

# ON PREDICTIVE LINEAR GAUSSIAN MODELS

by  
Matthew R. Rudary

A dissertation submitted in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
(Computer Science and Engineering)  
in The University of Michigan  
2009

Doctoral Committee:

Professor Satinder Singh Baveja, Chair  
Professor Alfred O. Hero III  
Professor Susan A. Murphy  
Professor Martha E. Pollack

© Matthew R. Rudary 2009  
All Rights Reserved

## ACKNOWLEDGEMENTS

It takes a village to write a dissertation, and I owe thanks to many people for their help and support.

My co-workers at Google have been very supportive as I finish my degree requirements, in particular by putting up with my somewhat distracted state at work in the weeks leading up to my defense.

My fellow Computer Science graduate students and other friends have helped shape my thinking, not to mention helping me find things to do on a Friday night. Special thanks go to Ashley Bangert, Meg Brennan, Dave and Nessa Butler, Jeff Cox, Dmitri Dolgov, Josh Estelle, Nick Gorski, Lisa Hsu, Patrick Jordan, Chris Kiekintveld, Kevin Lochner, Thede Loder, Leilah Lyons, Bob Marinier, Arnab Nandi, Andrew Nuxoll, Bart Peintner, Dan Reeves, Ali Saidi, Eugene Vorobeychik, and Julie Weber.

The other students in my lab (Ishan Chaudhuri, Mike James, Vishal Soni, Jon Sorg, Erik Talvitie, David Wingate, and Britton Wolfe) provided many interesting conversations during our time together and I am happy that they are my colleagues and friends. David Wingate in particular helped me during the early development of the Predictive Linear Gaussian model that is described in this dissertation.

I am grateful to all the faculty I interacted with at the University of Michigan, especially Edward Durfee, Alfred Hero, Susan Murphy, Martha Pollack, and Michael Wellman.

My parents, Bob and Nancy, my brother, Tom, and my extended family have supported me in every way possible during my academic career. My love and thanks go to them.

Finally, thank you to my advisor, Satinder Singh Baveja. His advice and guidance throughout my time at the university have been invaluable, starting from the time he took me aside after a session of his machine learning course and suggested that I become one of his students.

# TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS</b> . . . . .	<b>ii</b>
<b>LIST OF FIGURES</b> . . . . .	<b>vi</b>
<b>LIST OF TABLES</b> . . . . .	<b>vii</b>
<b>LIST OF APPENDICES</b> . . . . .	<b>viii</b>
<b>CHAPTER</b>	
<b>I. Introduction</b> . . . . .	<b>1</b>
<b>II. Related Models of Dynamical Systems</b> . . . . .	<b>4</b>
2.1 Predictive State Models . . . . .	4
2.1.1 The Linear PSR . . . . .	9
2.1.2 The Diversity Representation . . . . .	15
2.2 Traditional Continuous Models . . . . .	17
2.2.1 Autoregressive Models . . . . .	17
2.2.2 Linear Dynamical Systems . . . . .	19
2.2.3 Comparing LDS and AR Models . . . . .	23
<b>III. EPSRs: The First Class of Nonlinear PSRs</b> . . . . .	<b>25</b>
3.1 The E-test Prediction Matrix . . . . .	26
3.1.1 The Linear EPSR . . . . .	26
3.2 The Deterministic EPSR . . . . .	29
3.2.1 EPSRs May Be Exponentially Smaller Than Linear PSRs and POMDPs . . . . .	31
3.2.2 EPSRs and the Diversity Representation . . . . .	33
3.3 Remarks . . . . .	34
<b>IV. The Predictive Linear Gaussian Model</b> . . . . .	<b>36</b>
4.1 The Scalar PLG . . . . .	37
4.2 The Vector-Valued PLG . . . . .	43
4.2.1 Makeup of $Z_t$ . . . . .	44
4.2.2 Core Dynamics of the Uncontrolled PLG . . . . .	47
4.2.3 Modeling Actions in the PLG . . . . .	48
4.2.4 State Update . . . . .	49
4.3 The Variance-Adjusted PLG . . . . .	52
4.4 Optimal Control of the PLG . . . . .	53
4.4.1 Form of the Cost Function . . . . .	54

<b>V. Estimating the Parameters of the PLG</b> . . . . .	57
5.1 The Consistent Estimation Algorithm . . . . .	57
5.1.1 Linear Trends . . . . .	58
5.1.2 Initial State . . . . .	62
5.1.3 Noise Parameters . . . . .	63
5.2 Theoretical Results . . . . .	66
5.3 A Weighted Consistent Estimation Algorithm . . . . .	68
5.3.1 Linear Trend Parameters . . . . .	69
5.3.2 Noise Parameters . . . . .	70
5.4 Comparing CE to LDS Methods of Parameter Estimation . . . . .	70
5.4.1 CE vs. Subspace Methods . . . . .	70
5.4.2 CE vs. EM . . . . .	73
5.5 Experimental Results on Random Systems . . . . .	75
5.5.1 Scalar Uncontrolled Systems . . . . .	75
5.5.2 Scalar Controlled Systems . . . . .	77
5.5.3 Vector-Valued Uncontrolled Systems . . . . .	79
5.5.4 Vector-Valued Controlled Systems . . . . .	80
5.5.5 Comments on Experimental Results . . . . .	81
<b>VI. Applying the PLG to Real Data</b> . . . . .	87
6.1 Techniques for Modeling Applications . . . . .	87
6.1.1 Discounting the Estimate of $C_\eta$ . . . . .	88
6.1.2 Dealing with Trajectories of Different Lengths . . . . .	89
6.1.3 Modeling Features as Exogenous Inputs . . . . .	90
6.2 The NGSIM Traffic Data . . . . .	91
6.3 Learning Models of the Data . . . . .	93
6.4 Conclusions . . . . .	96
<b>VII. Conclusion</b> . . . . .	98
7.1 Contributions . . . . .	98
7.2 Future Work . . . . .	99
<b>APPENDICES</b> . . . . .	<b>100</b>
<b>REFERENCES</b> . . . . .	<b>138</b>

## LIST OF FIGURES

**Figure**

2.1	The system dynamics vector. . . . .	10
2.2	The system dynamics matrix. . . . .	12
2.3	The core tests, $Q$ , are selected so that $\mathcal{D}(Q)$ spans the column space of $\mathcal{D}$ . . . . .	13
2.4	The graphical model representation of the autoregressive model. . . . .	18
2.5	The graphical model representation of the linear dynamical system. . . . .	21
3.1	The $k$ -bit rotate register. . . . .	31
3.2	The $k$ -bit hit register. . . . .	33
4.1	A dynamical system with real-valued, scalar observations. . . . .	37
4.2	The skyline constraint on $Z_t$ . . . . .	45
5.1	Comparison of EM and CE on scalar uncontrolled systems. . . . .	76
5.2	A zoomed-in look at the results of CE on scalar, uncontrolled systems of dimension 8. . . . .	77
5.3	Comparison of EM and CE on scalar controlled systems. . . . .	78
5.4	Comparison of EM and CE on vector-valued uncontrolled systems. . . . .	83
5.5	A zoomed-in look at the results of CE and EM on vector-valued, uncontrolled systems of dimension 8. . . . .	84
5.6	Comparison of EM and CE on vector-valued controlled systems. . . . .	85
5.7	A histogram of the number of cycles required by the EM algorithm with $n = 4$ . . . . .	86
6.1	An aerial photograph of the section of Interstate 80 covered by the data set. . . . .	92
6.2	The distribution of trajectory lengths in the training and test sets are shown here with a bin width of 50 time steps (i.e. 5 seconds). . . . .	93
6.3	There are two types of radar-style features. . . . .	94

## LIST OF TABLES

### Table

2.1	A taxonomy of models of dynamical systems. . . . .	17
4.1	Scalar PLG notation. . . . .	39
4.2	PLG notation. . . . .	44
5.1	Time taken by the CE algorithm and each iteration of EM. . . . .	82
6.1	Performance of CE and EM on the test trajectories of the I80 data set using various feature sets. . . . .	95
6.2	Statistical significance of the differences in performance between models learned using different feature sets. . . . .	96

## LIST OF APPENDICES

### Appendix

A.	Proofs and Derivations . . . . .	101
A.1	Derivation of the PLG Update Equations . . . . .	101
A.1.1	Derivation of the Scalar PLG Update Equations . . . . .	101
A.1.2	Derivation of the Full PLG Update Equations . . . . .	104
A.2	Proof of Theorem 4.2 . . . . .	106
A.2.1	Populating $Z_t$ . . . . .	107
A.2.2	Computing the State Mapping . . . . .	109
A.2.3	Computing $J$ and $\Sigma_{\text{adj}}$ . . . . .	110
A.2.4	Computing the Remaining PLG Parameters . . . . .	111
A.2.5	Preserving the State Mapping Through Updates . . . . .	114
A.3	Proof of Lemma 4.3 . . . . .	115
A.4	Proof of Theorem 4.4 . . . . .	117
A.5	Proof of Theorem 5.1 . . . . .	119
B.	Reference Material . . . . .	123
B.1	Large Sample Behavior . . . . .	123
B.2	Changing the Conditions of Expectations and Variances . . . . .	125
C.	The Gradient of the Log-Likelihood Function . . . . .	128

## CHAPTER I

### Introduction

Models, implicit or explicit, are used in almost every field of human endeavor. For example, in baseball, the batter uses an implicit model of the baseball's physics to predict where the ball will cross the plate, based on the pitcher's release point and arm speed, the spin of the ball, and similar factors. Chip designers predict the speed of a new processor before the masks are even built, using their models of the properties of aluminum, copper, silicon, and other materials. Mutual fund managers use models of financial instruments and markets to predict future returns and the risk involved.

People use models to make predictions about future outcomes in systems that change over time, generally to help guide their behavior. In these examples, the baseball player uses the model's prediction to decide whether to swing and where to place the bat; the chip designers use the model to simulate their chip and guide design decisions (e.g., where to place various components so as to minimize wire lengths on the critical path); mutual fund managers use their predictions of risk to balance their funds' portfolios. Artificial agents can *also* use models to make predictions and guide their decisions; these models will be the focus of this dissertation.

The predictions that a model makes are based on its *state*—that is, its estimate

of the current situation of the world. At each time step<sup>1</sup>, the model updates its state based on the agent’s interaction with the system (i.e. the action taken by the agent and the information provided by the sensors). In most worlds of interest to agent designers, the sensor information available to the agent does not convey all information about the current situation in the world; systems with this property are called *partially observable*. Under partial observability, maintaining the system’s state is necessary to make good predictions about the future (in fully observable environments, the most recent observation can serve as state).

One important aspect to consider when selecting a model is the availability and performance of algorithms that learn the parameters of that model from data. An agent should be able to begin making good decisions after spending time in an unfamiliar environment. Thus, an agent should be able to learn a model based on its experience in the world.

Many model classes are used when designing artificial agents, but few models are suited to deal with partial observability *and* to be learned from experience. A new paradigm in models, *predictive state*, was designed with both these desiderata in mind. To date, most work on predictive state models has been focused primarily on *linear* models of systems with *discrete* observations. Here, I explore two different improvements to these models. First, I propose a *nonlinear* predictive state model of systems with discrete observations; this nonlinear model may have much smaller dimension than a linear model of the same system. Second, I propose a linear predictive state model of systems with *continuous, vector-valued* observations. This will allow predictive state models to be applied to an important domain of dynamical systems that is not served by the previous work.

---

<sup>1</sup>This dissertation treats only models in *discrete time*.

In Chapter II, I present some existing models of dynamical systems, including previous predictive state models as well as traditional models. Chapter III is devoted to the EPSR, a nonlinear predictive state model for deterministic systems with discrete observations. Chapter IV introduces the predictive linear-Gaussian model (PLG), a predictive state model for dynamical systems with continuous, vector-valued observations. Chapter V lays out several parameter estimation algorithms for the PLG and presents experimental results for one of those algorithms in artificial systems. Finally, Chapter VI describes a traffic prediction problem and presents experimental results in modeling vehicles using data taken from a California freeway.

## CHAPTER II

### Related Models of Dynamical Systems

There are many, many models of dynamical systems in existence. Here I present some of those models that are of interest because they have similar dynamics or similar applications to the models I will present in this thesis. The models I discuss in Chapter III and beyond will all be predictive state models; therefore, I will introduce the previous work in this field in Section 2.1.

However, these predictive state models almost exclusively model systems with discrete observations. Since the predictive linear-Gaussian model, which I discuss from Chapter IV on, deals with continuous observations, I also present two important (but non-predictive-state) models in that domain in Section 2.2.

#### 2.1 Predictive State Models

An important problem in modeling dynamical systems is that of *partial observability*; a partially observable system is one in which the most recent observation does not include all of the information that is relevant to making predictions about the system—that is, that the observation is not *state*. As an example of a partially observable system, consider riding in a vehicle in which you may observe the odometer (i.e. the distance traveled) once per second but nothing else. If the vehicle does not move at a constant velocity, just knowing the most recent odometer reading will

not allow you to predict the next reading as accurately as would be possible if you had access to every odometer reading so far. In order to model partially observable systems, more information than the most recent observation must be maintained; this information is called the state of the model.

There are several types of models that deal with partial observability. One of the simplest types assumes that, if the last observation by itself does not constitute state, the last  $k$  observations (for some finite  $k$ ) *does*. In the odometer example, the modeler might assume that the vehicle’s velocity will not change quickly. The difference between the last two observations can then be added to the last observation to estimate the next observation. Alternatively, it may be assumed that the vehicle’s *acceleration* will change slowly. In this case, the last *three* observations would be necessary. Models of this type are sometimes called “history window” models, and include autoregressive models (see Section 2.2.1) and  $k$ -Markov models.

There are two main downsides to this approach. First, as the number of observations to remember,  $k$ , grows, the size of the state space grows *exponentially*. This may not be apparent given the simple dynamics of the odometer example, but consider observing four LEDs. There are  $2^4 = 16$  different patterns that these LEDs can take on. If the next pattern depends only on the previous pattern, there are 16 states to consider. However, if the next pattern depends on the previous *two* patterns, there are 256 states to consider; when  $k = 3$ , there are *4096* states, and so on. The second drawback is that history window models are not as expressive as the other models I will address. Suppose that in the LED example, the next pattern depended only on the previous pattern and on the *first* pattern. No matter how large  $k$  is made, the model will eventually “forget” what the first pattern was and become inaccurate.

Latent state models address this second drawback, and to a lesser extent, the first

as well. In a latent state model, random variables are created that represent features of the world. Probability distributions of these random variables are then maintained; these distributions are the state of the model. In many cases, the random variables, if observed, would be the state of a fully observable model; since they cannot be observed directly, they are called *latent state*.

Returning to the odometer example, suppose again that the velocity changes only slowly. If the velocity were observed along with the odometer, the system would be fully observable. Since the speed is not observed, a random variable may be introduced that represents the current velocity. This variable might have a normal distribution whose variance depends on how slowly the vehicle's speed changes and how accurate the odometer readings are. Likewise, a variable representing the current location would also be introduced; if the odometer is perfectly accurate, this would simply be the most recent odometer reading. Assuming appropriate model parameters, this model will be at least as accurate as the history window approach, and will be *more* accurate if the odometer reading is noisy; I will return to this point in Section 2.2.2.

There are many positive aspects to this approach, particularly when the identities of all the relevant hidden features are known to the model designer, along with the way these features change over time and affect the observations emitted by the system. These models are relatively easy to specify, and they can often take advantage of well-established probabilistic machinery (for example, dynamic Bayes nets).

The latent state approach also has its drawbacks. When the dynamics of the system are not known a priori, learning the parameters of these models is subject to difficulties. Moreover, these models are not *verifiable* by an automatic agent; the probability distributions over latent variables are in a sense meaningless numbers to

a computer program, and detecting an inaccurate model may be quite difficult.

An alternative approach to the problem of partial observability that has received recent focus in the reinforcement learning community is to use *predictive state*. In a predictive state model, the state is not probability distributions over latent variables, it is probabilistic statements about future observable events.

In the odometer example, instead of remembering the last two observations or keeping probability distributions over location and velocity variables, a predictive state model might keep a probability distribution over the *next* two observations. In Chapter IV, I will describe the Predictive Linear-Gaussian model (PLG). A PLG *would* model this system by keeping a probability distribution over the next two observations. While it has a superficial resemblance to the history window approach (it deals directly in observations), it also models the system as well as the latent state approach, even when the odometer reading is noisy.

One reason predictive state models have received attention is that there is some hope that they may provide better generalization than traditional models; this is the *predictive representations hypothesis* (Rafols, Ring, Sutton, & Tanner, 2005). A model generalizes well in the sense meant here when it is able to make more accurate predictions in situations that it has not yet been exposed to. Better generalization also leads to faster convergence of the model during training, as learning about a given situation also improves the model's knowledge about similar situations.

The intuition behind the hypothesis is this: Suppose a model makes a set of predictions about the future. If this set is chosen well, the fact that the predictions in two situations resemble one another will mean that the optimal action, expected future rewards, expected outcomes of various courses of action, etc., will *also* resemble each other. For example, suppose that a robot has been designed to move around an

office environment, and that it has been trained using data taken on the second floor of a particular office building. If this same robot performs nearly as well (in terms of making predictions or accumulating reward) when placed on the third floor, the robot has generalized well. We would expect that this generalization is possible, because office floor layouts are frequently similar on different floors of the same building, and some aspects of an office are nearly universal (for example, a door is a door—turning the knob of an unlocked door will open it even in otherwise quite different office settings).

Rafols et al. (2005) provide some experimental results that suggest that the predictive representations hypothesis may hold in practice. Their evidence comes in part from experiments in a grid world that contains multiple rooms with similar layouts arranged along a hallway. Though I do not explore this hypothesis further in this thesis, it provides one motivation for developing and extending predictive state models, as well as an interesting direction for future work.

An advantage of PSRs that I *do* explore in this thesis is their potential for improved model learning. Because traditional models make use of hidden state, they are often learned using Expectation Maximization (EM), which is able to deal with latent variables but is subject to problems with local optima. Predictive state models, on the other hand, are made up of probabilistic predictions about the future conditioned on past interactions with the system. These predictions are *grounded* in the data; given enough experience with a system, these probabilities may be estimated. There has been a flurry of work in algorithms that learn PSRs by doing just that (e.g., James & Singh, 2004; Wiewiora, 2005; Bowling, McCracken, James, Neufeld, & Wilkinson, 2006). In one paper, Wolfe, James, and Singh (2005) compared learning one type of predictive state model to using EM to estimate the parameters of partially

observable Markov decision processes (POMDPs) by running experiments on several small problems, and found that learning predictive state models usually outperformed EM.

One of the foundational (and most widely studied) predictive state models is the linear predictive state representation (PSR), presented here.

### 2.1.1 The Linear PSR

The linear PSR was first introduced by Littman, Sutton, and Singh (2002); this work focused on the relationship of PSRs to POMDPs. A later report derived the linear PSR from the perspective of a system dynamics matrix (Singh, James, & Rudary, 2004). Their approach clarified the relationship between linear PSRs and other traditional models as well as allowing the PSR to be understood independently of the POMDP. Thus, to present the linear PSR, I first develop the theory of the system dynamics matrix.

**The System Dynamics Matrix** A dynamical system can be viewed abstractly as a probability distribution over trajectories. In an uncontrolled dynamical system, each trajectory  $t = o^1 o^2 \cdots o^k$ , where  $o^1, \dots, o^k \in \mathcal{O}$  are observations, is assigned a probability

$$(2.1) \quad p(t) = \Pr(o_1 = o^1, \dots, o_k = o^k),$$

where  $o_i$  is the actual observation generated by the system at time step  $i$ . A controlled dynamical system, on the other hand, takes an input (action) from some set  $\mathcal{A}$  at each time step and generates an observation from  $\mathcal{O}$ . Thus, in a controlled system, a trajectory is made up of action-observation pairs for each time step. For each

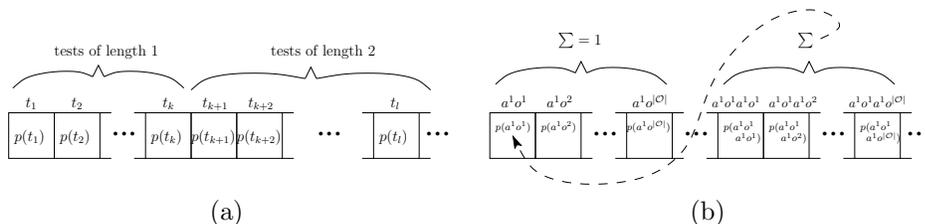


Figure 2.1: The system dynamics vector. (a) Each entry of the system dynamics vector corresponds to the unconditional prediction of a test. In this diagram,  $k = |\mathcal{A}||\mathcal{O}|$  is the number of one-step tests, and  $l = |\mathcal{A}|^2|\mathcal{O}|^2 + k$  is the number of one- and two-step tests. (b) Certain structure in the vector is implied by the properties of dynamical systems, as explained in the text.

trajectory  $t = a^1 o^1 \dots a^k o^k$ , an *action-conditional* probability is assigned:

$$(2.2) \quad p(t) = \Pr(o_1 = o^1, \dots, o_k = o^k | a_1 = a^1, \dots, a_k = a^k).$$

In controlled systems, it is convenient to think of a trajectory as a *test*,<sup>1</sup> or experiment, that can be performed on the system. The test  $t$  is executed by performing the specified actions; it succeeds if the specified observations are generated by the system. Then  $p(t)$  is the probability of the test succeeding given that it is executed, and can be thought of as a *prediction* for the test. To maintain a consistent vocabulary (and in keeping with the established practice in the literature), trajectories in both controlled and uncontrolled systems will be referred to as tests.

Given an ordering over all possible tests  $t_1, t_2, \dots$ , a dynamical system's probabilities of these tests define an infinite vector  $d$ , called the *system dynamics vector* because it specifies the dynamics of the system; the  $i$ th element of  $d$  is  $p(t_i)$ , the prediction of the  $i$ th test. A convenient ordering of the tests is to arrange them in order of increasing length, with ties broken by lexicographic ordering. This vector is illustrated in Figure 2.1(a).

<sup>1</sup>In particular, this is a *sequence test*, or *s-test*, because it consists of a sequence of actions and observations. For the remainder of this section, I shall use test to mean s-test, but the next section will introduce a new kind of test.

In order for  $d$  to represent a *consistent* probability distribution over the tests, it must satisfy the following constraints:

1. Each element of  $d$  must be a probability:

$$\forall i \ 0 \leq d_i \leq 1.$$

2. The predictions of the tests with a given action sequence must sum to one: If  $T(\bar{a})$  is the set of tests whose action sequences are  $\bar{a}$ , then

$$\forall k \ \forall \bar{a} \in \mathcal{A}^k \quad \sum_{t \in T(\bar{a})} p(t) = 1.$$

Note that this applies to uncontrolled systems as well; an uncontrolled system may be considered to be a controlled system in which  $\mathcal{A}$  contains a single element.

3. If a test  $t'$  has another test  $t$  as a prefix,  $p(t')$  must not exceed  $p(t)$ . In fact, the sum of the predictions of all tests of the same length that share a prefix  $t$  must equal the prediction of  $t$ :

$$\forall t \ \forall a \in \mathcal{A} \quad p(t) = \sum_{o \in \mathcal{O}} p(tao).$$

These properties, illustrated in Figure 2.1(b), guarantee that  $d$  contains a lot of structure. This structure can be exploited by creating a *system dynamics matrix*,  $\mathcal{D}$ , whose columns correspond to tests and whose rows correspond to histories. Each element of  $\mathcal{D}$  is a conditional prediction  $p(t|h)$ , i.e., the probability of  $t$  succeeding given that the history  $h$  has already been observed.

More formally, for a test  $t = a^1 o^1 \cdots a^k o^k$  and a history  $h = a_1 o_1 \cdots a_j o_j$ , the history-conditional prediction  $p(t|h)$  is given by

$$(2.3) \quad p(t|h) = \Pr(o_{j+1} = o^1, \dots, o_{j+k} = o^k | h, a_{j+1} = a^1, \dots, a_{j+k} = a^k).$$

	$t_1$	$t_2$	$\cdots$	$t_j$	$\cdots$
$h_1 = \phi$	$p(t_1)$	$p(t_2)$	$\cdots$	$p(t_j)$	
$h_2$	$p(t_1 h_2)$	$p(t_2 h_2)$	$\cdots$	$p(t_j h_2)$	
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	
$h_i$	$p(t_1 h_i)$	$p(t_2 h_i)$	$\cdots$	$p(t_j h_i)$	
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$	
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$	

Figure 2.2: The system dynamics matrix.

Suppose an ordering over the histories is defined so that  $h_1 = \phi$ , where  $\phi$  is the zero-length (i.e. initial) history and  $h_i = t_{i-1}$  for  $i = 2, 3, \dots$  (though the histories may technically be ordered arbitrarily, it is convenient to order them in the same way as the tests). Then let the  $ij$ th element of  $\mathcal{D}$  be

$$(2.4) \quad \mathcal{D}_{ij} \triangleq p(t_j|h_i) = \frac{p(h_i t_j)}{p(h_i)}.$$

This matrix is illustrated in Figure 2.2.

There are several things to notice about  $\mathcal{D}$ . First, the first row of  $\mathcal{D}$  is the system dynamics vector  $d$ —that is,  $p(t|\phi) = p(t)$ . Second, each row of  $\mathcal{D}$  has the 3 properties listed above that  $d$  has. Third, even though  $\mathcal{D}$  has infinitely many rows *and* columns, it is uniquely determined by  $d$ ; that is, it reorganizes the information in  $d$  but does not introduce any *new* information. That is because both the numerator and the denominator in the right-hand side of (2.4) are elements of  $d$ . Finally,  $d$  and  $\mathcal{D}$  *specify* the system, but they are not useful *models* of the system. They both require an infinite amount of memory to store, and so are unwieldy.

By contrast, a *model* of a system can be thought of a set of rules that could be used to reconstruct (parts of) these data structures. The state of the model “saves” the place in the structure—essentially, the model can be used to generate the row of  $\mathcal{D}$  corresponding to the history observed up to that point.

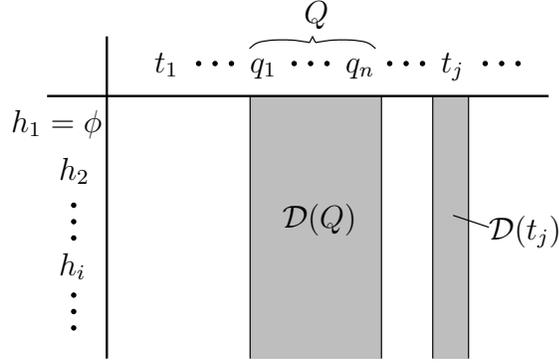


Figure 2.3: The core tests,  $Q$ , are selected so that  $\mathcal{D}(Q)$  spans the column space of  $\mathcal{D}$ .

**The Linear PSR Model** The linear PSR model is closely related to the system dynamics matrix. While all models can be used to generate elements of  $\mathcal{D}$ , the state of a linear PSR actually *consists* of elements of  $\mathcal{D}$ .

Linear PSRs exploit certain structure of the system dynamics matrix—in particular, its rank. Even though  $\mathcal{D}$  is an infinite matrix, it may still have a finite rank. If  $\text{rk}(\mathcal{D}) = n$ , there must be a set of  $n$  linearly independent columns (in fact, there will be infinitely many such sets). Select a single set of  $n$  linearly independent columns of  $\mathcal{D}$  and let the tests corresponding to these columns be  $Q = \{q_1 \ q_2 \ \cdots \ q_n\}$ . The tests  $q_1, q_2, \dots, q_n$  are the *core tests* of a linear PSR. Denote by  $\mathcal{D}(Q)$  the submatrix of  $\mathcal{D}$  consisting only of the columns corresponding to the core tests (see Figure 2.3).

The state representation of a linear PSR is a vector of history-conditional predictions for the core tests. That is, for any history  $h$ , the state of the PSR is given by the vector

$$(2.5) \quad P(Q|h) = \begin{pmatrix} p(q_1|h) \\ p(q_2|h) \\ \vdots \\ p(q_n|h) \end{pmatrix}.$$

The initial state of the system is  $p(Q|\phi)$ , the entries of the first row of  $\mathcal{D}(Q)$ .

Because  $\mathcal{D}(Q)$  spans  $\mathcal{D}$ , each column of  $\mathcal{D}$  is a linear combination of the columns of  $\mathcal{D}(Q)$ . In particular, for every test  $t$ , there is an  $n$ -vector  $m_t$  such that  $\mathcal{D}(t)$ , the column of  $\mathcal{D}$  corresponding to  $t$ , is given by  $\mathcal{D}(t) = \mathcal{D}(Q)m_t$ . This implies that

$$(2.6) \quad p(t|h) = p(Q|h)^\top m_t$$

for every history  $h$ . The fact that the prediction of every test is a linear function of the predictions of the core test gives the *linear* PSR its name.

In particular, this property allows the state to be updated whenever an action  $a$  is taken and an observation  $o$  is observed:

$$(2.7) \quad p(q_i|hao) = \frac{p(aoq_i|h)}{p(ao|h)} = \frac{p(Q|h)^\top m_{aoq_i}}{p(Q|h)^\top m_{ao}}.$$

The entire update can be combined into a single matrix equation by defining  $M_{ao}$  to be the matrix whose  $j$ th column is  $m_{aoq_j}$ :

$$(2.8) \quad p(Q|hao)^\top = \frac{p(Q|h)^\top M_{ao}}{p(Q|h)^\top m_{ao}}.$$

It can be shown that the weight vector for any test can be computed by a function of the matrices  $M_{ao}$  and vectors  $m_{ao}$ ; if  $t = a^1 o^1 \dots a^k o^k$ , then  $m_t$  is calculated by

$$(2.9) \quad m_t = M_{a^1 o^1} M_{a^2 o^2} \dots M_{a^{k-1} o^{k-1}} m_{a^k o^k}.$$

Thus, the matrices  $M_{ao}$  and vectors  $m_{ao}$ , and initial state  $p(Q|\phi)$ , suffice as model parameters for a linear PSR.

Using these parameters, the PSR can generate any element of the system dynamics matrix  $\mathcal{D}$  by the following operation:

$$(2.10) \quad \mathcal{D}_{ij} = p(t_j|h_i) = \frac{p(h_i t_j)}{p(h_i)} = \frac{p(Q|\phi)^\top m_{h_i t_j}}{p(Q|\phi)^\top m_{h_i}}.$$

**Theorem 2.1.** *A linear PSR with  $n$  core tests can represent a dynamical system if and only if its system dynamics matrix has rank no more than  $n$ .*

The fact that an  $n$ -dimensional linear PSR can model a system with rank  $n$  follows from the derivation of PSRs; the fact that it cannot model a system with rank more than  $n$  follows from the fact that  $\mathcal{D}(t) = \mathcal{D}(Q)m_t$  for every  $t$ ; since  $\mathcal{D}(Q)$  has only  $n$  columns, this limits the rank of  $\mathcal{D}$  to  $n$ .

Theorem 2 of (Singh et al., 2004) states that a POMDP with  $n$  nominal states cannot model a dynamical system whose system dynamics matrix has rank greater than  $n$ . A corollary to this and to Theorem 2.1 is that  $n$ -dimensional POMDPs are subsumed by  $n$ -dimensional linear PSRs:

**Corollary 2.2.** *For any dynamical system that can be modeled by a finite POMDP with  $n$  hidden states, there exists a linear PSR with no more than  $n$  core tests.*

Despite their representational power and theoretical elegance, linear PSRs have some drawbacks. Most of the parameter learning algorithms for linear PSRs require data sets whose size is exponential in the system dimension  $n$ . And despite their subsumption of POMDPs, it is rare for a linear PSR to have a smaller dimension than a given POMDP. Thus linear PSRs provide little hope of solving the curse of dimensionality. In Chapter III, I will describe a new type of PSR that takes a step in this direction. This new PSR is related to the diversity representation of Rivest and Schapire (1994), which is described in the next section.

### 2.1.2 The Diversity Representation

Rivest and Schapire (1994) describe a model of deterministic dynamical systems called the diversity representation. Where the linear PSR is built up on s-tests of the form  $t = a^1 o^1 a^2 o^2 \dots a^k o^k$ , the diversity representation makes use of the *e-test*, or *end test*. An e-test  $e$  is made up of a sequence of actions capped off with a single

terminal observation:  $e = a^1 a^2 \dots a^k o^k$ . The prediction of an e-test is

$$(2.11) \quad p(e|h) = \Pr(o_{j+k} = o^k | \text{length}(h) = j, a_{j+1} = a^1, a_{j+2} = a^2, \dots, a_{j+k} = o^k).$$

Define  $v(e)$ , the *outcome vector* of the e-test  $e$ .  $v(e)$  is an infinite vector whose  $i$ th element is  $v_i(e) = p(e|h_i)$ .

The diversity representation classifies e-tests according to their equivalence classes; two tests are in the same equivalence class iff their outcome vectors are identical. Thus, at any time, all e-tests in an equivalence class have the same prediction. The number of equivalence classes in a system is the diversity of that system, denoted  $D$ . The state representation of the mode, then, is the set of predictions for the equivalence classes.

At the heart of the diversity representation is its update graph. This directed graph contains nodes labeled with each of the equivalence classes. All edges are labeled with an action, and each node has one outbound edge and one inbound edge for each action. Whenever action  $a$  is taken, predictions flow along the edges labeled with  $a$ . For example, say that  $e_1$  is in equivalence class  $C_1$  and  $e_2$  is in  $C_2$ , and there is an edge labeled  $a$  from  $C_1$ 's node to  $C_2$ 's node. Then  $p(e_2|ha) = p(e_1|h)$ .

Suppose a deterministic POMDP with  $S$  states is given. Rivest and Schapire showed that, if there are no redundant states in the automaton, the diversity is bounded by  $\log_2 S \leq D \leq 2^S$ , and that these bounds are tight. That is, the diversity representation may be an exponential compression of the POMDP for a system, but it may also be an exponential expansion.

The EPSR that I introduce in Chapter III also makes use of e-tests. But instead of using equivalent outcome vectors and an update graph to model the system, the EPSR uses linear independence of outcome vectors and state update equations similar to those of the linear PSR. This allows the EPSR to avoid the risk of exponential

Table 2.1: A taxonomy of models of dynamical systems.

	Discrete Observations	Real Observations
History Window Models	$k$ -Markov Models	Autoregressive Models
Latent State Models	Hidden Markov Models / Partially Observable MDPs	Linear Dynamical Systems
Predictive State Models	Linear PSRs	PLG Models

expansion while reaping the benefits of exponential compression. This is discussed in more detail in the chapter on EPSRs.

## 2.2 Traditional Continuous Models

There are many extant models of discrete-time dynamical systems with real-valued actions and observations. However, two models in particular are of note. The first is the autoregressive (AR) model; AR models appear to have similar dynamics to the predictive linear-Gaussian model (see Chapter IV), though they are less expressive. Linear dynamical system models, on the other hand, appear to have quite different dynamics than the PLG, but have similar representative power.

Analogies can be drawn between these models and models of systems with discrete observations (see Table 2.1). Autoregressive models are similar to  $k$ -Markov models in that the dynamics depend on a “history window”; that is, the distribution of the next observations depends on a set number of previous actions and observations. Likewise, linear dynamical systems, like hidden Markov models and partially observable Markov decision processes, are based upon an unobserved state process that drives the observed process.

### 2.2.1 Autoregressive Models

In an  $n$ -dimensional autoregressive (AR) model, the distribution of the next observation depends on the previous  $n$  observations and actions; the controlled version is called the autoregressive model with exogenous inputs (ARX model). These are

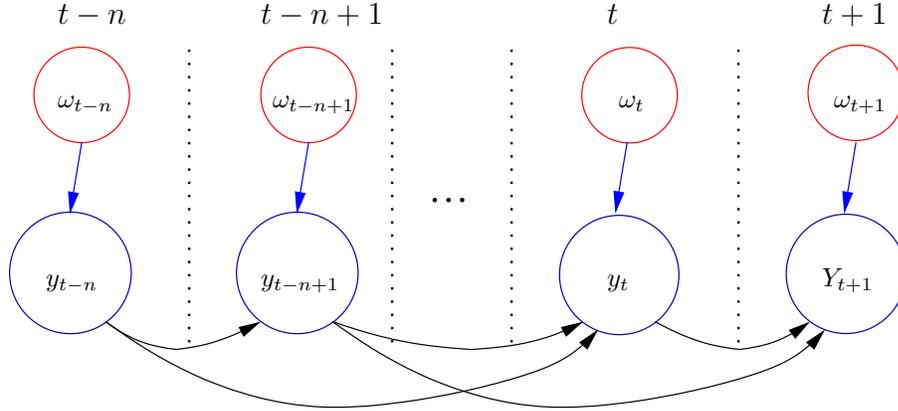


Figure 2.4: The graphical model representation of the autoregressive model. The  $t$ th observation has just been generated; observations that have already been generated are set in lowercase, while those that have not yet been observed are in uppercase.

generally formulated as time-series models, i.e. they model systems with scalar observations, though vector-valued versions exist. This presentation is limited to the time-series version. For a more detailed treatment of AR models, please refer to Pandit and Wu (1983).

The core dynamics of the AR model are given by

$$(2.12) \quad Y_{t+1} = \sum_{i=1}^n \alpha_i y_{t+1-i} + \omega_{t+1},$$

where  $\alpha_1, \dots, \alpha_n$  describe the linear trend in the observations and  $\omega_{t+1}$  is a random variable modeling the randomness in the observations (i.e. a noise variable). It is distributed according to

$$(2.13) \quad \omega_{t+1} \sim \mathcal{N}(0, \sigma_{AR}^2).$$

See Figure 2.4, which illustrates this situation: the observation at time  $t + 1$  is a function of the  $n$  observations leading up to it and a noise term.

These dynamics admit a simple sufficient statistic of history: the last  $n$  observations. By maintaining a record of just the last  $n$  observations and discarding the rest of history, correct history-conditional probability distributions of future observations

can be computed.

The fact that only the last  $n$  observations have an effect on the next observation means that the AR model is somewhat impoverished in its representational power, though its simplicity means that its parameters can readily be estimated from small amounts of data.

### 2.2.2 Linear Dynamical Systems

The linear dynamical system model (LDS) is widely used in many engineering and scientific fields. Its applications include airplane-wing design (Norlander, Nilsson, Ring, & Johansson, 2000), control of ore concentrators in the mining industry (Norlander, 2000), glass tube manufacturing (Overschee & De Moor, 1996), and many others. They are formulated as continuous-time systems governed by differential equations as well as discrete-time systems governed by difference equations. Only the latter is presented here; for a more detailed treatment of the LDS, please see Bar-Shalom, Li, and Kirubarajan (2001).

The LDS model is based on a pair of stochastic processes. The first is the latent (unobservable)  $X_t$  process, which evolves through a linear function that is perturbed by the actions and by i.i.d. Gaussian noise. That is,

$$(2.14) \quad X_{t+1} = AX_t + Bu_t + \omega_{t+1},$$

where  $A$  is the linear trend in the  $X_t$  process,  $B$  is the linear effect of the action, and  $\omega_{t+1}$  is the noise term. In an  $n$ -dimensional LDS,  $X_t$  is an  $n$ -vector. The action  $u_t$  is a vector of dimension  $l$ .

The second process is the observations process  $Y_t$ , which is a linear function of the first process, again with i.i.d. Gaussian noise added:

$$(2.15) \quad Y_t = HX_t + \nu_t.$$

The observation  $Y_t$  is a vector of dimension  $m$ . Each noise term is independent of every other:

$$(2.16) \quad \text{Cov}[\omega_t, \omega_s] = \delta_{t,s}Q, \quad \text{Cov}[\nu_t, \nu_s] = \delta_{t,s}R,$$

where  $\delta_{t,s}$ , the Kronecker delta, is 1 if  $t = s$  and 0 otherwise. Here,  $Q$  and  $R$  are the variance of the process noise and measurement noise, respectively. Both noise processes have mean zero:

$$(2.17) \quad \text{E}[\omega_t] = \mathbf{0}, \quad \text{E}[\nu_t] = \mathbf{0}.$$

Additionally, the  $X_t$  process is initialized with a Gaussian distribution:

$$(2.18) \quad X_1 \sim \mathcal{N}(x_1^-, P_1^-).$$

The dynamics of the LDS are completely described by (2.14)–(2.18) (see Figure 2.5), but these do not explain how the state of the LDS model is maintained.

The  $X_t$  process is often referred to as the “state process”; this is because  $X_t$  often represents what is assumed to be the underlying truth of a system. For example,  $X_t$  may be the true location and velocity of an object, the sizes of several populations, or the flow rates at various points in a series of pipe. However, the value of  $X_t$  cannot serve as the state of the model at time  $t$  because it is *unobserved*. Instead, the state of the model at time  $t$  is the probability distribution of  $X_t$  given the history of the system. This distribution is maintained by the Kalman filter (Kalman, 1960).

Because  $X_t$  is Gaussian, its distribution is described by a mean and a variance; in the context of the Kalman filter, these are sometimes referred to as the state estimate and uncertainty, respectively. The Kalman filter maintains the state estimate  $x_t^- =$

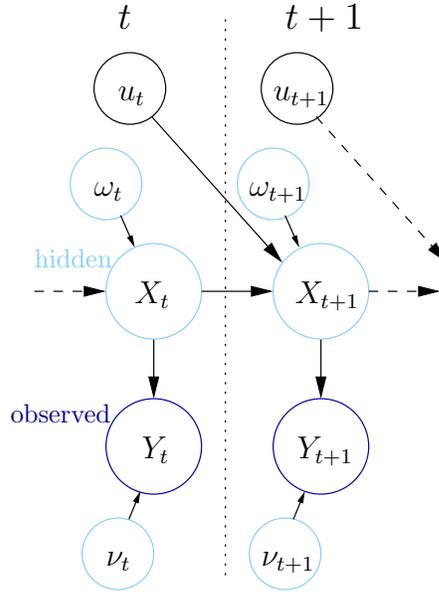


Figure 2.5: The graphical model representation of the linear dynamical system. The observations  $Y_t$  are observed; the latent state process  $X_t$  and the noise terms  $\omega_t$  and  $\nu_t$  are not. The process is controlled by the actions  $u_t$ .

$E[X_t|h_{t-1}]$  and its covariance  $P_t^- = \text{Var}[X_t|h_{t-1}]$  using the state update equations

$$(2.19) \quad K_t = P_t^- H^\top (H P_t^- H^\top + R)^{-1},$$

$$(2.20) \quad x_t^+ = x_t^- + K_t (y_t - H x_t^-),$$

$$(2.21) \quad P_t^+ = P_t^- - K_t H P_t^-,$$

$$(2.22) \quad x_{t+1}^- = A x_t^+ + B u_t,$$

$$(2.23) \quad P_{t+1}^- = A P_t^+ A^\top + Q.$$

$K_t$  is known as the *Kalman gain* at time  $t$ , and  $x_t^+$  and  $P_t^+$  have the semantics  $X_t|h_{t-1}, y_t \sim \mathcal{N}(x_t^+, P_t^+)$ ; that is, they are the versions of the state variables updated to take into account the observation at time  $t$ , but that have not yet taken the action or the passage of time into account.

**Linear-Quadratic Gaussian Models** The linear-quadratic Gaussian model, or LQG, consists of an LDS and a cost function that is quadratic in the actions taken and the

latent variables  $X_t$ . Formally, the cost function  $\Psi^x$  is given by

$$(2.24) \quad \Psi^x = \sum_{\tau=1}^T \psi^x(X_\tau, u_\tau) + X_{T+1}^\top W_{x,T} X_{T+1},$$

where  $\psi^x(X, u) = X^\top W_x X + 2u^\top W_{x,u} X + u^\top W_u u$  is the per-time-step cost and  $T$  is the planning horizon. The cost matrices are constrained so that both

$$\begin{pmatrix} W_x & W_{x,u}^\top \\ W_{x,u} & W_u \end{pmatrix}$$

and  $W_{x,T}$  be symmetric positive semidefinite so that the cost function is bounded from below and can thus sensibly be minimized, and that  $W_u$  be symmetric positive definite so that a unique minimum of  $\Psi^x$  exists.

The optimal action  $u_t^*$  at each time step minimizes the expected value of  $\Psi^x$ ; it is a linear function of the Kalman filter state estimate  $x_t^-$ :

$$(2.25) \quad u_t^* = \Pi_t^x x_t^-.$$

The parameters of this equation are computed through a Riccati recursion that is initialized by  $V_{T+1}^x = W_{x,T}$  and iterated by

$$(2.26) \quad V_t^x = W_x + A^\top V_{t+1}^x A - (W_{x,u}^\top + A^\top V_{t+1}^x B)(W_u + B^\top V_{t+1}^x B)^{-1}(W_{x,u} + B^\top V_{t+1}^x A).$$

$\Pi_t^x$  is given by

$$(2.27) \quad \Pi_t^x \triangleq -(W_u + B^\top V_{t+1}^x B)^{-1}(W_{x,u} + B^\top V_{t+1}^x A).$$

Because the recursion of (2.26) does not depend on the actions taken or observations recorded,  $\Pi_t^x$  can be precomputed, so that actions can be chosen very quickly while the system is being controlled.

**Full-Rank LQGs** Theorem 4.4 refers to the full-rank property of LQGs. A LQG with full rank is one whose LDS is *observable*. An observable LDS is one in which, in the absence of noise (i.e. when  $\omega_s$  and  $\nu_s$  are fixed at  $\mathbf{0}$  for all  $s$ ), for any possible starting value of  $X_t$  and any sequence of actions, the value of  $X_t$  can be determined in finite time using only the outputs  $Y_{t+1}, Y_{t+2}, \dots$ . A necessary and sufficient condition for observability is that the observability matrix  $\mathcal{M}_n$  have rank  $n$ , where

$$(2.28) \quad \mathcal{M}_n = \begin{pmatrix} H \\ HA \\ \vdots \\ HA^{n-1} \end{pmatrix}.$$

If an LDS is not observable, then for some values  $x_t$  of the latent process  $X_t$ , there are infinitely many  $x$  such that  $X_t = x$  induces the same distribution over future observations.

### 2.2.3 Comparing LDS and AR Models

There are two main differences between LDS and AR models that prevent AR models from being as expressive as the LDS. The first stems from the *finite memory* property of AR models, while the second has to do with *process noise* vs. *measurement noise*.

Because the dynamics of the AR model (given by (2.12)) provide that the next observation is a noisy average of the previous  $n$  observations, any event that occurred more than  $n$  time steps in the past has no effect on future observations. For example, in a 2-dimensional AR model, the observation sequences  $\langle 1.0, 1.1, 1.3 \rangle$  and  $\langle 2.0, 1.1, 1.3 \rangle$  induce an identical probability distribution over the next observation; it doesn't matter that one sequence began with 1.0 and the other with 2.0.

In an  $n$ -dimensional LDS, on the other hand, the latent state vector can preserve information about observations further in the past than  $n$  time steps. In fact, it's possible for an observation arbitrarily far in the past to affect the distribution of the next observation in an LDS.

The second difference between LDS and AR models is that the LDS models both process and measurement noise, as opposed to process noise only, as in the AR model. As a rough definition, process noise is when a dynamical system behaves differently than expected, while measurement noise is when sensors measure something other than the “true” value of the variable they are measuring. For example, consider a car with a speedometer and a throttle. If the car is normally expected to travel 60 mph with the throttle at 70%, but it actually travels 61 mph (perhaps because the road is smoother than expected), that is due to process noise. If the car is traveling at 60 mph but the speedometer reads 61 mph, that is due to measurement noise.

This is in some ways a false distinction, as these “separate sources” are not separately observable. However, it is sometimes useful to discuss the two noise types separately as intuitive notions. In addition, the LDS model treats these as separate random processes;  $\omega_t$  is the process noise at time  $t$ , while  $\nu_t$  is the measurement noise. In the AR model, any difference between the expected value of an observation and its actual value affects the distribution of future observations; that means that this difference is modeled as process noise.

Because of these two differences, the AR model can represent a subset of the dynamical systems that the LDS can model.

## CHAPTER III

### EPSRs: The First Class of Nonlinear PSRs

We have seen that linear PSRs represent the state of the system they model by predictions of certain *tests*, where a test is a sequence of future actions and observations. But other types of predictions may also be interesting. For instance, consider the probability that an observation occurs  $k$  steps in the future after we execute a particular sequence of  $k$  actions, ignoring the observations between now and then—that is, a test of the form  $e = a^1 a^2 \dots a^k o^k$ , whose prediction is  $p(e|h^t) = \Pr(o_{t+k} = o^k | h^t, a_{t+1} = a^1, a_{t+2} = a^2, \dots, a_{t+k} = a^k)$ . This is a new type of test, the *e-test* (end test), which will contrast with the *s-tests* (sequence tests) of the linear PSR. These e-tests are the the same tests used by the diversity representation (Rivest & Schapire, 1994); I will come back to this point later in the chapter.

A relevant question is, “Is there a model class that uses predictions of e-tests as its state?” In the deterministic case, the answer is yes, and this model will be the first general nonlinear class of PSRs.<sup>1</sup>

---

<sup>1</sup>The model presented in this chapter was first presented in (Rudary & Singh, 2004); the theoretical results presented here are also quite similar to results presented in that work, though in some cases slightly stronger.

### 3.1 The E-test Prediction Matrix

Recall that the system dynamics matrix  $\mathcal{D}$  is the matrix whose  $i$ th row corresponds to a history  $h_i$ ,  $j$ th column to an s-test  $t_j$ , and  $(i, j)$ th entry to the prediction  $p(t_j|h_i)$ . We would like to construct a similar matrix,  $\mathcal{E}$ , whose  $j$ th column corresponds to an e-test  $e_j$  rather than an s-test. We will call  $\mathcal{E}$  the *e-test prediction matrix*.

Let us first note that  $\text{rk}(\mathcal{E}) \leq \text{rk}(\mathcal{D})$ . This is because each column of  $\mathcal{E}$  is the sum of several columns of  $\mathcal{D}$ : Let  $e_j = a^1 a^2 \cdots a^k o^k$ ; the prediction of  $e_j$  is given by

$$(3.1) \quad p(e_j|\bullet) = \sum_{o^1 \in \mathcal{O}} \sum_{o^2 \in \mathcal{O}} \cdots \sum_{o^{k-1} \in \mathcal{O}} p(a^1 o^1 a^2 o^2 \cdots a^{k-1} o^{k-1} a^k o^k | \bullet),$$

where each summand is the prediction of an s-test.

#### 3.1.1 The Linear EPSR

It is interesting to try to construct an e-test PSR (EPSR) from  $\mathcal{E}$  analogously to the way linear PSRs are constructed from  $\mathcal{D}$ . That is, select a set of columns that span the column space of  $\mathcal{E}$ ; call the tests that correspond to these columns the *core e-tests*, and treat the predictions of these core e-tests as the state of a model. The problem is that this EPSR does not appear to be self-sufficient—the update function includes quantities that are not elements of  $\mathcal{E}$ .

To elaborate, recall the update function for the linear PSR:

$$(3.2) \quad p(q_i|hao) = \frac{p(aoq_i|h)}{p(ao|h)},$$

where  $q_i$  is the  $i$ th core test; the numerator and denominator on the right-hand side are both predictions of s-tests. Since the prediction of every s-test is a linear function of the predictions of the core tests, this can be rewritten as

$$(3.3) \quad p(q_i|hao) = \frac{m_{aoq_i}^\top p(Q|h)}{m_{ao}^\top p(Q|h)}.$$

However, let us consider the case of a core set of e-tests,  $R$ . If  $r_i$  is the  $i$ th core e-test, its update equation is

$$(3.4) \quad p(r_i|hao) = \frac{p(aor_i|h)}{p(ao|h)}.$$

Here, the denominator is the prediction of an e-test (and thus a linear function of  $p(R|h)$  by the definition of  $R$ ), but the numerator is *not* the prediction of an e-test (nor, for that matter, of a single s-test). Because it is not an e-test, its prediction will not generally be a function of the predictions of the core e-tests. However, there are two interesting special cases in which the update function *can* be computed: the case where  $\text{rk}(\mathcal{D}) = \text{rk}(\mathcal{E})$  and the case in which the system is deterministic.

When the e-test prediction matrix has the same rank as the system dynamics matrix, the system dynamics matrix can be written as a linear function of the e-test prediction matrix. To see this, find  $Q$ , a core set of s-tests, and  $R$ , a core set of e-tests. Then  $\mathcal{D}(Q)$  (the  $\infty \times |Q|$  matrix made up of the columns of  $\mathcal{D}$  corresponding to the s-tests in  $Q$ ) and  $\mathcal{E}(R)$  (the  $\infty \times |R|$  matrix made up of the columns of  $\mathcal{E}$  corresponding to the e-tests in  $R$ ) will both have rank  $n = \text{rk}(\mathcal{D}) = \text{rk}(\mathcal{E})$ . Since every e-test's prediction at any history is the sum of the predictions of some s-tests at that history, and every s-test's prediction is a linear function of the predictions of the s-tests in  $Q$ , we can write  $\mathcal{E}(R) = \mathcal{D}(Q)M_{Q,R}$ , where  $M_{Q,R}$  is some  $n \times n$  matrix. Because  $\text{rk}(\mathcal{E}(R)) = \text{rk}(\mathcal{D}(Q)) = n$ , the rank of  $M_{Q,R}$  must also be  $n$ . This means that  $M_{Q,R}$  is invertible, and we can write

$$(3.5) \quad \mathcal{D}(Q) = \mathcal{E}(R)M_{Q,R}^{-1}.$$

Since  $p(Q|h)$  (resp.  $p(R|h)$ ) is just the row of  $\mathcal{D}(Q)$  (resp.  $\mathcal{E}(R)$ ) corresponding to the particular history  $h$ , it also holds that

$$(3.6) \quad p(Q|h) = (M_{Q,R}^{-1})^\top p(R|h).$$

We can then construct any column of  $\mathcal{D}$  as a linear function of  $\mathcal{E}(R)$ : the  $j$ th column of  $\mathcal{D}$  corresponds to some s-test  $t_j$ , whose predictions can be computed by  $p(t_j|h) = m_{t_j}^\top p(Q|h)$ . Therefore this column can be written as

$$(3.7) \quad \mathcal{D}(t_j) = \mathcal{D}(Q)m_{t_j}$$

$$(3.8) \quad = \mathcal{E}(R)M_{Q,R}^{-1}m_{t_j}.$$

This is quite germane to the problem of updating the predictions of  $R$  because  $p(aor_i|h)$  can be written as the sum of the predictions of several s-tests; if  $r_i = a^1a^2 \cdots a^k o^k$ , then

$$(3.9) \quad p(aor_i|h) = \sum_{o^1 \in \mathcal{O}} \sum_{o^2 \in \mathcal{O}} \cdots \sum_{o^{k-1} \in \mathcal{O}} p(aoa^1o^1a^2o^2 \cdots a^k o^k|h)$$

$$(3.10) \quad = \left( \sum_{o^1 \in \mathcal{O}} \sum_{o^2 \in \mathcal{O}} \cdots \sum_{o^{k-1} \in \mathcal{O}} m_{aoa^1o^1a^2o^2 \cdots a^k o^k} \right)^\top (M_{Q,R}^{-1})^\top p(R|h)$$

$$(3.11) \quad \triangleq w_{aor_i}^\top p(R|h).$$

Since  $ao$  is an e-test (for any  $a \in \mathcal{A}$  and  $o \in \mathcal{O}$ ), its prediction is a linear function of the predictions of the core e-tests; define  $w_{ao}$  as the vector that satisfies  $p(ao) = w_{ao}^\top p(R|h)$ . The update equation (3.4) can thus be rewritten as

$$(3.12) \quad p(r_i|hao) = \frac{w_{aor_i}^\top p(R|h)}{w_{ao}^\top p(R|h)},$$

and  $R$  is a core set of e-tests for a *linear EPSR*. Thus we have proven the following theorem:

**Theorem 3.1.** *For any discrete dynamical system with system dynamics matrix  $\mathcal{D}$  and e-test prediction matrix  $\mathcal{E}$ , and any set of e-tests  $R$ , if  $\text{rk}(\mathcal{D}) = \text{rk}(\mathcal{E}) = \text{rk}(\mathcal{E}(R)) = |R|$ , then for every history  $h$ ,  $p(R|h)$  is the state (at  $h$ ) of a linear EPSR (with core e-tests  $R$ ) that models the dynamical system.*

Unfortunately, there seems to be little reason to choose a linear EPSR over a linear PSR based on s-tests. In particular, both models will have the same dimension because the derivation depends upon  $\mathcal{E}$  having the same rank as  $\mathcal{D}$ . A preferable model would be one in which

$$(3.13) \quad p(aor_i|h) = f_{aor_i}(p(R|h))$$

holds for some history-independent function  $f_{aor_i}$  without requiring that  $\mathcal{E}$  has the same rank as  $\mathcal{D}$ . In fact, there is an interesting class of systems for which this holds, and that is the class of deterministic systems.

### 3.2 The Deterministic EPSR

In deterministic systems, the prediction of every e-test and s-test is binary; all entries of  $\mathcal{D}$  and  $\mathcal{E}$  are ones and zeros. Among other things, this has the effect that, for any e-test  $e$ ,

$$(3.14) \quad p(aoe|h) = \begin{cases} p(ae|h) & p(ao|h) = 1 \\ 0 & p(ao|h) = 0. \end{cases}$$

This is the key fact that allows an EPSR to be defined for *all* deterministic systems, regardless of the relative ranks of the e-test prediction matrix and system dynamics matrix (which, as I will show, can be quite different).

**Theorem 3.2.** *For any discrete, deterministic dynamical system with e-test prediction matrix  $\mathcal{E}$ , and any finite set of e-tests  $R$ , if  $\text{rk}(\mathcal{E}) = \text{rk}(\mathcal{E}(R)) = |R|$ , then for every history  $h$ ,  $p(R|h)$  is the state (at  $h$ ) of an EPSR (with core e-tests  $R$ ) that models the dynamical system.*

*Proof.* All that must be shown is that the predictions  $p(R|h)$  can be updated when a new action  $a$  is taken and observation  $o$  observed. From (3.14), it follows that

$p(aor_i|h) = p(ao|h)p(ar_i|h)$ ; substituting into (3.4), the update becomes

$$(3.15) \quad p(r_i|hao) = \frac{p(ao|h)p(ar_i|h)}{p(ao|h)}$$

$$(3.16) \quad = p(ar_i|h)$$

$$(3.17) \quad = w_{ar_i}^\top p(R|h).$$

Canceling the  $p(ao|h)$  factors in going from (3.15) to (3.16) is valid because  $ao$  will never be observed when  $p(ao|h) = 0$ , so computing the update will never result in a division by zero. (3.17) follows from (3.16) because  $p(e|h)$  is a history-independent linear function of  $p(R|h)$  for any e-test  $e$  by the assumptions of the theorem; without loss of generality, the weights of that linear combination can be written as  $w_{ar_i}$ .  $\square$

Even though the update equations for the deterministic EPSR are linear, this is a *nonlinear* PSR. This because a nonlinear computation is required to obtain the prediction of an arbitrary  $s$ -test whenever  $\text{rk}(\mathcal{E}) < \text{rk}(\mathcal{D})$ .

**Corollary 3.3.** *For a deterministic EPSR with core e-tests  $R$ , the prediction of any  $s$ -test is a (generally nonlinear) function of  $p(R|h)$ .*

*Proof.* The prediction of an arbitrary s-test  $s = a^1 o^1 a^2 o^2 \dots a^k o^k$  is computed by

$$(3.18) \quad p(s|h) = \Pr(o_{t+1} = o^1, \dots, o_{t+k} = o^k | h, a_{t+1} = a^1, \dots, a_{t+k} = a^k)$$

$$(3.19) \quad = \Pr(o^1 | ha_1) \Pr(o^2 | ha^1 o^1 a^2) \dots \Pr(o^k | ha^1 o^1 \dots a^{k-1} o^{k-1} a^k)$$

$$(3.20) \quad = \Pr(o^1 | ha^1) \Pr(o^2 | ha^1 a^2) \dots \Pr(o^k | ha^1 a^2 \dots a^{k-1} a^k)$$

$$(3.21) \quad = (w_{a^1 o^1}^\top p(R|h)) (w_{a^1 a^2 o^2}^\top p(R|h)) \dots (w_{a^1 a^2 \dots a^{k-1} a^k o^k}^\top p(R|h))$$

(3.19) is just an application of the chain rule of probability to (3.18). In going from (3.19) to (3.20), the observations are eliminated from the conditions by noting that in a deterministic system, the observation emitted depends only on the sequence of

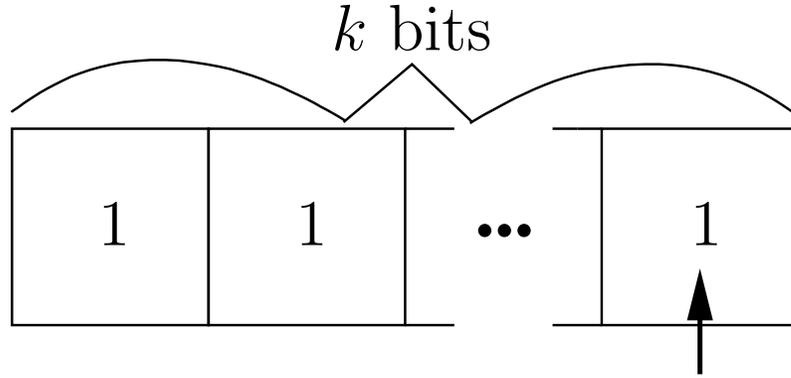


Figure 3.1: The  $k$ -bit rotate register. The value of the rightmost bit is observed. At each time step, this bit can be flipped or the register can be rotated either to the right or to the left with wraparound.

actions executed up to that point. Each probability in (3.20) is the prediction of an e-test; these can be rewritten as linear functions of the predictions of the core e-tests as in (3.21). □

### 3.2.1 EPSRS May Be Exponentially Smaller Than Linear PSRs and POMDPs

The advantage of the deterministic EPSR over the linear PSR comes whenever the rank of the e-test prediction matrix  $\mathcal{E}$  is smaller than the rank of the system dynamics matrix  $\mathcal{D}$ ; the size of the state representation for the two models is the same as those respective ranks. In fact,  $\text{rk}(\mathcal{E})$  may be *exponentially* smaller than  $\text{rk}(\mathcal{D})$ , and is never larger.

Figure 3.1 illustrates a dynamical system in which the EPSR uses exponentially fewer tests than the linear PSR model of the system.<sup>2</sup> It consists of a  $k$ -bit shift-register. There are two observations:  $O_1$  is observed if the rightmost bit is 1 and  $O_0$  is observed if that bit is 0. The three actions are  $A_R$ , which shifts the register one place to the right (with the rightmost bit wrapping around into the leftmost position),  $A_L$ , which does the opposite, and  $A_F$ , which flips (i.e., inverts) the leftmost bit.

<sup>2</sup>This example is due to Rivest and Schapire (1994).

The system dynamics matrix for this system has rank  $2^k$ . Start with  $2^k$  histories  $h_1, h_2, \dots, h_{2^k}$ , such that observing the history  $h_i$  means that the numerical value of the register is  $i$ . Take the  $2^k$  s-tests  $s_1, s_2, \dots, s_{2^k}$ , such that each s-test has an action sequence of  $k$  consecutive  $A_L$  actions and  $s_j$ 's (binary) observation sequence corresponds to the numerical value  $j$ . Then

$$(3.22) \quad p(s_j|h_i) = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$$

and the submatrix of  $\mathcal{D}$  whose  $i$ th row corresponds to  $h_i$  and whose  $j$ th column corresponds to  $s_j$  is the  $2^k \times 2^k$  identity matrix. Thus  $\text{rk}(\mathcal{D})$  must be at least  $2^k$ .

By contrast, the rank of  $\mathcal{E}$  in this system is only  $k + 1$ . This can be seen by the following argument: For a given history, there are  $2k$  equivalence classes of e-tests. For  $i = 1, \dots, k$ , predictions of tests in the  $i$ th equivalence class are 1 when the  $i$ th bit is 1 and 0 when it is 0. For  $i = k + 1, \dots, 2k$ , predictions of tests in the  $i$ th equivalence class are 0 when  $(i - k)$ th bit is 1 and 0 when it is 1. If you create the core set of e-tests  $R$  so that  $r_i$  is taken from the  $i$ th equivalence class for  $i = 1, \dots, k + 1$ , you can compute the prediction of an e-test  $e_j$  taken from equivalence class  $j$  by the function

$$(3.23) \quad p(e_j|h) = \begin{cases} p(r_j|h) & j = 1, \dots, k + 1 \\ p(r_1|h) + p(r_{k+1}|h) - p(r_j|h) & \text{otherwise.} \end{cases}$$

Thus,  $\text{rk}(\mathcal{E})$  may be *no more* than  $k + 1$ , and the size of the EPSR is exponentially smaller than the size of the equivalent linear PSR, and by Corollary 2.2, therefore exponentially smaller than the size of the equivalent POMDP.

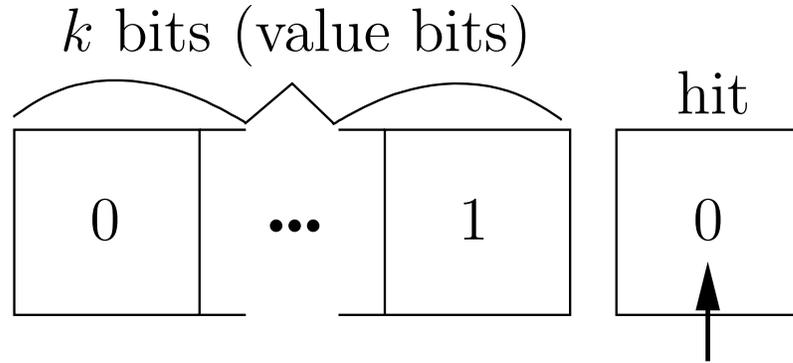


Figure 3.2: The  $k$ -bit hit register. There are  $k$  value bits and 1 special “hit bit.” The behavior of this system is described in the text.

### 3.2.2 EPSRs and the Diversity Representation

The diversity representation (Rivest & Schapire, 1994) is closely related to the deterministic EPSR. Both are models of deterministic systems and both represent state with predictions of e-tests. They differ, however, in the number of predictions they require. The diversity automaton uses as many e-tests as there are *distinct columns* in the e-test prediction matrix  $\mathcal{E}$ , while the EPSR uses as many as there are *linearly independent columns* in  $\mathcal{E}$ . Assuming that the system can be modeled by a POMDP with  $n$  (non-redundant) states, there are at least  $\lg(n)$  and at most  $2^n$  distinct columns in  $\mathcal{E}$ , and these bounds are tight. Thus the diversity representation can be exponentially smaller *or* exponentially bigger than the POMDP model. While the EPSR uses the same notion of tests as the diversity representation, the use of linear independence instead of equivalence classes means that while it may be exponentially *smaller* than a POMDP or linear PSR representation, it will never be larger.

Figure 3.2 shows an example of a system whose diversity representation is exponentially larger than its EPSR model.<sup>3</sup> The hit register consists of a  $k$ -bit value

<sup>3</sup>This example is due to Rivest and Schapire (1994).

register ( $x$ ) and a special “hit bit.” The POMDP model of this system has  $2^k + 1$  states—one in which the hit bit is 1 (and the value register  $x$  is all zeroes) and  $2^k$  states in which the hit bit is 0 and  $x$  takes on its  $2^k$  possible configurations. There are  $k + 1$  actions: a flip action  $F_i$  for each value bit  $i$  that inverts bit  $i$  if the hit bit is not set (and has no effect otherwise) and a set action  $S_h$  that sets the hit bit if all the value bits are 0 (and has no effect otherwise). There are two observations:  $O_s$  is observed if the hit bit is set, and  $O_c$  is observed otherwise. The diversity representation of this system requires  $O(2^{2^k})$  predictions to model the state:

Using these actions, for any subset  $X$  of the  $[k]$ -bit vectors, it is possible to construct a test which is true if and only if the initial state begins with  $x \in X$  or [the hit bit is 1] initially. (Selective complementation can bring  $x$  into the all-zero state iff it was originally in some particular  $[k]$ -bit state  $y$ ; [the hit bit can then be set], otherwise the original state can be restored by undoing the selective complementation. This can be repeated for each  $y \in X$ .) (Rivest & Schapire, 1994, pg. 5)

However, since the POMDP has only  $2^k + 1$  states, the rank of the system dynamics matrix (and thus the e-test prediction matrix) is no more than  $2^k + 1$ , and the EPSR is exponentially smaller than the diversity representation.

These two examples illustrate the following result.

**Theorem 3.4.** *For any deterministic discrete dynamical system, the size of the EPSR representation is no larger than the smaller of the diversity representation or the linear PSR that model the system, and it may be exponentially smaller than either.*

### 3.3 Remarks

In this chapter, I have presented the first nonlinear PSR for any general class of systems using a different type of test, the e-test. I proved that in some deterministic systems, the EPSR model is exponentially smaller than linear PSRs (and thus

POMDPs), and is never larger. Similarly, the EPSR may be exponentially smaller than the diversity representation of a system, and is never larger.

But moving away from linear models is only one departure from the original linear PSR model. In the remaining chapters, I will explore a predictive state model that moves away from the setting with discrete actions and observations.

## CHAPTER IV

### The Predictive Linear Gaussian Model

While the previous chapter focused on a nonlinear PSR model, I now return to a linear model; however, I will now consider systems with continuous actions and observations. I present the predictive linear-Gaussian model (PLG), a predictive state representation that models discrete-time dynamical systems with real-vector-valued actions and observations. This unifies and extends previous work, in which PLGs were developed that modeled uncontrolled systems with scalar-valued observations (Rudary, Singh, & Wingate, 2005) and controlled systems with scalar-valued observations and vector-valued actions (Rudary & Singh, 2006).

Whereas the predictive state models presented up to this point used probabilities of selected future trajectories to represent state, the PLG summarizes history with a joint *probability distribution* over a set of future trajectories. To explain this distribution more clearly and build the intuition of the PLG model, I first present the special case of the uncontrolled, scalar-observation PLG (first described by Rudary et al. (2005)), then develop the more general version that models controlled systems with vector-valued actions and observations.

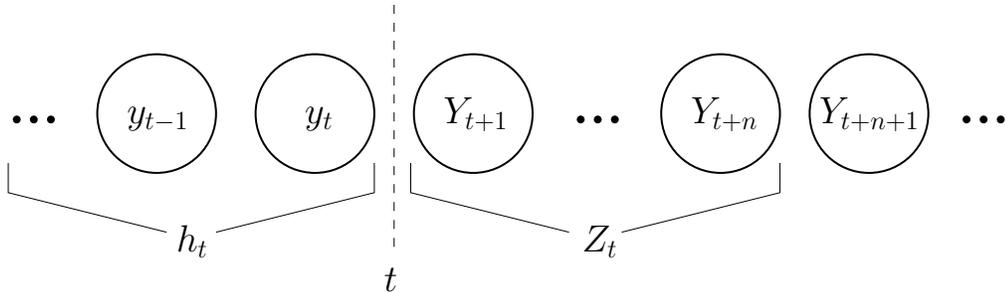


Figure 4.1: A dynamical system with real-valued, scalar observations. The history of observations through  $t$  is denoted  $h_t$ ;  $Z_t$  indicates the  $n$  observations immediately following time  $t$ . Uppercase variable names (e.g.  $Y_{t+1}$ ) indicate random variables, and lowercase names ( $y_t$ , etc.) indicate realizations of random variables.

#### 4.1 The Scalar PLG

The scalar PLG models uncontrolled systems with real-valued observations. The situation is illustrated in Figure 4.1. The observation at time  $t$  is modeled by the random variable  $Y_t$ , which may take on any real value. At time  $t$ , the past observations  $y_1, y_2, \dots, y_t$  have been observed; this is called  $h_t$ , the *history up to time  $t$* . The model must predict the distribution of the future observations  $Y_{t+1}, Y_{t+2}, \dots$  conditioned on this history. This illustrates a convention used throughout: a random variable is set in uppercase, but a realization of that variable is set in lowercase; hence  $y_t$  for the observation at time  $t$  that has already been observed, but  $Y_{t+1}$  for the observation at time  $t + 1$  that must be predicted.

In the setting discussed here, the future observations are jointly distributed according to the multivariate Gaussian distribution. When an  $r$ -dimensional random vector  $X$  is distributed according to a Gaussian distribution with mean  $\mu$  and vari-

ance  $\Sigma$  (written  $X \sim \mathcal{N}(\mu, \Sigma)$ ), the probability density at  $X = x$  is given by

$$(4.1) \quad p(X = x; \mu, \Sigma) = (2\pi)^{-r/2} |\Sigma|^{-1/2} e^{-(x-\mu)^\top \Sigma^{-1} (x-\mu)},$$

where  $|\cdot|$  denotes the determinant of a matrix. The form of the density function makes it apparent that this distribution is completely characterized by its mean  $\mu$  and covariance  $\Sigma$ . Thus, if the mean and variance of the future observations (conditioned on the observed history) are known, they can serve as state for a model of the system.

The difficulty is that there are an infinite number of future observations, so  $\mu$  and  $\Sigma$  are infinite matrices. The PLG solves this problem by positing that *any* future observation can be written as a linear function the *next  $n$  observations* and some Gaussian random variables whose distributions do not depend on history, where  $n$  is a quantity called *the dimension* of the system. Since any linear function of Gaussian random variables results in another Gaussian random variable, this preserves the assumption that the future is normally distributed. It also means that the distribution of the next  $n$  observations, collected together into the random vector

$$(4.2) \quad Z_t = \begin{pmatrix} Y_{t+1} \\ Y_{t+2} \\ \vdots \\ Y_{t+n} \end{pmatrix},$$

is sufficient to construct the entire distribution of the future. Since  $Z_t|h_t$  is normally distributed, its mean  $\mu_t$  and variance  $\Sigma_t$  are the state variables.

The assumption that future observations are linear functions of  $Z_t$  and Gaussian noise terms can be written

$$(4.3) \quad Y_{t+n+1} = g^\top Z_t + \varepsilon_{t+n+1}.$$

Table 4.1: Scalar PLG notation.

Variable	Description
$n$	The dimension of a model or dynamical system.
$Y_t$	The observation at time $t$ .
$h_t$	The history of observations through time $t$ , i.e. $y_1 y_2 \cdots y_t$ . Note the convention that a random variable is set in uppercase (e.g. $Y_t$ ) but that a realization of that variable is set in lowercase ( $y_t$ ).
$Z_t$	The random vector containing the $n$ observations occurring after $t$ .
$\mu_t, \Sigma_t$	The history-conditional mean and variance of $Z_t$ . That is, $Z_t h_t \sim \mathcal{N}(\mu_t, \Sigma_t)$ .
$g, C, \sigma^2, \mu_0, \Sigma_0$	The parameters of a scalar PLG.
$e_i$	The $i$ th column of the $n \times n$ identity matrix.
$X^i$	The $i$ th element of a vector $X$ .

Here,  $g$  is the linear trend in the observations and  $\varepsilon_{t+n+1}$  is the Gaussian noise term. Predictions about observations that are further in the future can be obtained by repeated application of this formula with increasing values of  $t$ . (4.3) describes the core dynamics of the model: The observations follow a linear trend but are perturbed by noise.

The distribution of this noise term is a choice that greatly affects the dynamics of the model. If  $\varepsilon_{t+n+1}$  is chosen to be i.i.d. Gaussian noise, this becomes the autoregressive model (see Section 2.2.1), but viewed from a different perspective; instead of specifying how the next observation depends on the last  $n$  observations (as in the autoregressive model), the PLG would specify how the  $(n + 1)$ st observation in the future depends on the *next*  $n$  observations. Despite this difference, the models would be equivalent.

Autoregressive models, however, are somewhat impoverished in terms of representational power. In many systems of interest to modelers, there are two sources of randomness: process noise (the system behaved differently than expected) and measurement noise (the sensors used were imperfect).<sup>1</sup> Autoregressive models are able to model process noise but *not* measurement noise, because the noise distribution is

<sup>1</sup>This is discussed in further detail in Section 2.2.3.

independent of history. Thus, a noise distribution should be considered that allows for process *and* measurement noise.

One such form of the noise allows  $\varepsilon_{t+n+1}$  to covary with  $Z_t$ ; this is the type of noise used in the PLG. The noise distribution can then be written as

$$(4.4) \quad \varepsilon_{t+n+1}|h_t \sim \mathcal{N}(0, \sigma^2)$$

$$(4.5) \quad \text{Cov}[Z_t, \varepsilon_{t+n+1}|h_t] = C.$$

That is, until  $Y_{t+1}$  or a later observation is observed,  $\varepsilon_{t+n+1}$  is assumed to be mean-zero Gaussian noise. However, if the next  $n$  observations ( $y_{t+1}, y_{t+2}, \dots, y_{t+n}$ ; collectively  $z_t$ ) differ from our history-conditional expectations ( $\mathbb{E}[Z_t|h_t] = \mu_t$ ), the (conditional) mean and variance of  $\varepsilon_{t+n+1}$  will be adjusted according to the history-independent covariance vector  $C$ .<sup>2</sup>

The fact that the noise term covaries with  $Z_t$  means that the distribution of future observations can depend on observations arbitrarily far in the past; if the noise were i.i.d., each observation's distribution would depend only on the  $n$  observations actually preceding it. For example, take two long trajectories,  $h_T^1$  and  $h_T^2$ , where  $T > n$ . Further, suppose that they are identical except in their first observation. If the PLG is parameterized so that  $C = \mathbf{0}$ , i.e. so that the noise is i.i.d., the distribution of  $Y_{t+n+1}|h_T^1$  will be identical to the distribution of  $Y_{t+n+1}|h_T^2$ . On the other hand, if  $C \neq \mathbf{0}$ , the two distributions will differ (although possibly by a very small amount). This dependence on the far past means that just maintaining the *values* of the *last*  $n$  observations is insufficient for making predictions about the future. However, maintaining the *distribution* of the *next*  $n$  observations *is* sufficient to make these predictions.

---

<sup>2</sup> $C$  is not an entirely free parameter. In particular, it is constrained with respect to the parameters  $g$  and  $\sigma^2$ ; see further discussion following (4.9).

It is therefore important to be able to update the mean vector  $\mu_t$  and covariance matrix  $\Sigma_t$  as new observations are seen and time moves forward. That is, in order for  $\mu_t$  and  $\Sigma_t$  to be feasible as state variables, it must be possible to use the previous information ( $\mu_t = \mathbb{E}[Z_t|h_t]$  and  $\Sigma_t = \text{Var}[Z_t|h_t]$ ) along with the new information (the observation  $Y_{t+1} = y_{t+1}$ ) to compute the new values  $\mu_{t+1} = \mathbb{E}[Z_{t+1}|h_t, Y_{t+1} = y_{t+1}]$  and  $\Sigma_{t+1} = \text{Var}[Z_{t+1}|h_t, Y_{t+1} = y_{t+1}]$ . This update can be achieved by noting that  $Y_{t+1}$  and  $Z_{t+1}$  are jointly Gaussian random variables, and that

$$(4.6) \quad \begin{pmatrix} Z_t \\ Y_{t+n+1} \end{pmatrix} = \begin{pmatrix} Y_{t+1} \\ Z_{t+1} \end{pmatrix}.$$

Thus the joint distribution of  $Y_{t+1}|h_t$  and  $Z_{t+1}|h_t$  can be computed from two distributions that are already known: those of  $Z_t|h_t$  and  $Y_{t+n+1}|h_t$ . There is a standard result that allows a conditional distribution to be computed from a joint Gaussian distribution (e.g., Catlin, 1989), and it can be applied here.

The details of the derivation may be found in Appendix A.1.1; to present the results of the derivation, three new matrices must be introduced. The first two are  $e_1$  and  $e_n$ , two column vectors that are the first and  $n$ th columns, respectively, of the  $n \times n$  identity matrix. The last,  $G$ , is the  $n \times n$  matrix that satisfies  $\mathbb{E}[Z_{t+1}|h_t] = G\mathbb{E}[Z_t|h_t]$ . That is, it selects the last  $n - 1$  elements of  $\mu_t = \mathbb{E}[Z_t|h_t]$  and moves them each up by one (because the first  $n - 1$  elements of  $Z_{t+1}$  are the same as the last  $n - 1$  elements of  $Z_t$ ), and adds a new  $n$ th element equal to  $g^\top \mu_t$ .  $G$  is given by

$$(4.7) \quad G = \begin{pmatrix} \mathbf{0} & \vdots & I_{n-1} \\ \cdots & \cdots & \cdots \\ & & g^\top \end{pmatrix},$$

where  $I_{n-1}$  is the  $(n - 1) \times (n - 1)$  identity matrix.  $G$ ,  $e_1$ , and  $e_n$  can be seen mostly as “shifting” matrices; their purpose is to shift elements around, so that the matrix equations used to produce means and variances of an  $n$ -vector and scalar

value, respectively, can be altered to produce means and variances of a scalar value and an  $n$ -vector, respectively.

Using these shifting matrices, the update equations can now be written as

$$(4.8) \quad \mu_{t+1} = G\mu_t + \frac{(G\Sigma_t + e_n C^\top) e_1}{e_1^\top \Sigma_t e_1} (y_{t+1} - e_1^\top \mu_t) \quad \text{and}$$

$$(4.9) \quad \Sigma_{t+1} = G\Sigma_t G^\top + G C e_n^\top + e_n C^\top G^\top + \sigma^2 e_n e_n^\top - \frac{(G\Sigma_t + e_n C^\top) e_1 e_1^\top (\Sigma_t G^\top + C e_n^\top)}{e_1^\top \Sigma_t e_1}.$$

**Intuition About Update Equations** The first term of (4.8),  $G\mu_t$ , gives the expected value of  $Z_{t+1}|h_t$ ; that is, ignoring the information imparted by  $y_{t+1}$ . The estimate of the mean is then adjusted by an amount that is proportional to both the difference between the observation and its expectation (that is,  $y_{t+1} - e_1^\top \mu_t$ ) and the covariance between each element of  $Z_{t+1}$  and the observation (i.e.  $(G\Sigma_t + e_n C^\top) e_1$ ), but inversely proportional to the variance of the observation ( $e_1^\top \Sigma_t e_1$ ). That is, the more we were wrong about the next observation, the more we change our predictions about the rest of the future; the more the future is correlated with the next observation is, the more we change our predictions; and the more unsure we are about the next observation, the *less* we change our prediction.

Similarly, the first four terms of (4.9),  $G\Sigma_t G^\top + G C e_n^\top + e_n C^\top G^\top + \sigma^2 e_n e_n^\top$ , give the marginal variance of  $Z_{t+1}|h_t$ . This is then adjusted downward by a quantity that increases as the product of the covariance between  $Z_{t+1}$  and  $Y_{t+1}$  and its transpose and as the inverse of the variance of  $Y_{t+1}$ . Intuitively, the more the value of the next observation will affect the expected value of future observations, the more actually observing it reduces our uncertainty about the future.

**Constraints Between Parameters** (4.9) introduces a constraint between  $g$ ,  $C$ , and  $\sigma^2$ . Because of the subtraction in the variance update equation,  $C$  must be selected so that the final term of (4.9) is smaller than the sum of the first four terms, in the sense that  $\Sigma_{t+1}$  must be symmetric positive definite (i.e. a valid covariance matrix) for all  $t$ . This is equivalent to requiring the covariance matrix of the joint distribution of  $Y_{t+n+1}$  and  $Z_t$  be symmetric positive definite for all  $t$ ; this covariance matrix is given by

$$(4.10) \quad \text{Var} \left[ \begin{pmatrix} Z_t \\ Y_{t+n+1} \end{pmatrix} \middle| h_t \right] = \begin{pmatrix} \Sigma_t & \Sigma_t g + C \\ g^\top \Sigma_t + C^\top & g^\top \Sigma_t g + g^\top C + C^\top g + \sigma^2 \end{pmatrix}.$$

With the update equations in place, the specification of the scalar PLG model is complete. It is parameterized by the linear trend parameter  $g$ , the noise parameters  $C$  and  $\sigma^2$ , and the initial state  $\mu_0$  and  $\Sigma_0$ .

The earlier discussion indicated that the PLG can model a richer class of systems than can autoregressive models. In fact, as a corollary of Theorem 4.1 in the next section,  $n$ -dimensional scalar PLGs have as much modeling power as  $n$ -dimensional linear dynamical system (LDS) models with scalar observations. In particular, given any  $n$ -dimensional LDS with scalar observations, an equivalent scalar PLG can be found.

## 4.2 The Vector-Valued PLG

When extending the scalar PLG to deal with vector-valued observations and actions, there are several challenges to overcome. The most basic, and probably the most important, is to decide with observations comprise  $Z_t$  and then to work out the details of the dynamics based on this decision. The effects of the actions must also be determined. Finally, I must also show that the vector-valued PLG can represent

Table 4.2: PLG notation.

Variable	Description
$n$	The dimension of a model or dynamical system.
$Y_t$	The observation vector at time $t$ .
$m$	The length of an observation vector.
$u_t$	The action vector at time $t$ .
$l$	The length of an action vector.
$h_t$	The history of interaction through time $t$ , i.e. $u_1y_1u_2y_2 \cdots u_t y_t$ .
$Z_t$	A random vector containing a set of observations occurring after time $t$ ; this vector contains $n$ elements. These observations are selected so that the history-conditional distribution of $Z_t h_t$ is sufficient to compute the history-conditional distribution of any future observation.
$\mu_t, \Sigma_t$	The history-conditional mean and variance of $Z_t$ . That is, $Z_t h_t \sim \mathcal{N}(\mu_t, \Sigma_t)$ .
$J, G, \Gamma_1, \dots, \Gamma_{\tau_{\max}}, C_\eta, \Sigma_\eta, \mu_0, \Sigma_0$	The parameters of a PLG.
$\Sigma_{\text{adj}}$	The variance adjustment; an additional parameter of the variance-adjusted PLG.
$X^i$	The $i$ th element of a vector $X$ .

any system that an LDS of the same dimension can; as discussed in Section 4.3, this will require a slight change to the next-observation semantics of the model presented here.

As in the scalar version, the state of the vector-valued PLG is the mean ( $\mu_t$ ) and variance ( $\Sigma_t$ ) of  $Z_t|h_t$ , where  $Z_t$  is an  $n$ -dimensional vector of future observations;  $n$  is called the *dimension* of the PLG model. As before,  $Z_t|h_t$  is distributed according to a multivariate Gaussian distribution. The PLG then uses this distribution to compute the distribution of any desired future observation(s).

#### 4.2.1 Makeup of $Z_t$

The concept at the heart of the vector-valued PLG is how  $Z_t$  is constructed. Each element of the vector  $Z_t$  is an element of an observation vector that will be observed after time  $t$ . In the scalar PLG,  $Z_t$  is just the concatenation of the next  $n$  observations  $Y_{t+1}, Y_{t+2}, \dots, Y_{t+n}$ . It would be possible to use this concatenation of the next  $n$  observations in the vector-valued PLG as well; in the class of systems we

$Y_{t+3}$	6			
$Y_{t+2}$	4			5
$Y_{t+1}$	1	2		3
	$Y^1$	$Y^2$	$Y^3$	$Y^4$

Figure 4.2: The skyline constraint on  $Z_t$ . This figure shows one possible makeup of  $Z_t$  in a system with dimension  $n = 6$  and observation length  $m = 4$ .  $Z_t^i$  corresponds to the box containing the number  $i$  (so  $Z_t^4 = Y_{t+2}^1$ , for example). Note that this figure forms a skyline—that is, no box is used in  $Z_t$  unless the boxes below it are also used.

consider ( $n$ -dimensional LDSs), knowledge of the distribution of this vector would be sufficient to compute the distribution of any future observation(s). However, this may be inefficient. When  $m > 1$ ,  $Z_t$  would have  $mn$  elements when  $n$  elements should suffice for an  $n$ -dimensional model. Because the number of parameters of the PLG is quadratic in the number of elements in  $Z_t$ , this is unacceptable.

The solution, then, is to pick a subset of those  $mn$  elements.  $Z_t$  consists of  $n$  *observational elements*—that is, elements of observation vectors—whose joint distribution is sufficient to reconstruct the full joint distribution of all future observations. It is of course not generally the case that all such subsets are sufficient; the particular subset chosen depends on the system to be modeled.

For a number of reasons, there are some subsets of the next  $mn$  observational elements that will not be permitted. In particular, if the  $j$ th element of the future observation  $Y_{t+k}$  is included as part of  $Z_t$ ,<sup>3</sup> then the  $j$ th elements of  $Y_{t+1}, Y_{t+2}, \dots, Y_{t+k-1}$  must be included as well. This may be thought of as the “skyline” requirement for  $Z_t$ , because when a diagram is made as in Figure 4.2, it forms a silhouette as of a city skyline. This restriction does not reduce the representative power of the PLG.

<sup>3</sup>It will be useful to refer compactly to specific elements of the  $Y_t$  and  $Z_t$  vectors; the  $i$ th element of  $Z_t$  (resp.  $Y_t$ ) shall be denoted  $Z_t^i$  (resp.  $Y_t^i$ ).

One reason for the skyline requirement is the consistency of the noise terms. In the scalar PLG, only the *last* term of  $\varepsilon_{t+n+1}$  was non-zero because we wanted only one noise variable associated with each observation. If the skyline requirement were violated, an observation would be in non-consecutive  $Z$  vectors. Considering the  $Z_t$  layout shown in Figure 4.2, suppose that the fourth element of  $Z_t$  were removed. Then  $Y_{t+3}^1$  would be an element of  $Z_t$  and  $Z_{t+2}$ , but not  $Z_{t+1}$ . It is not tracked at  $t + 1$ ; when it is reintroduced at  $t + 2$ , should another noise term be applied? This would break the assumption that each observational element has a single noise term associated with it and would lead to conflicting definitions of its distribution. On the other hand, the fact that it is not tracked at  $t + 1$  means that its distributional information is essentially lost, and so information about the original noise term is discarded and must be re-introduced, which suggests that another noise term *should* be used. This conflict argues in favor of the skyline requirement.

Another reason to introduce the skyline requirement is efficiency in the parameters. When an observational element is a member of consecutive  $Z$  vectors, one row of the linear trend  $G$  is predetermined to copy that element. The PLG referred to by Figure 4.2, for example, has a  $G$  whose fourth row copies the sixth element ( $Y_{t+3}^1$ ) onto the fourth ( $Y_{t+2}^1$ ) when  $t$  increases. When an observational element is in non-consecutive  $Z$  vectors, this copying would not be possible, and so an additional row of  $G$  must be made up of free parameters.

The elements that make up  $Z_t$  must be selected so that the distribution of  $Z_t|h_t$  is sufficient to compute the history-conditional distributions of any future observation(s). They must also be selected so that the next observation is a function of  $Z_t$ :

$$(4.11) \quad Y_{t+1} = JZ_t,$$

where the  $m \times n$  matrix  $J$  is a parameter of the model.<sup>4</sup> In many cases when  $m \leq n$ , this requirement can be satisfied by including every element of  $Y_{t+1}$  as the first  $m$  elements of  $Z_t$  and setting  $J$  to be the first  $m$  rows of the  $n \times n$  identity matrix.

#### 4.2.2 Core Dynamics of the Uncontrolled PLG

The core dynamics of the vector-valued PLG are similar to (4.3), the core dynamics of the scalar PLG. However, in this case it is more natural to define the dynamics in terms of  $Z_{t+1}$  and  $Z_t$ , not  $Y_{t+n+1}$  and  $Z_t$ :

$$(4.12) \quad Z_{t+1} = GZ_t + \eta_{t+1}.$$

Here,  $G$  is the linear trend in the predicted observations  $Z_t$  and  $\eta_{t+1}$  is the noise term, a random vector. As in the scalar case, the noise term covaries with  $Z_t$ ; it is distributed according to

$$(4.13) \quad \eta_{t+1}|h_t \sim \mathcal{N}(\mathbf{0}, \Sigma_\eta)$$

$$(4.14) \quad \text{Cov}[Z_t, \eta_{t+1}|h_t] = C_\eta.$$

Note that it is possible (even likely) that a particular observational element  $Y_t^i$  may appear in  $Z_s$  for multiple values of  $s$ . For example, in Figure 4.2, the element  $Y_{t+3}^1$  appears as  $Z_t^6$ ,  $Z_{t+1}^4$ , and  $Z_{t+2}^1$ . In general, this *will* happen whenever  $n > m$  and *may* happen when  $n \leq m$ . It is undesirable for different noise terms to apply to the same observational element when (4.12) is evaluated for different values of  $t$ , as this would lead to conflicting distributions for some observations. Therefore, fix  $\eta_{t+1}^i = 0$  whenever  $Z_t^i$  is not the first appearance of an observational element. In the running example of Figure 4.2, the first and fourth elements of  $\eta_{t+1}$  will be fixed at 0 for all values of  $t$ , but the sixth element may take on nonzero values. The nonzero elements of  $\eta_{t+1}$  correspond to the top shaded box of each “stack” in the figure.

---

<sup>4</sup>This requirement will be relaxed in Section 4.3, which will allow PLGs to model systems with observation dimension  $m$  greater than system dimension  $n$ .

### 4.2.3 Modeling Actions in the PLG

The dynamics of (4.12) don't quite tell the whole story: They ignore the effect that actions have on future observations. The PLG models actions so that that  $u_t$ , the action at time  $t$ , affects the expected value (but not the variance) of future observations—that is,  $Y_{t+1}, Y_{t+2}, \dots$ . Each action's effect on these means is a linear function of the action. With these strictures in place, the core dynamics of the *vector-valued* PLG can be written:

$$(4.15) \quad Z_{t+1} = GZ_t + \sum_{i=1}^{\tau_{\max}} \Gamma_i u_{t+i} + \eta_{t+1}.$$

The model parameters  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  are  $n \times l$  matrices that describe the linear effects of the actions. The upper limit of the sum,  $\tau_{\max}$ , is the maximum look-ahead horizon of  $Z_t$ . That is,  $Z_t$  contains at least one element of  $Y_{t+\tau_{\max}}$  but no elements of  $Y_{t+\tau_{\max}+1}$ . In the example of Figure 4.2,  $\tau_{\max} = 3$ . This means that  $Z_{t+1}$  contains at least one element of  $Y_{t+\tau_{\max}+1}$ , which can be affected by the action  $u_{t+\tau_{\max}}$  but not by  $u_{t+\tau_{\max}+1}$ ; hence  $\tau_{\max}$  is the upper limit on the summation.

As with the noise terms, in order for the effects of the actions to be well-defined, they should only apply to the *first* appearance of a particular observational element in a  $Z_t$  vector. This is accomplished by fixing the  $i$ th row of  $\Gamma_k$  to be  $\mathbf{0}$  when  $Z_t^i$  is not the first appearance of an observational element. Thus, as with the noise term, in the running example, the first and fourth rows of  $\Gamma_1$ ,  $\Gamma_2$ , and  $\Gamma_3$  are fixed at  $\mathbf{0}$ , but the sixth rows may take on any value. In addition, causality must be preserved; therefore, no action may have an effect on elements of  $Z_{t+1}$  that will be observed *before* the action will be taken. In the example, this means that the second row of  $\Gamma_1$  may be nonzero, but the second rows of  $\Gamma_2$  and  $\Gamma_3$  must be  $\mathbf{0}$ , because the actions  $u_{t+2}$  and  $u_{t+3}$  cannot affect elements of  $Y_{t+2}$ .

In controlled systems, the distribution of  $Z_t|h_t$  is undefined without information about the actions that will be executed between now ( $t$ ) and the time that the elements of  $Z_t$  will be observed. Because the actions are not modeled as random variables, it is impossible to marginalize out their effects. This is important because the PLG's state is defined in terms of the distribution of  $Z_t$ . The simplest solution is to assume that future actions will be  $\mathbf{0}$ ; because of the actions' linear effect on the future, this is the assumption that actions that have not yet been taken will not affect the values of future observations.

But note that this is just an assumption to make the state well-defined; the dynamics do not require that the actions actually be zero in order for the model to be correct. When the actual value of an action is determined, its effects are propagated onto the state using the state update equations defined in the next section.

#### 4.2.4 State Update

As in the scalar PLG, the state of the system is summarized by the mean and covariance of the distribution of  $Z_t$ , conditioned on the history of interaction (and assuming future actions to be  $\mathbf{0}$ ):

$$(4.16) \quad Z_t|h_t, u_{t+1} = \mathbf{0}, u_{t+2} = \mathbf{0}, \dots \sim \mathcal{N}(\mu_t, \Sigma_t).$$

With each new time step, a new observation becomes available and another action has been taken; the state of the system must be updated to take this information into account. Again, the update function must compute  $\mu_{t+1} = \mathbb{E}[Z_{t+1}|h_t, Y_{t+1} = y_{t+1}, u_{t+1}, u_{t+2} = \mathbf{0}, \dots]$  and  $\Sigma_{t+1} = \text{Var}[Z_{t+1}|h_t, Y_{t+1} = y_{t+1}, u_{t+1}, u_{t+2} = \mathbf{0}, \dots]$  from  $\mu_t$ ,  $\Sigma_t$ ,  $y_{t+1}$ , and  $u_{t+1}$ . The derivation of these update equations is essentially the same as the derivation of the scalar update equations (4.8) and (4.9), except that the mean update must take the action into account. The action has a linear effect;

this effect,  $L$ , should describe the cumulative effect that the action has on all the observations of  $Z_{t+1}$ , to wit:

$$(4.17) \quad Lu = \mathbb{E}[Z_{t+1}|h_t, u_{t+1} = u, u_{t+2} = \mathbf{0}, \dots] - \mathbb{E}[Z_{t+1}|h_t, u_{t+1} = \mathbf{0}, u_{t+2} = \mathbf{0}, \dots].$$

This is accomplished by defining

$$(4.18) \quad L \triangleq \Gamma_1 + G\Gamma_2 + G^2\Gamma_3 + \dots + G^{\tau_{\max}-1}\Gamma_{\tau_{\max}}.$$

Using this definition of  $L$ , the state update equations are written as

$$(4.19) \quad \mu_{t+1} = G\mu_t + Lu_{t+1} + (G\Sigma_t + C_\eta^\top)J^\top(J\Sigma_tJ^\top)^{-1}(y_{t+1} - J\mu_t)$$

$$(4.20)$$

$$\Sigma_{t+1} = G\Sigma_tG^\top + GC_\eta + C_\eta^\top G^\top + \Sigma_\eta - (G\Sigma_t + C_\eta^\top)J^\top(J\Sigma_tJ^\top)^{-1}J(\Sigma_tG^\top + C_\eta).$$

Comparing these updates to (4.8) and (4.9) helps illustrate the fact that the scalar PLG is just a special case of the vector-valued PLG, with  $\eta_{t+1} = \varepsilon_{t+n+1}e_n$ ,  $\Sigma_\eta = \sigma^2e_n e_n^\top$ ,  $C_\eta = Ce_n^\top$ , and  $J = e_1^\top$ . The derivations of  $L$  and the update equations (4.19) and (4.20) may be found in Appendix A.1.2.

**Constraints Between Parameters** As in the scalar version of the PLG, the variance update equation (4.20) introduces some constraints between  $G$ ,  $C_\eta$ ,  $\Sigma_\eta$ , and  $J$ . Because the last term of (4.20) is *subtracted* from the first four terms and covariance matrices are required to be symmetric positive semidefinite, these parameters are constrained to require that  $\Sigma[t+1]$  is symmetric positive semidefinite for all  $t$ . Equivalently, the following covariance matrix must be symmetric positive semidefinite for all  $t$ :

$$(4.21) \quad \text{Var} \left[ \begin{array}{c} Y_{t+1} \\ Z_{t+1} \end{array} \middle| h_t \right] = \begin{pmatrix} J\Sigma_tJ^\top & J\Sigma_tG^\top + JC_\eta \\ G\Sigma_tJ^\top + C_\eta^\top J^\top & G\Sigma_tG^\top + GC_\eta + C_\eta^\top G^\top + \Sigma_\eta \end{pmatrix}.$$

Failure to satisfy these constraints will lead to an invalid state at some time step, and thus an invalid model.

The predictive linear-Gaussian model can now be defined. It is parameterized by the linear trend parameters  $G$  and  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$ , the “next-observation function”  $J$ , the noise parameters  $\Sigma_\eta$  and  $C_\eta$ , and the initial state  $\mu_0$  and  $\Sigma_0$ . When a new action and observation become available, the state is updated using (4.19) and (4.20). The semantics of the model are given by the state semantics in (4.16), the makeup of  $Z_t$ , and the next-observation distribution of (4.11). Representationally, the PLG is as powerful as a large subset of linear dynamical systems.

**Theorem 4.1.** *Every LDS with  $n$ -dimensional state and in which the vector space  $\{E[Y_t|X_t = x] : x \in \mathbb{R}^n\}$ <sup>5</sup> is  $m$ -dimensional for any  $t$  has an equivalent  $n$ -dimensional PLG, where equivalence means that both models compute the same probability distribution over futures given the same history.*

This theorem can be proven by construction and is quite similar to the proof of Theorem 4.2, a stronger result presented in the next section; the proof of Theorem 4.1 is thus omitted.

I would like to give some intuition as to the condition in this theorem—that is, that the vector space  $\{E[Y_t|X_t = x] : x \in \mathbb{R}^n\}$  has rank  $m$  for some  $t$ . This essentially means that (for some  $t$ ) it is not possible to express the expected value of any element of  $Y_t$  as a linear function of the expected values of the other elements. This is equivalent to requiring that the LDS parameter  $H$  have rank  $m$ . This condition can be removed by altering the the relationship between  $Z_t$  and the next observation as given by (4.11). The next section presents such an alteration.

---

<sup>5</sup> $X_t$  is the value of the latent state process of an LDS; see Section 2.2.2.

### 4.3 The Variance-Adjusted PLG

When the observation vector space is underranked in expectation—that is, when  $\{E[Y_{t+1}|h_t] : h_t \text{ a length-}t \text{ history}\}$  has dimension less than  $m$  for every  $t$ —the PLG is unable to model the system. That is because the variance need not be underranked even when the means *are* linearly dependent—yet this relationship is forced by  $Y_t = JZ_t$ . words, while  $E[Y_t|h_t] = J\mu_t$ , it is *not* necessarily the case that  $\text{Var}[Y_t|h_t] = J\Sigma_t J^\top$ , thus violating the next-observation semantics of (4.11). If the variance relationship *did* hold, it would imply that the space of all possible observations were underranked, not just the space of the expected values of the observations.

For example, suppose we wish to control an oven in an industrial process. The oven has two temperature sensors that are unbiased but whose readings differ from the actual temperature inside the oven by i.i.d. Gaussian noise;<sup>6</sup> these sensors are the only observations (i.e.  $m = 2$ ). In this case,  $E[Y_{t+1}^1|h_t] = E[Y_{t+1}^2|h_t]$ , so the mean vector has a trivial linear dependency. However, because the noise experienced by the two sensors is *independent*, it is *not* the case that  $Y_{t+1}^1|h_t = Y_{t+1}^2|h_t$ . The linear dependence in the means does not carry over to the variance of the two elements of the observation vector. Thus, there is no  $J$  that satisfies  $Y_{t+1} = JZ_t$ .

To eliminate this problem, I propose a variant of the PLG: the variance-adjusted PLG. In this variant, the distribution of the next observation is given by

$$(4.22) \quad Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots \sim \mathcal{N}(J\mu_t, J\Sigma_t J^\top + \Sigma_{\text{adj}}),$$

where  $\Sigma_{\text{adj}}$  is a symmetric matrix. It need not be a valid covariance matrix, but the sum  $J\Sigma_t J^\top + \Sigma_{\text{adj}}$  must be symmetric positive semidefinite for all  $t$ . In all other respects, the variance-adjusted PLG has the same semantics as the standard model.

---

<sup>6</sup>While each sensor may differ from the temperature by an i.i.d. Gaussian random variable, the process as a whole may still have non-i.i.d. noise.

This new formulation requires that the state update equations be altered slightly:

$$(4.23) \quad \mu_{t+1} = G\mu_t + Lu_{t+1} + (G\Sigma_t + C_\eta^\top)J^\top(J\Sigma_tJ^\top + \Sigma_{\text{adj}})^{-1}(y_{t+1} - J\mu_t)$$

$$(4.24) \quad \Sigma_{t+1} = G\Sigma_tG^\top + GC_\eta + C_\eta^\top G^\top + \Sigma_\eta - \\ (G\Sigma_t + C_\eta^\top)J^\top(J\Sigma_tJ^\top + \Sigma_{\text{adj}})^{-1}J(\Sigma_tG^\top + C_\eta)$$

Adjusting the variance of the next observation with the additional parameter  $\Sigma_{\text{adj}}$  adds enough representational power to the PLG to allow it to represent *any* LDS.

**Theorem 4.2.** *Every LDS with  $n$ -dimensional state has an equivalent  $n$ -dimensional variance-adjusted PLG, where equivalence means that both models compute the same probability distribution over futures given the same history.*

This theorem is proven by construction: Given an  $n$ -dimensional LDS, the parameters of an equivalent  $n$ -dimensional variance-adjusted PLG can be computed. This construction can be found in Appendix A.2.

#### 4.4 Optimal Control of the PLG

One of the most important purposes of modeling a controlled dynamical system is to use the model to plan. In planning, the model is used to (attempt to) select actions that minimize some cost function in expectation.

To use the PLG to plan, the first step is to select a class of cost functions. There are two important criteria to consider: First, the cost function must be able to model realistic or important applications. Second, choosing actions that minimize (or at least perform well with respect to) this function must be a tractable problem. Here, the LDS literature provides inspiration—quadratic cost functions are widely used in conjunction with the LDS for control applications; this combination is known as a

linear-quadratic Gaussian model. Minimizing a quadratic function of the state and actions can model minimizing the energy input into a system, getting as close to a target state as possible, and other reasonably interesting tasks. In addition, optimal control under such a cost function is quite tractable. This is also true when the underlying model is a PLG rather than an LDS.

Here I will describe the PLG with quadratic cost (PLGQ), which applies to a PLG model a cost function that is quadratic in the actions  $u_t$  and state variables  $\mu_t$ . I will show that the optimal action in this framework is a linear function of the mean vector  $\mu_t$  and that, under reasonable conditions, a cost function can be selected that yields the same optimal controls for an LDS and its equivalent PLG.

#### 4.4.1 Form of the Cost Function

The cost function has the following form: At each time step  $t$  (up to a finite horizon,  $T$ ), the PLGQ assesses a cost that is quadratic in the mean vector  $\mu_{t-1}$  and the control  $u_t$ .<sup>7</sup> When the time horizon is reached, a terminal cost is incurred that is quadratic in just the mean vector  $\mu_T$ . In symbolic terms, the cost function  $\Psi^\mu$  can be written as

$$(4.25) \quad \Psi^\mu = \sum_{\tau=1}^T \psi^\mu(\mu_{\tau-1}, u_\tau) + \mu_T^\top W_{\mu,T} \mu_T,$$

where  $\psi^\mu(\mu, u) = \mu^\top W_\mu \mu + 2u^\top W_{\mu,u} \mu + u^\top W_u u$  is the per-time-step cost. So that there is a well-defined, unique minimum, the combined cost matrix

$$\begin{pmatrix} W_\mu & W_{\mu,u}^\top \\ W_{\mu,u} & W_u \end{pmatrix}$$

and the terminal cost matrix  $W_{\mu,T}$  must both be symmetric positive semidefinite, and the action cost matrix  $W_u$  must be symmetric positive definite.

<sup>7</sup>The time indices of  $\mu_{t-1}$  and  $u_t$  differ by one because the initial mean vector is  $\mu_0$  but the initial action is  $u_1$ .

To minimize this function, at each given time step  $t$ , after observing a sequence  $h_{t-1}$  of actions and observations through time  $t - 1$ , a sequence of actions  $u_t^*(h_{t-1})$ ,  $\dots$ ,  $u_T^*(h_{t-1})$  must be selected that minimizes  $E[\Psi^\mu | h_{t-1}]$ . However, this can be cast as a dynamic programming problem, so that  $u_t^*(h_{t-1})$  can be selected without explicitly computing actions further in the future. In fact, the optimal action can be computed as a linear function of the mean vector  $\mu_{t-1}$ .

**Lemma 4.3.** *The optimal action  $u_t^*(h_{t-1})$  to be taken at time  $t$  after observing history  $h_{t-1}$  is the certainty-equivalent optimal action and a linear function of the mean vector  $\mu_{t-1}$ .*

The certainty-equivalent optimal action is the action that would be optimal if the system were deterministic (i.e.  $\eta_{t+1} \equiv \mathbf{0} \forall t$ ). Lemma 4.3 is proven in Appendix A.3. As can be seen in the proof, the coefficients of this linear function are computed recursively, but without needing to refer to previous actions and observations. Therefore, the linear function for each time step can be determined a priori, allowing very fast action selection while controlling the system.

Theorem 4.2 states that any LDS has an equivalent PLG of the same dimension. It is also the case that for any *full-rank* linear-quadratic Gaussian model,<sup>8</sup> there is an equivalent PLGQ. The linear-quadratic Gaussian model is based on the LDS model and a quadratic cost function, and is used in a wide variety of control applications, including chemical plant control, aircraft autopilot control, and vibration cancellation. A PLGQ and linear-quadratic Gaussian model are equivalent if each model selects the same optimal action as the other after any sequence of actions and observations is observed. The optimal expected cost-to-go computed by the two models

---

<sup>8</sup>A full-rank linear-quadratic Gaussian model is one whose underlying LDS is full rank. This is discussed in further detail in Section 2.2.2, but it essentially means that there is a one-to-one correspondence between values of the LDS state and values of the state of the equivalent PLG, or in other words, that no two states induce an identical distribution over future observations.

will differ only by a constant that is independent of the history.

**Theorem 4.4.** *For any  $n$ -dimensional, full-rank linear-quadratic Gaussian model, an equivalent  $n$ -dimensional PLGQ exists that, given any history of interaction with the system, computes the same optimal action.*

In other words, the PLGQ can be used to specify and solve the same control problems as full-rank linear-quadratic Gaussian models. The proof of this theorem is given in Appendix A.4.

## CHAPTER V

# Estimating the Parameters of the PLG

Having introduced the PLG in the previous chapter, there is a natural question of how to build PLGs from data. In this chapter, I present a consistent parameter estimation algorithm; that is, it produces estimates that tend toward the true values of the parameters as the amount of data increases. I then compare this algorithm to existing parameter estimation algorithms for LDS models, both theoretically and experimentally. The experimental results presented here result from running the algorithms on data sets generated by random LDS models.

### 5.1 The Consistent Estimation Algorithm

The CE algorithm is based in large part on the direct connection of the PLG parameters with statistical patterns in the data. For instance,  $G$  is the linear trend in the mean of the observations. This is in contrast to the LDS, whose parameters have a less direct connection because of its latent variables. In the LDS, the same linear trend is a function of two model parameters:  $H$ , the mapping from hidden state to observations, and  $A$ , the next-state matrix.<sup>1</sup> Because of the direct connection PLG parameters have to the observations, CE is a *non-iterative* algorithm that consists of simple operations like linear regression and taking sample means and covariances.

---

<sup>1</sup>See Section 2.2.2 for an introduction to linear dynamical systems

CE estimates PLG parameters given a data set with  $K$  trajectories generated by a dynamical system; as  $K$  grows, the estimates converge to the true values of the parameters. This discussion will assume that a data set with  $K$  trajectories is given, each of which is a sequence of  $N$  actions and  $N$  observations. Each trajectory is started from the same (unknown) initial state  $\mu_0, \Sigma_0$ .<sup>2</sup> The  $t$ th observation from the  $k$ th trajectory is denoted  $y_t^k$ ; similarly,  $z_t^k$  is the observed value of  $Z_t$  from the  $k$ th trajectory.<sup>3</sup> In addition, the notation  $\bar{y}_t$  will denote the average of  $y_t^k$  taken over all the trajectories (similarly  $\bar{z}_t, \bar{\eta}_{t+1}$ , etc.), and  $\hat{G}$  refers to an estimate of  $G$  (similarly  $\hat{C}_\eta, \hat{\eta}_{t+1}^k$ , etc.). The presentation of the algorithm assumes that  $n$  is given and that  $Z_t$  is prepopulated (i.e. that it is known which observational element corresponds to each element of  $Z_t$ ).

The algorithm will be presented in three parts: estimating the linear trend parameters, estimating the initial state, and estimating the noise parameters.

### 5.1.1 Linear Trends

Three PLG parameters represent linear trends present in the data.  $G$  is the linear trend of the observations (specifically  $Z_t$ ) in the absence of actions,  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  are the linear effects of the actions, and  $J$  is the linear function from  $Z_t$  to  $Y_{t+1}$ . CE estimates each of these parameters using linear regression.

$G$  and  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  are estimated together. Because of the construction of the model, some of the rows of these matrices are known a priori. Some rows of  $G$  copy an observation element from  $i+1$  steps in the future onto the same element  $i$  steps in the future. For instance, if  $Z_t^1 = Y_{t+1}^1$  and  $Z_t^4 = Y_{t+2}^1$  (as in Figure 4.2), the first row

---

<sup>2</sup>In LDS terms, this is equivalent to requiring that the initial latent state  $X_1$  is drawn from a Gaussian distribution, *not* that it is set to a particular initial value. In other words, this puts a prior on the initial observations.

<sup>3</sup>Though this is the same notation used to indicate a specific element of  $y_t$  or  $z_t$ , the superscript  $k$  will always denote the entire vector, taken from the  $k$ th trajectory.

of  $G$  simply copies the fourth element of  $Z_t$  into the first—that is, it is the same as the fourth row of the  $n \times n$  identity matrix. Additionally, as discussed in Section 4.2, the  $i$ th row of  $\Gamma_j$  (for each  $j$ ) is fixed to be  $\mathbf{0}$  when  $Z_t^i$  is not the first appearance of an observational element. Note that all of the rows that are predetermined in  $G$  are also predetermined in  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$ . We refer to the rows that are *not* predetermined in this way as  $G^{\text{unk}}, \Gamma_1^{\text{unk}}, \dots, \Gamma_{\tau_{\max}}^{\text{unk}}$ . Likewise, let  $z_t^{\text{unk},k}$  be the sub-vector of  $z_t^k$  that corresponds to these rows.

However, not all the elements of  $\Gamma_1^{\text{unk}}, \dots, \Gamma_{\tau_{\max}}^{\text{unk}}$  are free parameters. Recall from the discussion after (4.15) that, in order to preserve causality, actions should not affect observations that will occur before the action is taken. So if  $Z_t^2 = Y_{t+1}^2$  (and no other entry of  $Z_t$  refers to  $Y_{t+2}^2$ ), the second row of  $\Gamma_1$  is unrestricted, but the second row of  $\Gamma_2, \Gamma_3, \dots$  must be  $\mathbf{0}$ , because actions taken after  $Y_{t+1}$  should not affect its probability distribution.

From the PLG's core dynamics (4.15), it follows that

$$(5.1) \quad z_{t+1}^{\text{unk},k} = G^{\text{unk}} z_t^k + \sum_{i=1}^{\tau_{\max}} \Gamma_i^{\text{unk}} u_{t+i}^k + \eta_{t+1}^{k,\text{unk}}$$

for all  $t$  and  $k$ . This equality holds when averaging across all trajectories in the data set:

$$(5.2) \quad \bar{z}_{t+1}^{\text{unk}} = G^{\text{unk}} \bar{z}_t + \sum_{i=1}^{\tau_{\max}} \Gamma_i^{\text{unk}} \bar{u}_{t+i} + \bar{\eta}_{t+1}^{\text{unk}}$$

for all  $t$ . As  $K$  grows large, the Weak Law of Large Numbers applies and it can be shown that  $\bar{\eta}_{t+1}^{\text{unk}} \xrightarrow{p} \mathbf{0}$  as  $K \rightarrow \infty$ , where  $\xrightarrow{p}$  denotes convergence in probability.<sup>4</sup>

This implies that

$$(5.3) \quad \bar{z}_{t+1}^{\text{unk}} \xrightarrow{p} \gamma \xi_t$$

---

<sup>4</sup>Please refer to Appendix B.1 for a short discussion on large sample theory and convergence in probability.

for all  $t$ , where

$$(5.4) \quad \gamma = \begin{pmatrix} G^{\text{unk}} & \Gamma_1^{\text{unk}} & \dots & \Gamma_{\tau_{\max}}^{\text{unk}} \end{pmatrix}, \quad \xi_t = \begin{pmatrix} \bar{z}_t \\ \bar{u}_{t+1} \\ \bar{u}_{t+2} \\ \vdots \\ \bar{u}_{t+\tau_{\max}} \end{pmatrix}.$$

(5.3) can be rewritten in matrix form as  $Z^{\text{unk}} = \gamma \Xi$ , where the  $t$ th column of  $\Xi$  is  $\xi_{t-1}$  and the  $t$ th column of  $Z^{\text{unk}}$  is  $\bar{z}_t^{\text{unk}}$ ; both of these matrices have  $N - \tau_{\max}$  columns. This is a straightforward linear regression problem; an estimate for these linear trends results from right-multiplying both sides by  $\Xi^\top (\Xi \Xi^\top)^{-1}$ .

The estimates of  $\Gamma_1^{\text{unk}}, \dots, \Gamma_{\tau_{\max}}^{\text{unk}}$  and  $G^{\text{unk}}$  provided by  $\hat{\gamma}$  can be combined with the rows that were predetermined by the structure of the model to obtain  $\hat{\Gamma}_1, \dots, \hat{\Gamma}_{\tau_{\max}}$  and  $\hat{G}$ , the CE algorithm's estimates of  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  and  $G$ .

Note that (5.34) is a slight simplification of the actual regression. In fact, each  $\Gamma_i$  has different predetermined rows. For instance, the  $i$ th row of  $\Gamma_2, \dots, \Gamma_{\tau_{\max}}$  is  $\mathbf{0}$  whenever  $Z_t^i$  is an element of  $Y_{t+1}$ , while the  $i$ th row of  $\Gamma_1$  need not be. Therefore, each row (or block of rows) of  $\hat{\gamma}$  must be estimated separately, with the columns of  $\Xi$  containing a different number of actions for each regression.

It should be noted that  $\hat{\gamma}$  is generally a *biased* estimate; while it is true that  $\bar{\eta}_{t+1} \xrightarrow{p} \mathbf{0}$  as  $K$  grows, it is also true that when  $K$  is small  $\bar{\eta}_t$  is not only non-zero, but it will in general be correlated with  $\bar{\eta}_r$  when  $r \neq t$ . However, it is a consistent estimate, and so it should be expected that this bias disappears as the data set grows.

The next-observation function  $J$  can be estimated in much the same way. As with  $\hat{G}, \hat{\Gamma}_1, \dots, \hat{\Gamma}_{\tau_{\max}}$ , some elements of  $\hat{J}$  are predetermined by the structure of the model. First, if the  $j$ th element of  $Z_t$  is not an element of  $Y_{t+1}$ , the  $j$ th column of  $J$  is fixed at

$\mathbf{0}$  (because each element of the next observation is taken as a linear function only of elements of the next observation that are part of  $Z_t$ ). Additionally, if  $Z_t^j = Y_{t+1}^i$ , then the  $i$ th row of  $J$  is  $e_j^\top$  (i.e. the  $j$ th row of the  $n \times n$  identity matrix). If all elements of  $Y_{t+1}$  are found in  $Z_t$ , then  $\hat{J} = J$  is completely predetermined. Otherwise, the submatrix  $J^{\text{unk}}$  that is *not* predetermined must be estimated, again through linear regression. Let  $Z_t^{\text{next}}$  be the sub-vector of  $Z_t$  whose elements correspond to elements of  $Y_{t+1}$ . Then

$$(5.5) \quad y_{t+1}^k = J z_t^{\text{next},k}$$

for all  $t$  and  $k$ .<sup>5</sup> Again, both sides of these equations can be averaged over all  $K$  trajectories to obtain

$$(5.6) \quad \bar{y}_{t+1} = J \bar{z}_t^{\text{next}}$$

and then collected together into a single matrix equation

$$(5.7) \quad \Upsilon = J Z_{\text{next}},$$

where the  $t$ th column of  $\Upsilon$  is  $\bar{y}_t$  and the  $t$ th column of  $Z_{\text{next}}$  is  $\bar{z}_{t-1}^{\text{next}}$ . As in the previous regression, both of these matrices have  $N - \tau_{\text{max}}$  columns. An estimate of  $J$  is then obtained by

$$(5.8) \quad \hat{J}^{\text{unk}} = \Upsilon (Z_{\text{next}})^\top (Z_{\text{next}} (Z_{\text{next}})^\top)^{-1}.$$

This submatrix can be combined with those elements of  $J$  that are predetermined by the structure to obtain  $\hat{J}$ , the CE estimate of  $J$ .

While (5.5) only holds in the case of the standard PLG,  $\hat{J}$  is also a consistent estimate for  $J$  in the variance-adjusted PLG. In the variance-adjusted PLG,  $E[Y_{t+1}] =$

---

<sup>5</sup>This is true for the standard PLG. In the variance-adjusted PLG, this equation is true in expectation.

$J\mathbb{E}[Z_t^{\text{next}}]$ . As  $K \rightarrow \infty$ , the left side of (5.6) converges in probability on  $\mathbb{E}[Y_{t+1}]$  and the right side converges to  $J\mathbb{E}[Z_t^{\text{next}}]$ . Thus  $\widehat{J} \xrightarrow{p} J$  as  $K \rightarrow \infty$ .

### 5.1.2 Initial State

At first blush, it may seem that  $\mu_0$  and  $\Sigma_0$  could be estimated through a simple sample mean and covariance of  $z_0^k$ . However, this fails to account for the actions; the initial state is the distribution of  $Z_0$  given that  $u_1 = \mathbf{0}, u_2 = \mathbf{0}$ , etc. This can be remedied by noting that effect of each action  $u_1, u_2, \dots, u_{\tau_{\max}}$  on  $Z_0$  is additive and a function of only  $G, \Gamma_1, \dots, \Gamma_{\tau_{\max}}$  and the action itself (and that the effect of actions further in the future is zero). By using the *estimates*  $\widehat{G}, \widehat{\Gamma}_1, \dots, \widehat{\Gamma}_{\tau_{\max}}$ , to compute this effect, an estimate of  $Z_0|u_1 = \mathbf{0}, \dots$  can be obtained for each trajectory, and the sample mean and covariance of *these estimates* can be taken to estimate  $\mu_0$  and  $\Sigma_0$ .

The issue, then, is determining the effect of the actions on  $Z_0$ . To derive this effect, consider two trajectories,  $k_1$  and  $k_2$ . Suppose that  $\eta_1^k = \eta_2^k$  and  $u_t^{k_1} = u_t^{k_2} = \mathbf{0}$  at every time step  $t$ . Recall the core PLG dynamics from (4.15):

$$(5.9) \quad Z_{t+1} = GZ_t + \sum_{i=1}^{\tau_{\max}} \Gamma_i u_{t+i} + \eta_{t+1}$$

Then  $Z_{t+1}^{k_1} - Z_{t+1}^{k_2} = \mathbf{0}$  at every  $t$ . That is, with identical noise terms and identical actions, both trajectories are identical. To determine the effects of actions of  $Z_0$ , it is only necessary to see how trajectory  $k_1$  becomes different from  $k_2$  when its actions are changed.

To that end, fix a particular  $t$ , and change  $u_{t+\tau_{\max}}^{k_1}$  to  $u_{\tau_{\max}}$ . Inspection of (5.9) reveals that the change in  $Z_t^{k_1}$  can be quantified by

$$(5.10) \quad Z_{t+1}^{k_1} - Z_{t+1}^{k_2} = \Gamma_{\tau_{\max}} u_{\tau_{\max}}.$$

Now change another action; this time, set  $u_{t+\tau_{\max}-1}^{k_1}$  to  $u_{\tau_{\max}-1}$ . This changes *two* terms in (5.9). As before, it affects  $Z_{t+1}$  through the summation. But it also changes

$Z_t$  through *its* summation over  $\Gamma_i u_{t-1+i}$ . So now, with the changes in bold,

$$(5.11) \quad Z_{t+1}^{k_1} - Z_{t+1}^{k_2} = \Gamma_{\tau_{\max}} u_{\tau_{\max}} + (\mathbf{\Gamma}_{\tau_{\max}-1} + \mathbf{G}\mathbf{\Gamma}_{\tau_{\max}}) \mathbf{u}_{\tau_{\max}-1}.$$

Performing similar substitutions for  $u_{t+1}^{k_1}$ , etc., the resulting difference is

$$(5.12) \quad Z_{t+1}^{k_1} - Z_{t+1}^{k_2} = \sum_{i=1}^{\tau_{\max}} \sum_{j=i}^{\tau_{\max}} (G^{j-i} \Gamma_j) u_i.$$

If it is assumed that  $Z_0$  behaves as  $Z_t$  for any other  $t$  (or equivalently, that (5.9) holds for  $t < 0$ ), then for each trajectory, a sample of  $Z_0 | u_1 = \mathbf{0}, u_2 = \mathbf{0}, \dots$  can be derived from the sample of  $Z_0$  given the actions actually taken in trajectory  $k$ :

$$(5.13) \quad \hat{z}_{0|0}^k = z_0^k - \sum_{i=0}^{\tau_{\max}-1} \sum_{j=i}^{\tau_{\max}-1} (\hat{G}^{j-i} \hat{\Gamma}_j) u_i^k,$$

where it can be assumed that  $u_0^k \triangleq \mathbf{0}$ .

Given these estimates, the initial state can be estimated through a sample mean,

$$(5.14) \quad \hat{\mu}_0 = \frac{1}{K} \sum_{k=1}^K \hat{z}_{0|0}^k,$$

and a sample covariance,

$$(5.15) \quad \hat{\Sigma}_0 = \frac{1}{K-1} \sum_{k=1}^K (\hat{z}_{0|0}^k - \hat{\mu}_0)(\hat{z}_{0|0}^k - \hat{\mu}_0)^\top.$$

These are consistent estimates of the initial state, though the proof is deferred for the moment.

### 5.1.3 Noise Parameters

The only parameters left to estimate are those that govern the stochastic element of the system—that is, the variance of  $\eta_{t+1} | h_t$  ( $\Sigma_\eta$ ) and its covariance with  $Z_t$  ( $C_\eta$ ), and, in the case of variance-adjusted PLGs, the variance adjustment ( $\Sigma_{\text{adj}}$ ).

The estimation of  $C_\eta$  and  $\Sigma_\eta$  is in a way analogous to the estimation of  $\mu_0$  and  $\Sigma_0$ . The latter pair are the parameters of the distribution of  $Z_0 | u_1 = \mathbf{0}, u_2 = \mathbf{0}, \dots$ ,

samples of which can be derived from the data as in (5.13). Similarly,  $C_\eta$  and  $\Sigma_\eta$  are parameters of the distribution of  $\eta_{t+1}|h_t$ , samples of which can likewise be computed from the data.

Solving the dynamics equation (5.9) for the noise term and substituting estimated parameters yields an estimate for  $\eta_{t+1}$ :

$$(5.16) \quad \widehat{\eta}_{t+1}^k = z_{t+1}^k - \widehat{G}z_t^k - \sum_{i=1}^{\tau_{\max}} \widehat{\Gamma}_i u_{t+i}^k$$

for all  $t$  and  $k$ .  $\eta_{t+1}$  and  $\eta_{t+1}|h_t$  have the same mean, variance, and covariance with  $Z_t$  (which is shown in the proof of Theorem 5.1; see Appendix A.5). Notice that  $z_t^k$ ,  $z_{t+1}^k$ , and  $u_{t+i}^k$  come directly from (averages over) the data set, and not (for example) state estimates derived from the parameters. CE is non-iterative, and does not contain recursions of that type.

A sample variance and sample covariance of these noise estimates provide consistent estimates for  $\Sigma_\eta$  and  $C_\eta$ . The CE estimate for  $\Sigma_\eta$  is given by the sample variance of  $\widehat{\eta}_{t+1}^k$ :

$$(5.17) \quad \widehat{\Sigma}_\eta = \frac{1}{N - \tau_{\max}} \sum_{t=1}^{N - \tau_{\max}} \frac{1}{K - 1} \sum_{k=1}^K \widehat{\eta}_t^k \widehat{\eta}_t^{k\top}.$$

Similarly, the estimate for  $C_\eta$  is given by the sample covariance of  $\widehat{\eta}_{t+1}^k$  with  $z_t^k$ :

$$(5.18) \quad \widehat{C}_\eta = \frac{1}{N - \tau_{\max}} \sum_{t=0}^{N - \tau_{\max} - 1} \frac{1}{K - 1} \sum_{k=1}^K (z_t^k - \widetilde{Z}_t^k) \widehat{\eta}_{t+1}^{k\top},$$

where

$$(5.19) \quad \widetilde{Z}_t^k = \begin{cases} \widehat{\mu}_0 + \sum_{t=0}^{\tau_{\max}-1} \sum_{j=t}^{\tau_{\max}-1} (\widehat{G}^{j-t} \widehat{\Gamma}_j) u_t^k & t = 0 \\ \widehat{G} \widetilde{Z}_{t-1}^k + \sum_{i=1}^{\tau_{\max}} \widehat{\Gamma}_i u_{t+i}^k & t > 0 \end{cases}$$

is an estimate of  $E[Z_t^k | u_1^k, u_2^k, \dots]$  (setting  $\Gamma_0 = \mathbf{0}$  and  $u_0^k = \mathbf{0}$ ).

There remains only one parameter to estimate—the variance adjustment parameter,  $\Sigma_{\text{adj}}$ . In the standard PLG, this is fixed at  $\Sigma_{\text{adj}} \equiv \mathbf{0}$ ; in the variance-adjusted PLG, it affects the variance of the next observation according to the equation

$$(5.20) \quad \text{Var}[Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots] = J\Sigma_t J^\top + \Sigma_{\text{adj}}.$$

$\text{Var}[Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots]$  is difficult to estimate from the data because each history occurs only once. However,  $\text{Var}[Y_{t+1}|u_{t+1} = \mathbf{0}, \dots]$  (the variance *not* conditioned on history) can be estimated readily by a sample variance. To find the relationship between this quantity and  $\Sigma_{\text{adj}}$ , the Law of Total Variance<sup>6</sup> may be applied:

$$(5.21) \quad \text{Var}[Y_{t+1}|u_{t+1} = \mathbf{0}, \dots] = \mathbb{E}_{h_t}[\text{Var}[Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots]]$$

$$(5.22) \quad \quad \quad + \text{Var}_{h_t}[\mathbb{E}[Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots]]$$

$$(5.23) \quad \quad \quad = J\Sigma_t J^\top + \Sigma_{\text{adj}} + \text{Var}_{h_t}[J\mu_t|u_{t+1} = \mathbf{0}, \dots]$$

$$(5.24) \quad \quad \quad = J(\Sigma_t + \text{Var}_{h_t}[\mu_t|u_{t+1} = \mathbf{0}, \dots])J^\top + \Sigma_{\text{adj}},$$

where  $\mathbb{E}_{h_t}$  and  $\text{Var}_{h_t}$  denote expectation and variance, respectively, taken over the distribution of histories of length  $t$ .

Another application of the Law of Total Variance, this time to the variance of  $Z_t$ , obtains

$$(5.25) \quad \text{Var}[Z_t|u_{t+1} = \mathbf{0}, \dots] = \mathbb{E}_{h_t}[\text{Var}[Z_t|h_t, u_{t+1} = \mathbf{0}, \dots]]$$

$$(5.26) \quad \quad \quad + \text{Var}_{h_t}[\mathbb{E}[Z_t|h_t, u_{t+1} = \mathbf{0}, \dots]]$$

$$(5.27) \quad \quad \quad = \Sigma_t + \text{Var}_{h_t}[\mu_t|u_{t+1} = \mathbf{0}, \dots]$$

$$(5.28) \quad \quad \quad \therefore \Sigma_t = \text{Var}[Z_t|u_{t+1} = \mathbf{0}, \dots] - \text{Var}_{h_t}[\mu_t|u_{t+1} = \mathbf{0}, \dots].$$

Substituting (5.28) into (5.24) and solving for  $\Sigma_{\text{adj}}$  results in

$$(5.29) \quad \Sigma_{\text{adj}} = \text{Var}[Y_{t+1}|u_{t+1} = \mathbf{0}, \dots] - \text{Var}[JZ_t|u_{t+1} = \mathbf{0}, \dots].$$

---

<sup>6</sup>See Appendix B.2.

Both terms on the right-hand side of (5.29) can be estimated through sample variances, resulting in an estimate of  $\Sigma_{\text{adj}}$  for each  $t$ :

$$(5.30) \quad \widehat{\Sigma}_{\text{adj}}^t = \frac{1}{K-1} \sum_{k=1}^K \left( (y_{t+1}^k - \bar{y}_{t+1})(y_{t+1}^k - \bar{y}_{t+1})^\top - (\widehat{J}\widehat{z}_{t|0}^k - \widehat{J}\widehat{z}_{t|0})(\widehat{J}\widehat{z}_{t|0}^k - \widehat{J}\widehat{z}_{t|0})^\top \right),$$

where  $\widehat{z}_{t|0}^k$  follows from the logic of (5.13):

$$(5.31) \quad \widehat{z}_{t|0}^k = z_t^k - \sum_{i=1}^{\tau_{\max}} \sum_{j=i}^{\tau_{\max}} (\widehat{G}^{j-i} \widehat{\Gamma}_j) u_{t+i}^k.$$

In theory, for each  $t$ ,  $\widehat{\Sigma}_{\text{adj}}^t$  is a consistent estimator for  $\Sigma_{\text{adj}}$ , and so  $\widehat{\Sigma}_{\text{adj}} = \frac{1}{N-\tau_{\max}} \sum_{t=1}^{N-\tau_{\max}} \widehat{\Sigma}_{\text{adj}}^t$  may seem like a reasonable estimator. However, in practice, the behavior of  $\widehat{\Sigma}_{\text{adj}}^t$  for  $t > 0$  is not good, and so the estimator for  $\Sigma_{\text{adj}}$  in my experiments is given by

$$(5.32) \quad \widehat{\Sigma}_{\text{adj}} = \widehat{\Sigma}_{\text{adj}}^0.$$

## 5.2 Theoretical Results

The main theoretical result of this section is that CE produces consistent estimates for the parameters of a PLG as the number of trajectories in the data set grows. However, there are some technical requirements on the system and the policy used to generate the data set that must be satisfied in order for the consistency result to hold.

The major requirement is that the matrix  $\Xi\Xi^\top$  from (5.34) be invertible in the limit as  $K \rightarrow \infty$ . This is both a theoretical and a *practical* requirement. It is a theoretical requirement because consistency is a property of the behavior of estimators in the limit. It is a practical requirement because, when this requirement does *not* hold, conditioning problems will lead to poor estimates of  $G$  and  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  on

large data sets. Since most of the other estimates depend on the estimates for these parameters, this will lead to a poor model overall.

An equivalent way of stating this requirement is that a)  $E[\Xi]$  have full rank and b) the trajectories have sufficient length to make  $\Xi$  at least as wide as it is tall. The latter requirement means that  $N$  must be at least  $n + (l + 1)\tau_{\max}$ . The former is a joint constraint on the *system* and the *policy* used to generate the data set. In an uncontrolled system, this condition will not be satisfied if, for example,  $\mu_0 = \mathbf{0}$  or  $\mu_0$  is an eigenvector of  $G$ . This is because in an uncontrolled system,

$$(5.33) \quad \Xi = \begin{pmatrix} \mu_0 & G\mu_0 & G^2\mu_0 & \cdots \end{pmatrix}.$$

In a *controlled* system, the actions affect  $\bar{z}_t$ , and so a well-chosen policy can overcome difficulties such as  $\mu_0 = \mathbf{0}$ . However, a new issue is raised: The actions form part of  $\Xi$ , so a random policy where  $E[u_1] = E[u_2] = \cdots = E[u_N]$  will lead to an under-ranked matrix. An exploration policy that satisfies the requirement that  $E[\Xi]$  have full rank for a particular system is called a *CE-learnable* policy for the system.

One policy that satisfies at least the requirement that the rows of  $E[\Xi]$  corresponding to actions be linearly independent is a periodic policy of the following form. Suppose that each action is drawn from a multivariate Gaussian distribution. For most actions, the mean of this distribution is  $\mathbf{0}$ . But when  $t = i\tau_{\max}$  for integer  $i$ , let a single element of  $u_t$  have a non-zero mean. The element that has non-zero mean will depend on  $i$ ; it will be the first when  $i \equiv 1 \pmod{l}$ , the second when  $i \equiv 2 \pmod{l}$ , etc. Thus the sequence of action means will repeat every  $l\tau_{\max}$  time steps. The reasoning behind this policy can be seen by inspecting the  $l\tau_{\max} \times N - \tau_{\max}$  submatrix of  $E[\Xi]$  corresponding to the actions. Each column of this matrix will be a unit vector with a single entry equal to 1. Over the course of  $l\tau_{\max}$  columns, each unit vector of this type will be present, thus guaranteeing that the rows of  $E[\Xi]$

corresponding to actions will be linearly independent. This policy is used in the experiments in Section 5.5.

The formal statement of the consistency result is made by the following theorem.

**Theorem 5.1.** *If a dynamical system can be modeled by an  $n$ -dimensional PLG, is controlled by a policy that is jointly CE-learnable with the system, and generates a training set whose  $K$  trajectories are each at least  $n+(l+1)\tau_{\max}$  time steps long, then, as the number of trajectories  $K$  grows, the parameter estimates computed by the CE algorithm from this training set will converge in probability to the true parameters of that PLG.*

A sketch of the proof of Theorem 5.1 has already been given in the development of CE. A formal proof is given in Appendix A.5.

### 5.3 A Weighted Consistent Estimation Algorithm

The CE algorithm described above gives equal importance to data from each time step in the estimators for  $G$ ,  $J$ ,  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$ ,  $C_\eta$ ,  $\Sigma_\eta$ , and  $\Sigma_{\text{adj}}$ . There are a number of reasons this may be considered undesirable. For instance, the standard linear regression for  $\hat{G}$  may put too much weight on the tail end of the trajectories; the variance of  $\bar{z}_t$  will usually increase as  $t$  grows larger, and so an estimate that gives equal weight to samples with large  $t$  will have higher variance than one that places more weight on earlier samples. Or, as in the next chapter, different trajectories may have different lengths, and so each time step may have a different number of samples.

Thus, a *weighted* version of CE may present attractive alternative. In this algorithm, each time step  $t$  is given a weight  $w_t$ . Choosing the weights is a matter of some importance as it will bear on the variance of the estimator. An example weighting scheme that places extreme importance on data points early in each trajectory is

$w_t = \frac{1}{t+1}$ . A weighting scheme used in the next chapter sets  $w_t$  to the number of trajectories with a length of  $t$  or more. Exploring different weight schemes is an area of future work.

Several estimators are changed from their non-weighted versions—in fact, all of them except for the initial state estimators are.

### 5.3.1 Linear Trend Parameters

Let  $W$  be a diagonal matrix whose  $(t+1, t+1)$ th element  $w_{t+1}$  is the weight to be placed on the squared error in (5.3). Then

$$(5.34) \quad \hat{\gamma}^w = Z^{\text{unk}} W \Xi^\top (\Xi W \Xi^\top)^{-1}$$

is a consistent estimate for  $\gamma$ . Recall that

$$(5.35) \quad \gamma = \begin{pmatrix} G^{\text{unk}} & \Gamma_1^{\text{unk}} & \dots & \Gamma_{\tau_{\max}}^{\text{unk}} \end{pmatrix}.$$

$\hat{\Gamma}_1^w, \dots, \hat{\Gamma}_{\tau_{\max}}^w$  and  $\hat{G}^w$  are computed by combining the appropriate submatrices of  $\hat{\gamma}^w$  with the elements of  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  and  $G$  that are predetermined by the structure of the model, in the same way that  $\hat{\Gamma}_1, \dots, \hat{\Gamma}_{\tau_{\max}}$  and  $\hat{G}$  were computed from  $\hat{\gamma}$ .

The non-predetermined elements of the next-observation matrix  $J$  can be estimated with the following weighted regression:

$$(5.36) \quad \hat{J}^{w, \text{unk}} = \Upsilon W (Z_{\text{next}})^\top (Z_{\text{next}} W (Z_{\text{next}})^\top)^{-1}.$$

When combined with the predetermined elements in the same manner as  $\hat{J}^{\text{unk}}$ , this produces  $\hat{J}^w$ , a consistent estimate for  $J$ .

### 5.3.2 Noise Parameters

The weighted version of each noise parameter's estimator is a straightforward weighted average of each time step's estimator. The estimator for  $\Sigma_\eta$  is given by

$$(5.37) \quad \widehat{\Sigma}_\eta^w = \frac{1}{\sum_{t=1}^{N-\tau_{\max}} w_t} \sum_{t=1}^{N-\tau_{\max}} \frac{w_t}{K-1} \sum_{k=1}^K \widehat{\eta}_t^k \widehat{\eta}_t^{k\top}.$$

Similarly, the weighted estimate of the covariance between the noise and  $Z_t$  is

$$(5.38) \quad \widehat{C}_\eta^w = \frac{1}{\sum_{t=1}^{N-\tau_{\max}} w_t} \sum_{t=0}^{N-\tau_{\max}-1} \frac{w_{t+1}}{K-1} \sum_{k=1}^K (z_t^k - \widetilde{Z}_t^k) \widehat{\eta}_{t+1}^{k\top}.$$

Finally, the weighted estimate for  $\Sigma_{\text{adj}}$ , the additional parameter for the variance-adjusted PLG, is given by

$$(5.39) \quad \widehat{\Sigma}_{\text{adj}}^w = \frac{1}{\sum_{t=1}^{N-\tau_{\max}} w_t} \sum_{t=1}^{N-\tau_{\max}} w_t \widehat{\Sigma}_{\text{adj}}^t.$$

Theorem 5.1 holds for the weighted version of CE under the same conditions as long as all the weights are positive.

## 5.4 Comparing CE to LDS Methods of Parameter Estimation

Because the PLG subsumes the LDS and there are many well-established methods to estimate the parameters of an LDS, it is sensible to compare these methods to the estimation algorithms presented in this chapter. There are two (families of) LDS methods of particular interest: subspace identification methods and expectation maximization.

### 5.4.1 CE vs. Subspace Methods

Subspace-based state-space system identification (4SID) methods are parameter estimation algorithms for LDSs (Overschee & De Moor, 1996; Viberg, 1995). Like CE, 4SID methods are non-iterative—in particular, they do not perform local searches in the parameter space, as expectation maximization does.

Algorithms in the 4SID family can be quite complicated. Here I present a basic intuition.

The core dynamics of the LDS can be written as

$$(5.40) \quad \begin{pmatrix} X_{t+1} \\ Y_t \end{pmatrix} = \begin{pmatrix} A \\ H \end{pmatrix} X_t + \begin{pmatrix} \omega_{t+1} \\ \nu_t \end{pmatrix}.$$

This is quite similar in form to the PLG dynamics (4.15); if an estimate for  $X_t$  were obtained,  $A$  and  $H$  could be estimated through least-squares linear regression, much as  $G$  is in the CE algorithm. The main question answered by 4SID is how to estimate  $X_t$ . Define the output block Hankel matrices

$$(5.41) \quad Y_p \triangleq \begin{pmatrix} y_1 & \cdots & y_j \\ \vdots & \ddots & \vdots \\ y_i & \cdots & y_{i+j-1} \end{pmatrix}$$

and

$$(5.42) \quad Y_f \triangleq \begin{pmatrix} y_{i+1} & \cdots & y_{i+j} \\ \vdots & \ddots & \vdots \\ y_{2i} & \cdots & y_{2i+j-1} \end{pmatrix},$$

where  $f$  stands for future and  $p$  for past,  $i$  is at least  $n$ , and  $j$  is chosen so that the entire data set is included in  $Y_p$  and  $Y_f$  (4SID methods operate on a single trajectory).

Define

$$(5.43) \quad \widehat{\mathbf{X}}_{i+1} \triangleq \begin{pmatrix} \widehat{x}_{i+1} & \cdots & \widehat{x}_{i+j} \end{pmatrix}$$

as the estimates of the latent variables  $\mathbf{X}_{i+1} = [X_{i+1} \cdots X_{i+j}]$ . Finally, let

$$(5.44) \quad \mathcal{M}_i \triangleq \begin{pmatrix} H \\ HA \\ \vdots \\ HA^{i-1} \end{pmatrix}$$

be the extended observability matrix of the LDS.

From the dynamics of the LDS,  $E[Y_f | \mathbf{X}_{i+1} = \widehat{\mathbf{X}}_{i+1}] = \mathcal{M}_i \widehat{\mathbf{X}}_{i+1}$ . But it can also be shown that  $Y_f/Y_p \xrightarrow{p} \mathcal{M}_i \widehat{\mathbf{X}}_{i+1}$  under certain conditions as  $j \rightarrow \infty$ , where  $Y_f/Y_p = Y_f Y_p^\top (Y_p Y_p^\top)^\dagger Y_p$  is the projection of the row space of  $Y_f$  onto the row space of  $Y_p$  and  $\cdot^\dagger$  denotes the Moore-Penrose pseudo-inverse. This means that a consistent estimate of  $\mathcal{M}_i$  can be recovered through a singular value decomposition of  $Y_f/Y_p$  and thus that  $\widehat{\mathbf{X}}_{i+1}$  can be computed by

$$(5.45) \quad \widehat{\mathbf{X}}_{i+1} = \widehat{\mathcal{M}}_i^\dagger(Y_f/Y_p).$$

4SID methods vary in their details—for instance,  $\widehat{\mathbf{X}}_{i+1}$  may not be computed explicitly, and other changes may be made for the sake of efficiency or statistical guarantees. Some variants of 4SID are consistent under quite general conditions. One way some 4SID methods differ is by decomposing  $W_1(Y_f/Y_p)W_2$ , where  $W_1$  and  $W_2$  are weight matrices, instead of  $(Y_f/Y_p)$ . Different choices of these matrices can result in different properties of the algorithm; see Overschee and De Moor (1996). These weight matrices provided the inspiration for the weighted version of CE given in this chapter.

One fundamental difference between CE and 4SID methods is that 4SID methods are consistent as the length of a *single* trajectory grows, while CE is consistent as the *number* of trajectories grows; I have been unable to find an example of a 4SID method that will accept multiple trajectories as input. Some problems are naturally episodic, and so it is easier to produce a data set with a large number of trajectories instead of a single trajectory. For example, when trying to model the behavior of other drivers on the road, it makes more sense to observe many vehicles' behavior and treat each vehicle's trajectory separately, than to observe a single vehicle for a very long time. Of course, the opposite is also true: In some problems, it is easier or

more sensible to obtain a single long trajectory than many trajectories.

Smith, Freitas, Robinson, and Niranjani (1999) compare 4SID and EM in a speech modelling task. They report that

Subspace methods produce a much richer formant structure than EM and the plot shows how formants change and evolve during speech. This is despite the fact that EM produces smaller sum squared errors between true and reconstructed data in the time domain.

They also find that a hybrid method, wherein the initial parameters used in EM are produced by a 4SID algorithm, captures some of the best of both worlds, reducing error while capturing the formant structure of 4SID.

#### 5.4.2 CE vs. EM

Expectation maximization and the consistent estimation algorithm are two quite different ways to approach the parameter estimation problem. Whereas CE uses linear regression and sample means and covariances to produce a set of parameters that converges on the correct parameters as the data set increases in size, EM uses an iterative algorithm to find a local maximum in the expected value of the likelihood.

Recall from Section 2.2.2 that the dynamics of the LDS are described by (2.14)–(2.18), reprinted here:

$$(5.46) \quad X_{t+1} = AX_t + Bu_t + \omega_{t+1}$$

$$(5.47) \quad Y_t = HX_t + \nu_t$$

$$(5.48) \quad \text{Cov}[\omega_t, \omega_s] = \delta_{t,s}Q$$

$$(5.49) \quad \text{Cov}[\nu_t, \nu_s] = \delta_{t,s}R$$

$$(5.50) \quad \text{E}[\omega_t] = \mathbf{0}$$

$$(5.51) \quad \text{E}[\nu_t] = \mathbf{0}$$

$$(5.52) \quad X_1 \sim \mathcal{N}(x_1^-, P_1^-).$$

Based on these equations, the following densities hold for all  $t$ :

$$(5.53) \quad P(y_t|x_t) = e^{-\frac{1}{2}(y_t-Hx_t)^\top R^{-1}(y_t-Hx_t)}(2\pi)^{-m/2}|R|^{-1/2}$$

$$(5.54) \quad P(x_t|x_{t-1}) = e^{-\frac{1}{2}(x_t-Ax_{t-1}-Bu_{t-1})^\top Q^{-1}(x_t-Ax_{t-1}-Bu_{t-1})}(2\pi)^{-n/2}|Q|^{-1/2}$$

$$(5.55) \quad P(x|1) = e^{-\frac{1}{2}(x_1-x_1^-)^\top (P_1^-)^{-1}(x_1-x_1^-)}(2\pi)^{-n/2}|P_1^-|^{-1/2},$$

where  $n$  is the length of  $x_t$  and  $m$  is the length of  $y_t$ . Thus the joint log probability of a trajectory of length  $N$  is given by

$$(5.56) \quad \begin{aligned} \log P(x_1, y_1, \dots, x_N, y_N) = & \\ & - \sum_{t=1}^N \left( \frac{1}{2}(y_t - Hx_t)^\top R^{-1}(y_t - Hx_t) \right) - \frac{N}{2} \log|R| \\ & - \sum_{t=2}^N \left( \frac{1}{2}(x_t - Ax_{t-1} - Bu_{t-1})^\top Q^{-1}(x_t - Ax_{t-1} - Bu_{t-1}) \right) \\ & - \frac{T-1}{2} \log|Q| - \frac{1}{2}(x_1 - x_1^-)^\top (P_1^-)^{-1}(x_1 - x_1^-) - \frac{1}{2}|P_1^-| \\ & - \frac{T(m+n)}{2} \log 2\pi \end{aligned}$$

The EM algorithm maximizes the expected log likelihood, which is given by

$$(5.57) \quad \ell = \mathbb{E}[\log P(x_1, y_1, \dots, x_N, y_N)|y_1, y_2, \dots, y_N].$$

Unlike CE, EM is iterative. Each iteration has two steps: The E step, which computes three expectations required to compute  $\ell$ — $\mathbb{E}[x_t|y_1, y_2, \dots, y_N]$ ,  $\mathbb{E}[x_t x_t^\top|y_1, y_2, \dots, y_N]$ , and  $\mathbb{E}[x_t x_{t-1}^\top|y_1, y_2, \dots, y_N]$ —and the M step, which maximizes  $\ell$  by setting its derivative with respect to each of the parameters in turn to 0 and solving for that parameter (Ghahramani & Hinton, 1996). Each repetition of the E and M step increases  $\ell$ . Typical termination conditions cause the algorithm to terminate after a set number of iterations and/or when the increase of likelihood falls below a certain threshold. For example, in the experiments in this dissertation, EM terminates after the  $n$ th iter-

ation if  $n = 1000$  or  $\ell_n - \ell_0 < 1.0001(\ell_{n-1} - \ell_0)$ , where  $\ell_0$  is the likelihood of the initial set of parameters and  $\ell_n$  is the likelihood after the  $n$ th iteration.

The next section continues the comparison of EM and CE empirically.

## 5.5 Experimental Results on Random Systems

In my previously published works on PLGs, I have run experiments that compare the CE and EM algorithms on data sets produced by randomly generated LDSs.

### 5.5.1 Scalar Uncontrolled Systems

Rudary et al. (2005) generated LDSs with scalar observations according to the following recipe. For dimensions  $n = 2, 4, 8$ , each element of  $H$ ,  $A$ , and  $x_1^-$  was drawn from the uniform distribution  $U(-1, 1)$ . To avoid systems that diverge,  $A$  was then normalized so that its spectral radius would be a random value drawn from  $U(0, 1)$ .

The covariance matrix  $Q$  was computed by  $Q = \Lambda Q' \Lambda$ , where the random correlation matrix  $Q'$  was generated using the algorithm of Marshall and Olkin (1984), and the diagonal matrix  $\Lambda$ 's  $i$ th element was  $2^{x_i}$ , with  $x_i \sim U(-1, 1)$ . This resulted in  $Q$  having variances between  $\frac{1}{4}$  and 4 with random correlations.  $P_1^-$  was generated in the same manner as  $Q$ , and  $R = 2^{x_R}$ , with  $x_R \sim U(-1, 1)$ .

Using each model generated in this way, a data set was produced with 500 000 trajectories; each trajectory contained  $10n$  observations. PLGs and LDSs were then trained on several subsets of the data set using CE and EM, respectively. The value reported for each subset was the difference in log-likelihood per trajectory between the estimated parameters and the generating parameters. The version of CE used in these experiments was a specialization of that presented in this section that handled only uncontrolled scalar PLGs.

The results of these experiments are depicted in Figure 5.1, with a detail of Fig-

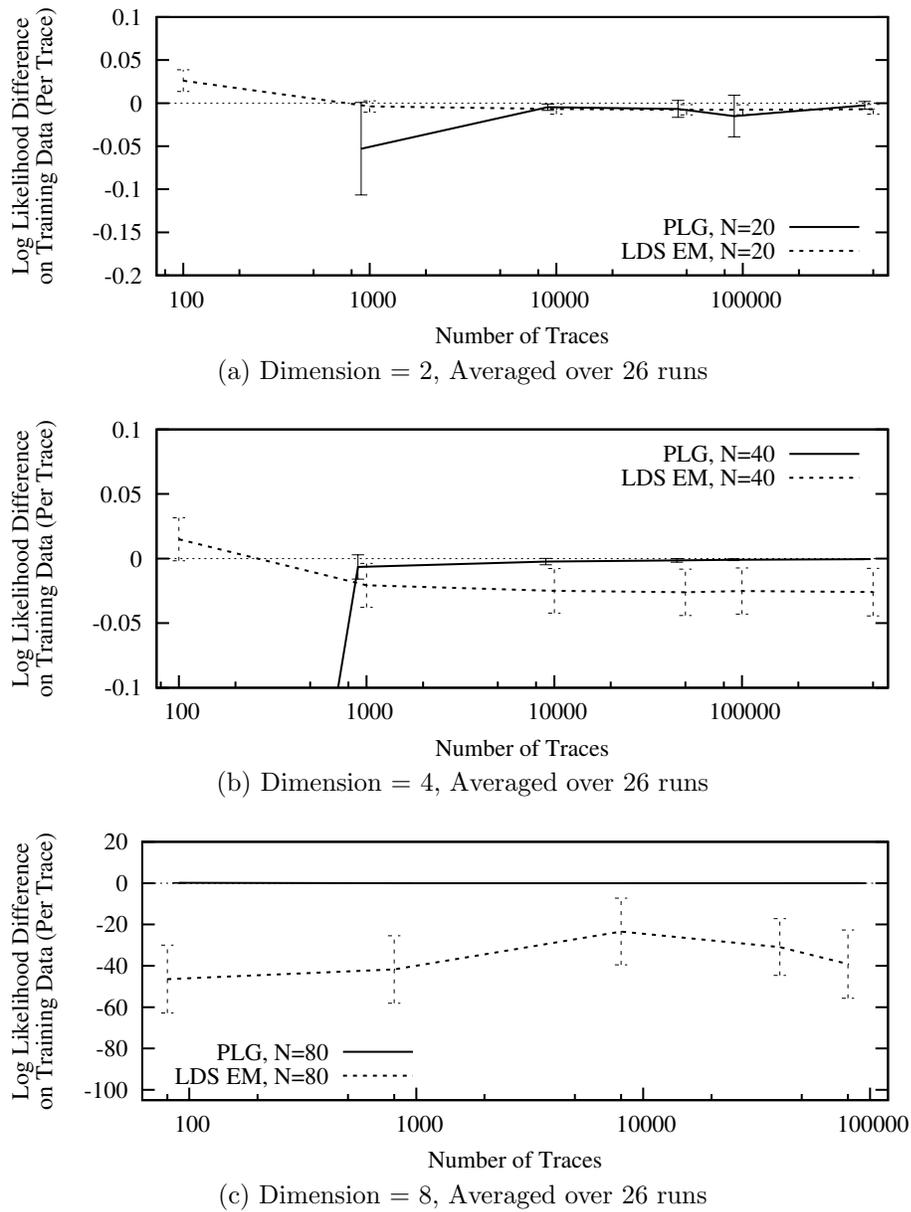


Figure 5.1: Comparison of EM and CE on scalar uncontrolled systems. The y-axis shows the difference in log-likelihood per trajectory between the actual and estimated parameters, over the data used to train the parameters. The x-axis shows the number of trajectories in the training set; the x-axis of each curve is slightly offset to improve readability. Higher values indicate higher likelihood in the learned parameters; at zero, the likelihood of the learned parameters is identical to that of the generating parameters. The error bars show the 95% confidence interval.

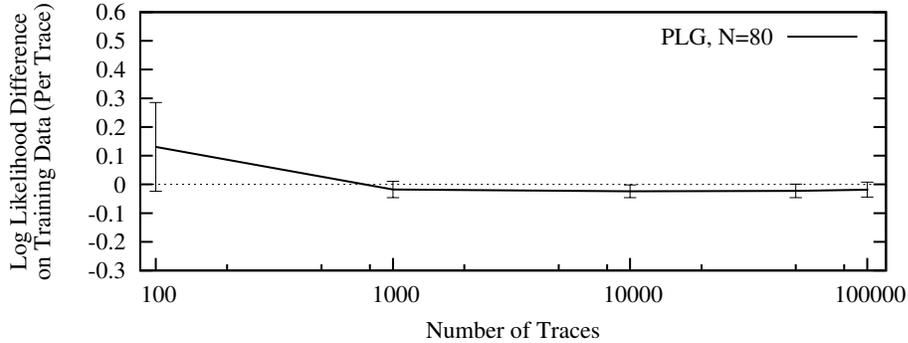


Figure 5.2: A zoomed-in look at the results of CE on scalar, uncontrolled systems of dimension 8.

Figure 5.1(c) shown in Figure 5.2. The per-trajectory difference in log-likelihood is given on the  $y$ -axis versus the number of trajectories in the training set on the  $x$ -axis. An error measure of 0 indicates that the estimated parameters have the same likelihood as the parameters used to generate the data set; higher values mean the estimated parameters have higher likelihood. Each plot shows results averaged over 26 test systems, with error bars showing the 95% confidence interval.

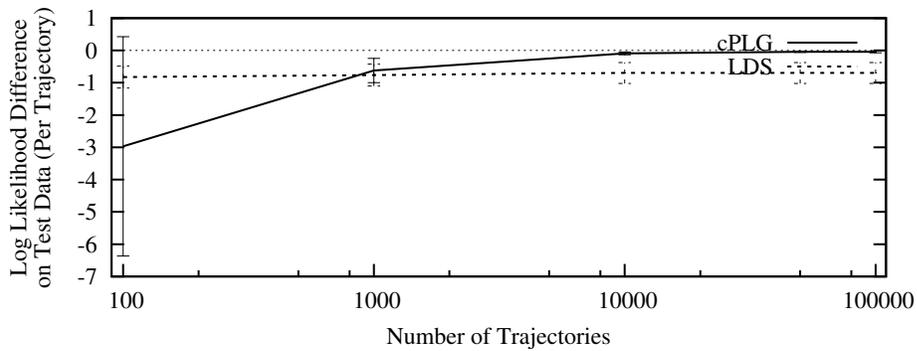
### 5.5.2 Scalar Controlled Systems

Rudary and Singh (2006) dealt with models with scalar observations and actions. They generated random LDSs as in the previous section, with the additional control matrix  $B$  being generated the same way as  $H$ . Again, the models had dimensions ( $n$ ) of 2, 4, and 8. They were used to create data sets with 100 000 trajectories, each of length  $5n$ . Again, PLGs and LDSs were trained on several subsets of each data set using CE and EM, respectively, reporting the difference in log-likelihood per trajectory between the estimated parameters and the generating parameters. The version of CE used in the experiments of Rudary and Singh (2006) was a specialization of the version presented here that estimated only controlled scalar PLGs.

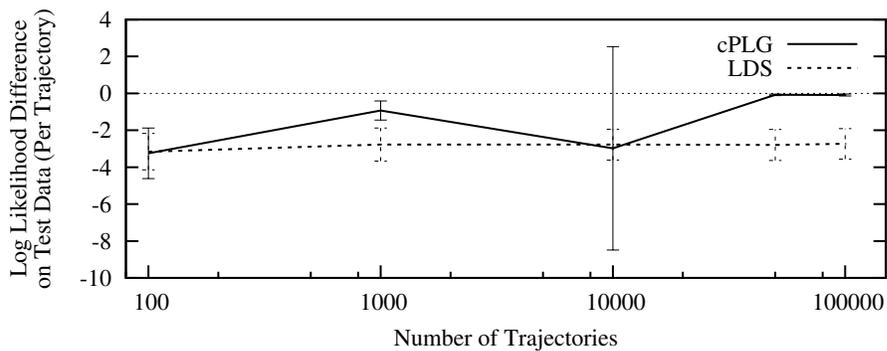
The results of these experiments are shown in Figure 5.3. As in Figure 5.1, the



(a) Dimension = 2, Averaged over 81 runs



(b) Dimension = 4, Averaged over 80 runs



(c) Dimension = 8, Averaged over 78 runs

Figure 5.3: Comparison of EM and CE on scalar controlled systems. The y-axis shows the difference in log-likelihood per trajectory between the actual and estimated parameters, over over a test set. The x-axis shows the number of trajectories in the training set; the x-axis of each curve is slightly offset to improve readability. Higher values indicate higher likelihood in the learned parameters; at zero, the likelihood of the learned parameters is identical to that of the generating parameters. The error bars show a 95% confidence interval.

$y$ -axis corresponds to the per-trajectory difference in log-likelihood and the  $x$ -axis to the number of trajectories in the training set. As before, increasing along the  $y$ -axis means that the estimated parameters have higher likelihood.

### 5.5.3 Vector-Valued Uncontrolled Systems

The random LDSs for these experiments on uncontrolled systems with vector-valued observations were generated in a slightly different manner than in the previous two sections. The elements of  $A$ ,  $H$ , and  $x_1^-$  were each drawn from the uniform distribution  $U(-1, 1)$ .  $A$  was then rescaled so that its spectral radius would be equal to a number drawn from  $U(0.4, 0.9)$ . In contrast to the previous experiments, in these experiments  $Q$  is diagonal with variances drawn from  $U(0, 2)$ . Finally,  $R$  and  $P_1^-$  were generated much like  $P_1^-$  and  $Q$  were in the previous experiments—a random correlation matrix was obtained using the algorithm described by Marshall and Olkin (1984), which was then adjusted so that each of the standard deviations was drawn from  $U(0, 2)$ .

As before, models of dimensions ( $n$ ) 2, 4, and 8 were generated; when  $n = 2$ , observations had length  $m = 2$ ; for the other two model sizes, the observations had length 3. A training set with 100 000 trajectories of length 75 and a test set with 10 000 trajectories of length 50 was generated using each random LDS. PLGs and LDSs were trained on several subsets of each training set using CE and EM, respectively, reporting the difference in log-likelihood *per observation* between the estimated parameters and the generating parameters for both the training sets and the test sets.

The results of these experiments are given in Figure 5.4, with a detail of Figure 5.4(c) shown in Figure 5.5. The per-observation difference in log-likelihood is given on the  $y$ -axis versus the number of trajectories in the training set on the  $x$ -

axis. Higher values on the  $y$ -axis indicate that the estimated parameters have higher likelihood. There are four curves on each plot; the dotted lines correspond to LDS results, while the solid lines show the results of CE. The curves with large solid points show the results on the test sets; the other two correspond to the training sets.

**Results** The results on scalar systems show CE outperforming EM on large data sets. Interestingly, EM appears to exhibit a leveling-off behavior; after a certain point, more data does not appear to improve the estimates. By contrast, the likelihood of CE’s parameters continues to increase as the training data grows. This despite the fact that CE does not explicitly attempt to maximize likelihood, while EM does. In the vector-valued experiments, this trend is again followed, though CE never clearly outperforms, and is sometimes outperformed by, EM. However, even when CE is outperformed by EM (e.g. Figure 5.4(b)), CE shows a clear improving trend, while EM’s results are flat.

#### 5.5.4 Vector-Valued Controlled Systems

The random LDSs for these experiments on controlled systems with vector-valued actions and observations were generated as in the previous section. In addition, each element of  $B$ , the control matrix, was drawn from the uniform distribution  $U(-1, 1)$ . Models of dimension ( $n$ ) 2, 4, and 8 were generated. In each case, observations had length  $m = 2$  and actions had length  $l = 3$ .

For each randomly generated model, a training set with 100 000 trajectories of length 75 and a test set with 10 000 trajectories of length 50 was generated using each random LDS. PLGs and LDSs were trained on several subsets of each training set using CE and EM, respectively, reporting the difference in log-likelihood *per observation* between the estimated parameters and the generating parameters for

both the training sets and the test sets.

The results of these experiments are given in Figure 5.6. The per-observation difference in log-likelihood is given on the  $y$ -axis versus the number of trajectories in the training set on the  $x$ -axis. Higher values on the  $y$ -axis indicate that the estimated parameters have higher likelihood. There are four curves on each plot; the dotted lines correspond to LDS results, while the solid lines show the results of CE. The curves with large solid points show the results on the test sets; the other two correspond to the training sets.

**Results** The results for this set of experiments were rather disappointing. CE performs rather poorly compared to EM on the test set for all data set sizes shown. On the other hand, it performs comparably to EM on all the training sets. I was unable to discover the reason for this discrepancy. It should be noted that Theorem 5.1 guarantees that with a large enough data set, the likelihood of the test set will match the likelihood of the generating parameters, but it is disappointing that 100 000 trajectories is not large enough.

#### 5.5.5 Comments on Experimental Results

On small data sets, EM usually outperforms CE by a wide margin. CE occasionally produces invalid parameter settings with small amounts of training data—when this occurs,  $\Sigma_t$  becomes non-positive semidefinite at some  $t$ , making the likelihood uncomputable unless this is compensated for. In Figure 5.1(a), no data point is shown for CE when  $K = 100$ ; this is because indefinite  $\Sigma_t$  was *not* worked around. Figures 5.3(a) and 5.3(c), on the other hand, show spikes in variance for CE at  $K = 10\,000$ ; that results from Rudary and Singh (2006) adding  $10I$  to non-definite  $\Sigma_t$ , which was necessary for one data set in each of the experiments corresponding

Model dimension	CE for PLGs	Each iteration of EM for LDSs
4	5.8s (0.10)	7.2s (0.12)
8	17.2s (0.16)	11.7s (0.18)

Table 5.1: Time taken by the CE algorithm and each iteration of EM. Times given are averaged over 10 samples, with standard deviations in parentheses.

to those figures.

In any case, analysis of the results reveals that learning PLGs with CE may result in better models than learning LDSs with EM when data sets are very large. Another trend that is less clear but is suggested by the data is that this effect is more pronounced with more complex systems.

As a final note, CE tends to be much faster than EM on any given system. Table 5.1 compares the time required for CE and EM for vector-valued controlled models; these numbers are taken for data sets with 10 000 trajectories. The per-iteration times for EM were obtained by timing two runs per sample—one with 1 EM iteration, and one with 6 iterations—then dividing the difference in times by 5. Figure 5.7 shows a histogram of the number of EM iterations executed in the experiments of Section 5.5.4 for  $n = 4$  and  $K = 10\,000$ .

The main thing to notice here is that each iteration of the EM algorithm takes roughly the same amount of time as the entire execution of CE, but that EM requires dozens or hundreds of iterations to achieve its results. This means that CE requires much less time to execute than EM.

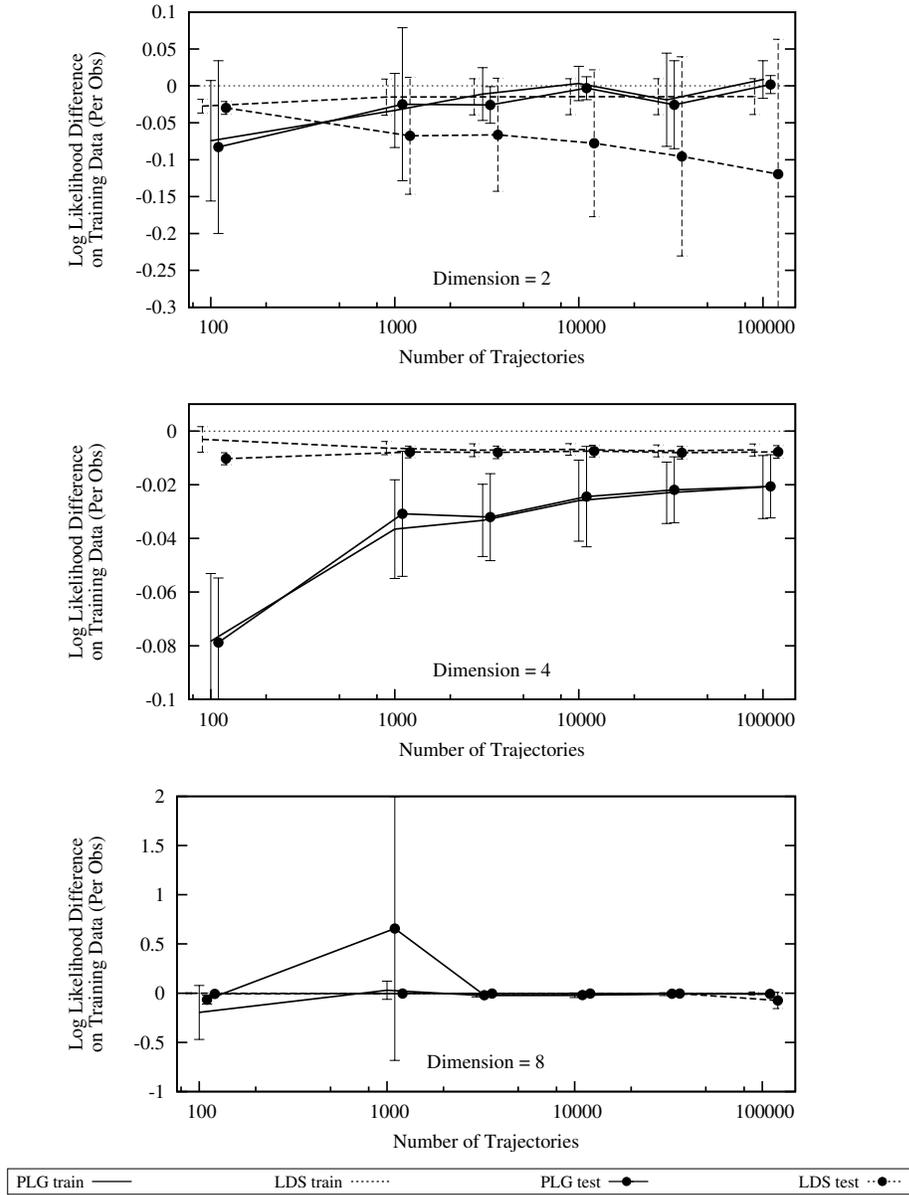


Figure 5.4: Comparison of EM and CE on vector-valued uncontrolled systems. The y-axis shows the difference in log-likelihood per observation between the actual and estimated parameters, over the data used to train the parameters and over a test set. The x-axis shows the number of trajectories in the training set; the x-axis of each curve is slightly offset to improve readability. Higher values indicate higher likelihood in the learned parameters; at zero, the likelihood of the learned parameters is identical to that of the generating parameters. Each curve shows the average across 66 runs; error bars show the 95% confidence interval.

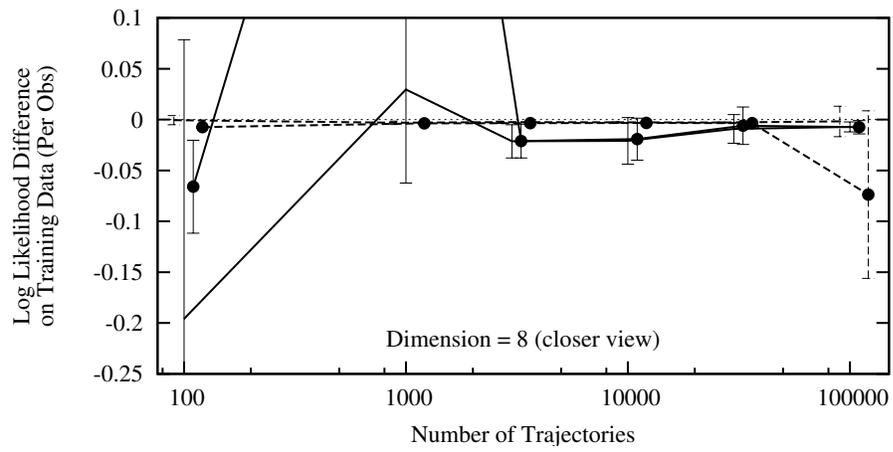
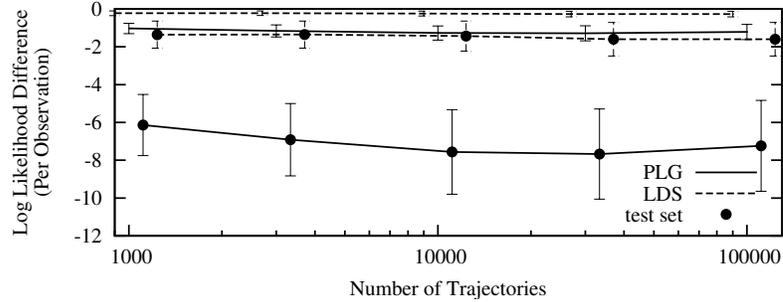
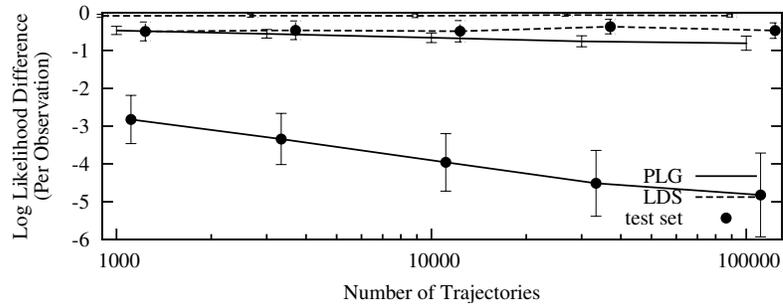


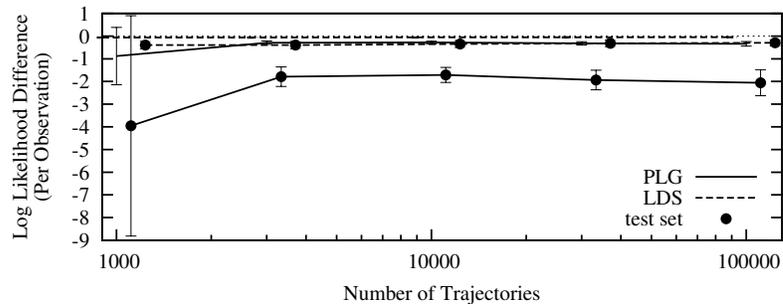
Figure 5.5: A zoomed-in look at the results of CE and EM on vector-valued, uncontrolled systems of dimension 8.



(a) Dimension = 2, Averaged over 70 runs



(b) Dimension = 4, Averaged over 70 runs



(c) Dimension = 8, Averaged over 70 runs

Figure 5.6: Comparison of EM and CE on vector-valued controlled systems. The y-axis shows the difference in log-likelihood per observation between the actual and estimated parameters, over the data used to train the parameters and over a test set. The x-axis shows the number of trajectories in the training set; the x-axis of each curve is slightly offset to improve readability. Higher values indicate higher likelihood in the learned parameters; at zero, the likelihood of the learned parameters is identical to that of the generating parameters. Each curve shows the average across 70 runs; error bars show the 95% confidence interval.

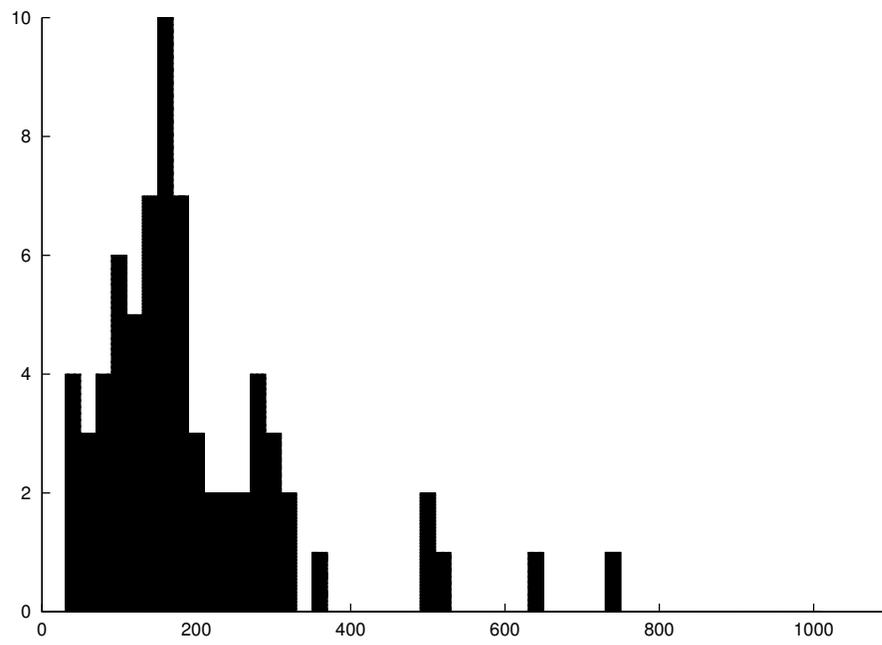


Figure 5.7: A histogram of the number of cycles required by the EM algorithm with  $n = 4$ .

## CHAPTER VI

# Applying the PLG to Real Data

Modeling randomly generated problems is fine as far as it goes, but it is far more interesting to be able to model real world problems. This chapter presents techniques for doing so, then applies those techniques to a traffic modeling problem using data collected from a California freeway.

### 6.1 Techniques for Modeling Applications

There are a number of issues that arise in real applications that are not a problem when working with randomly generated data. For instance, data sets seldom have trajectories that are all the same length. Furthermore, it is unlikely that the data recorded from sensors is all completely relevant to the problem at hand—some data may be extraneous, while other data may not be in its most useful format. Collecting data may also be expensive, and so data sets may be limited, necessitating the ability to deal with relatively small data sets. In this section, I discuss several techniques for dealing with these issues.

### 6.1.1 Discounting the Estimate of $C_\eta$

One of the most problematic estimators in the CE algorithm is  $\widehat{C}_\eta$ . Recall that its definition is

$$(6.1) \quad \widehat{C}_\eta = \frac{1}{N - \tau_{\max}} \sum_{t=0}^{N-\tau_{\max}-1} \frac{1}{K-1} \sum_{k=1}^K (z_t^k - \widetilde{Z}_t^k) \widehat{\eta}_{t+1}^{k\top}.$$

Recall also that  $\widehat{\eta}_{t+1}^k$  and  $\widetilde{Z}_t^k$  both depend on  $\widehat{G}$  and  $\widehat{\Gamma}_1, \dots, \widehat{\Gamma}_{\tau_{\max}}$ . That means that any variance in the estimate of those parameters is magnified in  $\widehat{C}_\eta$ . Though this is also true of  $\widehat{\Sigma}_\eta$ , the summand of that estimator is  $\widehat{\eta}_{t+1}^k \widehat{\eta}_{t+1}^{k\top}$  instead of  $(z_t^k - \widetilde{Z}_t^k) \widehat{\eta}_{t+1}^{k\top}$ ; since  $(z_t^k - \widetilde{Z}_t^k)$  generally has a larger absolute value than  $\widehat{\eta}_{t+1}^k$ , any error is magnified more.

A  $\widehat{C}_\eta$  with large error can cause the model to be quite inaccurate. For example, if some elements of  $\widehat{C}_\eta$  have the wrong sign, it may cause the state update to “correct” in the wrong direction when observations do not match the predictions. Perhaps more problematic is that errors in  $\widehat{C}_\eta$  can cause the  $\Sigma_t$  update to be unstable, leading to values of  $\Sigma_t$  that are not legal variance matrices.

One way to offset this problem is to combine  $\widehat{C}_\eta$  with a “known good” value in a weighted average. Note that when  $C_\eta = \mathbf{0}$ , the state update equation (4.20) reduces to

$$(6.2) \quad \Sigma_{t+1} = G\Sigma_t G^\top + \Sigma_\eta - G\Sigma_t J^\top (J\Sigma_t J^\top)^{-1} J\Sigma_t G^\top.$$

Recall that this is the conditional variance of  $X_2|X_1 = x_1$  when  $X_1$  and  $X_2$  are

Gaussian random variables whose joint variance is given by

$$(6.3) \quad \text{Var} \left[ \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \right] = \begin{pmatrix} G\Sigma_t G^\top + \Sigma_\eta & G\Sigma_t J^\top \\ J\Sigma_t G^\top & J\Sigma_t J^\top \end{pmatrix}$$

$$(6.4) \quad = \begin{pmatrix} G \\ J \end{pmatrix} \Sigma_t \begin{pmatrix} G^\top & J^\top \end{pmatrix} + \begin{pmatrix} \Sigma_\eta & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}.$$

Since  $\widehat{\Sigma}_\eta$  and  $\widehat{\Sigma}_0$  are necessarily symmetric positive semidefinite (they are both of the form  $\sum_k w_k w_k^\top$ , where  $w_k$  are vectors), this variance is symmetric positive semidefinite for  $t = 0$ ; by induction it will remain so for all  $t$  as long as  $J\Sigma_t J^\top$  is invertible.<sup>1</sup>

Therefore,  $C_\eta = \mathbf{0}$  can be taken as a “known good” value for the purposes of maintaining a stable state update equation. A reasonable weight for the weighted average is one that decays exponentially with the size of the data set, i.e.

$$(6.5) \quad \widehat{C}_\eta = (1 - e^{-K/T}) \widehat{C}_\eta^{(5.18)},$$

where  $\widehat{C}_\eta^{(5.18)}$  is the estimate for  $C_\eta$  produced by (5.18) and  $T$  is a tuning parameter. The experiments in this chapter use a value of  $T = 5 \times 10^5$ ; a reasonable value for this parameter will vary with the application.

### 6.1.2 Dealing with Trajectories of Different Lengths

Though the CE algorithm is designed for data sets whose trajectories are all of the same length, it is not uncommon for actual data sets to have trajectories of many different lengths. This is where the weighted version of the CE algorithm described in Section 5.3 comes into play.

The problem is this: since most of the estimators in CE are based on average values across the trajectories, failing to use the weighted version of CE would cause

---

<sup>1</sup>This guarantee does not hold in a variance-adjusted PLG if  $\widehat{\Sigma}_{\text{adj}}$  is not positive semidefinite, but in practice this is not a problem.

data points from the later part of the longest trajectories to be given more weight than data points from the beginning of trajectories. Each data point is given equal importance by using the weighted version of CE and setting the weights so that  $w_t$  is proportional to the number of trajectories with at least  $t$  time steps.

### 6.1.3 Modeling Features as Exogenous Inputs

Data sets frequently contain data from many types of sensors. This data can be divided into four categories. First, data whose values should be predicted by the model. When trying to model a sailboat’s motion, this would include compass bearing and speed. The second category is control inputs. In the sailboat example, rudder position and information about the sails’ trim would fall into the second category. A third category is sensor readings whose values are relevant to the first category but whose values need not be predicted. This might include wind speed and direction. Finally, the data set may include data that is irrelevant to the question at hand. For example, the sailboat data set may include water temperature and depth readings, which may become important at extreme values, but are unlikely to affect the motion of the sailboat.

Most of the categories are easy for a modeler to deal with—sensor readings in the first category become entries in the observation vector  $Y_t$ , those in the second category are entries in the action vector  $u_t$ , and those in the fourth category are omitted from the model. But what of the third category?

The immediate response may be to include data from the third category in the  $Y_t$  vector. And in some cases this may be appropriate. If the data set is large enough to learn an accurate predictive model, distributional information about data in this category may be useful when making predictions about the first category. But in many cases, it is more appropriate to treat these as *exogenous inputs*—that is, an

input that is not explained by the model, and in particular, an input whose future probability distribution is not predicted by the model. Note that the actions are a special case of exogenous inputs: They are inputs that are not explained by the model, but that are controlled by the agent in order to affect future observations. Non-control exogenous inputs can be thought of as actions taken by the environment; like control actions, they affect the future values of the observations, but, again like actions, their own future values are not predicted. Because of this similarity to actions, exogenous inputs can be integrated into the model by making them part of the  $u_t$  action vector, even though semantically they are not control actions.

Returning to the example, when wind speed and direction are treated as observations, the model may have to have a higher dimension, and even then a data set that is too small may learn spurious correlations between current behavior of the sailboat and future behavior of the wind. But when they are treated as exogenous inputs, the model can properly model the effect that current wind conditions have on the behavior of the sailboat.

## 6.2 The NGSIM Traffic Data

To illustrate the utility of the PLG model and CE algorithm, I applied CE to a real-world traffic domain to learn models of the motion of vehicles on the road. The Next Generation SIMulation (NGSIM) program (U.S. Federal Highway Administration, 2008) has collected detailed vehicle trajectory data in support of open algorithms in traffic simulation.

In particular, I focused on NGSIM’s first data set, which recorded data on vehicles driving on a half-kilometer of Interstate 80 in northern California on April 13, 2005 (U.S. Federal Highway Administration, 2006). Figure 6.1 shows an aerial view of

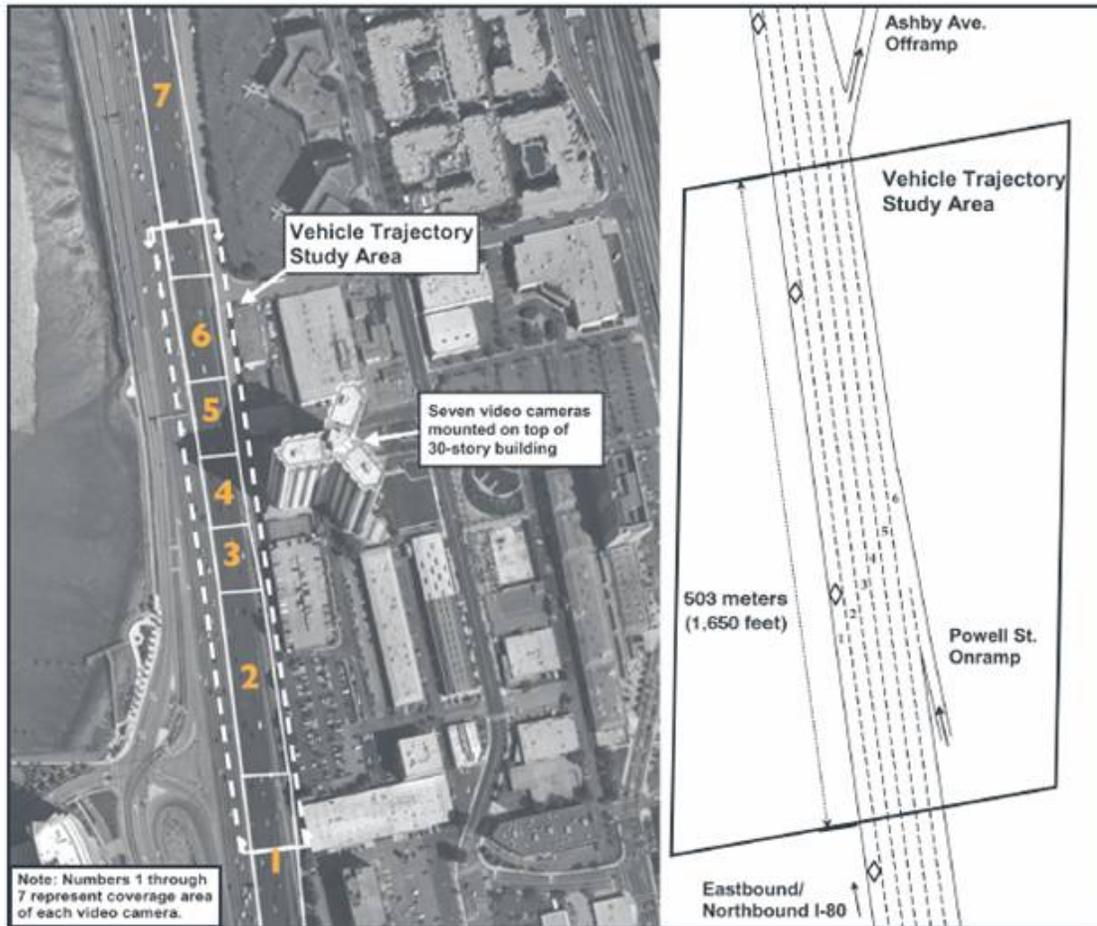


Figure 6.1: An aerial photograph of the section of Interstate 80 covered by the data set. The numbered areas correspond to the seven cameras mounted for the study. Note that the freeway has six lanes in the observed area, with an on-ramp merging into the road approximately one-third of the way in. (U.S. Federal Highway Administration, 2006)

the section of road covered in the experiment. There are three subsets of this data set; I used the subset covering 5:00–5:15 pm. The I-80 data set has a resolution of one-tenth of a second. At each time step, the following data is recorded: the position of the vehicle in both local (i.e. relative to the edge of the road) and global reference frames, its speed and acceleration, its lane, which vehicle is directly in front of it and how far in front it is in feet and seconds, and which vehicle is directly behind it. For each vehicle, the length and width are given, as well as its class (motorcycle, car, or truck).

In my experiments using this data set, my goal was to predict the position of the car in the local reference frame. For this reason, the performance measure I use for each model is the average squared error in the predicted value of the location starting with the fifth observation in each vehicle’s trajectory.<sup>2</sup> The data set contains 1836 vehicles’ trajectories. I divided these trajectories into training and test sets, with each trajectory being selected for the test set with probability 0.1; this resulted in a training set with 1670 trajectories and a test set with 166.

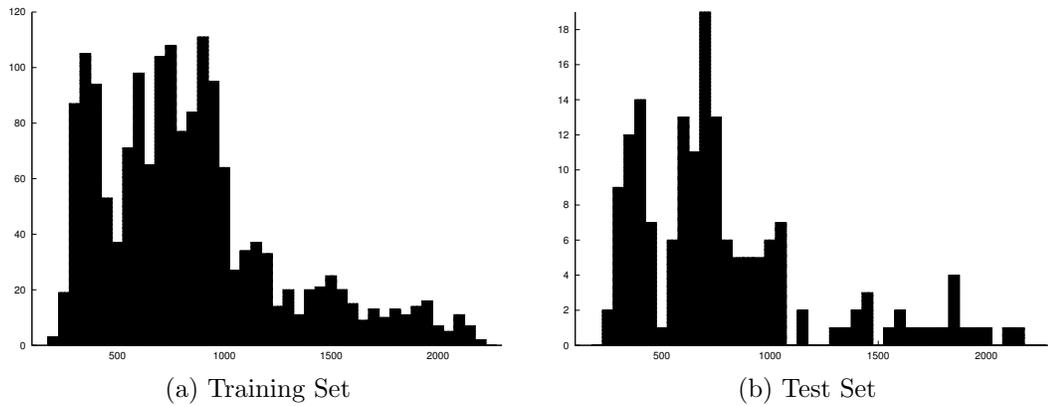


Figure 6.2: The distribution of trajectory lengths in the training and test sets are shown here with a bin width of 50 time steps (i.e. 5 seconds).

### 6.3 Learning Models of the Data

Taking the discussion of Section 6.1.3 into account, I explored models using a number of different subsets of the features provided by the data set, in addition to some derived features I computed. Figure 6.3 describes the two types of radar-style features I computed; these features are essentially occupancy grids for the road surrounding each vehicle.

The small size of the data set (fewer than 2000 vehicles) combined with the ex-

---

<sup>2</sup>With 0.1-second time intervals, this allows for a half-second of “registration” time—that is, it gives the models a few time steps to transition from their inaccurate initial states and register the actual state of the vehicles they are tracking.

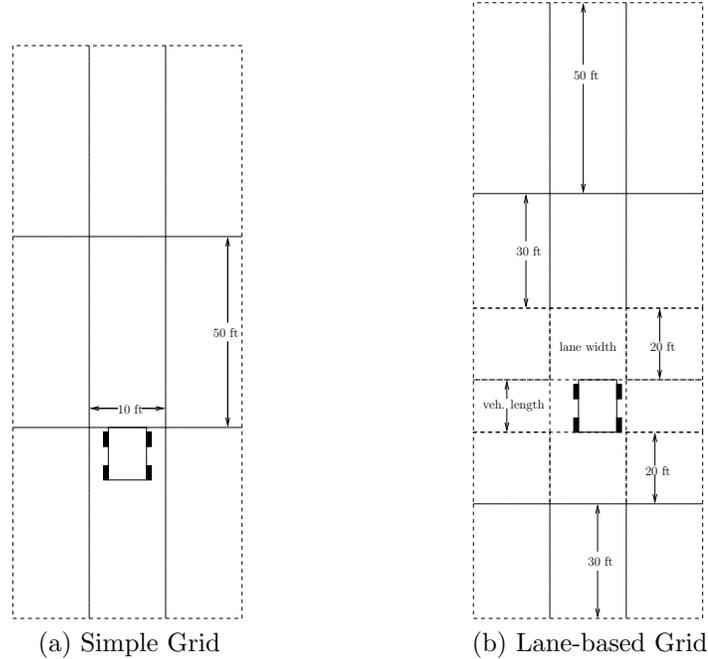


Figure 6.3: There are two types of radar-style features. The simple grid on the left is centered on the front middle of the car. For each rectangle, there is a binary feature that is 1 if there is a car (other than the central car) in the rectangle; in some cases, there is another feature with the velocity of the nearest car in each rectangle. The lane-based grid on the right is centered on the lane that the central car is in. As with the simple grid, there is a binary feature corresponding to each rectangle (except the central rectangle). Any car that is at least partly alongside the central car only counts for the rectangle on the corresponding side, and not the rectangles in front or behind.

perimental results described in the previous chapter caused me to focus on small models: All the models estimated in this chapter have dimension 2.

There are two *a priori* models I consider as a baseline. The first is a model that assumes that each car is stationary. The second assumes a simple kinetic model in which the x-coordinate of the car (i.e. the distance from the left edge of the road) does not change from one time step to the next, and the y-coordinate changes according to  $y' = y + vt + \frac{1}{2}at^2$ . The performance of these models on the test trajectories is given in Table 6.1 along with that of the models I am about to describe.

In all the feature sets presented here, the only observations are the x- and y-coordinates of the vehicle in a road-local coordinate system (measured in feet). Al-

Feature Set	# of features	CE ( $n = 2$ )	EM ( $n = 2$ )
D	27	0.003218	3.463
C	20	0.003243	1.729
E	36	0.003245	1.728
F	8	0.003248	12.28
B	11	0.003248	1.729
A	2	0.003261	0.003401
Kinetic		0.0215	
Stationary		2.759	

Table 6.1: Performance of CE and EM on the test trajectories of the I80 data set using various feature sets. The performance measure is the mean squared error in the local coordinates (measured in feet) of each vehicle at each time step, ignoring the first half second (5 time steps) of each vehicle’s trajectory. The error measure in the EM column gives the lowest error over each of three random initializations of the EM algorithm.

though I explored a few configurations in which other features (such as velocity and acceleration) were included in the observations, models learned using these features performed quite poorly in comparison to those presented here.

The first feature set (Feature Set A) is quite simple: add instantaneous velocity and acceleration as actions.

Feature Set B adds the binary features of the simple grid radar depicted in Figure 6.3(a) to the actions.

Feature Set C adds “radar velocity” to B—that is, for each rectangle in the radar grid, if there is a vehicle in that rectangle, its velocity relative to the target vehicle is given; if there is no vehicle in the rectangle, a 0 is given.

Feature Set D adds a binary feature vector to the actions of Feature Set C; each element of this length-7 vector is 1 if the vehicle is in the corresponding lane and 0 otherwise.

Feature Set E starts with Feature Set A (velocity and acceleration as actions) and adds the binary features of the lane-based grid radar shown in Figure 6.3(b) along with radar velocity features.

Finally, Feature Set F uses no radar-based features at all. Instead, its actions

Feature Set	A (EM)	A (CE)	Compared to				MSE
			B	F	E	C	
D	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	0.003218
C	< 0.001	< 0.001	0.033	0.443	0.360		0.003243
E	< 0.001	0.002	0.472	0.762			0.003245
F	< 0.001	0.006	0.679				0.003248
B	< 0.001	0.002					0.003248
A (CE)	< 0.001						0.003261

Table 6.2: Statistical significance of the differences in performance between models learned using different feature sets. The A (EM) column corresponds to the LDS model learned using EM on Feature Set A; the remaining columns and all rows correspond to PLG models learned using CE on the various feature sets. The table gives the  $p$ -value computed using a paired Student’s t-test, where the mean-squared errors of two models on a given vehicle’s trajectory are paired; the null hypothesis is that the errors are equal.

are a number of features directly from the data set: The length and width of the vehicle, its classification (1 for motorcycle, 2 for car, 3 for truck), its velocity and acceleration, its lane number, and the distance to the car directly in front of it in feet and in seconds at its current velocity.

Table 6.1 shows the mean-squared error in vehicle location for the models learned for each feature set by CE and EM. Table 6.2 shows the statistical significance of the differences between the PLG models learned by CE on each feature set.

## 6.4 Conclusions

EM performs very poorly on this data set. With the exception of Feature Set A, the error on the LDS models learned for these feature sets is about 3 orders of magnitudes worse than the corresponding error in the PLGs learned with CE. It is not clear why this should be; perhaps the LDS parameter space for this problem has an error surface with particularly many local minima. Alternatively, perhaps the EM algorithm just does not handle large action vectors well. In the case of Feature Set A, which has only two actions, EM performs nearly as well as CE.

By contrast, CE performs quite well on this problem. Not only does it attain

smaller average error than EM, but it also performs an order of magnitude better than the *a priori* kinetic model. Of the feature sets studied in this chapter, Feature Set D produces the best model with a mean squared error of  $0.003218\text{ft}^2$ . The difference in errors between this model and the others is highly statistically significant as illustrated in the first row of Table 6.2.

These results suggest that consistent estimation of PLGs is a reasonable approach to modeling real-world problems, even when there are relatively few trajectories in a data set. The fact that CE outperformed EM by orders of magnitude on several of the feature sets illustrates the ability of CE to avoid the local optima that are an inherent problem in an iterative algorithm like EM.

## CHAPTER VII

### Conclusion

#### 7.1 Contributions

I have presented two models: the e-test PSR and the predictive linear Gaussian model. The EPSR was the first nonlinear predictive state model for a general class of systems. This nonlinearity allows it to represent some deterministic systems with an exponentially smaller model than the equivalent linear PSR or POMDP. The PLG extends predictive state models into systems whose actions and observations are continuous-valued vectors rather than discrete scalar values. The main theoretical results regarding the PLG are Theorem 4.2, which shows the equivalence between LDS and PLG models, and Theorem 5.1, which shows that CE is consistent.

The equivalence theorem is important because it shows that the PLG, a predictive state model, is as expressive as the LDS, a widely-used traditional model. This brings all the advantages of predictive state models to the continuous system domain. These advantages, as described in Section 2.1, include verifiability by an automatic agent, better generalization than traditional models, and, potentially, improved learnability.

This last advantage is corroborated by the research into the CE algorithm. Its simplicity is notable—each estimator is at its heart a linear regression, a sample mean, or a sample covariance. Theorem 5.1 provides that CE will produce the correct

parameters given infinite data, but it gives no guarantees about performance with anything less. However, the experimental results of Chapters V and VI illustrate that learning PLGs with consistent estimation performs well compared to learning LDSs with expectation maximization, even in a real-world application with a relatively small data set.

## 7.2 Future Work

One important limitation of the PLG is its linearity. Many problems can not be satisfactorily modeled with linear dynamics. Happily, the problem of extending the PLG to nonlinear systems has been studied: the kernel-based PLG (Wingate & Singh, 2006a) and a mixture of PLGs (Wingate & Singh, 2006b) both use predictive state and Gaussian distributions to model nonlinear systems. These models are unified, along with the PLG, in the Exponential Family PSR (Wingate & Singh, 2008), which is quite general; these models are explored in depth in David Wingate’s doctoral dissertation (Wingate, 2008).

Further work in the area of estimation algorithms is also warranted. For example, as noted earlier, the consistent estimation algorithm gives no guarantees of its performance with finite data sets. A maximum-likelihood-style algorithm would be quite useful to improve learning performance on smaller data sets. Appendix C gives the gradients of the likelihood with respect to each of the PLG’s parameters, but they resist analytic maximization, and my experiments with quasi-Newton methods using these gradients proved fruitless.

## APPENDICES

## APPENDIX A

### Proofs and Derivations

#### A.1 Derivation of the PLG Update Equations

##### A.1.1 Derivation of the Scalar PLG Update Equations

The purpose of the state update is to compute the conditional distribution of  $Z_{t+1}|h_t y_{t+1}$  from  $\mu_t$  and  $\Sigma_t$ , the mean and covariance of  $Z_t|h_t$ , given that  $Y_{t+1} = y_{t+1}$ . Since  $Z_{t+1}$  and  $Y_{t+1}$  are jointly Gaussian random variables, the following lemma can be applied:

**Lemma A.1.** *If the random vectors  $Y$  and  $Z$  are drawn from the joint Gaussian distribution*

$$\begin{pmatrix} Y \\ Z \end{pmatrix} \sim \mathcal{N} \left( \begin{pmatrix} \mu_Y \\ \mu_Z \end{pmatrix}, \begin{pmatrix} \Sigma_{YY} & \Sigma_{YZ}^\top \\ \Sigma_{YZ} & \Sigma_{ZZ} \end{pmatrix} \right),$$

*then  $Z|Y = y \sim \mathcal{N}(\mu_Z + \Sigma_{YZ}\Sigma_{YY}^{-1}(y - \mu_Y), \Sigma_{ZZ} - \Sigma_{YZ}\Sigma_{YY}^{-1}\Sigma_{YZ}^\top)$  (e.g., Theorem 3.5.2 of Catlin, 1989).*

The problem, then, is to compute the joint distribution of  $Y_{t+1}$  and  $Z_{t+1}$  from the

distribution of  $Z_t$ . This can be achieved by noting that

$$(A.1) \quad \begin{pmatrix} Y_{t+1} \\ Z_{t+1} \end{pmatrix} = \begin{pmatrix} Y_{t+1} \\ Y_{t+2} \\ \vdots \\ Y_{t+n} \\ Y_{t+n+1} \end{pmatrix} = \begin{pmatrix} Z_t \\ Y_{t+n+1} \end{pmatrix}.$$

The two elements of the matrix on the right-hand side can be expressed in terms of  $Z_t$ ; one element is  $Z_t$ , and the other can be computed using (4.3), the core dynamics of the scalar PLG:

$$(A.2) \quad Y_{t+n+1} = g^\top Z_t + \varepsilon_{t+n+1}.$$

Since  $E[\varepsilon_{t+n+1}|h_t] = 0$ , the mean of (A.1) is given by

$$(A.3) \quad E \left[ \begin{pmatrix} Z_t \\ Y_{t+n+1} \end{pmatrix} \middle| h_t \right] = \begin{pmatrix} \mu_t \\ g\mu_t \end{pmatrix}.$$

Its covariance matrix can be computed piecewise.  $\text{Var}[Z_t|h_t] = \Sigma_t$  is given by the definition of the state variable. Computing the covariance of  $Z_t$  and  $Y_{t+n+1}$  makes use of (A.2):

$$(A.4) \quad \text{Cov}[Z_t, Y_{t+n+1}|h_t] = \text{Cov}[Z_t, g^\top Z_t + \varepsilon_{t+n+1}|h_t]$$

$$(A.5) \quad = \text{Cov}[Z_t, g^\top Z_t|h_t] + \text{Cov}[Z_t, \varepsilon_{t+n+1}]$$

$$(A.6) \quad = \Sigma_t g + C.$$

The variance of  $Y_{t+n+1}$  is

$$(A.7) \quad \text{Var}[Y_{t+n+1}|h_t] = \text{Cov}[g^\top Z_t + \varepsilon_{t+n+1}, g^\top Z_t + \varepsilon_{t+n+1}|h_t]$$

$$(A.8) \quad = \text{Var}[g^\top Z_t|h_t] + \text{Cov}[g^\top Z_t, \varepsilon_{t+n+1}|h_t] \\ + \text{Cov}[\varepsilon_{t+n+1}, g^\top Z_t|h_t] + \text{Var}[\varepsilon_{t+n+1}|h_t]$$

$$(A.9) \quad = g^\top \Sigma_t g + g^\top C + C^\top g + \sigma^2.$$

The variance of the combined vector, then, is

$$(A.10)$$

$$\text{Var} \left[ \begin{pmatrix} Z_t \\ Y_{t+n+1} \end{pmatrix} \middle| h_t \right] = \begin{pmatrix} \Sigma_t & \Sigma_t g + C \\ g^\top \Sigma_t + C^\top & g^\top \Sigma_t g + g^\top C + C^\top g + \sigma^2 \end{pmatrix}$$

$$(A.11) \quad = \begin{pmatrix} e_1^\top \Sigma_t e_1 & e_1^\top \Sigma_t G^\top + e_1^\top C e_n^\top \\ G \Sigma_t e_1 + e_n C^\top e_1 & G \Sigma_t G^\top + G C e_n^\top + e_n C^\top G^\top + \sigma^2 e_n e_n^\top \end{pmatrix}$$

$$(A.12) \quad = \text{Var} \left[ \begin{pmatrix} Y_{t+1} \\ Z_{t+1} \end{pmatrix} \middle| h_t \right],$$

where  $G$  is given by

$$(A.13) \quad G = \begin{pmatrix} \mathbf{0} & \vdots & I_{n-1} \\ \cdots & \cdots & \cdots \\ & & g^\top \end{pmatrix},$$

as in (4.7),  $I_n$  is the  $n \times n$  identity matrix, and  $e_i$  is the  $i$ th column of  $I_n$ . The matrix  $G$ , when multiplied by an  $n$ -vector, shifts each of the last  $n-1$  elements of the vector up by one element, and replaces the final element by the inner product of  $g$  with the vector. This is precisely what is called for when computing  $Z_{t+1}$  from  $Z_t$ , which is why it appears in the combined covariance matrix in (A.11), and why it is used to compute

$$(A.14) \quad \text{E}[Z_{t+1}|h_t] = G\mu_t.$$

All that remains is to apply Lemma A.1 to compute the conditional distribution of  $Z_{t+1}|h_t, Y_{t+1} = y_{t+1}$  from the joint distribution of  $Z_{t+1}|h_t$  and  $Y_{t+1}|h_t$ :

$$(A.15) \quad \text{E}[Z_{t+1}|h_{t+1}] = \mu_{t+1} = G\mu_t + (G\Sigma_t + e_n C^\top) e_1 (e_1^\top \Sigma_t e_1)^{-1} (y_{t+1} - e_1^\top \mu_t),$$

$$(A.16) \quad \text{Var}[Z_{t+1}|h_{t+1}] = \Sigma_{t+1} = \\ G\Sigma_t G^\top + GCe_n^\top + e_n C^\top G^\top + \sigma^2 e_n e_n^\top - (G\Sigma_t + e_n C^\top) e_1 (e_1 \Sigma_t e_1^\top)^{-1} e_1^\top (\Sigma_t G^\top + C e_n^\top).$$

### A.1.2 Derivation of the Full PLG Update Equations

This closely follows the derivation of the scalar state update function; the covariance of  $Y_{t+1}|h_t$  and  $Z_{t+1}|h_t$  is computed, and then Lemma A.1 is used to calculate the conditional distribution of  $Z_{t+1}|h_t, Y_{t+1} = y_{t+1}, u_{t+1}$ . But first,  $L$  must be computed.

$L$  is the matrix that captures the effect of the action  $u_{t+1}$  on  $Z_{t+1}$ ; that is, it satisfies

$$(A.17) \quad Lu = \mathbb{E}[Z_{t+1}|h_t, u_{t+1} = u, u_{t+2} = \mathbf{0}, \dots] - \mathbb{E}[Z_{t+1}|h_t, u_{t+1} = \mathbf{0}, u_{t+2} = \mathbf{0}, \dots].$$

The issue, then, is determining this effect. To do this, consider two PLG trajectories,  $k_1$  and  $k_2$ . Suppose that  $\eta_{t+1}^{k_1} = \eta_{t+1}^{k_2}$  and  $u_t^{k_1} = u_t^{k_2}$  at every time step  $t$ . Starting after some time step  $\tau$ , suppose that all actions are zero; that is,  $u_{\tau+1}^{k_1} = u_{\tau+1}^{k_2} = u_{\tau+2}^{k_1} = u_{\tau+2}^{k_2} = \dots = \mathbf{0}$ .

Recall the core PLG dynamics from (4.15):

$$(A.18) \quad Z_{t+1} = GZ_t + \sum_{i=1}^{\tau_{\max}} \Gamma_i u_{t+i} + \eta_{t+1}$$

Then  $Z_t^{k_1} - Z_t^{k_2} = \mathbf{0}$  at every  $t$ . Now change  $u_{\tau+1}^{k_1}$  to  $u$ . Now it still holds

$$(A.19) \quad Z_{\tau-\tau_{\max}}^{k_1} - Z_{\tau-\tau_{\max}}^{k_2} = \mathbf{0},$$

because none of the terms of (A.18) have been affected for  $t = \tau - \tau_{\max}$ . However,

$$(A.20) \quad Z_{\tau-\tau_{\max}+1}^{k_1} - Z_{\tau-\tau_{\max}+1}^{k_2} = \Gamma_{\tau_{\max}} u_{\tau+1}$$

because the last summand in the summation differs in the two trajectories.

Moving one step further forward in time to consider  $Z_{\tau-\tau_{\max}+2}$ , *two* terms in the dynamics equation differ: the penultimate summand, and the in first term—that is,  $GZ_{\tau-\tau_{\max}+1}^{k_1} \neq GZ_{\tau-\tau_{\max}+1}^{k_2}$ . This results in a difference of

$$(A.21) \quad Z_{\tau-\tau_{\max}+2}^{k_1} - Z_{\tau-\tau_{\max}+2}^{k_2} = G(\Gamma_{\tau_{\max}} u_{\tau+1}) + \Gamma_{\tau_{\max}-1} u_{\tau+1}$$

Continuing in this manner up to  $Z_{\tau+1}$  results in

$$(A.22) \quad Z_{\tau+1}^{k_1} - Z_{\tau+1}^{k_2} = \left( \sum_{i=1}^{\tau_{\max}} G^{i-1} \Gamma_i \right) u_{\tau+1}.$$

This difference is also the difference in expectations between  $Z_{t+1}|h_t, u_{t+1} = u_{\tau+1}$  and  $Z_{t+1}|h_t, u_{t+1} = \mathbf{0}$ ; therefore,  $L$  may be defined as

$$(A.23) \quad L = \sum_{i=1}^{\tau_{\max}} G^{i-1} \Gamma_i.$$

Deriving the state update equations is now simply a matter of computing the distributions of  $Y_{t+1}|h_t, u_{t+1}$  and  $Z_{t+1}|h_t, u_{t+1}$  from the distribution of  $Z_t|h_t$  and then applying Lemma A.1.

The former distribution is straightforward; (4.11) states that  $Y_{t+1} = JZ_t$ , so

$$(A.24) \quad Y_{t+1}|h_t, u_{t+1} \sim \mathcal{N}(J\mu_t, J\Sigma_t J^\top).$$

Computing the distribution of  $Z_{t+1}$  is also straightforward. From (A.18), and recognizing that the actions do not affect the variance calculations,

$$(A.25) \quad \text{Var}[Z_{t+1}|h_t, u_{t+1}] = \text{Var}[Z_{t+1}|h_t]$$

$$(A.26) \quad = \text{Cov}[GZ_t + \eta_{t+1}, GZ_t + \eta_{t+1}|h_t]$$

$$(A.27) \quad = \text{Var}[GZ_t|h_t] + \text{Cov}[GZ_t, \eta_{t+1}|h_t] \\ + \text{Cov}[\eta_{t+1}, GZ_t|h_t] + \text{Var}[\eta_{t+1}|h_t]$$

$$(A.28) \quad = G\Sigma_t G^\top + GC_\eta + C_\eta^\top G^\top + \Sigma_\eta.$$

Next, the covariance of  $Z_{t+1}$  and  $Y_{t+1}$ :

$$(A.29) \quad \text{Cov}[Y_{t+1}, Z_{t+1}|h_t, u_{t+1}] = \text{Cov}[Y_{t+1}, Z_{t+1}|h_t] = \text{Cov}[JZ_t, JZ_t + \eta_{t+1}|h_t]$$

$$(A.30) \quad = J\Sigma_t G^\top + JC_\eta$$

Finally, the expected values of the random vectors must be computed. Because of the construction of  $J$  ( $JZ_t$  must be a function only of those elements of  $Z_t$  that are also elements of  $Y_{t+1}$ ) and of  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  (the rows of  $\Gamma_i$  corresponding to elements of  $Z_t$  that belong to  $Y_{t+j}$  are  $\mathbf{0}$  when  $j < i$ ),  $J\Gamma_2 = JG\Gamma_3 = JG^2\Gamma_4 = \dots = \mathbf{0}$ . Therefore,

$$(A.31) \quad \text{E}[Y_{t+1}|h_t, u_{t+1}] = \text{E}[JZ_t|h_t, u_{t+1}] = \text{E}[JZ_t|h_t] = J\mu_t.$$

On the other hand, the expected value of  $Z_{t+1}$  is affected by the action taken:

$$(A.32) \quad \text{E}[Z_{t+1}|h_t, u_{t+1}] = \text{E}[Z_{t+1}|h_t] + Lu_{t+1} = G\mu_t + Lu_{t+1}.$$

Having computed the joint distribution, it is now possible to write down the conditional distribution:

$$(A.33)$$

$$\begin{aligned} Z_{t+1}|h_t, u_{t+1}, Y_{t+1} = y_{t+1} \\ \sim \mathcal{N}(G\mu_t + Lu_{t+1} + (G\Sigma_t + C_\eta^\top)J^\top (J\Sigma_t J^\top)^{-1}(y_{t+1} - J\mu_t), \\ G\Sigma_t G^\top + GC_\eta + C_\eta^\top G^\top + \Sigma_\eta - (G\Sigma_t + C_\eta^\top)J^\top (J\Sigma_t J^\top)^{-1}J(\Sigma_t G^\top + C_\eta)). \end{aligned}$$

This yields the state updates given in (4.19) and (4.20).

## A.2 Proof of Theorem 4.2

**Theorem.** *Every LDS with  $n$ -dimensional state has an equivalent  $n$ -dimensional variance-adjusted PLG, where equivalence means that both models compute the same probability distribution over futures given the same history.*

This theorem is proven by constructing an  $n$ -dimensional variance-adjusted PLG that computes the same probability distribution over futures as an arbitrary (given)  $n$ -dimensional LDS with parameters  $H$ ,  $A$ ,  $Q$ ,  $R$ , and initial state  $x_0^-$  and  $P_0^-$ .

This construction is accomplished by creating a mapping from Kalman filter state variables  $x_{t+1}^-$  and  $P_{t+1}^-$  to PLG state variables  $\mu_t$  and  $\Sigma_t$ , computing parameters  $J$  and  $\Sigma_{\text{adj}}$  so that both models compute the same distribution for  $Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots$ , and then computing the remaining PLG parameters so that the state mapping is preserved by the state update equations. An inductive argument is then used to show that the new PLG computes the same distribution over all futures as the LDS.

### A.2.1 Populating $Z_t$

In order to compute a mapping from Kalman filter state to PLG state, the observations that will make up  $Z_t$  must first be selected. As a matter of notation, the indexing functions  $\tau(\cdot)$  and  $\rho(\cdot)$  will be used to indicate which observational element is associated with each element of  $Z_t$ ; the  $i$ th element of  $Z_t$  is the  $\rho(i)$ th element of  $Y_{t+\tau(\cdot)}$ :

$$(A.34) \quad Z_t^i = Y_{t+\tau(\cdot)}^{\rho(i)}.$$

To determine which observations will be linearly independent of one another in expectation (and therefore should be included in  $Z_t$ ), the matrix  $\mathcal{M}_n$  will be computed, where

$$(A.35) \quad \mathcal{M}_i = \begin{pmatrix} H \\ HA \\ \vdots \\ HA^{i-1} \end{pmatrix}.$$

Note that

$$(A.36) \quad \mathcal{M}_i x_{t+1}^- = \begin{pmatrix} \mathbb{E}[Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots] \\ \mathbb{E}[Y_{t+2}|h_t, u_{t+1} = \mathbf{0}, \dots] \\ \vdots \\ \mathbb{E}[Y_{t+i}|h_t, u_{t+1} = \mathbf{0}, \dots] \end{pmatrix}.$$

Now select the first  $n$  rows that span  $\mathcal{M}_n$  (if  $\mathcal{M}_n$  has rank less than  $n$ , this won't be a *minimal* basis). Populate  $\tau(\cdot)$  and  $\rho(\cdot)$  by noting that each row corresponds to an observation:  $H_{i,\cdot} A^j x_{t+1}^- = \mathbb{E}[Y_{t+j+1}^i | h_t, u_{t+1} = \mathbf{0}, \dots]$ , where  $H_{i,\cdot}$  refers to the  $i$ th row of  $H$ ), so the row of  $\mathcal{M}_n$  that equals  $H_{i,\cdot} A^j$  corresponds to  $\tau(\cdot)$  and  $\rho(\cdot)$  values of  $j+1$  and  $i$ , respectively. Selecting elements of  $Z_t$  in this way results in a vector that follows the “skyline” rule illustrated in Figure 4.2— $Y_{t+j+1}^i$  will not be a member of  $Z_t$  unless  $Y_{t+j}^i$  is, for  $j \geq 1$ .

This is because of the following lemma:

**Lemma A.2.** *If the row of  $\mathcal{M}_n$  corresponding to  $Y_{t+j}^i$  is linearly dependent on the rows above it, then the row corresponding to  $Y_{t+j+1}^i$  is linearly dependent on the rows above it.*

*Proof.* Since the row corresponding to  $Y_{t+j}^i$  is linearly dependent on the rows above it, it may be written as a linear combination of those rows:

$$(A.37) \quad H_{i,\cdot} A^{j-1} = \sum_{k=1}^m \sum l = 0^{j-1} \alpha_{k,l} H_{k,\cdot} A^l$$

for some weights  $\alpha_{k,l}$ , with  $\alpha_{k,j-1} = 0$  for  $k \geq i$ . Right-multiplying both sides by  $A$ , an expression for the row corresponding to  $Y_{t+j+1}^i$  is obtained:

$$(A.38) \quad H_{i,\cdot} A^j = \sum_{k=1}^m \sum l = 0^{j-1} \alpha_{k,l} H_{k,\cdot} A^{l+1}.$$

That is, it is a linear combination of the rows above it. □

### A.2.2 Computing the State Mapping

In order to compute the mapping from Kalman filter state to PLG state, two identities will be useful. First, as mentioned in the previous section,

$$(A.39) \quad \mathbb{E}[Y_{t+j}^i | h_t, u_{t+1} = \mathbf{0}, \dots] = H_{i,\cdot} A^{j-1} x_{t+1}^-.$$

Second, the covariance of two observations in terms of the LDS parameters is given in the expression

$$(A.40) \quad \text{Cov}[Y_{t+i}^a, Y_{t+j}^b | h_t, u_{t+1} = \mathbf{0}, \dots] = \\ H_{a,\cdot} A^{i-1} P_{t+1}^- (A^{j-1})^\top H_{b,\cdot}^\top + H_{a,\cdot} A^{i-j} S_{j-1} H_{b,\cdot}^\top + \delta_{i,j} R_{a,b},$$

where  $i \geq j \geq 1$ ,  $R_{a,b}$  is the  $(a, b)$ th element of  $R$ , and

$$(A.41) \quad S_i = \sum_{k=1}^i A^{k-1} Q (A^{k-1})^\top$$

is the variance of the difference  $X_{t+i} - A^i X_t$ .

Now define  $M$  to be the matrix made up of the rows of  $\mathcal{M}_n$  selected while populating  $Z_t$ ; that is, the  $i$ th row of  $M$  is given by

$$(A.42) \quad M_{i,\cdot} = H_{\rho(i),\cdot} A^{\tau(i)-1}.$$

Then  $\mu_t = M x_{t+1}^-$  is a correct mapping, as by (A.39) it holds that

$$(A.43) \quad M x_{t+1}^- = \mathbb{E}[Z_t | h_t, u_{t+1} = \mathbf{0}, \dots].$$

Furthermore, each element of  $\Sigma_t$  is a covariance in the form of (A.40). It therefore holds that  $\text{Var}[Z_t | h_t, u_{t+1} = \mathbf{0}, \dots] = M P_{t+1}^- M^\top + \Phi$ , where

$$(A.44) \quad \Phi_{ij} = H_{\rho(i),\cdot} A^{\tau(i)-\tau(j)} S_{\tau(j)-1} H_{\rho(j),\cdot}^\top + \delta_{\tau(i),\tau(j)} R_{\rho(i),\rho(j)}$$

for  $\tau(i) \geq \tau(j)$ ; when  $\tau(i) < \tau(j)$ ,  $\Phi_{ij} = \Phi_{ji}$ . This implies that the state mapping for covariance matrices is  $\Sigma_t = M P_{t+1}^- M^\top + \Phi$ . It is important to note that  $\Phi$  and  $M$  are both independent of history.

### A.2.3 Computing $J$ and $\Sigma_{\text{adj}}$

The next-observation function  $J$  can now be computed in a straightforward manner by noting that

$$(A.45) \quad J\mu_t = \mathbb{E}[Y_{t+1}|h_t] = Hx_{t+1}^-,$$

where the equality on the left follows from the PLG model and the one on the right follows from the LDS. Since  $\mu_t = Mx_{t+1}^-$ , it follows that  $JM = H$  is a sufficient condition to satisfy (A.45). Because of the other constraints on  $J$  (namely, that  $JZ_t$  must be a function only of elements of  $Y_{t+1}$ ), the  $i$ th column of the solution to  $JM = H$  must be  $\mathbf{0}$  whenever  $\tau(i) > 1$ . This implies that  $JM$  must reconstruct  $H$  from only the rows of  $H$  that are included in  $M$ . Since *all* of the linearly independent rows of  $H$  are included in  $M$  ( $Z_t$  was created by selecting the *first*  $n$  spanning rows of  $\mathcal{M}_n$ ), this is possible, and such a  $J$  exists.

The next step is to compute  $\Sigma_{\text{adj}}$  so that both models compute the same variance for the next observation. In the LDS model, this is given by  $HP_{t+1}^-H^\top + R$ ; in the PLG, the same variance is  $J\Sigma_t J^\top + \Sigma_{\text{adj}}$ . Substituting  $MP_{t+1}^-M^\top + \Phi$  for  $\Sigma_t$  and  $H$  for  $JM$ , then setting these quantities equal to each other results in

$$(A.46) \quad J\Sigma_t J^\top + \Sigma_{\text{adj}} = JMP_{t+1}^-M^\top J^\top + J\Phi J^\top + \Sigma_{\text{adj}}$$

$$(A.47) \quad = HP_{t+1}^-H^\top + J\Phi J^\top + \Sigma_{\text{adj}}$$

$$(A.48) \quad = HP_{t+1}^-H^\top + R.$$

Solving for  $\Sigma_{\text{adj}}$ :

$$(A.49) \quad \Sigma_{\text{adj}} = R - J\Phi J^\top.$$

Because  $S_0 = \mathbf{0}$ , whenever  $\tau(i) = \tau(j) = 1$ ,  $\Phi_{ij} = R_{i,j}$ . Because  $J\Phi J$  simply copies these elements, it follows that in this case, the  $ij$ th element of  $\Sigma_{\text{adj}}$  will be 0.

In particular, when all elements of  $Y_{t+1}$  are also elements of  $Z_t$ ,  $\Sigma_{\text{adj}} = \mathbf{0}$ . Since all elements of  $Y_{t+1}$  will be part of  $Z_t$  whenever  $H$  has rank  $m$  (and pending the rest of the proof), this means that a standard (i.e. non-variance-adjusted) PLG will suffice when  $H$  has rank  $m$ , thus proving Theorem 4.1.

The method by which  $J$  and  $\Sigma_{\text{adj}}$  were chosen guarantees that the PLG will compute the same distribution for  $Y_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots$  as the LDS does whenever the state mapping holds. Thus the remaining parameters of the PLG must be chosen so as to preserve this mapping through the state update.

#### A.2.4 Computing the Remaining PLG Parameters

The linear trend  $G$  can be computed in a similar manner to  $J$ : In the PLG model,  $E[Z_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots] = G\mu_t$ ; in the LDS, this expectation is given by  $MAx_{t+1}^-$ . Since  $\mu_t = Mx_{t+1}^-$ , it follows that  $G$  is a solution to  $GM = MA$ . By the following argument, such a solution must exist.

All the rows of  $M$  are rows of  $\mathcal{M}_n$ , so the rows of  $MA$  are rows in  $\mathcal{M}_{n+1}$ . By Lemma A.3 below, all the rows of  $\mathcal{M}_{n+1}$  are linearly dependent on the rows of  $\mathcal{M}_n$ . By construction, all the rows of  $\mathcal{M}_n$  are linearly dependent on  $M$ . Thus, each row of  $MA$  is a linear combination of the rows of  $M$ , so  $G$  exists.

**Lemma A.3.** *All the rows of  $\mathcal{M}_{n+1}$  are linearly dependent on the rows of  $\mathcal{M}_n$ .*

*Proof.* By Lemma A.2, if  $H_{i,\cdot}A^{j-1}$  is linearly dependent on the rows above it in  $\mathcal{M}_j$ , then  $H_{i,\cdot}A^j$  is linearly dependent on the rows above it in  $\mathcal{M}_{j+1}$ . This implies that, if  $\text{rk}(\mathcal{M}_{j+1}) = \text{rk}(\mathcal{M}_j)$ , then  $\text{rk}(\mathcal{M}_k) = \text{rk}(\mathcal{M}_j)$  for all  $k \geq j$ .

Since  $\mathcal{M}_j$  is a submatrix of  $\mathcal{M}_k$  for  $k \geq j$ , having the same rank means that all the rows of  $\mathcal{M}_k$  are linearly dependent on  $\mathcal{M}_j$ . Since  $\mathcal{M}_n$  has only  $n$  columns, its rank is bounded from above by  $n$ .  $\mathcal{M}_1$ 's rank is bounded from below by 1.

Thus, either  $\text{rk}(\mathcal{M}_n) = n$  or  $\text{rk}(\mathcal{M}_{n-1}) = \text{rk}(\mathcal{M}_n)$ ; in either case, it must hold that  $\text{rk}(\mathcal{M}_{n+1}) = \text{rk}(\mathcal{M}_n)$ . The result follows directly.  $\square$

Any  $G$  that solves  $GM = MA$  is acceptable.

It is now possible to compute the effects of the actions. Recall that  $L$  describes the effect of  $u_{t+1}$  on  $Z_{t+1}$ :

$$(A.50) \quad Lu = \mathbb{E}[Z_{t+1}|h_t, u_{t+1} = u, u_{t+2} = \mathbf{0}, \dots] - \mathbb{E}[Z_{t+1}|h_t, u_{t+1} = \mathbf{0}, u_{t+2} = \mathbf{0}, \dots].$$

In LDS terms, this difference is written as  $MBu$ . Thus  $L = MB$ . For many applications, having  $L$  is sufficient. If  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  are desired, any solution to

$$(A.51) \quad L = \Gamma_1 + G\Gamma_2 + \dots + \Gamma^{\tau_{\max}-1}\Gamma_{\tau_{\max}},$$

with the additional constraint that the appropriate rows of  $\Gamma_1, \dots, \Gamma_{\tau_{\max}}$  are fixed to be  $\mathbf{0}$ , will suffice.

The only parameters left to compute are the noise parameters  $C_\eta$  and  $\Sigma_\eta$ . By the PLG model,

$$(A.52) \quad \begin{aligned} \text{Cov}[Z_t, Z_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots] &= \\ &= \text{Cov}[Z_t, GZ_t|h_t, u_{t+1} = \mathbf{0}, \dots] + \text{Cov}[Z_t, \eta_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots] \end{aligned}$$

$$(A.53) \quad = \Sigma_t G^\top + C_\eta.$$

Solving for  $C_\eta$  results in

$$(A.54) \quad C_\eta = \text{Cov}[Z_t, Z_{t+1}|h_t, u_{t+1} = \mathbf{0}, \dots] - \Sigma_t G^\top.$$

The last term on the right-hand side can be rewritten in terms of LDS parameters as

$$(A.55) \quad \Sigma_t G^\top = MP_{t+1}^- M^\top G^\top + \Phi G^\top = MP_{t+1}^- A^\top M^\top + \Phi G^\top.$$

From (A.40), it follows that

$$(A.56) \quad \text{Cov}[Z_t, Z_{t+1} | h_t, u_{t+1} = \mathbf{0}, \dots] = MP_{t+1}^- A^\top M^\top + \Omega,$$

where the  $(i, j)$ th element of  $\Omega$  is given by

$$(A.57) \quad \Omega_{ij} = \begin{cases} H_{\rho(i), \cdot} A^{\tau(i)-\tau(j)-1} S_{\tau(j)} H_{\rho(j), \cdot}^\top + \delta_{\tau(i), \tau(j)+1} R_{\rho(i), \rho(j)} & \tau(i) > \tau(j) \\ H_{\rho(j), \cdot} A^{\tau(j)-\tau(i)+1} S_{\tau(i)-1} H_{\rho(i), \cdot}^\top + \delta_{\tau(i), \tau(j)+1} R_{\rho(i), \rho(j)} & \text{else.} \end{cases}$$

$\Omega$  is essentially  $\Phi$  with all instances of  $\tau(j)$  replaced with  $\tau(j) + 1$ ; this makes sense, as  $\Phi$  was used to compute the covariance of  $Z_t$  with itself, while  $\Omega$  is used to compute the covariance of  $Z_t$  with  $Z_{t+1}$ . Computing  $C_\eta$  is now easy:

$$(A.58) \quad C_\eta = \Omega - \Phi G^\top.$$

A similar derivation is used to compute  $\Sigma_\eta$ . According to the PLG,

$$(A.59) \quad \begin{aligned} & \text{Cov}[Z_{t+1}, Z_{t+1} | h_t, u_{t+1} = \mathbf{0}, \dots] = \\ & = G \Sigma_t G^\top + G C_\eta + C_\eta^\top G^\top + \Sigma_\eta \end{aligned}$$

$$(A.60) \quad = G M P_{t+1}^- M^\top G^\top + G \Phi G^\top + G C_\eta + C_\eta^\top G^\top + \Sigma_\eta$$

$$(A.61) \quad = M A P_{t+1}^- A^\top M^\top + G \Phi G^\top + G C_\eta + C_\eta^\top G^\top + \Sigma_\eta.$$

In the LDS, this covariance is  $M A P_{t+1}^- A^\top M^\top + \Theta$ .  $\Theta$  can be computed by replacing  $\tau(i)$  by  $\tau(i + 1)$  in  $\Omega$ :

$$(A.62) \quad \Theta_{ij} = H_{\rho(i), \cdot} A^{\tau(i)-\tau(j)} S_{\tau(j)} H_{\rho(j), \cdot}^\top + \delta_{\tau(i), \tau(j)} R_{\rho(i), \rho(j)}$$

when  $\tau(i) \geq \tau(j)$  and  $\Theta_{ji}$  otherwise. Note that  $\Phi$  and  $\Theta$  are symmetric, while  $\Omega$  is not.

$\Sigma_\eta$  is thus given by

$$(A.63) \quad \Sigma_\eta = \Theta - G \Phi G^\top - G C_\eta - C_\eta^\top G^\top.$$

### A.2.5 Preserving the State Mapping Through Updates

Having computed the parameters, I now prove that, given that  $\mu_t = Mx_{t+1}^-$  and  $\Sigma_t = MP_{t+1}^-M^\top + \Phi$ , applying the PLG state update to compute  $\mu_{t+1}$  and  $\Sigma_{t+1}$  and applying the Kalman filter updates to compute  $x_{t+2}^-$  and  $P_{t+2}^-$  will result in  $\mu_{t+1} = Mx_{t+2}^-$  and  $\Sigma_{t+1} = MP_{t+2}^-M^\top + \Phi$ .

The PLG mean update is given by

$$(A.64) \quad \mu_{t+1} = G\mu_t + (G\Sigma_t J^\top + C_\eta^\top J^\top)(J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1}(y_{t+1} - J\mu_t)$$

$$(A.65) \quad = GMx_{t+1}^- + (GMP_{t+1}^-M^\top J^\top + \Omega^\top J^\top)(HP_{t+1}^-H^\top + R)^{-1}(y_{t+1} - JMx_{t+1}^-).$$

Recall that if  $\tau(j) > 1$ , the  $j$ th column of  $J$  is  $\mathbf{0}$ . Since  $\Omega_{ij} = 0$  whenever  $\tau(i) = 1$ ,  $J\Omega = \mathbf{0}$ . Making these substitutions as well as substituting  $MA$  for  $GM$  and  $H$  for  $JM$ :

$$(A.66) \quad \mu[t+1] = MAx_{t+1}^- + (MAP_{t+1}^-H^\top)(HP_{t+1}^-H^\top + R)^{-1}(y_{t+1} - Hx_{t+1}^-)$$

$$(A.67) \quad = Mx_{t+2}^-.$$

To show that the covariance matrix updates also preserve the mapping will require the identity  $\Theta = \Phi + MQM^\top$ . This can be seen by observing that

$$(A.68) \quad \Theta_{ij} = H_{\rho(i),\cdot} A^{\tau(i)-\tau(j)} S_{\tau(j)} H_{\rho(j),\cdot}^\top + \delta_{\tau(i),\tau(j)} R_{\rho(i),\rho(j)}$$

$$(A.69) \quad = H_{\rho(i),\cdot} A^{\tau(i)-\tau(j)} (S_{\tau(j)-1} + A^{\tau(j)-1} Q (A^{\tau(j)-1})^\top) H_{\rho(j),\cdot}^\top + \delta_{\tau(i),\tau(j)} R_{\rho(i),\rho(j)}$$

$$(A.70) \quad = \Phi_{ij} + H_{\rho(i),\cdot} A^{\tau(i)-1} Q (A^{\tau(j)-1})^\top H_{\rho(j),\cdot}^\top.$$

Rewriting this in matrix form results in  $\Theta = \Phi + MQM^\top$ .

Substituting from (A.55) and (A.63) into the PLG covariance update (4.24) results in

$$(A.71) \quad \begin{aligned} \Sigma_{t+1} &= GMP_{t+1}^- A^\top M^\top + G\Phi G^\top + GC_\eta + C_\eta^\top G^\top \\ &\quad + (\Theta - G\Phi G^\top - GC_\eta - C_\eta^\top G^\top) \\ &\quad - MAP_{t+1}^- H^\top (HP_{t+1}^- H^\top + R)^{-1} HP_{t+1}^- A^\top M^\top \end{aligned}$$

$$(A.72) \quad = MA(P_{t+1}^- - P_{t+1}^- H^\top (HP_{t+1}^- H^\top + R)^{-1} HP_{t+1}^-) A^\top M^\top + \Theta$$

$$(A.73) \quad = M [A(P_{t+1}^- - P_{t+1}^- H^\top (HP_{t+1}^- H^\top + R)^{-1} HP_{t+1}^-) A^\top + Q] M^\top + \Phi$$

$$(A.74) \quad = MP_{t+2}^- M^\top + \Phi.$$

Since the distribution of  $Y_{t+1}$  is preserved through the state mapping, and the mapping is preserved through state updates, the result is proven.  $\square$

### A.3 Proof of Lemma 4.3

**Lemma.** *The optimal action  $u_t^*(h_{t-1})$  to be taken at time  $t$  after observing history  $h_{t-1}$  is the certainty-equivalent optimal action and a linear function of the mean vector  $\mu_{t-1}$ .*

Let  $\Psi_t^\mu(h_s)$  be the optimal expected cost-to-go from time  $t$  after observing  $h_s$ , for  $s < t$ :

$$(A.75) \quad \begin{aligned} &\Psi_t^\mu(h_s, u_{s+1}, \dots, u_{t-1}) \\ &= \min_{u_t, \dots, u_T} \mathbb{E} \left[ \sum_{\tau=t}^T \psi^\mu(\mu_{\tau-1}, u_\tau) + \mu_T^\top W_{\mu, T} \mu_T \middle| h_s, u_{s+1}, \dots, u_{t-1} \right] \end{aligned}$$

$$(A.76) \quad = \min_{u_t} \left\{ \mathbb{E}[\psi^\mu(\mu_{t-1}, u_t) | h_s, u_{s+1}, \dots, u_{t-1}] + \Psi_{t+1}^\mu(h_s, u_{s+1}, \dots, u_t) \right\}.$$

This function can be divided into a history-independent constant and a history-dependent part that is quadratic in the state:

$$(A.77) \quad \Psi_t^\mu(h_s, u_{s+1}, \dots, u_{t-1}) = v_{t,s}^\mu + \bar{\mu}_{t-1,s}^\top V_t^\mu \bar{\mu}_{t-1,s},$$

where  $v_{t,s}^\mu$  and  $V_t^\mu$  are constants and  $\bar{\mu}_{t,s} = \mathbb{E}[\mu_t | h_s, u_{s+1}, \dots, u_t]$ . This is proven by induction.

In the base case,

$$(A.78) \quad \Psi_{T+1}^\mu(h_s, u_{s+1}, \dots, u_T) = \mathbb{E}[\mu_T^\top W_{\mu,T} \mu_T | h_s, u_{s+1}, \dots, u_T]$$

$$(A.79) \quad = \bar{\mu}_{T,s}^\top W_{\mu,T} \bar{\mu}_{T,s} + \text{tr}(W_{\mu,T} \text{Var}[\mu_T | h_s, u_{s+1}, \dots, u_T]).$$

The second equality follows because  $\mathbb{E}[X^\top W X] = \mathbb{E}[X^\top] W \mathbb{E}[X] + \text{tr}(W \text{Var}[X])$  whenever  $W$  is a symmetric constant matrix (as  $W_{\mu,T}$  is). Thus  $V_{T+1}^\mu = W_{\mu,T}$  and  $v_{T+1,s}^\mu = \text{tr}(W_{\mu,T} \text{Var}[\mu_T | h_s, u_{s+1}, \dots, u_T])$ .

Assuming that  $\Psi_{t+1}^\mu(h_s, u_{s+1}, \dots, u_t) = v_{t+1,s}^\mu + \bar{\mu}_{t,s}^\top V_{t+1}^\mu \bar{\mu}_{t,s}$ , it follows that

$$(A.80) \quad \begin{aligned} & \Psi_t^\mu(h_s, u_{s+1}, \dots, u_{t-1}) = \\ & = \min_{u_t} \{ \mathbb{E}[\psi^\mu(\mu_{t-1}, u_t) | h_s, u_{s+1}, \dots, u_{t-1}] + \Psi_{t+1}^\mu(h_s, u_{s+1}, \dots, u_t) \} \end{aligned}$$

$$(A.81) \quad \begin{aligned} & = \min_{u_t} \{ \mathbb{E}[\mu_{t-1}^\top W_\mu \mu_{t-1} + 2u_t^\top W_{\mu,u} \mu_{t-1} + u_t^\top W_u u_t | h_s, u_{s+1}, \dots, u_{t-1}] \\ & \quad + v_{t+1,s}^\mu + \bar{\mu}_{t,s}^\top V_{t+1}^\mu \bar{\mu}_{t,s} \} \end{aligned}$$

$$(A.82) \quad \begin{aligned} & = \bar{\mu}_{t-1,s}^\top W_\mu \bar{\mu}_{t-1,s} + \text{tr}(W_\mu \text{Var}[\mu_{t-1} | h_s, u_{s+1}, \dots, u_{t-1}]) \\ & \quad + v_{t+1,s}^\mu + \min_{u_t} \{ 2u_t^\top W_\mu \bar{\mu}_{t-1,s} + u_t^\top W_u u_t \\ & \quad \quad + (G \bar{\mu}_{t-1,s} + L u_t)^\top V_{t+1}^\mu (G \bar{\mu}_{t-1,s} + L u_t) \} \end{aligned}$$

$$(A.83) \quad \begin{aligned} & = \bar{\mu}_{t-1,s}^\top (W_\mu + G^\top V_{t+1}^\mu G) \bar{\mu}_{t-1,s} + \text{tr}(W_\mu \text{Var}[\mu_{t-1} | h_s, u_{s+1}, \dots, u_{t-1}]) \\ & \quad + v_{t+1,s}^\mu + \min_{u_t} \{ u_t^\top (2W_{\mu,u} + L^\top V_{t+1}^\mu G) \bar{\mu}_{t-1,s} \\ & \quad \quad + \bar{\mu}_{t-1,s}^\top (L^\top V_{t+1}^\mu G) u_t + u_t^\top (W_u + L^\top V_{t+1}^\mu L) u_t \}. \end{aligned}$$

Because  $W_u + L^\top V_{t+1}^\mu L$  is symmetric positive definite, the minimization in the

last line can be determined by taking the derivative and setting it to 0:

$$\frac{\partial}{\partial u_t} u_t^\top (2W_{\mu,u} + L^\top V_{t+1}^\mu G) \bar{\mu}_{t-1,s} + \bar{\mu}_{t-1,s}^\top (L^\top V_{t+1}^\mu G) u_t + u_t^\top (W_u + L^\top V_{t+1}^\mu L) u_t =$$

$$(A.84) \quad = 2(W_{\mu,u} + L^\top V_{t+1}^\mu G) \bar{\mu}_{t-1,s} + 2(W_u + L^\top V_{t+1}^\mu L) u_t$$

$$(A.85) \quad = \mathbf{0}$$

$$(A.86) \quad \therefore \quad u_t^* = -(W_u + L^\top V_{t+1}^\mu L)^{-1} (W_{\mu,u} + L^\top V_{t+1}^\mu G) \bar{\mu}_{t-1,s}$$

$$(A.87) \quad \triangleq \Pi_t^\mu \bar{\mu}_{t-1,s}.$$

Plugging this result into (A.83) yields

$$(A.88)$$

$$\begin{aligned} \Psi_t^\mu(h_s, u_{s+1}, \dots, u_{t-1}) &= \bar{\mu}_{t-1,s}^\top (W_\mu + G^\top V_{t+1}^\mu G - (W_{\mu,u}^\top + G^\top V_{t+1}^\mu L)(W_u + L^\top V_{t+1}^\mu L)^{-1} \\ &\quad \cdot (W_{\mu,u} + L^\top V_{t+1}^\mu G)) \bar{\mu}_{t-1,s} + \text{tr}(W_\mu \text{Var}[\mu_{t-1} | h_s, u_{s+1}, \dots, u_{t-1}]) + v_{t+1,s}^\mu. \end{aligned}$$

Thus,

$$(A.89) \quad W_\mu + G^\top V_{t+1}^\mu G - (W_{\mu,u}^\top + G^\top V_{t+1}^\mu L)(W_u + L^\top V_{t+1}^\mu L)^{-1} (W_{\mu,u} + L^\top V_{t+1}^\mu G)$$

and

$$(A.90) \quad v_{t,s}^\mu = \text{tr}(W_\mu \text{Var}[\mu_{t-1} | h_s, u_{s+1}, \dots, u_{t-1}]) + v_{t+1,s}^\mu.$$

Note that  $v_{t,s}^\mu$  depends only on the values of  $t$  and  $s$ , and *not* on the values of any observations or actions. By induction, for all  $t, s$  such that  $s < t$ ,  $\Psi_t^\mu(h_s) = v_{t,s}^\mu + \bar{\mu}_{t-1,s}^\top V_t^\mu \bar{\mu}_{t-1,s}$ . Note that  $\bar{\mu}_{t-1,t-1} = \mu_{t-1}$ . Therefore, the optimal action at time  $t$  is given by  $u_t^*(h_{t-1}) = \Pi_t^\mu \mu_{t-1}$ , i.e. the optimal action at time  $t$  is a linear function of the model's state at  $t$ .

#### A.4 Proof of Theorem 4.4

**Theorem.** *For any  $n$ -dimensional, full-rank linear-quadratic Gaussian model, an equivalent  $n$ -dimensional PLGQ exists that, given any history of interaction with the*

system, computes the same optimal action.

By Theorem 4.2, any  $n$ -dimensional LDS has an equivalent PLG. That means that it only remains to prove that an equivalent cost function exists; this is proven by construction.

Since the LQG is full-rank,  $\mathcal{M}_n$  is invertible, and thus so is  $M$  (see Appendix A.2 for a definition of these matrices). The following identities thus obtain:

$$(A.91) \quad x_{t+1}^- = M^{-1}\mu_{t-1}, \quad B = M^{-1}L, \quad \text{and} \quad M^{-1}G = AM^{-1}.$$

The cost-function parameters of the PLGQ are derived from the LQG's cost function parameters as follows:

$$(A.92) \quad W_{\mu,T} = M^{-\top}W_{x,T}M^{-1}, \quad W_{\mu} = M^{-\top}W_xM^{-1},$$

$$(A.93) \quad W_{\mu,u} = W_{x,u}M^{-1}.$$

The action cost matrix  $W_u$  is the same in the PLGQ and LQG.

To prove that both models select the same optimal action, I first show by induction that the Riccati recursion matrices of the two models are related by  $V_t^\mu = M^{-\top}V_t^xM^{-1}$ . In the base case,

$$(A.94) \quad V_{T+1}^\mu = W_{\mu,T}$$

$$(A.95) \quad = M^{-\top}W_{x,T}M^{-1}$$

$$(A.96) \quad = M^{-\top}V_{T+1}^xM^{-1},$$

so this property holds. Now, assuming that  $V_{t+1}^\mu = M^{-\top}V_{t+1}^xM^{-1}$ , it follows that

$$(A.97)$$

$$\begin{aligned} V_t^\mu &= W_\mu + G^\top V_{t+1}^\mu G \\ &\quad - (W_{\mu,u}^\top + G^\top V_{t+1}^\mu L)(W_u + L^\top V_{t+1}^\mu L)^{-1}(W_{\mu,u} + L^\top V_{t+1}^\mu G) \end{aligned}$$

$$\begin{aligned}
\text{(A.98)} \quad &= M^{-\top} W_x M^{-1} + G^\top M^{-\top} V_{t+1}^x M^{-1} G \\
&\quad - (M^{-\top} W_{x,u}^\top + G M^{-\top} V_{t+1}^x M^{-1} L) (W_u + L^\top M^{-\top} V_{t+1}^x M^{-1} L)^{-1} \\
&\quad \cdot (W_{x,u} M^{-1} + L^\top M^{-\top} V_{t+1}^x M^{-1} G)
\end{aligned}$$

$$\begin{aligned}
\text{(A.99)} \quad &= M^{-\top} [W_x + A^\top V_{t+1}^x A \\
&\quad - (W_{x,u}^\top + A^\top V_{t+1}^x B) (W_u + B^\top V_{t+1}^x B)^{-1} (W_{x,u} + B^\top V_{t+1}^x A)] M^{-1}
\end{aligned}$$

$$\text{(A.100)} \quad = M^{-\top} V_t^x M^{-1}.$$

Thus, by induction,  $V_t^\mu = M^{-\top} V_t^x M^{-1}$  for all  $t$ . Plugging this into the action selection equation results in

$$\text{(A.101)}$$

$$u_t^*(h_{t-1}) = \Pi_t^\mu \mu_{t-1}$$

$$\text{(A.102)} \quad = -(W_u + L^\top V_{t+1}^\mu L)^{-1} (W_{\mu,u} + L^\top V_{t+1}^\mu G) \mu_{t-1}$$

$$\text{(A.103)} \quad = -(W_u + L^\top M^{-\top} V_{t+1}^x M^{-1} L)^{-1} (W_{x,u} M^{-1} + L^\top M^{-\top} V_{t+1}^x M^{-1} G) M x_t^-$$

$$\text{(A.104)} \quad = -(W_u + B^\top V_{t+1}^x B)^{-1} (W_{x,u} + B^\top V_{t+1}^x A) M^{-1} M x_t^-$$

$$\text{(A.105)} \quad = \Pi_t^x x_t^-.$$

That is, both models select the same action, proving the theorem.

## A.5 Proof of Theorem 5.1

**Theorem.** *If a dynamical system can be modeled by an  $n$ -dimensional PLG, is controlled by a policy that is jointly CE-learnable with the system, and generates a training set whose  $K$  trajectories are each at least  $n + (l + 1)\tau_{\max}$  time steps long, then, as the number of trajectories  $K$  grows, the parameter estimates computed by the multiple-trajectory CE algorithm from this training set will converge in probability to the true parameters of that PLG.*

This proof will make use of some of the material in Appendix B.1; in particular, the definition of convergence in probability (Definition B.1), the Weak Law of Large Numbers (Theorem B.2), and Theorem B.4.

That the estimates for  $G, \Gamma_1, \dots, \Gamma_{\tau_{\max}}$  are consistent was proven in the text (which assumes that  $\Xi\Xi^\top$  is invertible), where it was shown that the noise terms converge in probability to 0 when averaged together, thus leaving an equation in which all the elements are known except  $\gamma$ . That  $\Xi\Xi^\top$  is invertible is guaranteed by the conditions in the theorem, namely that the trajectories are at least  $n + (l+1)\tau_{\max}$  time steps long, that the policy used to generate the data set is jointly CE-learnable with the system.

Similarly,  $\hat{J}$  was shown to be a consistent estimator for  $J$ . Since  $Z_{\text{next}}$  is a sub-matrix of  $\Xi$ ,  $Z_{\text{next}}Z_{\text{next}}^\top$  is invertible whenever  $\Xi\Xi^\top$  is; in particular, it is invertible under the conditions of this theorem.

The text showed that  $\hat{z}_{0|0}^k$  was an unbiased estimate of  $Z_0|u_{t+1} = \mathbf{0}, \dots$ . A simple application of the Weak Law proves that  $\hat{\mu}_0$  is consistent.

The estimate of  $\Sigma_0$  also makes use of the  $\hat{z}_{0|0}^k$  samples:

$$(A.106) \quad \hat{\Sigma}_0 = \frac{1}{K-1} \sum_{k=1}^K (\hat{z}_{0|0}^k - \hat{\mu}_0)(\hat{z}_{0|0}^k - \hat{\mu}_0)$$

Applying Theorem B.4:

$$(A.107) \quad \xrightarrow{p} \frac{1}{K-1} \sum_{k=1}^K (\hat{z}_{0|0}^k - \mathbb{E}[Z_0|u_{t+1} = \mathbf{0}, \dots])(\hat{z}_{0|0}^k - \mathbb{E}[Z_0|u_{t+1} = \mathbf{0}, \dots])^\top$$

And now the Weak Law of Large Numbers:

$$(A.108) \quad \xrightarrow{p} \mathbb{E}[(Z_0 - \mathbb{E}[Z_0|u_{t+1} = \mathbf{0}, \dots])(Z_0 - \mathbb{E}[Z_0|u_{t+1} = \mathbf{0}, \dots])^\top]$$

$$(A.109) \quad = \Sigma_0.$$

Thus  $\widehat{\Sigma}_0$  is consistent.

In order to show the the estimates of the noise parameters are consistent, it is necessary to show that  $\widehat{\eta}_{t+1}^k$  is a consistent estimator for  $\eta_{t+1}$  and that  $\eta_{t+1}$  has the same mean, variance, and covariance with  $Z_t$  as  $\eta_{t+1}|h_t$  has.

The consistency of  $\widehat{\eta}_{t+1}^k$  follows from the consistency of  $\widehat{G}, \widehat{\Gamma}_1, \dots, \widehat{\Gamma}_{\tau_{\max}}$ :

$$(A.110) \quad \widehat{\eta}_{t+1}^k = z_{t+1}^k - \widehat{G}z_t^k - \sum_{i=1}^{\tau_{\max}} \widehat{\Gamma}_i u_{t+i}^k$$

$$(A.111) \quad \xrightarrow{p} z_{t+1}^k - Gz_t^k - \sum_{i=1}^{\tau_{\max}} \Gamma_i u_{t+i}^k \quad \text{by Theorem B.4}$$

$$(A.112) \quad = \eta_{t+1}^k.$$

As for the distribution of  $\eta_{t+1}$  versus  $\eta_{t+1}|h_t$ , recall that  $E_{h_t}$  and  $\text{Var}_{h_t}$  denote expectation and variance, respectively, taken over the distribution of histories of length  $t$ . Then, by the Law of Total Expectation (Theorem B.5),

$$(A.113) \quad E[\eta_{t+1}] = E_{h_t}[E[\eta_{t+1}|h_t]] = \mathbf{0} = E[\eta_{t+1}|h_t].$$

By the Law of Total Variance (Theorem B.6),

$$(A.114) \quad \text{Var}[\eta_{t+1}] = E_{h_t}[\text{Var}[\eta_{t+1}|h_t]] + \text{Var}_{h_t}[E[\eta_{t+1}|h_t]]$$

$$(A.115) \quad = E_{h_t}[\Sigma_\eta] + \text{Var}_{h_t}[\mathbf{0}]$$

$$(A.116) \quad = \Sigma_\eta = \text{Var}[\eta_{t+1}|h_t],$$

and by the Law of Total Covariance (Theorem B.7),

$$(A.117) \quad \text{Cov}[Z_t, \eta_{t+1}] = E_{h_t}[\text{Cov}[Z_t, \eta_{t+1}|h_t]] + \text{Cov}[E[Z_t|h_t], E[\eta_{t+1}^\top|h_t]]$$

$$(A.118) \quad = E_{h_t}[C_\eta] + \text{Cov}[\mu_t, \mathbf{0}]$$

$$(A.119) \quad = C_\eta = \text{Cov}[Z_t, \eta_{t+1}|h_t].$$

Thus  $\eta_{t+1}$  has the same distribution as  $\eta_{t+1}|h_t$ , and statistics of  $\hat{\eta}$  will converge in probability to statistics of  $\eta_{t+1}|h_t$ . In particular, this implies that  $\hat{C}_\eta$  and  $\hat{\Sigma}_\eta$  are consistent estimated of  $C_\eta$  and  $\Sigma_\eta$ , respectively.

Finally, the consistency of  $\hat{\Sigma}_{\text{adj}}$  was proven in the text.

## APPENDIX B

### Reference Material

#### B.1 Large Sample Behavior

Large sample theory is a topic in statistics that describes the behavior of estimators as the amount of data they are based on becomes very large. An important concept here is *convergence in probability*, which is defined as follows:

**Definition B.1** (Convergence in Probability). The sequence  $\hat{x}_1, \hat{x}_2, \dots$  converges to  $x$  in probability if

$$\lim_{n \rightarrow \infty} \Pr(|\hat{x}_n - x| > \delta) = 0$$

for all positive  $\delta$ . This is denoted  $\hat{x}_n \xrightarrow{p} x$  as  $n \rightarrow \infty$ .

Essentially, this is equivalent to saying that the probability that the error of an estimator exceeds some bound  $\delta$  goes to zero as the size of the sample increases. A very important result regarding convergence in probability is the weak law of large numbers, which states that the average of independent variables with the same expected value will converge in probability to that expected value.

**Theorem B.2** (Weak Law of Large Numbers). *If  $X_1, X_2, \dots$  are independent variables with mean  $\mu$  and bounded variance, and*

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i,$$

then  $\bar{X}_n \xrightarrow{p} \mu$  as  $n \rightarrow \infty$ .

The proof of Theorem B.2 is based on Chebyshev's Inequality:

**Lemma B.3. Chebyshev's Inequality** For any random variable  $X$  and any constant  $a > 0$ ,

$$\Pr(|X| \geq a) \leq \frac{\mathbf{E}[X^2]}{a^2}.$$

*Proof.* For any value of  $a > 0$  and  $X$ ,

$$(B.1) \quad \mathbf{I}\{|X| \geq a\} \leq \frac{X^2}{a^2},$$

where  $\mathbf{I}\{A\}$  is the indicator function that is 1 when event  $A$  is true, and 0 otherwise. Since  $A$  is a random event,  $\mathbf{I}\{A\}$  is a random variable; its expectation is the probability of  $A$  occurring. So, taking the expectation of both sides of (B.1),

$$(B.2) \quad \Pr(|X| \geq a) \leq \frac{\mathbf{E}[X^2]}{a^2},$$

which is the result. □

Theorem B.2 is a direct application of this lemma.

An estimator that converges in probability to the value it is estimating is called a *consistent* estimator. An interesting property of consistent estimators is that evaluating a function  $f$  of a consistent estimator of  $x$  results in a consistent estimator of  $f(x)$ .

**Theorem B.4.** If  $\hat{x}_n \xrightarrow{p} x$  as  $n \rightarrow \infty$ , and  $f$  is continuous at  $x$ , then  $f(\hat{x}_n) \xrightarrow{p} f(x)$  as  $n \rightarrow \infty$ .

*Proof.* This proof again depends on the Chebyshev inequality. Because  $f$  is continuous at  $x$ , for any positive  $\epsilon$  there is a positive  $\delta$  that satisfies

$$(B.3) \quad |\hat{x}_n - x| < \delta \quad \implies \quad |f(\hat{x}_n) - f(x)| < \epsilon.$$

This implies that

$$(B.4) \quad \Pr(|\hat{x}_n - x| < \delta) \leq \Pr(|f(\hat{x}_n) - f(x)| < \epsilon).$$

It then follows that

$$(B.5) \quad \Pr(|f(\hat{x}_n) - f(x)| \geq \epsilon) \leq \Pr(|\hat{x}_n - x| \geq \delta).$$

Because  $\hat{x}_n \xrightarrow{p} x$ , the right-hand side goes to 0 as  $n \rightarrow \infty$ ; since the left-hand side is upper-bounded by the right-hand side,  $f(\hat{x}_n) \xrightarrow{p} f(x)$ .  $\square$

Theorem B.4 is used extensively in Chapter V to derive consistent estimators from consistent estimators of *other* parameters.

## B.2 Changing the Conditions of Expectations and Variances

**Theorem B.5** (Law of Total Expectation).

$$E[X] = E_Y[E[X|Y]],$$

where  $E_Y$  denotes the expectation over the distribution of  $Y$ .

*Proof.*

$$\begin{aligned} E_Y[E[X|Y]] &= \sum_y E[X|Y = y] \Pr(Y = y) && \text{(Defn. of Expectation)} \\ &= \sum_y \left( \sum_x x \Pr(X = x|Y = y) \right) \Pr(Y = y) && \text{(Defn. of Expectation)} \\ &= \sum_y \sum_x x \Pr(X = x|Y = y) \Pr(Y = y) \\ &= \sum_y \sum_x x \Pr(Y = y|X = x) \Pr(X = x) && \text{(Bayes' Law)} \\ &= \sum_x x \Pr(X = x) \left( \sum_y \Pr(Y = y|X = x) \right) \end{aligned}$$

$$\begin{aligned}
&= \sum_x x \Pr(X = x) \\
&= \mathbb{E}[X].
\end{aligned}$$

□

**Theorem B.6** (Law of Total Variance).

$$\text{Var}[X] = \mathbb{E}_Y[\text{Var}[X|Y]] + \text{Var}_Y[\mathbb{E}[X|Y]],$$

where  $\mathbb{E}_Y$  denotes the expectation over the distribution of  $Y$ , and  $\text{Var}_Y$  denotes the variance over the same distribution.

*Proof.*

$$\begin{aligned}
&\text{Var}[X] \\
&= \mathbb{E}[XX^\top] - \mathbb{E}[X] \mathbb{E}[X^\top] \\
&= \mathbb{E}_Y[\mathbb{E}[XX^\top|Y]] - \mathbb{E}_Y[\mathbb{E}[X|Y]] \mathbb{E}_Y[\mathbb{E}[X^\top|Y]] && \text{Law of Total Exp.} \\
&= \mathbb{E}_Y[\mathbb{E}[XX^\top|Y]] - \mathbb{E}_Y[\mathbb{E}[X|Y] \mathbb{E}[X^\top|Y]] \\
&\quad + \mathbb{E}_Y[\mathbb{E}[X|Y] \mathbb{E}[X^\top|Y]] - \mathbb{E}_Y[\mathbb{E}[X|Y]] \mathbb{E}_Y[\mathbb{E}[X^\top|Y]] \\
&= \mathbb{E}_Y[\text{Var}[X|Y]] + \text{Var}_Y[\mathbb{E}[X|Y]].
\end{aligned}$$

□

**Theorem B.7** (Law of Total Covariance).

$$\text{Cov}[X, Y] = \mathbb{E}_Z[\text{Cov}[X, Y|Z]] + \text{Cov}[\mathbb{E}[X|Z], \mathbb{E}[Y|Z]],$$

where  $\mathbb{E}_Z$  denotes the expectation over the distribution of  $Z$ .

*Proof.*

$$\begin{aligned}
\text{Cov}[X, Y] &= \text{E}[XY^\top] - \text{E}[X] \text{E}[Y^\top] \\
&= \text{E}_Z[\text{E}[XY^\top | Z]] - \text{E}_Z[\text{E}[X | Z]] \text{E}_Z[\text{E}[Y^\top | Z]] && \text{Law of Total Exp.} \\
&= \text{E}_Z[\text{E}[XY^\top | Z]] - \text{E}_Z[\text{E}[X | Z] \text{E}[Y^\top | Z]] \\
&\quad + \text{E}_Z[\text{E}[X | Z] \text{E}[Y^\top | Z]] - \text{E}_Z[\text{E}[X | Z]] \text{E}_Z[\text{E}[Y^\top | Z]] \\
&= \text{E}_Z[\text{Cov}[X, Y | Z]] + \text{Cov}[\text{E}[X | Z], \text{E}[Y | Z]] && \square
\end{aligned}$$

## APPENDIX C

## The Gradient of the Log-Likelihood Function

The log likelihood function

$$(C.1) \quad \ell(\theta; h_N^K) = \log \mathcal{L}(\theta; h_N^K)$$

$$(C.2) \quad \begin{aligned} &= -\frac{NKm \log 2\pi}{2} \\ &\quad - \frac{1}{2} \sum_{t=0}^{N-1} (K \log |J\Sigma_t J^\top + \Sigma_{\text{adj}}|) \\ &\quad - \frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k). \end{aligned}$$

must be differentiated with respect to the parameters  $J$ ,  $G$ ,  $C_\eta$ ,  $\Sigma_\eta$ ,  $\Sigma_{\text{adj}}$ ,  $\mu_0$ , and  $\Sigma_0$ . For each parameter  $\vartheta$ , the gradient  $\frac{\partial \ell(h_N^K; \theta)}{\partial \vartheta}$  will be written in terms of the data, the parameters, and the partial derivatives  $\frac{\partial \mu_t^k}{\partial \vartheta}$  and  $\frac{\partial \Sigma_t}{\partial \vartheta}$ ; these partial derivatives can be computed recursively, initializing the recursion with  $\frac{\partial \mu_0^k}{\partial \vartheta}$  and  $\frac{\partial \Sigma_0}{\partial \vartheta}$ .

The gradient with respect to  $J$  is given by

(C.3)

$$\begin{aligned}
\frac{\partial \ell(h_N^K; \theta)}{\partial J^{ij}} = & -\frac{K}{2} \sum_{t=0}^{N-1} \text{tr} \left[ (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (e_{m,i} e_{n,j}^\top \Sigma_t J^\top + J \frac{\partial \Sigma_t}{\partial J^{ij}} J^\top + J \Sigma_t e_{n,j} e_{m,i}^\top) \right] \\
& -\frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K \left\{ -(e_{m,i} e_{n,j}^\top \mu_t^k + J \frac{\partial \mu_t^k}{\partial J^{ij}})^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \right. \\
& \quad - (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (e_{m,i} e_{n,j}^\top \Sigma_t J^\top + J \frac{\partial \Sigma_t}{\partial J^{ij}} J^\top + J \Sigma_t e_{n,j} e_{m,i}^\top) \\
& \quad \cdot (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\
& \quad \left. - (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (e_{m,i} e_{n,j}^\top \mu_t^k + J \frac{\partial \mu_t^k}{\partial J^{ij}}) \right\},
\end{aligned}$$

where  $e_{n,i}$  is the  $i$ th column of the  $n \times n$  identity matrix. The recursions for the partials of the state variables are initialized with

$$(C.4) \quad \frac{\partial \mu_0^k}{\partial J^{ij}} = \mathbf{0} \quad \text{and} \quad \frac{\partial \Sigma_0}{\partial J^{ij}} = \mathbf{0}.$$

The recursion for  $\frac{\partial \mu_t^k}{\partial J^{ij}}$ :

(C.5)

$$\begin{aligned}
\frac{\partial \mu_{t+1}^k}{\partial J^{ij}} = & G \frac{\partial \mu_t^k}{\partial J^{ij}} + G \frac{\partial \Sigma_t}{\partial J^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\
& + (G\Sigma_t + C_\eta^\top) e_{n,j} e_{m,i}^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\
& - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \\
& \quad \cdot (e_{m,i} e_{n,j}^\top \Sigma_t J^\top + J \frac{\partial \Sigma_t}{\partial J^{ij}} J^\top + J \Sigma_t e_{n,j} e_{m,i}^\top) (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\
& - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}}) (e_{m,i} e_{n,j}^\top \mu_t^k + J \frac{\partial \mu_t^k}{\partial J^{ij}}).
\end{aligned}$$

And for  $\frac{\partial \Sigma_t}{\partial J^{ij}}$ :

(C.6)

$$\begin{aligned}
& \frac{\partial \Sigma_{t+1}}{\partial J^{ij}} \\
&= G \frac{\partial \Sigma_t}{\partial J^{ij}} G^\top - G \frac{\partial \Sigma_t}{\partial J^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
&\quad - (G \Sigma_t + C_\eta^\top) e_{n,j} e_{m,i}^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
&\quad + (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \\
&\quad \quad \cdot (e_{m,i} e_{n,j}^\top \Sigma_t J^\top + J \frac{\partial \Sigma_t}{\partial J^{ij}} J^\top + J \Sigma_t e_{n,j} e_{m,i}^\top) (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
&\quad - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} e_{m,i} e_{n,j}^\top (\Sigma_t G^\top + C_\eta) \\
&\quad - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial J^{ij}} G^\top.
\end{aligned}$$

The computation for  $G$  is similar:

(C.7)

$$\begin{aligned}
& \frac{\partial \ell(h_N^K; \theta)}{\partial G^{ij}} = \\
&\quad - \frac{K}{2} \sum_{t=0}^{N-1} \text{tr} \left[ (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial G^{ij}} J^\top \right] \\
&\quad - \frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K \left\{ - \left( J \frac{\partial \mu_t^k}{\partial G^{ij}} \right)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \right. \\
&\quad \quad - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \left( J \frac{\partial \Sigma_t}{\partial G^{ij}} J^\top \right) (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\
&\quad \quad \left. - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial G^{ij}} \right\}.
\end{aligned}$$

The recursions are initialized by

$$\text{(C.8)} \quad \frac{\partial \mu_0^k}{\partial G^{ij}} = \mathbf{0} \quad \text{and} \quad \frac{\partial \Sigma_0}{\partial G^{ij}} = \mathbf{0}.$$

Recurring on  $\frac{\partial \mu_t^k}{\partial G^{ij}}$ :

(C.9)

$$\begin{aligned} \frac{\partial \mu_{t+1}^k}{\partial G^{ij}} = & e_{n,i} e_{n,j}^\top \mu_t^k + G \frac{\partial \mu_t^k}{\partial G^{ij}} + (e_{n,i} e_{n,j}^\top \Sigma_t + G \frac{\partial \Sigma_t}{\partial G^{ij}}) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\ & - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial G^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\ & - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial G^{ij}}. \end{aligned}$$

Now recurring on  $\frac{\partial \Sigma_t}{\partial G^{ij}}$ :

(C.10)

$$\begin{aligned} \frac{\partial \Sigma_{t+1}}{\partial G^{ij}} = & e_{n,i} e_{n,j}^\top \Sigma_t G^\top + G \frac{\partial \Sigma_t}{\partial G^{ij}} G^\top + G \Sigma_t e_{n,j} e_{n,i}^\top + e_{n,i} e_{n,j}^\top C_\eta + C_\eta^\top e_{n,j} e_{n,i}^\top \\ & - (e_{n,i} e_{n,j}^\top \Sigma_t + G \frac{\partial \Sigma_t}{\partial G^{ij}}) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\ & + (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial G^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\ & - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\frac{\partial \Sigma_t}{\partial G^{ij}} G^\top + \Sigma_t e_{n,j} e_{n,i}^\top). \end{aligned}$$

Next is the gradient with respect to  $L$ . Since  $L$  is not involved in the computation of  $\Sigma_{t+1}$ ,  $\frac{\partial \Sigma_t}{\partial L^{ij}} = \mathbf{0}$  for all  $t$ .

$$(C.11) \quad \frac{\partial \ell(h_N^K; \theta)}{\partial L^{ij}} = -\frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K \left\{ -\left( J \frac{\partial \mu_t^k}{\partial L^{ij}} \right)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \right. \\ \left. - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \left( J \frac{\partial \mu_t^k}{\partial L^{ij}} \right) \right\}.$$

Differentiating  $\mu_{t+1}^k$  obtains

$$(C.12) \quad \frac{\partial \mu_{t+1}^k}{\partial L^{ij}} = G \frac{\partial \mu_t^k}{\partial L^{ij}} + e_{n,i} e_{l,j}^\top u_t^k - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial L^{ij}};$$

the recursion is initialized by

$$(C.13) \quad \frac{\partial \mu_0^k}{\partial L^{ij}} = \mathbf{0}.$$

The gradient with respect to  $C_\eta$  is computed by

(C.14)

$$\begin{aligned} \frac{\partial \ell(h_N^K; \theta)}{\partial C_\eta^{ij}} = & -\frac{K}{2} \sum_{k=1}^K \text{tr} \left[ (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} J^\top \right] \\ & -\frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K \left\{ -\left( J \frac{\partial \mu_t^k}{\partial C_\eta^{ij}} \right)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \right. \\ & \quad - (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\ & \quad \left. - (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial C_\eta^{ij}} \right\}, \end{aligned}$$

with the recursions

(C.15)

$$\begin{aligned} \frac{\partial \mu_{t+1}^k}{\partial C_\eta^{ij}} = & G \frac{\partial \mu_t^k}{\partial C_\eta^{ij}} + \left( G \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} + e_{n,j} e_{n,i}^\top \right) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\ & - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\ & - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial C_\eta^{ij}} \end{aligned}$$

and

(C.16)

$$\begin{aligned} \frac{\partial \Sigma_{t+1}}{\partial C_\eta^{ij}} = & G \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} G^\top + G e_{n,i} e_{n,j}^\top + e_{n,j} e_{n,i}^\top G^\top \\ & - \left( G \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} + e_{n,j} e_{n,i}^\top \right) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\ & + (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\ & - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \left( \frac{\partial \Sigma_t}{\partial C_\eta^{ij}} G^\top + e_{n,i} e_{n,j}^\top \right). \end{aligned}$$

These are initialized by

$$(C.17) \quad \frac{\partial \mu_0^k}{\partial C_\eta^{ij}} = \mathbf{0} \quad \text{and} \quad \frac{\partial \Sigma_0}{\partial C_\eta^{ij}} = \mathbf{0}.$$

Now  $\Sigma_\eta$ :

$$(C.18) \quad \begin{aligned} \frac{\partial \ell(h_N^K; \theta)}{\partial \Sigma_\eta^{ij}} = & \\ & - \frac{K}{2} \sum_{t=0}^{N-1} \text{tr} \left[ (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} J^\top \right] \\ & - \frac{1}{2} \sum_{k=1}^K \sum_{t=0}^{N-1} \left\{ - \left( J \frac{\partial \mu_t^k}{\partial \Sigma_\eta^{ij}} \right)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \right. \\ & \quad - (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\ & \quad \left. - (y_{t+1}^k - J\mu_t^k)^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \Sigma_\eta^{ij}} \right\}. \end{aligned}$$

The recursions are initialized with

$$(C.19) \quad \frac{\partial \mu_0^k}{\partial \Sigma_\eta^{ij}} = \mathbf{0} \quad \text{and} \quad \frac{\partial \Sigma_0}{\partial \Sigma_\eta^{ij}} = \mathbf{0}.$$

The recursion on  $\frac{\partial \mu_t^k}{\partial \Sigma_\eta^{ij}}$  is computed by

$$(C.20) \quad \begin{aligned} \frac{\partial \mu_{t+1}^k}{\partial \Sigma_\eta^{ij}} = & \\ & G \frac{\partial \mu_t^k}{\partial \Sigma_\eta^{ij}} + G \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\ & - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J\mu_t^k) \\ & - (G\Sigma_t + C_\eta^\top) J^\top (J\Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \Sigma_\eta^{ij}}. \end{aligned}$$

The recursion of  $\frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}}$  follows

$$\begin{aligned}
\text{(C.21)} \quad & \frac{\partial \Sigma_{t+1}}{\partial \Sigma_\eta^{ij}} = \\
& G \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} G^\top + e_{n,i} e_{n,j}^\top + e_{n,j} e_{n,i}^\top - e_{n,i} e_{n,j}^\top e_{n,i} e_{n,j}^\top \\
& - G \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
& + (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
& - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_\eta^{ij}} G^\top.
\end{aligned}$$

The gradient of the likelihood with respect the the variance adjustment  $\Sigma_{\text{adj}}$  is given by

$$\begin{aligned}
\text{(C.22)} \quad & \frac{\partial \ell(h_N^K; \theta)}{\partial \Sigma_{\text{adj}}^{ij}} = \\
& - \frac{K}{2} \sum_{t=0}^{N-1} \text{tr} \left[ (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \right. \\
& \quad \left. \cdot \left( J \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} J^\top + e_{m,i} e_{m,j}^\top + e_{m,j} e_{m,i}^\top - e_{m,i} e_{m,j}^\top e_{m,i} e_{m,j}^\top \right) \right] \\
& - \frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K \left\{ - \left( J \frac{\partial \mu_t^k}{\partial \Sigma_{\text{adj}}^{ij}} \right)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \right. \\
& \quad - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \\
& \quad \cdot \left( J \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} J^\top + e_{m,i} e_{m,j}^\top + e_{m,j} e_{m,i}^\top - e_{m,i} e_{m,j}^\top e_{m,i} e_{m,j}^\top \right) \\
& \quad \cdot (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\
& \quad \left. - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \Sigma_{\text{adj}}^{ij}} \right\}.
\end{aligned}$$

The recursions are initialized by

$$\text{(C.23)} \quad \frac{\partial \mu_0^k}{\partial \Sigma_{\text{adj}}^{ij}} = \mathbf{0} \quad \text{and} \quad \frac{\partial \Sigma_0}{\partial \Sigma_{\text{adj}}^{ij}} = \mathbf{0},$$

and continued with

$$\begin{aligned}
(C.24) \quad \frac{\partial \mu_{t+1}^k}{\partial \Sigma_{\text{adj}}^{ij}} = & \\
& G \frac{\partial \mu_t^k}{\partial \Sigma_{\text{adj}}^{ij}} + G \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\
& - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \\
& \cdot \left( J \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} J^\top + e_{m,i} e_{m,j}^\top + e_{m,j} e_{m,i}^\top - e_{m,i} e_{m,j}^\top e_{m,i} e_{m,j}^\top \right) \\
& \cdot (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\
& - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \Sigma_{\text{adj}}^{ij}}.
\end{aligned}$$

and

$$\begin{aligned}
(C.25) \quad \frac{\partial \Sigma_{t+1}}{\partial \Sigma_{\text{adj}}^{ij}} = & \\
& G \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} G^\top - G \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
& + (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} \\
& \cdot \left( J \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} J^\top + e_{m,i} e_{m,j}^\top + e_{m,j} e_{m,i}^\top - e_{m,i} e_{m,j}^\top e_{m,i} e_{m,j}^\top \right) \\
& \cdot (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
& - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_{\text{adj}}^{ij}} G^\top.
\end{aligned}$$

.

All that remains are the gradients with respect to the initial state variables. First,

$\mu_0$ :

$$\begin{aligned}
(C.26) \quad \frac{\partial \ell(h_N^K; \theta)}{\partial \mu_0^i} = & -\frac{1}{2} \sum_{t=0}^{N-1} \sum_{k=1}^K \left\{ - \left( J \frac{\partial \mu_t^k}{\partial \mu_0^i} \right)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \right. \\
& \left. - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \mu_0^i} \right\}.
\end{aligned}$$

Of course,  $\frac{\partial \Sigma_t}{\partial \mu_0^i} = \mathbf{0}$  for all  $t$ ; differentiating  $\mu_t^k$  gives the recursion

$$(C.27) \quad \frac{\partial \mu_0^k}{\partial \mu_0^i} = e_{n,i};$$

$$(C.28) \quad \frac{\partial \mu_{t+1}^k}{\partial \mu_0^i} = G \frac{\partial \mu_t^k}{\partial \mu_0^i} - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \mu_0^i}.$$

Finally,  $\Sigma_0$ :

$$(C.29)$$

$$\begin{aligned} \frac{\partial \ell(h_N^K; \theta)}{\partial \Sigma_0^{ij}} = & \\ & - \frac{K}{2} \sum_{t=0}^{N-1} \text{tr} \left[ (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} J^\top \right] \\ & - \frac{1}{2} \sum_{k=1}^K \sum_{t=0}^{N-1} \left\{ - \left( J \frac{\partial \mu_t^k}{\partial \Sigma_0^{ij}} \right)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \right. \\ & \quad - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} J^\top (J \Sigma_t J^\top)^{-1} (y_{t+1}^k - J \mu_t^k) \\ & \quad \left. - (y_{t+1}^k - J \mu_t^k)^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}}) J \frac{\partial \mu_t^k}{\partial \Sigma_0^{ij}} \right\}; \end{aligned}$$

$$(C.30) \quad \frac{\partial \mu_0^k}{\partial \Sigma_0^{ij}} = \mathbf{0};$$

$$(C.31)$$

$$\begin{aligned} \frac{\partial \mu_{t+1}^k}{\partial \Sigma_0^{ij}} = & \\ & G \frac{\partial \mu_t^k}{\partial \Sigma_0^{ij}} + G \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\ & - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} (y_{t+1}^k - J \mu_t^k) \\ & - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \mu_t^k}{\partial \Sigma_0^{ij}}; \end{aligned}$$

$$(C.32) \quad \frac{\partial \Sigma_0}{\partial \Sigma_0^{ij}} = e_{n,i} e_{n,j}^\top + e_{n,j} e_{n,i}^\top - e_{n,i} e_{n,j}^\top e_{n,i} e_{n,j}^\top;$$

(C.33)

$$\begin{aligned}
\frac{\partial \Sigma_{t+1}}{\partial \Sigma_0^{ij}} = & \\
& G \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} G^\top - G \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} J^\top (J \Sigma_t J^\top)^{-1} J (\Sigma_t G^\top + C_\eta) \\
& + (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J (\Sigma_t G^\top + C_\eta) \\
& - (G \Sigma_t + C_\eta^\top) J^\top (J \Sigma_t J^\top + \Sigma_{\text{adj}})^{-1} J \frac{\partial \Sigma_t}{\partial \Sigma_0^{ij}} G^\top.
\end{aligned}$$

## REFERENCES

## REFERENCES

- Bar-Shalom, Y., Li, X. R., & Kirubarajan, T. (2001). *Estimation with applications to tracking and navigation*. John Wiley & Sons, Inc.
- Bowling, M., McCracken, P., James, M., Neufeld, J., & Wilkinson, D. (2006). Learning predictive state representations using non-blind policies. In *Proceedings of the 23rd International Conference on Machine Learning*.
- Catlin, D. E. (1989). *Estimation, control, and the discrete Kalman filter*. Springer-Verlag New York.
- Ghahramani, Z., & Hinton, G. E. (1996). *Parameter estimation for linear dynamical systems* (Tech. Rep. No. CRG-TR-96-2). Dept. of Computer Science, U. of Toronto.
- James, M., & Singh, S. (2004). Learning and discovery of predictive state representations in dynamical systems with reset. In *Proceedings of the 21st International Conference on Machine Learning* (pp. 417–424).
- Kalman, R. E. (1960). A new approach to linear filtering and prediction problem. *Transactions of the ASME—Journal of Basic Engineering*, 82(Series D), 35–45.
- Littman, M. L., Sutton, R. S., & Singh, S. (2002). Predictive representations of state. In *Advances in Neural Information Processing Systems 14*.
- Marshall, G., & Olkin, I. (1984). Generating correlation matrices. *SIAM J. Sci. Stat. Comput.*, 5(2), 470–475.
- Norlander, T. (2000). *Control of a flotation process using LQ optimized PI controllers* (Research report No. 2000:08). Luleå University of Technology.
- Norlander, T., Nilsson, B., Ring, D., & Johansson, U. (2000). A study on active flutter detection and control. In *Proceedings of the IEEE National Aerospace & Electronics Conference* (pp. 172–179).
- Overschee, P. van, & De Moor, B. (1996). *Subspace identification for linear systems*. Kluwer Academic Publishers.
- Pandit, S. M., & Wu, S.-M. (1983). *Time series and system analysis with applications*. Krieger Publishing Company.
- Rafols, E. J., Ring, M. B., Sutton, R. S., & Tanner, B. (2005). Using predictive representations to improve generalization in reinforcement learning. In *Proceedings of the 2005 International Joint Conference on Artificial Intelligence* (pp. 835–840).
- Rivest, R. L., & Schapire, R. E. (1994, May). Diversity-based inference of finite automata. *Journal of the ACM*, 41(3), 555–589.

- Rudary, M., & Singh, S. (2004). A nonlinear predictive state representation. In S. Thrun, L. K. Saul, & B. Schölkopf (Eds.), *Advances in Neural Information Processing Systems 16* (pp. 855–862).
- Rudary, M., & Singh, S. (2006). Predictive linear-Gaussian models of controlled stochastic dynamical systems. In W. W. Cohen & A. Moore (Eds.), *Proceedings of the 23rd International Conference on Machine Learning* (pp. 777–784).
- Rudary, M., Singh, S., & Wingate, D. (2005). Predictive linear-Gaussian models of stochastic dynamical systems. In F. Bacchus & T. Jaakkola (Eds.), *Uncertainty in Artificial Intelligence 21* (pp. 501–508).
- Singh, S., James, M. R., & Rudary, M. (2004). Predictive state representations: A new theory for modeling dynamical systems. In M. Chickering & J. Halpern (Eds.), *Uncertainty in Artificial Intelligence 20* (pp. 512–519).
- Smith, G., Freitas, J. F. de, Robinson, T., & Niranjana, M. (1999). Speech modelling using subspace and em techniques. In S. A. Solla, T. K. Leen, & K.-R. Miller (Eds.), *Advances in Neural Information Processing Systems 12* (pp. 796–802).
- U.S. Federal Highway Administration. (2006). *Interstate 80 freeway dataset (fhwa-hrt-06-137)*. (<http://www.tfhrc.gov/about/06137.htm>)
- U.S. Federal Highway Administration. (2008). *Next generation simulation project*. (<http://ngsim.fhwa.dot.gov/>)
- Viberg, M. (1995). Subspace-based methods for the identification of linear time-invariant algorithms. *Automatica*, 31(12), 1835–1851.
- Wiewiora, E. (2005). Learning predictive representations from a history. In *Proceedings of the 22nd International Conference on Machine Learning*.
- Wingate, D. (2008). *Exponential family predictive representations of state*. Unpublished doctoral dissertation, University of Michigan.
- Wingate, D., & Singh, S. (2006a). Kernel predictive linear gaussian models for nonlinear stochastic dynamical systems. In *Proceedings of the 23rd International Conference on Machine Learning* (pp. 1017–1024).
- Wingate, D., & Singh, S. (2006b). Mixtures of predictive linear gaussian models for nonlinear stochastic dynamical systems. In *Proceedings of the 21st National Conference on Artificial Intelligence*.
- Wingate, D., & Singh, S. (2008). Exponential family predictive representations of state. In *Advances in Neural Information Processing Systems 20* (pp. 1617–1624).
- Wolfe, B., James, M. R., & Singh, S. (2005). Learning predictive state representations in dynamical systems without reset. In *Proceedings of the 22nd International Conference on Machine Learning*.