repeat
for $k = 1, 2, \ldots$
Update the estimate of the agents, $\mathbf{f}_k$

$$\mathbf{f}_k = \mathbf{f}_{k-1} + \hat{\psi}(\boldsymbol{\ell} - A\mathbf{f}_{k-1} - B^*\mathbf{p}_{k-1})$$

Update the estimates of the multipliers, $\mathbf{p}_k$

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \alpha B\mathbf{f}_k$$

until Convergence

Figure 3.2.2: Primal Operator Form of Inexact Uzawa Algorithm

approximation $\mathbf{g}_a$ to the solution $\mathbf{g}$ of the equation

$$A\mathbf{g} = \boldsymbol{\ell} - A\mathbf{f}_{k-1} - B^*\mathbf{p}_{k-1}$$

To reach a computable version of the inexact Uzawa algorithm we must derive a matrix form of the update equations. We claim the result is as shown in Figure 3.2.3. To prove our claim we use the same method used in deriving the matrix form of the saddle point problem in Section 2.3.4.

Algorithm 3.2.3: TWO STAGE LEARNING PROCESS($\mathbf{F}_k, \mathbf{P}_k$)

Choose initial estimates $\mathbf{F}_0, \mathbf{P}_0 \in l_2 \times l_2$
repeat
for $k = 1, 2, \ldots$
Update the estimate of the agents, $\mathbf{F}_k$
$\mathbf{F}_k = \mathbf{L}_{a,k-1} - \mathbf{B}^\top \mathbf{P}_{k-1}$
Update the estimates of the multipliers, $\mathbf{P}_k$
$\mathbf{P}_k = \mathbf{P}_{k-1} + \alpha \mathbf{B}\mathbf{F}_k$
until Convergence

Figure 3.2.3: Coefficient Form of Inexact Uzawa Algorithm

Proof. We begin with the first update equation of the Uzawa Algorithm for an agent i,

$$f_k^i = f_{k-1}^i + \psi(\ell^i - A f_{k-1}^i - \sum_m B_{im}^* p_{k-1}^m) \tag{3.2.1}$$

For our problem we have $A = I_H$ and hence we can define the approximate inverse operator as,

$$\psi(\ell^i - A f_{k-1}^i - \sum_m B_{im}^* p_{k-1}^m) := \ell_{a,k-1}^i - f_{k-1}^i - \sum_m B_{im}^* p_{k-1}^m$$

where ℓ_a^i is an approximation of the unknown $\ell^i = f_{\rho^i}$. We substitute this back into Equation 3.2.1 and obtain the update equation,

$$f_k^i = \ell_{a,k-1}^i - \sum_m B_{im}^* p_{k-1}^m$$

We substitute in the same expansions we used in our finite dimensional approximation formulation,

$$f_k^i = \sum_j \phi_j^{H^i} F_{k,j}^i \tag{3.2.2}$$

$$\ell_{a,k}^i = \sum_j \phi_j^{H^i} L_{a,k,j}^i \tag{3.2.3}$$

$$p_{k-1}^i = \sum_j \phi_j^{H^i} P_{(k-1),j}^i \tag{3.2.4}$$

into our update equation and take the inner product over H^i with a basis function $\phi_r^{H^i}$. This yields the following matrix formulation,

$$\mathbf{M}^{H^i} \mathbf{F}_k^i = \mathbf{M}^{H^i} \mathbf{L}_{a,(k-1)}^i - \sum_m \mathbf{B}_{im(rj)}^* \mathbf{P}_{k-1}^m$$

In the finite dimensional approximation section we chose our basis vectors $\{\phi_j^{H^i}\}_{j\in\mathbb{N}} = \{\phi_j^Q\}_{j\in\mathbb{N}}$, where $\{\phi_j^Q\}_{j\in\mathbb{N}}$ is an orthonormal set. We do so again here and hence $\mathbf{M}^{H^i} = \mathbf{I}$, producing the coefficient formulation of Equation 3.2.1,

$$\mathbf{F}_k^i = \mathbf{L}_{a,(k-1)}^i - \sum_m \mathbf{B}_{im(jr)}^\top \mathbf{P}_{k-1}^m$$

where the relationship between $\mathbf{B}_{im}^*$ and $\mathbf{B}_{im}^\top$ should be recalled from Section 2.3.4. Fixing i and summing over m gives the final form,

$$\mathbf{F}_k = \mathbf{L}_{a,(k-1)} - \mathbf{B}^\top \mathbf{P}_{k-1}$$

The second update equation of the Uzawa algorithm for an agent i is

$$p_k^i = p_{k-1}^i + \alpha \sum_m B_{im}^* f_k^m \tag{3.2.5}$$

Again, we substitute in our basis expansions shown in Equations 3.2.2 - 3.2.4 into Equation 3.2.5 and take the inner product in Q with an arbitrary basis element ϕ_r^Q. This yields the matrix equation

$$\mathbf{M}^{Q^n}\mathbf{P}_k^i = \mathbf{M}^{Q^n}\mathbf{P}_{k-1}^i + \alpha \sum_m \mathbf{B}_{im(rj)}\mathbf{F}_k^m$$

As above we simplify our equation by noting $\{\phi_j^Q\}_{j\in\mathbb{N}}$ is orthonormal and perform the same summations as above to get our final form,

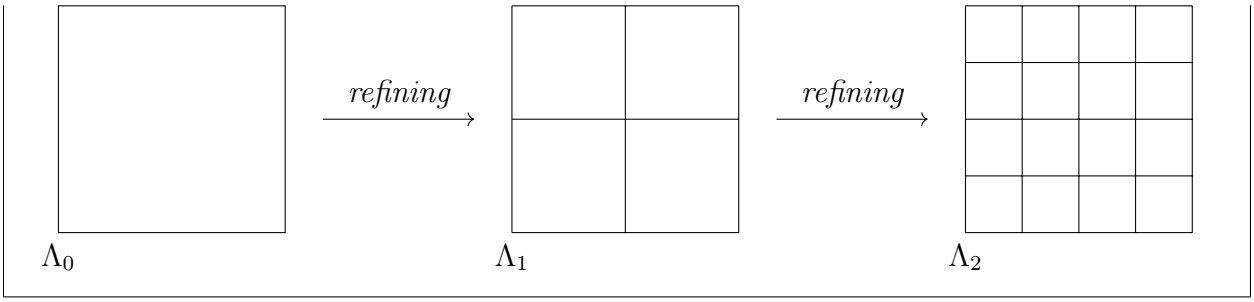
$$\mathbf{P}_k = \mathbf{P}_{k-1} + \alpha \mathbf{B}\mathbf{F}_k$$

□

3.3 Approximation on Uniform Partitions

Although we cannot solve Equation 2.3.12 directly, we can generate approximations of the solution which entails approximating $\boldsymbol{\ell}$. We can use the recursion summarized in Figure 3.2.3 to construct an approximation of $\boldsymbol{\ell}$ using the methodology derived in [46]. The approximation is defined in terms of the projection operator $P_{\Lambda_l} : H^i \to V$, that projects $f \in H^i$ onto the space U of piecewise constant functions on the uniform partition Λ_l at resolution level l. The partition Λ_l has a^l cells where $a \geq 2$ is the number of children induced in each cell when the partition is refined.

Example 7. *For $a = 2$ and partition level $l = 0, 1, 2$, the partition Λ_l will be*



Define $\Gamma^s_{\rho^i}$ as the approximation space that consists of all the functions in $\mathsf{L}^2(\rho^i_X) = H^i$ such that the approximation error satisfies,

$$\|f - P_{\Lambda_l} f\|_{H^i} \lesssim a^{-ls}$$

where s is the index of smoothness, also known as the approximation rate. In other words, $\Gamma^s_{\rho^i} := \{f \in \mathsf{L}^2(\rho^i_X) : \|f - P_{\Lambda_l} f\|_{H^i} \lesssim a^{-ls}\}$. The approximation space $\Gamma^s_{\rho^i}$ are nested as shown in Figure 3.3.1. Roughly speaking, the index s that measures approximation rate (often) corresponds to smoothness. The smoother a function is, the faster the rate of approximation converges to zero. See [5] for a detailed discussion.

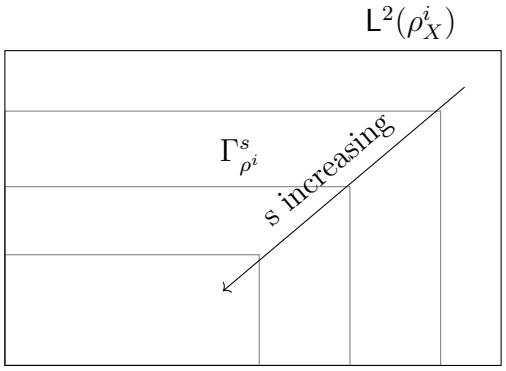


Figure 3.3.1: Approximation Spaces. As the smoothness of the functions increase the size of their space decreases

We will use a result from [46] that constructs approximations of $f^i_{\mathbf{z}^i,\Lambda_l}$ to the regressor $f^i_{\rho^i}$ on uniform partitions Λ_l and show that this approximation strategy, when used with the learning dynamic of the Uzawa algorithm, yields convergent approximations of the optimal consensus estimate.

Recall from Section 2.1.2 that one way to construct a loss functional that can be evaluated in the absence of a known probability measure ρ is to use empirical data as in Equation 2.1.3 based on the training set $\mathbf{z}^i_{1:m_k} = \{x^i_j, y^i_j\}_{j=1:m_k}$. The approach in [46] defines such a loss functional for each $i = 1, \ldots, n$ on the training set as

$$\mathcal{J}^i_{emp,k}(f^i) := \frac{1}{m_k} \sum_{j=1}^{m_k} (y^i_j - f^i(x_j))^2$$

where again m_k is the number of observations made prior to the construction by agent i of estimate f_k^i. The empirical estimate $f^i_{\mathbf{z}^i_{1:m_k},\Lambda_l}$ is defined to be

$$f^i_{\mathbf{z}^i_{1:m_k},\Lambda_l} = \underset{f\in\mathcal{S}_{\Lambda_l}\subseteq H^i}{\arg\min}\, \mathcal{J}^i_{emp,k} \tag{3.3.1}$$

where $\mathcal{S}_{\Lambda_l}$ is the span of piecewise constant functions on the uniform partition Λ_l at refinement level l. Because the space $\mathcal{S}_{\Lambda_l}$ is a collection of piecewise constant functions, the optimization problem in Equation 3.3.1 can be recast in terms of a^l independent minimization problems over each cell $\lambda \in \Lambda_l$. In each of these problems we seek the constants $c_{\lambda,k}$

$$c_{\lambda,k} = \underset{c\in\mathbb{R}}{\arg\min} \underbrace{\frac{1}{m_k}\sum_{j=1}^{m_k}(y^i_j - c)^2\chi_\lambda(x^i_j)}_{\text{local average}}$$

where $\chi_\lambda(\cdot)$ is the characteristic function of the cell λ. We then define

$$f_{\mathbf{z}^i_{1:m_k},\Lambda_l} = \sum_{\lambda\in\Lambda_l} c_{\lambda,k}\chi_\lambda$$

From [46] we have the following theorem that gives a probabilistic rate of convergence of this approximation of the regressor. We will use this rate in our analysis of convergence of our learning dynamic in the next section.

Theorem 3.3.1 ([46])**.** *Assume that $f_{\rho^i} \in \Gamma^s_{\rho^i}$ and choose the partition level of resolution $l := l(m_k)$ as the smallest integer such that $a^{l(1+2s)} \geq \frac{m_k}{\log m_k}$. Then, given any $\beta > 0$, there is a constant $\tilde{c}^i := \tilde{c}^i(M^i, \beta, a)$ such that*

$$\text{Prob}\left\{\|f_{\rho^i} - f^i_{\mathbf{z}^i,\Lambda_l}\|_{\rho^i_X} > (\tilde{c}^i + \|f_{\rho^i}\|_\Gamma)\left(\frac{\log m_k}{m_k}\right)^{\frac{s}{2s+1}}\right\} \leq C^i m_k^{-\beta}$$

and

$$\mathrm{E}\left[\|f_{\rho^i} - f_{\mathbf{z}^i,\Lambda_l}\|^2\right] = \left(C^i + \|f_{\rho^i}\|^2_\Gamma\right)\left(\frac{\log m_k}{m_k}\right)^{\frac{s}{2s+1}}$$

where C^i depends only on a and M^i.

Recall in Section 3.2 that we defined ℓ^i_a as the approximation of f_{ρ^i}. We now use ℓ^i_a to denote the approximation of f_{ρ^i} based on the data $\mathbf{z}^i_{1:m_k}$, in other words, $\ell^i_a := f^i_{\mathbf{z}^i,\Lambda_l}$. The structure of Theorem 3.3.1 can seem quite complicated on first encounter. However, it's structure has become a standard feature of many regression approximation problems, and it does make intuitive sense on reflection. The theorem guarantees the existence of a "good set" of samples that generate approximations with small error.

Suppose we denote the good set of samples by

$$G = \text{“Good Set”} = \left\{ \mathbf{z}^i_{1:m_k} : \|f_{\rho^i} - f^i_{\mathbf{z}^i,\Lambda_l}\| \lesssim \left(\frac{\log m_k}{m_k}\right)^{\frac{s}{2s+1}} \right\}$$

and its complement by

$$B = \text{“Bad Set”} = \{\mathbf{z}^i_{1:m_k} \notin G\}$$

This theorem guarantees that the bad set of samples, for which we do not have an explicit upper bound, is a small set:

$$\text{Prob}(B) = \{\mathbf{z}^i_{1:m_k} \notin G\} \leq C^i m_k^{-\beta}$$

As the number of samples m_k gets large, the measure of the set of bad samples decreases to zero asymptotically (with geometric rate β). In addition, as the number of samples m_k increases, the rate of convergence of approximations built from the set of good samples is $\mathcal{O}(\left(\frac{\log m_k}{m_k}\right)^{\frac{s}{2s+1}})$. The probability measure in Theorem 3.3.1 represents the likelihood of

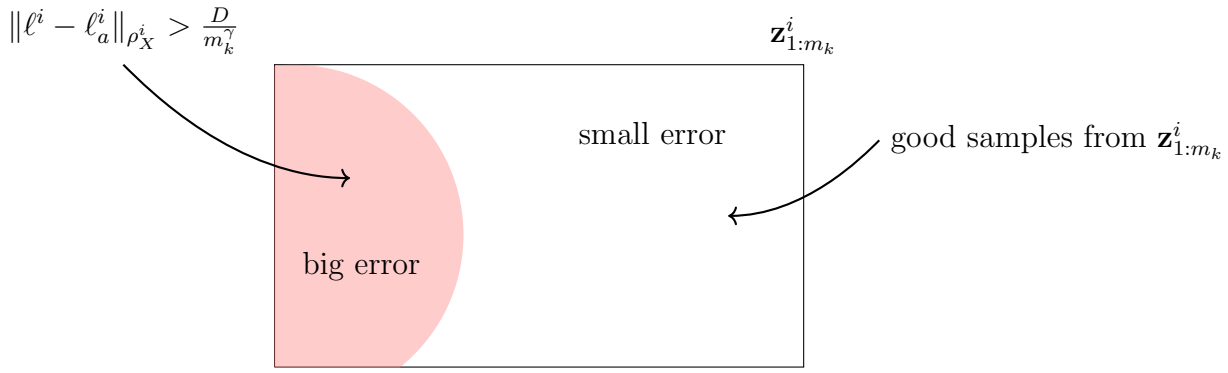


Figure 3.3.2: Interpretation of Probability Measure in Theorem 3.3.1

an estimate ℓ^i_a being in the set of big errors. In other words, if we have good data then $\|\ell^i - \ell^i_a\|_{\rho^i_X} \leq \frac{D}{m_k^\gamma}$. This gives a condition on ℓ^i_a being a good approximation of f_{ρ^i}.

3.4 The Solution and Convergence Rate

From Bacuta [45] the following theorem gives necessary conditions for convergence of the inexact Uzawa algorithm (IUA),

Theorem 3.4.1 (Bacuta). *Let $0 < \alpha < 2/M^2$ and assume that ψ satisfies*

$$\|\psi(\mathbf{r}_k) - A^{-1}\mathbf{r}_k\|_{\mathcal{H}} \leq \delta \|A^{-1}\mathbf{r}_k\|_{\mathcal{H}}, \quad k = 0, 1, \ldots$$

with

$$\delta < \frac{2 - \alpha M^2}{2 + \alpha M^2}$$

and

$$M = \sup_{\mathbf{p} \in Q} \sup_{\mathbf{h} \in \mathcal{H}} \frac{b(\mathbf{h}, \mathbf{p})}{\|\mathbf{p}\|_{\mathcal{Q}} \|\mathbf{h}\|_{\mathcal{H}}} < \infty \tag{3.4.1}$$

$$m = \inf_{\mathbf{p} \in \mathcal{Q}} \sup_{\mathbf{f} \in \mathcal{H}} \frac{b(\mathbf{f}, \mathbf{p})}{\|\mathbf{f}\| \|\mathbf{p}\|} > 0 \tag{3.4.2}$$

Then, IUA converges. There exists $\rho = \rho(\alpha, \delta, m, M) \in (0, 1)$ *such that*

$$(\delta \|\varepsilon_k^{\mathbf{f}}\|_{\mathcal{H}}^2 + \|\varepsilon_k^{\mathbf{p}}\|_{S_0}^2)^{1/2} \leq \rho^k (\delta \|\varepsilon_0^{\mathbf{f}}\|_{\mathcal{H}}^2 + \|\varepsilon_0^{\mathbf{p}}\|_{S_0}^2)^{1/2} \quad \textit{for } k = 1, 2, \ldots$$

Conditions Equations 3.4.1 and 3.4.2 are known as the Babushka-Brezzi conditions and are essential to guarantee a solution of the saddle point problem exists. Applied to our problem we get the following necessary conditions for convergence of the algorithm described in Figure 3.2.3, and the rate of convergence

Theorem 3.4.2. *Suppose that (1)* $f_{\rho^i} \in \Gamma^s_{\rho^i}$ *for* $i = 1, \ldots, n$*, (2) the constants* β *and* $\tilde{c}^i$ *are defined as in Theorem 3.3.1, (3) the approximate solution tolerance* δ *satisfies*

$$\delta < \frac{2 - \alpha \sigma_1^2}{2 + \alpha \sigma_1^2}$$

where σ_j *for* $j = 1, \ldots, (n-1)$ *are the singular values of the constraint matrix* $\mathbf{M}^B$*, (4) the learning gain* α *in the Inexact Uzawa Algorithm satisfies* $0 < \alpha < \frac{2}{\sigma_1^2}$*, and (5) choose the number of samples* m_k *to use in learning step* k *to be large enough so that*

$$\frac{\log m_k}{m_k} \leq \left(\frac{|\|\boldsymbol{\ell}\|_{\mathcal{H}} - \|\mathbf{f}_k - B^* \mathbf{p}_k\|_{\mathcal{H}}|}{\sqrt{\sum_{i=1}^n \left(\tilde{c}^i + \|f_{\rho^i}\|_{\Gamma}\right)^2}} \delta \right)^{\frac{2s+1}{s}}$$

Then the errors in the collective estimates $\varepsilon_k^{\mathbf{f}} := (\mathbf{f}^* - \mathbf{f}_k) + B^*(\mathbf{p}^* - \mathbf{p}_k)$ *and the errors in the multipliers* $\varepsilon_k^{\mathbf{p}} := \mathbf{p}^* - \mathbf{p}_k$ *satisfy*

$$\delta \|\varepsilon_k^{\mathbf{f}}\|_{\mathcal{H}}^2 + \sigma_{n-1}^2 \|\varepsilon_k^{\mathbf{p}}\|_{\mathcal{Q}}^2 \leq \eta^{2k} \left(\delta \|\varepsilon_0^{\mathbf{f}}\|_{\mathcal{H}}^2 + \sigma_1^2 \|\varepsilon_0^{\mathbf{p}}\|_{\mathcal{Q}}^2 \right)$$

for some $\eta < 1$ *with probability at least*

$$\prod_{i=1}^n (1 - C^i m_k^{-\beta})$$

in $(\mathbf{z}_{1:m_k})^n$.

Proof. Let's define,

$$\mathbf{r}_k := \boldsymbol{\ell} - A\mathbf{f}_k - B^*\mathbf{p}_k$$

then the approximation used in the primal form of the learning dynamic is defined via the identity,

$$\hat{\psi}(\mathbf{r}_k) := A^{-1}(\boldsymbol{\ell}_{a,k} - A\mathbf{f}_k - B^*\mathbf{p}_k)$$

Each entry $\boldsymbol{\ell}_{a,k}$ is given by the uniform partition approximations,

$$\boldsymbol{\ell}_{a,k} := \begin{bmatrix} f^1_{\mathbf{z}^1_{1:m_k},\Lambda_{l(m_k)}} \\ f^2_{\mathbf{z}^2_{1:m_k},\Lambda_{l(m_k)}} \\ \vdots \\ f^n_{\mathbf{z}^n_{1:m_k},\Lambda_{l(m_k)}} \end{bmatrix}$$

Using the approximation method for the regressors described in Section 3.3 then we know from Theorem 3.3.1 that our approximation satisfies

$$\|A^{-1}\mathbf{r}_k - \hat{\psi}(\mathbf{r}_k)\|^2_{\mathcal{H}} = \|A^{-1}(\boldsymbol{\ell} - \boldsymbol{\ell}_{a,k})\|^2_{\mathcal{H}}$$

In our case, however, $A = I$ thus the above equation easily simplifies. For the case where $A \neq I$ then we would have to require that A, a linear operator, was a bounded below, thus guaranteeing that it has an inverse.

$$\begin{aligned}
\|\boldsymbol{\ell} - \boldsymbol{\ell}_{a,k}\|^2_{\mathcal{H}} &= \left\| \begin{bmatrix} f_{\rho^1} - f^1_{\mathbf{z}^1_{1:m_k},\Lambda_{l(m_k)}} \\ f_{\rho^2} - f^1_{\mathbf{z}^2_{1:m_k},\Lambda_{l(m_k)}} \\ \vdots \\ f_{\rho^1} - f^n_{\mathbf{z}^n_{1:m_k},\Lambda_{l(m_k)}} \end{bmatrix} \right\|_{\mathcal{H}} \\
&\leq \sum_{i=1}^{n} \left(\tilde{c}^i + \|f_{\rho^i}\|_{\Gamma^s_{\rho^i}} \right)^2 \left(\frac{\log m_k}{m_k} \right)^{\frac{2s}{2s+1}} \\
&\leq \left| \|\boldsymbol{\ell}_{\mathcal{H}}\| - \|f_k + B^*\mathbf{p}_k\|_{\mathcal{H}} \right|^2 \delta^2 \\
&\leq \|\boldsymbol{\ell} - f_k + B^*\mathbf{p}_k\|^2_{\mathcal{H}} \delta^2 \\
&= \|\mathbf{r}_k\|^2_{\mathcal{H}} \delta^2
\end{aligned}$$

According to [45] the Schur complement operator on the space $\mathcal{Q}$ is

$$\begin{aligned}
S &:= BA^{-1}B^* \\
&= BB^*
\end{aligned}$$

where $B : \mathcal{H} \to \mathcal{Q}$. Recall from the previous sections that by definition if $\mathbf{f} \in \mathcal{H}$ and $\mathbf{p} \in \mathcal{Q}$

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{bmatrix}, \quad \mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_{n-1} \end{bmatrix}$$

that $B : \mathbf{f} \mapsto \mathbf{p}$ is expressed as

$$\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_{n-1} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1n} \\ B_{21} & B_{22} & \cdots & B_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ B_{n-1,1} & B_{n-1,2} & \cdots & B_{n-1,n} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{bmatrix}$$

where $B_{ij} : H_j \to Q_i$, so

$$p_i = \sum_{j=1}^{n} B_{ij} f_j$$

Recall from Section 2.3 that the operator B is induced by the matrix $\mathbf{M}^B$, therefore,

$$p_i(\cdot) = \sum_{j=1}^{n} m_{ij}^B f_j(\cdot)$$

From Bacuta [45] the Babushka-Brezzi conditions will hold if we can show that

$$m^2 \|\mathbf{p}\|_{\mathcal{Q}}^2 \leq (S\mathbf{p}, \mathbf{p})_{\mathcal{Q}} \leq M^2 \|\mathbf{p}\|_{\mathcal{Q}}^2$$

where S is the Schur complement operator, M and m are defined as in Theorem 3.4.1 As stated above, the Schur complement operator is $S = BB^*$, which for our problem is $\mathbf{S} = \mathbf{M}^B(\mathbf{M}^B)^\top$. We thus have,

$$\begin{aligned} (S\mathbf{p}, \mathbf{p})_{\mathcal{Q}} &= \sum_{j=1} \left(\sum_{i=1} \mathbf{S}_{ji} p_i, p_j \right)_Q \\ &= \sum_{j=1} \sum_{i=1} \mathbf{S}_{ji} (p_i, p_j)_Q \\ &= \sum_{j=1} \sum_{i=1} \mathbf{S}_{ji} \int \big(p_j(\xi), p_i(\xi)\big)_{\mathbb{R}} d\xi \\ &= \int \sum_{j=1} \left(\sum_{i=1} \mathbf{S}_{ji} p_i(\xi), p_j(\xi) \right) d\xi \\ &= \int (\mathbf{S}\mathbf{p}(\xi), \mathbf{p}(\xi))_{\mathbb{R}^{n-1}} \, d\xi \end{aligned} \tag{3.4.3}$$

Looking at the singular value decomposition of $\mathbf{S}$ where $\mathbf{M}^B = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$.

$$\begin{aligned} \mathbf{S} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top \\ &= \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}\mathbf{U}^\top \\ &= \mathbf{U} \begin{bmatrix} \diagdown & & \\ & \sigma_i^2 & \\ & & \diagdown \end{bmatrix} \mathbf{U}^\top, \qquad i = 1, \ldots, n-1 \end{aligned}$$

where recall that $V \in \mathbb{R}^{n\times n}$ is orthogonal and $\Sigma \in \mathbb{R}^{(n-1)\times n}$ is the matrix of singular values of $\mathbf{M}^B$, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{n-1}$. Substituting this expression into Equation 3.4.3,

$$\begin{aligned}
(S\mathbf{p},\mathbf{p})_{\mathcal{Q}} &= \int (\mathbf{S}\mathbf{p}(\xi),\mathbf{p}(\xi))_{\mathbb{R}^{n-1}}\, d\xi = \int \left(\mathbf{U}\begin{bmatrix} \ddots & & \\ & \sigma_i^2 & \\ & & \ddots \end{bmatrix}\mathbf{U}^\top \mathbf{p}(\xi),\mathbf{p}(\xi)\right)_{\mathbb{R}^{n-1}} d\xi \\
&\leq \int \left(\mathbf{U}\begin{bmatrix} \ddots & & \\ & \sigma_1^2 & \\ & & \ddots \end{bmatrix}\mathbf{U}^\top \mathbf{p}(\xi),\mathbf{p}(\xi)\right)_{\mathbb{R}^{n-1}} d\xi \\
&= \sigma_1^2 \int \left(\cancel{\mathbf{U}\mathbf{U}^\top}^{I}\mathbf{p}(\xi),\mathbf{p}(\xi)\right)_{\mathbb{R}^{n-1}} d\xi \\
&= \sigma_1^2 \|\mathbf{p}\|_{\mathcal{Q}}^2
\end{aligned}$$

We thus have,

$$(S\mathbf{p},\mathbf{p})_{\mathcal{Q}} \leq \sigma_1^2 \|\mathbf{p}\|_{\mathcal{Q}}^2$$

By a similar argument we can show $\sigma_{n-1}^2\|\mathbf{p}\|_{\mathcal{Q}}^2 \leq (S\mathbf{p},\mathbf{p})_{\mathcal{Q}}$. We have now shown that

$$\sigma_{n-1}^2\|\mathbf{p}\|_{\mathcal{Q}} \leq (S\mathbf{p},\mathbf{p})_{\mathcal{Q}} \leq \sigma_1^2\|\mathbf{p}\|_{\mathcal{Q}}$$

and $M^2 = \sigma_1^2$. Define the norm $\|\cdot\|_S$ on $\mathcal{Q}$ by $\|\mathbf{p}\|_S := (S\mathbf{p},\mathbf{p})$ for all $\mathbf{p} \in \mathcal{Q}$. The conclusion of the theorem follows from Theorem 3.4.1

$$(\delta\|\varepsilon_k^{\mathbf{f}}\|_{\mathcal{H}}^2 + \|\varepsilon_k^{\mathbf{p}}\|_S^2)^{1/2} \leq \eta^k(\delta\|\varepsilon_0^{\mathbf{f}}\|_{\mathcal{H}}^2 + \|\varepsilon_0^{\mathbf{p}}\|_S^2)^{1/2}$$

when we employ the equivalence of norms on $\|\cdot\|_{\mathcal{Q}}$ and the norm $\|\cdot\|_S$ induced by the Schur complement operator. □

Chapter 4

Numerical Studies

The numerical examples in this section have a common structure. Each agent i, for $i = 1, 2, \ldots, n$, performs sampling that is characterized by a probability measure ρ^i that is defined on $Z = X \times Y$. The samples collected by agent i, for $i = 1, 2, \ldots, n$, represent noisy information regarding the external field $f : X \to Y$. As described in the Section 2.3.2 the sampling measure for a single agent can always be decomposed as the product

$$\rho^i(z^i) = \rho^i(x^i, y^i) = \rho^i_X(x^i)\rho^i(y^i|x^i)$$

In each of our numerical examples we must specify ρ^i for each agent $i = 1, 2, \ldots, n$. We will construct the sampling measure for each agent in terms of a "piecewise uniform" distribution that we discuss below.

Consider dividing the set X into $X = A \cup \bar{A}$ where the distribution $\rho^i_X(\cdot)$ is uniform over each region A and $\bar{A}$. Then $\rho^i_X(B)$ for any measurable subset $B \subseteq X$ is defined as,

$$\rho^i_X(B) = \int_B c_1 \mathbb{1}_A(x)dx + \int_B c_2 \mathbb{1}_{\bar{A}}(x)dx$$

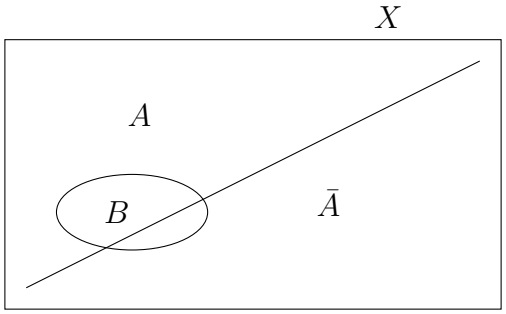


Figure 4.0.1: Division of Set X into Subsets with Local Support

where,

$$\mathbb{1}_A(x) = \begin{cases} 1 & \text{for } x \in A \\ 0 & \text{for } x \notin A \end{cases}$$

The coefficients c_1, c_2 must be defined such that $\int \rho_X(\cdot)dx = 1$. It is clear that the choice above is equivalent to constructing a piecewise constant probability density function(pdf) $p(x)$ such that

$$P(x \in B) = \rho_X(B) = \int_B p(x)dx$$

Thus we assign $p(x) = \frac{p}{\rho_X^i(A)}$ for $x \in A$ and we assign $p(x) = \frac{(1-p)}{\rho_X^i(\bar{A})}$ for $x \in \bar{A}$. In other words, we have chosen

$$\rho_X^i(B) = \int_B \left[\frac{p}{\rho_X^i(A)} \mathbb{1}_A(x) + \frac{1-p}{\rho_X^i(\bar{A})} \mathbb{1}_{\bar{A}}(x) \right] dx$$

We have shown how we define $\rho_X^i(\cdot)$ and now turn to the conditional probability, $\rho^i(\cdot|x^i)$. We assume the following relationship between the samples y^i and a fixed x^i,

$$y^i = f(x^i) + w^i$$

where w^i is noise that is assumed to be a random variable with Gaussian distribution $N(0, \sigma)$. Since $w^i \sim N(0, \sigma)$ then for a fixed x^i the difference

$$y^i - f(x^i)$$

has the same distribution as w^i and y^i is also a random variable. Therefore, for any set B and fixed $x^i \in X$,

$$P(w^i \in B) = \int_B \frac{1}{c} e^{-\frac{{w^i}^2}{2\sigma^2}} dw^i = P(y^i - f(x^i) \in B) \tag{4.0.1}$$

where c is the normalizing constant $\frac{1}{\sigma\sqrt{2\pi}}$.

We can now rewrite Equation 4.0.1 as,

$$P(w^i \in B) = P((y^i - f(x^i)) \in B) = P(y^i \in (B + f(x^i))) = \int\limits_{B+f(x^i)} g^i(\eta)d\eta \tag{4.0.2}$$

where $g^i(\eta)$ is a probability density function. Introducing a change of variable, let $\eta \in (B + f(x^i))$ and thus $\eta = \xi + f(x^i)$, $\xi \in B$. Substituting η into Equation 4.0.2 we now have,

$$\int_B g(\xi + f(x^i))d\xi = \int_B \frac{1}{c} e^{-\frac{\xi^2}{2\sigma^2}} d\xi$$

Thus $g^i(\xi + f(x^i)) = \frac{1}{c}\, e^{-\frac{\xi^2}{2\sigma^2}}$ and hence,

$$g^i(\eta) = \frac{1}{c}\, e^{-\frac{(\eta - f(x^i))^2}{2\sigma^2}}$$

Given the relationship $y^i = f(x^i) + w^i$ we have shown that for a fixed x^i the probability of y^i is,

$$\rho^i(dy^i|x^i) = g^i(y^i)dy^i$$

4.1 Function Approximation

As mention in Section 3.3 we use a multi-grid method to approximate the coefficients of our function $f(x)$. The approximations are generated over a uniform partition of the domain X. This partition is refined to have $2^{l/2}$ cells in the x_1 and in the x_2 directions respectively. Over each cell we define a basis function. To best understand how we construct this basis we start with the one-dimensional case and then progress up to the two-dimensional case.

For the basis functions $\phi(x)$ we have the following two-scale relationship in 1D

$$\phi(x) = \phi(2x) + \phi(2x - 1)$$

where,

$$\phi(2x) = \begin{cases} 1 & x \in [0, \frac{1}{2}] \\ 0 & \text{otherwise} \end{cases} \tag{4.1.1}$$

and

$$\phi(2x - 1) = \begin{cases} 1 & x \in [\frac{1}{2}, 1] \\ 0 & \text{otherwise} \end{cases} \tag{4.1.2}$$

This relationship is depicted in Figure 4.1.1.

In compact form it is expressed as

$$\phi(x) = \sqrt{2} \sum_{k=0}^{1} a_k \phi(2x - k)$$

where $a_k = \frac{1}{\sqrt{2}}$ is called the filter coefficient. In Figure 4.1.1 we refined from grid level $l = 0$ to $l = 1$. If we refine further from $l = 1$ to $l = 2$ then we would have as in Figure 4.1.2. The general formulation for going from grid level l to grid level $l + 1$ is

$$\phi(2^l - m) = \sqrt{2} \sum_{k=0}^{1} \phi(2^{l+1}x - 2m - k)$$

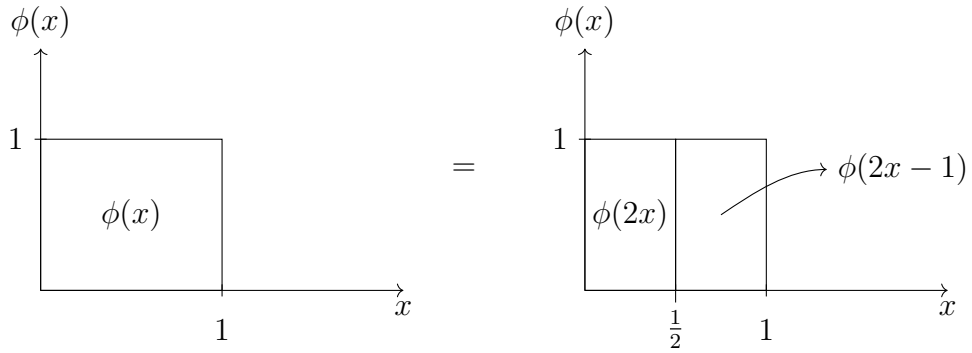


Figure 4.1.1: Two-Scale Basis Relationship in 1D

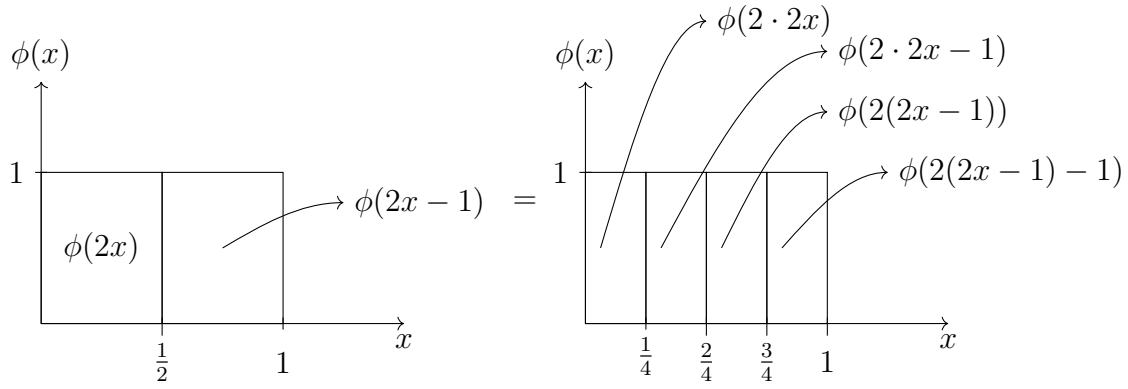


Figure 4.1.2: Basis Functions Refined from Grid $l = 1$ to $l = 2$

However, we require that our basis be orthonormal and therefore we premultiply both sides of the equation by $2^{l/2}$. We now have the following orthonormal basis for our function approximations,

$$\begin{aligned}
\underbrace{2^{l/2}\phi((2^l x - m)}_{\phi_{l,m}(x)} &= 2^{l/2}\sqrt{2}\sum_{k=0}^{1} a_k \phi(2^{l+1}x - 2m - k) \\
&= \sum_{k=0}^{1} a_k \underbrace{2^{\frac{l+1}{2}}\phi(2^{l+1}x - 2m - k)}_{\phi_{l+1,2m+k}(x)} \\
\phi_{l,m}(x) &= \sum_{k=0}^{1} a_k \phi_{l+1,2m+k}(x)
\end{aligned}$$

Proof. To show that this basis is orthonormal we simply take the inner product of two basis

elements,

$$\int \phi_{l,m}(x)\phi_{l,n}(x)dx = \int 2^{l/2}\phi(2^l x - m)\, 2^{l/2}\phi(2^l x - n)dx$$
$$= 2^l \int \phi(2^l x - m)\phi(2^l x - n)dx$$

By our definitions in Equations 4.1.1 and 4.1.2 we know that this integration will be zero everywhere except where $m = n$. Consider the case where $m = n$, and introduce a change of variable, $\xi = 2^l x - m$, thus $d\xi = 2^l dx$,

$$2^l \int \phi(2^l x - m)\phi(2^l x - m)dx = 2^l \int \phi(\xi)\phi(\xi)2^{-l}d\xi = 1$$

□

The extension to our 2D scenario is simple enough, we define the 2-dimensional basis as the product of two one dimensional basis functions.

$$\phi_{l,(m,n)}(x_1, x_2) = \left(\sum_{k=0}^{1} a_k \phi_{l+1,2m+k}(x_1)\right)\left(\sum_{j=0}^{1} a_j \phi_{l+1,2n+j}(x_2)\right)$$
$$\phi_{l,(m,n)}(x_1, x_2) = \sum_{k=j=0}^{1} a_k a_j \phi_{l+1,(2m+k,2n+j)}(x_1, x_2) \tag{4.1.3}$$

We now have all the necessary definitions to explain how approximations of the function coefficients are projected or injected from one level to another. Define $S_l = \text{span}\{\phi_{l,(r,s)}\}$ and suppose that $f \in S_l$. We know that we can express $f(x_1, x_2)$ as

$$f(x_1, x_2) = f(x_1, x_2)$$
$$\sum_{r,s} F_{l,(r,s)}\phi_{l,(r,s)}(x_1, x_2) = \sum_{u,v} F_{l+1,(u,v)}\phi_{l+1,(u,v)}(x_1, x_2)$$

Substituting in our expression in Equation 4.1.3, we can re-write the above as,

$$\sum_{u,v} F_{l+1,(u,v)}\phi_{l+1,(u,v)}(x_1, x_2) = \sum_{r,s} F_{l,(r,s)} \sum_{p,q} a_p a_q \phi_{l+1,(2r+p,2s+q)}(x_1, x_2)$$

Taking the inner product of the above with an arbitrary basis element $\phi_{l+1,(m,n)}(\mathbf{x})$, where

$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top$,

$$\begin{aligned}
\sum_{u,v} F_{l+1,(u,v)} \phi_{l+1,(u,v)}(\mathbf{x}) \phi_{l+1,(m,n)}(\mathbf{x}) &= \sum_{r,s} F_{l,(r,s)} \sum_{p,q} a_p a_q \phi_{l+1,(2r+p,2s+q)}(\mathbf{x}) \phi_{l+1,(m,n)}(\mathbf{x}) \\
\sum_{u,v} F_{l+1,(u,v)} \delta_{(m,n),(u,v)} &= \sum_{r,s} F_{l,(r,s)} \sum_{p,q} a_p a_q \delta_{(m,n),(2r+p,2s+q)} \\
F_{l+1,(m,n)} &= \sum_{r,s} F_{l,(r,s)} \sum_{\xi,\eta} a_{\xi-2r} a_{\eta-2s} \delta_{(m,n),(\xi,\eta)} \\
F_{l+1,(m,n)} &= \sum_{r,s} F_{l,(r,s)} a_{m-2r} a_{n-2s} \qquad (4.1.4)
\end{aligned}$$

where we used the change of variable $\xi = 2r + p$ and $\eta = 2s + q$. This gives us the injection from S_l to S_{l+1}. Going in the opposite direction, from the $l+1$ grid level to the l, we use the standard projection procedure. The vector $f_l - f_{l+1}$, where $f_l \in S_l$ and $f_{l+1} \in S_{l+1}$, must be orthogonal to S_l.

$$(f_l - f_{l+1}) \perp S_l \implies (f_l - f_{l+1}, \phi_{l,(u,v)}) = 0$$

Using the properties of norms,

$$(f_l, \phi_{l,(u,v)}) - (f_{l+1}, \phi_{l,(u,v)}) = 0$$

Expanding f_l and f_{l+1} in the usual way and using the properties of norms,

$$\begin{aligned}
\left(\sum_{i,j} F_{l,(i,j)} \phi_{l,(i,j)}, \phi_{l,(u,v)} \right) &= \left(\sum_{h,k} F_{l+1,(h,k)} \phi_{l+1,(h,k)}, \phi_{l,(u,v)} \right) \\
\sum_{i,j} F_{l,(i,j)} \left(\phi_{l,(i,j)}, \phi_{l,(u,v)} \right) &= \sum_{h,k} F_{l+1,(h,k)} \left(\phi_{l+1,(h,k)}, \phi_{l,(u,v)} \right) \\
F_{l,(i,j)} \delta_{(u,v),(i,j)} &= \sum_{h,k} F_{l+1,(h,k)} \left(\phi_{l+1,(h,k)}, \sum_{m,n} a_m a_n \phi_{l+1,2u+m,2v+n} \right) \\
F_{l,(u,v)} &= \sum_{h,k} F_{l+1,(h,k)} \sum_{\xi,\eta} a_{\xi-2u} a_{\eta-2v} \delta_{(h,k),(\xi,\eta)} \\
F_{l,(u,v)} &= \sum_{h,k} F_{l+1,(h,k)} a_{h-2u} a_{k-2v}
\end{aligned}$$

Again, we used the change of variable $\xi = 2u + m$ and $\eta = 2v + n$. We summarize these formulations in Figure 4.1.3. It is important to note that the subscripts of the filter coefficients a can only take values $\{0, 1\}$. This dictates the limits of the summation indices r, s, h, k for each (m, n) and (u, v) pair.

Injection

$$F_{l+1,(m,n)} = \sum_{r,s} F_{l,(r,s)} a_{m-2r} a_{n-2s}$$

Projection

$$F_{l,(u,v)} = \sum_{h,k} F_{l+1,(h,k)} a_{h-2u} a_{k-2v}$$

Figure 4.1.3: Mapping of Coefficients Between Grid Levels

4.2 Simulations

For our simulations we ran a two agent and a four agent case. We used the tree topology depicted in Example 5 for our communication network. We ran both simulations using a large number of samples, $m_k = 450,000$ for the purpose of obtaining verification of the rate of convergence in the large sample limit. The agents take samples in non-overlapping regions shown in Figure 4.2.1. In this figure we depict the piecewise constant probability density

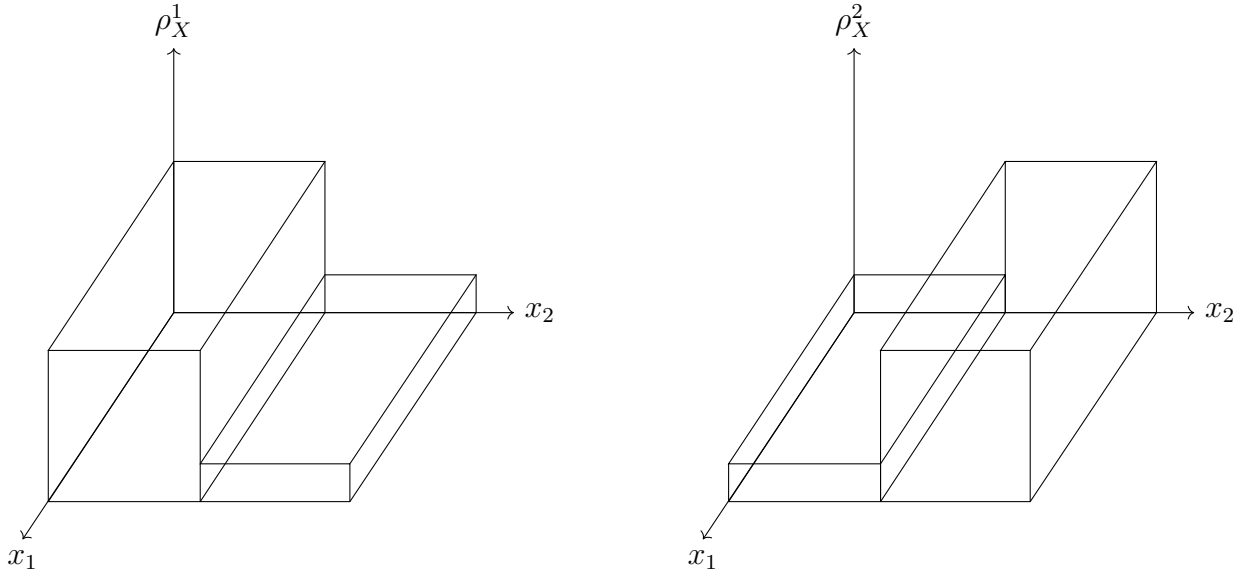


Figure 4.2.1: Sampling Densities for Two Agents

functions that describe the sampling carried out by each agent. The densities are defined so that each agent takes 90% of its measurements in its primary region and the remaining 10% in its complement. The unknown external field that we are approximating is,

$$f(x_1, x_2) = \left(x_1 - \frac{1}{2}\right)^2 + \left(x_2 - \frac{1}{2}\right)^2$$

whose true form is shown in Figure 4.2.2.

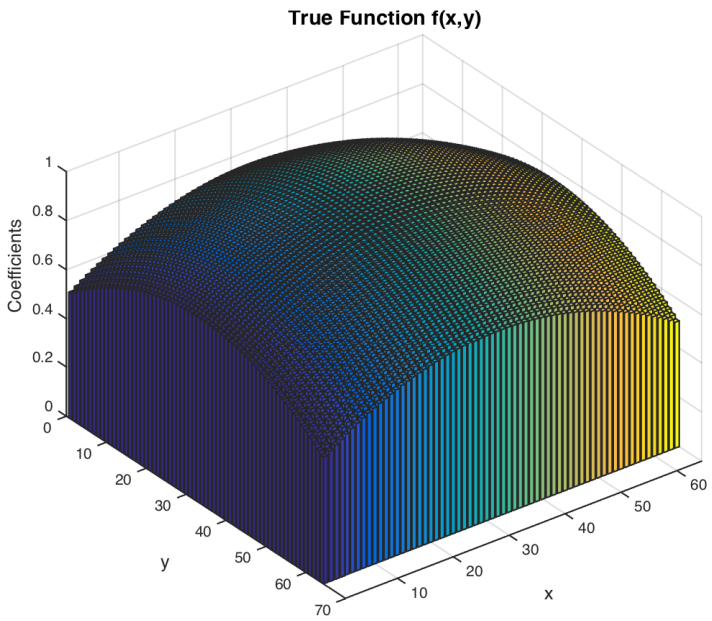
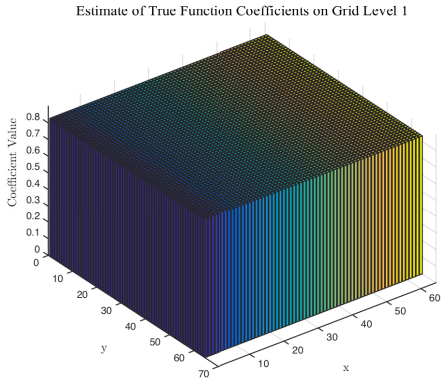


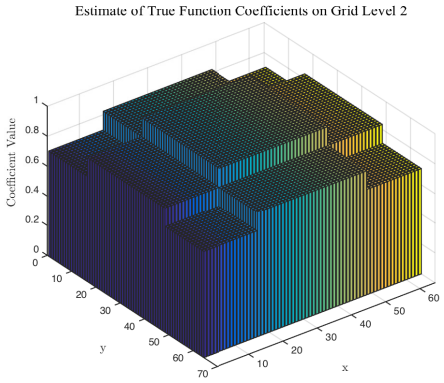
Figure 4.2.2: Coefficients of the True Function $f(x, y)$

Inside our code we have two iterative loops that are executed. We have an inner loop for the inexact Uzawa algorithm that iterates $j \in \mathbb{N}$ times until the error in the function coefficients approximately converges on the current grid level. Then we have an outer iteration that iterates through the grid levels $l = 1, 2, \ldots$. Each iteration l corresponds to a new epoch k, when a new set of m_k measurements are made.

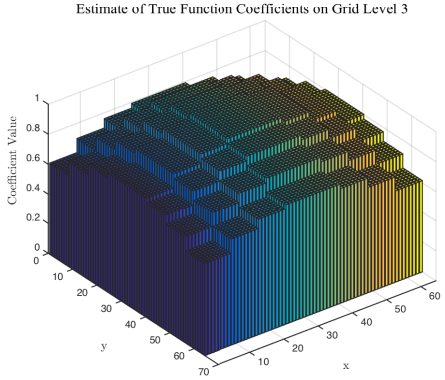
Below are the results for the two agent case. We first plot each agents' approximation of f for grid levels $l = 1, 2, \ldots, 6$. As can be seen in Figures 4.2.3 and 4.2.4 the two agents' approximation are almost identical.



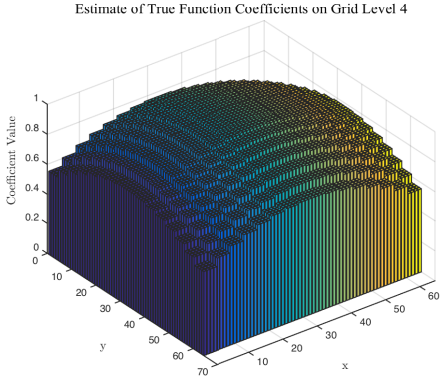
(a) Grid Level 1



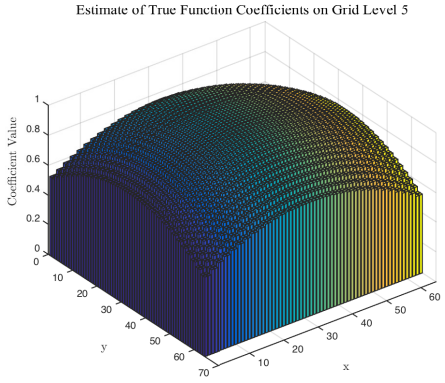
(b) Grid Level 2



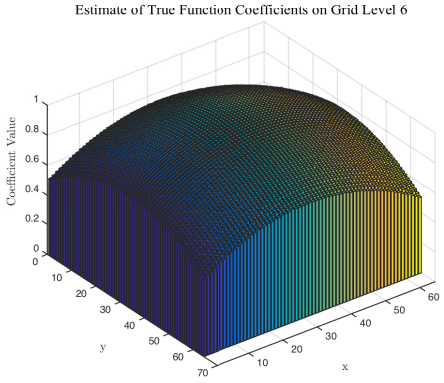
(c) Grid Level 3



(d) Grid Level 4

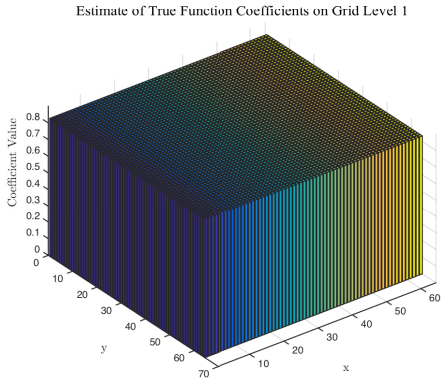


(e) Grid Level 5

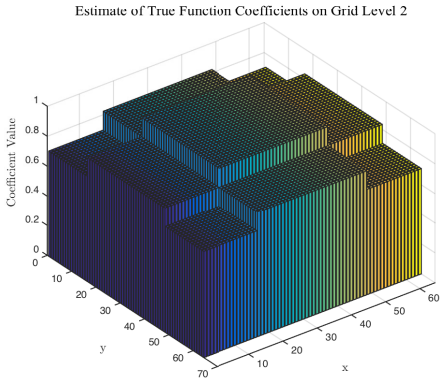


(f) Grid Level 6

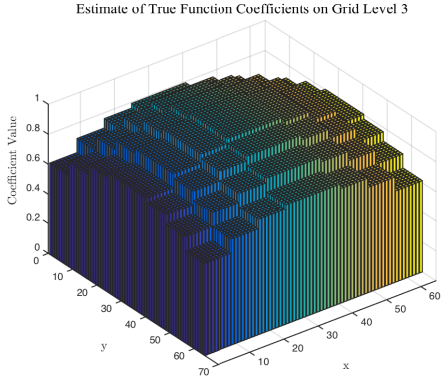
Figure 4.2.3: Refinement of Approximation of Function Coefficients for Agent One



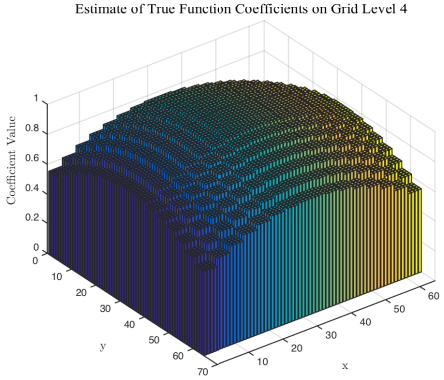
(a) Grid Level 1



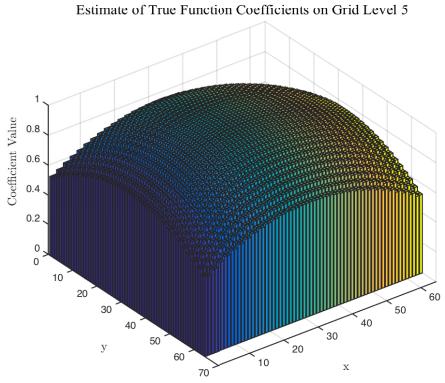
(b) Grid Level 2



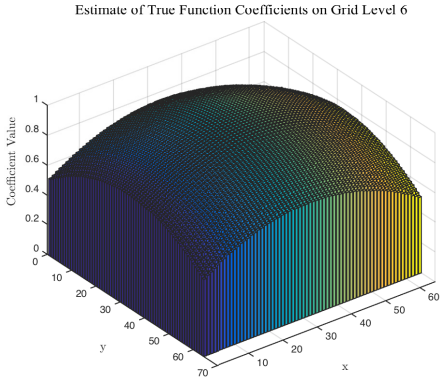
(c) Grid Level 3



(d) Grid Level 4



(e) Grid Level 5



(f) Grid Level 6

Figure 4.2.4: Refinement of Approximation of Function Coefficients for Agent Two

Next, in Figure 4.2.5, we plot the log of the collective error squared

$$\delta\|\varepsilon_k^{\mathbf{f}}\|_{\mathcal{H}}^2 + \sigma_{n-1}^2\|\varepsilon_k^{\mathbf{P}}\|_{\mathcal{Q}}^2$$

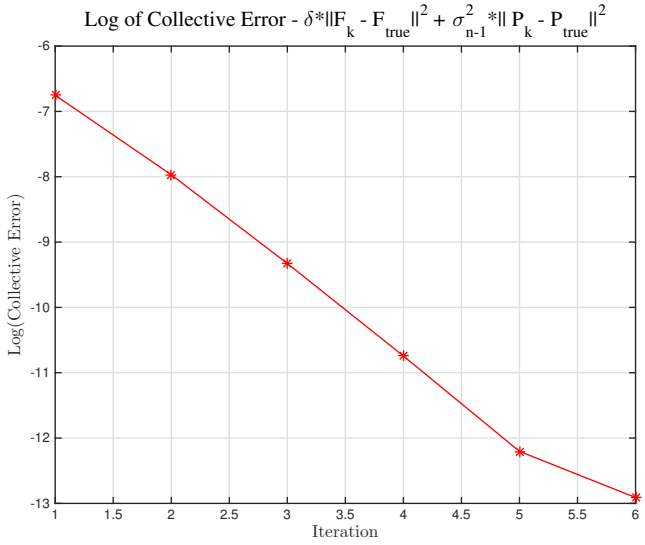


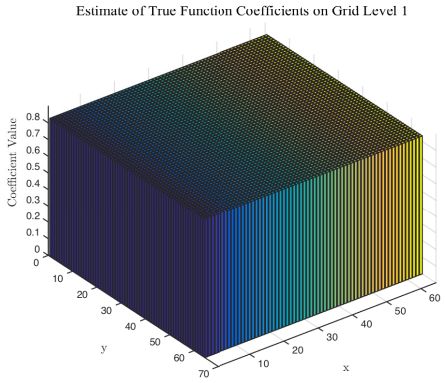
Figure 4.2.5: Log of the Collective Error Squared Calculated at Each Level

We can approximate our geometric rate of convergence η^k from the slope of this line. Taking two points we approximate the slope as

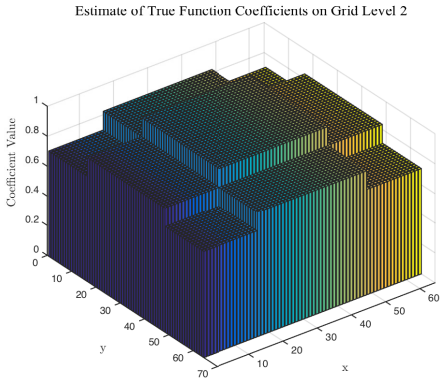
$$2\eta = |-1.369|$$
$$\eta = 0.68$$

Thus our rate of convergence is 0.68^k. This tells us that with grid level $l = 6$ our collective error will be less than or equal to $1/100$ of our initial error.

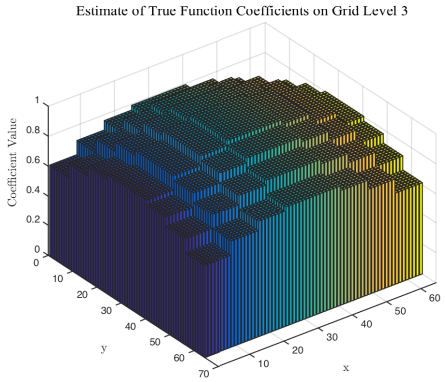
For the four agent case we have the following results as shown in Figures 4.2.6 - 4.2.9. They are close to identical to those of the two agent case and give us a similar rate of convergence. Since we have just shown the calculation for the two agent case we will not repeat it for the four agent case.



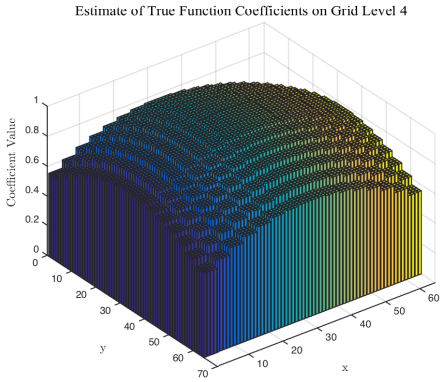
(a) Grid Level 1



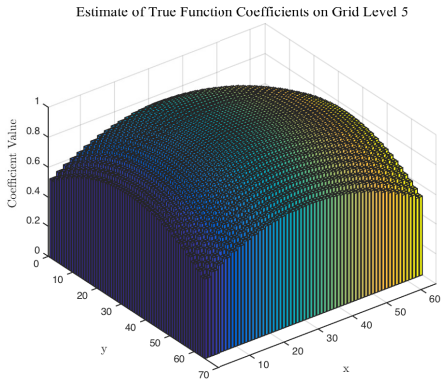
(b) Grid Level 2



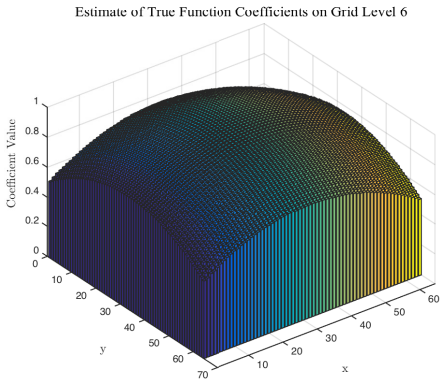
(c) Grid Level 3



(d) Grid Level 4

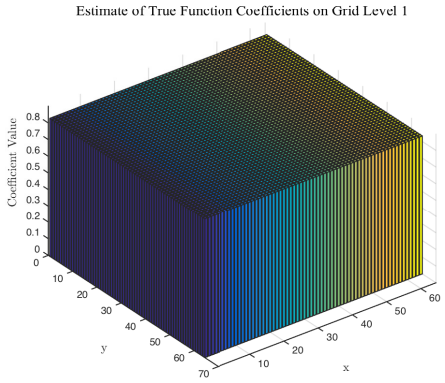


(e) Grid Level 5

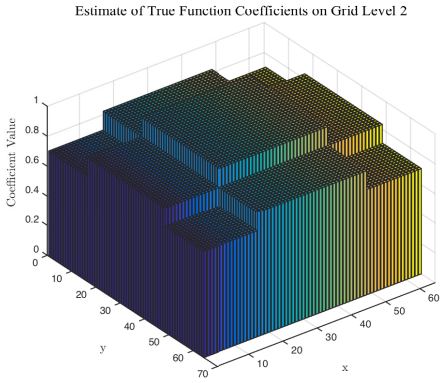


(f) Grid Level 6

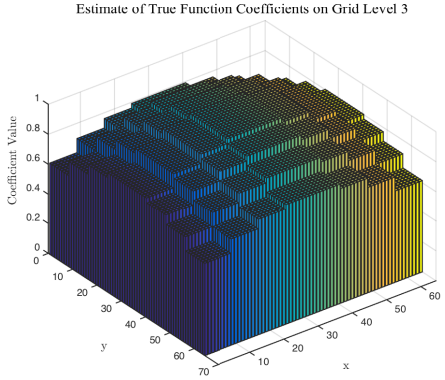
Figure 4.2.6: Refinement of Approximation of Function Coefficients for Agent One



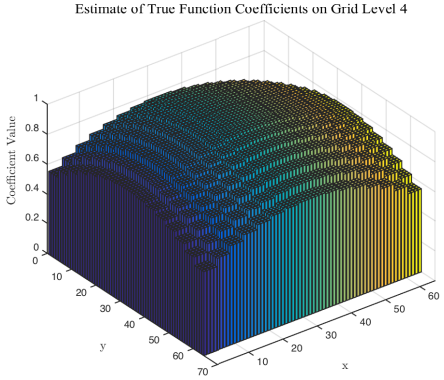
(a) Grid Level 1



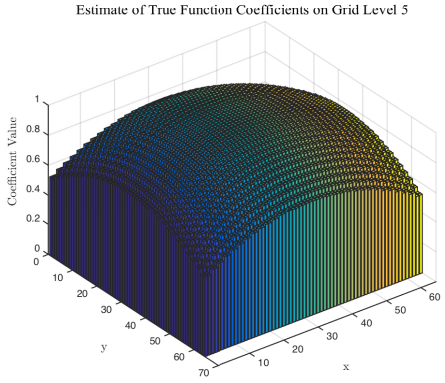
(b) Grid Level 2



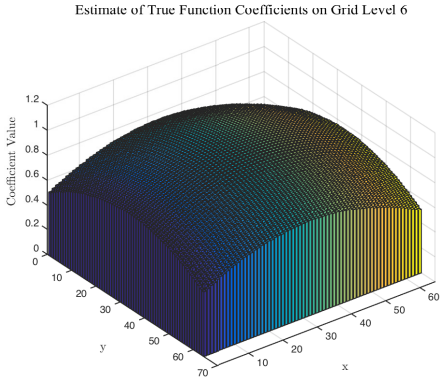
(c) Grid Level 3



(d) Grid Level 4

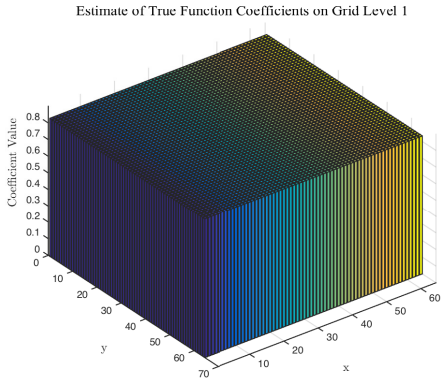


(e) Grid Level 5

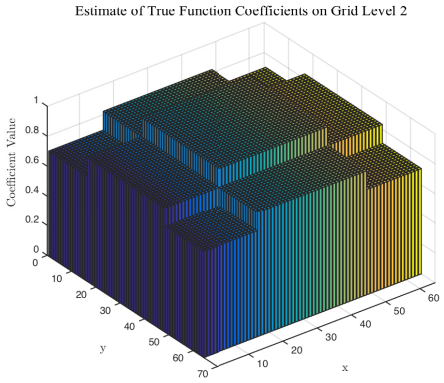


(f) Grid Level 6

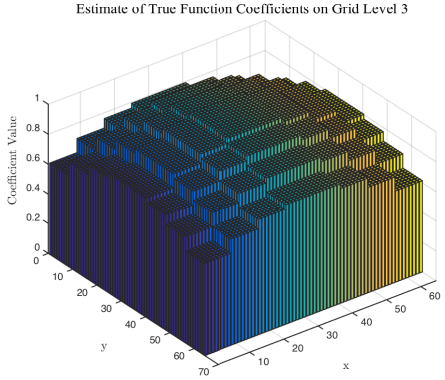
Figure 4.2.7: Refinement of Approximation of Function Coefficients for Agent Two



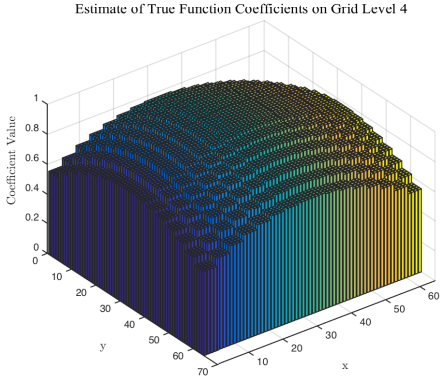
(a) Grid Level 1



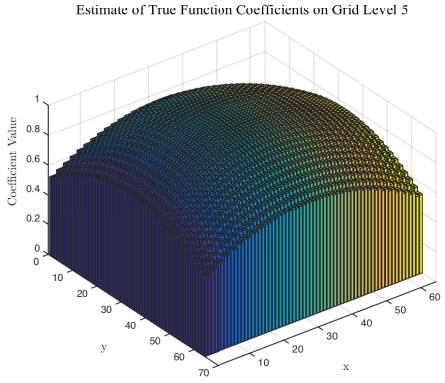
(b) Grid Level 2



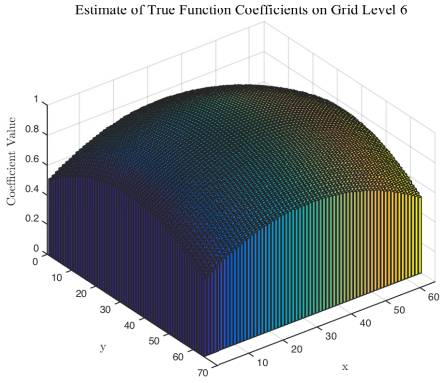
(c) Grid Level 3



(d) Grid Level 4

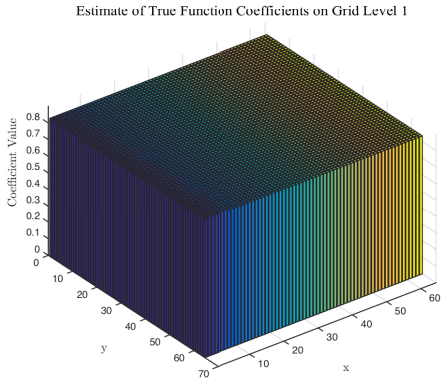


(e) Grid Level 5

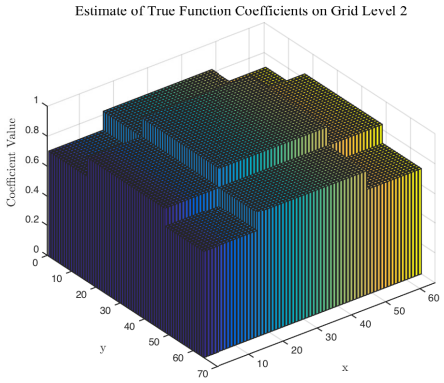


(f) Grid Level 6

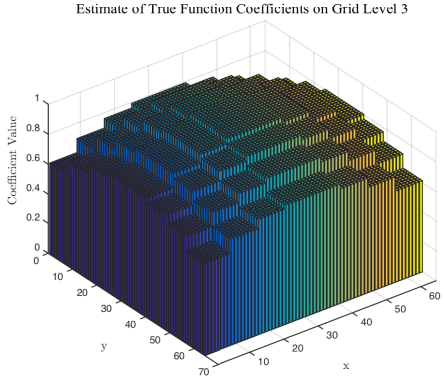
Figure 4.2.8: Refinement of Approximation of Function Coefficients for Agent Three



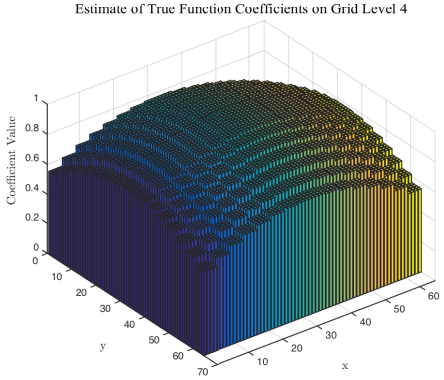
(a) Grid Level 1



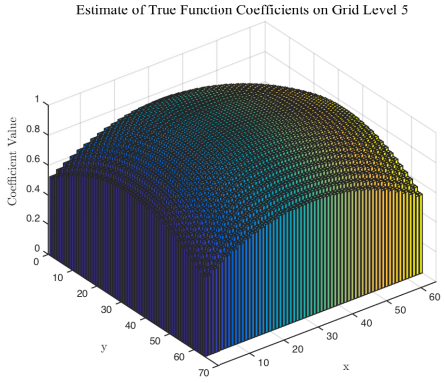
(b) Grid Level 2



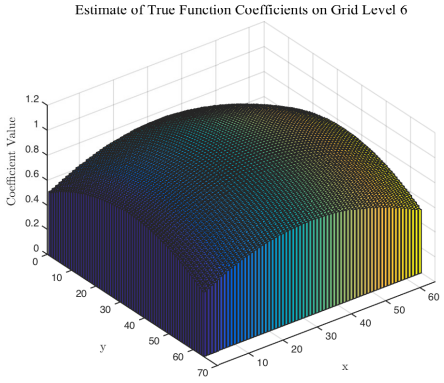
(c) Grid Level 3



(d) Grid Level 4



(e) Grid Level 5



(f) Grid Level 6

Figure 4.2.9: Refinement of Approximation of Function Coefficients for Agent Four

We conducted a simulation to demonstrate our theoretical results. In our theorem we stated that one would achieve convergence in the function f and the Lagrange multipliers p at the rate proposed with probability

$$\prod_{i=1}^{n}(1 - C^i m_k^{-\beta})$$

where C^i depends upon σ_1 and a. We have shown this to be true with the two/four agent simulation over $m_k = 450,000$ measurements.

Chapter 5

Conclusion and Future Work

In this thesis we have derived a rate of convergence (in space) for a distribution-free learning theory problem with consensus. We have employed aspects of statistical learning theory, approximation theory, and consensus to achieve this goal. To recap, we formulate our learning problem with consensus as a saddle point problem. We pose the saddle point problem into operator form where the solution is the ordered pair $(\mathbf{f}^*, \mathbf{p}^*)$. Both $\mathbf{f}^* \in \mathcal{H}$ and $\mathbf{p}^* \in \mathcal{Q}$ reside in the cross product spaces $\mathcal{H} = H^1 \times H^2 \times \cdots \times H^n$ and $\mathcal{Q} = Q \times Q \times \cdots \times Q := Q^{n-1}$ where H and Q are the infinite dimensional spaces $\mathsf{L}^2(\rho_X)$ and $\mathsf{L}^2(m)$, respectively. Our analysis requires that an equivalence of norms on $\|\cdot\|_H$ and $\|\cdot\|_Q$ exists. This assumption puts a restriction on the sampling measures that we can use, and it allows us to use a common basis for H and Q.

To generate an algorithm suitable for computations, we next construct a finite dimensional approximation of the infinite dimensional form of the saddle point problem. We expand the elements f^i, p^i, and ℓ^i into their basis representations and then take inner products of them with arbitrary basis elements. We first did this with each element in its respective basis. This gave us a matrix representation of the saddle point problem. Alternatively, we go through the same procedure but using a common basis. We now have an operator and a matrix representation of the saddle point problem.

At this point we still have not addressed the fact that we cannot calculate ℓ^i since it is a function of the unknown probability measure ρ_X^i. We address this by employing a probabilistic implementation of the inexact Uzawa algorithm as described in [44]. First, using the Riesz map and bilinear forms we cast our problem into the dual operator form of the saddle point problem. Next, using the dual operator form and the deterministic rate of convergence derived by Bacuta in [45], we derive a probabilistic rate of convergence. A critical step in our approach defines the approximate inverse operator ψ. The inexact Uzawa algorithm constructs a sequence of convergent approximations to the solutions of the saddle point problem on grid level l where we have approximated our unknowns as piecewise constant functions on the grid Λ_l. We now use Theorem 3.4.1 to establish our rate of convergence, and we use

Theorem 3.3.1 to describe the probability of achieving such a rate.

Lastly, we implement the coefficient form of the inexact Uzawa algorithm and show our results for the two agent and four agent case.

This topic of research still leaves many open issues to be explored. We hope in the future to look at the influence of the graph topology on the rate of convergence and it's accompanying probability. We would also like to explore the performance when there is a much larger number of interacting agents. Finally, we would like to look into the impact of dropped communication between agents on our results.

Appendix A

Functional Analysis

Definition 4. *A topological space $(X, \mathcal{T})$ is a set X and collection of subsets $\mathcal{T}$ of X which satisfy the following three axioms:*

1. $\emptyset \in \mathcal{T}$, $X \in \mathcal{T}$
2. *If $\{T_\alpha\} \subset \mathcal{T}$ is an arbitrary collection of sets of $\mathcal{T}$, then $\bigcup_\alpha T_\alpha \in \mathcal{T}$*
3. $A, B \in \mathcal{T} \implies A \cap B \in \mathcal{T}$

If more than one topological space is being considered then for clarity we may denote it by $(X, \mathcal{T}_X)$.

Definition 5. *A **normed linear vector space** (NVS) is a vector space X on which there is a defined real-valued function which maps each element $x \in X$ into a real number $\|x\|$ called the norm of x. The norm must satisfy*

1. $\|x\| \geq 0 \quad \forall x \in X,\ \|x\| = 0$ *iff* $x = 0$
2. $\|x + y\| \leq \|x\| + \|y\|$ *for each* $x, y \in X$
3. $\|\alpha x\| = |\alpha| \|x\|$ *for all scalars* α *and each* $x \in X$.

Definition 6. *Suppose we have a real-valued function ρ defined on the set $V \times V$ and we stipulate that ρ obey the following axioms for all $x, y, z \in V$*

1. $\rho(x, y) \geq 0$

2. $\rho(x,y) = \rho(y,x)$

3. $\rho(x,y) + \rho(y,z) \geq \rho(x,z)$

4. $\rho(x,y) = 0 \iff x = y$

Then ρ is called a ***metric*** *for V. If only the first three properties are satisfied then ρ is a* ***semi-metric*** *for the space V.*

Definition 7. *A* ***semi/metric space*** *is a non-empty set X of elements with a metric ρ, denoted by the set and the metric, (X, ρ)*

A metric space is a subset of a topological space, therefore it inherits all the properties listed above for a topological space. It has subsets which are open and the space X and its complement $\{\emptyset\}$ are both open.

Definition 8. *A sequence $\{x_n\}$, $n \in \mathbb{N}$ in a metric space is called a* ***Cauchy sequence*** *if for each $\epsilon > 0$ there exists N such that for all $m, n \geq N$, $\rho(x_m, x_n) < \epsilon$ or that $\rho(x_m, x_n) \to 0$.*

Definition 9. *A space V is called a* ***complete*** *space if every convergent Cauchy sequence $\{x_n\}$ in V, has its limit in the space, $\{x_n\} \to x \in V$*

Definition 10. *An* ***inner product space*** *is a normed vector space along with the operation $(\cdot,\cdot) : H \times H \to \mathbb{R}$ or $\mathbb{C}$ that satisfies the following properties for $u, v, w \in H$ and $\alpha \in \mathbb{R}$ or $\mathbb{C}$,*

1. $(u,v) = \overline{(v,u)}$ *conjugate symmetry*

2. $(u+v,w) = (u,w) + (v,w)$ *linear in the first argument*

3. $(\alpha u, v) = \alpha(u,v)$ *linear in the first argument*

4. $(u,u) \geq 0$ *and* $(u,u) = 0$ *if and only if* $u = 0$

Definition 11. *A complete inner product space is a* ***Hilbert space****.*

Definition 12. *For an operator $T : U \to V$ where U, V are normed vector spaces, the* ***operator norm*** *$\|T\|$ is defined as*

$$\|T\| = \sup_{\mathbf{u} \neq 0} \frac{\|T\mathbf{u}\|_V}{\|\mathbf{u}\|_U} = \sup_{\|\mathbf{u}\|_U = 1} \|T\mathbf{u}\|_V$$

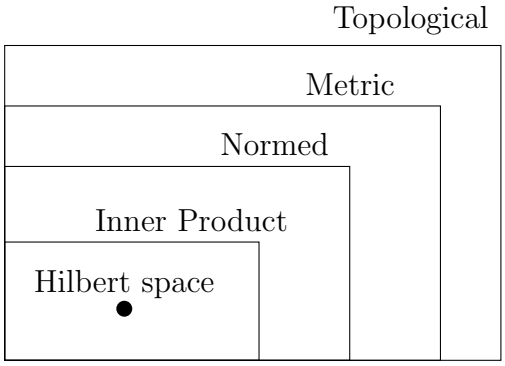


Figure A.0.1: Graphical Representation of Relationship Between Topological Spaces

Definition 13. *A bounded linear operator is* ***bounded below*** *provided that there exists a positive constant c such that,*

$$\|T\mathbf{u}\|_V \geq c\|\mathbf{u}\|_U \quad \forall \mathbf{u} \in U$$

Theorem A.0.1. *A bounded linear operator that is bounded below has a bounded inverse.*

Definition 14. *Let X be a normed space and let $L(X, \mathbb{R})$ be the set of linear functionals on X. Then the set $\mathcal{L}(X, \mathbb{R}) = \{f \in L(X, \mathbb{R}) : \|f\| < \infty\}$ of bounded linear functionals on X constitutes a normed space with norm,*

$$\|f\| = \sup_{\substack{x \in X \\ x \neq 0}} \frac{|f(x)|}{\|x\|}$$

and is called the ***dual space*** *of X and is denoted by X^*.*

We highlight the fact that linear functionals that are bounded are thus continuous. Recall that a function(al) is continuous at x_0 if for every ϵ there exists a δ such that if $\|x - x_0\| < \delta$ then that implies $\|f(x) - f(x_0)\| < \epsilon$. It is continuous everywhere if this applies for all $x_0 \in X$. Using the linearity property of $f \in \mathcal{L}(X, \mathbb{R})$ we can easily show that boundedness implies continuity.

Proof. Assume that $\|x - x_0\| < \delta$ for an arbitrary $x_0 \in X$. Then from linearity and boundedness we have,

$$\|f(x) - f(x_0)\| = \|f(x - x_0)\| \leq M\|x - x_0\| \leq M\delta$$

Set $\delta = \epsilon/M$ and we have $\|f(x) - f(x_0)\| < \epsilon$. Since we choose an arbitrary x_0, then a linear bounded function(al) is uniformly continuous. □

Theorem A.0.2 (Riesz Representation Theorem)**.** *Let X be a Hilbert space. For any functional $f \in X^*$ there exists a unique $u \in X$ such that $f(x) = (x, u)_X$ for all $x \in X$. Additonally,*

$$\|f\|_{X^*} = \|u\|_X$$

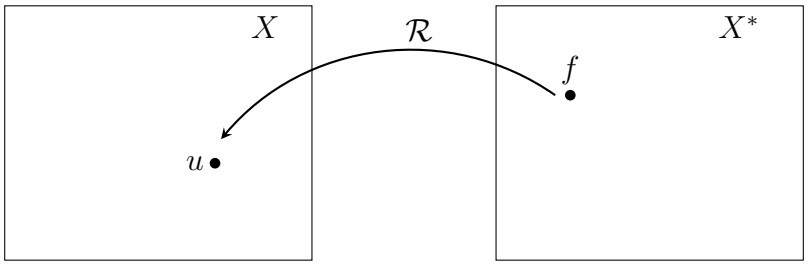


Figure A.0.2: Riesz Map $\mathcal{R}_u$

The proof of this theorem can be found in many textbooks and the reader is directed toward [47].

Let $\mathcal{R} : X^* \to X$ represent the function that maps $f \mapsto u$ where

$$\mathcal{R}f = u$$

Since both X and X^* are Hilbert spaces, then we can use the algebraic and topological structure of these spaces to say a few things about $\mathcal{R}$. We can define a norm on $\mathcal{R}$ and note that

1. $\|\mathcal{R}\|$ is bounded

2. $\mathcal{R}$ is bijective

3. $\mathcal{R}$ is linear

4. $\mathcal{R}$ is continuous and has a continuous bounded inverse $\mathcal{R}^{-1} : X \to X^*$

5. $\mathcal{R}$ is an isometry

Proof. To prove (1) note that X^* is the space of bounded linear operators hence $f \in X^*$ is bounded and to prove boundedness we want to prove that $\|\mathcal{R}f\|_X \leq M\|f\|_{X^*}$.

$$\|Rf\|_X = \|u\|_X = \|f\|_{X^*}$$

which we know from Theorem A.0.2. If we take a close look at the norm of f,

$$\|f\|_{X^*} = \sup_{\substack{x \in X \\ x \neq 0}} \frac{|f(x)|}{\|x\|_X} = \sup_{\substack{x \in X \\ x \neq 0}} \frac{(u, x)_X}{\|x\|_X} \leq \sup_{\substack{x \in X \\ x \neq 0}} \frac{\|u\|_X \|x\|_X}{\|x\|_X} = \|u\|_X$$

where we have used the Riesz Representation theorem and the Cauchy-Schwartz inequality above. Therefore,

$$\|\mathcal{R}f\|_X \leq M\|u\|_X = M\|f\|_{X^*}$$

for $M = 1$. Note that in the examination of $\|f\|_{X^*}$ we have shown the first direction in the proof that $\|f\|_{X^*} = \|u\|_X$.

To prove (2) is trivial, it comes directly from Theorem A.0.2.

To prove (3), recall that the dual space X^* is a normed vector space and hence for two elements $f, g \in X^*$, $f + g \in X^*$. This translates into $f(x) + g(x) = (f + g)(x)$. We will use this and the Riesz Representation theorem to prove that $\mathcal{R}(f + g) = \mathcal{R}f + \mathcal{R}g$. First let,

$$\mathcal{R}f = u$$
$$\mathcal{R}g = v$$

Then using the properties of inner products the following holds,

$$(f + g)(x) = f(x) + g(x) = (u, x)_X + (v, x)_X = (u + v, x)_X$$

The Riesz Representation theorem states that for every element $h \in X^*$ there is a unique element $w \in X$. We have just shown that for the element $f + g$ the unique element in X is $u + v$. Therefore, $\mathcal{R} : f + g \mapsto u + v$. In other words, $\mathcal{R}(f + g) = u + v = \mathcal{R}f + \mathcal{R}g$.

To prove the first part of (4) is easy, it is a bounded linear operator and as we showed above that implies it is continuous. To prove the second part we simply need to show that $\mathcal{R}$ is bounded below. We have already done this in our examination of $\|f\|$

$$\|\mathcal{R}f\|_X = \|u\|_X \geq M\|f\|_{X^*}$$

for $M = 1$.

To prove property (5) is a truly a matter of proving the other direction for showing that $\|f\|_{X^*} = \|u\|_X$. From the continuity of f we know that it is a bounded function. And since f is a linear operator we have the following relation,

$$|f(u)| = |(u, u)_X| = \|u\|_X\|u\|_X \leq M\|u\|_X$$

The M above comes from a set of all possible M that meet the above requirement. We can, however, choose the $M = M_{min}$. This choice of M is our operator norm $\|f\|_{X^*} = \inf\{M : \|fu\|_{\mathbb{R}} \leq M\|u\|_X\}$. This gives us,

$$\|u\|_X\|u\|_X \leq M\|u\|_X = \|f\|_{X^*}\|u\|_X$$
$$\|u\|_X \leq \|f\|_{X^*}$$

It was not necessary to prove (5) since it came straight out of the Riesz Representation theorem. However, since the proof of the theorem was omitted, (5) was proven to show the powerfulness of the theorem. □

Definition 15. *The **duality pairing** is defined on the cross product space $X^* \times X$ as,*

$$\langle f, x \rangle_{X^* \times X} = f(x) \quad \forall x \in X, \quad \textit{for each } \ f \in X^*$$

This is commonly used to state the Riesz Representation as,

$$\langle f, x \rangle_{X^* \times X} = (u, x)_X \quad \forall x \in X$$

Perhaps it is easiest to appreciate how all the pieces connect from the commutative diagram in Figure A.0.3.

$$\begin{array}{ccc} (\mathcal{R}f, x)_X & \Leftrightarrow & \langle f, x \rangle_{X^* \times X} \\ \Updownarrow & & \Updownarrow \\ (u, x)_X & \Longleftrightarrow & f(x) \end{array}$$

Figure A.0.3: Commutative Diagram of Riesz Representation Theorem

We now look at the adjoint operator which is used frequently in this thesis. Consider the cross product spaces $\mathcal{H} = H^1 \times H^2 \times \cdots \times \cdots H^n$ and $\mathcal{Q} = Q^1 \times Q^2 \times \cdots \times Q^m$ and let $\mathbb{T}$ be a bounded linear operator, $\mathbb{T} : \mathcal{H} \to \mathcal{Q}$. Then $\mathbb{T}$ can be written as a matrix of bounded linear operators,

$$\mathbf{q} = \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^m \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ T_{21} & T_{22} & \cdots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{m1} & T_{m2} & \cdots & T_{nm} \end{bmatrix} \begin{bmatrix} h^1 \\ h^2 \\ \vdots \\ h^n \end{bmatrix} = \mathbb{T}\mathbf{h}$$

where $T_{ij} : H^j \to Q^i$.

Definition 16. *The **adjoint operator** of a bounded linear operator $\mathbb{T} : \mathcal{H} \to \mathcal{Q}$, denoted by $\mathbb{T}^*$ where $\mathcal{H}$ and $\mathcal{Q}$ are Hilbert spaces, is defined to be the unique operator that satisfies the following relationship,*

$$(\mathbb{T}\mathbf{h}, \mathbf{q})_{\mathcal{Q}} = (\mathbf{h}, \mathbb{T}^*\mathbf{q})_{\mathcal{H}} \quad \forall \mathbf{h} \in \mathcal{H}, \ \forall \mathbf{q} \in \mathcal{Q} \tag{A.0.1}$$

This definition, however, requires a little fleshing out in the case where $\mathbb{T}$ acts on product spaces. For any product space U, we define taking the inner product by $(\mathbf{v}, \mathbf{u})_U = \sum_i (v_i, u_i)_{U^i}$. Keeping this in mind, we take the left-hand side of Equation A.0.1 and express it as,

$$(\mathbb{T}\mathbf{h}, \mathbf{q})_{\mathcal{Q}} = \sum_i \left(\sum_j T_{ij} h_j, q_i \right)_{Q^i} = \sum_i \sum_j (T_{ij} h_j, q_i)_{Q^i}$$

From the adjoint definition then it follows,

$$(T_{ij} h_j, q_i)_{Q^i} = (h_j, (T_{ij})^* q_i)_{H^i}$$

and hence,

$$\sum_i \sum_j (T_{ij} h_j, q_i)_{Q^i} = \sum_i \sum_j (h_j, (T_{ij})^* q_i)_{H^i} \tag{A.0.2}$$

From our knowledge of product spaces we know that the left-hand side of Equation A.0.2 can be written as,

$$\left(\begin{bmatrix} h^i \\ h^2 \\ \vdots \\ h^n \end{bmatrix}, \mathbb{T}^* \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^m \end{bmatrix} \right)_{\mathcal{H}} = (\mathbf{h}, \mathbb{T}^* \mathbf{q})_{\mathcal{H}}$$

Since $\mathbb{T}^*$ is a linear operator, then we know its standard structure as,

$$\mathbb{T}^* \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^m \end{bmatrix} = \begin{bmatrix} T_{11}^* & T_{12}^* & \cdots & T_{1m}^* \\ T_{21}^* & T_{22}^* & \cdots & T_{2m}^* \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1}^* & T_{n2}^* & \cdots & T_{nm}^* \end{bmatrix} \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^m \end{bmatrix}$$

We can represent $\mathbb{T}^* = [T_{kl}^*]$. But from the property of product spaces we have,

$$\sum_i \sum_j (h_j, (T_{ij})^* q_i)_{H^i} = \sum_j \left(h_j, \sum_i (T_{ij})^* q_i \right)_{H^i} = \begin{bmatrix} (T_{11})^* & (T_{21})^* & \cdots & (T_{m1})^* \\ (T_{12})^* & (T_{22})^* & \cdots & (T_{m2})^* \\ \vdots & \vdots & \ddots & \vdots \\ (T_{1n})^* & (T_{2n})^* & \cdots & (T_{mn})^* \end{bmatrix} \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^m \end{bmatrix}$$

Therefore, we can now see that $T_{kl}^* = (T_{lk})^*$.

The above demonstrates the relationship between the entries of the adjoint of a bounded linear operator $\mathbb{T}$, which is a matrix of bounded linear operators, and the entries of the adjoint operator $\mathbb{T}^*$ which is a matrix of adjoint operators. If we were to extract one of the entries from $\mathbb{T}$, T_{ij}, since it is a bounded linear operator then we can represent this as a matrix. We are now interested in how the entries of the matrix representation of T_{ij} relate to the matrix representation of its adjoint $(T_{ij})^*$. For simplicity, we drop the subscripts and define $T : H^i \to H^j$. By the definition of the adjoint above,

$$(Tf, g)_{H^j} = (f, T^* g)_{H^i} \quad \forall f \in H^i \text{ and } \forall g \in H^j$$

We can expand our elements f and g into their respective bases and from the inner product define the entries of the matrix representation of T and T^*.

$$\left(T\sum_i \alpha_i\phi_i, \sum_j \beta_j\phi_j\right) = \left(\sum_i \alpha_i\phi_i, T^*\sum_j \beta_j\phi_j\right)$$
$$\sum_{i,j} \alpha_i\beta_j \left(T\phi_i, \phi_j\right) = \sum_{i,j} \alpha_i\beta_j \left(\phi_i, T^*\phi_j\right)$$
$$\boldsymbol{\beta}^{\mathrm{T}}\mathbf{T}\boldsymbol{\alpha} = \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{T}^*\boldsymbol{\beta}$$
$$\beta_j T_{ij}\alpha_i = \alpha_i (T^*)_{ji}\beta_j$$

α_i and β_j are scalars therefore it must be true that,

$$T_{ij} = (T^*)_{ji}$$

where the above are the scalar entries of the matrices $\mathbf{T}$ and $\mathbf{T}^*$. Since this holds for all i, j then it clearly follows that,

$$\mathbf{T}^\top = \mathbf{T}^*$$

Similar to the adjoint operator is another operator called the transpose operator.

Definition 17. *Let $B : X \to Y$ be bounded linear operator where X and Y are normed spaces. Then the* ***transpose operator*** *$B^\top : Y^* \to X^*$ of B is defined by*

$$\left\langle B^\top g, x\right\rangle_{X^*\times X} = \langle g, Bx\rangle_{Y^*\times Y} \quad \textit{for } g \in Y^*$$

We can relate the adjoint operator in Definition 16 to that of the transpose operator defined above. If in Definition 17 we let $X = H_1$ and $Y = H_2$ be Hilbert spaces. From the definition of the transpose operator we have the following relations:

$$\begin{aligned} B^\top g &= f \\ g(Bx) &= f(x) \end{aligned} \qquad\qquad f \in H_1^*,\ g \in H_2^*$$

Since f and g are functionals that act on Hilbert spaces, then from the Riesz Representation theorem,

$$\begin{aligned} f(x) &= (u, x)_{H_1} \\ g(y) &= (v, y)_{H_2} \end{aligned} \tag{A.0.3}$$

We define our Riesz maps for our relations in Equations A.0.3 as,

$$\begin{aligned} \mathcal{R}_1 f_0 &= u_0 \qquad && \text{where } \mathcal{R}_1 : H_1^* \to H_1 \\ \mathcal{R}_2 g_0 &= v_0 && \text{where } \mathcal{R}_2 : H_2^* \to H_2 \end{aligned}$$

From our description of the Riesz map above we know that it is linear and if we look at the commutative diagram in Figure A.0.4, we can see that we can form a linear operator from the composition of linear operators which we label as,

$$B^\top = \mathcal{R}_2 B^* \mathcal{R}_1^{-1} \tag{A.0.4}$$

We will now show that the $B^\top$ we defined in Equation A.0.4 is our transpose operator. From the Riesz Representation theorem and the definition of the transpose we immediately have,

$$(v_0, Bx)_{H_2} = \langle g_0, Bx \rangle_{H_2^* \times H_2} = \left\langle B^\top g_0, x \right\rangle_{H_1^* \times H_1} = (u_0, x)_{H_1}$$

And from the definition of the adjoint operator we get the result we want,

$$\left(\mathcal{R}_1 B^\top g_0, x\right)_{H_1} =$$

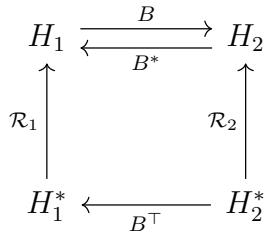


Figure A.0.4: Commutative Diagram Showing Relation Between B^* and $B^\top$

Appendix B

Probability

Definition 18. *A* ***σ-field*** *of subsets of a set* Ω *is a collection* $\mathcal{F}$ *of subsets of* Ω *that has* $\emptyset$ *as a member and is closed under complementation and countable unions. The sets that are members of the σ-field are said to be* ***measurable with respect to*** $\mathcal{F}$*, or* $\mathcal{F}$***-measurable****, or just* ***measurable*** *if the σ-field is understood from the context.*

It should be noted that since a σ-field is closed under complementation and $\emptyset \in \mathcal{F}$, then $\Omega \in \mathcal{F}$. Additionally, from De Morgan's Laws, we have that $\mathcal{F}$ is closed under countable intersections.

Definition 19. *A σ-field generated by the collection of open sets is called a* ***Borel σ-field*** *which will be denoted by* $\mathcal{B}$*.*

Definition 20. *A* ***probability measure*** P *on a σ-field* $\mathcal{F}$ *of subsets of a set* Ω *is a function from* $\mathcal{F}$ *to the unit interval* $[0, 1]$ *such that* $P(\Omega) = 1$ *and*

$$P\left(\bigcup_{m=1}^{\infty}\right) = \sum_{m=1}^{\infty} P(A_m)$$

for each pairwise disjoint sequence $(A_m : m = 1, 2, 3, \dots)$ *of members of* $\mathcal{F}$*. Because* P *satisfies this summation condition it is said to be* ***countably additive****. For* $A \in \mathcal{F}$*,* $P(A)$ *is the* ***probability of*** A*.*

When first introduced to a probability measure it is often introduced as an idealized relative frequency, a function that approximates the relative frequency of an event in a large number of trials. While the definition above is more formal than the one we are first introduced to, it should be clear that they describe the same mathematical phenomena. Additionally, it should be noted that a probability measure is a distribution (defined below) and vice-versa, so often times the terms are used interchangeably.

Definition 21. *A **probability space** is a triple $(\Omega, \mathcal{F}, P)$, where Ω is a set, $\mathcal{F}$ is a σ-field of subsets of Ω, and P is a probability measure on $\mathcal{F}$. The set Ω is called the **sample space** and its members are called **sample points**. The members of $\mathcal{F}$ are called **events**.*

Definition 22. *A **measurable space** is a pair $(\Psi, \mathcal{G})$, where Ψ is a nonempty set and $\mathcal{G}$ is a σ-field of subsets of Ψ.*

Definition 23. *Let $(\Omega, \mathcal{F})$ and $(\Psi, \mathcal{G})$ be measurable spaces. A **measurable function** from $(\Omega, \mathcal{F})$ to $(\Psi, \mathcal{G})$ is a function $X : \Omega \to \Psi$ such that $X^{-1}(B) \in \mathcal{F}$ for every $B \in \mathcal{G}$. When a probability measure P is attached to the measurable space $(\Omega, \mathcal{F})$, so that $(\Omega, \mathcal{F}, P)$ is a probability space, X is then called a **random variable** from $(\Omega, \mathcal{F}, P)$ to $(\Psi, \mathcal{G})$.*

Definition 24. *Let X be a random variable from a probability space $(\Omega, \mathcal{F}, P)$ to a measurable space $(\Psi, \mathcal{G})$. For $B \in \mathcal{G}$ let $Q(B) = P(X^{-1}(B))$. Then $(\Psi, \mathcal{G}, Q)$ is a probability space. The probability Q is called the **distribution** of the random variable X and is said to be induced by X.*

Definition 25. *A real-valued function $F(\cdot)$ defined on $\mathbb{R}$ is called a **distribution function** for $\mathbb{R}$ if it is increasing and right-continuous and satisfies,*

$$\lim_{x \to -\infty} F(x) = 0 \quad \textit{and} \quad \lim_{x \to \infty} F(x) = 1$$

Proposition 2. *Let Q be a probability measure on $(\mathbb{R}, \mathcal{B})$. Then the function $F(x) = Q((-\infty, x])$ is a distribution function.*

For an expected values we can think of it as we do with our initial introduction to a probability measure, as a function that represents idealized averages of random variables. First, though, we will need to introduce the indicator function.

Definition 26. *Let $(\Omega, \mathcal{F})$ be a measurable space. For each subset A of Ω, define a function I_A on Ω by*

$$I_A(\omega) = \begin{cases} 1 & \text{if } \omega \in A \\ 0 & \text{otherwise} \end{cases}$$

Definition 27. *Let X be a random variable defined on a probability space $(\Omega, \mathcal{F}, P)$ and having the simple form*

$$X = \sum_{j=1}^{n} c_j I_{C_j}$$

for some distinct real constants c_j and events C_j that constitute a partion Ω. Then the expectation of X, $E(X)$, is

$$E(X) = \sum_{j=1}^{n} c_j P(C_j)$$

Appendix C

Graph Theory

In examining much of the literature on consensus and estimation that references graph theory a couple of terms pop up that are not used in the literature on graph theory. In this appendix we want to establish proper terminology for describing graphs and their properties.

We defined a graph in the main text as a set $G = (V, E)$ of vertices and the edges that join any two pair of vertices. The following two definitions are standard terms to describe aspects of vertices and edges.

Definition 28. *A pair of vertices that are joined by an edge are called* ***adjacent****.*

Definition 29. *If an edge e_i joins vertices u and v, then e_i is said to be* ***incident*** *to u and v, respectively.*

This next definition is simple enough to understand and hence its usefulness gets easily overlooked.

Definition 30. *A* ***path*** *is an ordered list of vertices in a graph where two vertices are adjacent if and only if they are consecutive in the list.*

We use this definition to define a connected graph. Note that in terms of connectedness a graph is either connected or disconnected. To place additional qualifiers on these terms is meaningless. Often in the literature the term "fully connected" is used. The adverb "fully" provides no additionally information about the graph and should be omitted.

Definition 31. *A graph G is* ***connected*** *if for any two vertices $v_1, v_2 \in V$ of G there exists a path between them. A graph is* ***disconnected*** *if this condition is not meet.*

So far in the discussion of graphs we have assumed that the edges have no direction associated with them.

Definition 32. *A graph $G = (V, E)$ where the edges are ordered pairs (v_i, v_j) is a* ***directed graph****, sometimes called a* ***digraph****. The* ***underlying graph*** *is the graph of the directed graph where edge direction is ignored.*

Appendix D

Learning Theory

Rosenblatt's learning algorithm assumed a neural network of n neurons with the typology of a pyramid. The outputs of the base layer of neurons was the inputs of the next layer of inputs, and so on until the top layer, consisting of a single neuron, received the inputs of the layer below it. Learning in this model meant finding the appropriate $\mathbf{w}_i$, $i = 1, 2, \ldots, n$, for all neurons using the given training data set. Rosenblatt's scheme was to fix the coefficients for all neurons except the last neuron and thus simply find the weight vector of this single neuron. Geometrically this means transforming the input space X to a new space Z and use the training data to separate this new space. Take an element from the training data set $|S| = s$, $(\mathbf{z}^k, y^k) \in \{(\mathbf{z}^l, y^l\}$, $1 \leq l \leq s$ and feed this into the perceptron. Let $\mathbf{w}(k)$ be the coefficient vector of the last neuron at time step k, then

$\mathbf{w}(1) = 0$

1. if $(y_{k+1}, \mathbf{z}_{k+1})$ is classified correctly, i.e.,

   i. $y_{k+1} = 0 \implies \mathbf{w}(k) \bullet \mathbf{z}_{k+1} < 0$

   ii. $y_{k+1} = 1 \implies \mathbf{w}(k) \bullet \mathbf{z}_{k+1} > 0$

   then don't change $\mathbf{w}(k)$, and $\mathbf{w}(k+1) = \mathbf{w}(k)$.

2. if $(y_{k+1}, \mathbf{z}_{k+1})$ is misclassified, i.e,

   (a) $y_{k+1} = 0 \implies \mathbf{w}(k) \bullet \mathbf{z}_{k+1} > 0$

   (b) $y_{k+1} = 1 \implies \mathbf{w}(k) \bullet \mathbf{z}_{k+1} < 0$

   then update the coefficient vector as follows:

   for case (a) $\mathbf{w}(k+1) = w(k) - z_{k+1}$

   for case (b) $\mathbf{w}(k+1) = w(k) + z_{k+1}$

Definition 33. *The* ***VC dimension of a set of indicator functions*** $Q(z, \alpha)$, $\alpha \in \Lambda$ *is the maximum number* h *of vectors,* $z_1, z_2, \ldots z_h$*, that can be separated into two classes in all* 2^h *possible ways using functions of the set. Note that the indicator functions naturally separate a set of vectors into two subsets: those vectors for which the indicator function outputs a* 0 *and those for which it outputs a* 1*.*

Definition 34. *The* ***VC dimension of a set of real functions****. Let* $A \leq Q(z, \alpha) \leq B$*,* $\alpha \in \Lambda$*, be a set of real functions bounded by constants* $A \in \mathbb{R}$ *and* $B \in \mathbb{R}^+$*.*

Let us consider along with the set of real functions $Q(z, \alpha)$, $\alpha \in \Lambda$*, the set of indicators*

$$I(z, \alpha, \beta) = \theta\{Q(z, \alpha) - \beta\}, \quad \alpha \in \Lambda,\ \beta \in (A, B)$$

where $\theta(z)$ *is the step function*

$$\theta(z) = \begin{cases} 0 & \text{if } z < 0, \\ 1 & \text{if } z \geq 0. \end{cases}$$

The VC dimension of a set of real functions $Q(z, \alpha)$, $\alpha \in \Lambda$ *is defined to be the VC dimension of the set of corresponding indicators above with parameters* $\alpha \in \Lambda$ *and* $\beta \in (A, B)$*.*

Appendix E

Constructions

Each of the entries of $\mathbb{A}$ is an inner product in $L^2(\rho_X)$. Recall that for $f, g \in L^2$, $(f,g)_{L^2(\rho_X)} = \int_X f \cdot g\rho(dx)$. The explicit form of the matrix $\mathbf{A}$ is listed below.

$$\mathbf{A} = \begin{bmatrix} \begin{bmatrix} \int \phi_1\phi_1 A_{11} & \cdots & \int \phi_1\phi_j A_{11} \\ \int \phi_2\phi_1 A_{11} & \cdots & \int \phi_2\phi_j A_{11} \\ \vdots & \vdots & \vdots \\ \int \phi_r\phi_1 A_{11} & \cdots & \int \phi_r\phi_j A_{11} \end{bmatrix} \begin{bmatrix} \int \phi_1\phi_1 A_{12} & \cdots & \int \phi_1\phi_j A_{12} \\ \int \phi_2\phi_1 A_{12} & \cdots & \int \phi_2\phi_j A_{12} \\ \vdots & & \vdots \\ \int \phi_r\phi_1 A_{12} & \cdots & \int \phi_r\phi_j A_{12} \end{bmatrix} \cdots \begin{bmatrix} \int \phi_1\phi_1 A_{1k} & \cdots & \int \phi_1\phi_j A_{1k} \\ \int \phi_2\phi_1 A_{1k} & \cdots & \int \phi_2\phi_j A_{1k} \\ \vdots & & \vdots \\ \int \phi_r\phi_1 A_{1k} & \cdots & \int \phi_r\phi_j A_{1k} \end{bmatrix} \\ \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ \begin{bmatrix} \int \phi_1\phi_1 A_{i1} & \cdots & \int \phi_1\phi_j A_{i1} \\ \int \phi_2\phi_1 A_{i1} & \cdots & \int \phi_2\phi_j A_{i1} \\ \vdots & \vdots & \vdots \\ \int \phi_r\phi_1 A_{i1} & \cdots & \int \phi_r\phi_j A_{i1} \end{bmatrix} \begin{bmatrix} \int \phi_1\phi_1 A_{i2} & \cdots & \int \phi_1\phi_j A_{i2} \\ \int \phi_2\phi_1 A_{i2} & \cdots & \int \phi_2\phi_j A_{i2} \\ \vdots & \vdots & \vdots \\ \int \phi_r\phi_1 A_{i2} & \cdots & \int \phi_r\phi_j A_{i2} \end{bmatrix} \cdots \begin{bmatrix} \int \phi_1\phi_1 A_{ik} & \cdots & \int \phi_1\phi_j A_{ik} \\ \int \phi_2\phi_1 A_{ik} & \cdots & \int \phi_2\phi_j A_{ik} \\ \vdots & \vdots & \vdots \\ \int \phi_r\phi_1 A_{ik} & \cdots & \int \phi_r\phi_j A_{ik} \end{bmatrix} \end{bmatrix}$$

References

[1] B. Yamauchi, "Decentralized coordination for multirobot exploration," Robotics and Autonomous Systems, vol. 29, no. 2, pp. 111–118, 1999.

[2] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in AAAI/IAAI, pp. 746–752, 1998.

[3] V. N. Vapnik, Statistical Learning Theory. New York: John Wiley & Sons, 1998.

[4] S. Smale and F. Cucker, "On the mathematical foundations of learning," American Mathematical Society, vol. 39, no. 1, pp. 1–49, 2002.

[5] R. DeVore, G. Kerkyacharian, D. Picard, and V. Temlyakov, "Approximation methods for supervised learning," Foundations of Computational Mathematics, vol. 6, no. 1, pp. 3–58, 2006.

[6] V. Krishnamurthy, K. Topley, and G. Yin, "Consensus formation in a two-time-scale markovian system," Multiscale Modeling & Simulation, vol. 7, no. 4, pp. 1898–1927, 2009.

[7] S. S. Stankovic, M. S. Stankovic, and D. M. Stipanovic, "Decentralized parameter estimation by consensus based stochastic approximation," Automatic Control, IEEE Transactions on, vol. 56, no. 3, pp. 531–543, 2011.

[8] M. Porfiri and D. J. Stilwell, "Consensus seeking over random weighted directed graphs," Automatic Control, IEEE Transactions on, vol. 52, no. 9, pp. 1767–1773, 2007.

[9] M. Demetriou and S. Nestinger, "Adaptive consensus estimation of multi-agent systems," in Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on, pp. 354–359, IEEE, 2011.

[10] H. Paul, J. Fliege, and A. Dekorsy, "In-network-processing: Distributed consensus-based linear estimation," Communications Letters, IEEE, vol. 17, no. 1, pp. 59–62, 2013.

[11] G. C. Calafiore and F. Abrate, "Distributed linear estimation over sensor networks," International Journal of Control, vol. 82, no. 5, pp. 868–882, 2009.

[12] J. D. Kelly and D. Zyngier, "Hierarchical decomposition heuristic for scheduling: Coordinated reasoning for decentralized and distributed decision-making problems," Computers & Chemical Engineering, vol. 32, no. 11, pp. 2684–2705, 2008.

[13] G. Inalhan, D. M. Stipanovic, and C. J. Tomlin, "Decentralized optimization, with application to multiple aircraft coordination," in Decision and Control, 2002, Proceedings of the 41st IEEE Conference on, vol. 1, pp. 1147–1155, IEEE, 2002.

[14] T. Qu, G. Q. Huang, Y. Zhang, and Q. Dai, "A generic analytical target cascading optimization system for decentralized supply chain configuration over supply chain grid," International Journal of Production Economics, vol. 127, no. 2, pp. 262–277, 2010.

[15] R. L. Raffard, C. J. Tomlin, and S. P. Boyd, "Distributed optimization for cooperative agents: Application to formation flight," in Decision and Control, 2004. CDC. 43rd IEEE Conference on, vol. 3, pp. 2453–2459, IEEE, 2004.

[16] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli, "A new class of distributed optimization algorithms: Application to regression of distributed data," Optimization Methods and Software, vol. 27, no. 1, pp. 71–88, 2012.

[17] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams with complex constraints," Autonomous Robots, vol. 32, no. 4, pp. 385–403, 2012.

[18] F. J. Vazquez-Abad, C. G. Cassandras, and V. Julka, "Centralized and decentralized asynchronous optimization of stochastic discrete-event systems," Automatic Control, IEEE Transactions on, vol. 43, no. 5, pp. 631–655, 1998.

[19] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: convergence analysis and network scaling," Automatic Control, IEEE Transactions on, vol. 57, no. 3, pp. 592–606, 2012.

[20] R. Couillet, P. Bianchi, and J. Jakubowicz, "Distributed convex stochastic optimization under few constraints in large networks," in Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2011 4th IEEE International Workshop on, pp. 289–292, IEEE, 2011.

[21] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," Automatic Control, IEEE Transactions on, vol. 31, no. 9, pp. 803–812, 1986.

[22] D. P. Bertsekas, "Incremental proximal methods for large scale convex optimization," Mathematical programming, vol. 129, no. 2, pp. 163–195, 2011.

[23] T. Poggio and S. Smale, "The mathematics of learning: Dealing with data," Notices of the AMS, vol. 50, no. 5, pp. 537–544, 2003.

[24] S. Smale and D.-X. Zhou, "Learning theory estimates via integral operators and their approximations," Constructive approximation, vol. 26, no. 2, pp. 153–172, 2007.

[25] S. Smale and D.-X. Zhou, "Estimating the approximation error in learning theory," Analysis and Applications, vol. 1, no. 01, pp. 17–41, 2003.

[26] D.-X. Zhou, "Capacity of reproducing kernel spaces in learning theory," Information Theory, IEEE Transactions on, vol. 49, no. 7, pp. 1743–1752, 2003.

[27] S. Konyagin and V. Temlyakov, "The entropy in learning theory. error estimates," Constructive Approximation, vol. 25, no. 1, pp. 1–27, 2007.

[28] J. B. Predd, S. R. Kulkarni, and V. H. Poor, "Distributed learning in wireless sensor networks," Signal Processing Magazine, vol. 23, no. 4, pp. 56–69, 2006.

[29] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "Consistency in models for distributed learning under communication constraints," Information Theory, IEEE Transactions on, vol. 52, no. 1, pp. 52–63, 2006.

[30] J. B. Predd, S. R. Kulkarni, and V. H. Poor, "A collaborative training algorithm for distributed learning," Information Theory, IEEE Transactions on, vol. 55, no. 4, pp. 1856–1871, 2009.

[31] J. B. Predd, S. R. Kulkarni, and H. Poor, "A collaborative training algorithm for multi-sensor adaptive processing," in Computational Advances in Multi-Sensor Adaptive Processing, 2007. CAMPSAP 2007. 2nd IEEE International Workshop on, pp. 297–300, IEEE, 2007.

[32] F. Pérez-Cruz and S. R. Kulkarni, "Robust and low complexity distributed kernel least squares learning in sensor networks," Signal Processing Letters, IEEE, vol. 17, no. 4, pp. 355–358, 2010.

[33] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Collaborative training in sensor networks: A graphical model approach," in Machine Learning for Signal Processing, 2009. MLSP 2009. IEEE International Workshop on, pp. 1–6, IEEE, 2009.

[34] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Attribute-distributed learning: models, limits, and algorithms," Signal Processing, IEEE Transactions on, vol. 59, no. 1, pp. 386–398, 2011.

[35] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Cooperative training for attribute-distributed data: Trade-off between data transmission and performance," in Information Fusion, 2009. FUSION'09. 12th International Conference on, pp. 664–671, IEEE, 2009.

[36] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Attribute-distributed learning: The iterative covariance optimization algorithm and its applications," in American Control Conference (ACC), 2010, pp. 6783–6788, IEEE, 2010.

[37] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Agent selection for regression on attribute distributed data," in Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on, pp. 2242–2245, IEEE, 2010.

[38] F. Rosenblatt, "The Perceptron: A Probablistic Model For Information Storage and Orginization in the Brain," Psychological Review, vol. 65, no. 6, pp. 386–408, 1958.

[39] F. Harary, Graph Theory. Reading, MA: Addison-Wesley, 1969.

[40] M. Newman, Networks, An Introduction. New York: Oxford University Press, 2010.

[41] D. Zelazo, A. Rahmani, and M. Mesbahi, "Agreement via the Edge Laplacian," in Proceedings of th 46th IEEE Conference on Decision and Control, pp. 2309–2314, 2007.

[42] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," Automatic Control, IEEE Transactions on, vol. 49, no. 9, pp. 1520–1533, 2004.

[43] D. G. Luenberger, Optimization by Vector Space Methos. New York: John Wiley & Sons, Inc., 1969.

[44] R. Glowinski and P. Le Tallec, Augmented Lagrangian and operator-splitting methods in nonlinear mechanics, vol. 9. SIAM, 1989.

[45] C. Bacuta, "Schur complements on hilbert spaces and saddle point systems," Journal of Computational and Applied Mathematics, vol. 225, no. 2, pp. 581–593, 2009.

[46] P. Binev, A. Cohen, W. Dahmen, R. DeVore, V. Temlyakov, and P. Bartlett, "Universal algorithms for learning theory part i: Piecewise constant functions.," Journal of Machine Learning Research, vol. 6, no. 9, 2005.

[47] J. T. Oden, Applied Functional Analysis, A First Course for Students of Mechanics and Engineering Science. Englewoods Cliffs, NJ: Prentice Hall, 1979.

[48] B. Fristedt and L. Gray, A Modern Approach to Probability Theory. Boston: Birkhauser, 1997.

[49] D. B. West, Introduction to Graph Theory. Upper Saddle River, NJ: Prentice Hall, 2001.