The London School of Economics and Political Science

Applied Probabilistic Forecasting

Author:

Roman Binter

London, December 17, 2012

A thesis submitted to the Department of Statistics of the London School of Economics for the degree of Doctor of Philosophy.

Declaration

I declare that my thesis consists of 34,772 words.

Abstract

In any actual forecast, the future evolution of the system is uncertain and the forecasting model is mathematically imperfect. Both, ontic uncertainties in the future (due to true stochasticity) and epistemic uncertainty of the model (reflecting structural imperfections) complicate the construction and evaluation of probabilistic forecast. In almost all nonlinear forecast models, the evolution of uncertainty in time is not tractable analytically and Monte Carlo approaches (”ensemble forecasting”) are widely used. This thesis advances our understanding of the construction of forecast densities from ensembles, the evolution of the resulting probability forecasts and methods of establishing skill (benchmarks). A novel method of partially correcting the model error is introduced and shown to outperform a competitive approach.

The properties of Kernel dressing, a method of transforming ensembles into probability density functions, are investigated and the convergence of the approach is illustrated. A connection between forecasting and Information theory is examined by demonstrating that Kernel dressing via minimization of Ignorance implicitly leads to minimization of Kulback-Leibler divergence. The Ignorance score is critically examined in the context of other Information theory measures.

The method of Dynamic Climatology is introduced as a new approach to

establishing skill (benchmarking). Dynamic Climatology is a new, relatively simple, nearest neighbor based model shown to be of value in benchmarking of global circulation models of the ENSEMBLES project. ENSEMBLES is a project funded by the European Union bringing together all major European weather forecasting institutions in order to develop and test state-of-the-art seasonal weather forecasting models. Via benchmarking the seasonal forecasts of the ENSEMBLES models we demonstrate that Dynamic Climatology can help us better understand the value and forecasting performance of large scale circulation models.

Lastly, a new approach to correcting (improving) imperfect model is presented, an idea inspired by [63]. The main idea is based on a two-stage procedure where a second stage 'corrective' model iteratively corrects systematic parts of forecasting errors produced by a first stage 'core' model. The corrector is of an iterative nature so that at a given time t the core model forecast is corrected and then used as an input into the next iteration of the core model to generate a time $t + 1$ forecast. Using two nonlinear systems we demonstrate that the iterative corrector is superior to alternative approaches based on direct (non-iterative) forecasts. While the choice of the corrector model class is flexible, we use radial basis functions. Radial basis functions are frequently used in statistical learning and/or surface approximations and involve a number of computational aspects which we discuss in some detail.

To my loving family, especially to the three most special women,
my grandmother Anna, my mother Helena,
and my wonderful wife Lenka.

Acknowledgments

I would like to thank my supervisor Prof. Leonard A. Smith for his relentless support that he so generously provided me with throughout the development of this thesis. My very special thanks goes to him for offering me the ‘blue pill’[1] and giving me the opportunity to peak into the world behind the mirror.

I would also like to thank all the past and present crew of the Centre for the Analysis of Time Series for being wonderful CATS, as well as members of the Department of Statistics for being inspirational friends, colleagues and teachers.

[1]The Matrix, 1999: Morpheus talking to Neo

Contents

List of Figures

List of variables

Here we list variables used throughout this thesis. We order the variables by chapters, since occasionally across different chapters a given symbol may correspond to different variables. Within a chapter each symbol represents one variable only. Chapter 2 is an exception to this rule as it provides background information across various fields; consequently, we could not avoid occasionally using a single symbol multiple times. The variables of Chapter 2 are therefore not listed, and are clarified in the text only.

Properties of Kernel Dressing

a	-	scaling parameter in Affine Kernel dressing
α	-	blending parameter between two densities
$c(\cdot)$	-	density of a house (betting)
$f(\cdot)$	-	punter forecasting density
$G(\cdot)$	-	rate of growth
$H(\cdot)$	-	Shannon's entropy
IGN	-	Ignorance
k	-	index of an outcome (a density bin)
K	-	number of discrete outcomes (density bins)
KL	-	Kullback-Leibler divergence
l	-	forecasting leadtime
μ	-	expected value of a distribution
M	-	ensemble size
N	-	number of observations
$N(\cdot)$	-	normal distribution
o	-	offset parameter in Kernel dressing
$p(\cdot)$	-	model forecasting density
$\hat{p}(\cdot)$	-	estimate of model forecasting density
$q(\cdot)$	-	system density
σ	-	bandwidth of a kernel or standard deviation of a distribution
s_k	-	k-th outcome
θ	-	parameter vector
$o(\cdot)$	-	odds issued by a house
t	-	time
τ	-	time delay
$v(\mathbf{z})$	-	variance of transformed ensemble
V_t	-	wealth at time t
W_k	-	number of occurrences of k-th outcome
x_t	-	initial condition or an observation (potentially noisy)
y_t	-	verification (potentially noisy)
z	-	transformed ensemble member
$\mathbf{z}$	-	transformed ensemble

Dynamic climatology and its benchmarking utility

b	-	damping friction (pendulum)
d	-	dimensionality of feature space (length of feature vector)
Δx^a	-	difference of an image and analog
θ	-	angle between two vectors
D_E	-	Euclidean distance
g	-	external forcing (pendulum)
K_0	-	number of analogs to be selected at leadtime 1
K	-	number of analogs to be selected beyond leadtime 1
l	-	forecasting leadtime
m	-	number of RBF centers
M	-	size of an ensemble
N	-	number of observations
ω_F	-	forcing frequency (pendulum)
t	-	time
τ	-	time delay
w	-	phase window
x_t	-	initial condition or an observation (potentially noisy)
$\mathbf{x}_t$	-	d-dimensional feature vector
$\mathbf{x}^a_{t-\tau}$	-	analogous feature vector
$x^a_{t-\tau}$	-	analog of an observation
$x^i_{t-\tau+1}$	-	image of an analog
$\hat{x}_{t+l}$	-	leadtime l point forecast

Forecasting with Radial Basis Functions

A	-	distance or RBF matrix
$B_j(\cdot)$	-	j-th basis function
c	-	center
$\mathbf{c}$	-	vector of centers
d	-	dimensionality of observation point
f	-	system function
$\tilde{f}$	-	estimate of system function f
λ_j	-	j-th basis function coefficient
$\boldsymbol{\lambda}$	-	vector of basis function coefficient
M	-	number of centers
N	-	number of observations
ϕ	-	radial function
r	-	distance in d-dimensional space
t	-	time
$\mathbf{x}$	-	d-dimensional observation point
y	-	observed value

Forecast correction PC v. $\Psi\Phi$

d	-	dimension (embedding dimension)
ε	-	forecasting error
ϵ^M	-	systematic part of a forecasting error, model error
ϵ	-	stochastic part of a forecasting error
l	-	forecasting leadtime
L	-	maximum leadtime
K	-	size of a forecasting ensemble
M	-	number of RBF centers
N	-	size of data
t	-	time
τ	-	embedding time delay
x_t	-	initial condition or an observation (potentially noisy)
$\mathbf{x}_t$	-	ensemble of initial conditions
$\mathbf{x}_{t+l}$	-	ensemble forecast at leadtime l
$\tilde{x}_t$	-	‘true’ system state
$\hat{x}_t$	-	forecast at time t
z_t	-	core model forecast in $\Psi\Phi$ and PC

Chapter 1

Introduction

In many fields where predictions of future states of systems [5] are required, probabilistic forecasting has become a frequently used method of choice. Probabilistic methods in forecasting have been pioneered by the US Weather Bureau, which in 1965 appended probabilities of precipitation as a standard part of public weather forecasts [95]. Since then, weather forecasting institutions around the world have shifted their focus from single point forecasts toward probabilities when producing forecasts at short term weather scales or seasonal or climate scales [100–102]. Economics is another influential field which has embraced probabilistic forecasting. The Bank of England (BoE) has been issuing 'fan charts', i.e. probabilistic forecasts of inflation, since 1997 [13] as part of the quarterly Inflation Report. Since then, many central banks, including ECB, Federal Reserve, Sveriges Riksbank [10], Czech National Bank [45], etc., have followed the BoE lead and included probabilistic forecasts of inflation and other macro-economic variables as part of their regular reporting. Probabilistic forecasting has also been used in finance to predict stock prices [160], in health to predict epidemics of diseases [23, 94] and, indeed, in insurance to price potential losses of extreme

Weather events [1, 96].

Probabilistic forecasting deploys Monte Carlo experiments [3, 56, 70], to produce a collection (ensemble) [78, 93, 98, 103, 143] of multiple point forecasts by using slightly different initial conditions [69, 93, 143]. Building on the ensemble of forecasts, probabilistic methods can be used to assign probabilities to future outcomes, yielding forecasting distributions. The forecasting distributions express uncertainty about future evolution of a system. Probabilistic forecasting is in stark contrast with point forecasts, where a forecast is given in terms of an expectation and hence much less information, if any, regarding uncertainty is provided.

Any forecasting process, probabilistic or not, typically involves several stages, including:

- data retrieval and transformation,
- current state estimation,
- forecast formation,
- post-processing,
- forecast evaluation
- benchmarking.

Probabilistic approaches introduce additional challenge due to the fact that they work with forecasting distributions as opposed to expectations. In this work we focus on three particular steps of the forecasting process, all of which will be studied within the probabilistic context. The steps of our interest are:

1. error correction

2. evaluation
3. benchmarking

The three above steps enter the forecasting process after the forecast formation. **Our focus is therefore on improvement and understanding of the value of a forecast and not on forecast formation. In this thesis the value of a forecast is the common thread linking the three areas of our interest.**

Error correction
Since all models are wrong [12] every forecast is prone to systematic errors. Error correction is a part of the post-processing stage and is designed to add additional value to a forecast by detecting and correcting systematic errors. If systematic errors are present it may be possible to detect and, to some extent, correct them [61, 99]. While simple error correction approaches focus on bias correction, more complex correctors may take the form of a two-stage procedure deploying an additional modeling layer on top of the forecasting model. In this thesis we deploy a two-stage procedure where the second stage corrector is designed to 'learn' the systematic errors of the first stage model and correct the errors in an iterative manner. The corrector is based on radial basis functions (RBF) [39, 40] and is shown to significantly improve forecasts for models with medium-to-large systematic errors while not degrading performance of models with very low systematic error. Deployment of RBF as the corrector introduces some computational issues, relating the 'power' of the available computational device to the quality of the corrector. A critical discussion of the computational issues is also presented.

Evaluation
The evaluation stage assigns value to a forecast by scoring the performance of a forecasting model. The performance measure (score) as well as the object,

i.e. what is being scored, are of crucial importance if one aims to achieve a meaningful understanding of the value of a forecast. For point forecasts the frequently used score is the Root Mean Square Error (RMSE) and the object to score is the value of the point forecast. A similar approach is often adopted in probabilistic setting when some distance metric, e.g. RMSE, is applied to each of the ensemble forecasts (or their mean or other statistics). While such an approach could yield some intuition as to 'how far' the ensemble forecasts are from the target, it is questionable whether we would gain a real insight regarding the value of the forecast. The issue is that both RMSE and the individual ensemble forecasts only hold information about expectations, and **disregard the very useful information contained within the distribution of the ensemble forecasts**. To make use of all the available information captured by an ensemble, the ensemble forecast must be transformed into a forecasting distribution, [15, 58, 112, 113, 115, 116] and the forecasting distribution itself then becomes an object of the scoring. There are two immediate challenges related to the process. First, what method should be used in order to transform the ensemble forecast into a distribution and is the method statistically sound? Second, what score should be applied to the forecasting distributions?

Ensemble transformation and scoring represent two integral parts of an evaluation method, the properties of which must be well understood. Using a biased evaluation method could lead to the selection of an inferior forecasting model. In this thesis we study Kernel Dressing (KD), a probabilistic evaluation method which uses, but is not restricted to, a logarithmic scoring rule called Ignorance [49, 115]. We show that its properties make KD particularly useful for probabilistic setting and, under some conditions, prevent selection of an inferior model. Also, since in practical applications Ignorance seems often overlooked despite its desirable properties, we show how this scoring rule relates to the well-known Information Theory measures of Shannon's

Entropy [124] and Kullback-Leibler divergence [74]. By exposing the links among the three measures we hope to further highlight the usefulness of Ignorance as a probabilistic scoring rule.

Benchmarking
The real value of a forecasting model can only be understood on a relative basis, i.e. when compared to an alternative model (benchmark) [4, 48, 53, 156]. Benchmarking can thus be considered as an interpretation of the score of a forecast. In probabilistic setting, the output of a benchmarking model is transformed into distributions so that they are comparable to the forecasting distributions of the forecasting model. The benchmarking distributions often take very simple form, e.g. unconditional distribution of past observations (climatology). While simple benchmarks are often robust, they may lack performance. More complex benchmarks may be required to improve on the performance and thus become a 'stronger' benchmark. One way of creating a stronger benchmark is to construct conditional distributions. Conditioning on some event (e.g. month of a year in seasonal forecasting) may yield improvements in performance at an affordable cost of a moderate reduction in robustness.

Another appealing method of benchmark construction is to use simple statistical models. Statistical models are capable of incorporating simple forecasting rules that may greatly improve performance, while preserving the desired level of robustness. Statistical models can therefore be more useful benchmarks than climatological distributions, and provide us with a much deeper understanding of the value of a forecasting model. In this thesis we present Dynamic Climatology (DC), a simple statistical model that poses a stronger benchmark than climatological distributions. We demonstrate some of its properties in simplified settings and then use it to benchmark the state-of-the-art seasonal-to-annual weather forecasting models of the ENSEMBLES project [28, 29, 36, 57, 152].

Thesis structure
This thesis is structured as follows: In Chapter 2 we provide background information that is drawn upon throughout the thesis. We discuss fundamental concepts such as ensemble forecasting [78, 98, 143], forecasting scenarios [65, 66] and sources of uncertainty [31, 131, 132]. We also define the concept of a forecasting framework, discuss methods of forecasting density construction [15, 58, 110] and (un)conditional climatological forecasts.

In Chapter 3 we investigate the properties of Kernel Dressing (KD) [15, 58], a method of transforming ensemble forecasts into forecasting densities, and study the properties of Ignorance [14, 49, 115], a measure of forecasting skill. We show that, although similar in concept, KD substantially differs from Kernel Density Estimation as understood by [11, 20, 129]; a fact that has not been fully recognized. We show analytically that minimizing Ignorance implicitly leads to minimization of Kullback-Leibler divergence [75]. We numerically demonstrate that under the perfect model scenario (PMS) [133], KD recovers the system density, suggesting that KD is an unbiased estimator. We also perform a novel numerical analysis of the KD properties outside PMS and demonstrate that caution must be exercised when deploying Affine KD (an extended version of KD), a new fact that has been previously overlooked [14]. Using the Kelly betting framework [72] we clarify important links between Ignorance and alternative Information theoretical measures, namely Shannon's entropy [124–126] and Kullback-Leibler divergence.

In Chapter 4 we introduce Dynamic Climatology (DC), a new approach to defining a zero skill reference (benchmark) [157]. DC is a new, and relatively simple, statistical model shown to be a valuable benchmarking tool, which we deploy to benchmark the forecasting skills of the state-of-the-art global circulation models of the ENSEMBLES project. The rationale for constructing DC is that traditional zero skill references such as climatological forecast (climatology) may not yield lead an adequate quantification of model skill [53, 68, 89]. By defining a stronger reference, e.g. DC, we may obtain a more thorough understanding of the value of a forecasting model. **We construct DC to outperform climatology and demonstrate that DC indeed does outperform climatology. We also construct DC to accommodate for degradation of forecasting skill at long leadtimes and to be capable of producing 'new' values, not contained within the training set.** We deploy DC to forecast Sea Surface Temperatures over two regions important for seasonal weather forecasting: the Nino3.4 and the Main Development Region. We contrast the DC forecast with those of ENSEMBLES models [36, 57, 152] and demonstrate that DC is a comparable, and occasionally a 'better', performer.

In Chapter 5 we provide background information on radial basis functions (RBF), as they are extensively used in Chapter 6. We discuss the basic setting of RBF interpolation and approximation, and give arguments for formulating forecasting problems in terms of RBF approximation.

In Chapter 6 we present predictor-corrector (PC), a new approach to improving the forecasts of an imperfect model, based on iterative corrections of the systematic part of a model error [61, 67, 137]. For several nonlinear systems, we show the PC significantly improves imperfect model forecasts and is superior to an alternative approach of $\Psi\Phi$ [62, 63].

To demonstrate the skill of the two approaches, PC and $\Psi\Phi$, we consider two well-known dynamic systems, Lorenz84 [83] and Lorenz63 [81]. Using Lorenz84, we show that, for a low-complexity imperfect model, PC improves the forecasts by > 1.5 Bits and outperforms $\Psi\Phi$ by up to 1 Bit at long leadtimes. We also test the behavior of both PC and $\Psi\Phi$ as we gradually improve the forecasting skill of the imperfect model. We show that PC

maintains its superiority at medium range, even in settings with a (much) more skillful imperfect model. The computational aspects of the PC and $\Psi\Phi$ approaches are also discussed in detail, both in general and specific terms.

We also use the Lorenz84 and Lorenz63 systems to study the impact of Root Mean Square Error (RMS) [97], when used as a meassure of forecasting performance [134]. Intuitively, given a measure of forecasting performance (say RMS), it is reasonable to require that the measure rates a perfect model (i.e. a model equivalent with the system) higher than an imperfect model (an approximation of the system). Measures that do not meet such a requirement may be misleading and may lead a forecaster to believe that an approximation of a system may be more useful then the system itself. Whilst a number of measures do meet the requirement, i.e. are proper measures, it has been shown that RMS is not a proper measure. Despite its potentially harmful properties RMS continues to be frequently used in the evaluation of forecasting models. Our aim is to expose the danger of using RMS. To do so, we numerically demonstrate how RMS-based evaluation may be misleading, and how it can lead to the selection of an inferior model.

Chapter 2

Background

This thesis capitalizes on a number of concepts of statistics, information theory and dynamical systems. In this chapter, we provide background information that will be drawn upon throughout the thesis.

We begin with a general description of a forecasting problem in Section 2.1. We discuss a system, a forecasting model, and give our definition of a forecasting framework. We follow with a discussion of forecasting scenarios [65, 66] and briefly describe sources of uncertainty [31, 131, 132] obscuring the future evolution of a dynamical system.

In Section 2.2 we provide a brief description of ensemble forecasting [78, 98, 143] and discuss several methods of forecasting density construction [15, 58, 110]. Both ensemble forecasting and forecasting densities are key concepts in probabilistic forecasting and will be used throughout this work. We also define unconditional and conditional climatological forecasts and discuss their application. These climatological forecasts are used extensively in Chapters 4 and 6

Chapter 3 is concerned with the properties of Kernel dressing, a method of forecasting density construction and evaluation. We discuss the general aspects of forecast evaluation in Section 2.4, where we also define Kernel dressing [15].

At the center of forecast evaluation are measures of forecasting performance. Measures originating from Information theory [22, 72, 74, 124, 125] have proved to be of particular importance [46, 115]. The Information theoretical measures and their importance for evaluation of forecasting performance are discussed in Section 2.3.

Lastly, we briefly discuss issues related to seasonal weather forecasting and describe the forecasting models of the ENSEMBLES project [36, 57], as they will be a subject of study in Chapter 4.

2.1 Forecasting

The future evolution of any physical system is clouded with uncertainty, which limits our ability to precisely determine future states of the system. In our efforts to better understand the behavior of a system, we construct inherently imperfect mathematical models describing the underlying rule governing the system. These imperfect models are often used to issue statements about future system states - a process called *forecasting* [5].

In forecasting, systems are often classified as either *deterministic* or *stochastic*. Deterministic systems are governed by a fixed behavioral rule and do not involve any randomness. Stochastic systems, on the contrary, do involve random behavior, although their behavioral rule may also involve a fixed (non-random) part.

Throughout this work, we are concerned with forecasting of *dynamical sys-*

tems, a subclass of deterministic systems. A good definition of a dynamical system can be found on Wikipedia [155]: a dynamical system is '*a mathematical concept where a fixed rule describes the time dependence of a point in a geometrical space*'. An example of dynamical systems are physical laws, typically described in terms of differential equations, e.g. a simple pendulum[8] derived from Newton's second law

$$A\ddot{\theta} = -g \sin\theta \tag{2.1}$$

where A is the length of the pendulum, θ is the angular displacement of the rod and g is gravitational force.

In our work (e.g. Chapter 4), we apply probabilistic methods in the context of weather forecasting. While the pendulum system of Eq. 2.1 is an example of a simple dynamical system, weather is an example of a very complex dynamical system. Weather is defined as a state of the atmosphere, and is often described using a set of fixed behavioral rules, i.e. physical laws such as the one in Eq. 2.1, which form a dynamical system. Weather is also considered a chaotic system [81]. A chaotic system is a type of dynamical system sensitive to initial conditions. Given two initial states that are very close (as quantified by some measure), the future states obtained by evolving the two initial states may end up very far from each other. In other words, small differences in initial states (conditions) may yield entirely different forecasts. Obtaining a useful forecast of complex, chaotic system such as weather is a challenging task. Yet due to its direct impact on many fields, including agriculture, transport, insurance, etc., weather forecasting is of crucial importance.

In this work, our aim is to apply forecasting approaches based on probabilistic methods. Forecasting is used throughout a wide spectrum of scientific fields as well as industries and is applied to a large variety of dynamical and stochastic systems differing in complexity as well as their nature (natural vs. mathematical). It is not possible to apply our approaches to all possible classes of systems; there are simply too many of them. What we can do is apply our methods to a few selected systems for which we believe our approaches are useful. Our applications therefore involve simple mathematical dynamical systems (e.g. Lorenz63 [81], pendulum) but also a highly complex chaotic system (weather).

Although we deliver a number of results that may help to better understand the studied systems (or their models), our main goal is to present probabilistic methods and how they may be applied in forecasting. Our emphasis is not on system analysis, but rather on a thorough description of (new or existing) probabilistic methods and their properties, as well as a comprehensive illustration of their application.

2.1.1 System model pair

Forecasting a system requires a vehicle, a *forecasting model*. By a forecasting model, we will understand a mathematical description of a system. Models are constructed to describe systems, and so the models themselves can also be classified as deterministic and stochastic. To describe a dynamical system, a physicist would typically use a deterministic model, i.e. differential equation (or a set of equations) similar to Eq: 2.1. On the other hand, a statistician would probably choose a stochastic model to describe the system. In this work, we will be using both approaches.

A given system may be described by different models; where a physicist uses differential equations, a statistician may use regression. But even if we stick with the physicist's toolbox, there might be different sets of differential equations describing the same system. The obvious reason for having more than a single mathematical description of a system is that it may be impossible to describe the system exactly. In fact, no model of a physical dynamical system is able to exactly describe the system at hand, simply because forecasters do not possess a perfect knowledge of all laws occurring in nature.

2.1.2 Forecasting framework

The process of forecasting involves several stages, the collection of which we call a *forecasting framework*. Although the number of stages may vary with an application, in general, the forecasting framework involves:

(1) collection of observations,

(2) data quality control,

(3) current system state estimation,

(4) forecast generation,

(5) forecast post-processing,

(6) construction of forecasting density,

(7) forecast evaluation.

Items 1-3 are often referred to as *data assimilation*. In fields such as weather forecasting, the data assimilation requires more human, computational, and financial resources than the rest of the forecasting framework. For example, the European Center for Medium and Short Range Forecast (ECMWF) processes a total of 75 million observations from satellites and conventional observation devices every 12 hours. To collect and quality control such an amount of data is a major task.

Collected and quality-controlled data then may serve as an input for *current state estimation* procedures. In nonlinear dynamical systems small errors in a current state estimation may ruin a point forecast as chaotic (or highly nonlinear) systems, such as weather, display high sensitivity to initial conditions [79, 81]. Due to the significant impact on the quality of a forecast, state estimation methods, such as 4DVaR [139, 142], Particle Filter [50, 108], Gradient Descent [59, 64] etc., are a very active area of research.

Forecast generation is a central part of the forecasting framework. The forecasting model is initialized by initial conditions determined within the state estimation stage. The initial conditions are than integrated by the model to produce a raw ensemble of point forecasts for a required future time, which we call a *leadtime*.

The raw ensemble often contains biases. Model output post-processing is a type of quality check applied to the ensemble and designed to remove biases. Outliers may also be handled within this stage.

If a probabilistic forecast is to be issued, the post-processed ensemble needs to be turned into a forecasting PDF, which assigns probabilities to possible future states of the forecasted system. The forecasting PDFs are also used in the forecast evaluation to ex-ante assess the rate of success of a forecast.

By providing this very short, and possibly incomplete, description of a forecasting framework, we are trying to show that forecasting is a complex process involving a number of procedures and methods. It must be stressed that each and every stage is prone to errors and imperfections, which negatively impact on the quality of a forecast. In an attempt to improve a forecast, a forecaster should pay attention to all of the stages of the forecasting framework.

2.1.3 Perfect and imperfect model scenario

When investigating properties of a forecasting model, both the *Perfect model scenario* (PMS) [65, 133] and the *Imperfect model scenario* (IMS) [66, 133] are useful concepts. In Chapter 3, we rely on PMS to demonstrate properties of a forecast evaluation method. To explain what we understand by PMS, we first describe IMS and than define PMS as somewhat the opposite of IMS. The descriptions loosely follow [133].

Imperfect model scenario: IMS is a scenario in which the forecasting model provides an imperfect description of a system. It assumes that a forecaster does not know the system rule exactly. Often it is also assumed that the system's current state is not known exactly.

Every 'real world' forecasting exercise is an example of IMS; there are no perfect models in the real world. A physicist does not know exactly the governing equations of a particle motion, a weather forecaster has at best a crude description of a weather system and no social scientist has a model that would perfectly predict an election outcome. Under IMS the forecasting model always produces an *imperfect forecast*.

Perfect model scenario: PMS is the opposite of IMS to some extent. Consider a forecaster with a complete knowledge of the laws governing a system. This perfect knowledge allows him/her to construct an ideal description of a system, a *perfect model*.

We mentioned above that in IMS a forecast is always imperfect. Does it mean that in PMS, with a perfect model at hand, a perfect forecast can be achieved? The answer is no. To produce a perfect forecast of a non-linear dynamical system, a forecaster needs to know exactly the current state of a system. Any uncertainty about the current state due to an *observational error*, or any other source of uncertainty, accumulates over time [98, 132, 133] leading to an imperfect forecast. Under PMS, a forecast is always imperfect unless the exact current state is known.

With real world forecasting exercises falling under IMS, is there any example of PMS? The answer is that PMS can only be constructed. This can be achieved by letting the model act as both the model and the system. Consider a forecasting model taking the form of a standard normal distribution, $N(0, 1)$, and a system producing a sample from the same distribution at each time t, i.e. $x_t \overset{iid}{\sim} N(0, 1)$. The forecasting model will produce an imperfect but highly valuable forecast of x_t, since the distributions of the model forecast and system states will be equivalent. This may look like cheating, but PMS is in fact an ideal testbed for understanding the properties of a forecasting model, as we show in Chapter 3.

Perfect ensemble: Often, it is useful to use the concept of a *perfect ensemble* in combination with PMS. With a perfect ensemble, we assume a distribution of a current system state, which is a much weaker assumption than assuming an exact knowledge of a current state. The distributional assumption allows for sampling current state 'candidates', so that an ensemble of 'perfect initial conditions' may be constructed. Under PMS and perfect ensemble, a forecast remains imperfect, but the forecast distribution is equivalent to the distribution of the future system states.

The combination of PMS and perfect ensemble involves rather strong assumptions. So when is such a combination useful? The two concepts are ideal tools when trying to understand the properties of the whole forecasting framework (see Section 2.1.2). For example, if PMS with a perfect ensemble yields a biased forecast, a forecaster knows that the bias is due to the evaluation method; both the forecasting model and the current state estimation method may be ruled out as a sources of the bias. Without the perfect ensemble, the forecaster could not be certain whether it is the evaluation stage or the current state estimation stage causing the bias.

2.1.4 Sources of uncertainty in physical systems

Forecasts of any dynamical systems are inherently inaccurate. There are two major sources of uncertainty directly influencing the accuracy of a forecast [98, 99]: uncertainty about the current state of a system, i.e. the *initial condition* uncertainty, and uncertainty due to the inaccuracies in a forecasting model specification, the *model error* [61, 137].

Initial condition uncertainty: Under IMS conditions, uncertainty stems from the forecaster's inability to accurately observe a current state of a system, an *observation error*. The observation error is particularly influential when sets of differential equations are used as a model. It is well known [81], that for nonlinear (and in particular chaotic) systems/models, an arbitrarily small initial error in the initial state will accumulate (grow) over the forecasting *leadtime*. The error growth will eventually cause the forecasting model to reach its *predictability limit* [133], a leadtime beyond which the forecast is no longer 'useful'. Although stochastic models may be less susceptible, they also may suffer from the imprecise determination of the initial condition.

Model error: The uncertainty related to the model formulation arises due to an imperfect understanding (or description) of the predicted system. The imperfect model formulation leads to a model error, which will eventually cause the model to reach its predictability limit.

The model error is often caused by a number of factors. One way of disentangling the model error is as follows:

(1) *Structural error*: The equations describing the systems are 'incorrect', i.e. there may be a variable missing or a function is incorrectly specified.

(2) *Error due to a choice of approximation and discretization method*: A forecasting model is often deployed on a computational device, which is by nature of a finite precision. i.e. a digital (discrete) device. Model equations describing a given system frequently take a continuous form. To deploy a model on a computer, the continuous equations need to be discretized. This transformation involves numerical methods that often require approximations and truncations [145], and thus introduce 'imperfections' into the model's original (analytical) equations. The choice of a transformation method influences the severity of imperfections. A poor choice of transformation may lead to a significant model error, even in computers with very high precision.

(3) *Error due to model resolution*: Computational devices always work at a limited capacity. Their main constraints, limited memory and speed of processing, translate into restricted precision, limitations of volume of tasks, and constrained execution time. Due to the constraints, the values of system variables are calculated on a discrete mesh to which model equations are adjusted (see above paragraph). The density of the mesh is another source of imperfections. Using a very dense mesh means that less approximation is needed to obtain state values between mesh points, giving less room for an imperfection due to approximation. Note that model error due to resolution differs from that due to discretization. This is because a high density mesh cannot compensate for errors introduced by a poorly chosen discretization method. Hence models with low mesh density and high quality approximation methods

may exhibit lower errors than those with high mesh density but poor approximation/discretization method.

To minimize the observation error, a forecaster may attempt to improve the measuring device. This, however, may prove difficult due to budget constraints or the fact that the device is already state-of-the-art. A more realistic approach is to use a number of observational devices and to determine the initial state as some function of their outputs. Alternatively, a forecaster may attempt to sample the distribution of initial states as discussed in Section 2.2.1.

To address model error, a forecaster may attempt to obtain a better understanding of the system. Another option is to see whether the model error contains a systematic part and if so to correct it. When using deterministic models, the systematic part of the model error is frequently present. In such cases, error detection and correction algorithms can be very successful in reducing the systematic part of the model error. We suggest a framework that deals with the systematic part of a model error in Chapter 6.

2.2 Probabilistic forecasting

2.2.1 Ensemble forecasting and forecasting densities

In Section 2.1.4, we have discussed how an inaccuracy in determination of a current state accumulates over the integrations of a forecasting model, and degrades the quality of a forecast. We have also mentioned that a forecaster may address the initial condition uncertainty using Monte Carlo methods. *Ensemble forecasting* [78] is one such approach. Ensemble forecasting is frequently deployed in weather forecasting, and aims to provide a representative sample of possible future states of a system.

To produce an ensemble forecast (or simply an *ensemble*), a forecaster samples possible current states, generating M initial conditions $\mathbf{x}_t = \{x_{t,k}\}_{k=1}^{M}$, which form an *initial condition ensemble* at the initial time t. The initial conditions $x_{t,k}$ are then iteratively input into a forecasting model to produce an ensemble of M point forecasts, $\hat{\mathbf{x}}_{t+1} = \{\hat{x}_{t+1,k}\}_{k=1}^{M}$, i.e., an ensemble forecast at leadtime $t+1$. Fig: 2.1 shows a simple schematic of ensemble forecast generation, for the case of $M = 4$ initial conditions. In an ideal case, the forecast values would be a representative sample of the possible system states at time $t = 1$.

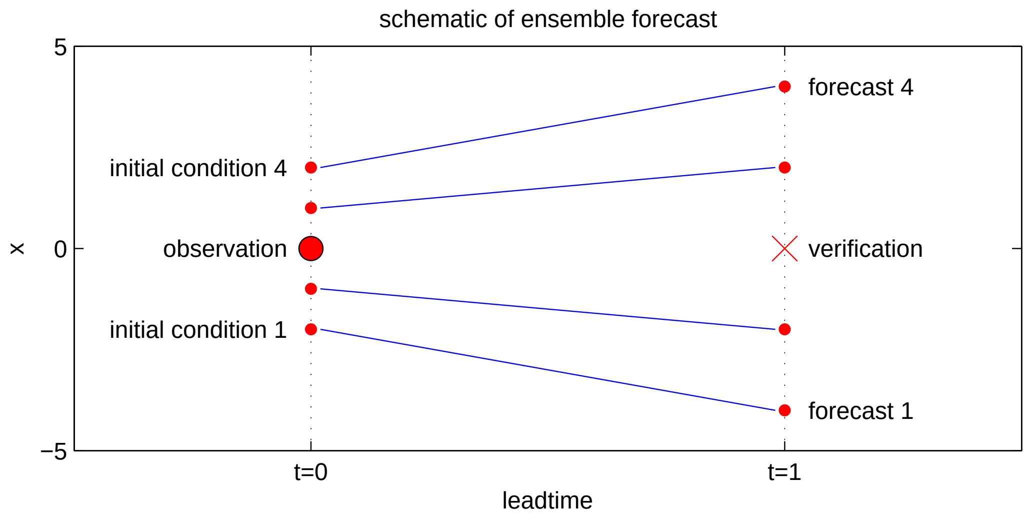

Figure 2.1: **Ensemble forecast:** Using an observation (large red circle) at time $t = 0$, an ensemble of initial conditions (small red circles at $t = 0$) is constructed and iterated forward using the model (blue lines) to obtain an ensemble forecast (small red circles at $t = 1$). At time $t = 1$ the forecast can be verified with an outcome (red cross), i.e. system state observed at time $t + 1$.

Forecast-verification archive: A forecasting model is often used to produce ensemble forecasts extending to several lead times. At some later time, when the forecasted system states are actually observed, an *archive of forecast-verification pairs* may be formed. The archive is constructed simply by pairing an ensemble with the observed system state at each leadtime. The archive of forecast-verification pairs can than serve as a training set when constructing forecasting densities and/or evaluating forecasting performance of a model.

Forecasting distribution: A natural way of describing an uncertainty is via distributions. A sample of possible future states, an ensemble forecast, may be turned into a *forecasting distribution*, $\hat{p}$. The forecasting distribution is able to concisely summarize the uncertainty caused by observational and modeling errors. When producing forecasts for $L > 1$ leadtimes, a forecasting distribution is constructed for each leadtime l, giving a sequence of distributions $\{\hat{p}_{t+l}\}_{l=1}^{L}$. There are a number of methods to construct the forecasting distributions, and we briefly discuss some of them in the next section 2.2.2.

An important question is: why would a forecaster wish to turn ensemble forecasts into forecasting densities? There are three main reasons for doing so. First, a forecasting density can be used to assign probabilities to the possible future states of the system. Second, a forecasting density enables evaluation of a forecast in probabilistic terms. This is an important part of the forecasting framework discussed in more detail in Section 2.4. Third, the density construction is often designed to include post-processing procedures such as bias correction, designed to improve raw forecasts ex post.

2.2.2 Constructing forecasting densities

There are number of methods to turn ensemble forecasts into forecasting densities. Here we briefly discuss a selected few, namely *Logistic regression*, *Gaussian dressing*, and *Bayesian model averaging*. By choosing these three particular methods, we aim to provide insight regarding ensemble-based density construction. Since our selection represents rather different approaches, we hope to demonstrate the diversity of existing approaches. We note that for this work, the most relevant is the Gaussian dressing, which provides a link to our preferred method of *Kernel dressing* (section 2.4.3). Although Logistic regression and Bayesian averaging are not directly linked to Kernel dressing, we hope that understanding the alternatives will help the reader to better appreciate the concept of density construction from ensemble forecasts. A detailed account of methods discussed, as well as alternative methods, can be found in [157].

Logistic regression: In weather forecasting, logistic regression has been a frequent tool of forecasting density estimation [157] [54]. Logistic regression is a useful method when a forecaster takes an interest in the quantiles, q, of possible outcomes. In ensemble forecasting the logistic regression often takes the form of

$$P(y \leq q) = \frac{1 + exp(\beta_0 + \beta_1 \bar{\mathbf{x}} + \beta_2 \hat{\sigma})}{exp(\beta_0 + \beta_1 \bar{\mathbf{x}} + \beta_2 \hat{\sigma})} \tag{2.2}$$

where $\bar{\mathbf{x}}$ is a mean of an ensemble forecast, $\hat{\sigma}$ is the ensemble variance and y is the observed system state, i.e., the verification. The parameters $\beta_0, \beta_1, \beta_2$ are determined by maximizing likelihood of the verification y being in the quantile q over all forecast-verification pairs contained within an archive.

Gaussian dressing: Gaussian dressing (GD) is a sort of *ensemble dressing* a density construction method proposed by [116]. GD applies gaussian distribution to each forecast-verification pair and takes the form of

$$p(y; \mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\mathbf{x}-\mu)^2}{2\sigma^2}}, \tag{2.3}$$

There are different ways of determining the parameters μ and σ. In the simplest form of GD μ and σ are set to some function of the ensemble mean and ensemble variance respectively, i.e., $\mu = f(\bar{\mathbf{x}})$ and $\sigma = g(\hat{\sigma})$. As noted by [14], this has the undesirable consequence that the forecasting distribution is solely based on the ensemble forecast, i.e. information provided by the actually observed state y is neglected. Other authors, e.g. [48, 58] suggested to determine μ and σ according to a forecasting performance. This leads to maximization of some function of $p(y)$ over the forecast-verification pair. We will discuss the performance-based approach in Section 2.4.3.

Bayesian model averaging: Bayesian model averaging (BMA) [110] addresses an uncertainty stemming from selecting a suboptimal model. Instead of relying on a single model BMA considers K candidate models and calculates a posterior probability of an outcome by conditioning on the K models. Consider a model k producing a l step ahead point forecast $x_{k,t+l}$, where x_k represents the k-th ensemble member. The forecast $x_{k,t+l}$ is associated with a PDF, $q_k(y|x_{k,t+l})$. The PDF q_k is a PDF of the outcome y conditional on x_k being the best forecast in the ensemble forecast. Dropping the $t+l$ term for simplicity, the BMA model can be written as

$$p(y|x_1, \dots, x_K) = \sum_{k=1}^{K} w_k q_k(y|x_k) \tag{2.4}$$

where w_k is the probability that the forecast x_k produced by model k is the best forecast in the ensemble. The w_k thus may be viewed as weights assigned to each of the ensemble members and are estimated via maximization of the log-likelihood of $p(y|x_k)$ over a testing data. The functional form of p can take the form of a parametric distribution. When a Gaussian distribution is assumed, the BMA can be described as a weighted sum of K Gaussian kernels, each centered at the ensemble member x_k, with a variance determined from the historical errors of the forecast x_k.

2.2.3 Unconditional climatology and climatological forecast

The climatological distribution, or simply *climatology*, is an unconditional distribution, $p_c(x)$, of observed values of a system variable x over the time period $[t - \Delta, t + \Delta]$. Climatology may be constructed using parametric or non-parametric density estimation methods. In our work, we estimate climatology using the Kernel dressing method described in Section 2.4.3.

Climatology is not only useful as a statistical description of the forecasted data, it may also be used as a simple probabilistic forecasting model. When used as a forecasting model, the forecasting density for a given leadtime l simply coincides with the climatology, $p_{t+l}(x) = p_c(x)$. This approach yields a probabilistic forecast that does not change over leadtime.

Assuming that the systems dynamics is (to some degree) captured by data used to construct climatology, the climatological forecast may be considered a robust forecast. In other words there will be a few surprises to the climatology if the dataset reasonably captures the system's past dynamics and the dynamics do not change abruptly.

As the climatological forecast does not change with leadtime, it might not seem a very useful forecasting model. However, its usefulness can be better appreciated when compared with forecasts produced by alternative models. Since climatology is simple to construct, quick to use, and produces robust forecasts, it may serve as a useful reference forecast defining a *zero skill*. An alternative model should always outperform a zero skill forecast. What would be the purpose of a model whose forecast is 'worse' than a zero skill reference forecast?

Climatology may be constructed in a number of ways, e.g. via parametric density estimation [58], kernel density estimation [11, 129] or kernel dressing (see Section 2.4.3). The climatological distribution can be used to produce a very basic *climatological forecast*, simply by sampling a single (or multiple) value(s) from the distribution.

2.2.4 Conditional climatology

Climatological distribution may take more complex forms. We can construct so called *conditional climatologies* by conditioning on some feature of the data. For instance, when a seasonality is present, a forecaster may choose to condition on a particular season. In weather forecasting, it is common practice to construct a *monthly climatology* by conditioning on a month of a year. In economics, climatologies conditional on a business or political cycle are often used.

In Chapter 4 we will be using a monthly climatology to benchmark alternative forecasts. Monthly climatology is a special case of a climatology where we condition on phase information. In general, climatology conditioned on a phase can be defined as

$$p_\theta(x) = p(x|x_t : t \in [t - \theta_1, t + \theta_2]) \tag{2.5}$$

where x_t is a time series of x and t is a point on a circle with circumference equal to length of a period and θ_1, θ_2 are phase angles defining an arc over the circle. In case of the monthly climatology and considering monthly observations of some variable, the period would be 12 months long, the time

t would be set to the number indexing a given month and $\theta_1 = \theta_2 = 0$. In practice, one may construct monthly climatologies that span over more than 1 month by setting $\theta_1 = \theta_2 > 0$. Asymmetric monthly climatologies can also be constructed by setting $\theta_1 \neq \theta_2$.

We note that it only makes sense to construct monthly climatology when a monthly seasonality is actually present in the data. Constructing monthly climatology of temperatures over sub-Saharan Africa may not make much sense, since the seasonality at these latitudes may not be related to months but rather to seasons.

In the following sections we use data from the application in Section 4.3.2 to demonstrate how monthly climatology can be a more useful climatology than the unconditional. In Fig: 2.2 we show a measure of forecasting performance against the varying size of the phase angle. Since the data used to generate the plot are monthly data, the phase angle has a simple interpretation. Window size of 0 corresponds to a 'pure' monthly climatology, i.e. only the same months are considered in the conditioning. Window size of 6 means that all months of a year may be considered, which leads to unconditional climatology. The measure of performance will be explained in Section 2.4.2; for now we can say that lower value mean a better performance. Clearly, the window size of 0, which corresponds to the 'pure' case of monthly climatology, gives much better results then the window size of 6 which corresponds to unconditional climatology.

2.2.5 Blending climatological and model forecasts

At the 'short' leadtimes, a forecasting model is expected to outperform the climatological (or any other zero skill) forecast. Should it fail to do so, the model should not be selected (see Section 2.2.3). At the long leadtimes,

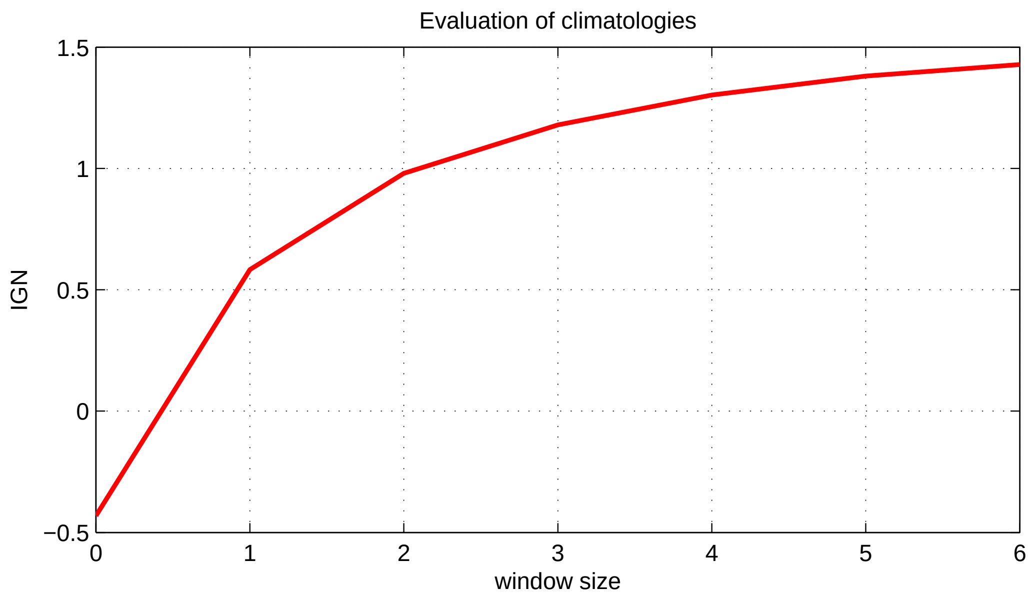

Figure 2.2: **Climatological ignorance in Nino34:** Forecasting performance of climatology of monthly SST temperatures over Nino3.4 region evaluated in terms of Ignorance. Ignorance is calculated for a varying size of a phase angle (window). For window size of 6 the conditional climatology becomes unconditional, as all observations will fall within the window. The optimal window in this case is 0 (the lowest level of Ignorance), i.e. the climatology should condition only on a given month of a year.

however, the error growth will cause a model forecast to deteriorate. Thus a forecasting model that performs well at short leadtimes may be outperformed by climatology at long leadtimes. The concept of *blending* aims to prevent the model from underperforming the climatological forecast.

The idea is simple. For a given leadtime, a model forecasting density $p_m(x)$ is ‘blended’ with the climatology $p_c(x)$ producing a ‘final’ forecasting density $p(x)$

$$p(x) = \alpha \times p_m(x) + (1 - \alpha) \times p_c(x) \tag{2.6}$$

where $\alpha \in [0, 1]$ is a blending parameter. As a result, at leadtimes where the model significantly outperforms climatology, the parameter $\alpha \to 1$. At the long leadtimes, where the model forecast is eroded by the growth of the forecasting error, $\alpha \to 0$ and the climatology 'takes' over. The final forecasting density p thus always outperforms or is comparable to the climatological forecast.

In this work, unless stated otherwise, all model forecasting densities are blended with climatology. Where suitable, the blending is done using a conditional climatology, e.g. monthly climatology.

2.3 Information theory in forecasting

Achieving a high quality forecast is a central point of any forecasting exercise. The question is how to measure the quality of a forecast, or, in other words, the performance of a forecasting model? When assessing performance of a model, *predictability* of the system should be considered. Quantification of predictability dates back to [82], where it is defined in relation to climate as 'the time span through which a given climate is supposed to last'. The intuitive link between a model performance and predictability is that for systems with low predictability the forecast starts to deviate from its target very soon, i.e. at short leadtimes. Intuitively, if a state of a system changes too frequently and the changes do not follow detectable patterns, it becomes very difficult for any model to produce a forecast that stays with the target over a long time span. Ideally, a forecaster would assess forecast quality simply by comparing the time over which a forecast stays close to a target with the predictability of the system. The problem is that, apart from some special cases [162], the predictability of a system is rarely known.

Most (if not all) measures of model performance are based on the notion of closeness, i.e. they aim to evaluate how 'close' a forecast is from its target. Probably the best known measure is the predictive mean squared error, where the 'closeness' is defined as the Euclidean distance of a forecast from its target. Typically, the mean squared error is applied to point forecasts. However, with the development of ensemble forecasts we can form so-called forecasting densities, distributions constructed from an ensemble forecast, and confront the distribution with the target. The reason for working with distributions is that a distribution contains more information than a single point and so measures based on distributions may provide more robust evaluation of forecasting performance [112, 133].

The question is, what performance measures are useful? For example, earlier studies in weather forecasting [87, 128] often deployed measures of predictability based on the signal-to-noise ratio. The noise-to-signal ratio can be measured at different locations and different situations. However, as [77] point out, to obtain an overall measure of forecasting performance the changes in the signal should be weighted by the frequencies of their occurrence and then aggregated. The problem is that aggregating signal-to-noise ratios over different times (or locations) has no clear interpretation. This drawback of the signal-to-noise ratios led to a search for alternative measures.

An important class that overcomes the issues of signal-to-noise ratios have been developed within Information theory [124, 125]. The class of Information theory-based measures is more general than signal-to-noise ratios [77] since it can compare forecasts across different locations or times. Furthermore, these measures take into account forecasting distributions and thus can deliver a more robust performance assessment. Since this class of measures has proved very influential across a variety of forecasting fields, and has been successfully applied to assess forecasts of both low-dimensional as well as complex dynamical systems, we will be using the information-based measures throughout this work. The focus in this subsection is to provide some brief background information on these measures.

2.3.1 Information-based measures of performance

The work of [77] suggests using measures developed within Information theory as measures of predictability in simple climate models. Their approach uses the *entropy*

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i), \tag{2.7}$$

which measures uncertainty related to states x_i occurring with probabilities p_i. Since by definition, the entropy is a measure weighted by the frequency of occurrence of states, it can be used as an aggregated measure of predictability. Moreover, [77] make the point that upon making a measurement an uncertainty, hence entropy can be used as a measure of information gain.

They further consider the *mutual information*

$$I(X;Y) = \sum_{x,y} p(x,y) \log \left(\frac{p(x|y)}{p(x)} \right) \tag{2.8}$$

and use it to measure how fast information about a given initial change of signal (or anomaly) is degraded.

2.3.2 Forecast performance and optimal compression

An alternative approach based on Information theory is suggested by [115], who construct a score that minimizes the amount of information needed to describe the distribution which governs occurrence of a system state. In Information theory this is the problem of optimal data compression discovered and solved by [124–126]. In data compression, one aims to encode given information using fewer binary digits (bits) than needed to exactly express the given information. This can be achieved by exploiting redundancy in the information. For example, consider a string of characters 'AAAAABBBBCCC' needs to be transmitted to a recipient. Transmitting the original string would require 96 bits, i.e. 8 bits per each of the 12 characters. Instead of transmitting the original string the source transmits an encoded version as 5A4B3C, that is the message would read: there are 5 letters A, 4 letters B, and 3 letters C. The total amount of bits being transferred is only 54 (assuming 32-bit architecture and numbers being represented as characters)

To describe the basic idea behind the score in the forecasting context [115] provide the following example. Consider 2 forecasters A and B, both having access to a forecasting distribution p used to forecast an outcome x. Assume that forecaster A has been told what the outcome is, and needs to share/communicate the result to forecaster B. The question is, what is the minimum bits of information A needs to communicate the outcome to B? Based on the optimal compression, A needs the minimum of

$$B_i = -\log_2(p(x_i)) \tag{2.9}$$

bits to describe to B that the i-th outcome of K possible outcomes has occurred.

What is the rationale for deploying the optimal compression vehicle? Consider that A attempts to communicate the outcome to B using two alternative forecasting distributions p and q. We can think of p and q as being two alternative encodings of the same outcome, just like in the above example with letters. If using p means that, on average, the forecaster A needs less bits to communicate the outcome to the recipient B, then A will choose p as her preferred encoding. In other words, if it takes less effort (bits) to transmit p rather than q, then p is a more useful description of the outcome. Consequently, by assigning a number of bits to each of the two alternative encodings the Ignorance selects the forecasting distribution that is, in the Information theoretic sense, 'closer' to the true distribution governing the outcomes.

Since Ignorance is based on the idea of optimal data compression, which is central to Information theory, it is a close relative of other Information-based measures, namely Shannon's entropy and also cross-entropy (Kullback-Leibler divergence). We describe Ignorance in greater detail in Section 2.4.2 and we also show mathematically how it relates to other information measures in Section 3.3.

2.4 Probabilistic forecast evaluation

A forecast evaluation is an important part of the forecasting framework (see Section 2.1.2). In this section we provide a description of the forecast evaluation process and discuss why it is important. We also provide a detailed description of an evaluation method of *Kernel dressing* (KD), which is our method of choice to be used in all applications below.

2.4.1 Background on forecast evaluation

Forecast evaluation aims to assess, the general quality of a forecast by comparing the forecasted system states to actual observed states. The forecast quality is quantified in terms of a *scoring rule* (score).

The forecast evaluation provides a forecaster with:

(a) *The ability to better understand and improve a forecast.* The forecast evaluation exposes sub-spaces of the model state space where the forecasting error is large. A forecaster then has the opportunity to analyze the sub-spaces and use the analysis to improve the forecasting model;

(b) *Justification of the cost of resources invested into a forecasting framework.* The assessment of a forecast performance in terms of score provides a measure that can be directly linked to the utility a forecast user. The utility can then be compared with the costs.

(c) *The ability to perform model selection.* The forecast score of competing models can be compared in order to select the best model.

The process of probabilistic forecast evaluation involves:

(1) construction of the forecasting densities,

(2) use of the densities to assign probabilities to the verifications,

(3) score calculations.

When evaluating a forecast, we are interested in a performance at a given leadtime. If we want to evaluate forecasting performance at the first leadtime, we collect forecasting densities of the first leadtime only and use those to evaluate the verifications. For the second leadtime, we use densities constructed for the second leadtime only etc.

2.4.2 Ignorance and other scoring rules

A scoring rule or a score, S, is a measure of a quality of forecasting performance of a model. In probabilistic forecasting, a score measures how much of a probability the forecasting density $p(\cdot)$ has assigned to the verification y. The forecasting performance can then be expressed as an average value of scores [14] collected over a number of forecast-verification pairs

$$E[S] = \frac{1}{N} \sum_{i=1}^{N} S(p_i(y_i)) \tag{2.10}$$

where N is a number of forecast-verification pairs, $p_i(\cdot)$ is the i-th forecasting density constructed using the i-th ensemble forecast x_i and y_i is the i-th verification.

There are a number of scoring rules currently used in forecast evaluation procedures, we list a few selected scores:

(1) *Ignorance*:

$$S(p(y)) = -log(p(y)) \tag{2.11}$$

(2) *Naive Linear Score*:

$$S(p(y)) = -p(y) \tag{2.12}$$

(3) *Proper linear score*:

$$S(p(y)) = \int p^2(y)dy - 2p(y) \tag{2.13}$$

(4) *Mean square error*:

$$S(p(y)) = \int (\hat{p}(y) - p(y))^2 dy \tag{2.14}$$

where $\hat{p}$ is an estimate or a forecast of density p. In Section 2.4.1 we have stated that forecast evaluation provides a forecaster with the means of model selection. An important property of a score directly related to model selection is *propriety.* Given two PDFs p and q, a score is proper if the following inequality holds

$$\int S(p(y))q(y)dy \geq \int S(q(y))q(y)dy. \tag{2.15}$$

Note that a score is strictly proper if the inequality $\geq$ is replaced by strict inequality $>$. In terms of the probabilistic forecasting, Eq: 2.15 can be interpreted as saying that a proper score should recognize when the forecasting density p differs from the 'true' density q that produced the verification y. Clearly, the best forecasting density for y is the one that actually generated y. Any other forecasting density should score less than the 'true' one. Proper scores therefore ensure that in the (rare) cases when one has to select between an imperfect forecast and a perfect forecast, the perfect forecast will be identified as the better one. This is indeed a useful property since in forecasting one tries to achieve the best possible forecast. When a forecaster has a perfect density (or model) available it makes sense to prefer it over any alternative forecasting density since, on average, the perfect forecasting model will produce better forecasts. In other words, there is no point in using an imperfect forecasting model when a perfect model is available.

Although the propriety property seems an obvious requirement, not all scores are proper. For instance, it can be shown [14] that the root mean squared error (RMS) is not a proper score. Using root mean squared error thus may lead the forecaster to select an imperfect model over a perfect one. The naive linear score is also not proper, while its augmented version, the proper linear score, is. So what are the potential consequences of using scores that are not proper? As before, when a forecaster in fact has a perfect model available, he/she would be foolish to use some alternative imperfect model instead. The perfect model will certainly deliver a better performance. A proper score helps the forecaster to detect which one of the alternative models is perfect.

It could be argued that in practice, a forecaster never has a perfect model to hand. In such cases, the impact of using an improper score may only be evaluated via some utility function which a forecaster aims to maximize. In Section 6.4, we aim to demonstrate this by showing how RMS may lead to selection of an inferior model.

It is important to realize that deployment of a proper score does not ensure that the best model is selected. For example, consider a case in which the forecasting model is perfect but the method used to construct the forecasting densities is flawed. Despite the model producing perfect ensemble forecasts the densities constructed from them will diverge from the true density, simply because the density construction is erroneous. If a proper score is applied to the (biased) forecasting densities it may fail to detect that the model is perfect. In other words, judged by the biased densities, the proper score may 'think' that it deals with an imperfect model.

In this work, the score of our choice is Ignorance [49, 115]. Forecasting performance in our applications is quantified by the expected Ignorance defined as

$$\mathrm{IGN} = -\frac{1}{N}\sum_{i=1}^{N}\log_2(p_i(y_i)) \tag{2.16}$$

For simplicity, we will call the expected Ignorance simply Ignorance from now on. Ignorance is a proper score. It is also a *local score*. Locality is another property of a score that means only the probability assigned to the verification y enters the scoring rule. An example of a non-local score is the proper linear score, which involves an integral over the whole domain of the verification y.

Note that Ignorance is a score of our choice not only because it is proper. Another good reason of using Ignorance is that it has direct links to other information measures, i.e. Kullback-Leibler divergence and Shanon entropy. We will study links between these measures in Section 3.3.

When comparing a forecasting performance of alternative models, the Relative Ignorance is a useful metric. Relative Ignorance is defined as

$$\mathrm{RIGN} = \mathrm{IGN}_A - \mathrm{IGN}_B \tag{2.17}$$

where A and B designate two alternative models being compared. In Relative Ignorance, model B, a benchmark, defines a zero skill reference forecast. Model A outperforms model B if the relative Ignorance is negative, i.e. below the zero line defined by the reference forecast of B. The lower the Relative Ignorance, the better the model A is relative to model B.

In subsequent analyses, a climatological forecast is often used as the zero skill reference. A forecasting model outperforms the climatological reference if its Ignorance is lower than that of the climatology, i.e. if Relative Ignorance of a model is negative.

2.4.3 Kernel dressing

Kernel dressing (KD) is a non-parametric method of turning ensemble forecasts into forecasting densities. For a given leadtime, the method utilizes multiple ensemble forecasts to construct forecasting densities. Kernel dressing is similar to *kernel density estimation* (KDE) [56, 80, 114, 120, 129], a procedure used to estimate a single density $\hat{p}$ given a sample from the unknown 'true' distribution p. Since KDE is a well-known method, and due to the similarities between KD and KDE, we first briefly describe a version of KDE, which will help to describe the KD estimator.

Given a sample $\mathbf{x} = x_1, \ldots, x_m$ from the density p a density estimate, $\hat{p}$, is constructed using a linear combination of m *kernel functions* $K(\cdot)$. Each kernel is entered at one of the m sampled data points $x_j, j = 1, \ldots, m$ and is assumed to have properties of a PDF. There are different forms of kernel functions [37, 105, 129], a frequently used one being the Gaussian kernel

$$K(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2}, \tag{2.18}$$

The kernel density estimate is given by

$$\hat{p}_\sigma(x) = \frac{1}{m\sigma} \sum_{j=1}^{m} K\left(\frac{x - x_j}{\sigma}\right). \tag{2.19}$$

where σ is the *bandwidth* or a smoothing parameter. The core issue in the kernel density estimation is how to set the bandwidth. The optimal bandwidth should be chosen so that the divergence of the estimate $\hat{p}$ from the true p is minimized, i.e. $d(\hat{p}, p) = \|\hat{p} - p\|$, where $d(\hat{p}, p)$ is some measure of the divergence. An obvious obstacle when minimizing $d(\hat{p}, p)$ is that p is not known, i.e. the divergence cannot be measured. To deal with the problem, a number of automated selection methods have been suggested, including crossvalidation or plug-in selectors [11, 20, 52].

Similarly to the KDE, the kernel dressing deploys a linear combination of kernels centered at a data point x_j, which in this case is an ensemble member of an ensemble forecast $\mathbf{x} = x_1, \ldots, x_m$ produced by the forecasting model. In its general form [15], the KD forecasting density can be expressed as

$$\hat{p}_\theta(y|x, \theta = \{\sigma, a, o\}) = \frac{1}{m\sigma} \sum_{j=1}^{m} K\left(\frac{y - ax_j - o}{\sigma}\right) \tag{2.20}$$

where y represents the verification, θ is a vector containing the parameters $\{\sigma, a, o\}$, σ is the bandwidth and the scaling parameter a is designed to rescale the ensemble forecast, while the offset parameter o shifts the location of the ensemble. In our applications, the kernel K is chosen to be a Gaussian kernel. We note that $\hat{p}$ is an estimate of the distribution of the system state at a given leadtime.

Similarly to KDE, the issue in KD is how to set the parameters $\{\sigma, a, o\}$. Since the main goal of forecasting is to produce a useful forecast, 'usefulness' being measured by a score, it is sensible to maximize the forecasting score

$$\arg\max_{\theta} S(\hat{p}_\theta) := \frac{1}{N} \sum_{i=1}^{N} S(\hat{p}_\theta(y)) \tag{2.21}$$

with N being a number of 'relevant' forecasts and $S(\cdot)$ a score of a choice, in our case Ignorance as given by Eq: 2.16.

The parameters σ, a, o are set simultaneously using all 'relevant' forecast-verification pairs. The KD procedure is applied at each leadtime, which means that the relevant forecasts-verification pairs are all those produced (observed) at the given leadtime. As a consequence, there is a different parameter vector θ_l at each leadtime.

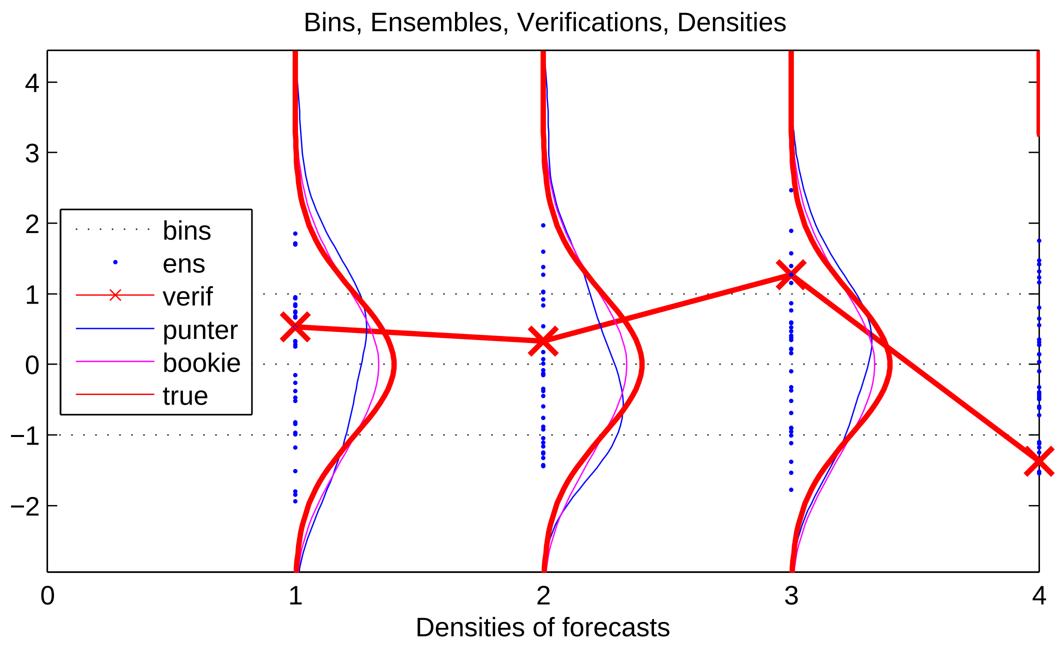

Figure 2.3: **Forecast distributions (Gaussian-like curves)** somewhat resemble the true p (red Gaussian-like curve). The punter uses the (blue) ensemble members to construct her estimate (blue Gaussian-like curve) of the true p. The bookie constructs his estimate (magenta) using climatological approach with all realized verifications (red crosses joined with red line) being included in his ensembles.

KD in the forecasting mode: In the forecasting mode, a model produces a forecasting ensemble x, but the verifications are not (just yet) available. To construct a forecasting density for a given leadtime l, we simply use the parameters θ_l as obtained in training. To obtain the forecasting density, the parameter values are plugged into

$$\hat{p}_{t+l}(y|x, \theta_l) = \frac{1}{m\sigma_l} \sum_{j=1}^{m_l} K\left(\frac{a_l \times x_{t+l,j} - o_l}{\sigma_l}\right) \tag{2.22}$$

where $\hat{p}_{t+l}$ is a forecasting density constructed using an ensemble $\mathbf{x}_{t+l} = \{x_{t+l,j}\}_{j=1}^{m}$. $\mathbf{x}_{t+l}$ is an ensemble generated by a model initialized at time t to produce a forecast for leadtime l.

Apart from the quality of the ensemble, there are two other factors influencing the quality of the KD forecasting density. The first is the number of 'relevant forecast', i.e. the size of the forecast-verification archive at a given leadtime. The second is the size of the forecasting ensemble. In real-world applications, both are often restricted in size, which is a fact that needs to be appreciated when constructing the forecasting densities. The consequences of these restrictions, as well as some properties of the KD estimator, will be discussed in Chapter 3.

2.4.4 Crossvalidation and subsampling

Although the score provides information about the forecasting performance of a model, it may be useful to evaluate robustness of the information. This is often very useful when the forecast-verification archive is limited which may lead to high sensitivity to outliers. In such cases, it is advisable to crossvalidate the score. In fact, in this work we always perform crossvalidation and report quantiles of the crossvalidated scores.

For cases when the training set is small, i.e. Chapter 4, we use a leave-one-out type of crossvalidation [107] as k-fold crossvalidation would not be feasible. For larger training sets, such as for those used in Chapter 6, the 2-fold crossvalidation is performed.

2.5 Seasonal forecasting

Section 2.2.3 has provided some preliminary rationale behind the benchmarking of forecasting models. Since in later chapters we will use a benchmarking model to evaluate performance of seasonal weather forecasting models, our aim here is to provide some background on seasonal weather forecasting.

Weather forecasters aim to produce forecasts that are useful at both short and long leadtimes. Long-term weather forecasts are of great importance to industries including agriculture, health, energy, insurance, and many others. Take agriculture for example. Knowing 6 months ahead whether there will be a cold summer, a farmer may decide what type of plants to sow. Or take insurance for example. Knowing 6 months ahead that there will be a cold summer in the Atlantic may provide crucial information about the number of hurricanes during the hurricane season. A reliable forecast could therefore decrease the expected costs, e.g. damaged oil-facilities, and lead to a reduction in insurance premiums. In response to such demands (and for many other reasons) weather forecasting institutions around the world aim to extend the usefulness of their forecasts to seasonal (up 6 months), annual, and even decadal time scales.

2.5.1 Seasonal forecasting and models

Seasonal weather forecasting is concerned with producing forecasts at the time scale of months, with 6 months horizon often being the leadtime of choice. Given that achieving a good forecast several days ahead is a challenge, can we expect a long term forecast to be useful? The argument for a useful seasonal forecast is based on an important observation that the conditions of the lower boundary of the atmosphere tend to have a long memory, i.e. are rather persistent and change only gradually. The lower boundary conditions include for example sea surface temperature (SST), soil moisture, or snow cover, and due to the persistence, they can be forecast at time scales of weeks or even months.

The SST plays a prominent part in seasonal forecasting. This is because the temperature changes in oceans surface layers have a profound impact on global weather conditions. Since changes in oceans propagate at a much slower pace than changes in the atmosphere, the SST changes are predictable at longer time scales making the long term weather forecasts feasible.

Since seasonal forecasting is concerned with both the ocean as well as the atmosphere, the modeling requires coupling of an ocean model with an atmospheric model. The resulting coupled model, also called the coupled atmosphere-ocean general circulation model (AOGCM), is then used to produce global weather forecasts. To generate the forecasts, GCM must first be initialized with some initial state of the atmosphere and ocean. The initial state just represents a starting point from which the model starts integrating equations for fluid motion forward in time. The final state of this integration represents a forecasted future state of the atmosphere (or ocean). For a more detailed description of circulation models see, for example, [158]

2.5.2 Long-term patterns and regions of importance

Throughout history, scientists have identified several climate patterns that have a profound impact on the global weather. The patterns are measured in terms of climate indices which include:

- El-Niño - Southern Oscillation (ENSO), a global pattern of irregular warming of sea surface (increase of SST)

- Maden - Julian Oscilation, an equatorial traveling pattern of anomalous rain
- North Atlantic Oscillation - the difference of normalized sea level pressure between the Azores and Iceland, with positive difference signaling stronger than average westerly winds over the middle latitudes.

In terms of seasonal weather forecasting, the most important climate event is the ENSO, the strength of which is measured by the Nino3.4 index. The Nino3.4 index is the average sea surface temperature anomaly taken over a region in the south pacific within the box defined by N:5.0, S:-5.0, E:-120.0, W:-170.0. The sea surface temperature (SST) in this region is important for the El Niño or La Niña climate events [144]. The El Nino event is classified if the 5-month moving average of the Nino3.4 index exceeds +0.4°C. The La Niña event is classified if the Nino3.4 index is below -0.4°C.

Another important climate condition (not listed above) is the level of SST in the so-called Main Development Region. The MDR is an area covering mid latitudes of the Atlantic with the coordinates N:10.0, N:20.0, W:20.0, W:-80.0 and the sea surface temperature in this region plays a crucial role for the hurricane formation in the Atlantic basin. Since hurricanes have a great potential to inflict significant harm on US cities and energy facilities located on and off the south-east US coast a good seasonal forecast of SST in MDR is crucial.

Due to the significant impact on weather and its impact on large numbers of people, as well as industries, we are interested in the ability of weather models to provide useful weather forecasts in both the MDR and Nino3.4. In Section 4.3 we evaluate and benchmark the ability of ENSEMBLES [152] models to forecast the SST in the two regions.

2.5.3 The ENSEMBLES models

The ENSEMBLES [36, 57, 152] is a 5-year program funded by the European Commission, which aims to develop an ensemble prediction system for seasonal forecasting. The ensemble prediction system is based on 5 state-of-the-art coupled atmosphere-ocean circulation models developed by major European weather forecasting institutions. The models taking part in the project include [31, 152]:

- IFS/HOPE, a model operated by the European Centre for Medium Range Weather Forecast (ECMWF)
- ARPEGE/OPA, operated by Meteo-France
- GloSea, DePreSys_IC, both operated by UK Met Office
- ECHAM5/OM1, operated by IfM-GEOMAR Kiel
- MF, operated by Euro-Mediterranean Centre for Climate Change

The ENSEMBLES models were used to generate hind-casts (forecasts of past observations) at seasonal-to-annual horizons. The hind-casts consist of nine-member ensembles. The initial states of the atmosphere (and the verifications) are taken from the ERA-40 reanalysis [85], a dataset consisting of global atmosphere and surface conditions over the period of 1957 - 2002 constructed by the ECMWF.

The ENSEMBLES models were used to generate forecasts over the period of 1960 to 2001. There were 4 different forecast initializations: February, May, August, and November. The February, May, and August initializations yielded 7-months-ahead forecasts, while the November initialization forecasted 14 months ahead.

The models used in ENSEMBLES are high-resolution Global Circulation Models (GCM). All the involved models are expensive to build, complex to deploy, and require large computational resources, often supercomputers. The cost of construction and operation in terms of both the human and computational resource, is considerably large. The size of the invested resources is expected to be outweighed by the forecasting performance.

The forecasts produced by the ENSEMBLES models represent an example of a periodically forced system. In fact, all the geophysical processes on Earth are of this kind. In the periodically driven systems we are looking at phases and periods, which translates to seasonalities in the statistical sense.

In this thesis, we will re-evaluate the ENSEMBLES models using Kernel dressing and Ignorance (see Chapter 3) and benchmark their performance using our proprietary benchmark suggested and described in Chapter 4.

2.5.4 Multi-model ensembles in seasonal forecasting

In seasonal forecasting, a multi-model ensemble (MME) is a combination of ensembles of individual forecasting models, often used to improve the forecasting skill of ensemble forecasts. MME accounts for uncertainties due to misspecification (systematic model errors) of individual models [151] by reducing overconfidence of the individual ensembles. In particular, since the multi-model combines ensembles of several models, the resulting grand-ensemble has larger spread than any of the individual ensembles. Ensembles with larger spread are then more likely to capture a verification and hence improve the skill of the ensemble forecast. Indeed, it has been shown that a multi-model ensemble outperform the skill of individual models as measured by Brier score, ROC statistics, and RMSE [30, 73].

In [152], the skill of the ENSEMBLES models was evaluated using a multi-model ensemble (MME). The MME was constructed by combining ensembles of all five ENSEMBLES models, where the individual ensembles were equally weighted (see [30, 73] for other weighting approaches). In the Nino3 region, at all leadtimes, the MME achieved substantially smaller RMSE when compared to the individual models. It was also shown, that the ENSEMBLES MME significantly outperformed the MME of the previous generation models, DEMETER [104], as evaluated by Brier skill score, RMSE, and Anomaly correlation.

While MME is a useful concept, the improved skill due to increased spread of an ensemble comes at the cost of diluted performance when an individual model actually captures the verification. Consider that one of the individual models has a small spread but also a very small bias; this model almost always captures a verification and does so with a high precision, i.e. many ensemble members will be very close to the verification. When comparing the MME with the small-spread/small-bias model, the MME might not outperform the model. While both the MME and the individual model will almost always capture the verification, the precision of the MME will suffer. Due to its large spread, the MME will have many ensemble members far away from the verification.

The MME is not the only way of improving performance of ensemble forecasts. In Section 2.1.2 we mentioned forecast post-processing, a part of the forecasting framework concerned with bias removal. Bias removal may also involve inflation of an ensemble spread. Indeed, in Section 2.4.3 we discussed Kernel dressing (KD), which is designed to produce forecasting densities that account for model error by implicitly increasing an ensemble spread. When forecast skill is evaluated via KD, the MME may not yield a significant advantage over individual models, simply because in KD the spreads of individual ensembles are already large enough. It is therefore important to note that while MME may be helpful in some evaluation approaches, in others, i.e. Kernel dressing, it may not yield additional value.

2.5.5 Benchmark models and their value

Since an absolute value of a score is not very informative, forecast evaluation should be performed in relative terms. A forecaster is more often interested in which model generates a 'better' forecast rather than what is an exact value of a score. Relative performance evaluation gives a much better perspective of a model's forecasting performance. A benchmarking model is a useful tool when evaluating the relative performance of a forecasting model as it defines a zero skill which a forecasting model is expected to outperform.

A good benchmarking model should possess several properties. First of all it should be robust so that it sets a clear and understandable performance threshold that would distinguish a good model from a bad one. It also should be easy to operate since investing a lot of time and effort just to run a benchmark model is not efficient. Finally, it should be simple. If a forecasting model is unable to beat a simple model, it is not useful and should be abandoned.

In probabilistic forecasting, the obvious choice for a benchmark is climatology. Indeed we use unconditional and conditional climatology benchmarks throughout this work. Although climatologies are natural choices for a benchmark, we can use more complex benchmarks. Simple statistical models with a better forecasting performance when compared to climatology can be constructed and contrasted with more complex forecasting models. In Chapter 4 we use a version of a nearest neighbor model to provide a benchmark to state-of-the-art seasonal weather forecasting models of the ENSEMBLES project.

We note that at the time of the experiment, due to technical issues within our database infrastructure, forecasts of only 4 ENSEMBLES models were available. Hence the benchmarking results presented below exclude the MF model operated by the Euro-Mediterranean Centre for Climate Change. We note, however, that the goal of the exercise is to show usefulness of our benchmarking model, for which purpose, we believe, the four ENSEMBLES models represent a sufficient sample.

2.6 Correcting a model error

In practice, forecasting models are always an imperfect representation of reality. Consequently, forecasts produced by a model always suffer from errors that stem from the model imperfections. The model error presence applies to both stochastic as well as deterministic models. Stochastic forecasting models are almost always built using noisy data, and hence are prone to over-fitting [32, 56, 141]. Both dynamic and parametric stochastic models are subject to model mis-specification. In addition the dynamic models are exposed to an error due to assimilation procedure [137], a procedure used to estimate the current state of a system.

The main impact of model error on the quality of a forecast is mostly due to model integration, as an arbitrarily small initial error accumulates at each integration step. While the sources of the error may differ in both statistical and dynamical systems, the error propagation and accumulation has the same consequence - degradation of forecast utility.

Reducing the impact of model error inevitably leads to an improved forecast. Bias correction has therefore become an important part of forecast post-processing in many fields. In Chapter 6 we suggest an iterative method of error correction and contrast it with an alternative method introduced in [60].

2.6.1 Systematic part of a model error

A necessary condition for any error correction procedure to be successful is that an error in a forecast contains a systematic part. Should forecasting error be purely random, any systematic approach to limiting the error would inevitably fail. The good news is that in practice, forecasts indeed contain systematic parts of an error; often the systematic part is due to the model error. For example, in the case of model error due to misspecification, a model will be 'incorrect' in a similar way each time a similar initial condition re-occurs. The same applies to the overfitting. A stochastic model will be incorrect in a similar way each time its prediction starts from a similar initial state.

If it were possible to detect a pattern of forecast imperfections, it should also be possible to either link the imperfections back to the model and correct the model error, or to build a correcting layer that would, at least to some extent, correct the imperfections and improve the forecasts. Linking the imperfections back to a model may often prove difficult. For large models consisting of hundreds or thousands of equations, it might be very difficult to exactly pinpoint the source of the error. Constructing a correcting layer on top of the forecasting model seems more pragmatic. In Chapter 6 we build on the idea of a correcting layer, and construct a two-stage procedure, which substantially reduces the forecasting error of a first-stage model.

2.6.2 Iterative vs. direct predictors

The accumulation of a forecasting error over a long horizon closely relates to the type of forecasting approach. A frequent forecasting approach is to use an iterative model, where a predictor with a short forecasting horizon is constructed and then iterated to build a forecast over a longer horizon. An alternative approach is to build a 'direct' predictor, i.e. a forecasting model with a forecasting horizon long enough to obtain a long term forecast with only a single iteration.

In both approaches, the forecasting error grows with each iteration as the forecasting model propagates both the system state and the model error. It has been shown [38], that under some conditions, the iterative approach achieves a lower forecasting error. However, in cases where the forecasting model is imperfect and the data are noisy, the conditions are not met and the theoretical result does not hold. In such cases, the direct approach seems to be a better choice as it undergoes only a single iteration and hence the amplification of a model error is smaller.

The problem with direct predictors is that they are often difficult to construct. Iterative predictors are therefore of interest. Consequently, methods involving forecast corrections can be very useful.

2.6.3 Correction models

Correction models may take a number of forms. A frequently used class of error correction models is based on a two-stage approach, where in the first stage a predictor is used to generate a forecast and then in the second stage, a corrector is applied to correct the forecast. Typically, the first stage produces a forecast over the whole forecasting horizon. The corrector is applied only after the complete forecast has been generated. This approach was used to develop the $\Psi\Phi$ corrector [60], which has been been successful in producing longer-term forecasts when contrasted with a single-stage procedure.

Regarding the modeling approach, it is intuitive to design correction models as stochastic. Dynamic correctors may be constructed, but since knowing the dynamics of the model error implies that we know where the dynamics of the first stage model is wrong, rather than constructing such a corrector we could simply improve the model. Also, constructing a dynamic corrector for stochastic models does not make sense. Knowing the dynamics of the error implies that we have some knowledge about the dynamics of the system. Consequently, a stochastic model of the system could and should be replaced by a dynamic model with a stochastic corrector.

By definition, correction models can be successful only in the presence of systematic errors. The larger the systematic error, the more likely a correction model is to succeed in reducing it. Consequently, in cases when the first-stage model is a good predictor the correction model is not likely to further improve the forecasting performance.

An advantage of a corrector is that if the first stage model does not suffer large systematic errors, the corrector does not degrade the first stage forecast, i.e. ‘it does not make things worse’. Given its stochastic nature (see Section 2.6.3) the correction model estimates the expected value of the forecasting error, which is zero in the absence of systematic errors. The second stage therefore does not alter the first stage forecasts if they are already ‘good’.

Our method presented in Chapter 6 is constructed as a two-stage procedure but with an important feature that makes it distinct from other two-stage procedures. The corrector we present works on an iterative basis. This means that the corrector is applied after each iteration of the first stage model. In our method, the **corrected** single step forecast serves as an input into the next iteration of the first stage model.

2.6.4 Applications and limitations of correction models

Although in general correction models may be applied to both low-dimensional and high-dimensional models we note that our approach has some limitations when applied to a high-dimensional setting. Recall that the iterative nature of our approach requires a correction to be made after each integration of the first stage model, so that the 'corrected' first stage forecast can be input into the next model integration. In other words, if there are T integration steps of the first stage model, there also must be T corrections, giving the total amount of 'operations' as $2T$. For non-iterative correctors that are applied only after all integrations of the first stage model have been performed, the total number of operations is simply $T + 1$, T for first stage integrations and one for the single round of forecast correction.

The obvious drawback of the iterative approach is that more operations take more time. Given finite computational resources this may lead to limitations in the length of the predicted horizon. In reality, however, the severity of the limitation depends on the complexity of the correction model. Since correction models are often stochastic, their execution time is typically moderate when compared with the integration step of a large scale deterministic model. Thus 'simple' correction models with moderate execution time still may be applicable to high-dimensional systems. Non-iterative correctors do not impose any limitations on the first stage, since they are applied only once.

In this work, we apply the iterative approach to low-dimensional systems only. This is mainly because we do not have any high-dimensional model available. We note that operating a high-dimensional deterministic model is a research task in its own right and is beyond the scope of this work.

Regarding applications in the low-dimensional setting, there are a number of potential applications in fields such as economics, finance, transport, but also physics. For example, our approach can be applied to low-dimensional convection models, e.g. Lorenz systems [81, 83], rotating annulus [111, 161], but also to models of inflation, stock prices, or transport peak predictions.

Chapter 3

Properties of Kernel Dressing

In this chapter we aim to derive and illustrate some of the properties of Kernel Dressing (KD) [15, 58], a method used to construct and evaluate forecasting densities [48, 156] described in Section 2.4.3. We also study the properties of Ignorance [49, 115] (see Section 2.4.2) when deployed within KD.

We first show that Kernel Dressing substantially differs from the Kernel Density Estimation [11, 20, 129], an important observation that has often been overlooked. We then provide a description of the KD properties and describe the role of Ignorance within KD [14, 115, 116]. We highlight the importance of treating KD as a standalone estimator and argue that the (desirable) properties of Ignorance do not ensure a high-quality forecasting density. We also show that under a perfect model scenario, (PMS) [133] Ignorance minimization within the KD implicitly minimizes the Kullback-Leibler divergence.

In Section 3.2 we demonstrate the behavior of KD numerically both under the PMS and outside it. We study the convergence of the KD estimator and demonstrate the impact of the ensemble size on the quality of a forecasting density. We also discuss the impact of the size of the forecasting archive and look at the potential trade-off between size of the archive and ensemble sizes.

In Section 3.3 we show important links between Ignorance and other well-established information measures, i.e. Shannon's entropy [124–126] and Kullback-Leibler divergence [74, 75]. In particular, we show that Ignorance under Kelly betting [72] leads to optimal betting. By setting the Ignorance within the broader context of Information theory, we aim to demonstrate its relevance as a measure for probabilistic forecasting.

In this chapter, the following are new contributions:

- clarifying the distinctions of KD and KDE
- disentangling Ignorance and the KD framework
- showing that, under PMS, properties of KDE are applicable to KD
- novel analysis of numerical properties of KD under PMS and perfect ensembles
- new analysis of numerical properties of KD under IMS
- novel analysis of numerical properties of KD under varying ensemble size
- demonstrating that increased archive size leads to more robust estimates, but does not reduce IGN/KL-divergence
- clarifying analogy of the Ignorance, KL-divergence, and Shannon's entropy under the Kelly betting framework

3.1 Contrasting Kernel Dressing

In this section, we describe the basic framework to be used in studying the properties of KD [15]. In particular, we describe the PMS scenario [65, 133] which will enable us to demonstrate how the forecasting densities converge to the system density. We also comment on differences between KD and the pseudo-likelihood based KDE [52, 129]. Although the difference between the two might seem superficial, we argue that the two approaches differ substantially. Finally, we derive an analytical result showing that the Ignorance deployed within the KD approach implicitly leads to minimization of the KL-divergence.

3.1.1 Perfect model testbed

In Section 2.1.2 we have classified an evaluation method as a particular stage of a forecasting framework. We have also argued that all stages of the forecasting framework must perform well in order to obtain a good forecast. A good performance of the evaluation method is of particular importance. Should the evaluation be unreliable, it would be impossible to determine if poor forecast quality is due to the model, observation errors, or flawed evaluation.

The evaluation method of our choice is Kernel Dressing; how can we assess the quality of KD? We can assess KD quality by studying bias in PMS with a perfect initial condition ensemble. We argue that if KD is unbiased under PMS, it can be considered an appropriate evaluation method. Our rationale for this approach is as follows. When discussing PMS and IMS in Section 2.1.3, we stated that all real-world data exercises fall under IMS. We have also said that under IMS, the forecasting model always produces an imperfect forecast. Considering a forecasting framework with a flawed evaluation method, one cannot rule out the evaluation method being a reason behind the forecast imperfection; under IMS the forecasting distributions are imperfect, whether KD is flawed or not. However, under the PMS, the system and the model coincide as there is no model error involved, hence only in PMS are flaws in the evaluation method detectable.

To test KD, we combine PMS with a perfect ensemble (PE) (see Section 2.1.3). Under the PMS/PE environment, we expect the model's ensemble members to be distributed in the same way as the system states. Assuming the model's forecasting density is $p(x)$ and the states (verifications) produced by the systems are distributed according to $q(x)$, we expect the model distribution to converge to the system, i.e.

$$\lim_{n\to\infty} p_N(x) = q(x) \tag{3.1}$$

where N denotes number of samples (ensemble members). We note, however, that the forecasting model merely produces ensemble members; the forecasting distribution is yet to be obtained via KD. The KD therefore provides estimates, $\hat{p}_N$, of the model's forecasting distribution p_N. For KD to be considered a good evaluation method, the estimate $\hat{p}_N$ must converge to p_N, which in turn converges to the system distribution q as shown in Eq: 3.1. So we can write that

$$\lim_{n\to\infty} \hat{p}_N(x) \;\;=\;\; p_N(x) \tag{3.2}$$

Should KD fail to meet Requirement 3.2, it would be considered a biased evaluation method.

In Sections 3.1.3 and 3.2, we present numerical results showing that KD is indeed a good estimator under PMS/PE. The analyses are examined in two different settings. In the first, the model and system are simply Gaussian distributions. In the second, we choose the model and the system to be the logistic map [91]

$$x_{t+1} = rx_t(1 - x_t). \tag{3.3}$$

In the Gaussian setting, there is no specific rule for obtaining the initial conditions. To produce the ensemble forecast, we sample M ensemble members from the inverse of the assumed noise model [33]. To obtain verifications, we sample once from the same Gaussian distribution. For the second case, we select M initial conditions $\mathbf{x}_t = \{x_1, \ldots, x_M\}$ by uniformly sampling the support of the logistic map $x \in (0, 1)$. We then iterate the initial conditions with the logistic map l times yielding a leadtime l ensemble forecast. To generate the verification, we repeat the same process with only a single initial condition. In both cases the procedure is performed N times giving N different ensemble forecasts or forecast initializations.

The quality of the KD densities is measured by the closeness of the forecasting density to the system climatology. The metrics used are Ignorance and the KL-divergence.

3.1.2 Distinguishing Kernel dressing and Kernel density Estimation

Looking at Equation 2.19 it may seem that KD and KDE are almost identical. The two concepts differ substantially, however, both in their assumptions and in the way their parameters are determined. We repeat some of the differences cited in [15] but also add new important items to the list.

(1) Probably the most important difference is that KDE assumes that the samples used to construct the density estimate are drawn from the target density. KD does not.

(2) Since the samples in KDE are drawn from the target density, KDE is expected to provide an asymptotically unbiased estimate. In KD, we never expect to obtain an asymptotically unbiased estimate as there is no 'target' density.

(3) In KDE, we have number of samples from the target density, while in KD we only have one - the verification.

(4) If pseudo-likelihood is deployed to estimate the KDE bandwidth, the likelihood is maximized over the whole sample. In KD with Ignorance being used as a measure, we only minimize the Ignorance at a single point (the locality property of Ignorance).

(5) The Ignorance in KD is minimized over many distinct *forecast-verification* pairs. In fact, we have an ensemble forecast and related verification at each forecasted time t. The forecasts and the verifications may or may not be independent. Except for special cases, the distributions at the different times t are expected to be independent. In KDE there are no independent distributions across different times t; in fact the time dimension is not relevant at all, i.e. in KD samples are assumed to be obtained at a single time t.

(6) The offset parameter in KD is meant to remove a potential bias of a forecasting model, i.e. KD expects to encounter biases and implicitly removes them.

(7) The forecasting density coming out of the KD is often a mixture of the model density, and the climatological distribution. This relates to point (2) above in the sense that there is no target density. Also, since the 'target' density of KDE is replaced by a mixture of model and system distributions, we cannot use the concept of unbiasedness.

(8) All the KD parameters are determined simultaneously so that the choice of bandwidth σ cannot be separated from the offset o or the blending parameter α.

To our knowledge, the items (2), (4), (5), and (7) are new.

3.1.3 Ignorance as a minimizer of Kullback-Leibler divergence

Ignorance is often a measure of choice when KD is used. There have been several efforts [14, 48, 115] to show some of the important properties of Ignorance. In this section we aim for another contribution, as we argue that under some assumptions Ignorance is an unbiased estimator for *Kullback-Leibler* (KL) divergence [74]

$$D_{\mathrm{KL}}(p\|q) = \sum_k p(k) \log \frac{p(k)}{q(k)}. \tag{3.4}$$

where p and q are two possibly different densities.

To show this, we use a simple trick. If we can show that under some assumptions KD becomes equivalent to a pseudo-likelihood based KDE, we can then show that minimizing Ignorance leads to minimization of KL-divergence. We adopt two assumptions:

1. PMS, i.e. the forecasting model is identical to the system,
2. PE, i.e. the initial conditions have the same underlying distribution as the verifications (see Section 2.1.2).

Recall that the in-sample forecasting density at a given time t and a given verification point y_t is

$$\hat{p}_t(y_t) = \frac{1}{n\sigma}\sum_{j=1}^{m} K\left(\frac{x_j - y_t}{\sigma}\right) \tag{3.5}$$

Substituting in Eq: 2.16 and dropping the σ notation we get

$$\begin{aligned}
\text{IGN} &= E[\frac{1}{N}\sum_{t=1}^{N} -\log \hat{p}_t(y_t)] && (3.6)\\
&= E[-\log \hat{p}(y)] && (3.7)\\
&= E\int -p(y)\log \hat{p}(y)dy && (3.8)\\
&= E\int [(-p(y)\log \hat{p}(y) + p(y)\log p(y)) - p(y)\log p(y)]dy && (3.9)\\
&= E\left[D_{\text{KL}}(p\|\hat{p})\right] - \int p\log p dy && (3.10)
\end{aligned}$$

which shows that minimization of Ignorance implicitly minimizes KL distance, up to the constant $\int p\log p$. Assuming the above along with unbounded support for p and $K(\cdot)$, while increasing the ensemble size and number of forecast-verification pairs, asymptotically leads to

$$E\left[D_{\text{KL}}(p\|\hat{p})\right] = -\int p\log\hat{p} + \int p\log p \tag{3.11}$$

where the first element on the right hand side represents *cross-entropy* and the second represents *entropy.* If the two distributions coincide, $p = \hat{p}$, the Kullback-Leibler divergence reduces to zero.

Eq: 3.11 has important consequences. It says minimum Ignorance is obtained as the KL-divergence of the forecasting density $\hat{p}$ and the system distribution p. This implicitly suggests that under the PMS/PE scenario, the forecasting densities will approach the system density. Under PMS/PE, we have a model which coincides with the system, hence the model generated states that (in the limit) have an identical distribution to those produced by the system. The only difference between the model and system density as measured by Ignorance will be due to the (small) sample size of the ensemble and verifications. As we increase the ensemble size to infinity and compare ensembles with verifications over an infinite horizon, two things happen. First, the small sample distribution of the ensemble will approach the underlying distribution of the model and the small sample distribution of verifications will approach the system distribution. Since we deal with a perfect model, in the limit the two distributions must coincide, i.e. the Kullback-Leibler divergence will reach its minimum (defined by the value of entropy $\int p \log p$). This result is indeed evidenced by the numerical analysis in Section 3.2.1.

3.2 Numerical analysis of Kernel Dressing properties

Recent works regarding Ignorance have focused on the properties of the measure [115] [48] [14] [15]. One important output of this research was that Ignorance was shown to be a proper score, i.e. it has the ability to distinguish a perfect model forecast from an imperfect model forecast. However, using

Ignorance in an evaluation procedure does not guarantee that the best model will be selected. This is because the Ignorance deals with densities that are constructed via some density construction method, KD in our case. In fact, it is both the KD and the scoring rule that impact on the quality of the forecasting density and consequently the model selection. In this section, we study properties of KD using Ignorance as a measure.

The following analyses will rely on both the perfect and imperfect model scenarios, with a perfect ensemble also having a role (see Section 2.1.3). We aim to show that KD is an unbiased estimator. It is only possible to show this under PMS. Since, in the real world, forecasting is always under IMS, we will also be looking at KD properties outside PMS. In both scenarios, we use simple Gaussian settings. To see KD behavior under a more complex setting, we also include analyses using the logistic map.

3.2.1 Convergence in perfect model scenario

We utilize the test-bed described in Section 3.1.1 to show numerically that under a PMS/PE scenario, the KD evaluation method yields forecasting densities that converge toward the target density.

We begin by showing a numerical example supporting our claim in Section 3.2.1. We use the Gaussian PMS described in Section: 3.1.1 with $\mu = 0$ and $\sigma = 1$ and produce 2,048 forecasting densities by minimizing Ignorance within KD. Since we operate under PMS, the offset o and scale a parameters become irrelevant. In Fig: 3.1 we show values of Ignorance and the KL-divergence plotted against the bandwidth parameters σ for a model with an ensemble size of $M = 512$. For this particular archive of 2,048 ensemble-verification pairs, the value of σ_{IGN} that minimizes Ignorance is equal to the value of σ_{KL} that minimizes the KL-divergence.

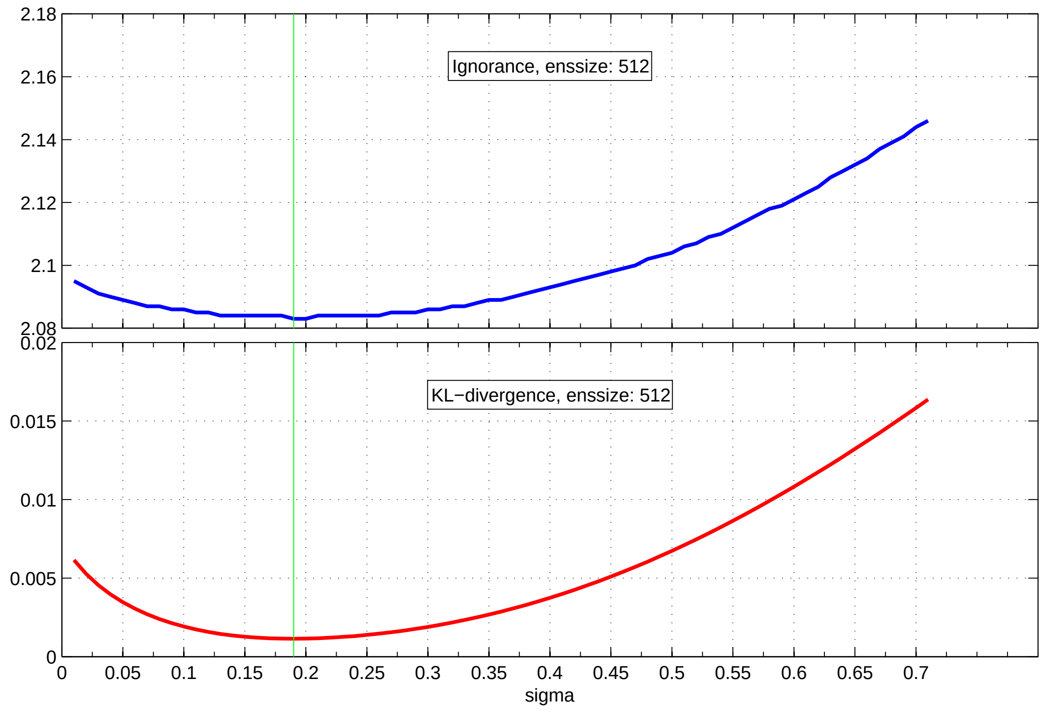

Figure 3.1: **Ignorance minimizes KL-divergence (PMS):** Ignorance as a function of the parameter σ (top) calculated for a particular sample for a model with 512 ensemble members. The minimum value of Ignorance (green line) coincides with the minimum of the KL-divergence (bottom) at the value of $\sigma = 0.18$.

The numerical result in Fig: 3.1 supports our previous claim, but only with a single archive. To obtain a more robust result, we generate 128 such archives using the PMS test-bed. For each archive we locate σ_{IGN} and σ_{KL} and plot them in Fig: 3.3. Although across different archives the optimal bandwidths may vary significantly, within a given archive the σ_{IGN} (green) and σ_{KL} (red) are frequently close or very similar.

The values of the σ_{IGN} (green) and σ_{KL} are only expected to be equal asymptotically. In small archives with forecasts of limited ensemble size, they are expected to be close but not necessarily equal. Increasing the ensemble size

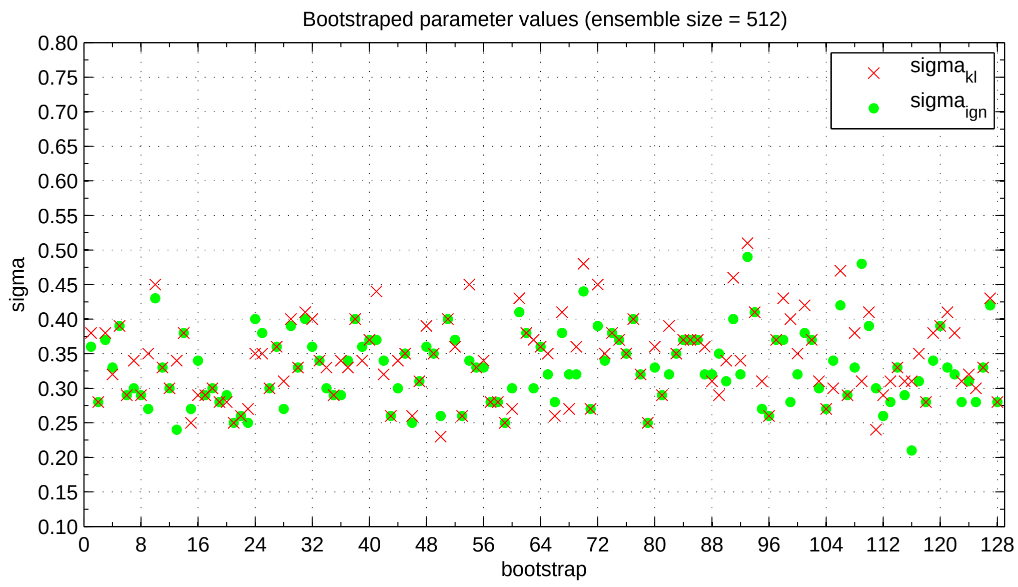

Figure 3.2: **Resampled parameters for IGN/KL:** For 128 samples, the kernel width σ minimizing KL-divergence (red crosses) is centered closely around 0.35. The σ that minimizes Ignorance (green dots) oscillates around the same value, supporting the suggestion that minimization of Ignorance implicitly leads to minimization of KL-divergence.

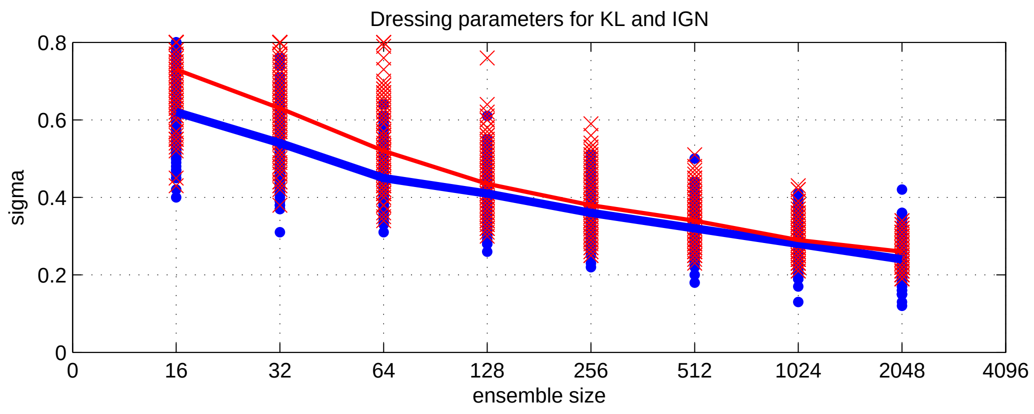

Figure 3.3: **Resampled parameters for IGN/KL:** A different view of the result in Fig 3.3. For each ensemble size, we plot all 128 samples of σ minimizing both the Ignorance (blue) and KL-divergence (red). As the ensemble size increases the two sets of σ converge.

should cause the distance between the bandwidths to decrease. In Fig: 3.3 we reproduce the exercise for 8 different ensemble sizes while keeping the size of the archive constant at 2,048 forecasts. The ensemble size is doubled every time, beginning with 16 and ending up with an ensemble size of 2,048 ensemble members. As in Fig: 3.1, we generate 128 archives for each ensemble size and plot $\{\sigma_{IGN}, \sigma_{KL}\}$ and their medians. We see that with small ensemble sizes the medians of the two bandwidths are rather far apart. As the ensemble size increases, they become closer, indicating the minimum of the Ignorance function is found in the same part of the parameter space as the minimum of the KL-divergence.

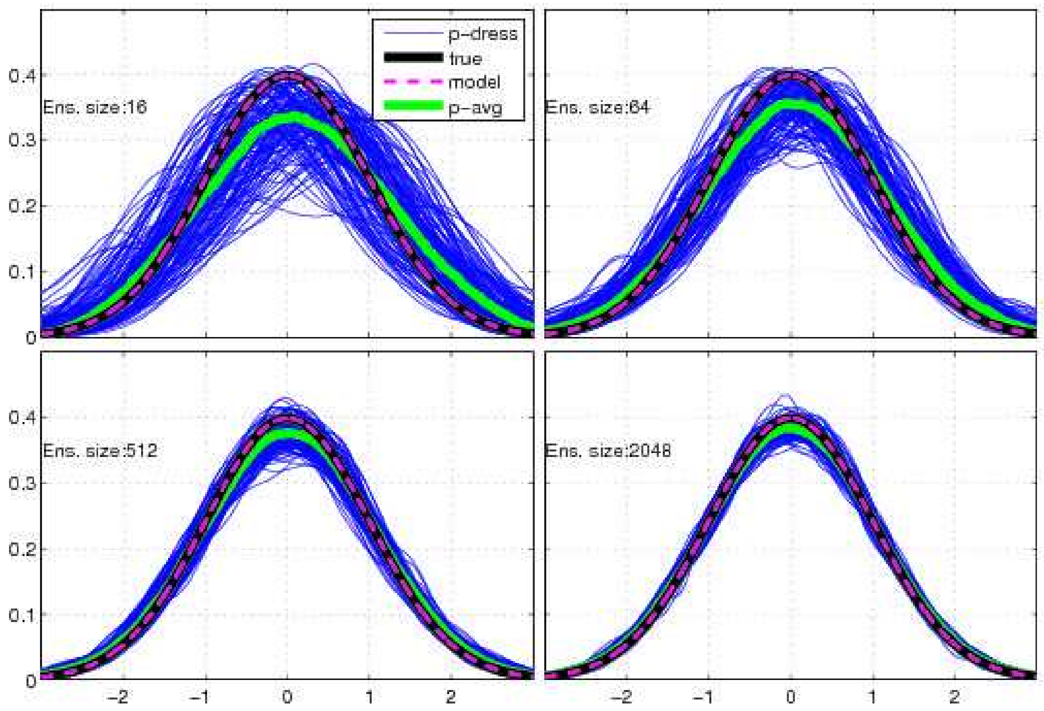

Figure 3.4: **Gaussian PMS:** The model (magenta dashed) as well as the system (black) are $N(0,1)$ distributions. Both KD probabilities (blue dots) and their average (green dots) over 128 samples converge to the system distribution as the ensemble size increases.

Another effect to observe in Fig: 3.3 is that both the bandwidths decrease when the ensemble size is increased. The intuition behind this is that a larger ensemble leads to a forecast with 'better probabilities'. In other words, a larger ensemble is equivalent to a larger sample, and a larger sample always provides a better description of the distribution it comes from. A consequence of having more ensembles is that the kernel dressing procedure can form sharper kernels, i.e. the optimal bandwidth of the kernels become smaller. Contrarily, for small ensembles the optimal bandwidth must be larger to compensate for areas that are not well covered by ensemble members. Having plenty of ensemble members therefore means that the kernels may be sharper, hence the narrower bandwidth.

Since we have numerically shown that under the Gaussian PMS/PE, the Ignorance minimizes KL-divergence, we can move on to look at the actual forecasting densities produced by KD. We show the 128 forecasting densities (blue) in Fig: 3.4 along with their median (green) and the system (black) and model (magenta) PDFs for 4 different ensemble sizes. For the small ensemble size of 16 members, the forecast densities do not estimate the system density well. For 2,048 ensemble members, we obtain a much better result; the densities converge toward the system as we increase the ensemble size.

For completeness we show bias and variance of the 128 forecasts as a function of ensemble size in Fig 3.5. The result just confirms the finding pictured in Fig: 3.4 that both bias and variance decrease with the ensemble size. Note that the bias is calculated as the expected value of the difference of the 'average' density (green) from the true system distribution (black).

What is the behavior of KL-divergence and Ignorance during convergence? In Fig: 3.6 we plot values of both as a function of an increasing ensemble size. We also plot Ignorance/KL-divergence calculated using the system density (black line); this defines the maximum skill level. In the case of the KL-

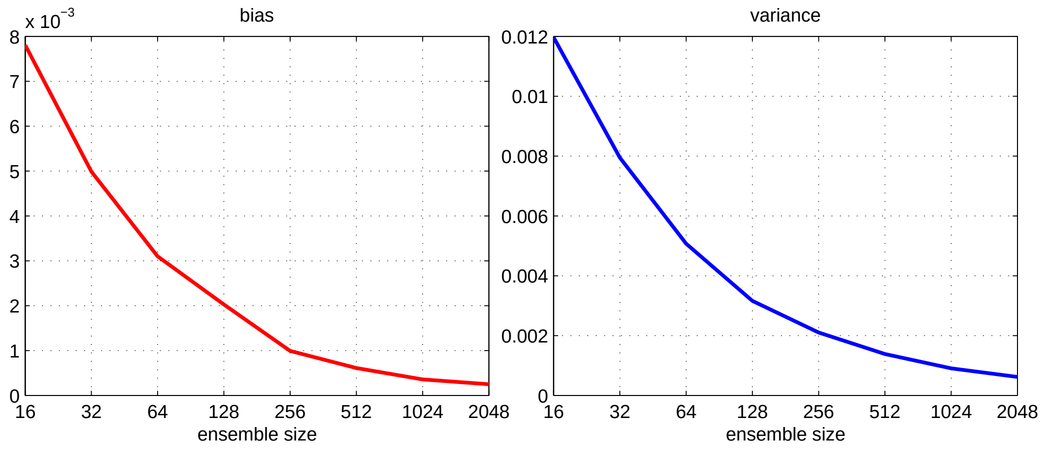

Figure 3.5: **Bias and variance in PMS:** Both bias (red) and variance (blue) quickly decay as the ensemble size (the horizontal axis) increases. Note that bias is measured as the expected value of the distance between the green and the black lines of Fig 3.4.

divergence, the maximum skill is reached at zero, since when the model distribution is equal to the system, the KL-divergence is zero. Values of both metrics decrease as the larger ensembles provide better forecasts. The median KL-divergence (red) is very close to zero when the ensemble size is 2,048, signaling that the (median) forecasting density is very close to the system. The (median) Ignorance (blue) also decreases, although at a slower pace. As the number of ensemble members increases, the minimum Ignorance of the model must become the same as that of the system. Under PMS (and in the limit) the Ignorance reduces to the entropy of the distribution (see Eq 3.9). This is why the Ignorance of the system (black horizontal line) is located at a value of 2.047 Bits, the entropy of $N(0, 1)$ distribution.

We have shown KD properties in simple settings using a Gaussian distribution and now shift our attention to a more complex setting; we use the logistic map under PMS/PE. The purpose of this example is to confirm and demonstrate that even for complex multimodal densities, we can observe the

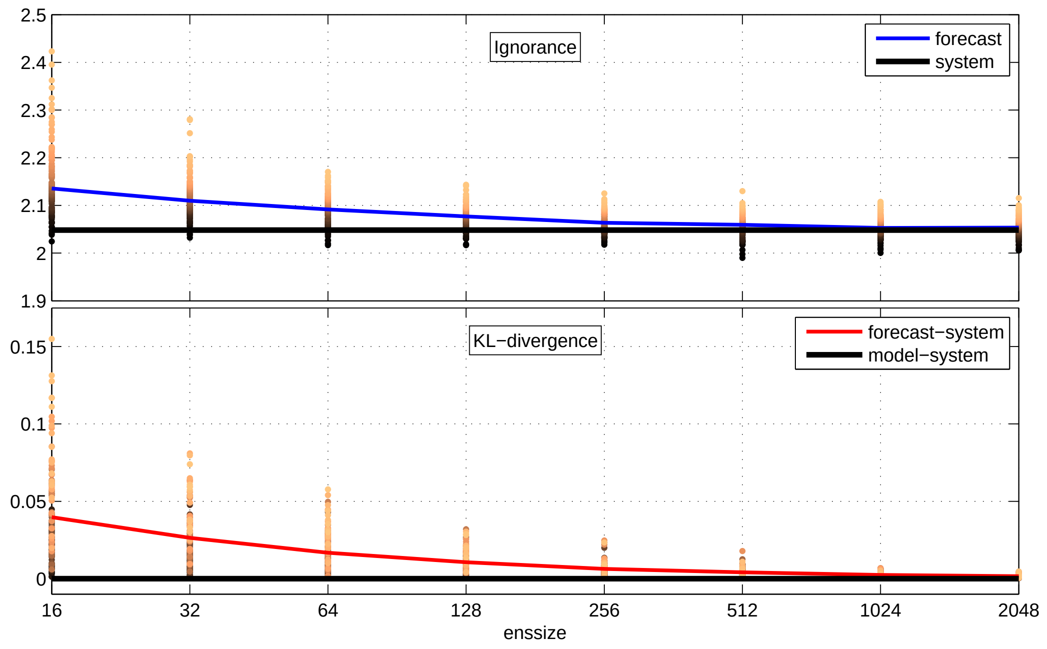

Figure 3.6: **Ignorance and KL distance in PMS:** Under PMS, increasing the size of an ensemble brings the forecasting density produced by kernel dressing closer to the system density. The 128 sample average of KL-divergence (red line) converges to zero, while the ignorance (blue line) approaches the ignorance obtained by using probabilities assigned by the system itself, i.e. the system defines a zero skill reference (black line).

convergence between the model and the system. Fig: 3.7 shows 128 forecasting densities for the same ensemble sizes as in Fig: 3.4. Even in this much more complex example, KD with Ignorance minimization leads to improved forecasts as the ensemble size increases. The Ignorance and KL-divergence plotted in Fig: 3.8 show substantial reduction in both metrics, with KL-divergence being reduced significantly but saturating at a non-zero level with an ensemble size of 2,048. The complexity/multimodality of the logistic map density is the most likely reason that the KL-divergence is prevented from coming closer to the zero line; the convergence is much slower when compared to the normal distribution example. This is because the abrupt changes in density (multiple modes) are more difficult to approximate than the smooth uni-modal density of $N(0,1)$. Much larger ensembles would be required to see the full convergence. With more ensemble members the multiple modes would be better covered, kernels would require smaller bandwidth and hence the description of the distribution would be more detailed.

The two examples presented here are strictly within the PMS/PE scenario. Outside PMS/PE there will be no convergence as the system distribution is completely different from the model distribution. It is only under the PM-S/PE where the properties of the KD can be studied and appreciated; outside the PMS/PE there is no target distribution, hence no absolute measure of 'goodness'. This is, of course, a consequence of the basic fact of forecasting: the system is not known and if it was, we would not need to forecast its future states, they would just be calculated.

3.2.2 Archive size and archive-ensemble size tradeoff

In terms of data, the forecast quality depends on two factors: the size of the forecast-verification archive and the size of the ensemble. Using the Gaussian PMS/PE scenario (Section 3.2.1), we have demonstrated that increasing an ensemble size has a positive impact on the forecast quality as measured by both the Ignorance and the KL-divergence. Intuitively, we might speculate that increasing the size of the archive would also lead to improved forecasts. So what size should an archive be to obtain robust forecasting performance?

Fig: 3.9 shows several experiments where the ensemble size is fixed at $M = 2,048$, while the size of an archive is doubled for each experiment. The Ignorance is then plotted as a function of the ensemble size and, as before, we generate 128 samples for each archive size. The median Ignorance value (green) shows that when $M = 2,048$, increasing the archive size only leads

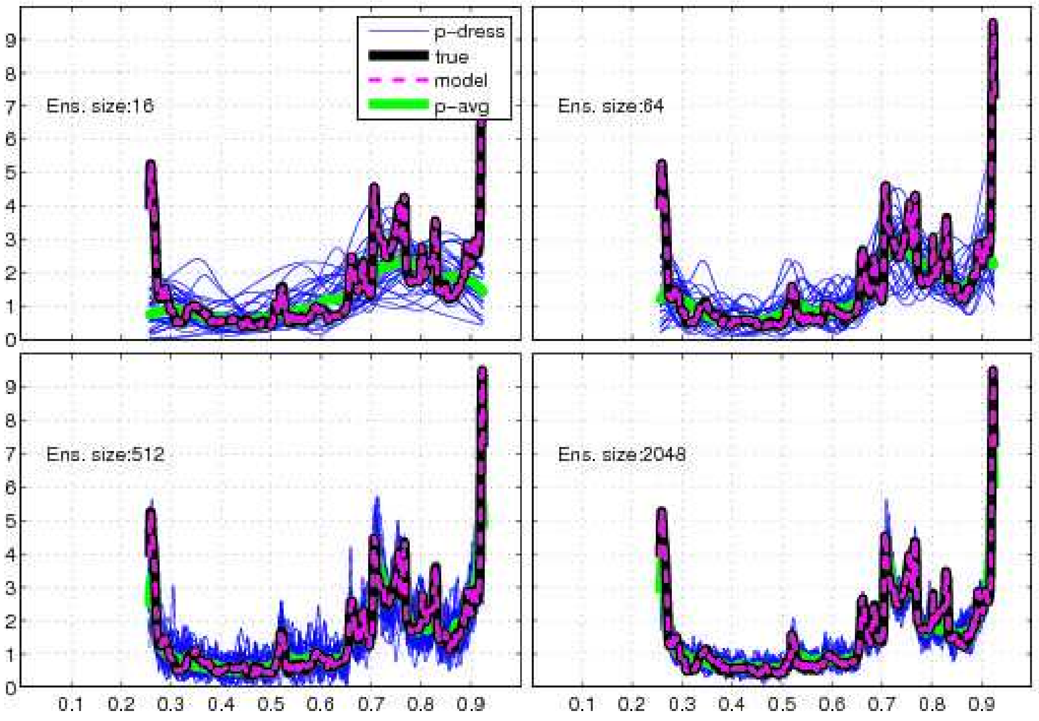

Figure 3.7: **Logistic map PMS:** The mean (green) of 128 estimates (blue) gradually converges to the underlying density of the Logistic map (black) as ensemble size is increased. Pseudo-likelihood type of kernel density estimation was used to construct the underlying density. The climatology and the forecasts were obtained by iterating the set of initial conditions using the Logistic map 1,000 times.

to a small reduction in the Ignorance. The Ignorance decreased by about 0.1 Bits, which accounts for about a 10% improvement in forecasting performance (a decrease of 1 Bit leads to doubling of a performance). The main observation is that the variance of the Ignorance sample is being reduced as we increase the archive size. This suggests that the size of the archive mainly affects the efficiency of the forecast.

In Fig: 3.10, we repeat the experiment for different ensemble sizes; Fig: 3.9 is plotted in the last panel. There are two pieces of information captured

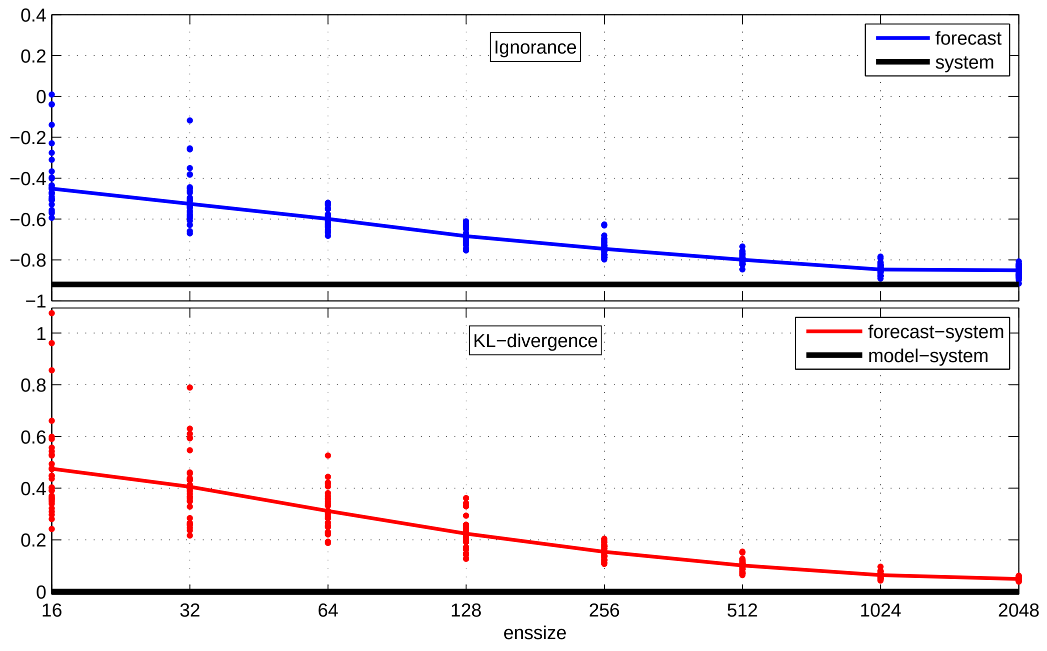

Figure 3.8: **Ignorance of Logistic map PMS:** Similarly to Gaussian PMS, both the Ignorance and the KL-divergence decrease significantly with the size of ensemble. Due to the large second derivative of the underlying density, the KL-divergence is rather high at small ensemble sizes when compared to the Gaussian case. The maximum skill level is defined by the system density, similarly to Fig 3.6. The maximum skill in terms of Ignorance is -0.8, and zero for the KL-divergence.

by the Figure. First, the suggestion that the archive size reduces variance significantly, while the Ignorance only marginally, holds for all ensemble sizes. Second, the notion that the ensemble size has a greater impact on the forecast quality than the size of the archive is also confirmed. To see this, we can compare the first and the last panels. The Ignorance is lower in the last panel with the larger archive size of 2,048. In fact, the Ignorance seems to be decreasing across all the panels; however, we have not statistically tested the significance of the decreases. As noted above, the decrease in Ignorance as a result of larger archive size is only small; statistical testing did not seem

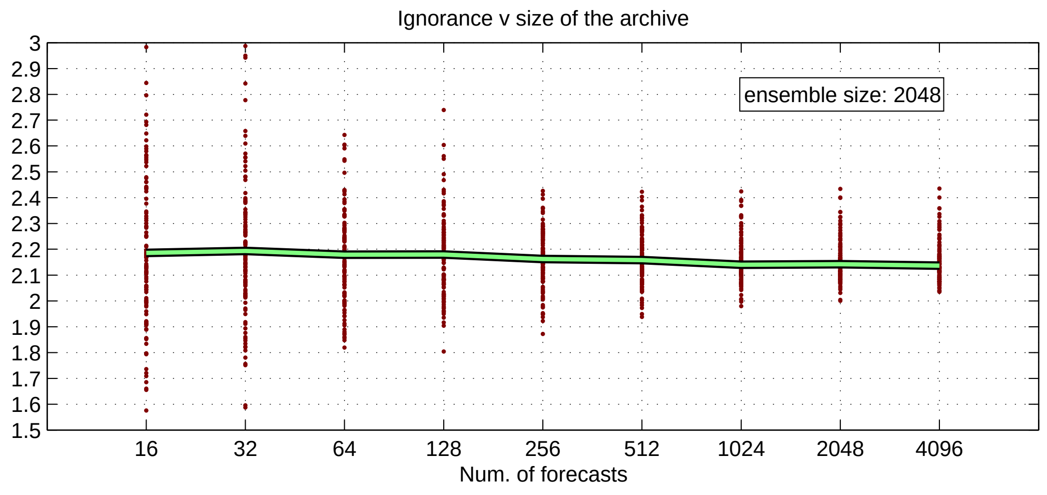

Figure 3.9: **Increasing the archive:** For the ensemble size of 16 members, the size of forecast-verification pairs is gradually increased (doubled at each step). For a given archive size Ignorance values are calculated using 128 samples. For small archive sizes, e.g. 16, the Ignorance values are widely dispersed. Increasing the archive size produces more stable estimates. The median value (green line) of the Ignorance also decrease, as the archive size increases.

necessary as it would at best confirm a significant but 'small' reduction. Given a constrained computational resource, it seems that an ensemble of 512 members and archive size of 1,024 forecast-verification pairs produce a good forecast in this particular case.

The conclusion suggested by the numerical results is that increasing an ensemble size is more beneficial in terms of the forecast quality than the size of the archive. That is not to say that size of the archive is unimportant since clearly a larger archive leads to a greater efficiency. In real-world applications, however, the archive size is limited, while the ensemble size may often be increased at a given computational cost.

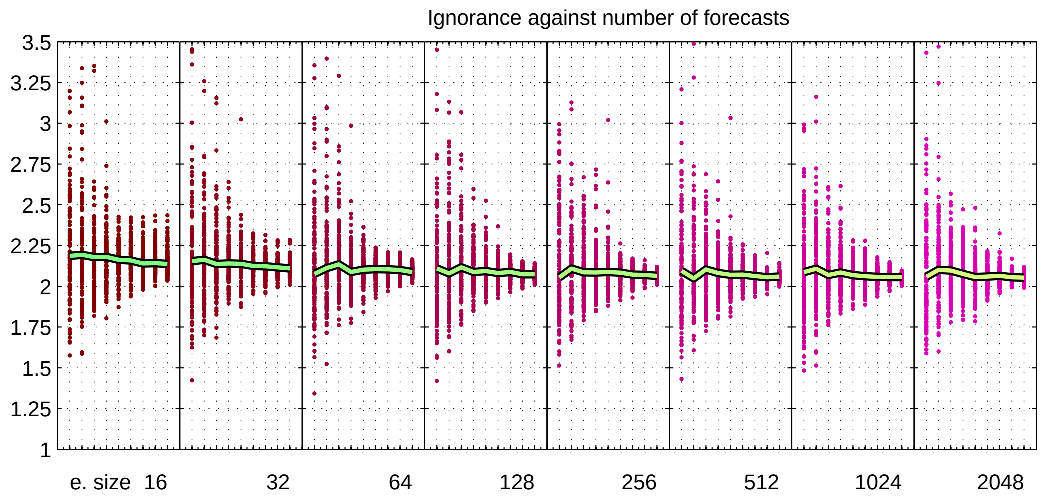

Figure 3.10: **Increasing the archive:** Each block shows same information as Fig: 3.9 but for a different ensemble size. A given block represents a given ensemble size stated on the x-axis. Within a block there are 9 different archive sizes and for each size 128 samples are generated and evaluated by Ignorance.

3.2.3 Kernel Dressing outside perfect model scenario

Section 3.2.1 provided some insight regarding the properties and behavior of KD under PMS/PE. To complete the picture, we briefly investigate its behavior outside PMS.

Considering the previous setting, a good starting point for IMS could be setting the model to be $N(0,1)$ while the system is $N(0,2)$. The idea is to study a situation in which the model is underdispersive and produces ensembles that are too tight. Fig: 3.11 shows that KD has no problem compensating for the larger variance in the system. KD simply increases the bandwidth σ. Even for small ensemble sizes, the forecasting densities coming out of KD are very close to the system and far from the model distribution. Also, the variability is much smaller when compared to the PMS/PE scenario in Fig: 3.4. When the ensemble size is 512 members, the forecasts are tightly

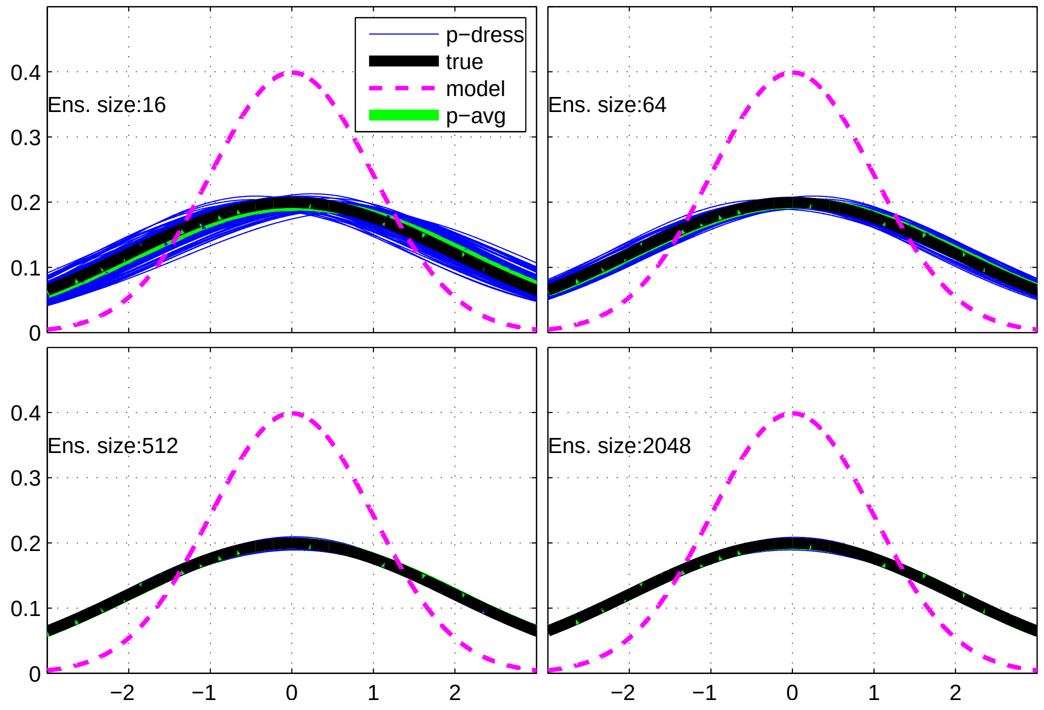

Figure 3.11: **IMS - System larger variance then model:** Similar to Fig: 3.4 but outside PMS. The model produces ensembles with lower variance that the system distribution. KD easily accommodates by increasing the bandwidth and the forecasting densities quickly uncover the system density.

located around the system distribution.

The Ignorance and KL-divergence profiles pictured in Fig: 3.12 show that there is very little additional improvement due to increased ensemble size. Beyond an ensemble size of 128 the Ignorance stops improving and the KL-divergence of the forecasting and the system distributions is almost 0. This result corresponds with the findings in Fig: 3.11. At this point, increasing the ensemble size merely adds value to the model PDF - detail that is inconsequential given the differences between the model PDF and the system PDF.

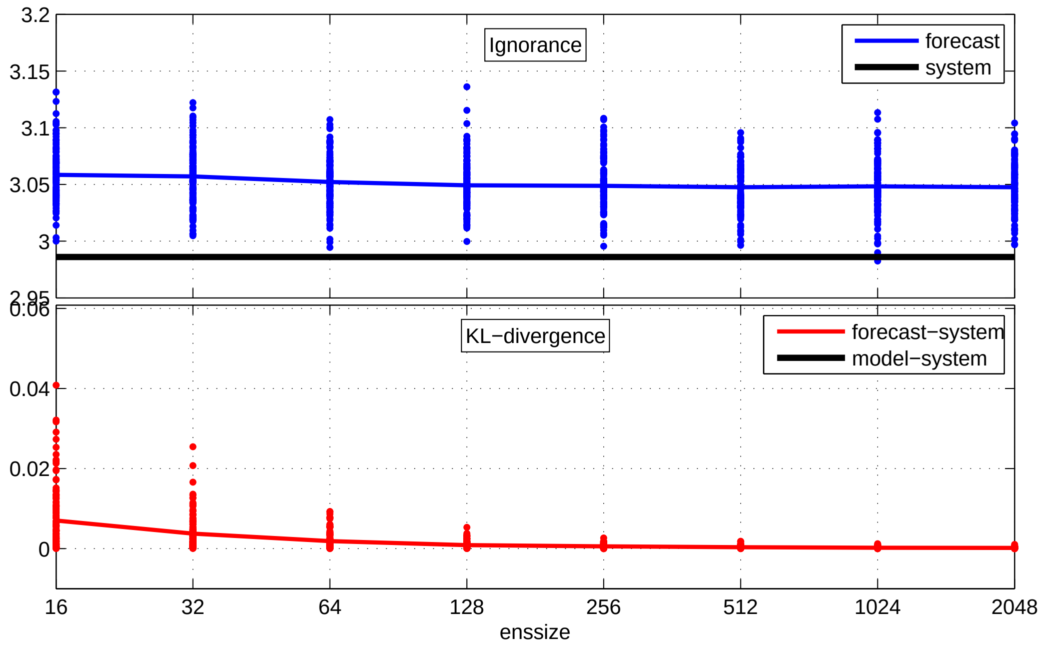

Figure 3.12: **IMS - System larger variance than model:** The Ignorance of the model (top panel) is rather flat but decreasing. Note the scale of the figure, the model Ignorance (blue) is very close to the system (black). The KL-divergence (bottom panel) decreases as KD utilizes more ensemble members. Also KL-divergence is very close to its potential for all ensemble sizes as KD very efficiently adjusts the bandwidth even for small ensemble sizes.

Next we look at an opposite example, i.e. the model is $\sim N(0, 2)$, having a larger variance then the system, $\sim N(0, 1)$. Fig: 3.13 shows the results. KD is not capable of shrinking the ensemble so it cannot recover the system distribution. The forecasting densities (blue) remain located around the model (green), far away from the system distribution (black). Increasing an ensemble size cannot help since the ensemble members are too dispersed. Setting the bandwidth to small values would only yield multimodal densities with a high a Ignorance level. This effect is somewhat visible in all 4 panels plotted as a number of densities are multimodal.

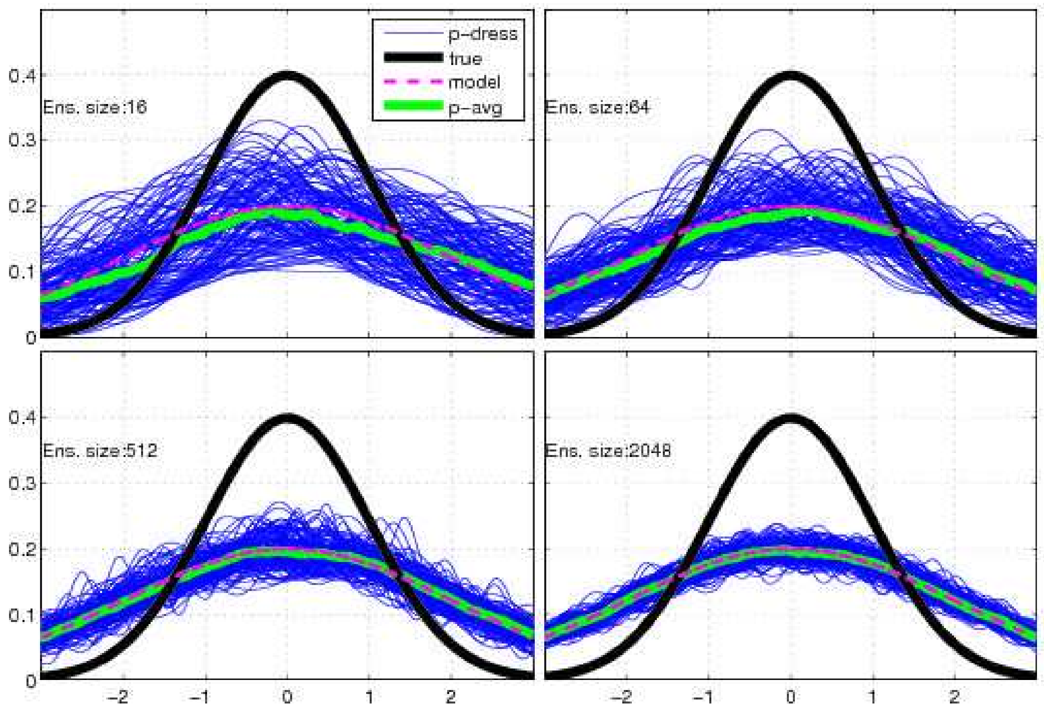

Figure 3.13: **IMS - System smaller variance then model:** Same as in Fig: 3.11 but this time the model produces ensembles with larger variance then the system distribution. KD fails to recover the system distribution as it cannot find bandwidth that would produce forecasting distributions close to the system.

3.2.4 Affine kernel dressing outside perfect model scenario

In Fig: 3.13 KD was not able to adjust the variance of the forecasting density. When a forecast has a larger variance than the system, KD can only recover climatology of the model. We note that the setting used in that example only optimized the offset o and bandwidth σ parameters.

In *affine kernel dressing* (AKD) [15], an extended version of KD, a scaling parameter addressing the problem was introduced. In AKD, the idea is to

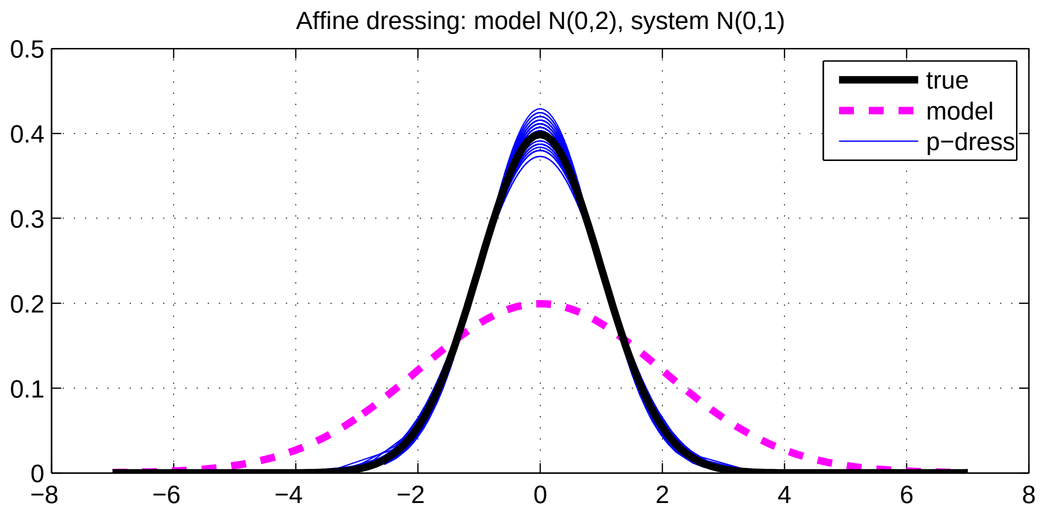

Figure 3.14: **AKD - System smaller variance then model:** The overdispersiveness of the model (magenta) seems to be corrected by the AKD. The forecasting densities (blue) closely surround the system density (black). The size of the forecast archive is 2,048, the ensemble size is 512.

work with transformed ensemble and bandwidth

$$z_i = ax_i + r_2\overline{\mathbf{x}} + r_1 \tag{3.12}$$

$$\sigma^2 = h^2_{\mathcal{S}}(s_1 + s_2 v(\mathbf{z})) \tag{3.13}$$

where $\overline{\mathbf{x}}$ is the ensemble mean, x_i is an ensemble member, a is the scaling parameter, $\{r_1, r_2, s_1, s_2\}$ are parameters determined during dressing and $v(\mathbf{z})$ is the ensemble variance. The $h^2_{\mathcal{S}}$ is the so-called Silverman's factor, which plays a role in selection of an optimal bandwidth (see [129]). The forecasting density is then obtained as

$$\hat{p}_\theta(y|x,\theta) = \frac{1}{m\sigma}\sum_{j=1}^{m} K\left(\frac{y - z_i}{\sigma}\right) \tag{3.14}$$

Intuitively, including the scaling parameter in the dressing procedure should help KD to correct the overdispersive ensemble by rescaling it. In Fig: 3.14 we show the densities produced by the AKD. In this case, the AKD successfully dealt with the problem; the forecasting densities (blue) are close to the system (black) and far from the model (magenta dashed).

However, scrutinizing the optimized parameters reveals that all the scaling parameters are set to zero, giving the trivial result

$$\begin{aligned} z_i &= 0 \times x_i + 0 \times \overline{\mathbf{x}} + 0 && (3.15)\\ z_i &= 0. && (3.16)\end{aligned}$$

Although unfavorable to AKD, this in fact a sensible result. Rescaling the ensemble to zero yields

$$\hat{p}_\theta(y|x,\theta) = \frac{1}{m\sigma}\sum_{j=1}^{m} K\left(\frac{y}{\sigma}\right) \tag{3.17}$$

which centers all the kernels at zero. In this particular case of system being $N(0,1)$, the 'average' distance of the verification from zero is one standard deviation, hence the optimum bandwidth is $\sigma = 1$ as correctly determined by the AKD. In other words, scaling the ensemble may lead to a trivial solution at which the ensemble has no contribution and the forecasting density is determined solely based on the verification. With AKD, the danger is that it may exclude any contribution of a forecasting model even if it indeed has a tangible forecasting skill.

3.3 Linking information measures

In this section, our aim is to describe the connections between Ignorance [49, 115] and other well established information measures, in particular the KL-divergence [74, 75] and Shannon's entropy [124–126]. We approach the problem using the framework of Kelly betting [72]. We exploit the connections between the different measures by showing that under different scenarios of a gambling game, the expected wealth of a player can be expressed in terms of Shannon's entropy, KL-divergence, or Relative Ignorance. By linking the Ignorance to the widely used information measures, we hope to provide additional information regarding its desirable properties.

3.3.1 Roulette, a punter, and the house

Consider a game of roulette where a punter repeatedly bets on K discrete outcomes $\{s_k\}_{k=1}^{K}$, over subsequent games (times) $t = 1, \ldots, T$. The occurrence of the outcomes is governed by a discrete PDF, $q_t = q_t(s)$. For simplicity, assume $q \equiv q_t$, i.e. the outcome distribution does not change over the distinct games.

The punter bets on each of K possible outcome, s_k, and always spreads all her wealth across the K outcomes. The amount wagered corresponds to $f_{t,k} \equiv f_t(s_k)$ where $f_t \equiv f_t(s)$ is her estimate of the 'true' q governing the outcomes. The punter's return (payoff) is given by the odds $o_{t,k}$ set by the house. The house sets the odds according to $o_{t,k} = \frac{1}{c_{t,k}}$, where $c_t \equiv c_t(s)$ is the house's estimate of q. The punter is motivated to maximize her wealth over the T games and we assume infinitely many games, $T \to \infty$.

In the following we show that, depending on the quality of her forecast f_t, the punter's expected profit/loss can be expressed in terms of the three information measures, Shannon's entropy [124–126], Kullback-Leibler distance [74, 75], and Ignorance [49, 115].

3.3.2 Forecasting the roulette outcome

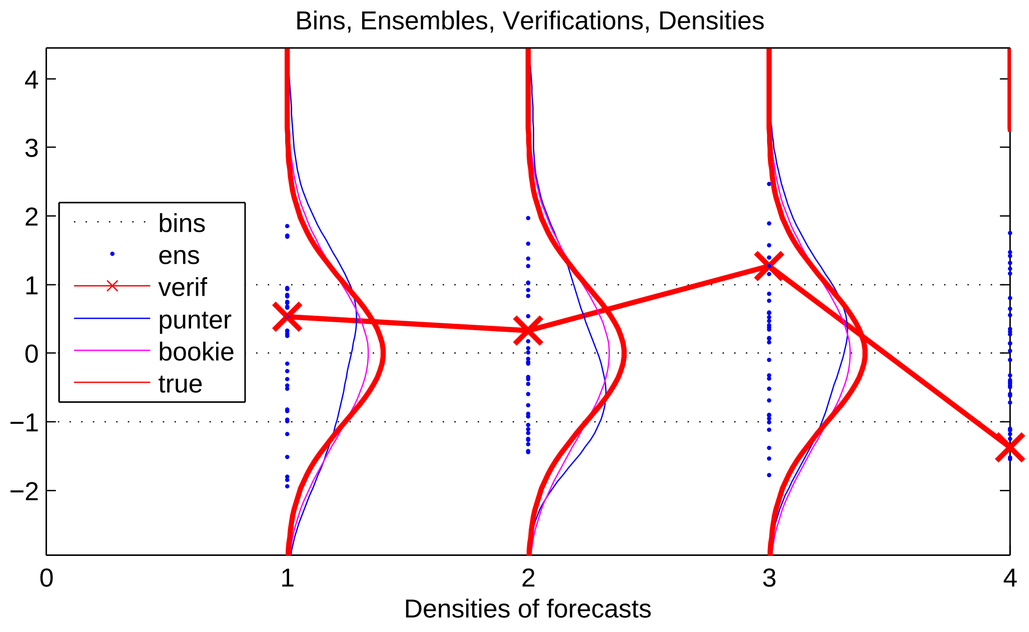

Figure 3.15: **Forecast distributions:** The punter uses the ensemble members (blue dots) to construct her estimate (blue curves) of the true density (red curves) governing the outcomes (red crosses). The house constructs its estimate (magenta curves).

We begin by providing a general framework of how the punter and the house use their respective models to forecast the roulette outcome [72]. Since we are concerned with probabilistic forecasting throughout this work, we apply it to the gambling framework that we use.

Fig: 3.15 depicts 3 subsequent games (or gaming times). The true density governs the outcomes (red crosses) of the roulette game. In this case, the outcomes are continuous but binned into 4 distinct bins separated by the dashed horizontal lines. The discrete probability of an outcome falling into a bin k is given as

$$\begin{aligned} q(s_k) &= P(s_{k,lower} \leq S \leq s_{k,upper}) \\ &= \int_{x_{k,lower}}^{x_{k,upper}} q(s)ds \end{aligned} \tag{3.18}$$

where $x_{k,upper}$ is the upper bound and $x_{k,lower}$ is the lower bound of the k-th interval. Both the punter and the house use their forecasting models to construct the forecasting densities for a given game. They may either construct the discrete densities directly, or discretize continuous densities as they do in this particular example. To produce her forecasting density, f (blue curve), the punter issues an ensemble forecast of N ensemble members (blue dots) and constructs the density (possibly using KD). The house follows the same steps when constructing it's forecasting density, c (magenta line).

The question is, who is expected to win over the repetitive games and how much does the winner obtain? Intuitively, we would expect the punter to win if her forecast f of the true q is 'better' than the house's forecast c. The ultimate answer therefore depends on a gambling scenario; in the following we distinguish 3 basic scenarios.

(1) Both the punter and the house know the true probabilities.

(2) Only the house knows the true probabilities.

(3) Neither the punter nor the house know the true probabilities.

We also note that an important element of a scenario is whether the punter (or the house) change the forecasting density as they proceed through the games. In scenarios (1) and (2), the forecasting densities do not change. In scenario (3), we allow the punter to produce a different forecasting density for each game (time). Note that in Fig: 3.15 the punter modifies her density between the games while the house uses an imperfect but constant forecasting density. The true density governing the outcomes does not change (as in the real game of roulette) but this assumption does not change the results in any way.

In Sections 3.3.4, 3.3.5, and 3.3.6, we explore the three scenarios, and show how under different scenarios different information measures arise. The measures then provide an insight into if, and at what rate, the punter's total wealth grows.

3.3.3 Growth of punter's wealth

Here we define the punters wealth, show how it grows in time, and show the expected wealth in the limit as number of games $T \to \infty$.

A good measure of the punter's success is the utility derived from her wealth at the end of the evening, i.e. after T games. In Utility theory [150], wealth is often quantified via a log of (per game) geometric average of the growth rate of wealth

$$G(T) = \frac{1}{T} \log \frac{V_T}{V_0} \tag{3.19}$$

where V_T is punter's wealth after the T-th game and V_0 is her initial wealth. We follow the well-known approach of Kelly [72] by assuming that the punter bets her total capital and places bets on all of the k outcomes. The punter's wealth evolves according to

$$\begin{aligned}
V_1 &= f(s_{1,k}) \times o(s_{1,k}) \times V_0 \\
V_2 &= f(s_{2,k}) \times o(s_{2,k}) \times V_1 \\
\vdots \\
V_t &= f(s_{2,k}) \times o(s_{2,k}) \times V_{t-1} \\
&\vdots \\
V_T &= f(s_{T,k}) \times o(s_{T,k}) \times V_{T-1} \\
&= \prod_{t=1}^{T} [f(s_{t,k}) \times o(s_{t,k})] \times V_0 \qquad (3.20)
\end{aligned}$$

where $s_{t,k}$ is k-th outcome of t-th game, f is the punter's forecasting density and $o_{t,k}$ are odds issued by the house on the k-th outcome of the t-th game. Assuming that both the punter and the house keep their estimates of the true density q constant over the games, we may write

$$V_T \equiv \prod_{k=1}^{M} [f(s_k) \times o(s_k)]^{W_k} \times V_0 \qquad (3.21)$$

where W_k is a number of occurrences of an outcome s_k over the T games. Note, that the product over t in Eq. 3.20 has been replaced by a product over k in Eq: 3.21. This is because neither the punter's nor the house's forecast change over the T distinct games and so for distinct games the same amount $f(s_k)$ is wagered and the same odds o_k are issued for an outcome s_k.

The expected value of the utility growth $E[G(T)]$ is then given as

$$\lim_{T\to\infty} G(T) = \sum_k \frac{W_k}{T}\log(f(s_k)\times o(s_k))$$
$$= \sum_k q(s_k)\log\left(f(s_k)\times o(s_k)\right) \quad (3.22)$$

Since the occurrence of s_k is governed by the true probabilities q, the frequency $\frac{W_k}{T}$ converges to $q(s_k)$ as $T\to\infty$.

3.3.4 Scenario: Punter, the house both know true probabilities

Suppose the house knows the true probabilities, $q(s_k)$, and assigns odds according to $o(s_k) = \frac{1}{q(s_k)}$. Suppose that the punter also knows $q(s_k)$ and places bets on the occurrence of a k-th outcome s_k according to $q(s_k)\times V_{t-1}$. Substituting q for f in Eq. (3.22) yields expected log-utility growth

$$\lim_{T\to\infty} G(T) = \sum_k \left(q(s_k)\log q(s_k) - q(s_k)\log q(s_k)\right) \quad (3.23)$$
$$= \sum_k q(s_k)\log q(s_k) - \sum_k q(s_k)\log q(s_k) \quad (3.24)$$
$$= H(s) - H(s) \quad (3.25)$$
$$= 0 \quad (3.26)$$

where $H(s)$ is Shannon's entropy, i.e. both the punter and the house reach the expected value of the information contained in q. The punter's bets thus cancel out with the odds, she cannot win or lose. We note that the emergence of Shannon's entropy is due to the assumption that both the punter and the house know the true probabilities of the roulette outcomes.

3.3.5 Scenario: Only the house knows the true probabilities

Now suppose that the punter does not know the true probability and instead estimates $q(s)$ by $f(s)$. Furthermore, the punter does not change her forecast over the T games. Consequently, she bets $f(s_k) \times V_{t-1}$ fraction of her wealth on the kth outcome of the tth game. The house's strategy is the same as in the previous case. It knows q and sets odds to $o_k = \frac{1}{q(s_k)}$. Using Eq. 3.21, the punter's wealth after T games is

$$V_T = \prod_{k}^{M} [f(s_k) \times o(s_k)]^{W_k} \times V_0 \tag{3.27}$$

and the expected growth of utility is

$$\begin{aligned} \lim_{T\to\infty} G(T) &= \sum_k (q(s_k) \log f(s_k) - q(s_k) \log q(s_k)) && (3.28) \\ &= \sum_k q(s_k) \log\left(\frac{f(s_k)}{q(s_k)}\right) && (3.29) \end{aligned}$$

where the term $q(s_k) \log f(s_k)$ is the *cross entropy*, and the term $-q(s_k) \log q(s_k)$ is the entropy and Eq. (3.29) is the Kullback-Leibler divergence. To understand the meaning of Eq 3.29 in this setting, we can use Gibbs' inequality [22, 86], a statement about the entropy of discrete distributions, which states

$$-\sum_k q(s_k) \log f(s_k) \geq -\sum_k q(s_k) \log q(s_k) \tag{3.30}$$

where equality only holds when $f(s_k) = q(s_k)$. So if $f(s_k) \neq q(s_k)$, the sum in Eq. (3.28) is negative and the gambler will be ruined. The loss rate depends on the distance between $f(s)$ and $q(s)$, the cross entropy term of the Kullback-Leibler divergence.

In this scenario, the house has a clear advantage. It knows the exact probabilities of the outcome, while the punter is forced to construct an imperfect estimate of the true probabilities. The best but least likely scenario for the punter is that by chance she happens to construct a perfect estimate of the true density and draws with the house. With an imperfect model she loses all her wealth as $T \to \infty$. The emergence of KL-divergence in this scenario is due to the assumption that unlike the house, the punter does not know the true probabilities.

3.3.6 Scenario: None knows the true probabilities

Suppose neither the house nor the punter knows the true q and so they both estimate. Further suppose that their estimates of q change over the T games and are given as c_t and f_t respectively for the game t. In this scenario the product over t in Eq: 3.20 cannot be exchanged for the product over k, as the amounts wagered and the odds differ from one forecast to another. The punter's wealth after T games is

$$V_T = \prod_{t=1}^{T} (f_t(s_{k,t}) \times o_{t,k}) \times V_0$$

and the rate of growth is

$$\lim_{T\to\infty} G(T) = \frac{1}{T}\sum_{t=1}^{T}(\log f_t(s_{k,t}) - \log c_t(s_{k,t})) \tag{3.31}$$

The limiting frequency of bets/odds does not converge to q, and so the terms in Eq. (3.31) cannot simplify to either Shannon's entropy or Kullback-Leibler divergence. Note, however, that the two terms in Eq. (3.31) are respective Ignorances of the punter and the house (see Eq. 2.16). The punter would maximize her wealth by setting $f_t(s) = q(s)$, but she does not know q, and so her win/ruin status depends on whether her Ignorance is smaller or larger relative to that of the house, i.e.

$$\text{RIGN} = -\sum_{t=1}^{T}\log f_t(s) + \sum_{t=1}^{T}\log c_t(s) \tag{3.32}$$

If the punter's Ignorance is lower, then relative ignorance RIGN < 0 and the punter wins. The rate of profit/loss depends on the magnitude of RIGN. In this scenario, Ignorance emerges due to the assumption that both the house and the punter estimate the true probabilities. This shows the importance of Ignorance as a measure of forecasting performance and its close links to KL-divergence and Shannon's entropy.

3.4 Conclusions

We have studied the properties of Kernel dressing, a framework for turning ensemble forecasts into forecasting densities. Since Kernel dressing (KD) is often confused with Kernel density estimation (KDE), we have discussed the important distinctions between the two approaches. While some of the differences have been previously described in the literature, we add several new items to the list. The new additions include the fact that while KDE aims to recover true underlying density, the concept of true underlying density is absent in KD. Consequently, the objective functions in both methods are substantially different. Also, while the calculation of the minimization criterion in KDE utilizes more than one data point, the KD minimization criterion, Ignorance, is calculated at a single data point - verification. Yet another distinction is that unlike in KDE, where a single sample from a single true density is assumed, in KD we deal with a number of samples from independent distributions. All the distinctions listed here (and above) expose the nature of KD and show that despite using kernels, KD is a truly unique approach to density construction.

Assuming a perfect model (PMS) and a perfect ensemble (PE), we have shown that minimizing Ignorance within KD implicitly leads to minimization of KL-divergence. We have stressed that this is only valid under the PMS/PE assumption, and the deployment of Ignorance and KD. In other words, it is the combination of KD and Ignorance that leads to implicit minimization of KL-divergence. The implicit minimization of KL-divergence is an important property of KD as it establishes the asymptotic unbiasedness under PMS/PE assumption. To our knowledge, the unbiasedness of KD has neither been formally established nor informally discussed.

Building on the unbiasedness, we have numerically studied the behavior of KD under both the Perfect and Imperfect model scenarios (IMS). Given that KD is unbiased under PMS, it is expected to recover the true underlying density. Using a simple case of a standard normal distribution we have demonstrated that this is indeed the case. In addition, we used the numerical experiments to gain some intuition about convergence properties, and have shown how bias and variance depend on ensemble size (Fig. 3.5). In particular, we have shown that for the simple case of normal distribution the density is well recovered at ensemble size of 2,048 members (Fig. 3.4).

To see how KD performs in a more complex setting we repeated the numerical experiments for Logistic map, a system with complicated density (high multimodality). The density was well recovered with ensemble size of 2,048, which is in line with our results from the normal distribution exercise. The convergence in the Logistic map was slower, as can be seen by comparing Fig. 3.6 and Fig. 3.8. The slower convergence is expected due to the multimodality. It is more challenging to describe a function with abrupt changes than one with low variation.

To analyze KD behavior under the Imperfect model scenario, we have constructed two experiments. In both experiments both the model and the system take the form of a normal distribution with zero mean. However, in the first experiment, the model variance is smaller than the variance of the system. In the second, the model variance is larger than the variance of the system. We found that in the first experiment, KD successfully recovers the system density (Fig. 3.11), while in the second experiment KD fails (Fig. 3.13). The rationale is as follows: When the ensemble is under-dispersive (first experiment), KD simply inflates the kernels to cover observation outside the ensemble range, i.e. wider kernels compensate for under-dispersion. When ensemble is over-dispersive, shrinking the kernels does not help; smaller bandwidth only leads to multimodality. Shrinking kernel width therefore does not compensate for over-dispersion of the ensemble and KD is bound to fail.

To overcome the problem of over-dispersive ensembles, we have studied Affine KD, an alternative version of KD designed to address the problem of over-dispersion. Affine KD utilizes an additional offset parameter, which allows it to re-position ensemble members and thus change the ensemble dispersion.

Using the formulation of Eq. 3.12, we have found, that in some cases, affine KD leads to a trivial solution and rescales all members of the forecasting ensemble to zero. Although this eventually leads to recovery of the system density, the contribution of a potentially skillful model is discarded.

Despite its desirable properties, Ignorance is not a frequently used measure in forecast evaluation. We attempt to further demonstrate its usefulness by exposing links to important Information theoretical measures. We have used the framework of Kelly betting to show that depending on some simple assumptions, the Kelly betting framework yields three measures, Shanon's entropy, KL-divergence, and Ignorance, as optimal criterion of wealth distribution. To show this, we have assumed three scenarios; First, both the punter and the house know the true probabilities of a betting outcome. Second, only house knows the true probabilities. Third, neither the house nor the punter know the true probabilities. We found that the wealth maximization of Eq. 3.21 yields Shannon's entropy under the first scenario, KL-divergence under the second and Ignorance under the third. Since all scenarios are based on the same framework and differ only in a simple assumption we argue that there are strong links between the three measures. By exposing these links we have demonstrated that Ignorance is optimal under Kelly betting and hence is a suitable measure for evaluation of probabilistic forecasts.

Chapter 4

Dynamic climatology and its benchmarking utility

In weather forecasting, a forecasting model is often evaluated relative to a *zero skill* reference (benchmark) [157]. The (un)conditional climatologies described in Sections 2.2.3 are commonly used definitions of zero skill. As reported in [53, 68, 89] climatological reference may lead to over-reporting of a forecasting skill of the evaluated model. We propose to define zero skill via more robust yet simple models, simple statistical models [56] being a convenient choice.

In this chapter, we construct the Dynamic Climatology (DC), an easy-to-use statistical model designed to provide a zero skill reference to a forecasting model. We require the DC to:

a) outperform (or perform comparably to) climatology,

b) be easily deployable and moderate in terms of computational cost,

c) be capable of producing 'new' values, i.e. values not seen within a training dataset,

d) to partially compensate for the degradation of the forecasting performance at long leadtimes, which naturally arises in simulations due to accumulation of a forecasting error.

The particular version of DC deployed here is based on a *nearest neighbor* (NN) search [9, 24, 34], but we stress the the idea of using a simple statistical model as a more robust reference is general. Although our version of DC may be further improved, we do find the DC properties satisfactory enough to render DC a useful benchmark. We seek to continually improve our DC models in our future work.

In our analyses, we deploy DC to forecast sea surface temperature (SST) over the Nino3.4 and the MDR regions (see Section 2.5.2) and use the forecasts to benchmark the state-of-the-art seasonal-to-annual global coupled climate models of the ENSEMBLES project [57]. The seasonal forecasts of the ENSEMBLES models were evaluated in [152], who find significant improvement of the ENSEMBLES models over the previous generation of models (DEMETER). The evaluation metrics used in [152] included RMSE and the Brier Skill Score and the performance was contrasted, namely via a multi-model ensemble (MM). Our evaluation approach differs in that we use Kernel dressing (Chapter 3) to turn the ENSEMBLES forecasts into forecasting distributions and evaluate the performance using Ignorance.

Our main goal is to demonstrate the ability of Dynamic climatology to benchmark performance and assess value of a given forecasting model. While our benchmarking model can be applied in a variety of fields, we choose the application of seasonal forecasting due to its great importance to many industries. As described in Section 2.5, seasonal forecasting is concerned with weather forecasts at time scales of several months. A good seasonal forecast may therefore be very useful in forming long-term decisions in fields such as agriculture, health, transport, insurance, economics, and many others.

The most important tool of seasonal forecasting is the use of coupled ocean-atmosphere global circulation models. The idea behind coupling is very powerful, and is based on the fact that ocean processes propagate at a much slower pace than atmospheric processes, and have a large and persistent impact on global weather. Due to the coupling of ocean and atmospheric processes, coupled models enable forecasters to predict global weather conditions over long horizons. The state-of-the-art ENSEMBLES models therefore represent an interesting object for a benchmarking exercise.

The best known example of the ocean processes important for seasonal forecasting is El Niño/La Niña, a quasi-periodic pattern of warming/cooling of the surface (sea surface temperatures) of the tropical eastern Pacific Ocean. This global scale event has been shown to greatly influence weather across the globe on the time scale of months (e.g. [146, 147]). In a similar fashion, the sea surface temperatures in the Main Development Region (see Section 2.5.2) have a large impact on the formation of hurricanes in the Atlantic basin. Forecasting the number of hurricanes formed in the Atlantic is crucial for energy industries along the southeastern US coast, which often suffer heavy losses during the hurricane season. Due to the impact of sea surface temperatures, the eastern Pacific as well as the Atlantic basin, are regions of particular interest. They will be the focus of our benchmarking exercise.

In this exercise, we show that the forecasting skill of DC is comparable to, and in some cases higher than, the skill of the ENSEMBLES models. Given the attention that has been recently payed to multi-model ensembles [30, 73, 104, 151, 152] we also compare DC with a MM constructed by combining equally weighted individual ensembles of the four ENSEMBLES models into a grand-ensemble. We show that the MM does not significantly outperform either the individual models or the DC. We will argue that this result is due to our evaluation approach, Kernel dressing, which implicitly improves/debiases the individual models, so that they are directly comparable with the MM. We also briefly study the return to skill [4] that the ENSEMBLES models exhibit at an annual forecast in the eastern Pacific (see Section 2.5.3). We investigate whether the return to skill is due to improvement of the model performance at the long leadtime or simply due to the variations in climatology [53]. To see this, we separately evaluate an unconditional climatology and investigate how difficult it is to forecast different time periods (in relative terms).

This chapter is structured as follows. In Section 4.1 we provide a detailed description of the DC model and illustrate the basic properties of DC by forecasting both a simple and a noisy sine curve in Section 4.2.1. We then follow this by demonstrating the DC properties in a more complex setting, using a simple chaotic system of the damped forced pendulum, in Section 4.2.2. The benchmarking of the ENSEMBLES models with the DC is performed in Section 4.3.

In this chapter, the following are new contributions:

- The DC is demonstrated to be a useful benchmarking model.
- Forecasting performance of ENSEMBLES models is evaluated using a KD framework.
- The DC is shown to perform comparably with ,and at points to outperform, the ENSEMBLES models.
- The difficulty of forecasting different months in the Nino34 area is investigated.

4.1 Dynamic climatology

We are interested in developing a simple-to-use forecasting model that would provide a robust benchmark to alternative models. We introduce the Dynamic Climatology, a simple statistical model that uses analogs [148, 149] of a current state to generate ensemble forecasts [78, 98]. This section provides a technical description of the DC model.

4.1.1 Analogs, and how to find them

The DC is based on the *method of analogs* [82, 131, 148], which produces forecasts by locating situations observed in the past that are analogous to a current situation. This approach requires a definition of a quantitative measure of similarity [9, 34], a particular form of which we discuss below.

Analogous situations, *analogs*, are utilized to produce an ensemble forecast. There are two important stages in the DC. The first stage is concerned with the definition of a current situation, while the second stage focuses on how to construct ensembles.

The presented version of DC locates analogs using a k-nearest neighbor (KNN) algorithm [24]. The KNN classifies past system states $x_{t<0}$ based on their distance from the current state $x_{t=0}$, which we call a *query point*. The distance of system states from a query point is measured by some metric (specified below), within a d-dimensional space called a *feature* space.

The definition of the feature space plays a crucial role in producing good analogs. In our version of DC, a 'situation' is defined in terms of a *feature vector* containing not only the current but also several preceding states, so that at a given time t a feature vector can be written as $\mathbf{x}_t = \{x_t, x_{t-1}, \ldots, x_{t-d}\}$, where $\{x_{t-j}\}_{j=0}^{d-1}$ are d most recently observed states of a system. We note that constructing a feature vector in this manner resembles delay embedding [138]. However, in our case we are not required to know the embedding dimension.

While the usual choice of the distance metric in KNN is the Euclidean distance, we use a correlation type of measure. Our aim is to capture the dynamics of the time series; correlation-like measures are convenient as they relate the dynamical behavior of two variables. To measure the distances in the DC we use an angle between two nonzero vectors as given by the dot product

$$\theta = \arccos\left(\frac{\mathbf{x_t} \cdot \mathbf{x_{t-\tau}}}{\|\mathbf{x_t}\|\|\mathbf{x_{t-\tau}}\|}\right) \tag{4.1}$$

where $\mathbf{x_t}$ captures situation at time t and $\mathbf{x}_{t-\tau}$ is a candidate for an analog, i.e. the situation observed at an earlier time $t-\tau$.

The reason why we work with sequences of d subsequent states of a system is easier to understand in the following example. Consider a time series of monthly mean temperatures measured at London Heathrow between January 1960 and May 2011. The task is to use data up to May 2011 and forecast the temperature in June 2011. Now further consider using only the single measurement of May 2011 and Euclidean distance as a KNN metric. If September 1999 is colder than usual (or May 2011 warmer), it may well be that the temperature measured in September 1999 is the closest to May 2011. Despite being the closest analog, is September really useful in predicting June? Most likely not. Ideally an analog of May 2011 would be determined as another May (say May 1960); intuitively May should be more useful then September in predicting June. So using only a single observation (and the Euclidean distance) may not be an ideal way of searching for an analog.

We may improve our chances of finding a suitable analog by forming a feature vector using a sequence of, say $d = 3$, past observations,

$$\mathbf{x}_{\text{MAY11}} = \{x_{\text{MAY11}}, x_{\text{APR11}}, x_{\text{MAR11}}\} \tag{4.2}$$

It is unlikely that a sequence formed for September 1999 would be similar, but we expect that sequences constructed for the May months will. Working with sequences thus may prevent selection of a false analog due to chance.

Another problem may arise due to the Euclidean distance when used as a KNN metric. Consider two sequences, say

$$\begin{aligned} \mathbf{x}_{\text{MAY09}} &= \{x_{\text{MAY09}}, x_{\text{APR09}}, x_{\text{MAR09}}\} \\ \mathbf{x}_{\text{NOV99}} &= \{x_{\text{NOV99}}, x_{\text{OCT99}}, x_{\text{SEP99}}\} \end{aligned}$$

both being candidates for an analog of the May 2011 sequence (Eq: 4.2). Further assume that temperatures measured for all three sequences are:

$$\begin{aligned} \mathbf{x}_{\text{MAY11}} &= \{11, 10, 9\} \\ \mathbf{x}_{\text{MAY09}} &= \{9, 8, 7\} \\ \mathbf{x}_{\text{NOV99}} &= \{9, 10, 11\} \end{aligned}$$

Using the Euclidean distance we get

$$\begin{aligned} D_E(\mathbf{x}_{\text{MAY11}}, \mathbf{x}_{\text{MAY09}}) &= 3.4641 \\ D_E(\mathbf{x}_{\text{MAY11}}, \mathbf{x}_{\text{NOV99}}) &= 2.8284 \end{aligned}$$

where D_E is the Euclidean distance. Based on the Euclidean distance, KNN would select November 1999 as an analog of May 2011, not a very intuitive choice when one is interested in forecasting June temperature. Although the temperatures of the May 2009 sequence are more distant, they do have the correct direction, i.e. they are increasing (unlike the November sequence). The correlation coefficient between the two May sequences is 1, while correlation between May and November sequences is -1. Deploying a correlation-like measure in this case yields the intuitive choice by selecting May 2009 as the analog. Although this example is rather artificial, we hope it is sufficient to illustrate that correlation-like measures are better suited to capture the system dynamics then the Euclidean distance. We stress that DC is mainly concerned with dynamical properties of the time series, hence the metric in Eq: 4.1 is better suited for our purposes then Euclidean distance.

4.1.2 Producing a single ensemble member

The DC forecasts are constructed by utilizing first differences of analogs and their *images*, image being defined as a state immediately following an analog. To produce a single point forecast, the DC proceeds in the following stages

(1) The feature vector for a query point x_t is defined as

$$\mathbf{x}_t = \{x_{t-1}, \ldots, x_{t-d+1}\} \tag{4.3}$$

(2) The KNN locates the nearest analogous feature vectors,

$$\mathbf{x}^a_{t-\tau} = \{x_{t-\tau}, \ldots, x_{t-\tau-d+1}\} \tag{4.4}$$

from the archive.

(3) The leading element of this analogous feature vector is an analog, $x^a_{t-\tau}$. The leading element is simply the most recent state within the feature vector.

(4) The image of the analog is determined as $x^i_{t-\tau+1}$.

(5) Dropping the subscripts for simplicity, the first difference of the image and an analog, $\Delta x^a = x^i - x^a$, is calculated.

(6) The first difference Δx^a is added to the current state x_t, yielding a leadtime 1 point forecast of $\hat{x}_{t+1} = x_t + \Delta x^a$.

This procedure is generalized to K ensemble members in Section 4.1.4

To provide more insight into the process, consider the above example of Heathrow monthly temperatures. With May 2011 being the query point we are interested in forecasting the mean temperature of June 2011. Assume, that the KNN algorithm correctly determines the feature vector of May of 1968, $\mathbf{x}_{\mathrm{MAY68}} = \{x_{\mathrm{MAY68}}, x_{\mathrm{APR68}}, x_{\mathrm{MAR68}}\}$, as closest to the feature vector of the query point, $\mathbf{x}_{\mathrm{MAY11}} = \{x_{\mathrm{MAY11}}, x_{\mathrm{APR11}}, x_{\mathrm{MAR11}}\}$. The DC then selects May 1968 as an analog, x^a_{MAY68}, and June 1968 as its image, x^i_{JUN68}. The difference $\Delta x^a = x^i_{JUN68} - x^a_{MAY68}$ is then added to the query point giving the June 2011 forecast as $\hat{x}_{JUN11} = x_{MAY11} + \Delta x^a$.

What is the rationale behind using the first differences? Why not directly use the images as a forecast, i.e. $\hat{x}_{t+1} = x^i$? If we were to use the images directly, we would implicitly restrict space of forecast outcomes. DC would yield only values that have been observed in the past. By using the first differences, DC is not restricted to the space defined by the training set. It is capable of generating new previously unobserved values; it can venture outside the training set.

4.1.3 Growing ensembles

DC can be set to produce ensembles of size M. This is done by letting DC use $K > 1$ analogs. Instead of selecting only the closest feature vector, the KNN locates K closest feature vectors yielding K analogs $\{x^{a,k}\}_{k=1}^{K}$ and K images $\{x^{i,k}\}_{k=1}^{K}$. The first differences $\Delta x^{a,k}$ are then added to the current state x_t to produce an ensemble of $M = K$ ensemble members $\hat{\mathbf{x}}_{t+1} = \{\hat{x}_{t+1}^{1}, \ldots, \hat{x}_{t+1}^{M}\}$ for leadtime 1.

Consider a leadtime 2 forecast and assume we set $K = 2$. For leadtime 1, we have an ensemble consisting of $M = 2$ ensemble members x_{t+1}^{1} and x_{t+1}^{2}. To produce a single leadtime 2 forecast, we need to construct a query point based on the leadtime 1 ensemble. But which of the 2 members of the leadtime 1 forecast should we use? The answer is both. The DC iterates through all the leadtime 1 ensemble members and generates K of 1-step ahead forecasts for each. Thus the leadtime 2 ensemble consists of $M = 4$ members, 2 based on x_{t+1}^{1} and another 2 based on x_{t+1}^{2}. Thus setting $K > 1$ not only leads to an ensemble, but it also leads to ensembles that grow in size with each leadtime. From now on, we denote the ensemble size at a given leadtime as M_l.

Is there any benefit from having ensembles growing with leadtime? To show the benefits, we first note that to forecast short leadtimes is usually less difficult than forecasting long leadtimes. It is the long leadtimes where we need plenty of forecasting power.

The first benefit is the following. A growing ensemble may improve performance at the long leadtimes since at every leadtime it increases the probability of hitting the target by deploying more ensemble members. The second benefit is that at short leadtimes, where even a small ensemble can work well, we do not waste CPU time by producing large ensembles when they are not needed. Instead, we get through the short leadtimes quickly and efficiently and save the computational resources for longer leadtimes where, hopefully, the increasing ensemble size will improve forecasting performance.

4.1.4 Pruning the ensembles

We must be cautious when working with the growing ensembles. Setting $K = 2$ and letting the ensemble size grow freely would mean that forecasting 20 leadtimes ahead would yield an ensemble of $M = 2^{20} = 1048576$ ensemble members. The unlimited growth of ensemble size quickly proves computationally prohibitive even for simple models. To avoid the problem, the DC uses a pruning mechanism. Three pruning examples are given here, each of which is applied when an ensemble size exceeds a threshold, $M^{\max}$.

The default pruning method randomly samples the $M_l > M^{max}$ ensemble members to bring the size of the ensemble down to M^{max}. Another method sorts analogs according to their closeness and keeps only the M^{max} closest ones. The last method selects every n-th member of the sorted ensemble producing a uniform sample based on analog closeness.

The presented version of DC has also built in the optionality of setting how many analogs should be used at the first leadtime and how the ensemble size should grow beyond leadtime 1. For example, for the leadtime 1 forecast we may require an ensemble of 16 members. Setting $K_0 = 16$ produces 16 analogs and hence 16 point forecasts at leadtime 1. Had we set $K = K_0$ for the leadtime 2, the DC would identify 16 analogs of each of the 16 leadtime 1 ensemble members, i.e. the ensemble size would be $M = 16 \times 16 = 256$. For leadtime 3, we obtain $M = 256 \times 16 = 4,096$, for leadtime 4 we get $M = 65,104$ etc. and the problem quickly grows intractable. To prevent the intractability, DC resets the number of analogs beyond leadtime 1. Thus DC produces a reasonably sized ensemble at leadtime 1 while keeping the growth of the subsequent ensembles in check. For example, setting $K_0 = 16$ and $K = 2$ gives the size of the leadtime 1 ensemble $M = 16$, size of leadtime 2 ensemble $M = 32$, leadtime 4 $M = 64$, etc.

4.1.5 The DC parameters, further comments

The preceding sections have introduced a number of properties of the DC model, all of which are parameterized. The following table lists the relevant parameters:

DC Parameter	Parameter description
w	Phase window, allows to condition on a phase of a system
d	maximum delay, i.e. size of the feature vector
K_0	Number of initial analogs, i.e. number of analogs to be selected to generate leadtime 1 forecast
K	Number of analogs for leadtimes higher then 1
$M^{(max)}$	Pruning threshold; Ensembles of size greater then $M^{(max)}$ are subject to pruning
L	Maximum leadtime

Ideally, all the parameters except for the maximum leadtime should be set via an optimization procedure. When an archive is of moderate size this is not computationally intensive. For large archives, some parameter values might need to be set subjectively as the computations become intensive. For small archives we face the risk of overfitting.

Here we provide some insight regarding the parameter values when working with the seasonal weather forecasting data of Section 4.3. The phase window w in that case relates to the monthly temperatures, and it should be safe to set it to $w = 0$. A rule of thumb value for the maximum delay is 3, representing a quarter of a year. Quarterly comparisons seem to be a reasonable base when working with seasons. The number of initial analogs is set to 16. The ensemble growth rate is set to $K = 2$ which keeps the ensemble size manageable up to the leadtime of 12 months, where the ensemble has 4,080 members. For small seasonal weather datasets, the pruning threshold often does not need to be applied. For larger datasets it should be set according to the computational resources available. For desktop computers with less than 4 cores and less then 16GB of memory a good maximum size to maintain is $64 \leq M^{max} \leq 1024$.

The setting described in the previous paragraph is one that we use in our applications below. We note that in practice the setting should be optimized for a given application. The methods of optimizing the parameter setting are beyond the scope of this work.

4.2 Numerical illustration

Here we provide more insight regarding the DC. Using a very basic example, we show the typical behavior of the DC forecasts. We also aim to illustrate a tree-like structure of the growing DC ensembles. Since an intuitive understanding is our main concern, the systems used in this section are simple; we use a sine function without a noise, as well as a noisy sine function.

The sine function system is mainly to demonstrate how the DC constructs its forecasts. Before applying DC to real-world data, we use a basic chaotic system of a forced damped pendulum, which demonstrates DC performance in a more complex setting.

4.2.1 Demonstration using a sine function

For our first numerical demonstration of the DC, we choose to forecast a sine function (no noise applied). Although a trivial task from the forecasting point of view, this simple example will demonstrate that DC indeed has the properties required in Section 4.1.

The sine function,

$$x = \sin(t) \tag{4.5}$$

is sampled at the frequency of $\frac{\pi}{2^4+1}$ over 128 of 2π periods. This gives 16 values to be forecasted within each interval of length 2π. The dataset is split into two parts, forming training and testing sets, each consisting of 64 periods.

The DC is initialized 64 times, yielding one (multiple-leadtime) forecast for each period in the testing set. The first forecasted leadtime is $\frac{\pi}{17} + 2\pi k$, the last is $2\pi + 2\pi k$. The DC parameters are set as follows:

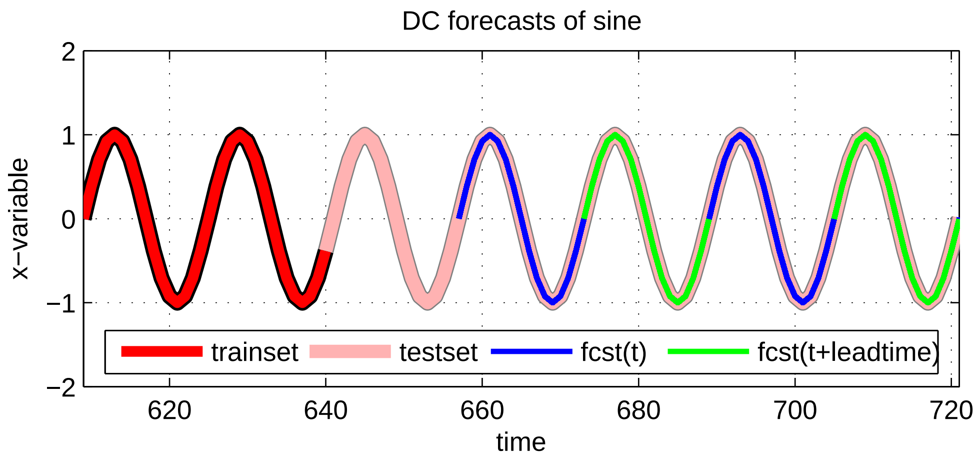

Figure 4.1: **Forecasting sine function:** DC correctly identified analogs within the training set (red), as demonstrated by the perfect forecast (green/blue) at different initializations. All ensemble members, at all leadtimes and initializations of the testing set (light red), lie on top of each other and on top of the forecasted target.

$$\begin{aligned} w &= \pi \\ d &= 8 \\ K_0 &= 4 \\ K &= 2 \\ M^{(max)} &= 64 \\ L &= 16 \end{aligned}$$

This setting yields 16 ensemble forecasts for each initialization, with the ensemble size growing at the rate M_l^l, l being the leadtime. The size of the initial ensemble is 4 and an ensemble is pruned via random sampling when it exceeds a size of 64 members (reached at the leadtime $l = 6$). Although sine function represents a periodic system, the phase window is set to be wide enough to encompass all observations within a period. In other words, the phase is not considered. The size of a feature vector is set to $d = 8$ so that we have $\mathbf{x}_t = \{x_t, x_{t-1}, \ldots, x_{t-7}\}$.

Fig: 4.1 shows the DC forecasts, which are colored in alternating fashion (blue/green) to distinguish different initializations. The main result here is that all the ensemble members across all leadtimes and initializations lie exactly on top of the verification (light red). This means that the KNN search has correctly identified (ideal) analogs to each query point x_t. Due to the perfect periodicity, locating an ideal analog means that the image will be a perfect forecast of the forecasted state. Consequently, the image-analog differences yield a perfect forecast when added to a query point x_t.

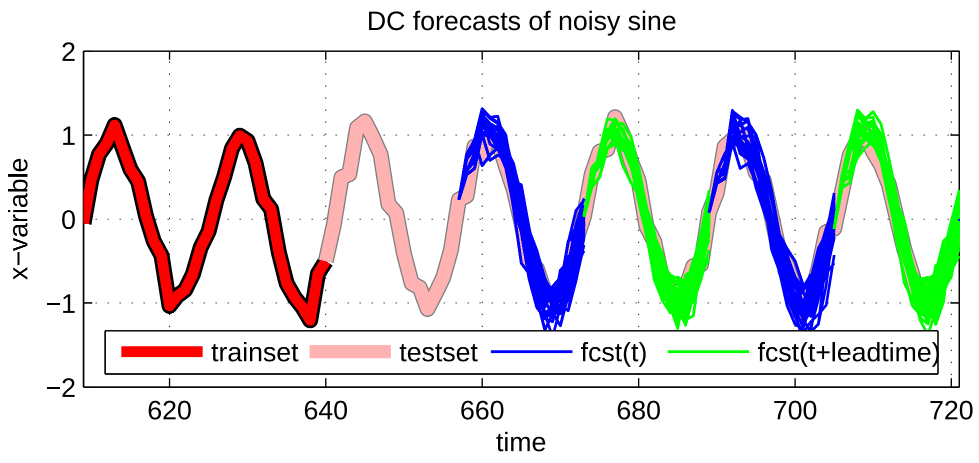

Figure 4.2: **Noisy sine forecasts:** Adding noise to the sine function makes it more difficult for DC to identify analogs. Compared to Fig: 4.1, the ensemble forecasts (blue/green) are no longer perfect. This is expected, as the training set (red) no longer contains perfect analogs.

Forecasting the simple sine shows that DC indeed is able to identify correct analogs and produce good forecasts in a simple setting.

We now slightly increase the complexity of the exercise by including additive noise. The Eq: 4.6 becomes

$$x = \sin(t) + \epsilon \tag{4.6}$$

where $\epsilon \sim N(0, 1)$ is a noise term and we apply DC to forecast the noisy sine function. By adding the noise, the perfect periodicity of the system is removed, making it more difficult to find analogs for DC.

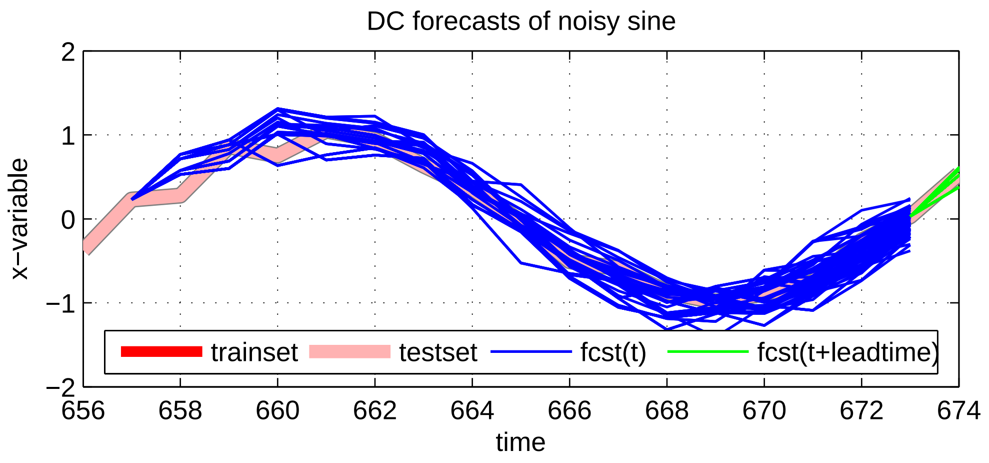

Figure 4.3: **Detail of noisy sine forecast:** A detail of a single initialization of the DC forecast (blue) showing the ensembles growing with leadtime. Beyond leadtime 6, the pruning method starts being applied, keeping the ensemble size at 64. The initial ensemble is of size 4, we observe 2 groups of 2 ensemble members very close to each other at leadtime 1.

The DC is left with the same parameter setting as in the simple sine example. The DC forecasts, shown in the Fig: 4.2, no longer lie on top of the targets. Nevertheless, the DC displays a very good qualitative forecasting performance.

Zooming in on one of the initializations in Fig: 4.3 we get a better picture of the growing ensembles. We can see that at leadtime 1 we have 2 distinct groups of 2 ensemble members, which then double at each leadtime until leadtime 6 where the pruning procedure starts restricting the ensemble size. The ensembles are rather tightly scattered around the verification, with the exception of leadtime 1. Note that the leadtime 1 ‘miss’ is specific to the particular initialization plotted here.

We note that for both sine examples, the conditional climatology (conditioning on phase of the sine) is able to perform comparably to the DC model as long as there is a moderately large archive, say 2^8 states. This is because the system is too simple, or more precisely, the training set contains enough information for the conditional climatology to uncover the recurrent behavior. In more complex systems the DC indeed does outperform conditional climatology at short leadtimes as expected. In the next section, where we use DC to forecast a pendulum, DC outperforms climatology by about 1 Bit, even at leadtime 15, which is beyond a typical period of the system.

4.2.2 Demonstration using forced damped pendulum

Before moving on to real-world applications, we test DC on the forced dumped pendulum (FDP), a more complex system that in many ways resembles the real-world applications. FDP is a simple nonlinear dynamical system exhibiting chaotic behavior. It is a periodically forced system, i.e. it has a phase upon which conditional climatologies may be based. The phase information and nonlinear (potentially chaotic) behavior links FDP well to the applications analyzed below (Section 4.3). Some details of FDP are provided in Appendix A.3; for a detailed overview of FDP and other systems based on the pendulum see [8].

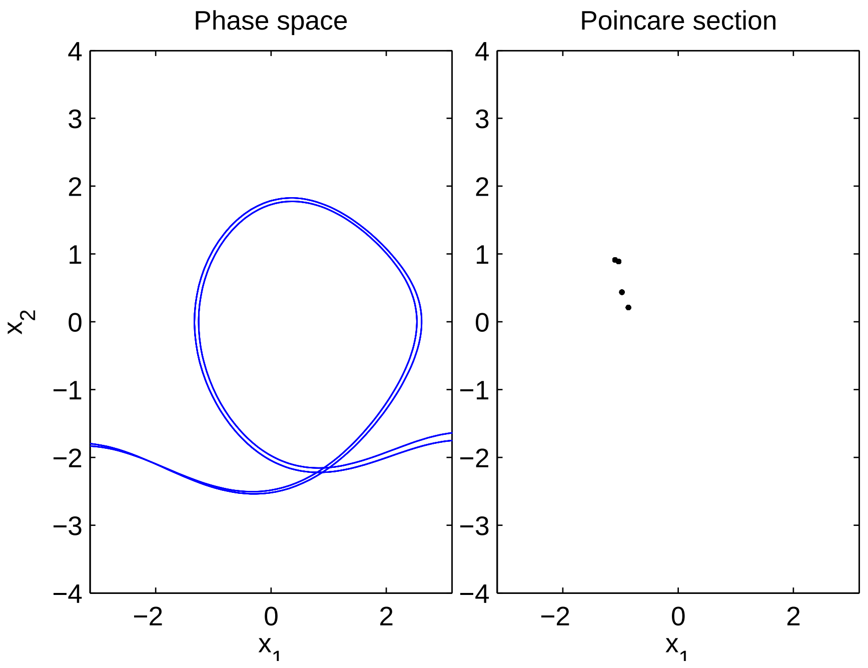

Figure 4.4: **Pendulum in periodic (non-chaotic) regime:** The left panel shows a phase plot of the FDP at a stable non-chaotic regime. The system trajectory displays periodic behavior. The Poincare section on the left shows only a small number of discrete points, a clear sign of periodicity. The parameter values are: $\omega_F = 2/3$, $b = 2$ and $g = 1$.

In this example, we use FDP in a chaotic regime (see App A.3 for parameter values of the different regimes). In the chaotic regime, the recurrence and periodicity are broken through the instability of the system. The chaotic behavior becomes apparent when we compare Fig: 4.5 and Fig: 4.4. In Fig: 4.4, the pendulum exhibits a periodic regime; its phase plot (left panel) displays simple periodic behavior. The Poincare section is also very simple. In Fig: 4.5 the pendulum is in a chaotic regime. Its phase plot is very congested and a Poincare map complex. The simple periodic behavior we observed in Fig 4.4 is no longer present.

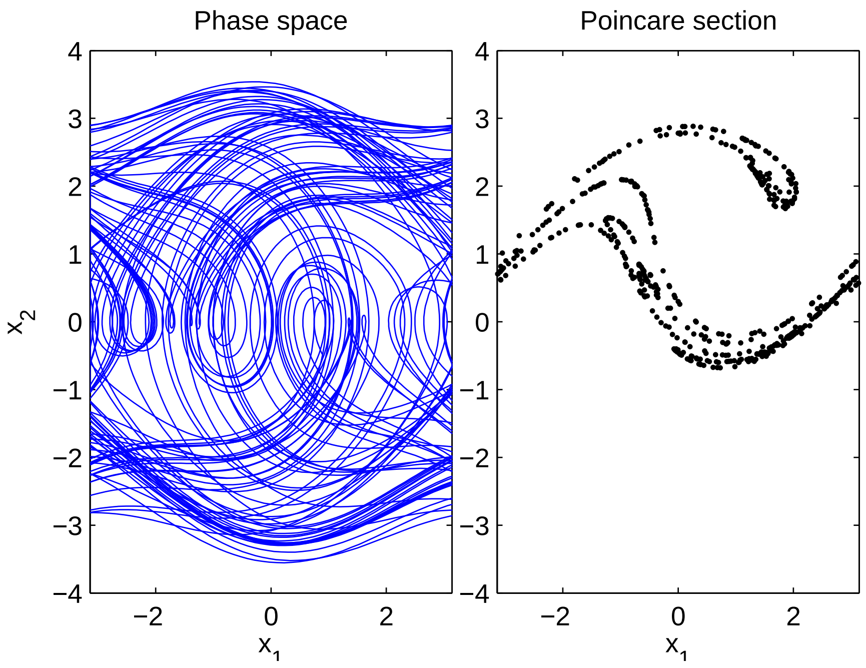

Figure 4.5: **Pendulum in chaotic regime:** The phase portrait (left) reveals a very complex structure. The Poincare section (right) also show a complex pattern suggesting a chaotic behavior. The parameter values are: $\omega_F = 2/3$, $b = 2$ and $g = 1$.

In this demonstration, we only forecast one of the two system variables - the angular velocity labeled x_2 in Fig: 4.5. The DC parameters are as in Section 4.2.1: $w = 0$, $k = 4$,$M^{max} = 64$. The initial ensemble size is set to $K_0 = 16$, and the pruning starts at leadtime $l = 4$. There are 80 forecast initializations in the testing set, each of which forecasts 30 leadtimes ahead. The dataset and evaluation settings are described in Appendix B.

In Fig: 4.6 we present the raw DC forecasts of the angular velocity; as before, the forecasts are colored in alternate fashion (blue/green) to distinguish different initializations. The horizontal axis shows dimensionless time. To put the forecasting horizon into perspective, note that a simple (non-chaotic)

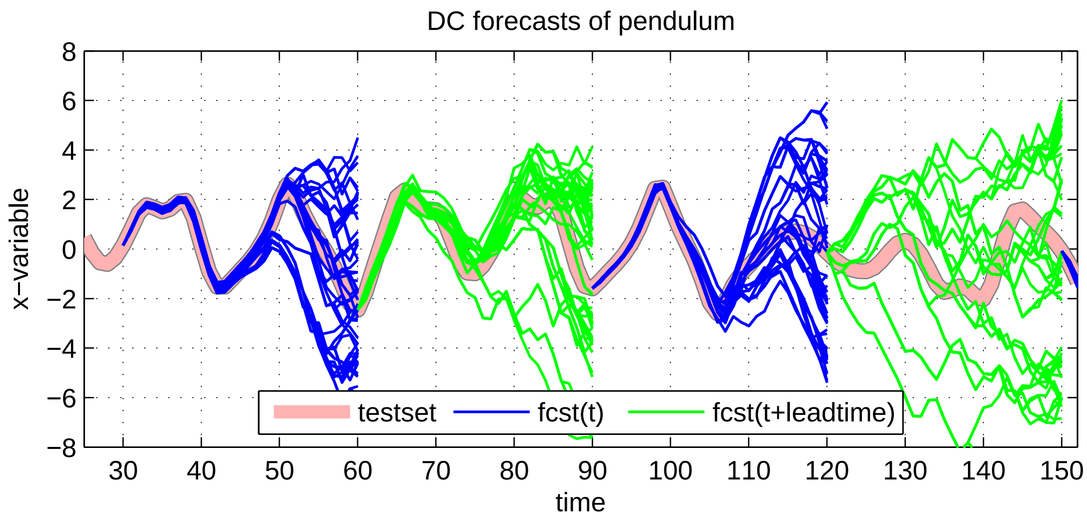

Figure 4.6: **DC forecasts of pendulum:** For consecutive initializations of the DC forecasts (alternating blue/green) of the angular velocity (light red) of FDP in the chaotic regime. Three of the four displayed forecasts successfully capture even a rather complex behavior (peaks/troughs) up to 2/3 of the maximum leadtime, $L = 30$. The initialization at $t = 1.57$ (green) performs poorly relative to climatology.

pendulum takes 20 time units to undergo 1 period. The forecasting leadtime of 30 time units therefore corresponds to 1.5 periods of the simple pendulum. The leadtime of 30 was, therefore, chosen to forecast time scales longer than one period of a simple pendulum.

The skill is quantified in Fig 4.8. For the first, second, and third forecasts plotted, the DC performs rather well up to about a half of the maximum leadtime, i.e. about a half of a single pendulum period. For the first forecast the DC very successfully passes the double peak and the trough. The forecasts of the fourth initialization visibly perform poorly at all leadtimes.

The growing ensembles seem to work quite well, as many of the ensemble members lie close to the forecast even at long leadtimes. This is better

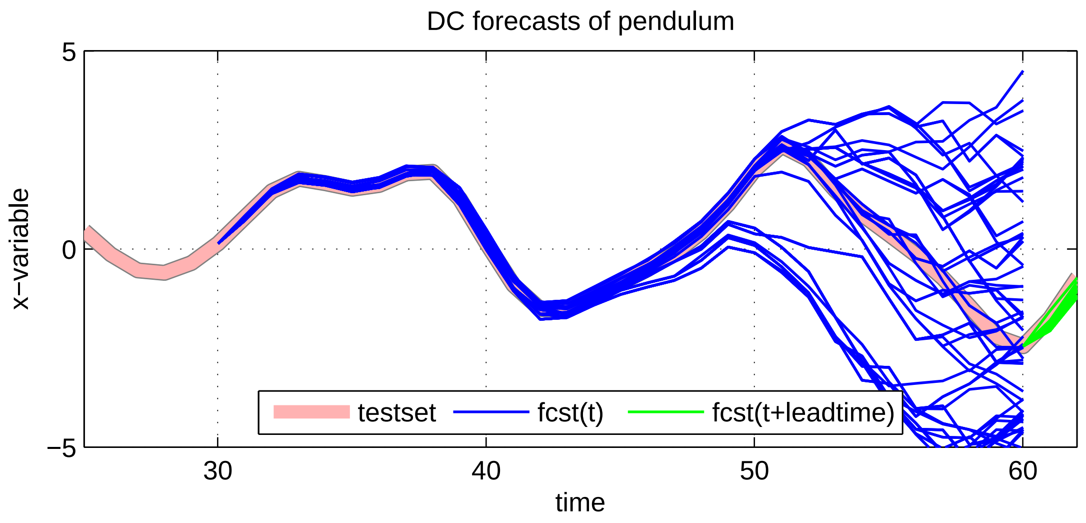

Figure 4.7: **DC forecast detail:** Detail of the first initialization displayed in figure 4.6. The nonlinearities of the system cause the ensemble forecasts to spread out. The growth of an ensemble is apparent beyond time of 1.525

demonstrated in Fig: 4.7 where we zoom on the first forecast. Beyond the time of 46 units, the DC forecasts diverge. Despite that, the ensemble at the long leadtimes ($l > 20, t > 50$) nicely spread around the verification. Although, the long leadtime forecasts are not very close to the verifications, the forecasting PDF still may assign non-negligible probability to the verification.

The actual forecasting performance is captured in Fig: 4.8, where we plot the Ignorance of the Kernel dressed forecasts relative to unconditional climatology. The unconditional climatology defines a zero skill reference (black line). For a model to outperform the zero skill reference its Ignorance relative to the reference must be below the zero skill line. In this figure, the ignorance of DC relative to unconditional climatology stays below the zero skill up to leadtime 18. DC significantly (> 3 Bits) outperforms the climatology at the short leadtimes gaining an advantage of up to 4 Bits before leadtime

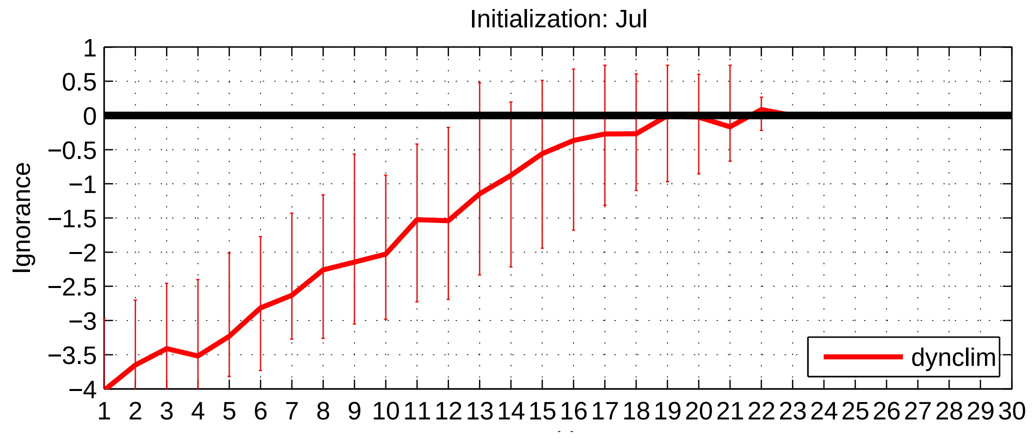

Figure 4.8: **DC Ignorance in pendulum:** Ignorance of the DC forecast relative to climatology. The zero line (black) defines zero skill reference, i.e. skill of the unconditional climatology in this case. For a model to outperform climatology, the relative Ignorance must be below the zero line. The DC forecast displays a good forecasting performance up to leadtimes 15-18. Beyond that, the climatology takes over.

5. DC maintains a good performance (> 1 Bit) up to leadtime 15, which corresponds to about 3/4 of a simple pendulum period. Given the chaotic nature of the system, beating the climatology at time scales comparable with 3/4 of a simple pendulum cycle is, in our opinion, rather satisfactory. These quantitative results support our theoretical expectations, in particular the requirement of DC to outperform climatology.

We note that in a chaotic regime there is no obvious period that could be used in order to create conditional climatology, hence the climatology is unconditional.

A different look at the DC forecasts is provided in Fig: 4.9, where we plot the forecasting densities of DC ensembles. The figure displays 5 subsequent initializations. The short leadtimes exhibit sharp forecasting distributions.

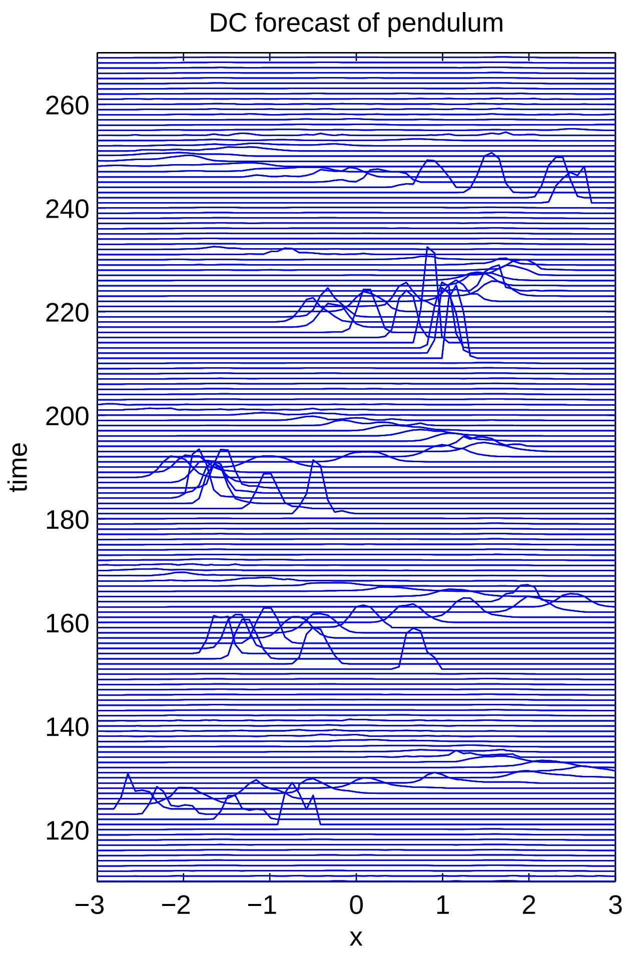

Figure 4.9: **DC density in pendulum:** Forecasting densities of the DC ensemble forecast. Selection of five different initializations of the DC forecast are shown. At the initial leadtimes the densities are sharp, with increasing leadtime they spread out as DC loses forecasting performance.

At long leadtimes, the distributions spread out as DC loses forecasting skill.

Another way of studying the forecasting performance is to plot probability plumes of the forecasting distributions, see Fig: 4.10. In this figure, each patch of a distinct color shows a contour of a given value of forecasting density

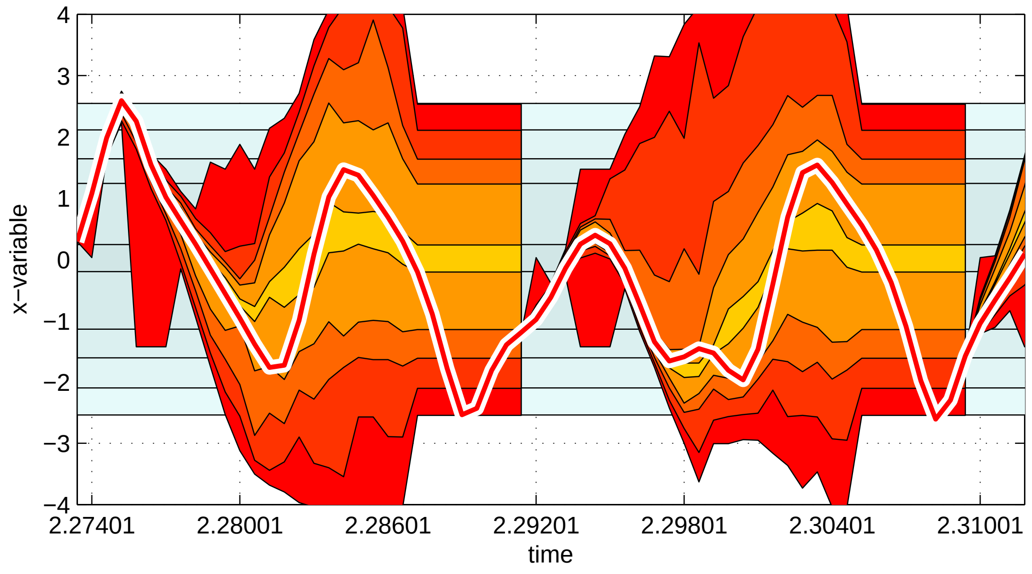

Figure 4.10: **DC Probability plumes (pendulum):** The orange to yellow patches represent probability plumes of the DC forecast. Each patch of color is a contour of a given percentile of a forecasting distribution. The contours are created by connecting a given percentiles values over all leadtimes. Note that the percentiles of climatology (light blue) are not changing; this is because we deploy unconditional climatology, which does not change over time. Also note that at long leadtimes the plumes of DC coincide with climatological plumes. This is because DC forecasts lose skill and climatology takes over, i.e. the blending parameter $\alpha = 1$.

percentiles as they are joined over all forecasted leadtimes. It is similar to looking at the densities of Fig: 4.9 from the top and coloring percentiles of the PDFs, with the distinction that here we use CDFs. The percentiles of unconditional climatology (light blue) are also plotted. The climatological plumes are just straight lines since the unconditional climatology is constant across different leadtimes.

An observation to make is that at a given point, DC plumes converge to the climatological percentiles. That is a moment where the DC loses its forecasting advantage (the blending parameter α becomes zero). Extending the forecast to a larger leadtime L is a waste of computational resource as the DC component of the forecasting PDF is effectively ignored.

4.3 Application: DC Benchmarking of ENSEMBLES

Any simulation model is expected to outperform climatology, but to obtain a more complete picture of a model's forecasting skill we should use more complex benchmarks. In this section, we aim to obtain a good understating of the forecasting skill of the ENSEMBLES [36, 57] models (see section 2.5.3). To do that, we evaluate the forecasting performance of ENSEMBLES relative to climatology, but also relative to a more complex benchmark, the Dynamic Climatology (DC). As described above, the ENSEMBLES models include 5 global coupled atmosphere-ocean climate models (described in Section 2.5.3). Due to technical issues on our part, the model operated by the Euro-Mediterranean Centre for Climate Change is excluded from the analysis.

Both the ocean and atmospheric parts of the models were initialized with estimates of their states. Each model was initialized with 9 different initial conditions, producing an ensemble forecast consisting of 9 ensemble members. For more details on the initialization methods see [152].

All 5 models produce retrospective forecasts over the period of 1960 to 2001, with 4 forecast initializations each year. Although all the models produce global forecasts the focus of our benchmarking is on two regions - Nino3.4 and the Main Development Region (MDR). As described above, these regions are important for the measurement of sea surface temperatures for the El-Niño event and for hurricane formation in the Atlantic basin.

The evaluation part of the exercise utilizes verifications taken from the ERA-40 reanalysis [85], a dataset consisting of global atmosphere and surface conditions over the period of 1957 - 2002 constructed by the ECMWF. For more details on the dataset used see Appendix B.3.

4.3.1 Experimental design issues and DC calibration

The details of the datasets used, along with the description of evaluation settings, are provided in Appendix B. Here we state only basic facts:

(1) We work with a forecast of the spatially-averaged monthly Sea Surface Temperature (SST).

(2) The (retrospective) forecasts are produced by four of the ENSEMBLES models with the first being run in February 1960 and the last in November 2001, yielding 41 years of forecast. Consequently, the forecasts yield 41 values for each leadtime (month) of a given initialization.

(3) There are 4 different forecast initializations, 3 of which produce forecasts of 7 months ahead and 1 which forecasts 14 months ahead

(4) We used DC to produce forecasts of the same variable, period, initializations and leadtimes, i.e. the DC and ENSEMBLES forecasts are fully comparable.

The formation of probability forecasts from an ensemble adds additional risks for information contamination. The better balance when data are precious remains an active area of research. Inasmuch as the effective forecast-verification archive consists of only 41 forecasts per an initialization, we avoid crossvalidation via separate training and testing sets; the data are too precious. Instead we adopt a leave-one-out crossvalidation.

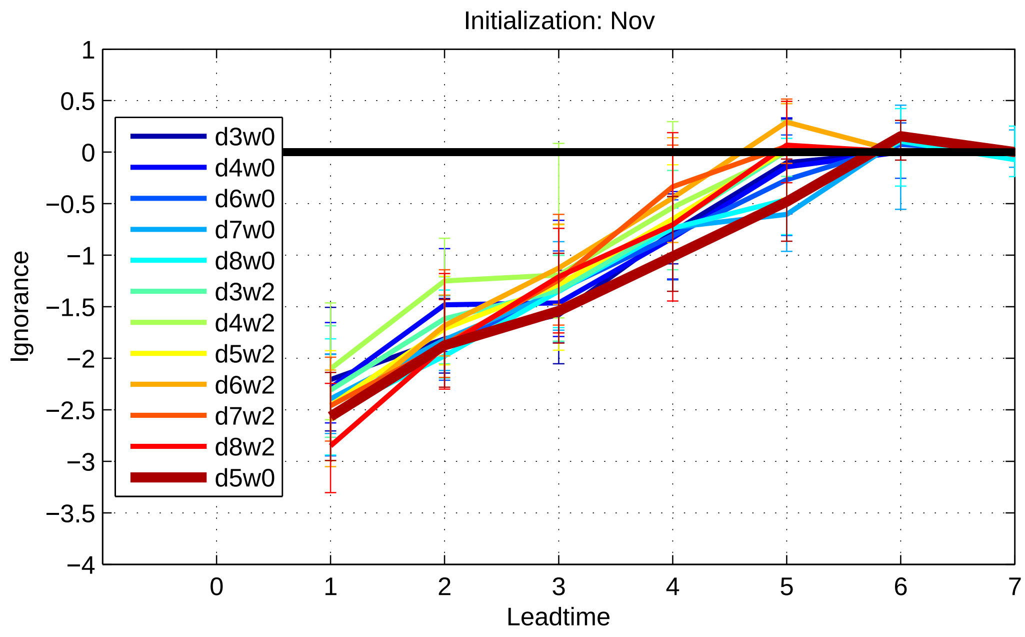

Figure 4.11: **Varying DC parameters (Nino34):** Parameters of the DC are varied and the resulting DC forecasts are evaluated relative to climatology. The best performer is setting with the delay $d = 5$ and phase window of $w = 0$. The zero skill (black line) is defined by conditional (monthly) climatology.

To produce the DC forecast, we adopt a leave-one-out approach. When forecasting a given month, the archive that DC searches for analogs consists of all months apart from the one being forecasted. Although one must take care not to overfit, it is unavoidable that, to some extent, both the DC and the ENSEMBLES models are trained/tuned in-sample. To the advantage of DC, the leave-one-out approach reduces the in-sample effect, to some extent. The only true measure of skill is an out-of-sample evaluation; in seasonal forecasting this will require waiting many years.

Another problem caused by scarcity of the data regards the restriction placed on K_0 and the ensemble growth K. Given only 41 datapoints, there may be no 'good' analogs to be found, much less K_0 of them. While this weakens DC as a forecasting method, it strengthens it as a benchmark that a simulation model should be able to beat.

We also need to be aware of issues related to evaluation. As discussed in Section 2.4.4, for small datasets a leave-one-out crossvalidation is preferred over subsampling. A further consideration is the robustness of the chosen skill score due to small size of the archive, and also potential bias due to small ensemble size demonstrated in Chapter 3.

The DC parameters are set via optimization of DC forecasting performance. In Fig: 4.11 we show evaluations of DC forecasts for the Nino3.4 temperatures for different parameter settings. The meaning of the figure is the same as described in Fig 4.8. The zero skill reference is, in this case, defined by conditional (monthly) climatology, which will be the zero skill reference in the subsequent analyses of this chapter unless stated otherwise. The overall best performer is the setting with the delay set to $d = 5$ and phase window of $w = 2$. We have also performed sensitivity analysis regarding the rest of the parameters. Based on the results, the settings for Nino34 are: $K_0 = 2$, $K = 2$, $d = 5$, $w = 2$, $m = 64$; and for MDR:$K_0 = 2$, $K = 2$, $d = 4$, $w = 0$, $m = 64$. We note that the size of the window w is sensitive to the frequency of the data. If we were to use weekly data the window size would be likely to differ from its counterpart based on the monthly data. We remind the reader that the parameters are discussed in Section 4.1.5.

4.3.2 ENSEMBLES, DC performance in Nino 3.4

In this section, we evaluate the forecasting performance both of the ENSEMBLES models and of the DC relative to climatology. In addition, we evaluate a multi-model ensemble (MM) constructed by combining equally weighted ensembles of the 4 ENSEMBLES models. Due to time constraints, we restrict the MM application only to the important November initialization. However, the arguments made below are general. Finally, we also present an evaluation of the ENSEMBLES models relative to the DC.

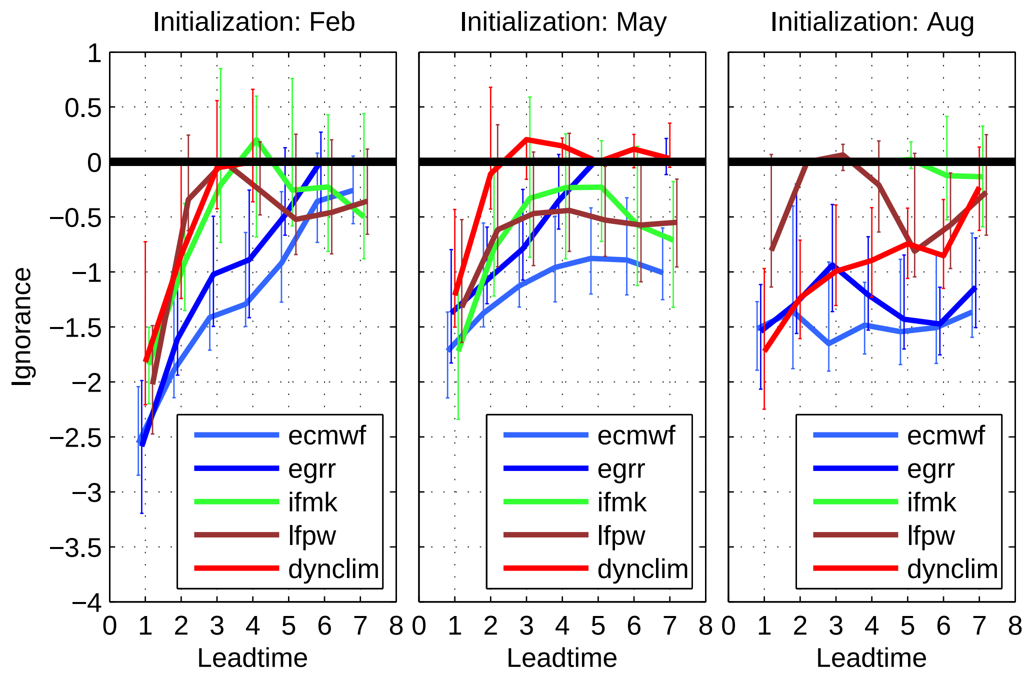

Figure 4.12: **DC Ignorance in Nino34:** Three initializations of ENSEMBLES and DC forecasts evaluated relative to climatology. The August initialization shows a poor performance by the LFPW and IFMK, while the DC outperforms both. The ECMWF is the best performer in all three initializations. The overall performance of the DC is comparable (or better at points) with the ENSEMBLES.

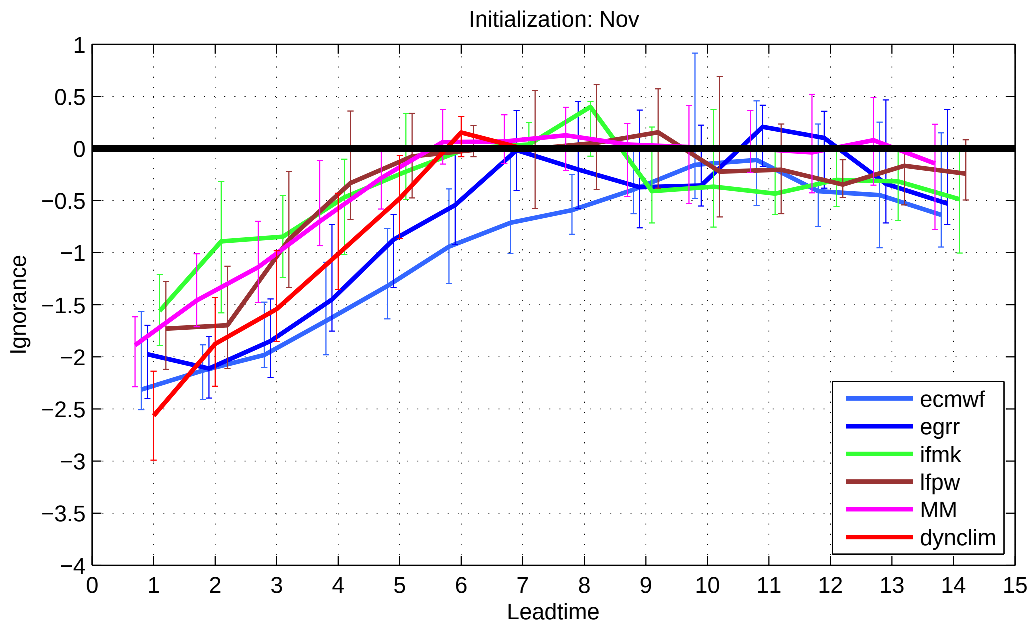

Figure 4.13: **DC Ignorance of the 'long run' in Nino34:** Same as in Fig 4.12 but for November initialization. The ECMWF is the best overall performer although at leadtime 1 the DC outperforms all the ENSEMBLES models. The DC, IFMK and LFPW models maintain forecasting skill up to leadtime, 5 EGRR up to leadtime 7, and ECMWF considerably longer (up to leadtime 10). There seems to be an improvement in performance toward the final leadtime for all ENSEMBLES models.

In Fig: 4.12 we show the Ignorance ENSEMBLES and DC relative to climatology for the initializations of February, May, and August. We also provide crossvalidated error bars to assess the significance of differences between particular models. The most striking result is the August initialization, where the IFMK has almost no advantage over climatology and LFPW performs poorly. The ECMWF and EGRR models perform rather well across the initializations and leadtimes, with the ECMWF being the best performer.

Starting with the February initializations, the DC model performs well at the first two leadtimes (> 1 bit) and, similarly to IFMK and LFPW, saturates beyond a leadtime of 3 months. Note that the error bars of IFMK and LFPW reach the climatological level already at leadtime 2. In February, the overall performance of DC is comparable with all the ENSEMBLES models at the short leadtimes and with the IFMK, LFPW at long leadtimes. In the May initialization, DC is comparable to the ENSEMBLES models only at leadtime 1. For the August initialization, DC significantly outperforms IFMK and LFPW, by more then 1 Bit, up to a leadtime of 3 months. The IFMK and LFPW models are already not significantly different from climatology at leadtime 1; however, LFPW regains skill at leadtime 6, where its error bar descends below the climatological level.

In general, IFMK and LFPW maintain forecasting skill up to 3 months with the exception of August, although at some leadtimes the significance may be questioned, as their error bars reach climatology. The ECMWF model performs well for considerably longer, up to 7 months. In the February initialization, both the ECMWF and EGRR maintain a very good performance over the entire forecast period and DC is a comparable performer to the ENSEMBLES models. The statistical significance of their performance is generally confirmed by the crossvalidated error bars.

The Ignorance in Fig: 4.13 shows a similar picture for the November initializations. IFMK and LFPW are outperformed by ECMWF and EGRR although the out-performance is not statistically significant at leadtimes 2 and 4 in case of the IFMK model. Both IFMK and LFPW lose skill relative to climatology around leadtime 5, but their crossvalidated error bars reach climatology at leadtime 4. The DC performs comparatively well and, at short leadtimes, outperforms IFMK although significantly only at lead time 1. At leadtime 1 DC also outperforms ECMWF and EGRR although not significantly. The ECMWF model is again the best performer, never signif-

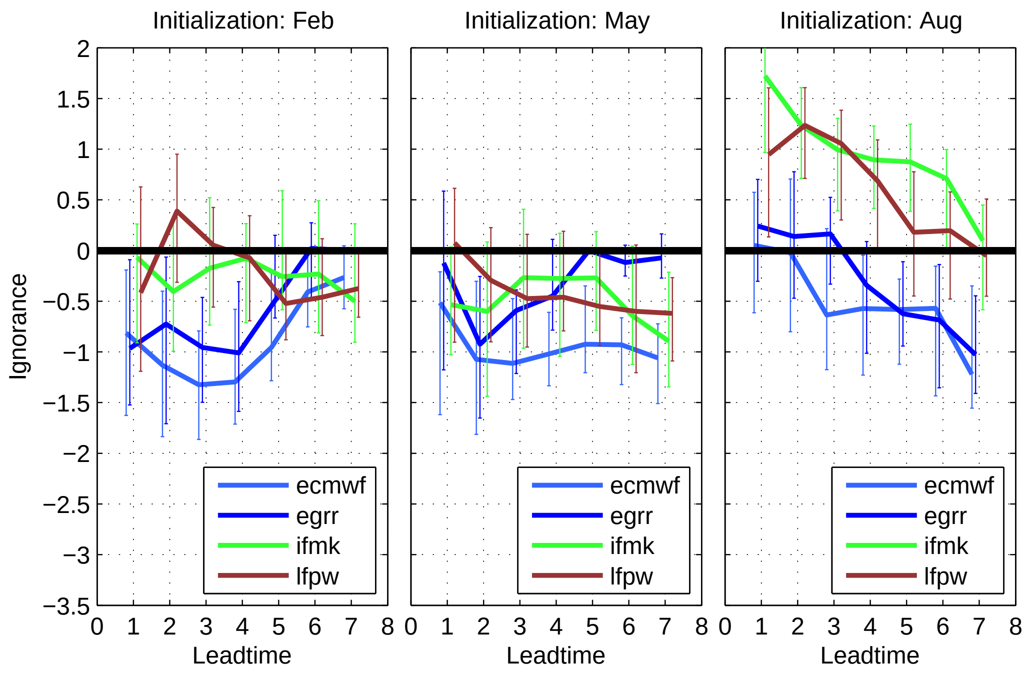

Figure 4.14: **Ignorance relative to DC in Nino34:** Same as Figure 4.12 but here ENSEMBLES is evaluated relative to DC. Above the zero line a model's performance is worse then that of DC. The DC is significantly outperformed only by ECMWF and EGRR in the February initializations and at long leadtimes of the August initializations. For the August initialization the DC outperforms IFMK and LFPW. Overall, the DC performs comparably to ENSEMBLES.

icantly crossing the zero line defined by the zero skill of climatology. The ECMWF model maintains good performance up to leadtime 9. Interestingly, at long leadtimes there seems to be a return to skill as 3 of the ENSEMBLES models drop further below the zero line. However, the cross-validated error bars show that the return to skill is not statistically significant.

The Ignorance profile of the multi-model shows that the MM is comparable to the two weakest models - IFMK and LFPW. The DC benchmark outperforms the MM, although not significantly, up to leadtime 5. This result may seem in stark contrast to [152], who find that the MM generally outperforms the individual models. However, the important difference between our work and [152] is that our evaluation approach involves Kernel dressing (KD). As suggested by [152], MM improves performance by reducing overconfidence of individual ensembles. By combining individual ensembles, the spread of the resulting grand-ensemble is inflated and overconfidence reduced. In our approach, KD implicitly debiases/inflates the ensembles at an individual model level (see Section 3.2 and Section 2.5.4). While in [152] raw (overconfident) ensembles are compared with the (spread-inflated) MM, we compare ensembles that are already debiased/inflated with the MM. In our case, both the individual ensembles and the MM perform similarly, because both have been debiased. In our approach KD does a similar job to that which MM does in the approach of [152]. Consequently, in the KD/Ignorance evaluation approach, the MM is not required in order to improve performance of individual models.

We can obtain a better insight into DC performance by directly evaluating the ENSEMBLES models relative to the DC, as shown in Fig: 4.15.

The figure clearly shows that DC forecasts outperform all the models at leadtime 1, where only for the ECMWF model the statistical significance does not hold. Also DC performs better than, or comparatively well to, IFMK, LFPW and MM, although the error bars do not ensure statistical significance. The ECMWF and EGRR models significantly outperform DC at leadtimes 4 to 7 for which the out-performance is significant. At a leadtime of 5 months, for example, ECMWF is almost 1.5 Bits better then DC, placing almost three times ($2^{1.5}$) more probability on the verification.

The probability plumes of the ECMWF are given in Fig: 4.17 and the DC in Fig: B.2. The meaning of the figures is the same as in Fig: 4.10 (described in Section 4.2.2). The zero skill reference is defined by the monthly climatology

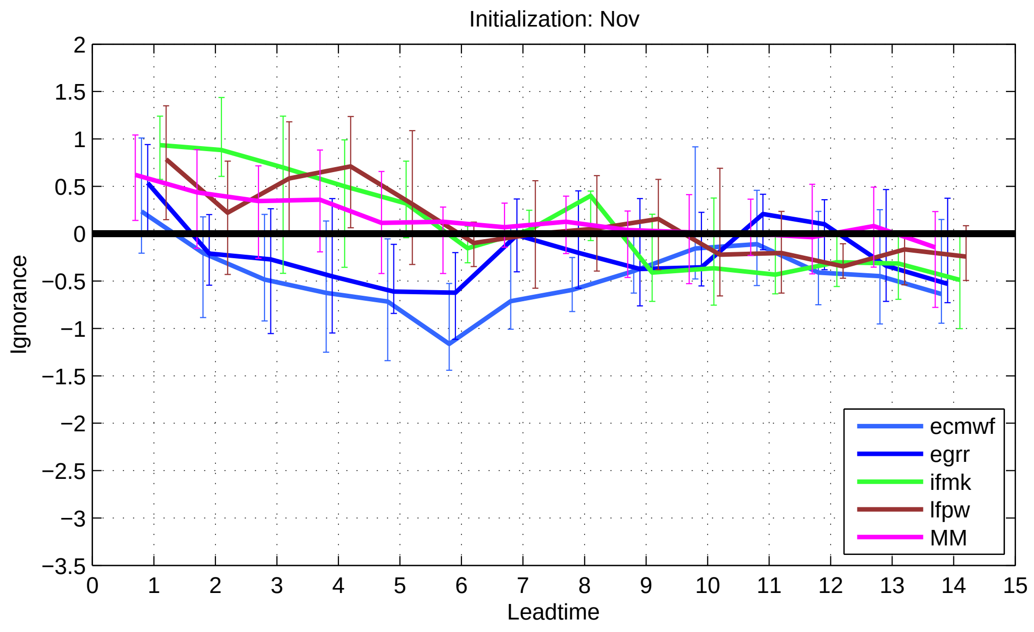

Figure 4.15: **The 'long run' ignorance relative to DC in Nino34:** Same as in Fig 4.14 but for the November initialization. Similar conclusion emerges; the DC is significantly outperformed by only ECMWF and EGRR models at leadtimes 3-7. For all other leadtimes and models, DC is a comparable performer.

(light blue plumes), which changes with leadtime. The plumes of the forecasting model are plotted for the November initialization (so the ends and beginnings of the plumes overlap). We only plot a selected part of the whole timeseries, one that contains interesting El-Nino events. The first fact to observe is that the plumes of the ECMWF models are sharper and never approach the plumes of monthly climatology (light blue). On the contrary, the plumes of the DC start converging toward the climatological plumes when the DC starts losing performance. This is in line with our previous finding shown in Fig: 4.13. Further, note how the DC managed to capture the offset of the large El Nino event during 1973. The ECMWF model also performed very well during 1973, where it sharply identified the variable progression

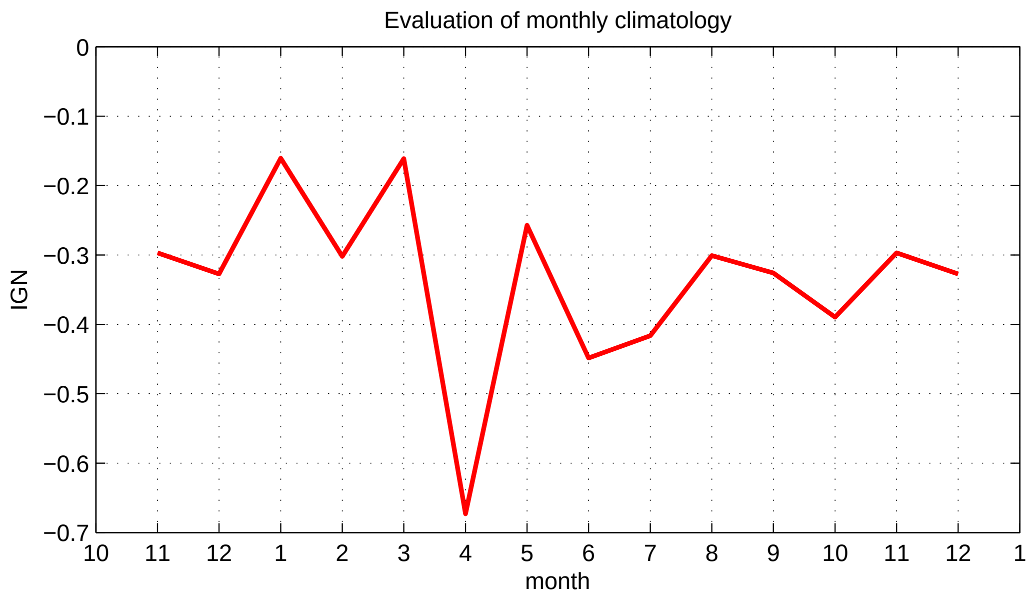

Figure 4.16: **Ignorance of the monthly climatology: Nino34:** The Ignorance of forecasts of monthly climatology shows that some months are more difficult to (climatologicaly) forecast. January and March are the most difficult months; April is rather easy, as the Ignorance of -0.7 is much lower than the January/March Ignorance. A possible explanation of the forecast improvement for November initialization is that the monthly climatology does not provide a strong benchmark, so in relative terms the ENSEMBLES model forecasts improve as the climatological forecast weakens.

of the 1976/1977 El-Nino event. We provide plumes of both the ECMWF and DC models for all forecasted periods of the November initialization in Appendix B.4.

As a final point of this subsection, we discuss the return to skill occurring at leadtimes 12 to 14 of the November initializations, which corresponds to December and January. It is rather odd for a model to regain a forecasting skill after hitting a zero skill mark. A possible explanation of the return

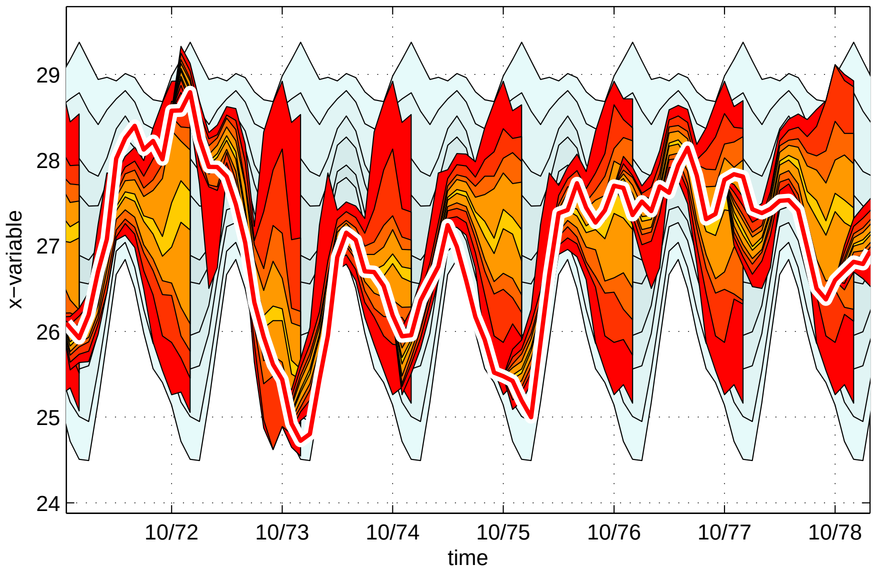

Figure 4.17: **Probability plumes of ECMWF model in Nino3.4:** Percentiles as given by the CDF of the ECMWF forecasts (red/yellow) show the performance of selected November forecast initializations in probabilistic terms. The plumes are plotted against a background of the plumes of monthly climatology (light blue). Note how the climatological plumes vary with leadtimes. This is due to monthly climatology changes with leadtime. The time period selected shows 2 large El-Nino events in 1972/1973 and 1976/1977. The ECMWF forecast performs well in capturing the onsets but also the offsets of the events (red with white rim). Note how the plumes overlap; this is due to the overlapping of initialization of one forecast with the end of the preceding forecast.

to skill is that in December and January the monthly climatology does not pose a strong benchmark. In Fig: 4.16, we show Ignorance of the monthly climatology for months starting in November and going out 14 months up to December of the following year. We plot the months in this way to make the figure comparable to Fig: 4.13. In April, climatology performs well relative to other months; for example, it gains almost 1 Bit of advantage relative to

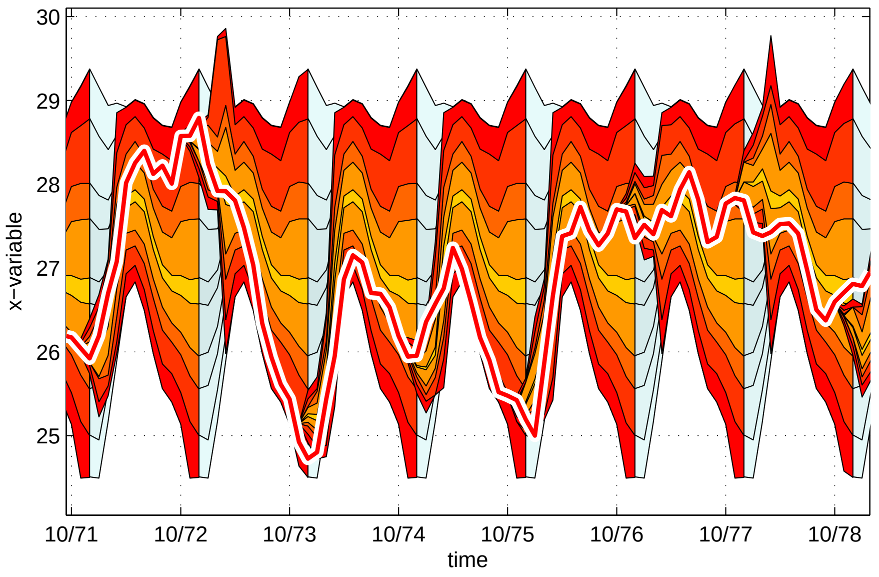

Figure 4.18: **Probability plumes of DC in Nino3.4:** Same as in Fig: 4.17 but for the DC forecasts (red/yellow). The DC does a good job in forecasting the onset of both the 1972 and 1975 events. The forecasting performance is reduced after leadtime 5, beyond which the plumes relax to the plumes of the monthly climatology.

March. The worst performance is delivered in January and March. Although not being the worst months, in terms of climatological performance November and December reach only 0.3 Bits of expected Ignorance. It may be possible that the return to skill of the models is rather a loss of skill of the climatology in these particular months.

4.3.3 ENSEMBLES, DC performance in MDR

Here we follow a similar path to that in Section 4.3.2 and evaluate the ENSEMBLES, a multi-model ensemble, and DC relative to monthly climatology and also the ENSEMBLES relative to the DC model in the MDR.

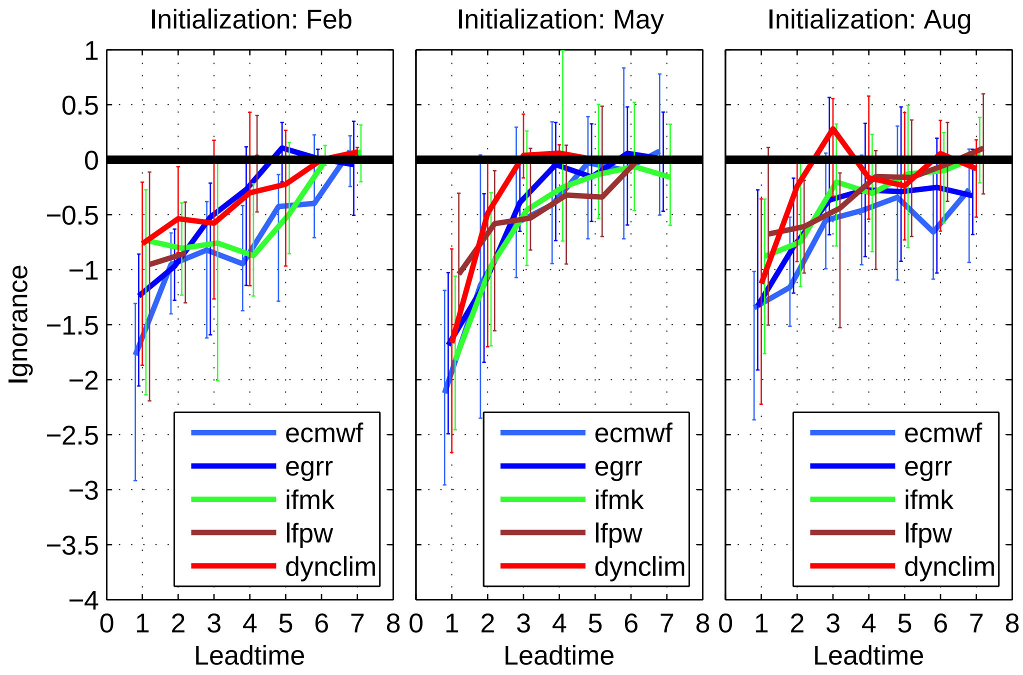

Figure 4.19: **DC Ignorance in MDR:** ENSEMBLES and DC model forecasts evaluated relative to monthly climatology in the MDR region. The ENSEMBLES model performs rather well up to leadtimes 4-5 at all initializations. The DC forecast performs comparably to ENSEMBLES as was the case in the Nino3.4 region.

We begin with the forecasting performance relative to climatology for the February, May, and August initializations in Fig: 4.19. The ECMWF model slightly outperforms all the other models, although never significantly. As compared to the Nino3.4 region, here the models display similar skill but the error bars do not signal statistical significancy. Across the May/August initializations, the models seem to lose forecasting skill before 5 months. The DC model performs comparably to all the ENSEMBLES models across the 3 initializations. For the February initialization, ECMWF and IFMK maintain a very good performance over most of the forecasted period, with their error bars crossing the zero skill of climatology at leadtime 6.

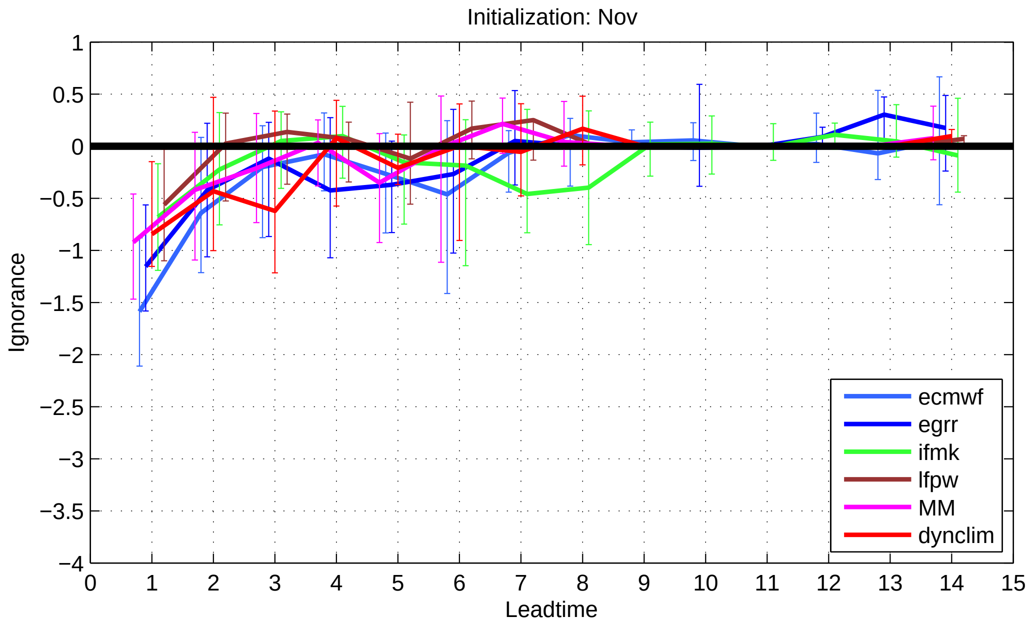

Figure 4.20: **Ignorance of the ‘long run in MDR:** The November initialization evaluated relative to monthly climatology. All the ENSEMBLES models perform poorly as they loose skill at leadtimes 2-3. The DC is the best performer as at leadtime 3 it still has an advantage of 0.5 Bits over climatology.

The November initializations are shown in Fig: 4.20. All the models, including the MME, perform rather poorly and lose skill at leadtime 3. Also, the models do not significantly differ from each other as their respective error bars overlap at almost all leadtimes. The ECMWF, EGRR and the DC models are overall the best performers, but significantly beat climatology only at the first leadtime. DC is the most skillful model at leadtime 3, although the large error bar suggests that the out-performance is not statistically significant. Beyond leadtime 1 the advantage over monthly climatology for all models is about 0.5 Bits at best, which renders all of the models poor performers. The performances of IFMK, LFPW, and the MM lose skill beyond leadtime 1.

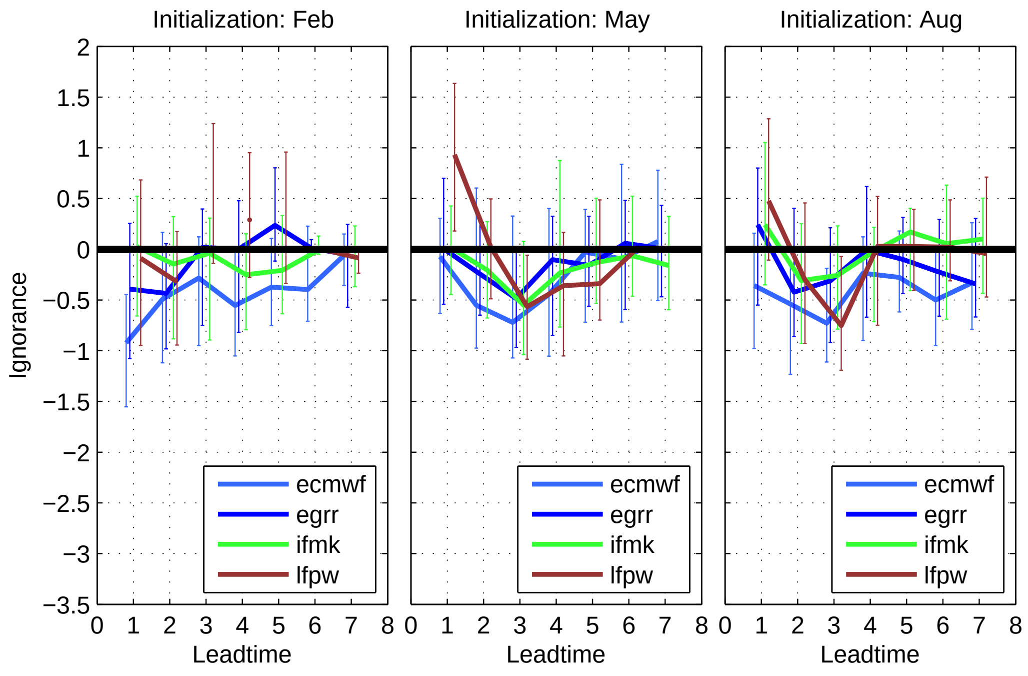

Figure 4.21: **Ignorance relative to DC in MDR:** ENSEMBLES models evaluated relative to DC. None of the models outperforms DC for in any of the three (Feb, May, Aug) initializations.

In Fig: 4.21, the ENSEMBLES models are evaluated relative to DC. Similarly to the Nino3.4 region, the DC model is perfectly comparable with the ENSEMBLES models, although it does not have any specific advantage at any of the initializations. The ECMWF significantly outperforms DC at leadtime 1 of the February initialization. DC significantly outperforms LFPW at leadtime 1 of the May initialization.

For the November initialization shown in Fig: 4.22, the ENSEMBLES models evaluated relative to DC do not significantly outperform the DC with the

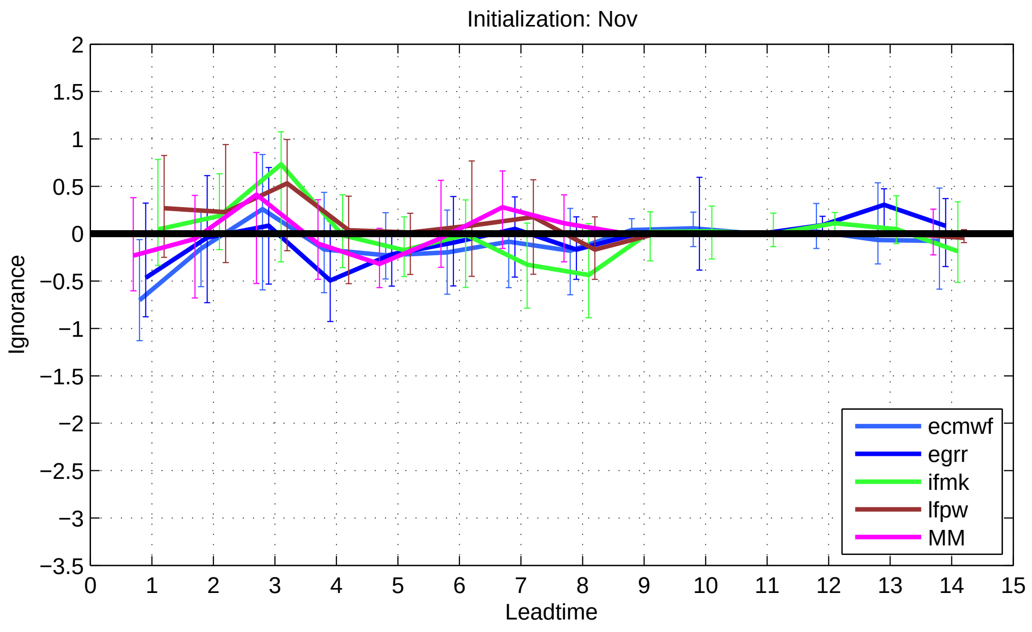

Figure 4.22: **The 'long run Ignorance relative to DC in MDR:** Same as in Fig: 4.21 but for November initialization. Since all models perform poorly at this initialization, none are expected to outperform DC significantly. This is indeed the case; moreover, the DC is the best performer at leadtime 3.

exception of the ECMWF at the first leadtime. Recall that, as demonstrated by Fig: 4.20, all the models are poor at this initialization.

4.4 Conclusions

In this section we have introduced DC, a statistical model based on analogs, which can be used in forecast evaluation as an alternative, stronger benchmark of (un)conditional climatologies. Built-in features which make DC a viable benchmark tool include:

- straightforward application to any periodically-driven dynamical system,
- compensation for reduction in forecasting performance at long leadtimes,
- ability to produce values not previously seen in the training set,
- robust performance at a very low computational cost.

To demonstrate the properties of the DC, we have performed several numerical tests using simple systems. Using both a simple and a noisy sine curve we have demonstrated the ability to identify ideal analogs when these are available (see Fig. 4.1). Using the chaotic pendulum, we have demonstrated how DC grows the forecasting ensemble (see Fig. 4.7). We have also shown that DC displays a good forecasting performance of the chaotic pendulum, as it maintains forecasting skill at leadtimes comparable with up to 3/4 of a period of simple pendulum system (see Fig. 4.8).

To test DC in a real world application, we have considered the SST monthly averages in both the Nino34 and MDR regions, and used DC to benchmark seasonal forecasting performance of four ENSEMBLES models and their multi-model ensemble (MM). The ENSEMBLES models, MM, and the DC were evaluated using Kernel dressing. While the ECMWF model emerged as the best performer across different initializations, the DC posed a strong benchmark, and in many cases performed better than, or as well as, some of the ENSEMBLES models.

For February and August initializations in the Nino34 region (Fig. 4.12), we find that all ENSEMBLES models outperform the zero-skill benchmark of monthly climatology up to 3 months, with EGRR and ECMWF maintaining skill up to leadtimes 6 and 7 respectively. In addition, for the May initialization the ECMWF, EGRR, and LFPW outperform monthly climatology at all leadtimes. For the August initialization, the IFMK model does not outperform climatology at any leadtime, the LFPW experiences loss of skill at leadtimes 2 and 3 and regains skill at leadtimes 5 and 6. The ECMWF and EGRR, along with DC, outperform monthly climatology at all leadtimes. When using the DC as the zero-skill benchmark, only ECMWF and EGRR significantly outperform DC in February and August initializations, although not at all leadtimes. IFMK and LFPW strongly underperform DC in the August initialization. For the May initialization all the ENSEMBLES model outperform DC.

We have also benchmarked the very important November initialization of the ENSEMBLES (Fig. 4.13), which forecasts 14 leadtimes ahead. In this exercise we also included a multi-model ensemble, constructed by combining equally weighted ensembles of the four models into a grand-ensemble. All ENSEMBLES models, including the MM, outperform monthly climatology up to leadtimes 4 to 5. When DC is set to be the benchmark, only the ECMWF and EGRR significantly beat the DC, but only at leadtimes 3 to 6. At leadtime one EGRR is outperformed by DC while ECMWF performs comparably.

For the November initialization, the performance of the MM was relatively weak; comparable to the two weakest models IFMK and LFPW. Although this finding may seem to contradict previous literature on multi-model ensembles, we argue that in Kernel dressing the individual models are implicitly debiased, and therefore the individual models are directly comparable with MM.

Benchmarking ENSEMBLES models in the MDR region leads to similar conclusions. For the February, May and August initializations, Fig. 4.19, all ENSEMBLES models significantly outperform the monthly climatology up to leadtime 3, with ECMWF maintaining the skill slightly longer. When using DC as the zero-skill benchmark (see Fig. 4.21), none of the models (with the exception of ECMWF) significantly outperform DC at the first leadtime of the February initialization. For the November initialization (see Fig. 4.20), none of the ENSEMBLES models maintain skill beyond leadtime 2. Consequently, none of the models significantly outperform DC when used as a benchmark (see Fig. 4.21). Similarly to the Nino34, the MM did not significantly outperform any of the individual models.

Finally, for the November initialization in the Nino34 region, we have observed a return to skill at leadtimes 13 and 14. Although the return to skill is not statistically significant, see error bars of Fig. 4.13, we have investigated the hypothesis that the return to skill is due to variations in climatology. We have found that for the relevant months, i.e. November and December, the Ignorance of monthly climatology is rather weak, around -0.3 in Fig. 4.16. Consequently, the monthly climatology does not pose a strong benchmark to ENSEMBLES models. We take this fact to be an interesting observation rather than an explanation of the return to skill of the ENSEMBLES models, and intend to investigate it further in our future work.

Chapter 5

Forecasting with Radial Basis Functions

In this chapter we provide background information on the use of Radial Basis Functions (RBF) [127] in dynamical system forecasting. As background, this chapter may be skipped on the first reading.

We begin with a description of RBF interpolation [40, 118]. We then discuss RBF approximation and emphasize the differences between the approximation and interpolation [39]. We also discuss the computational costs of the two approaches and describe the RBF training process and the generation of RBF-based ensemble forecasts [130].

A crucial task in RBF approximation is center selection. We discuss several selection methods [41, 43] and choose the power function method as the most suitable for our later analyses.

This chapter is organized as follows. In Section 5.1.1, we provide an overview of the RBF interpolation and approximation. In Section 5.2 we discuss methods of center selection.

In this chapter we merely prepare ground for chapter 6, where RBF modeling is extensively used. The concepts and ideas discussed below are known, and well described in the cited literature, and there are no new contributions here.

5.1 Model fit and ensemble forecast

In this section we briefly discuss Radial Basis Functions (RBF) modeling [39, 40, 43]. We first describe the RBF interpolation, since RBF modeling was originally designed to interpolate functional surfaces [127]. We also define, and provide several examples of, radial functions [43, 44, 55] and state that the Wendland function [153, 154] will be the function of choice in the later analyses.

We explain why, in large datasets, the interpolation problem needs to be reformulated as an approximation problem. We provide a technical description of in-sample fitting of the RBF and out-of-sample ensemble forecast generation.

This chapter provides merely the basic fundaments of RBF modeling. For a detailed account of RBF methods, applications and related concepts see [39, 40, 118], which we loosely follow.

5.1.1 Interpolating with Radial Basis Functions

In many fields a common problem is to interpolate a function $f(\cdot)$ which generates observed values $y_i = f(\mathbf{x}_i)$ at a number of observation points $\mathbf{x}_i$, where $\mathbf{x}_i \in \mathcal{R}^d$ and $\mathbf{x} = \{\mathbf{x}_i\}_{i=1}^N$ is a vector of observation points. The interpolant of f is often constructed as a linear combination of a set of M *basis functions* $\{B_j\}_{j=1}^{M}$ as

$$\tilde{f}(\mathbf{x}) = \sum_{j=1}^{M} \lambda_j B_j(\mathbf{x}) \tag{5.1}$$

where λ_j are coefficients of the basis functions. The coefficients are easily obtained by solving the linear system

$$A\boldsymbol{\lambda} = \mathbf{y} \tag{5.2}$$

where $\boldsymbol{\lambda} = \{\lambda_j\}_{j=1}^{M}$ is the vector of coefficients, and the matrix A holds values of the M basis functions at a given observation point x_i, i.e. $A_{i,j} = B_j(x_i)$.

The matrix A must be invertible for the linear system of Eq: 5.2 to be well posed. To ensure the invertibility of A the *Mairhuber-Curtis* theorem [88] suggests that for d-dimensional spaces with $d \geq 2$, the basis functions B_j must be data-dependent. A natural choice for data dependent basis is the Euclidean distance function $\phi_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_i\|$. Using the Euclidean distance as basis function the system of Eq: 5.2 can be written as

$$\tilde{f}(\mathbf{x}) = \sum_{j=1}^{M} \lambda_j \|\mathbf{x} - \mathbf{x}_i\|. \tag{5.3}$$

where $\mathbf{x}$ are called *datasites* and $\mathbf{x}_i$ are called *centers* [39]. The coefficients $\boldsymbol{\lambda}$ are obtained by solving the linear system

$$\underbrace{\begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \|x_1 - x_1\| & \|x_1 - x_2\| & \dots & \|x_1 - x_M\| \\ \|x_2 - x_1\| & \|x_2 - x_2\| & \dots & \|x_2 - x_M\| \\ \vdots & \vdots & \ddots & \vdots \\ \|x_N - x_1\| & \|x_N - x_2\| & \dots & \|x_N - x_M\| \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix}}_{\boldsymbol{\lambda}} \tag{5.4}$$

When the entries of the matrix A are given as euclidean distances of the datapoints and the centers, A is called the *distance matrix*. Note that in Eq: 5.4 the centers were chosen to coincide with the datapoints. Although in some applications it may be sensible to choose a set of centers $\mathbf{c} = \{\mathbf{c}_j\}_{j=1}^{M}$ such that $\mathbf{c} \not\subseteq \mathbf{x}$, in interpolation problems the centers usually coincide with the datapoints, $\mathbf{c} = \mathbf{x}$, which yields $M = N$, i.e. the matrix A is a square $(N \times N)$ matrix.

Using the distance matrix as a basis function may be convenient but has a significant drawback. First, [39] shows that the distance matrix interpolant $\tilde{f}$ has a rather limited accuracy, especially near the boundary of the datapoints. Second, the smoothness of the interpolated functions is limited. Third, the distance matrix A is often ill-conditioned, i.e. has a low condition number [19]. As the size N of the system grows the ill-conditioning often becomes severe.

The limitations of the distance matrix may be substantially reduced by using a *radial basis function* (RBF). We adopt the definition used in [39] to define a radial function:

Definition: A function $\Phi : \mathcal{R}^d \to \mathcal{R}$ is radial provided there exists a univariate function $\phi : [0, \infty)$ such that

$$\Phi(x) = \phi(r), \quad \text{where} \quad r = \|\mathbf{x}\| \tag{5.5}$$

where $\| \cdot \|$ is some norm of $\mathcal{R}^d$.

Radial function is therefore a function that takes the same value at any point that is located a given distance from a fixed center. The Euclidean distance function used in Eq. 5.3 is a special case of radial basis function. The distance function used in the distance matrix is a simple example of a radial function, since for all points $\mathbf{x}$ located at a given distance from a given center $\mathbf{x}_i$ the function $\|\mathbf{x} - \mathbf{x}_i\|$ takes the same value.

Some commonly used examples of RBF include:

- Gaussian

$$\phi(r) = \exp^{(-\sigma r)^2} \tag{5.6}$$

- power function

$$\phi_\alpha(r) = \|r\|^\alpha \tag{5.7}$$

- truncated power function

$$\phi_\alpha(r) = (1 - r)_+^\alpha \tag{5.8}$$

- generalized inverse multiquadric

$$\phi(r) = (1 + \|r\|^2)^{-\frac{\beta}{2}} \tag{5.9}$$

- Wendland's functions

$$\begin{aligned} \phi(r) &= (1 - r)_+^4 (4r + 1) && (5.10)\\ \phi(r) &= (1 - r)_+^6 (35r^2 + 18r + 3) && (5.11)\\ \phi(r) &= (1 - r)_+^8 (32r^3 + 25r^2 + 8r + 1) && (5.12)\\ &\cdots && (5.13) \end{aligned}$$

Using the radial functions to expand the basis of the RBF interpolant, Eq: 5.1 can be expressed as

$$\tilde{f}(\mathbf{x}) = \sum_{j=1}^{M} \lambda_j \phi(\|\mathbf{x} - \mathbf{x}_i\|). \tag{5.14}$$

The coefficient vector $\boldsymbol{\lambda} = \{\lambda_j\}_{j=1}^{M}$ is determined by solving

$$\underbrace{\begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \phi\|x_1 - x_1\| & \phi\|x_1 - x_2\| & \dots & \phi\|x_1 - x_N\| \\ \phi\|x_2 - x_1\| & \phi\|x_2 - x_2\| & \dots & \phi\|x_2 - x_N\| \\ \vdots & \vdots & \ddots & \vdots \\ \phi\|x_N - x_1\| & \phi\|x_N - x_2\| & \dots & \phi\|x_N - x_N\| \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix}}_{\boldsymbol{\lambda}} \tag{5.15}$$

In the subsequent analyses, our choice of the radial basis function $\phi(\cdot)$ is the Wendland function given in Eq: 5.10, probably the most popular radial function presently in use. The choice of Wendland function was based on results of a number of in-sample experiments using dynamical systems deployed in this thesis (Lorenz63 and Lorenz84), but also using Franke's test function [42] which is often deployed to test interpolation models. Throughout the experiments, the Wendland function proved to be the most stable and achieved a very good approximation performance of all tested functions, which included Wendland, Gaussian, and power function. Due to its stability and good performance the Wendland function has also been recommended by a number of authors, [39, 40], which puts our findings in line with the current literature. We note that stability of solutions is of crucial importance. For example, the cubic function (one of the power functions tested) was often found unstable and often produced explosive forecasts.

5.1.2 Interpolation, approximation, and computational cost

In the previous section (5.1.1), we considered an interpolation problem, for which $M = N$, i.e. A is a square $(N \times N)$ matrix. With A being square, the size of the linear system of Eq: 5.17 quickly grows, and may become computationally infeasible. For moderate datasets this is not an issue. However, considering a large dataset containing, say 10^6 observations, the size of matrix A becomes $10^6 \times 10^6$. The inversion of such a matrix becomes prohibitive on standard computational devices.

For a limited computational resource, there is always a threshold at which the matrix inversion becomes infeasible. In Fig: 5.1 we show the cost of solving a linear system for different sizes of the square matrix A in terms of the CPU time on a personal computer using an Intel duo-core 2.4 GHz processor and 8 MB of memory with the linear algebra package [2]. We observe that the CPU time grows at a rate given by a power law. Already by the size of 10^5 it takes 35 seconds to calculate the solution. Although solving a single system of that size is not a major issue, when forecasting we often consider problems where the system must be solved repeatedly, possibly hundreds or even thousands of times.

To deal with a system when A is too large to invert, and/or to reduce the CPU time required by the solution, we may reformulate the interpolation problem as an approximation. Under RBF approximation, we let $M \ll N$, i.e. we select many fewer centers than there are datapoints, $\mathbf{c} \subseteq \mathbf{x}$. The matrix A then becomes

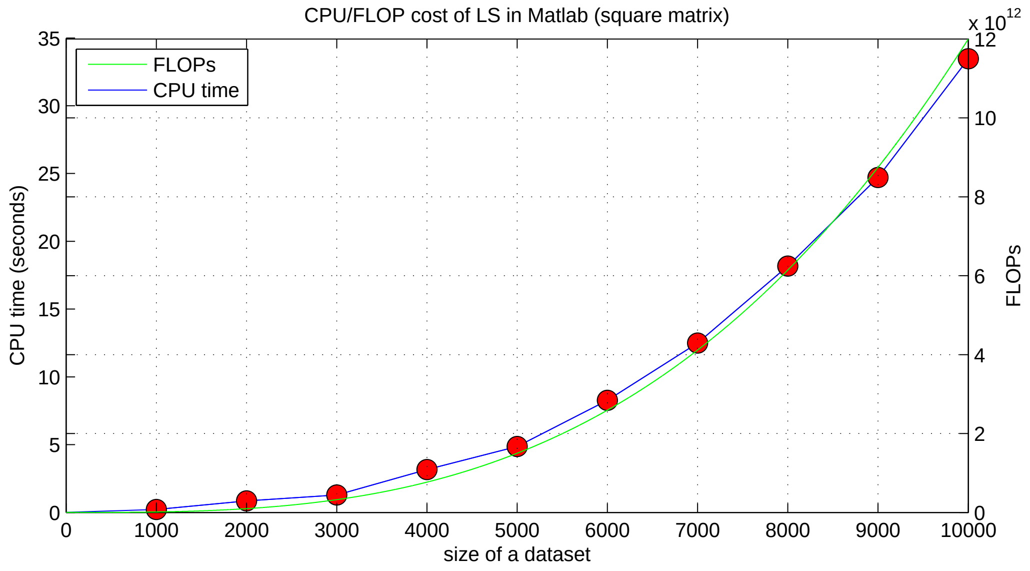

Figure 5.1: **Growth of computational intensity.** The computational cost of least square solution in terms of CPU (blue line) is plotted along with a theoretical computational cost, given by FLOPs, for different sizes of a $M \times M$ square matrix. The growth of the computational cost follows a power law, at $M = 10^4$ it takes 35 seconds to calculate the solution on a platform with Intel duo-core 2.4 GHz processor and 8 MB of memory. Increasing M may quickly render the solution intractable.

$$A = \begin{bmatrix} \phi\|x_1 - c_1\| & \phi\|x_1 - c_2\| & \dots & \phi\|x_1 - c_M\| \\ \phi\|x_2 - c_1\| & \phi\|x_2 - c_2\| & \dots & \phi\|x_2 - c_M\| \\ \vdots & \vdots & \ddots & \vdots \\ \phi\|x_N - c_1\| & \phi\|x_N - c_2\| & \dots & \phi\|x_N - c_M\| \end{bmatrix} \tag{5.16}$$

Setting $M \ll N$ yields a 'long' $N \times M$ matrix and the solution becomes tractable. The linear system is now overdetermined and the coefficients $\boldsymbol{\lambda}$ may be obtained via least squares.

Although, in large datasets, approximation makes the RBF problem solvable, it comes at a cost. Fewer centers mean that the surface of the approximated function f will be less well described. Hence there is a tradeoff between computational tractability of the problem and the quality of approximation. The tradeoff poses an important question: How many centers to use? Here we only note that the optimal number of centers is always problem-dependent. In our work, we experiment and find the optimal number of centers as a subjective balance between the speed of, and the quality of, the solutions.

The interpolation/approximation and related computational cost are always present when working with RBF. The discussion in this section is therefore applicable to any real-world application when the underlying model takes the form of RBF. The decision as to how many centers to use is implicitly a decision about whether to use interpolation or approximation, and lies right at the heart of any RBF application. Further discussion of the computational issues, as well as center selection, is provided in sections where RBFs are applied and also in Appendix D.

5.1.3 Training for forecasting

Although RBF is a general method, the original application is surface interpolation. Due to its generality, the approach has been extended to timeseries forecasting [109, 130]. In the analyses below, we use RBFs to forecast dynamical systems. The datasets involved are large, hence we apply the RBF approximation.

One way of formulating the RBF approximation as a forecasting problem is as follows. First, consider a d-dimensional dynamical system generating a time series $\mathbf{x} = \{\mathbf{x}_t\}_{t=1}^{N}$ of system states $\mathbf{x}_t = \{x_t^1, \ldots, x_t^d\}$. Assume that we observe, and are interested in forecasting, only a single dimension, say

x^1, giving a time series of observations $\mathbf{x}^1 = \{x^1_t\}_{t=1}^N$. We can embed the time series $\mathbf{x}^1$ in s dimensions [117], and use the embedded values to forecast future values x_{t+l}. In the training, the overdetermined linear system takes the form of

$$\underbrace{\begin{bmatrix} x^1_t \\ x^1_{t-1} \\ \vdots \\ x^1_{t-N+s+1} \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \phi\|\mathbf{x}^1_{t-1} - \mathbf{c}_1\| & \dots & \phi\|\mathbf{x}^1_{t-1} - \mathbf{c}_M\| \\ \phi\|\mathbf{x}^1_{t-2} - \mathbf{c}_1\| & \dots & \phi\|\mathbf{x}^1_{t-2} - \mathbf{c}_M\| \\ \vdots & \ddots & \vdots \\ \phi\|\mathbf{x}^1_{t-N+s} - \mathbf{c}_1\| & \dots & \phi\|\mathbf{x}^1_{t-N+s} - \mathbf{c}_M\| \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix}}_{\boldsymbol{\lambda}} \tag{5.17}$$

where x^1_t is a value of variable x^1 observed at time t, $\mathbf{x}^1_{t-1}$ is an embedded vector available at time $t-1$ and $\{\mathbf{c}_1, \dots, \mathbf{c}_M\} \subset$ are s-dimensional centers selected from the embedded vectors.

5.1.4 Generating an ensemble forecast

Using the example of the previous section, in the forecasting mode we use the current and past observations of $\mathbf{x}^1$ to create an embedded vector $\mathbf{x}_t$ and measure the M distances of $\mathbf{x}_t$ from the M centers $\mathbf{c}$. Using the given radial function ϕ and the parameter vector $\boldsymbol{\lambda}$, which was determined during the training, the leadtime 1 forecast of x^1_{t+1} is generated as

$$\begin{bmatrix} \hat{x}^1_{t+1} \end{bmatrix} = \begin{bmatrix} \phi\|\mathbf{x}^1_t - \mathbf{c}_1\| & \dots & \phi\|\mathbf{x}^1_t - \mathbf{c}_M\| \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_M \end{bmatrix}.$$

The ensemble forecast is obtained by repetitively applying the forecasting step to each initial condition.

5.2 Center selection

Working with fewer centers than datapoints requires some form of center selection method. There are a number of useful approaches that can be used. For all approaches, we decide how many centers will be selected prior to application of the selection method, noting that the number of centers K is much smaller then the number of datapoints $K << N$. To determine the best center selection method, we use $K = 128$; considering the typical sizes of the datasets used throughout the thesis (order of 10^6, see Appendix C.1) and our computational constraints, this is borderline feasible. In this work, we have tested the following center selection methods:

- simple attractor covering
- k-means
- knot insertion (greedy RMS minimization)
- adaptive knot insertion (adaptive greedy RMS minimization)
- power function (adaptive greedy minimization)

5.2.1 Simple attractor covering

Simple attractor covering aims to distribute centers along the system attractor in order to capture local behavior of the system. An ideal distribution of the centers should be able to well describe the system's behavior at any subspace of the space spanned by the system attractor. Simple attractor covering is based on a requirement that subsequent centers are separated by some minimum distance. This requirement, however, does not guarantee that all 'dynamically important' subspaces will be covered by a sufficient number of centers. If the minimum distance between centers is too large, some subspaces will not be well covered and the local dynamics of the system will not be well captured. On the other hand, setting the minimum distance too small may yield too many centers, which may become computationally prohibitive in subsequent modeling/forecasting. Despite some drawbacks, this approach can be useful in a number of cases, especially when local behavior is not confined to small subspaces or a significant computational power is available. Due to its simplicity, this approach is straightforward to use and may be applied to obtain a quick insight concerning system dynamics.

In this method we select K centers c_k, $\{k = 1, \ldots, K\}$, from the T points x_t, $\{t = 1, \ldots, T\}$, on a trajectory. We begin by setting a minimum length l between the centers such that $\|c_k - c_{k-1}\| \geq l$. We then initialize the algorithm by taking the first point on the trajectory $x_{t=0}$ to be the first center $c_{k=1}$. To obtain the second center $c_{k=2}$ we keep accumulating the distances of the subsequent datapoints $\|x_{t+1} - x_t\| + \|x_{t+2} - x_{t-1}\|$ until the cumulative distance exceeds the minimum required distance l. The last data point of the accumulation is then selected as the second center $c_{k=2}$. The procedure is iteratively applied until we run out of datapoints. The simple attractor covering is described by Algorithm 1.

5.2.2 K-means

We use the well-known method of cluster analysis, k-means [121], as another method of center selection. K-means is a two-step algorithm that minimizes

Algorithm 1 Attractor covering center selection

```
1.  $l = 0$ // *set initial length*
2.  **for** $i = 1$ to $N - 1$ **do**
3.    **for** $j = 0$ to $K - 1$ **do**
4.      $l = l + \|x_i - x_{i-1}\|$ // *accumulate distance of subsequent points*
5.
6.      **if** $l > l_{min}$ **then**
7.        **for** $k = 0$ to $K - 1$ **do**
8.          $d_k = \|x_i - c_k\|$ // *find distance to the closest center*
9.          $d = min(d_k)$
10.       **end for**
11.
12.       **if** $d < \epsilon$ **then**
13.         $c_j = x_i$ // *set the datapoint to be a center; reset length*
14.         $l = 0$
15.       **end if**
16.
17.     **end if**
18.   **end for**
19. **end for**
```

the sum of distances of points from a centroid summed over K clusters. To initialize the algorithm, an initial set of centers c_k, $k = 1, \ldots, K$ is selected (no matter how). The algorithm than proceeds to the first step (assignment step) in which each data point on a trajectory is assigned to one of the K clusters. The assignment is based on the distance between the data point and the centers, so that a data point is in the same cluster as its closest center. In the second step (update step) means of all K clusters are calculated using only points within the K-th cluster. The K means become new centers. In the next iteration, the datapoints are re-assigned based on distances from the new centers. The iterations proceed until the sum of the within-cluster distances accumulated across all clusters is minimized. The details of the procedure are given in Algorithm 2.

The drawback of K-means, when used to analyze dynamical systems, is that the centers do not necessarily coincide with data points, i.e. the centers may not lie on the system trajectory, and so can be located outside the system's attractor where the dynamics is not representative of the system. However, K-means is a well-established method shown to perform well across a diversity of settings, and certainly can be used as a good initial approach.

5.2.3 Knot insertion

The Knot insertion method is based on the reduction of a distance between forecast and verification. The algorithm starts by setting a random datapoint to be a center, the in-sample RBF forecast is produced using only this single datapoint as center. Next, the in-sample distances of one-step-ahead forecasts and verifications are calculated, and the datapoint at which the distance is largest is taken to be a new center. Subsequently a new RBF forecast is calculated, now using 2 centers. The procedure then follows in

Algorithm 2 K-means center selection

```
1.  // Requirement:
2.  $\arg\min_{\mathbf{S}} \sum_{i=1}^{K} \sum_{x_i \in S_i} \|x_k - \mu_i\|$
3.
4.  // Assignement Step:
5.  for $i = 1$ to $N - 1$ do
6.      $d_{min} = \text{Inf}$ // initialize distance
7.      for $k = 0$ to $K - 1$ do
8.
9.          $d_k = \|x_i - c_k\|$ // calculate distance of point i from center k
10.         if $d_j < d_{min}$ then
11.             $d_{min} = d_k$
12.             $indx = k$
13.         end if
14.
15.     end for
16.     $d = min(d)$
17.     $\mathbf{x}^k = \{\mathbf{x}^k, x_i\}$
18. end for
19.
20. // Update Step:
21. for $k = 0$ to $K - 1$ do
22.     $m_k = mean(\mathbf{x}^k)$ // calculate means of clusters
23.     $c_k = x_i^k$ closest to $m_k$
24. end for
```

a repetitive manner until either a target number of centers is achieved or some tolerance, measured by RMS, is reached. The algorithm is described in Algorithm 3.

Algorithm 3 Knot insertion

```
1. $c_1 = x_{U(1,N)}$
2. for $i = 1$ to $N - 1$ do
3.     $\tilde{\mathbf{f}}_{t+1} = \Phi(\mathbf{x}_t, \mathbf{c})$
4.     $indx = find(max(\tilde{\mathbf{f}}_{t+1} - \mathbf{v}_t))$
5.     $c_i = x_{indx}$
6. end for
```

5.2.4 Adaptive knot insertion

Adaptive knot insertion is an extension of the knot insertion method. The additional feature is that after K centers have been collected using knot insertion a leave-one-out crossvalidation is performed. Only those centers for which RMS error increases significantly when left out are kept, while others are excluded. After the crossvalidation $C < K$ centers are left and the knot insertion is invoked again to replace the $K - C$ excluded centers. The significancy is determined as some threshold, typically given by a percentage of RMS error before the crossvalidation. The algorithm stops after a required number of iterations is reached or the RMS error saturates. Although this method produces much better results in terms of in-sample RMS error, it is computationally intensive as it alternates between exclusion and addition of the centers. The algorithm is described in Algorithm. 4.

5.2.5 The power function

In the RBF interpolation, the power function has been used as a tool for local error estimation. It has been shown, e.g. [119, 159], that the RBF interpolation error is bounded

Algorithm 4 Adaptive knot insertion

```
1.  run knot insertion
2.  while $iteration < \#iterations$ do
3.      calculate total RMS ($rms_{total}$)
4.      for $k = 1$ to $K - 1$ do
5.          $\tilde{\mathbf{f}}_{t+1} = \Phi(\mathbf{x}_t, \mathbf{c}_{-i})$
6.          $rms_k = RMS(\tilde{\mathbf{f}}_{t+1} - \mathbf{v}_t)$
7.          if $(rms_k - rms_{total}) < \epsilon$ then
8.              $indx = \{indx, k\}$
9.          end if
10.     end for
11.     $\mathbf{c} = \mathbf{c}_{-indx}$ // exclude centers with small contribution to RMS
12.     run knot insertion to replace $K - C$ excluded centers
13. end while
```

$$|f(\mathbf{x}) - s_{f,X}(\mathbf{x})| \leq \kappa P_{\Theta,\mathcal{X}}(\mathbf{x}) \tag{5.18}$$

where $s_{f,X}(\mathbf{x})$ is an interpolant of $f(\mathbf{x})$, κ a positive constant and $P_{\Theta,\mathcal{X}}(\mathbf{x})$ is the power function. The power function is defined by the quadratic form

$$P_{\Theta,\mathcal{X}}(\mathbf{x})^2 = \Theta(\mathbf{x},\mathbf{x}) - 2\sum_{j=1}^{N} u_j \Theta(\mathbf{x},\mathbf{x}_j) + \sum_{i=1}^{N}\sum_{j=1}^{N} u_i u_j \Theta(\mathbf{x}_i,\mathbf{x}_j) \tag{5.19}$$

where Θ is positive definite kernel and $u_j(\mathbf{x}_k) = \delta_{j,k}$ are *cardinal functions*. Setting $\mathbf{u} = (-1, u_1(\mathbf{x}), \ldots, u_N(\mathbf{x}))$ yields a matrix representation

$$P_{\Theta,\mathcal{X}}(\mathbf{x})^2 = \mathbf{u} A_{\Theta,Y} \mathbf{u}^T \tag{5.20}$$

with $A_{\Theta,Y}$ being the interpolation matrix with entries $A_{i,j} = \theta(\mathbf{x}_i - \mathbf{x}_k)$.

It has been noted by [26] that good centers for RBF interpolation can be located by minimizing the power function. Since the power function is independent of the functional values f($\mathbf{x}$), only datapoints (not function values) are needed to locate the centers.

To minimize the power function, we deploy the adaptive greedy algorithm as described in Alg. 4. The only change is that the minimization of RMS is replaced by the minimization of the power function.

5.2.6 Which method to use?

While the center selection algorithms play an important role in obtaining a good forecast, their construction and evaluation of performance is beyond the scope of this thesis. Instead of studying suitability of center selection methods, we rely on the literature devoted to the topic. In particular, we follow [25, 26, 118] who have shown that centers selected by algorithms based on minimization of the power function are optimal. The optimality is a strong argument for us to select the Power function method described in Section 5.2.5 as our method of choice.

The line of arguments supporting the optimality of the method can be summarized as follows. It has been shown [119, 159] that estimates of the local error of an interpolant take the form of the power function defined in Eq. 5.19. Further, it has been shown that if $X \subseteq Y$ are alternative sets of centers, then the associated power functions satisfy

$$P_{\Theta,\mathcal{X}}(\mathbf{x})^2 \geq P_{\Theta,\mathcal{Y}}(\mathbf{x})^2 \quad x \in \Omega \tag{5.21}$$

where Ω is the state-space and $\{P_{\Theta,\mathcal{X}}(\cdot)^2, P_{\Theta,\mathcal{Y}}(\cdot)^2\}$ are power functions of the sets X and Y respectively. Given the properties of the power function, in particular the maximality property [159], the inequality of Eq. 5.21 holds everywhere in the space Ω. As such, the power function is able to determine which of two alternative sets of centers is superior. In other words, minimizing the power function is guaranteed to yield an optimal set of centers. Since this appealing property has not been shown for the other methods discussed above, minimization of power function is not only suitable, but also the preferred center selection method.

5.3 Conclusions

In this chapter, we have provided background on Radial Basis Functions (RBF), a statistical modeling technique that we heavily rely upon in Chapter 6 on model error correction. We remind the reader that the aim of this chapter is to merely provide background information. While implementation of the above methods in this thesis is new, the concepts and ideas described in this chapter are known and documented in the literature cited above. However, to our knowledge, the issue of computational costs (see Fig. 5.1 and Appendix D) and the trade-off between number of centers and computational tractability for large datasets is a practical observation that is rarely discussed, and perhaps has the quality of a new contribution.

However, our main focus was to describe the RBF interpolation problem, to discuss how RBF interpolation differs from RBF approximation, to show how RBF approximation can be used in forecasting and to describe the tradeoff between the quality of a model and the tractability of the problem. We have shown that RBF interpolation faces important computational constraints in large datasets. In Fig. 5.1, we have calculated CPU time as a function of number of centers and shown that a matrix inversion problem with 10^5 centers results in 35 seconds of CPU time (on an Intel duo-core 2.4 GHz processor with 8MB of memory). While 35 seconds may seem a reasonable cost, we note that in general forecasting exercises, the inversion will be executed many times. For example, if the inversion is to be executed 1,000 times, as is indeed the case in our applications, the CPU time required totals about 10 hours. This may be prohibitive in many cases.

We discussed the RBF approximation, an approach designed to overcome the tractability issue in large datasets. We have also discussed the fact that going from RBF interpolation to RBF approximation involves loss of model quality, which is then reflected in poorer description of the surface being approximated. Consequently, shifting away from RBF interpolation involves an important trade-off between quality of modeling and tractability of the exercise. We present more detailed practical observations and findings in Appendix D.

Finally, we have described several methods of center selection, a crucial part of RBF approximation. We have described simple attractor covering, a method designed to distribute centers by selecting centers as observations that are separated by some minimum distance. We have discussed the method of K-means and noted that the method may yield centers that do not coincide with observed data points. While this is not an issue in stochastic systems, it may be of concern when modeling dynamical systems, as the centers may lie outside the system's attractor, i.e. in subspaces which the system does not visit. We have also described simple and adaptive knot insertion, two methods which aim to reduce the approximation error by inserting centers to subspaces where the error is large. And lastly, we have described a version of knot insertion, which inserts centers based on minimizing Power function (see Eq. 5.19). Based on the recent findings in the literature, we have concluded that our method of choice for the applications presented below will be the Power function method.

Chapter 6

Forecast correction: Predictor Corrector and ΨΦ

Forecast correction is an important part of a forecasting framework. By definition, any forecasting model is wrong, i.e. all models suffer from built-in imperfections, an inevitable consequence of imperfect understanding of the forecasted system. In forecasting mode, the model imperfections often result in errors that are of a systematic nature. Given their nature, the systematic forecasting errors can be detected and, potentially, corrected. It is therefore natural to equip a forecasting system with a set of procedures (post-processing methods) designed to detect and correct systematic errors in its forecasts. Forecast correction, along with forecast evaluation and model benchmarking (see Chapters 4 and 3), forms an important set of methods designed to improve, assess, and interpret the value of a forecasting model. While benchmarking and evaluation has been discussed above, in this chapter we focus on forecast correction.

We construct the predictor-corrector model (PC), a two-stage novel approach designed to iteratively correct a systematic part of the model error [61, 67, 137] resulting from imperfections of a core (first stage) model. We show that *PC* greatly improves the forecasts of a given core model and significantly exceeds the forecasting performance of ΨΦ [62, 63], an alternative two-stage approach based on direct corrections of the model error.

To both demonstrate and contrast the forecasting skill of *PC* and ΨΦ, we consider two dynamic systems, Lorenz84 [83] and Lorenz63 [81] (see Appendix A) with observational noise. Using the Lorenz84 system, we show that for a core model with low complexity, *PC* outperforms ΨΦ by up to 1 Bit even at long leadtimes, and maintains its superiority, although to a lesser extent (0.5 bits), at medium leadtimes even under an improved, i.e. more complex, core model.

The core model complexity is expected to influence the forecasting quality of both approaches [134]. Intuitively, the better the core model, the lower the model error, hence less space for a corrective action. To study the impact of complexity, we test both *PC* and ΨΦ under varying complexity of the core model. We show that *PC*, unlike ΨΦ, delivers a robust performance.

We also study the behavior of the two methods while 'starving' them of data, i.e. using shorter dataset sizes. Again, *PC* delivers a robust performance while the performance of ΨΦ is degraded.

In addition, we use numerical forecasts of *PC*, ΨΦ and a given core model to demonstrate that evaluation based on the root mean square error (RMS) [97] can be misleading, even dangerous, as the RMS criterion may select an inferior model as the best performer.

Finally, we pay special attention to the computational intensity of both methods. Since our versions of *PC* and ΨΦ extensively utilize the radial basis functions described in Chapter 5, the computational issues are related to a solution of an overdetermined linear system. We discuss the issues in Appendix D.

This chapter is structured as follows. We first provide some insight into our version of a core model in Section 6.1. A detailed description of the core model, along with a description of the datasets, is given in Appendix C. The $\Psi\Phi$ method is described in Section 6.2 and the description of the PC method follows in Section 6.3. The numerical results of the two approaches are analysed and contrasted in Section 6.4.

In this chapter, the following are new contributions:

- We construct PC, a two-stage iterative approach.
- PC delivers significant improvement of core model forecasts via model error correction.
- PC performs robustly under varying complexity of a core model.
- Superior performance of PC when contrasted with $\Psi\Phi$.
- We demonstrate that RMS-based evaluation may be misleading.

6.1 Constructing the core model Φ

Both methods, the $\Psi\Phi$ and PC, are designed to correct an output of a common, core forecasting model. Neither of the methods is concerned with how the core model is constructed; it is assumed to be a given. In weather forecasting, the core model is typically a large Global Circulation models [92], such as the ENSEMBLES models [57] described in Section 2.5.3. Since in our analyses a core model is not exogenously given, we must construct it.

In this section we provide an intuition about the core model construction; a detailed description is given in Appendix C.4.

6.1.1 Systematic model error

Consider a time series of observed system states $x_1, \ldots, x_t, \ldots, x_N$ and an imperfect forecasting model Φ initialized at time t with an observation x_t producing a leadtime 1 forecast

$$\begin{aligned} z_{t+1} &= \Phi(x_t) && (6.1) \\ &= x_{t+1} + \varepsilon_{t+1}(x_t, \epsilon) && (6.2) \end{aligned}$$

where ε_{t+1} is a forecasting error, which is a function of the current observation, x_t, and some stochastic part, $\epsilon \sim F(\cdot)$ distributed according to some distribution $F(\cdot)$. In the simplest case, the forecasting error ε may be assumed additive with the stochastic part being i.i.d., $\epsilon \overset{i.i.d.}{\sim} F(\cdot)$, so that we can decompose the forecasting error as

$$\varepsilon_{t+1} = \epsilon^M_{t+1}(x_t) + \epsilon. \qquad (6.3)$$

where ϵ^M_{t+1} represents a systematic part of the forecasting error, i.e. the model error at point x_t at time t.

We aim to construct a model Ψ attempting to capture the model error ϵ^M_{t+1}, i.e. we require a correcting model Ψ to produce an estimate of the model error at time $t+1$ given a current observation x_t

$$\begin{aligned} \hat{\epsilon}^M_{t+1} &= \Psi(x_t) && (6.4)\\ &= \epsilon^M_{t+1}(x_t) + \eta && (6.5) \end{aligned}$$

where η is an error of the model Ψ. Assuming that Ψ perfectly captures the model error so that $\eta = 0$, we can reduce the forecasting error by subtracting the model error estimate from the Φ model forecast. The final forecast can then be written as

$$\begin{aligned} \hat{x}_{t+1} &= \Phi(x_t) - \Psi(x_t) && (6.6)\\ &= z_{t+1} - \hat{\epsilon}^M_{t+1} && (6.7)\\ &= x_{t+1} + \varepsilon_{t+1}(x_t, \epsilon) - \hat{\epsilon}^M_{t+1} && (6.8)\\ &= x_{t+1} + \epsilon. && (6.9) \end{aligned}$$

Comparing the Φ model forecast z_{t+1} of Eq: 6.2 with the $\Psi\Phi$ forecast $\hat{x}_{t+1}$ of Eq: 6.9, we see that the $\Psi\Phi$ forecast has the smaller error. The point of constructing the Ψ model is to improve the forecast of x_{t+1} by reducing the model error of the Φ model.

6.1.2 Generating core model ensemble forecasts

To produce a core model Φ ensemble consisting of K ensemble members, we require an ensemble of initial conditions $\mathbf{x}_t = \{x_{t,1}, \ldots, x_{t,K}\}$. One way to generate the initial condition ensemble is to perturb an observed state x_t by sampling from an inverse of the assumed observational noise model. In the case of additive noise the observed state is given as

$$x_t = \tilde{x}_t + e \tag{6.10}$$

where $\tilde{x}_t$ is the 'true' system state and $e \overset{iid}{\sim} F(\cdot)$ is the observational noise term given by some noise model $F(\cdot)$. In the subsequent analyses, we assume the noise model to be Gaussian, $e \sim N(0, \sigma)$ and generate the initial conditions as

$$x_{t,k} = x_t + e. \tag{6.11}$$

The leadtime L ensemble forecast $\mathbf{x}_{t+L}$ for a given initialization time t may be obtained using an *iterative approach*, i.e. by iterating the initial condition ensemble $\mathbf{x}_t$ with the core model L times

$$\begin{aligned}
\hat{\mathbf{x}}_{t+1} &= \Phi(\mathbf{x}_t) && (6.12)\\
\hat{\mathbf{x}}_{t+2} &= \Phi(\hat{\mathbf{x}}_{t+1}) && (6.13)\\
&\vdots \\
\hat{\mathbf{x}}_{t+l} &= \Phi(\hat{\mathbf{x}}_{t+l-1}) && (6.14)\\
&\vdots \\
\hat{\mathbf{x}}_{t+L} &= \Phi(\hat{\mathbf{x}}_{t+L-1}) && (6.15)
\end{aligned}$$

where $\hat{\mathbf{x}}_{t+l}$ is an ensemble forecast for a leadtime l, and L is the final or maximum leadtime of the forecast.

An alternative way of producing the leadtime L forecast is to use a *direct approach*, i.e. mapping initial conditions $\mathbf{x}_t$ directly to $\hat{\mathbf{x}}_{t+L}$ as

$$\hat{\mathbf{x}}_{t+L} = \Phi_L(\mathbf{x}_t). \tag{6.16}$$

The parameters of the functions Φ and Φ_L are different, hence the leadtime L forecasts of the direct and iterative approach also differ. In our analyses, we apply the iterative approach.

6.1.3 Using the core model

In the subsequent analyses, the core model Φ is used to:

a) Provide an uncorrected out-of-sample ensemble forecast of a system. An ensemble is generated by perturbing some initial state and integrating the initial conditions ensemble using Φ, see Section 6.1.2.

b) Provide forecasting errors used when fitting the $\Psi\Phi$ corrector. For this purpose, Φ is initialized at an actually observed, unperturbed state. The Φ forecasts are compared with out-of-sample verifications and the forecasting errors are collected separately for each leadtime.

c) Provide forecasting errors to be used by PC. In this case the procedure under b) is repeated, but only leadtime 1 forecasting errors are collected.

Under items b) and c), we generate a non-overlapping forecast. Considering a size N of the evaluation dataset (see Appendix C.1), we can generate up to $\lfloor N/L \rfloor$ of non-overlapping forecasts, the $\lfloor \cdot \rfloor$ is the floor function. For example, for $N = 101$ and $L = 10$ we can generate 10 non-overlapping forecasts launched at every 10-th point. This fact will have an impact on the number of data available to $\Psi\Phi$ or PC, and we will comment on it when describing the two methods.

6.2 The $\Psi\Phi$ method

$\Psi\Phi$ [63] is a two-stage approach designed to improve forecasts of an iterative predictor by exploiting information inherent in the systematic part of a model error. In the first stage, a core model Φ is deployed to produce a base forecast. In the second stage a correcting model, Ψ, corrects the systematic parts of the model error. As a result, the corrected $\Psi\Phi$ forecast is expected to outperform the uncorrected Φ forecast whenever there is an imperfect model and a detectable systematic error.

In Section 6.1, we have described how to construct the Φ model and use it to generate forecasts. Here we focus on correcting the Φ forecasts. In our analyses we use two versions of the model - one based on least squares, $\Psi\Phi_{\mathrm{LSQ}}$, and one based on radial basis functions, $\Psi\Phi_{\mathrm{RBF}}$. In this section we provide some insight into construction of the corrector Ψ; the details are provided in Appendix: C.5.1. We stress that the two-stage model $\Psi\Phi$ used in this work is a version of a modeling approach published in [63], to which the reader is referred for a more detailed description.

While our construction of $\Psi\Phi$ is unique, the new material in this chapter is the *PC* method defined in Section 6.3.

6.2.1 The core model forecast and forecasting errors

The out-of-sample forecast is obtained by initializing Φ with initial conditions obtained from the learning set. To collect the forecasting errors, only a point forecast - not an ensemble forecast - is generated. The forecasting errors are obtained by comparing the Φ forecasts to the verifications contained within the learning set.

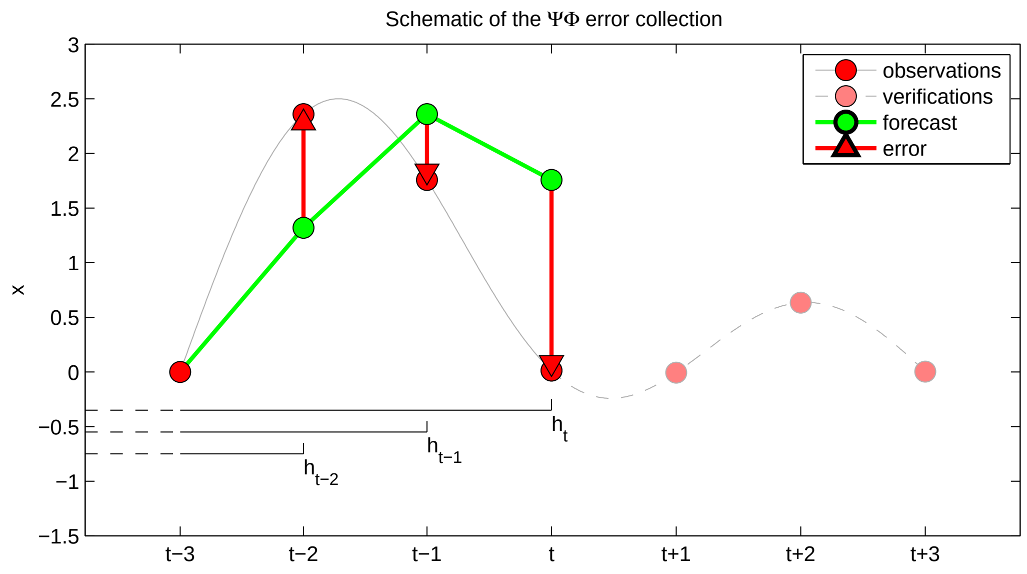

Figure 6.1: **ΨΦ collecting the forecasting errors:** Within a learning set (gray solid curve), a core model is initialized with a single initial condition (red) and produces forecasts (green dots) up to leadtime $l = t$. The forecast is contrasted with verification (red) and the forecasting error (red arrow) is collected. The light red dots represent verifications of the testing set (gray dash curve). The symbols h_t, h_{t-1} and h_{t-2} represent information available at times t, $t-1$ and $t-2$.

In Fig: 6.1, we show a schematic of the process. Assume we aim to generate forecasts for a maximum leadtime of $L = 3$. The trained Φ model is initialized at time $t-3$ with a single initial condition. The Φ model then generates a forecast for each of the 3 leadtimes (green dots), which are compared to the verifications (red dots) within the learning set (gray solid line). The 3 forecasting errors (red arrows) may then be collected and used in the correcting stage.

6.2.2 Fitting the corrector, quality of the error surface

Collecting forecasting errors yields, for each leadtime, as many forecasting errors as there are forecasts. Assuming we forecast 5 leadtimes and have 10 forecasts available, we end up with 10 forecasting errors at leadtime 1, 10 forecasting errors at leadtime 2 and so on up to leadtime 5.

In our version of the $\Psi\Phi$ approach, the parameters of Ψ are determined for each leadtime separately, which means that for each leadtime we have a different Ψ_l. Consequently, each Ψ_l depends only on the forecasting errors of the given leadtime l. For non-overlapping forecasts, the degrees of freedom available to fit the parameters of Ψ_l is therefore constrained by the the maximum leadtime L. If $L = 1$ we can generate as many non-overlapping forecasts as there are datapoints (minus 1). If $L = 10$ we have 10 times less forecasts, hence 10 times less forecasting errors available for each Ψ_l.

Although this might seem to be a bad news, there is one factor that in the case of $\Psi\Phi_{RBF}$ somewhat compensates for the lower amount of errors available. In Section 5.1.2 we have discussed that for large datasets we must deploy a RBF approximation, which may yield a poor description of the surface to be fitted. While the amount of available data is constrained due to this we may actually be able to afford to use more centers. With more centers, we can obtain a better description of the surface to be fitted, which, in our case, is the surface given by the Φ model forecasting errors. In the analyses below we can afford to use all the datapoints as centers, which means that we can deploy RBF interpolation instead of RBF approximation.

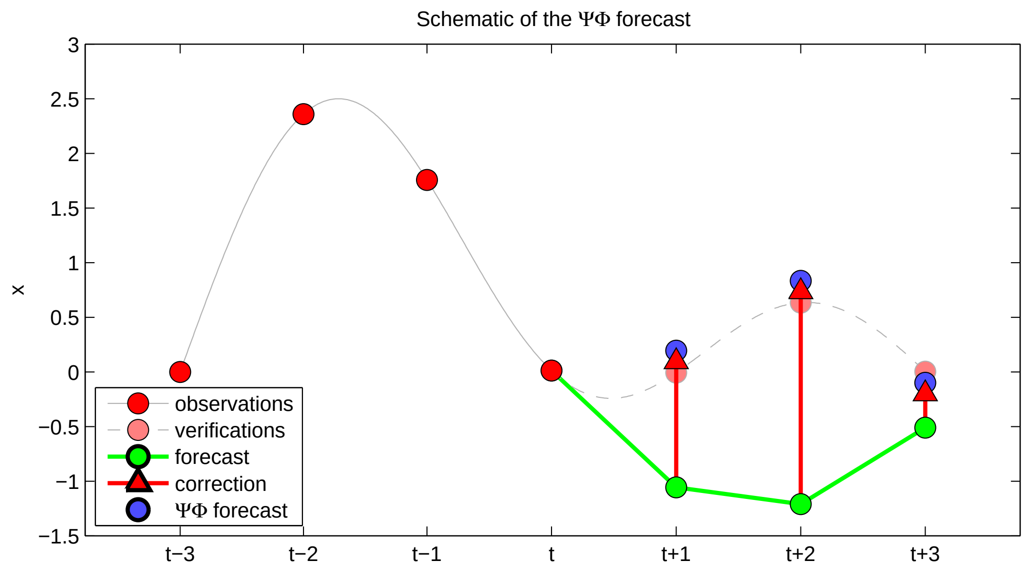

Figure 6.2: ΨΦ **forecast:** Initialized with an observation at time t (red point), the core model Φ produces leadtime 3 forecast (green). The Ψ corrector produces corrections (blue) which are added to the forecasts to obtain final forecast (blue) of the yet unobserved states (light red points).

6.2.3 ΨΦ in forecasting mode

In the forecasting mode, we first use Φ to simulate an l-step-ahead ensemble of size K. The initial conditions are obtained by K times perturbing an observation of a testing dataset at time t. At a given leadtime l, we apply the Ψ_l model, and estimate the systematic error of Φ for all the ensemble members. With the model error estimate in hand, we add it to the Φ forecast to obtain a ΨΦ forecast at leadtime l.

In Fig. 6.2, we show a schematic of the forecasting mode. The green line represents a single ensemble member produced by Φ initialized at time t, forecasting 3-steps ahead. The red arrows represent the estimate of the magnitude and direction of the systematic error and the blue points show the Φ forecasts corrected by the Ψ model.

The general ΨΦ procedure is as follows. The Ψ model uses the residuals from the first-stage fit of the core model Φ (red arrows in Fig. 6.2) to learn the structure of the systematic errors. The fitted Ψ is then used to generate the out-of-sample corrections, red arrows in Fig. 6.2, to obtain corrected forecasts of the core model Φ (blue dots). The corrected forecasts do not exactly hit the target (red dots in Fig. 6.2) but corrected forecasts are closer to the target than the first stage forecasts of the Φ model. Note that Ψ is applied after all the Φ forecasts in the training set have been generated. This is a crucial distinction from the PC method described in the next section.

6.3 The Predictor Corrector method

We suggest a novel predictor-corrector (PC) model, which, similarly to ΨΦ, generates forecasts in two stages; first, the core model P produces non-corrected forecasts and than a corrector C is applied to improve them.

PC is an iterative method that alternates between the prediction and the correction step. In the P step, some core model P produces a leadtime 1 forecast. In the C step a corrector is applied. The C corrected forecast then serves as an input into the next P step.

PC substantially differs from ΨΦ in that the ΨΦ corrector does not work iteratively. A typical sequence for PC is:

forecast - correct - forecast - correct

A typical sequence for ΨΦ is:

forecast - forecast - correct - correct

Note that in PC, the core model P will be structurally the same as the core model Φ within $\Psi\Phi$, i.e. in our version of PC the core model takes the same form of RBF approximation. Having $P = \Phi$ gives us the option of comparing both approaches. From now on when the notation P is used it is understood to be the $P = \Phi$.

6.3.1 Constructing the C-corrector

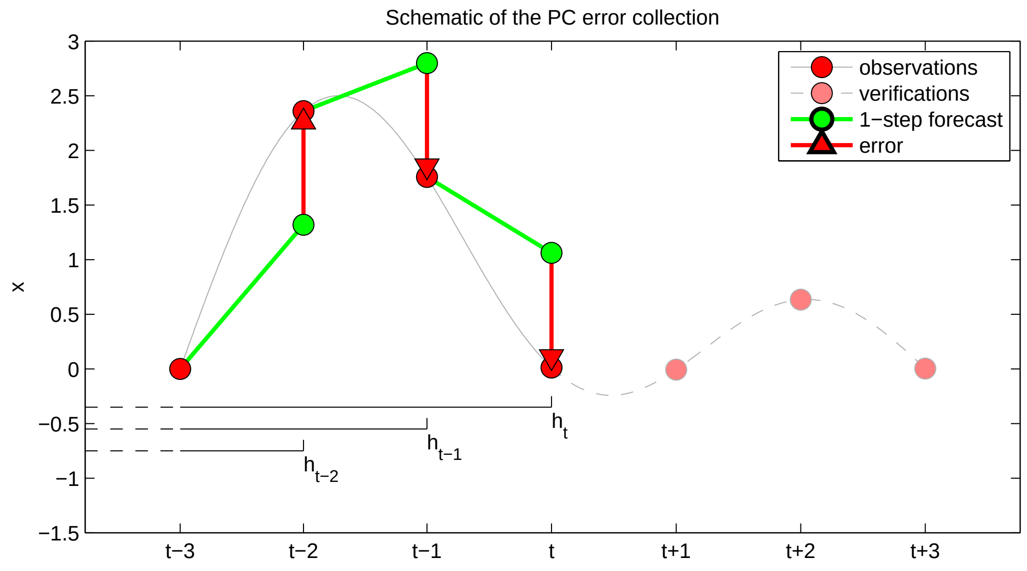

Figure 6.3: **Error collection in PC:** A core model P produces leadtime 1 forecasts (green), which are then contrasted with the verifications (red dots) to obtain forecasting errors (red arrows) used as input to the C corrector. As in $\Psi\Phi$ errors are collected within a learning set (gray solid curve).

The corrector C is designed to correct leadtime 1 forecasting errors of the core model P. As before, the core model is trained in-sample using a training set. Similar to the Ψ we use the RBF approximation as the corrector C. To construct C for a forecast initialized at time t forecasting L leadtimes ahead we require a vector of leadtime 1 forecasting errors $\varepsilon = \{\varepsilon_{t+1}, \varepsilon_{t+2}, \dots, \varepsilon_N\}$. The forecasting errors are given as

$$\begin{aligned} \varepsilon_{t+1} &= x_{t+1} - P(x_t) && (6.17) \\ \varepsilon_{t+2} &= x_{t+2} - P(x_{t+1}) && (6.18) \\ &\vdots \\ \varepsilon_N &= x_N - P(x_{N-1}) && (6.19) \end{aligned}$$

Consider the hypothetical example where the learning set has size $N = 101$ observations and the maximum leadtime is set to $L = 10$. Since the core model P only produces leadtime 1 forecasts it can be launched $N - 1 = 100$ times, giving 100 leadtime 1 non-overlapping forecasts and 100 forecasting errors. For datasets with large N we face the issue of tractability of A; in such cases the RBF approximation must be deployed.

Fig: 6.3 shows a schema of the forecasting error collection. In the learning set (gray solid curve), the core model is initialized 3 times to produce 3 leadtime 1 forecasts (green dots). Forecasts are then compared to the verifications (red dots) and the leadtime 1 forecasting errors (red arrows) are collected.

The parameters of the corrector C are determined using Radial Basis Functions described in Chapter 5. In particular, the parameters are obtained by solving the system in Eq: C.3 (Appendix C.5), where $\mathbf{y}$ represents the residuals, i.e. model errors, resulting from fitting the first stage core model P. Recall, that the first stage model errors are obtained by subtracting the in-sample leadtime 1 forecasts (or re-forecasts) from a target. The entries of A are given as values of some radial basis function ϕ at a given distance. The distance is an Euclidean distance of a predictor x_t from an RBF center. The predictors x_t are obtained by embedding the time series of the in-sample observations of the system states (see Appendix C.3). The centers are selected from the predictors using the Power function method of Section 5.2.5. Solving the system then yields the parameters of the corrector C; see Chapter 5 for more details on RBF fitting.

6.3.2 Selecting center for the C-corrector

As mentioned in the previous section, for large datasets C will be taking the form of RBF approximation. In such cases an important issue is how to select the centers, and how many centers should we use?

The question of how many centers to use may be resolved rather simply. We aim to use as many centers as possible, while keeping the size of the matrix A tractable. Given our computational resource, we set the number of centers to $M = \min(N-1, 4,096)$, so that if $N - 1 < 4,096$ we use all available residuals. The default maximum number of 4,096 centers is set experimentally, as in our applications 4,096 centers provide reasonably good forecasting results while keeping the linear system computationally tractable.

Since $M = 4,096 \ll N$, we must deploy a center selection method. The center selection methods of Section 5.2 work well for the core model, but for the corrector a faster method is required. We suggest a method based on inverse transform sampling [27, 136]

$$X_s = F_X^{-1}(Y) \tag{6.20}$$

where F_X is cumulative distribution function of random variable X, F_X^{-1} its inverse, $Y \sim U(0,1)$ and X_s is the sub-sample obtained via inverse sampling. We aim to sub-sample a set of forecasting errors x_{t-i} to obtain a subsampled

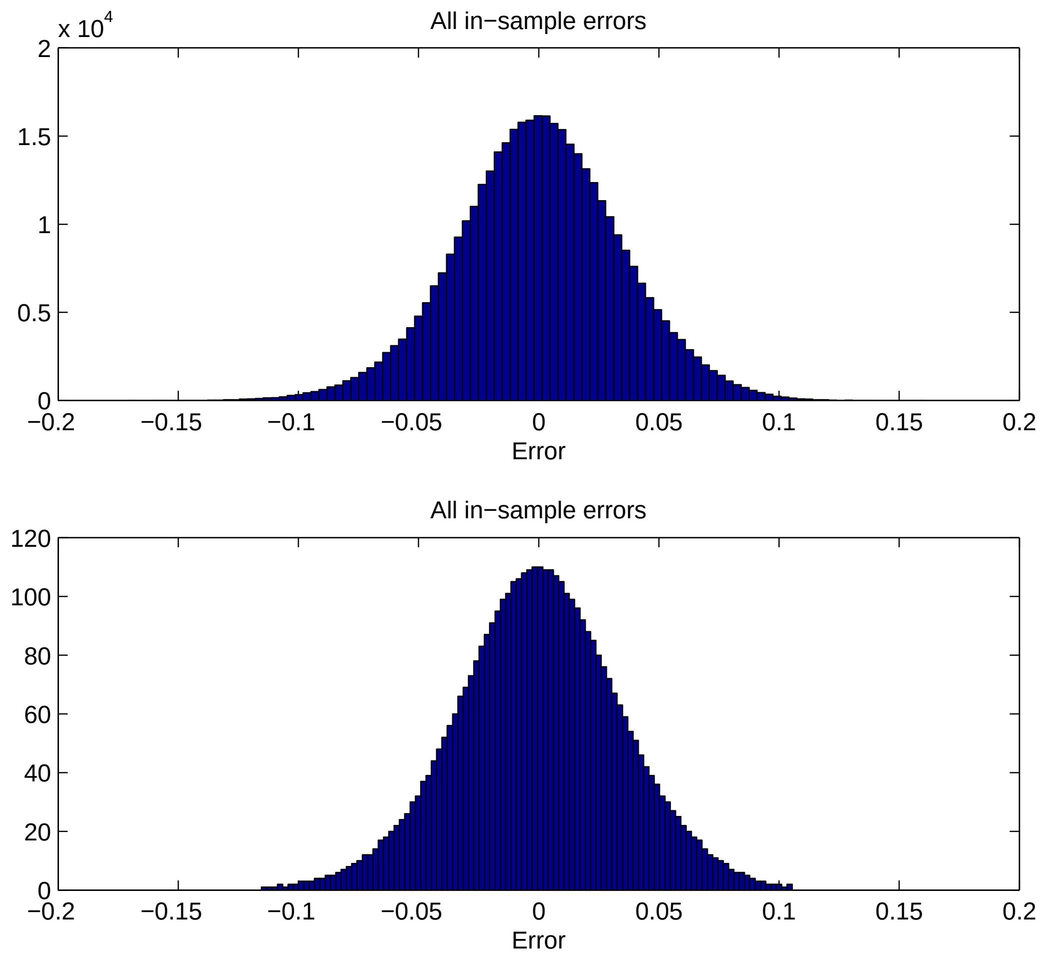

Figure 6.4: **Error histograms:** The histogram of about 400,000 collected forecasting errors (top) is very similar to the histogram based on the 4,096 subsampled errors (bottom). The inverse sampling preserves the shape of the histogram.

set of M errors x_{t-s_k}. To do this we first use a kernel density estimator [129] to obtain an empirical c.d.f. F_X of the model errors. Then we generate a random sample Y of size $M = 2^{12}$ from the uniform distribution $U(0, 1)$ and use it in the inverse c.d.f to obtain the M (subsampled) errors x_{t-s_k}.

In the top panel of Fig: 6.4, we show the distribution of the full set of forecast-

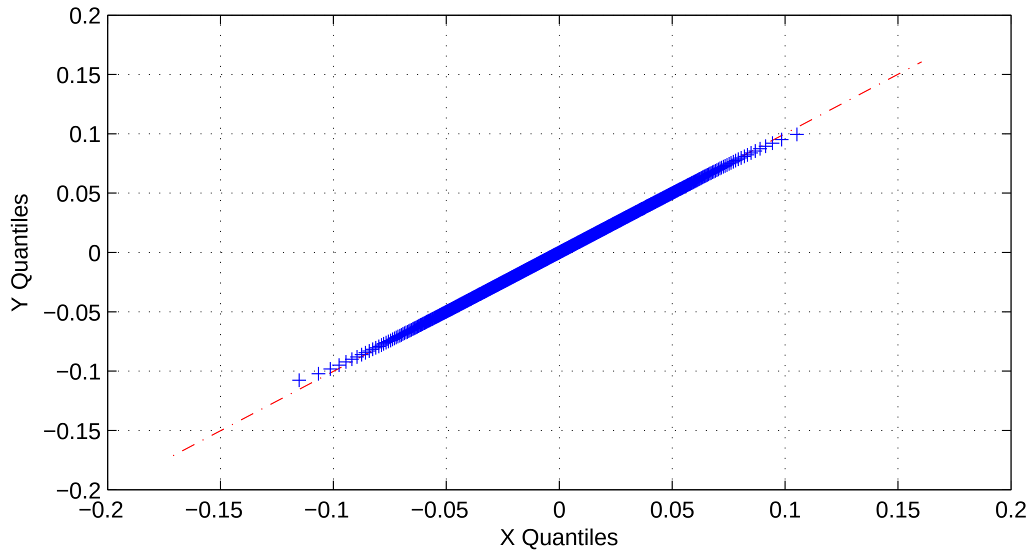

Figure 6.5: **Subsampled v. all errors:** In another evidence of the inverse sampling preserving the distribution of the forecasting errors, the qq-plot is concentrated around the 45 degree line, the tails are also close to the line.

ing errors (about 4×10^5 values) obtained by launching 1-step ahead forecasts of the core model at about 4×10^5 different datapoints of the Lorenz 84 in-sample dataset. The histogram of the subsampled set of forecasting errors ($M = 2^{12} = 4,096$ values) is displayed in the lower panel. In Fig 6.5 we also show the *qq-plot* of the full and subsampled sets. The qq-plot suggests that the distribution of forecasting errors in the subsampled set is well preserved even in the tails.

6.3.3 Forecasting mode

Given a core model P, a corrector model C and an embedded time series of historical observations $h_t = \{x_1, x_2, \ldots, x_t\}$, a leadtime 1 forecast, $\hat{x}_{t+1}$, of the PC model is obtained as

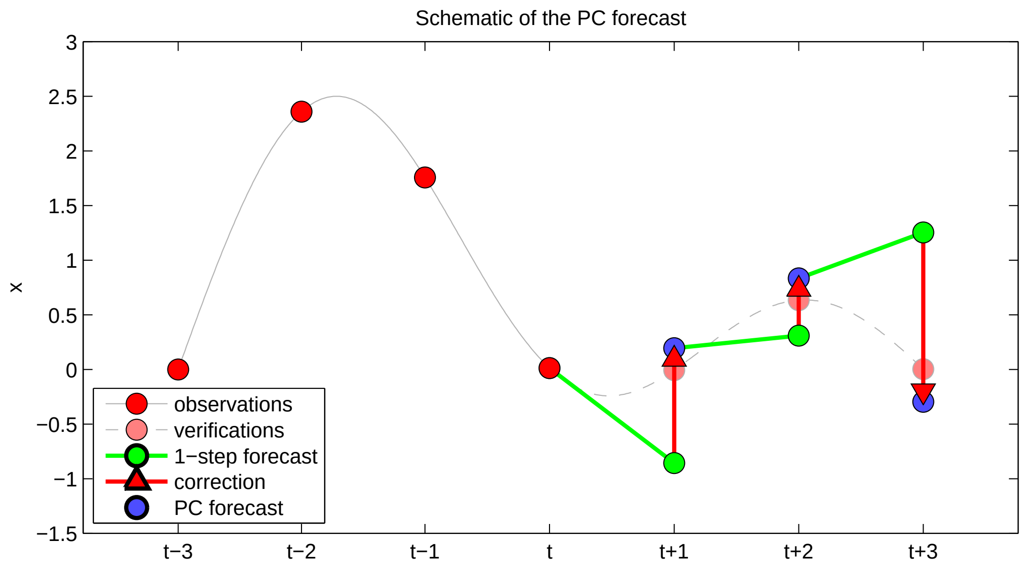

Figure 6.6: *PC* **forecast:** Core model P is initialized at time t (red point) of the testing set (gray dash curve) and produces leadtime 1 forecast of $t+1$ (green line), which is then corrected (red arrow) to produce *PC* forecast (blue dot). The core model is then initialized with the *PC* forecast at $t+1$ to produce leadtime 1 forecast of time $t+2$, which is then again corrected. The procedure is repeated as many times as required to obtain a leadtime L forecast. In this schematic $L = 3$.

$$\begin{aligned} z_{t+1} &= P^{(1)}(x_t) && \text{prediction step} && (6.21)\\ \hat{x}_{t+1} &= C^{(1)}(x_t) + P^{(1)}(x_t) && && (6.22)\\ &= C^{(1)}(x_t) + z_{t+1} && \text{correction step} && (6.23) \end{aligned}$$

where the superscript emphasizes that a model is iterated only once. Below we drop the superscript and assume that $P = P^{(1)}$ and $C = C^{(1)}$. The next iteration yields a leadtime 2 forecast

$$\begin{aligned}\hat{x}_{t+2} &= C(\hat{x}_{t+1}) + P(\hat{x}_{t+1}) \\ &= C[C(x_t) + P(x_t)] + P[C(x_t) + P(x_t)]\end{aligned}$$

Since P and C are nonlinear functions, the [] brackets cannot be expanded.

The iterative nature of the PC forecasts is demonstrated in Fig: (6.6). Initialized at the most recent observation (red dot) of the training set (gray dash line), the core model uses the embedded state to generate a leadtime 1 forecast $\hat{x}_{t+1}$ (green dot). The corrector C is then applied to $\hat{x}_t$ to obtain a forecast correction (red arrow). Adding the correction to the P forecast yields the final PC forecast $\hat{x}_{t+1}$ (blue dot). The leadtime 1 forecast $\hat{x}_{t+1}$ is then used as an input for the second iteration of the PC model. The process is repeated L times, yielding the leadtime L forecast.

The iterative nature of PC means that PC aims to correct the trajectory of the system, while $\Psi\Phi$, due to its direct approach, attempts to reduce a long-term bias Assuming both methods work well, we expect PC to outperform $\Psi\Phi$ as it concentrates on the trajectory correction rather then bias reduction.

For some systems, we expect that evaluation in terms of the root mean square error (RMS) will not fully appreciate the additional performance of PC. For example, the Lorenz63 [81] system is known to produce bimodal behaviour, as the system's attractor occupies two distinct subspaces (resembling butterfly wings), and the system tends to switch between them. As a result, the x-variable of the system displays bimodality, see Fig. 6.8. Since the $\Psi\Phi$ tends to predict long term behavior, it is expected to forecast the mean of the variable x, see [63]. But the mean forecast is completely wrong; the system never settles in the mean, it is either on one wing or the other, not between them. The PC on the other hand is expected to capture the bimodality of the x variable. If PC happens to predict bimodality but is not correct as to which wing the Lorenz system has settled on, the RMS will punish it for being too far from the target (located on the other wing). The $\Psi\Phi$ approach will be rewarded by RMS for forecasting the mean, the mean is always half-way, and so despite its forecasts being useless, $\Psi\Phi$ may be rated higher than PC by RMS. This will be discussed in more detail below.

6.4 Forecasting Lorenz84/63

In the following forecasting exercises, we deploy the perfect model scenario (PMS) as a benchmark model. In total we deploy 5 different forecasting models for both systems:

(1) Perfect model (PMS),

(2) the non-corrected core model $\Phi = P$,

(3) predictor-corrector model PC,

(4) the $\Psi\Phi$ model based on least squares ($\Psi\Phi_{LSQ}$) and

(5) the $\Psi\Phi$ model based on radial basis functions ($\Psi\Phi_{RBF}$).

In Fig: 6.7 we present forecasts of the Lorenz84 system over a selected time period. In the selected time period the non-corrected core model (green) loses forecasting skill after about 15 hours. The PC method (blue) improves the forecast significantly; several ensemble members stay close to the verification (red line) for the whole time period of 100 hours. The $\Psi\Phi_{RBF}$ (magenta) also improves the core model forecast, although not as dramatically as the PC model. The correction of the $\Psi\Phi_{LSQ}$ model (orange) has very limited impact; the forecasts quickly converge to the mean.

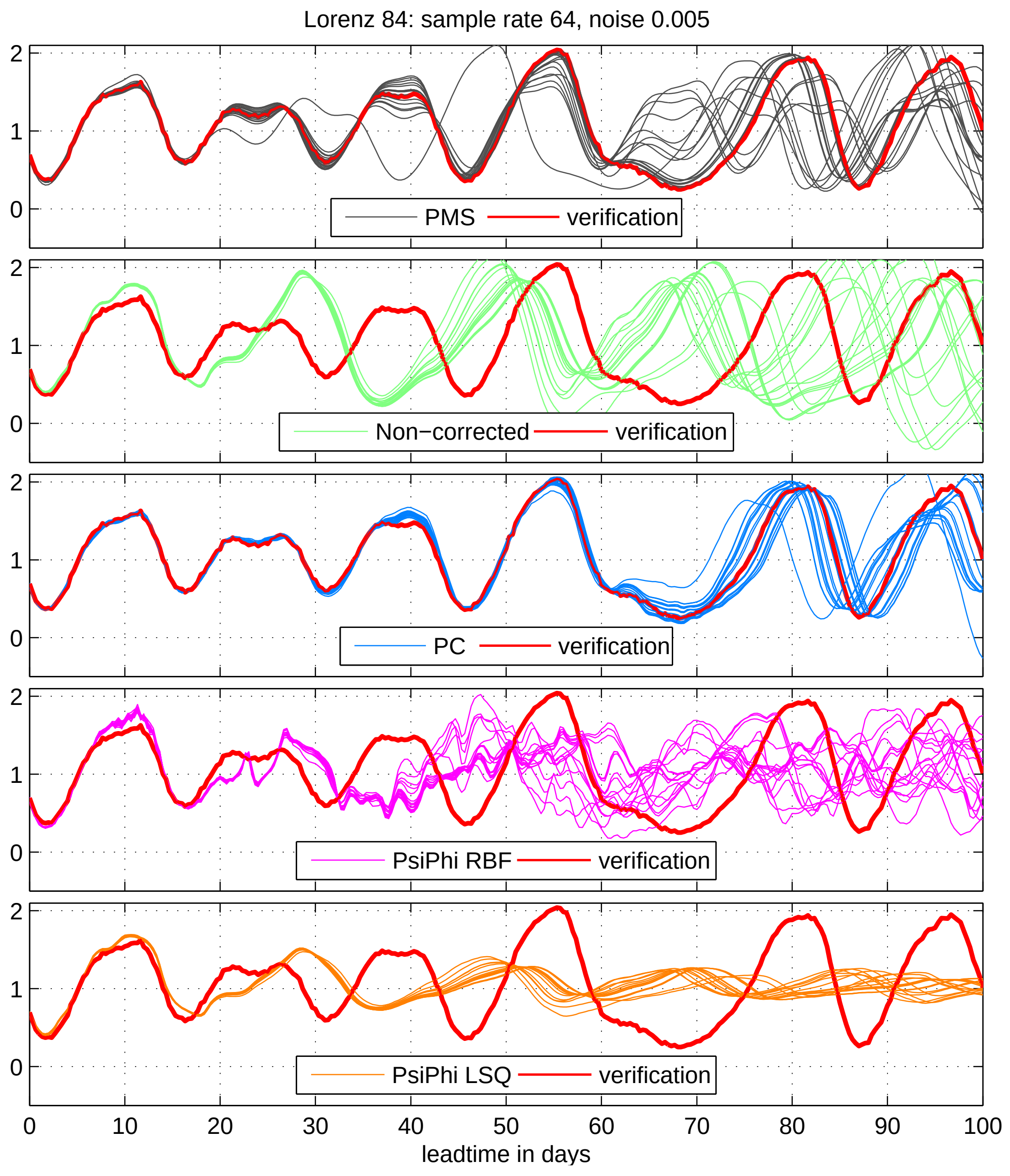

Figure 6.7: **Lorenz84 forecasts:** From top to bottom, PMS, non-corrected core model Φ, PC, $\Psi\Phi_{RBF}$ and $\Psi\Phi_{LSQ}$ forecasts of a selected 100-hour-long segment of the Lorenz84 system. *PC* outperforms the other 4 (imperfect) models. The $\Psi\Phi_{LSQ}$ quickly resorts to forecasting the mean of x, the non-corrected Φ model loses skill after 20 hours. The $\Psi\Phi_{RBF}$ marginally improves in this particular forecast.

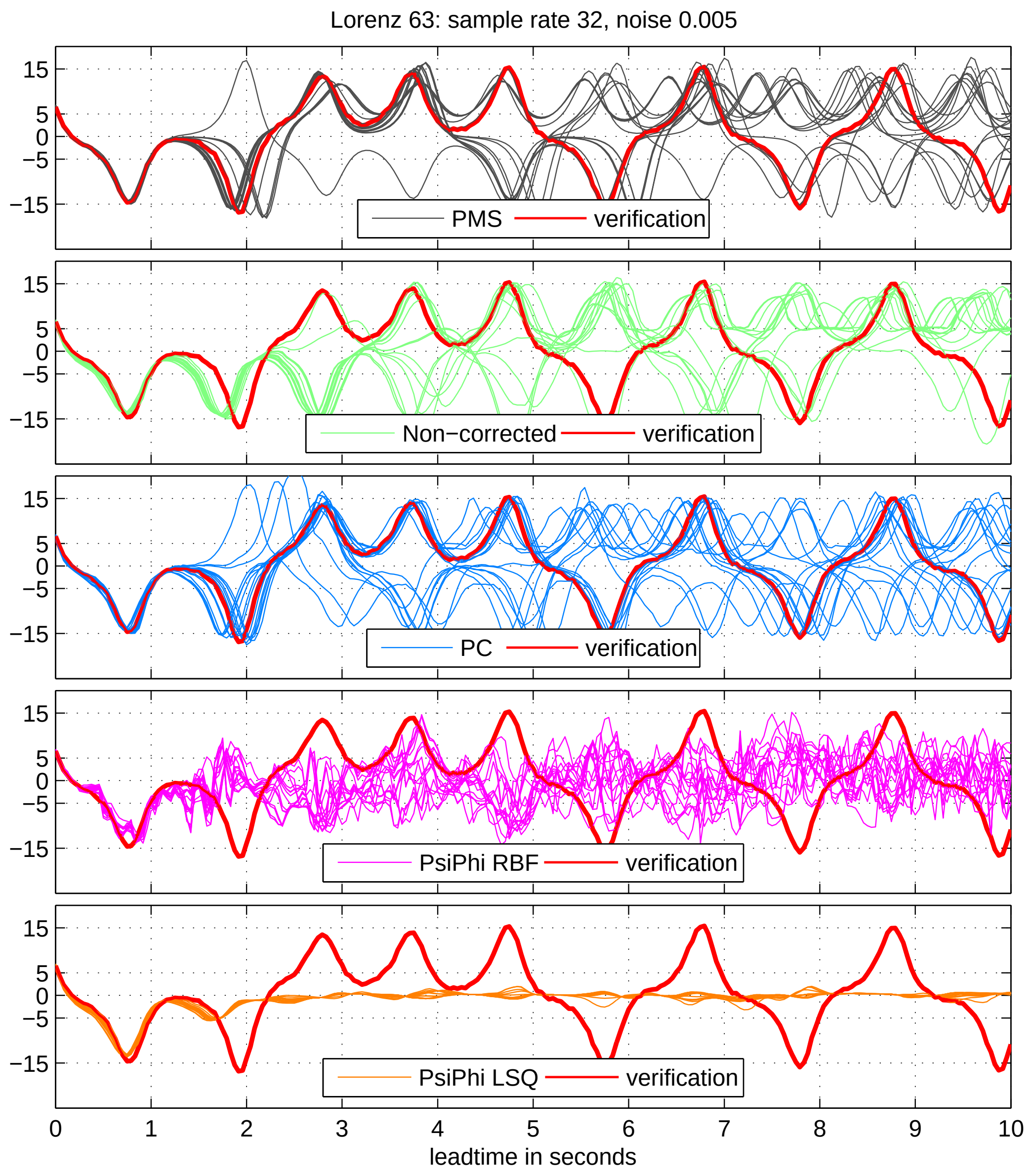

Figure 6.8: **lorenz63 Forecasts:** Same models as in Fig: 6.7, used here to generate forecasts of x variable of the lorenz63 system 10 seconds ahead. The non-corrected model loses skill after 1.5 seconds. *PC* performs very well, while $\Psi\Phi_{LSQ}$ converges to mean rapidly.

For this particular forecast, it seems that *PC* outperforms PMS. This would be a false impression though. Although some ensemble members of PMS diverge from the verification, many lie so close to the verification, literally on top of it, that they are difficult to distinguish. This is not the case for PC, where the ensemble members are close to the verification but do not lie on top of it. Overall, PMS has many more ensemble members that are much more accurate, i.e. they lie on top of the verification, than the PC. Another point to make is that Fig. 6.7 shows a single forecast and one should not think that all PC forecasts are as good as this one. There will be forecasts in which PC does not so well so well, but PMS retains its strong performance. The only relevant summary of the skill must be calculated across all forecasts and is captured in terms of Ignorance below. As expected, the summary shows that the PMS outperforms PC, and not the other way around.

Next, we look at forecasts of the lorenz63 system for a selected time period presented in Fig: 6.8. The $\Psi\Phi_{LSQ}$ forecast quickly converges to the mean; beyond the leadtime of 1.5 seconds it is not useful. The $\Psi\Phi_{RBF}$ also loses its forecasting skill at around 1.5 seconds. Although the mean convergence is not so strong the forecast of this particular time period is not very useful. In fact both the $\Psi\Phi_{LSQ}$ and $\Psi\Phi_{RBF}$ seem to actually degrade the forecast of the core model, possibly due to over-fitting or the error surface being too complex for the models to capture. *PC*, on the other hand, performs very well and a number of ensemble members stay close to the verification for the whole time period.

6.4.1 Evaluating the correctors

Here we evaluate the Lorenz84 and Lorenz63 forecasts using the Ignorance relative to the unconditional climatology (see Section 2.2.3). In this exercise

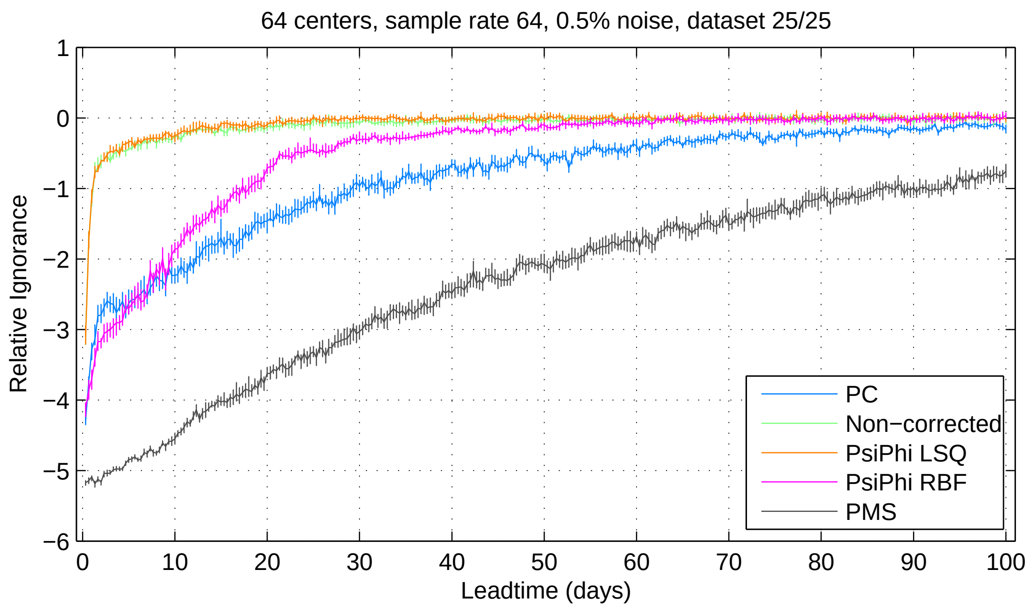

Figure 6.9: **Ignorance evaluation of Lorenz84.** Based on Ignorance the best performer of the 4 imperfect models is the *PC* (blue) followed by $\Psi\Phi_{RBF}$ (magenta). Both the non-corrected Φ model and $\Psi\Phi_{LSQ}$ quickly lose forecasting skill beyond leadtime of 2-3 days. The Ignorance of PMS (black) is shown for comparison.

we set the number of centers of the core model to $M = 64$. We are aware that the limited number of centers constrains the quality of the core model forecasts. Our aim, however, is to obtain insight regarding the performance of the different correctors.

In Fig: 6.9 we show the results for the Lorenz84 system for forecasts of up to leadtime $L = 100$. As expected, in terms of Relative Ignorance the non-corrected core model (green) performs rather poorly and loses skill before the leadtime of 5 days. The Ignorance of $\Psi\Phi_{LSQ}$ (orange) is comparable to the core model and suggests that the linear based corrector does not improve the core model forecasts. The $\Psi\Phi_{RBF}$ model (magenta) outperforms the core model significantly and maintains a forecasting skill up to 30 days. At the very short leadtimes, $\Psi\Phi_{RBF}$ outperforms PC; the statistical significance is confirmed by the crossvalidated error bars, which do not overlap. The *PC* initially loses up to 0.5 Bits of performance to $\Psi\Phi_{RBF}$ at the very short leadtimes, but then the performance stabilizes and PC starts outperforming $\Psi\Phi_{RBF}$ around the leadtime of 10 days. Overall, PC produces very good corrections, maintaining an advantage of 0.5 Bits of Ignorance, even at the leadtime of 60 days. The *PC* forecasts remain useful about twice as long as $\Psi\Phi_{RBF}$ (blue) and 10 times longer then the non-corrected Φ model. We also note that at leadtime 30 days, the *PC* has about 0.8 Bits of advantage over $\Psi\Phi$, making its forecasts almost twice as good as those of $\Psi\Phi$.

The PMS (black) is plotted for benchmarking reference and, as expected, heavily outperforms all models at all leadtimes. Note that the PMS loses 4 out of 5 Bits over the 100 leadtimes. This means that 100 leadtimes can be safely considered as a long prediction horizon since even the (far) superior perfect model loses a large proportion of skill over 100 leadtimes.

A similar result emerges in Fig: 6.10, where we show results for the Lorenz63 system, zooming in on the first 8 seconds of the forecasts. The *PC* model delivers a very significant improvement, and in terms of Ignorance outperforms $\Psi\Phi_{RBF}$ by about 1 Bit at leadtime 2 seconds. As above, the non-corrected core model does not perform very well. The $\Psi\Phi_{LSQ}$ model is at points outperformed by the non-corrected core model, which suggests that $\Psi\Phi_{LSQ}$ may over-fit and consequently degrade the core model forecasts. The $\Psi\Phi_{RBF}$ model does improve the core model to some extent, especially at leadtimes up to 1 second.

Based on this particular example, the *PC* method is superior to the other models (apart from PMS of course). However, to provide a more complete assessment, a more capable core model should be deployed. We will look at

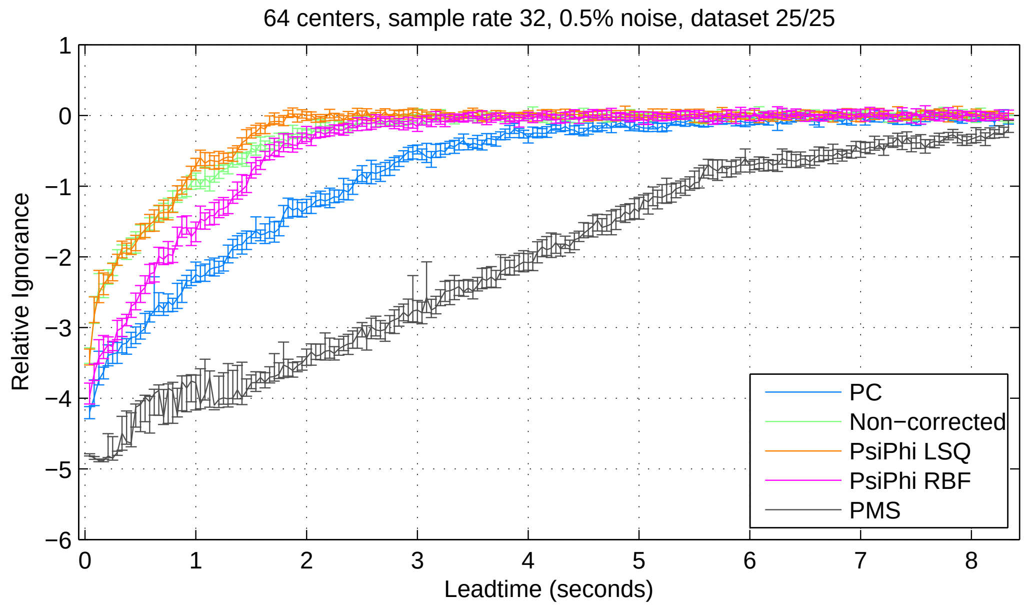

Figure 6.10: **Ignorance of Lorenz63.** Similarly to Fig 6.9, the PC model (blue) delivers a significant improvement. $\Psi\Phi_{RBF}$ (magenta) improves the core model forecast (green) only marginally, while $\Psi\Phi_{LSQ}$ (orange) degrades the core model forecasts, possibly due to over-fitting.

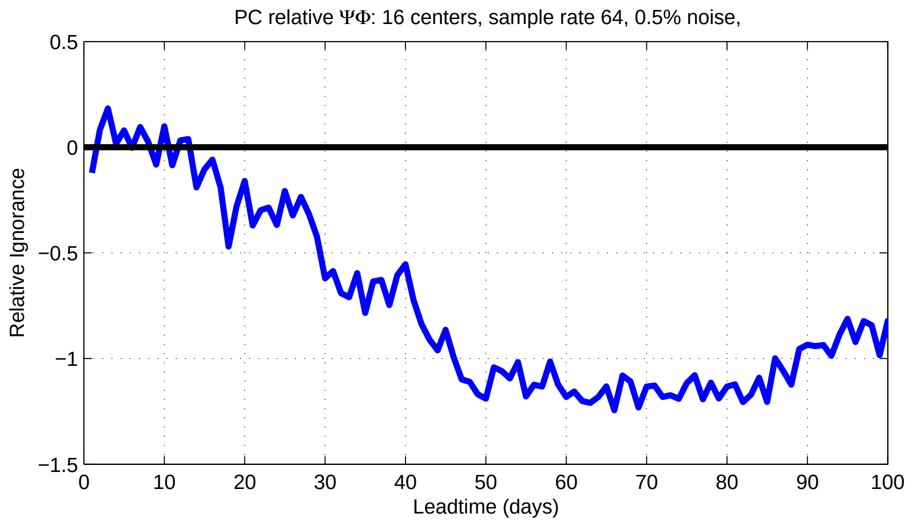

Figure 6.11: **Lorenz84** *PC* **v. ΨΦ.** Ignorance of the *PC* relative to ΨΦ (blue) shows that beyond the leadtime of 15 days the *PC* is outperforming the ΨΦ. Despite initial underperformance (up to 10 days), the *PC* gains a great advantage of up to 1 Bit between leadtimes 45 to 100 days over ΨΦ. Note that below the zero line, *PC* outperforms ΨΦ.

settings using core models of higher complexity in Section 6.4.3.

One final look at the relative performance of the *PC* and ΨΦ is provided in Fig: 6.11. We plot the median of the bootstrapped Ignorance of the *PC* relative to ΨΦ, $\mathrm{I}_{PC} - \mathrm{I}_{\Psi\Phi}$, for the Lorenz84 model, where we use a core model with 16 centers. At the very short leadtimes, up to 10 days, ΨΦ marginally outperforms PC, by about 0.2 Bits, but both models significantly improve the core model. We are not certain why ΨΦ outperforms PC at the very short leadtimes; we note, however, that at those leadtimes *PC* undergoes a period of instability, which might be pointing to some numerical issues arising from the iterative nature of the algorithm, and we aim to investigate this in our future work. As the leadtime increases beyond 15 days the *PC* starts outperforming ΨΦ (drops below zero line), and beyond the leadtime of 45 days it has 1 Bit advantage over $\Psi\Phi$. The PC maintains the advantage all the way up to a leadtime of 100 days, making it twice as good a predictor as $\Psi\Phi$ beyond leadtime 45 days. In fact, PC still has an advantage of 0.5 Bits even at the long leadtime of 200 days, which is not plotted here. We are aware that this remarkable superiority of the PC is related to the weakness of the core model, but as we have seen in the above Fig: 6.9, the PC is a much better performer even in settings with a more complex core model.

6.4.2 Limitations of RMS evaluation

In Section 2.4.2 we have stated that Root Mean Square Error (RMS) is not a proper score. The propriety of RMS is discussed in [14], which points out that the mean squared error (MS)

$$\int (X - z)^2 p(z) dz$$

depends on the the distribution p(x) only through its first and second moments and does not reflect any other aspects of p(x). The same applies to RMS; since RMS is just a nonlinear function (square root) of MS, it also depends on p(x) only through its first and second moments. One of the direct implications is that the RMS, by its very nature, cannot capture multimodality. Since multimodality is often present both in dynamical, e.g. Lorenz63, and stochastic systems, RMS may not be a good choice of performance measure.

Despite the fact that impropriety of the MS (and hence RMS) has been established, a number of studies continue to use RMS to evaluate probabilistic

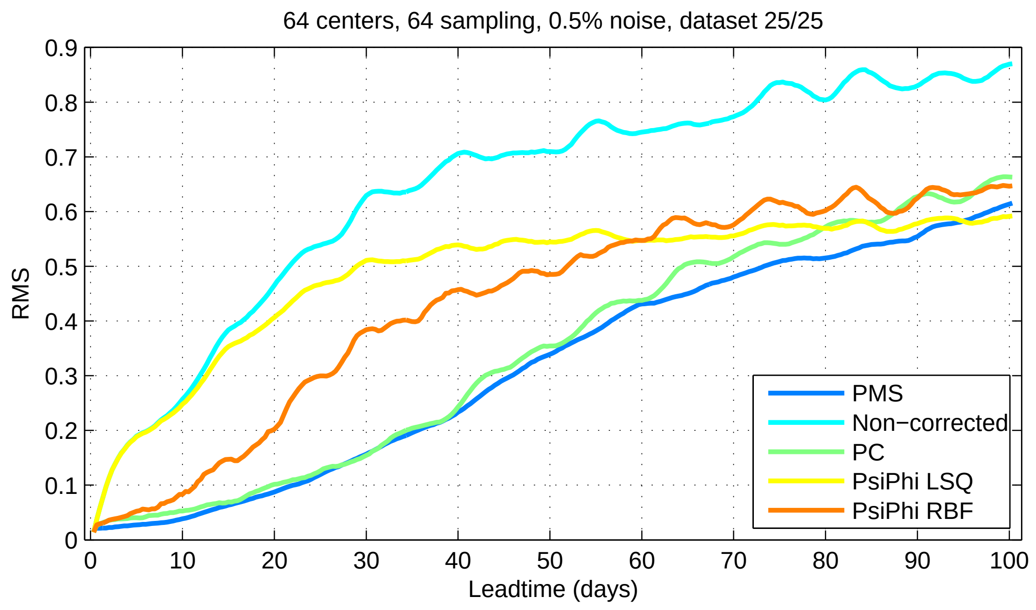

Figure 6.12: **RMS Lorenz84.** The RMS of the ensemble mean is plotted against leadtime for the 5 model forecasts of Lorenz84. Based on the RMS the forecasting performance of the PC appears comparable to the PMS, a result that is completely incorrect as the PMS is by definition far superior model to the PC.

forecasts, thus discarding much of the information content present in a forecasting ensemble. We believe that using an improper measure such as RMS in a probabilistic setting is misleading and aim to demonstrate some undesirable effects of doing so. Although the material in this section is not new in terms of theoretical results, we believe that the practical demonstrations are both novel and important. To demonstrate the implications of using RMS in a probabilistic setting, we evaluate the results of the previous Section 6.4.1 in terms of RMS and compare them with Ignorance-based evaluation.

In Fig: 6.12 we plot the RMS for the Lorenz84 system. The most striking observation is that under the RMS, the PC appears to perform almost as well as the PMS. Up to the leadtime of 40 days the PC maintains almost the same

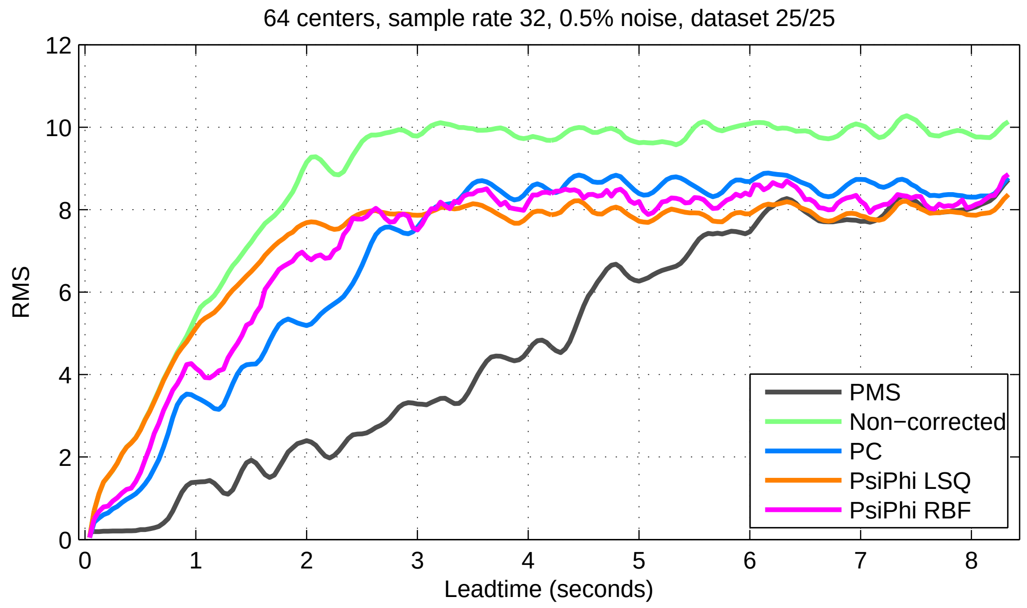

Figure 6.13: **RMS Lorenz63.** At the long leadtimes beyond 5.5 seconds the RMS evaluation rates $\Psi\Phi_{LSQ}$ as the best model, between leadtimes 6 and 8 even comparable to the PMS, an incorrect conclusion since PMS must, by construction, outperform $\Psi\Phi_{LSQ}$.

(apparent) skill, and beyond this leadtime it is outperformed only marginally. This result is not only counter-intuitive, but shows how dangerous the RMS evaluation can be. The PMS is by definition far superior to any imperfect model, and hence is expected to outperform any competing model by a large margin.

It is useful to draw a comparison with Fig. 6.4, which shows that for the single (subjectively selected) forecast, PC seems to be as close to the verification as the PMS up to leadtime 40 days and perhaps even closer beyond that. Although, as noted above, Fig. 6.4 only shows a single forecast, it is important to note that PC across the forecasts indeed stays close to the verification up to leadtime 40 days. However, many of the PMS ensemble members stay much closer to the verification for much longer. While Ignorance-based evaluation greatly rewards the PMS ensemble members, the RMS evaluation fails to do so. Consequently, RMS fails to detect the basic fact that PMS is, by construction, a far better model than PC. Furthermore, under the RMS $\Psi\Phi_{LSQ}$ performs comparably to the PMS beyond leadtime 95 days. These results are, again, counter-intuitive and in stark contrast with the previous findings where the PMS outperforms all the other models by a large margin and at leadtime 100 days maintains an advantage of 1 Bit, which is expected by construction.

The issues are less striking, but still persist, for the Lorenz63 system. The RMS plotted in Fig: 6.13 correctly suggests that the PMS is the best model, although it is deprived of its lead slightly early at a leadtime of 6 seconds. The important observation here is that $\Psi\Phi_{LSQ}$ significantly outperforms the non-corrected core model, despite the fact that $\Psi\Phi_{LSQ}$ only forecasts the mean, while the core model is able of reproducing the bimodality, see Fig: 6.8. While the forecast of $\Psi\Phi_{LSQ}$ is completely useless, the RMS rates it higher than the core model because under RMS the mean forecast is closer to the verifications. For bimodal variables, the RMS tends to score higher in the mean forecasts higher than imperfect bimodal forecast. In such situations the RMS may choose a useless model over a useful one.

6.4.3 Impact of number of centers

In Section 6.4.1 we have shown that the PC achieved the best forecasting performance of the 4 imperfect models. We have noted that, for a fair evaluation, the forecasting ability of the core model should not be restricted so heavily. We therefore set up an experiment in which we increase the number of centers of the core model, making it more complex (and more competitive), thus allowing us to study the additional value of a corrector as the forecasting performance of the core model improves. In this experiment we use the core model with 32, 64, and 128 centers. We only present results for the PC and $\Psi\Phi_{RBF}$ since $\Psi\Phi_{LSQ}$ was outperformed by both PC and $\Psi\Phi_{RBF}$ in all analyses. The analysis was performed for both Lorenz84 and Lorenz63 systems. However, since both analyses produced similar results, we only present the Lorenz84 case.

In Fig: 6.14 we plot Ignorance for the non-corrected core model and the PC for the first 50 forecast days of the Lorenz84 system. As we increase the number of centers of the core model the non-corrected forecasts improve significantly. The forecasting performance of the PC remains somewhat unchanged. This finding suggests that improving the core model, which indeed is part of the PC model, may have very little effect on the final forecast of PC. In Fig: 6.15 we repeat the experiment for $\Psi\Phi_{RBF}$. We again see improvement of the non-corrected forecasts and this time we also see some, although not large, improvement in $\Psi\Phi_{RBF}$ forecasts.

The results show that correctors of both the PC and $\Psi\Phi_{RBF}$ models are

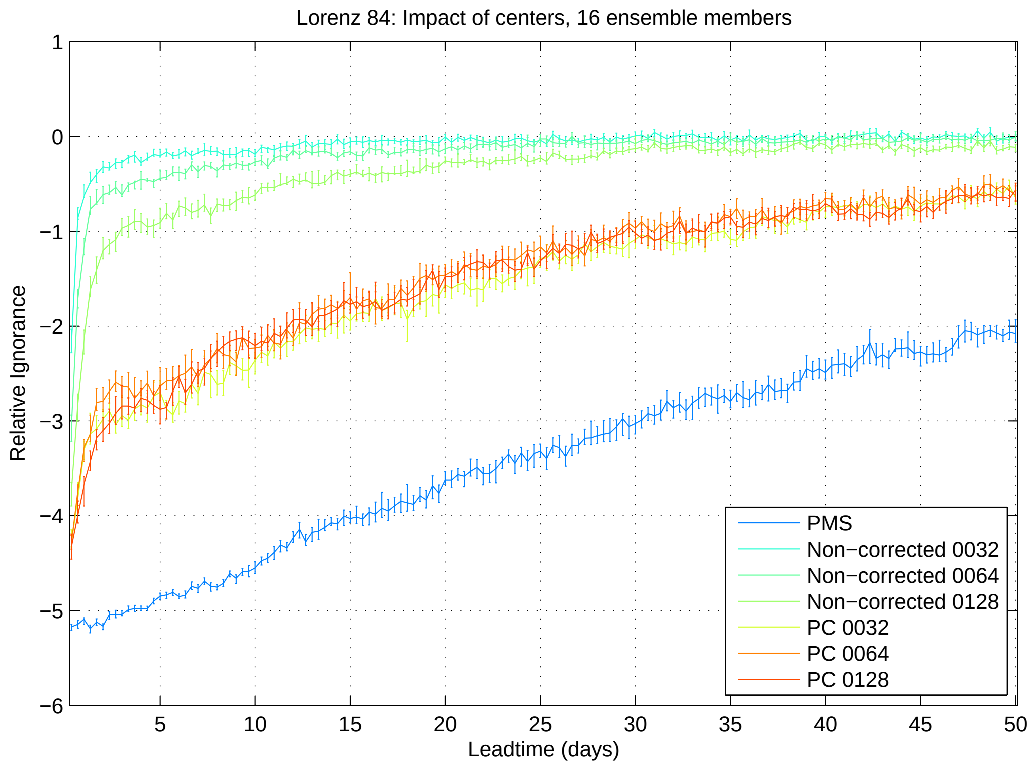

Figure 6.14: **Impact of number of centers on** *PC* **(Lorenz84):** The core model complexity is increased by gradually increasing the number of centers from 32 to 64 and 128 (green shades). The performance of the core model improves with the complexity; the model with 128 centers has up to 0.8 Bit of advantage over the 32 center version at leadtime 5. The performance of the *PC* is insensitive to the improvements of the core model. *PC* appears to operate at its potential.

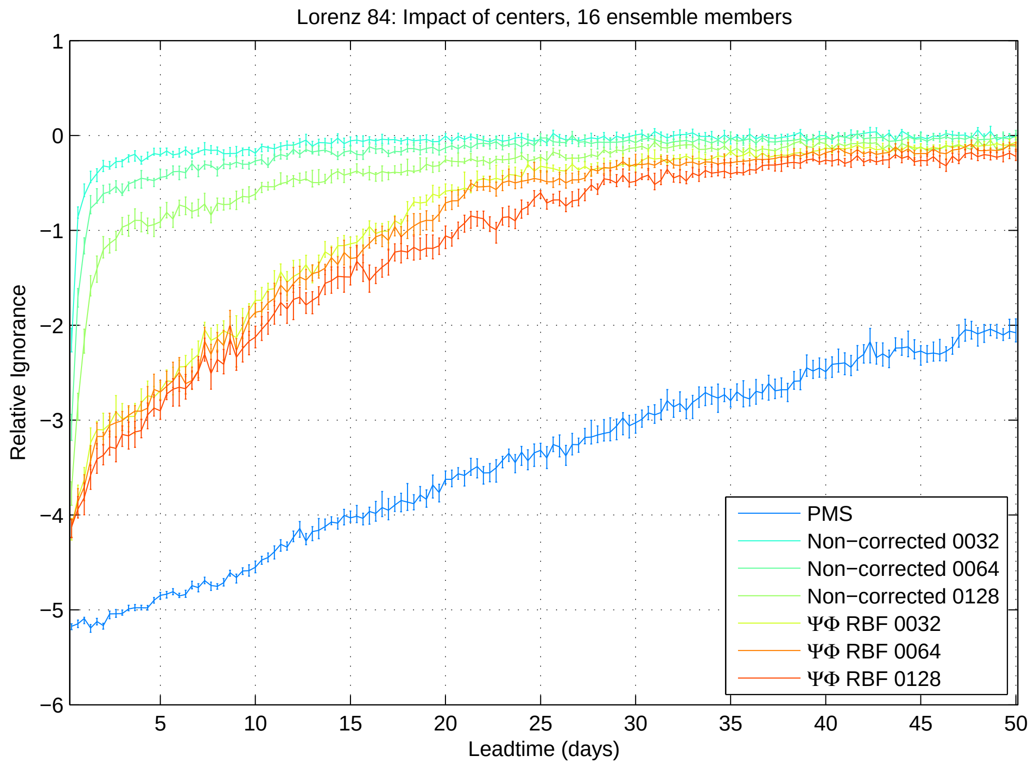

Figure 6.15: **Impact of number of centers on $\Psi\Phi_{RBF}$ (Lorenz84):** Similar to Fig: 6.14 but here $\Psi\Phi_{RBF}$ is studied. Increasing the complexity of the core model leads to a marginal improvement of $\Psi\Phi_{RBF}$ forecasts. For the core model with 128 centers $\Psi\Phi_{RBF}$ slightly outperforms the less complex versions between leadtimes 15-30 days.

robust, in the sense of being insensitive to improvements of the core model.

6.4.4 Dataset size and number of centers tradeoff

In Section 5.1.2 we noted that, for large datasets, the matrix A of the linear system of Eq: 5.1 becomes intractable and RBF approximation must be adopted. In Section 6.2.2 we suggested that the number of centers may have a significant influence on the forecasting performance of a model. We argued that more centers provide a better description of the approximated surface. It follows, that in short datasets, we can afford to deploy more centers than in large ones. There is a tradeoff, between the amount of information available and quality of the surface description. Small datasets may contain less information but allow for more centers to be used, while keeping A tractable.

To study the tradeoff we artificially decrease the training and learning dataset at the benefit of increasing number of centers. In particular we only use one tenth of the two datasets, which allows us to use $M = \{128, 256, 512, 1024\}$ centers for the core model. We are interested to see whether the loss of information will be offset by a better description of the approximated function.

In Fig: 6.16 we plot Ignorance relative to climatology for the non-corrected core model and the PC model forecasts of the Lorenz84 system. The core model is trained on the reduced training dataset containing only one fifth of the original observations. Similarly, the C corrector is trained on the reduced learning dataset. We observe that neither the data reduction nor the increasing number of the centers influences the PC model. The non-corrected model improves significantly; its forecasting performance becomes comparable with the PC model when the number of centers is increased to $M = 1,024$ centers, i.e. when the complexity of the core model is significantly increased. This means that it is indeed relatively easy to correct a model

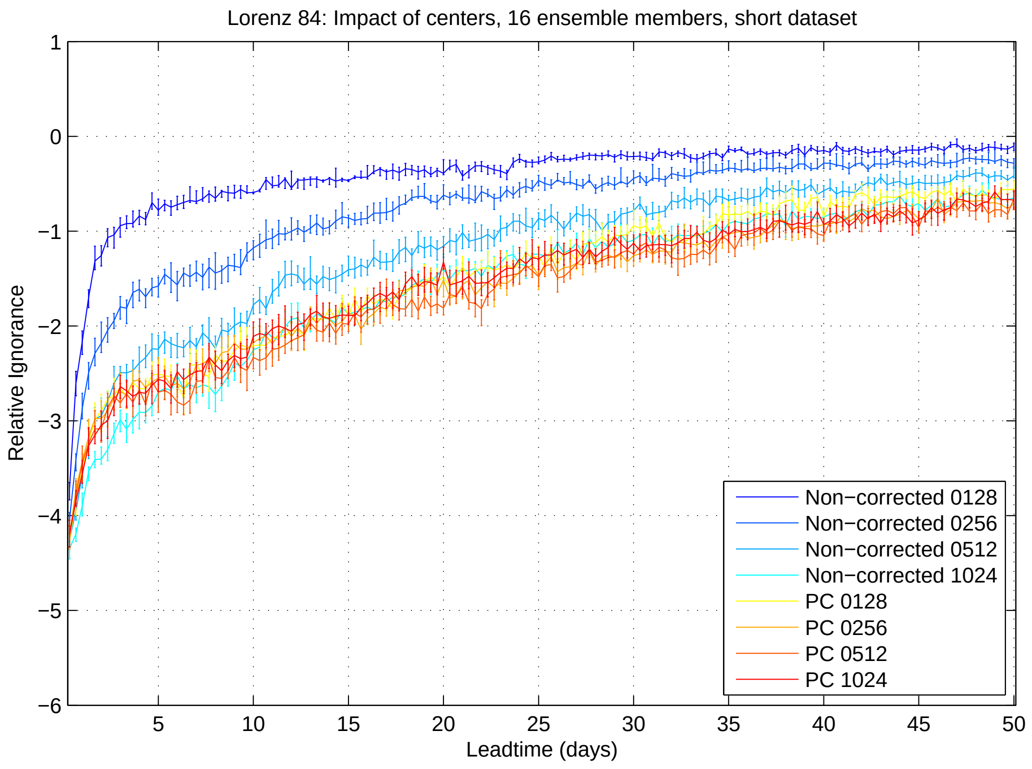

Figure 6.16: **Impact of dataset size on *PC* (Lorenz84):** The short dataset (see Appendix C.1) is used to perform core model and *PC* forecasts. The complexity of the model is varied by using $M = \{128, 256, 512, 1024\}$ centers; more centers can be afforded due to the lower dataset size. The core model performs much better then in Fig: 6.14. The version with 1,024 centers is comparable to the performance of the *PC*. The *PC* is robust and delivers significant improvements for the low complexity models.

with large systematic errors (not necessarily a simple model), and less easy to correct a model with low systematic errors (not necessarily a complex model). This trivial conclusion, however, is not the main finding here. The main finding is that a two-stage predictor-corrector is indeed capable of improving forecasts if the core model is poor, while it does not degrade forecasts when the core model is very good. In other words, the two-stage predictor-corrector performs at the upper bound of the forecasting potential. Even a very good single core model is not able to outperform the two-stage procedure.

In real-world applications, single-stage procedures are often deployed and core models frequently suffer from over/under fitting and the resulting forecasting biases. Applying the two-stage approach suggested above may significantly reduce forecasting biases.

6.5 Why does PC outperform $\Psi\Phi$?

There are a number of reasons why *PC* outperforms $\Psi\Phi$ in terms of providing better probability forecasts. First of all, $\Psi\Phi$ aims at minimizing RMS error, which does not necessarily lead to better probability forecasts. Secondly, *PC* focuses on interpolating one surface, the leadtime 1 error surface, while $\Psi\Phi$ looks at each error surface at each leadtime (there is one $\Psi\Phi_{LSQ\backslash RBF}$ for each leadtime). For long leadtimes, $\Psi\Phi$ is then attempting to approximate a very complicated surface, a complication *PC* avoids. In addition, *PC* has more centers available to it than $\Psi\Phi$.

So why does *PC* use more centers? We recall that the $\Psi\Phi$ approach is applied at each leadtime. Only errors recorded at a given leadtime are available for the corrector. The number of errors at each leadtime corresponds to the number of forecasts available at a given learning set. If there are, say, $1,500$

forecasts, only 1,500 errors are available. The *PC*, on the other hand, uses a leadtime 1 forecast only, i.e. a forecast can be launched at every point of the learning dataset. If there are N datapoints in the learning set, then there are $N-1$ forecasting errors available.

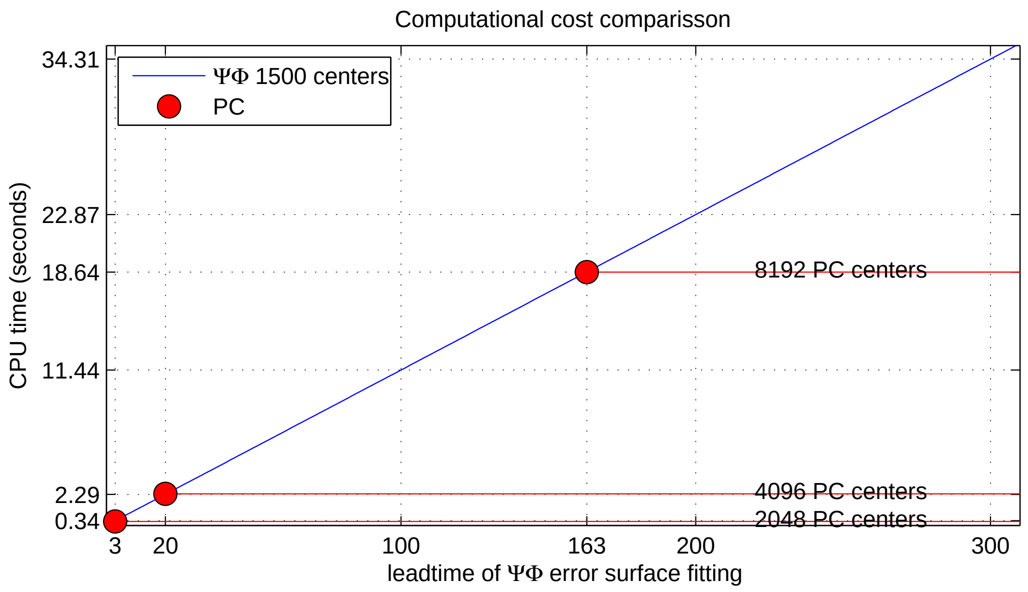

Figure 6.17: **Impact dataset size.** Comparison of computational intensity of *PC* and ΨΦ. Considering ΨΦ with 1,500 errors available at each leadtime. To train ΨΦ for 163 leadtimes takes about the same time as solving *PC* system for all leadtimes with 8,192 centers. 8,192 centers yields much better surface description than 1,500 centers of the ΨΦ. Moreover *PC* with 8,192 is fully trained by the time ΨΦ is in the middle of the training process.

There is another point to make. Assume that we know that for a given dataset the error surface is well described by 4,096 centers. Why not increase the number of forecasts so that ΨΦ also has 4,096 forecasting errors available? Increasing the number of forecasts from, say, 1,500 to 4,096 would lead to almost tripling the size of the evaluation dataset, with the obvious consequence of higher computational costs. In both Fig: D.1 and Fig: 6.17

we see that solving a linear system of $4,096 \times 4,096$ takes about 2.3 seconds. Solving $\Psi\Phi$ with 4,096 centers at each leadtime L times longer, L being the maximum leadtime. Some of our analyses use $L = 800$; the time required to form $\Psi\Phi$ would be 800 times longer than the time required to form PC. In this setting, we are looking at 30 minutes for $\Psi\Phi$ solution, compared to 2 seconds for the PC solution.

The final point is that the $\Psi\Phi$ corrector uses less centers then the PC, so that the computational cost of calculating the LS solution at a given leadtime is rather small. How much do we gain from the lower (per leadtime) CPU cost? Compared to the PC, we gain very little. In Fig: 6.17 we plot the CPU time against the leadtime for the $\Psi\Phi$ (blue line). It takes about 2.3 seconds to fit the $\Psi\Phi$ for the first 20 leadtimes. Now we can ask, how long does it take to fit the entire PC corrector with 4,096 centers? In fact it also takes 2.3 seconds for this example. Going even further, to obtain $\Psi\Phi$ correctors needed to forecast the first 163 leadtimes takes about 18.6 seconds. What can the PC do in 18.6 seconds? It can either fit the entire PC corrector with 4,096 centers about 8 times in a row or it can fit a much larger matrix A, using up to 8,192 centers. Despite the high number of centers deployed by the PC approach, overall the PC is still faster to form when considering the above analyses.

6.6 Conclusions

In this chapter, we have introduced PC, a novel two-stage approach to correct a systematic part of the forecasting error. When applied to low-dimensional chaotic systems, Lorenz84 and Lorenz63, PC has been shown to significantly reduce systematic errors of a core model, in particular when applied to core models of low-to-high complexity (number of centers $< 1,024$). For the

Lorenz84 system, at short leadtimes with a medium complexity model (64 centers), PC outperforms the single stage core model by up to 2 Bits, making the PC forecast 4 times more valuable (see Fig. 6.9). For the Lorenz63 system, at short leadtimes with a medium complexity model (64 centers), we see improvement of about 1.5 Bits, improving the original forecast more than twice.

When applied to core models of high complexity ($\geq 1,024$ centers), PC does not significantly improve the forecasting performance of the single stage core model. This is an intuitive finding; increased model complexity reduces scope for systematic errors which renders the PC corrector less effective. In other words, if there are no errors, PC has nothing to correct. An important finding, however, is that PC does not degrade the forecasting performance, i.e. applying PC to a good model does not hurt the forecast.

PC has also been contrasted with an alternative state-of-the-art approach, $\Psi\Phi$. In general, PC significantly outperforms $\Psi\Phi$ in both studied systems, Lorenz84 and Lorenz63. This finding is robust under the variation of the core model complexity. For the Lorenz84 system PC outperforms $\Psi\Phi$ by up to 1 Bit, even at long leadtimes in a low complexity setting (16 centers), and up to 0.8 Bits at medium leadtimes in a medium complexity setting (64 centers). Interestingly, when applied to the Lorenz84 system, at the very short leadtimes (up to 5 days) PC is marginally outperformed by $\Psi\Phi$ (0.2 Bits) in both low and medium complexity settings. While we have no direct explanation of this observation, our intuition points toward computational issues; at short leadtimes PC undergoes numerical instabilities potentially related to the use of radial basis functions.

To study the tradeoff between the quality of information (captured by the number of observations) and the quality of the model (captured by the number of centers), we 'starved' PC of data by cutting the size of the data available for training. By using less data we could afford to increase the quality of the core model while keeping the computational cost fixed. For the Lorenz84 system, PC has maintained its efficiency and improved the core model forecasts by up to 2 Bits for a model with 128 centers, and by up to 1 Bit for a model with 256 centers (see Fig. 6.16). An important finding is that even with short datasets, PC is able to substantially improve core model forecasts. Similar results were obtained for Lorenz63.

As an additional exercise, using the forecasts of PC, $\Psi\Phi$ and PMS, we have numerically illustrated how misleading, and potentially dangerous, evaluation based on the RMS can be. In the Lorenz84 system we have shown that under RMS evaluation, an imperfect model can be rated as high as a perfect model; a result that is incorrect by definition of PMS. Also, in the Lorenz63 system we have shown that the RMS prefers a model which produces a mean forecast over a model which is, to some extent, able to capture the system dynamics.

We note that PC has been tested on low-dimensional systems only and that in high-dimensional systems our findings might differ from those reported here. At the same time, the design of our approach is general and the method is indeed applicable to medium-to-high dimensional systems. We note that the iterative nature of PC introduces an additional challenge in applying PC to high dimensional models such as weather models. However, PC is primarily designed for low-to-medium dimensional systems where the application is straightforward and PC is able to greatly reduce systematic errors and improve forecasting performance. In our opinion, PC can be successfully applied in a number of fields including economics, finance, biology, and low-dimensional applications in physics.

Appendices

Appendix A

Dynamical systems

A.1 The Lorenz84 system

The Lorenz84 system [16, 47, 83, 84] is a three-dimensional system given by

$$\begin{aligned}\dot{x} &= -ax - y^2 - z^2 + aF \\ \dot{y} &= xy - bxz - y + G \\ \dot{z} &= bxy + xz - z\end{aligned}$$

where the dot is differentiation with respect to the time, F represents the symmetric cross-latitude heating contrast and G represents the asymmetric heating contrast between oceans and continents. Unless stated otherwise the standard parameter setting [135] is: $a = .25$, $b = 4$, $F = 8$ and $G = 1$; and the initial conditions are $x_0 = 0, y_0 = 0, z_0 = 1.3$.

In this work we numerically solve the system using the *Runge-Kutta* fourth order method, e.g. [6], with the step size set to $h = 10^{-3}$. To ensure that we sample the systems attractor and that the dynamics of the timeseries is not affected by transient behavior, we discard the first 10^4 values of each integration output.

A.2 The Lorenz63 system

The Lorenz63 system [18, 81, 140] is a three-dimensional system given by

$$\begin{aligned}\dot{x} &= -\sigma x + \sigma y \\ \dot{y} &= -xz + rx - y \\ \dot{z} &= xy - bz\end{aligned}$$

The standard parameter setting [135] is: $sig = 10$, $r = 28.1$ and $b = 8/3$ and the initial conditions $\{x_0 = 0, y_0 = -0.01, z_0 = 9\}$.

In this work the system equations are solved numerically as described above in Appendix A.1.

A.3 The damped forced pendulum

A damped driven pendulum is a basic example of a chaotic system. The system consists of a pendulum, a point mass at the end of a mass-less rod. The pendulum is subject to a frictional damping proportional to the angular velocity of the pendulum, which slows the pendulum down (to a standstill in absence of a driving force). The pendulum is driven by an external force which provides a periodic torque. The dimensionless equation of motion [8] can be written as

$$\ddot{\theta} + \frac{1}{b}\dot{\theta} + sin\theta = g\cos(\omega_F t) \tag{A.1}$$

where θ is angle of the pendulum, ω_F is the angular driving frequency, g is the strength of the forcing and b is the inverse of the frictional damping. The term $\sin\theta$ represents gravitational restoring torque. Note that ω_F does not need to be equal to the natural frequency of the system.

Parameter value	Dynamics
b < 1.085	periodic
1.085 < b < 1.110	chaotic
1.110 < b < 1.140	periodic
1.140 < b < 1.220	chaotic
b = 1.220	periodic
1.220 < b < 1.280	chaotic
1.280 < b < 1.475	periodic
1.475 < b < 1.485	chaotic
1.485 < b < 1.493	periodic
1.493 < b < 1.495	chaotic
1.495 < b < 1.497	periodic
1.497 < b	chaotic

Table A.1: **Pendulum dynamics**. Changing the strength of external forcing, b modifies the system dynamics from periodic to chaotic. The parameter values for different regimes are reproduced from [7].

The equation of motion can be solved numerically by deploying an integration procedure such as the *Runge-Kutta* fourth order method [6]. For the purposes of this work the pendulum was integrated using the Matlab's numerical solver ODE45.

The system exhibits a variety of dynamics, including periodic orbits as well as chaotic motion. The standard parameter setting [8] is given by $\omega_F = 2/3$ and $b = 2$. The forcing parameter g then modifies the system dynamics as summarized in [7] (see Table: A.1).

Appendix B

Dynamic Climatology: data and evaluation

B.1 Evaluation

The forecasting densities are produced via Kernel Dressing (see Section 2.4.3), minimizing Ignorance over the bandwidth σ, the offset o and the blending parameter α, i.e. the PMS and climatological densities were blended (see Section 2.2.5) and Ignorance is subsampled in the case of the pendulum system of Section 4.2.2, or crossvalidated for the ENSEMBLES results of Sections 4.3.2 and 4.3.3. For subsampling and crossvalidation of Ignorance see Section 2.4.4. The forecasting performance is measured by the Relative Ignorance, Eq: 2.17. The reference forecast is either climatology or the DC as designated in the text or the captions of the relevant figures.

B.2 Pendulum dataset

The dataset used when demonstrating DC behavior on the pendulum (Section 4.2.2) was generated by integrating the equations of the damped forced pendulum in chaotic regime (see Section: A.3) with parameter settings: $\omega_F = 2/3$, $b = 2$ and $g = 1$. The integration scheme used was the Matlab's ODE45 [122] with default setting and tolerance of 10^{-8}. The output of the integration scheme was sampled at the sampling frequency $f_s = 10$ to obtain a time series of 2^{12} system states. The observations are created by the addition of Gaussian noise $N(0, \sigma_P)$, where $\sigma_P = 0.005 * \text{range}$. A perfect model is used to forecast the data.

B.3 Nino34 and MDR datasets

While there were datasets of 5 ENSEMBLES models available at the time of writing of this thesis, due to technical issues on our infrastructure, only 4 models were available to us. We use forecasts produced by the 4 models over the period of 1960 - 2001. Some ENSEMBLES models have covered longer periods (beyond 2001), however we use the longest common period of all the 4 models available to us. All of the ENSEMBLES models forecasts are initialized from 9 initial conditions, i.e. consist of 9 ensemble members at each forecast leadtime. The forecasts are initialized in February, May, August and November of a given year, yielding 41 forecasts over the considered period. The February, May and August launches forecast 7 leadtimes ahead. The November launch forecasts 14 months ahead and is designed to test annual forecasting capabilities of the models. The DC forecasts are initialized at the same dates as the ENSEMBLES forecasts. The leadtimes are also synchronized so that the DC and ENSEMBLES models are fully comparable.

The ENSEMBLES models are simulation models, which forecast number of fields such as temperature, pressure etc. We only study a single variable, the sea surface temperature (SST), in the two regions of Nino3.4 and MDR (see Section 2.5.2 for details). In particular, we calculate monthly spatial averages of the SST over the two regions. The ECMWF ERA 40 reanalysis is used as verification in the evaluation procedure.

B.4 Probability plumes: ECMWF and Dynamic Climatology

In this appendix we show an extended version of Fig: 4.17. In particular we show probability plumes (red to yellow patches) of all SST, November initialization forecasts over the Nino3.4 region produced by both the ECMWF model and the Dynamic Climatology (DC). Each patch of a distinct color represents a single contour of a forecasting distribution over all leadtimes. The contour levels are given by the following percentiles: 1-99 (red), 5-95, 15-85, 25-75 and 45-55 (yellow).

Since the maximum leadtime is 14 months, the adjacent forecasts partially overlap. There are several notable SST events stretching over the periods of 1971-1973, 1975-1978, 1981-1983 and 1996-1998.

Fig: B.2 shows plumes of the ECMWF model. Note that, with the exception of the 1962 initialization, the probability forecasts never fully approach climatology (light blue patches). This means that the ECMWF model maintains some forecasting skill at all leadtimes, i.e. the blending parameter α is never zero, although it might be close to zero. Also note that both the onset and the offset of the 1996 event are captured very well, assigning high probabilities to the verification (red line).

In Fig B.2 we show the same results for the DC forecasts. Firstly note, DC plumes approach climatological plumes around leadtime 6, which is consistent with the Ignorance evaluation of Section 4.3.2. At leadtime 6 the blending parameter α becomes effectively zero, i.e. DC no longer outperforms climatology in terms of Ignorance; the climatological forecast takes over. Further note, DC produces values not contained within the training set as the plumes of several forecasts (e.g. 1982, 1994, 1997 initializations) reach outside the climatological plumes. Finally, DC forecasts of the 1996-1998 event captures both the onset and offset quite well, although to a lesser extent than the state-of-the-art ECMWF model.

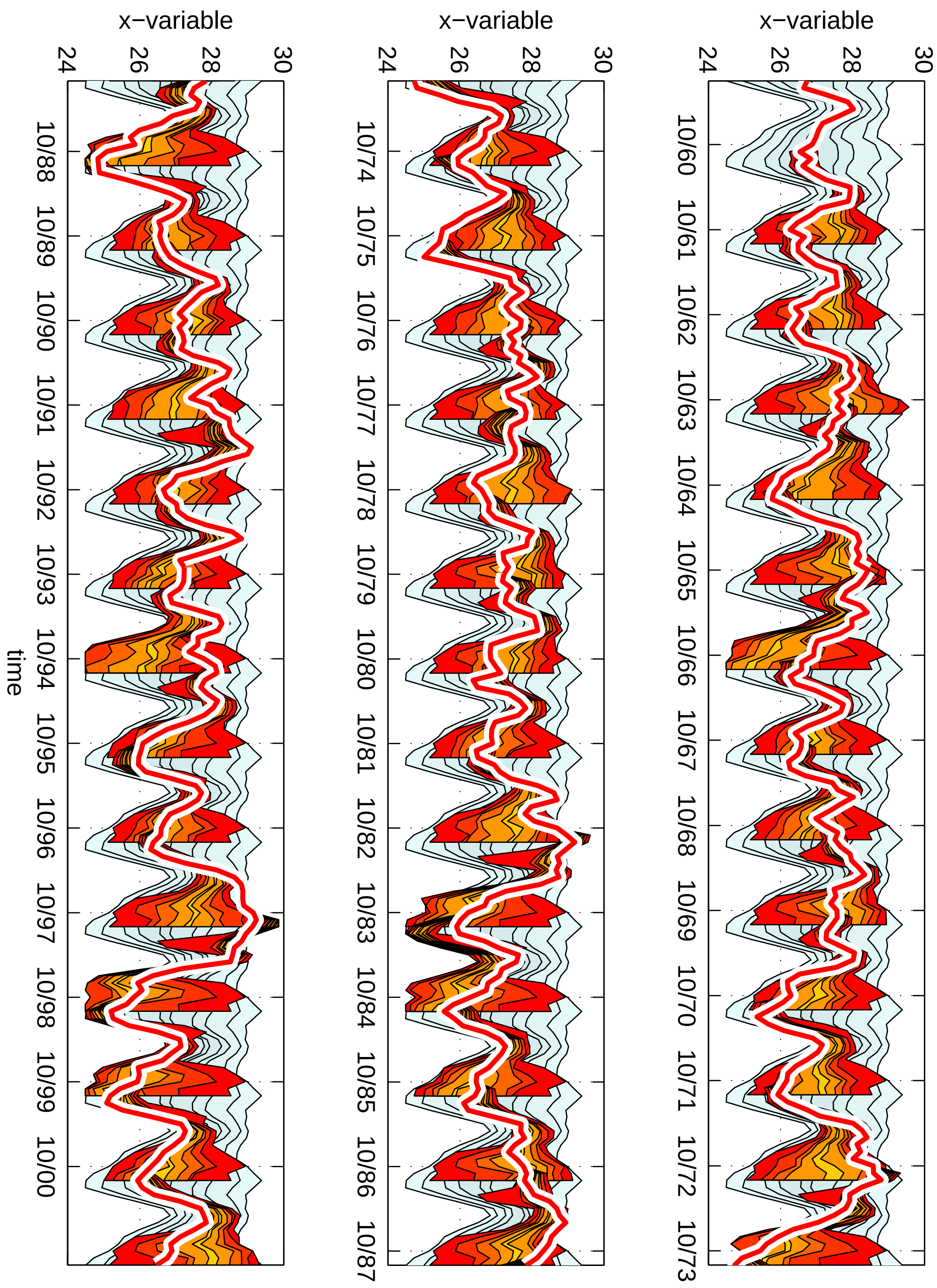

Figure B.1: **Probability plumes of ECMWF in Nino3.4:** Same as in Fig: 4.17 but here we show all forecasted time periods. Note the difficult-to-forecast events of 1975-1978, 1981-1983, 1996-1998

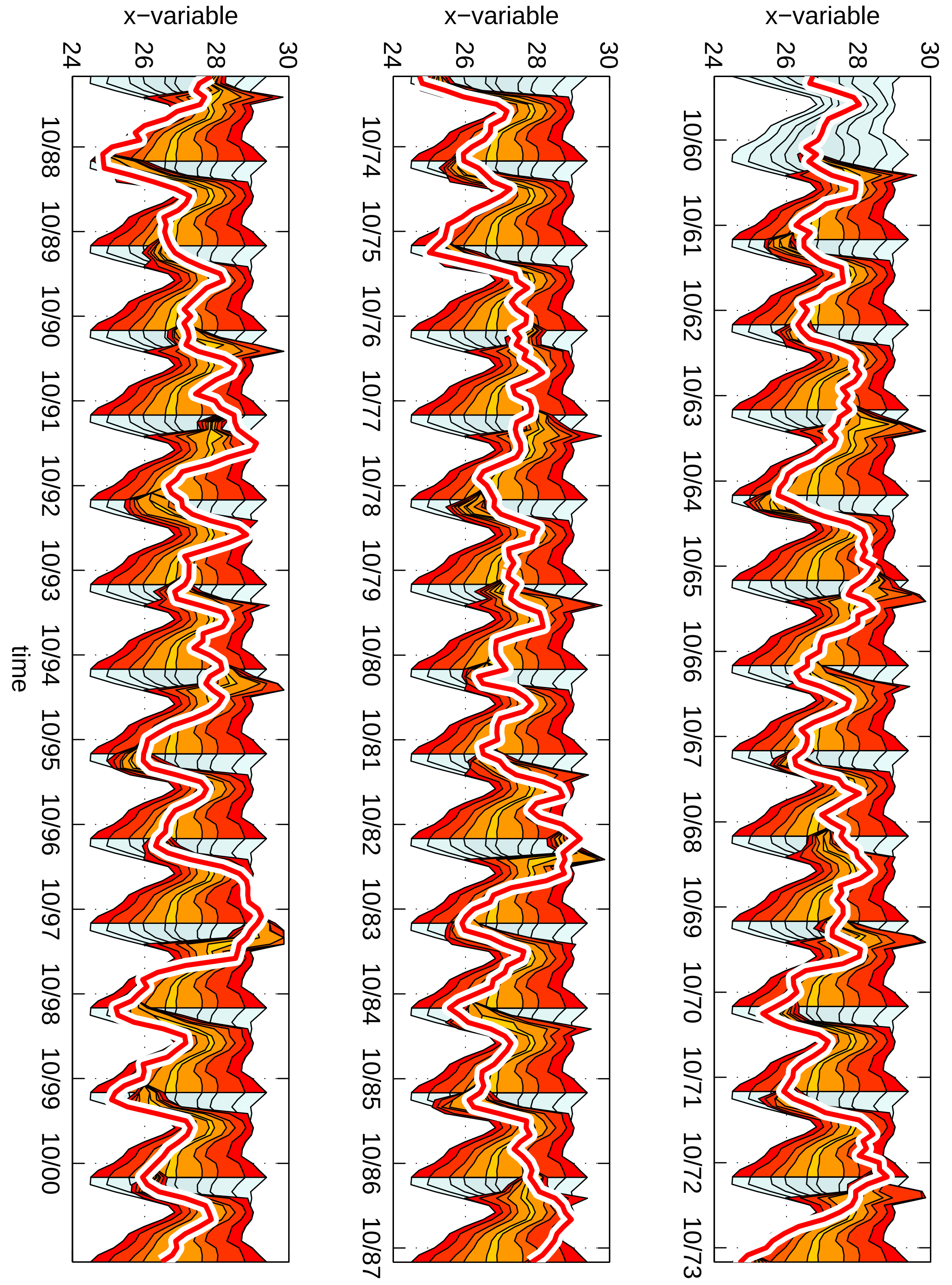

Figure B.2: **Probability plumes of DC in Nino3.4:** Same as in Fig: B.2 but here we show all DC forecasted time periods.

Appendix C

The ΨΦ/PC: data, core model, and evaluation

C.1 The datasets

The results presented were obtained using two dataset sizes for each system, Lorenz84 [83] and Lorenz63 [81]. We call the datasets simply long and short depending on how many forecasts are available. The size of the long dataset is $N = 1.5 \times 10^6$, the size of the short dataset is $N = 2.5 \times 10^5$. In both systems we forecast the x variable (see Appendix A). Details of the datasets are summarized in Table C.1 and discussed in the following text.

To obtain the system states we integrate the systems using the fourth-order Runge-Kutta integration scheme [17, 76, 123]. The integration step is set to $h = 10^{-3}$. For each analysis we generate three datasets: the *training set*, the *learning set* and the *testing set*, all of equal size. The training and learning sets are used to construct the base model described in Section 6.1. The learning set is used to collect the out-of-sample forecast residuals and

	Lorenz84	Lorenz63
size long dataset (train.+eval.+test.)	1.5×10^6	1.5×10^6
number of forecasts (long dataset)	665	665
size short dataset (train.+eval.+test.)	2.5×10^5	2.5×10^5
number of forecasts (short dataset)	100	100
max leadtime (L)	100 days	25 seconds
embedding delay (τ)	14	10
sample rate (f_s)	64	32
range of testing set (R)	2.9606	39.105
noise level as proportion of range (NL)	0.5%	0.5%
std of changes (σ)	0.0689	2.6960
noise in terms of σ	20%	7%

Table C.1: **Datasets description.** Long and short datasets are used to study the performance of $\Psi\Phi$ and PC when forecasting the two systems, Lorenz84 and Lorenz63.

training the second stage models of Sections 6.2 and 6.3. The testing set is then used to evaluate the forecasting performance of the models.

Time series of the observations were created by sampling the system states at a given sampling rate f_s and adding additive noise. The noise level is set to 0.5% of the range in all presented cases. The noise corresponds to about 20% of standard deviation of the changes of x for Lorenz84 and to about 7% for Lorenz63. The sampling rates used in the presented analyses are $f_s = 32$ for the Lorenz63 and $f_s = 64$ for Lorenz84. We note that we have tested a number of different datasets that included sampling rates of $f_s = \{16, 32, 64, 128\}$ and varying noise levels. The results obtained were consistent with those presented above.

Regarding the size of the datasets we used 'long' and 'short' datasets, with long consisting of hundreds of forecasts, and short only consisting of 100 forecasts. The number of forecasts available for each leadtime vary depending on the maximum leadtime. Consider a testing dataset of 1,000 observations and an exercise where we forecast 10 leadtimes ahead. The testing dataset thus contains 100 non-overlapping forecast initializations. Each of the 10 leadtimes will have 100 datapoints available. Keeping the size of the testing set fixed, but increasing the maximum leadtime from 10 to, say, 100 means that the training set will only consist of 10 different initializations. Consequently, there would only be 10 datapoints available to evaluate a forecast at each leadtime.

C.2 Forecasting settings

Regarding the forecasts, we set the maximum leadtime and the ensemble size. The maximum leadtime implicitly determines the number of non-overlapping forecasts in a manner described in the previous section, C.1. The ensemble size can be set to $E = \{8, 16, 32, 64, 128, 256\}$. Based on a number of testing runs, we set the typical ensemble size to $M = 64$. This ensemble size provided a good forecasting performance while keeping the computational cost (CPU time) manageable.

Regarding the core model parameters, in order to test sensitivity to variations in model quality we study settings with $M = \{128, 256, 512, 1024\}$ centers.

C.3 Data transformation

In all estimation and forecasting exercises we use embedding of the observed time series [138]. Embedding is a frequently-used method of attractor reconstruction. Assuming a d-dimensional dynamical system where only some of the state variables are observed, the attractor reconstruction uses the observed timeseries to reconstruct the full state space [46, 71, 117]. To embed a time series, two parameters, time delay τ and *embedding dimension* d_e are required.

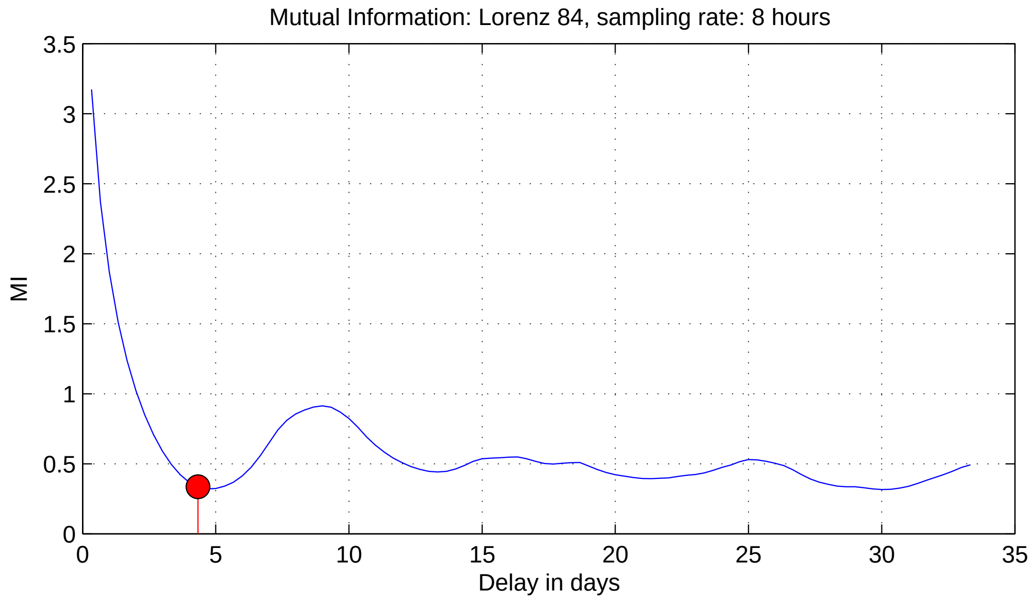

Figure C.1: **Mutual information:** Mutual information of Lorenz84 system for delays of up to 35 days of system time. The first local minimum is detected at the delay of about 4 days, indicating an optimal delay for the embedding of the timeseries.

The time delay τ is chosen according to the *mutual information*

$$I(X_t; X_{t-\tau}) = \sum_{x_t \in X} \sum_{x_{t-\tau} \in X} p(x_t, x_{t-\tau}) \log \left(\frac{p(x_t, x_{t-\tau})}{p_1(x_t)\, p_2(x_{t-\tau})} \right)$$

Fig. C.1 shows an example of a mutual information calculated for the time series of the Lorenz 84 system sampled at 8 hours. In this particular case τ is about 4.3 days. We note that the lag parameter is a function of the sampling rate f_s. The Fig. C.2 shows how τ changes with f_s.

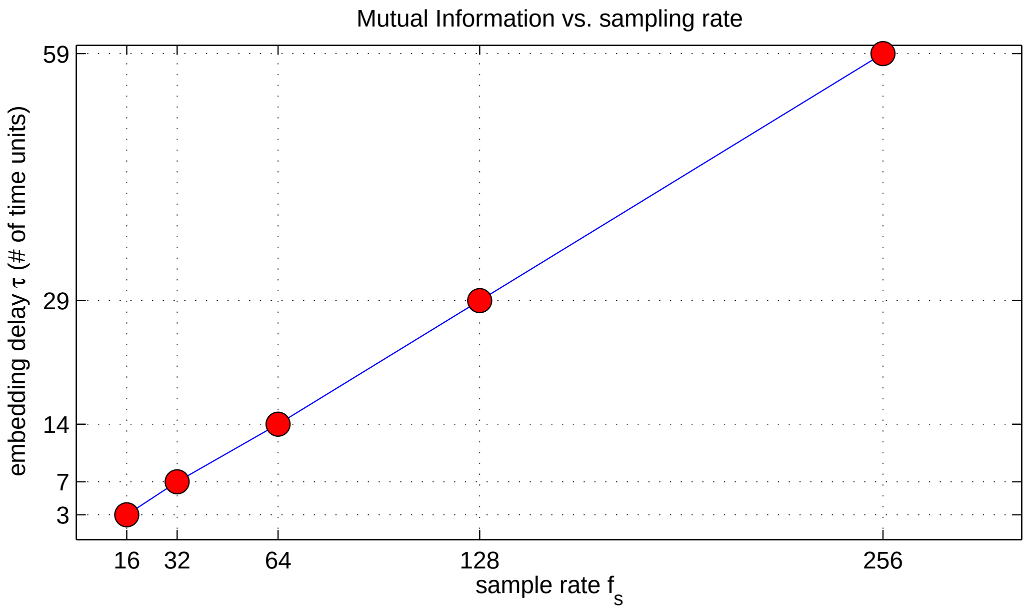

Figure C.2: **Mutual information v. sample rate:** This figure demonstrates how the embedding optimal delay depends on the sampling rate of the system. The relationship is linear as sampling at higher rates requires longer delays.

There are a number of ways to determine the embedding dimension [35, 131], with the most influential being the correlation dimension suggested by [51]. We choose a more direct approach and estimate the embedding dimension d by minimizing Ignorance of the out-of-sample leadtime 1 forecast of the core forecasting model (See section 6.1). This approach yields an optimal estimate for the given forecasting problem. For the Lorenz84 system the dimension is determined as $d = 7$ while for the Lorenz63 it is $d = 3$.

As noted above, when determining the parameter vectors of models of Sections 6.1, 6.2 and 6.3 we work with the embedded time series. A datapoint of an embedded time series is given as

$$x_t^d = \{x_t, x_{t-\tau}, x_{t-2\tau}, \ldots, x_{t-d\tau}\} \tag{C.1}$$

where $x_t^d \in \mathcal{R}$ is a point in a d-dimensional real space. In the following, we drop the superscript d so when we consider d-dimensional datapoint we understand a datapoint of the embedded timeseries. Also, we intentionally avoid the vector notation here; thinking in terms of d-dimensional points greatly simplifies the complexity of the notation.

C.4 RBF approximation as the core model

For the later analyses, we require a model for which we can easily manipulate the forecasting performance. The RBF approach provides such a functionality since its performance is largely influenced by the number of centers deployed. More centers provide a better description of the function being estimated, hence increasing the number of centers should result in improved performance. For this reason, the RBF model will be the core model of our choice, and its forecasting performance will be regulated by changing the number of centers.

In the above analyses we deploy large datasets containing up to $N = 2^{20} \sim \mathcal{O}(10^6)$ observations. For such a large number of observations handling a square matrix A becomes intractable given our computational platform (see Section 5.1.2). To work around the problem we define the core model to be an RBF approximation. The centers will be selected from the datapoints so that $\mathbf{c} \subset \mathbf{x}$ and $M \ll N$. The method used to select the centers will be the power function method of Section 5.2.5.

We also note that the core model has the same setting across related analyses to ensure comparability of the evaluated forecasts.

C.5 Model specification

In the following we call the core model Φ. As noted above, we choose $M << N$, in particular, the number of centers considered is $K \in \{4, 8, 16, 32, 64, 128\}$. The core model takes a general form of

$$\Phi = \sum_{k=1}^{M} \lambda_k \phi(\|\mathbf{x} - c_k\|) \tag{C.2}$$

with $\|\mathbf{x} - c_k\|_2$ being the $N \times M$ matrix of euclidean distances of the points $\mathbf{x}$ from the M centers $\mathbf{c}$. The particular formulation for core models used in the analyses below is

$$\underbrace{\begin{bmatrix} \phi\|x_t - c_1\| & \phi\|x_1 - c_2\| & \dots & \phi\|x_t - c_M\| \\ \phi\|x_{t+1} - c_1\| & \phi\|x_{t+1} - c_2\| & \dots & \phi\|x_{t+1} - c_M\| \\ \vdots & \vdots & \ddots & \vdots \\ \phi\|x_{T-1} - c_1\| & \phi\|x_{T-1} - c_2\| & \dots & \phi\|x_{T-1} - c_M\| \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_M \end{bmatrix}}_{\boldsymbol{\lambda}} = \underbrace{\begin{bmatrix} y_{t+1} \\ y_{t+2} \\ \vdots \\ y_T \end{bmatrix}}_{\mathbf{y}} \tag{C.3}$$

where x_t, is an embedded d-dimensional datapoint at time t and c_k is a d-dimensional center. The datapoints contained in the matrix A represent embedded delayed observations. The vector $\mathbf{y}$ consists of 'current' non-embedded observations of the variable of interest.

C.5.1 Constructing Ψ

We construct 2 types of Ψ correctors, least squares (LSQ) correctors and RBF correctors. Both correctors work with delayed vectors in order to determine the model parameters. For the purposes of fitting and forecasting, the time series are embedded as described in Section C.3.

The LSQ corrector, Ψ_{LSQ}, is simply a linear model

$$\mathbf{y} = A\beta \tag{C.4}$$

where $\mathbf{y} = \{\varepsilon_{t+1}, \varepsilon_{t+2}, \ldots, \varepsilon_T\}$ holds the forecasting residuals of the state variable x, and $A = \{x_t, x_{t+1}, \ldots, x_{T-1}\}$ holds the delayed, embedded d-dimensional datapoints. The parameter vector β contains d regression parameters determined via least squares.

The RBF corrector is a radial basis function model of the form of Eq: C.3, with x_t being delayed d-dimensional datapoints, c_k, $k = \{1, \ldots, M\}$, being d-dimensional centers selected from the datapoints and y_{t+1} is the forecasting error of the base model Φ. There are M parameters λ_k to be determined as described in Section 5.1.

The construction of each corrector Ψ_{RBF} (or Ψ_{LSQ}) proceeds in the following stages. For each leadtime:

(1) Generate out-of-sample leadtime L core model (Φ) forecasts.

(2) Collect the out-of-sample forecasting residuals.

(3) Determine the parameters of the $\Psi_{RBF/LSQ}$ model.

C.6 Out-of-sample evaluation

The performance of the Φ model ensemble forecasts will be evaluated over a testing set, i.e. out-of-sample using Ignorance. To achieve comparability among different forecasting approaches all the models involved will be evaluated under the same setting, i.e. over the same dataset, leadtimes, ensemble size and initial conditions.

Appendix D

Computational considerations of RBF and $\Psi\Phi$/PC

As discussed in Section 5.1.2, for large datasets and given a constrained computational resource, the matrix A of Eq: 5.17 becomes intractable; the RBF approximation must be deployed, i.e. less centers than datapoints $M \ll N$ must be selected. In this section we attempt to answer the question of how many centers we can afford given a limited computational resource. Two questions arise in this regard:

(1) What is a 'large' dataset, and

(2) How many centers are affordable?

We study the problem by looking at the least squares solution (LS) of the (potentially) overdetermined linear problem 5.17. The most intensive part of the solution is calculating the inverse or pseudo-inverse [106] of the matrix A. For the overdetermined case, i.e. the RBF approximation, the solution of the linear system is found via least squares (LS). The calculations are performed in Matlab which uses the linear algebra package (LAPACK) [2] on a PC with 2 Intel(R) Core(TM)2 CPU at 2.4 GHz with 8GB of memory.

To express computational intensity of a given problem we use two measures: the *CPU time* (CPU) and the *floating point operations per second* (FLOPS) [21, 90]. In terms of FLOPS, calculating the LS problem [21] with a matrix of size $m \times n$ takes

$$\text{FLOPS} = 4mn^2 + 8n^3 \tag{D.1}$$

D.1 The cost RBF interpolation

When estimating the RBF parameters of the core model the entire training set is used. The datasets used in the above analyses are rather large, with some containing up to $N = 10^6$ data points. If we were to use all the datapoints as centers, the solution would involve an inversion of a $10^6 \times 10^6$ square matrix. How costly would such an inversion be?

In Fig: D.1 we plot CPU time (left vertical axis) and number of FLOPS (right vertical axis) against datasets of varying size. In this example the largest dataset contains 10,000 datapoints. The theoretical FLOPs of Eq. D.1 (green line) are consistent with the actual CPU time (blue line) of the LS solution for a given size of a dataset. Both measures display an accelerated growth. For sizes up to $7,000 \times 7,000$ an additional increase in the size does not dramatically change the CPU time. However, increasing the size from, say, $9,000 \times 9,000$ to $10,000 \times 10,000$ the CPU time starts increasing rather significantly.

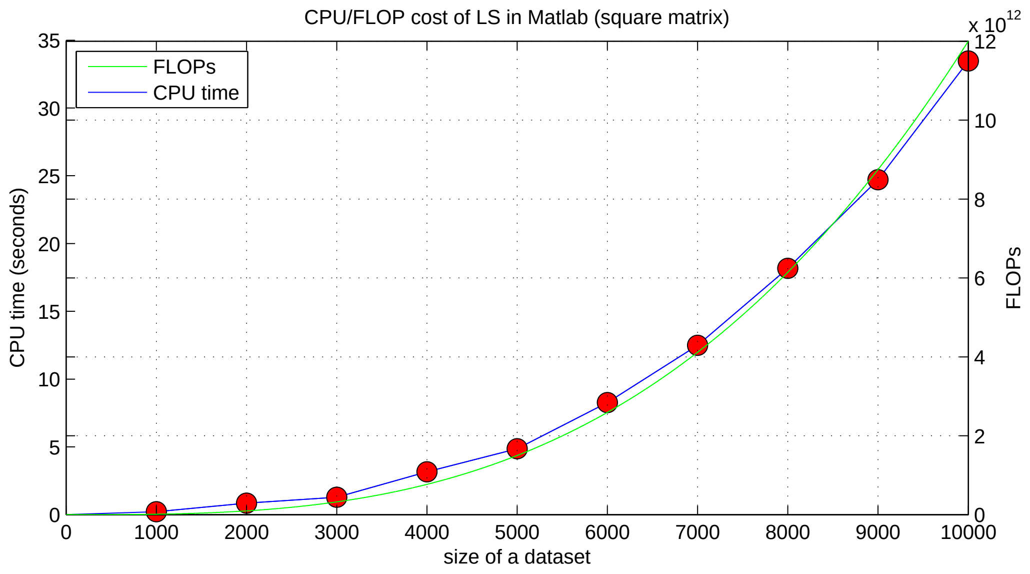

Figure D.1: **Computational intensity of least squares.** This figure has been used in Section 5.1.2. For a square matrix of size $M \times M$ we plot the CPU time (blue) and number of FLOPs (green) required to calculate a least square solution for different values of M. While it takes about 5 seconds to calculate LS solution for matrix with $M = 5 \times 10^3$, it takes about 35 seconds when the size is doubled. The computational cost follows a power law.

In the left panel of Fig: D.2 we show a log-log version of the Fig: D.1. The graph suggests an exponential growth of the computational intensity of the LS solution. We use the linear relationship of the log values and extrapolate CPU time for larger datasets, shown in the right panel. To solve a single RBF interpolation problem for dataset sizes on the order of those used throughout our work would take up to 10 days.

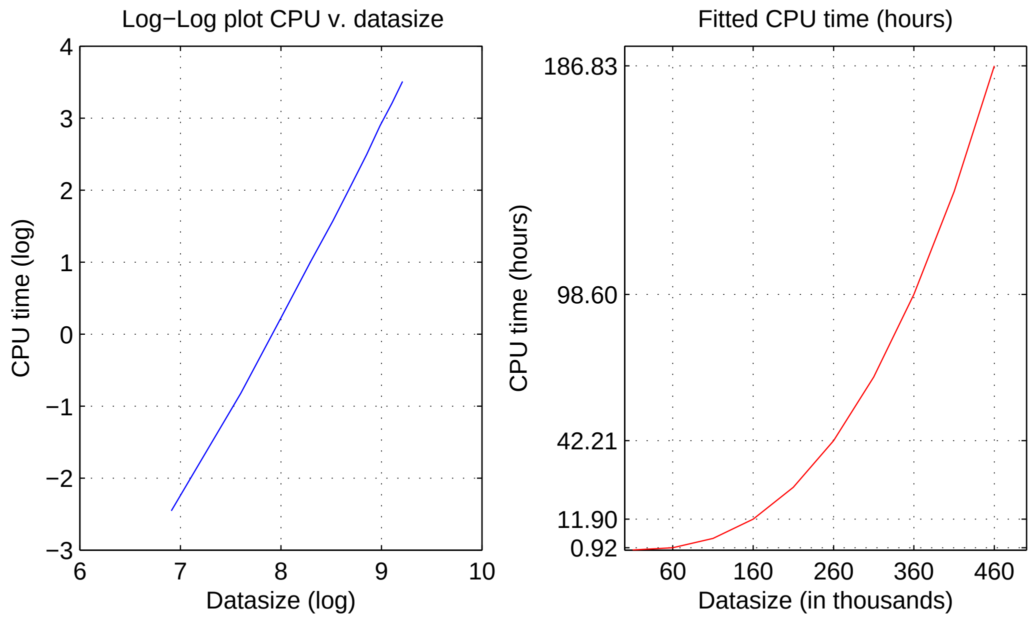

Figure D.2: **Extrapolation of computational cost.** Similar to Fig D.1 but for extrapolated values. Solving an exactly determined system for datasets used in our work would take about 10 days. The values were extrapolated using the linear relationship between the log-log values plotted in the left panel.

D.2 How many centers are affordable?

Considering the finding of the previous section, the question is: how many centers can we afford? To answer the question we consider a dataset size of $N = 400,000$ and vary the number of centers, changing the size of the matrix A. Fig: D.3 shows that to solve a linear system with 128 centers (matrix size $400,000 \times 128$) takes about 24.5 seconds. Extrapolating for a larger number of centers, Fig: D.4, we find that the solution takes about 10 minutes when 1,024 centers are used, and about 1.3 hours when 4,096 centers are used.

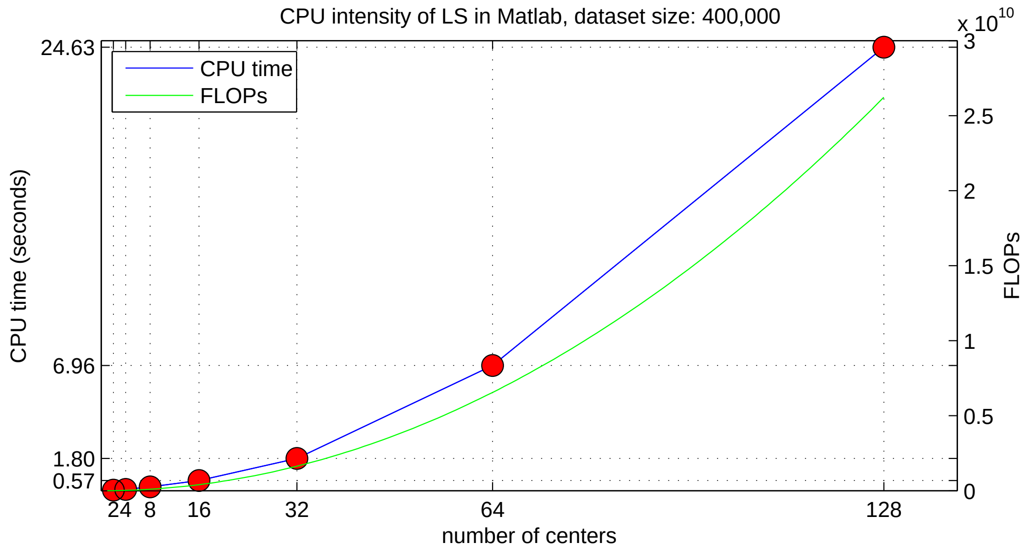

Figure D.3: **Impact of number of centers:** Similar to Fig: D.1 but for an overdetermined system where $N = 4 \times 10^5$ and varying number of centers M. Solving overdetermined system for the dataset size of $N = 4 \times 10^5$ and 128 centers takes up to 25 seconds, a moderate computational cost.

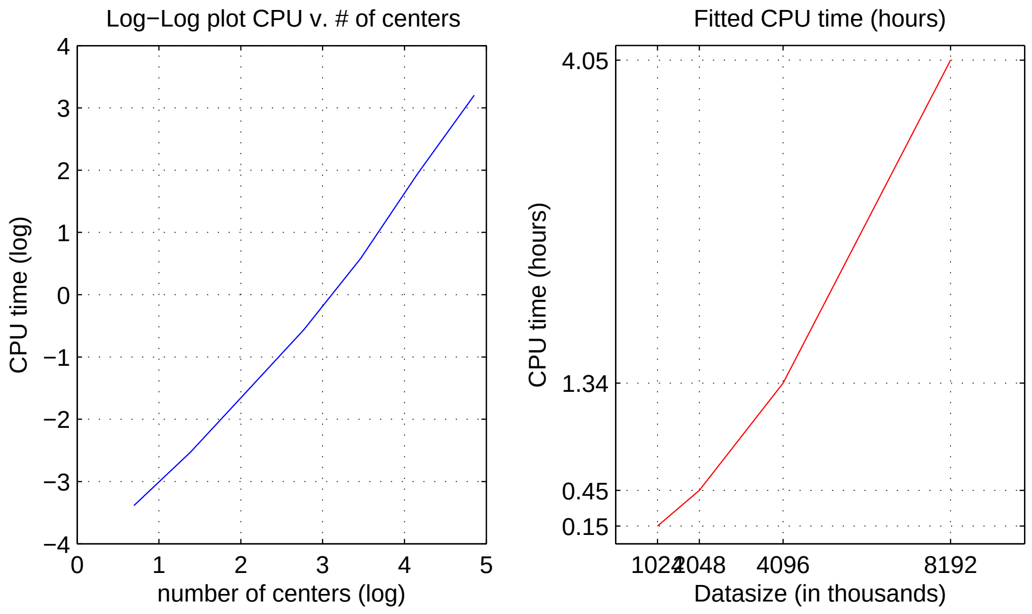

Figure D.4: **Cost for large number of centers.** We use similar extrapolation as in Fig D.2. To solve a system for a dataset of size $N = 4 \times 10^5$ and $M = 8,192$ centers takes up to 4 hours of CPU time on our platform. Beyond 1,024 centers the problem becomes intractable when many iterative solutions of the linear system are required.

Bibliography

[1] H. Abarbane, S. Koonin, and G.and Rothaus O. Levine, H.and MacDonald. Statistics of extreme events with application to climate. *JASON JSR-90-30S*, 1992.

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK User's Guide, Third Edition*. SIAM, 1999.

[3] H. L. Anderson. Metropolis, monte carlo and maniac. *Los Alamos Science*, 14:96–108, 1986.

[4] H. M. Anderson, J. L. van den Dool. Skill and return of skill in dynamic extended-range forecasts. *Mon. Weather Rev.*, 122:507–516, 1994.

[5] S. J. Armstrong. *Principles of forecasting: A handbook for researchers and practitioners.* Kluwer Academic Publishers, 2001.

[6] E. Atkinson, K. *An Introduction to Numerical Analysis.* Wiley, 1989.

[7] G. L. Baker and J. P.tate estimation Gollub. Chaotic dynamics: An introduction. 1996.

[8] G.L. Baker and J.A. Blackburn. *The Pendulum: a case study in physics.* Oxford University Press, 2009.

[9] C. Bishop. *Pattern Recognition and Machine Learning.* Springer, Berlin, 2006.

[10] M. Blix and P. Sellin. *Uncertainty Bands for Inflation Forecasts.* Working paper. Sveriges Riksbank, 1998.

[11] A. Bowman. A comparative study of some kernel-based nonparametric density estimators. *Journal of Statistical Computation and Simulation*, 21:313–327, 1985.

[12] P. Box, E. and N. R. Draper. *Empirical model-building and response surfaces.* Wiley, 1987.

[13] E. Britton, P. Fisher, and J. Whitley. The inflation report projections: Understanding the fan chart. *Bank of England Quarterly Bulletin*, 1998.

[14] J. Bröcker and L. A. Smith. Scoring probabilistic forecasts: The importance of being proper. *Weather and Forecasting*, 8, 2006.

[15] J. Bröcker and L. A. Smith. From ensemble forecasts to predictive distribution functions. *Tellus*, 4:663–678, 2008.

[16] H. Broer, R. Vitolo, and C. Simo. Bifurcations and strange attractors in the lorenz-84 climate model with seasonal forcing. *Nonlinearity*, 15:1205–1267, 2002.

[17] J. C. Butcher. *The numerical analysis of ordinary differential equations, Runge-Kutta and general linear methods.* Wiley, 1987.

[18] M. Chekroun, E. Simonnet, and M. Ghil. Stochastic climate dynamics: Random attractors and time-dependent invariant measures. *Physica D, preprint*, 2010b.

[19] E. W. Cheney and D. R. Kincaid. *Numerical mathematics and computing*. Cengage Learning, 2007.

[20] S. T. Chiu. An automatic bandwidth selector for kernel density estimation. *Biometrika*, 79:771–782, 1992.

[21] H. Colub, G. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[22] T. M. Cover and J. A. Thomas. *Elements of Information Theory 2nd Edition*. Wiley, second edition, 2006.

[23] X. T. Cui, D. J. Parker, and A. P. Morse. The drying out of soil moisture following rainfall in a numerical weather prediction model, and implications for malaria prediction in west africa. *Weather and Forecasting*, 24:1549–1557, 2009.

[24] B. V. Dasarathy. *Nearest neighbor pattern classification techniques*. IEEE Computer Society Press, 1990.

[25] S. De Marchi. On optimal center locations for radial basis interpolation: computational aspects. *Rend. Sem. Mat. Torino*, 61(3), 2003.

[26] S. De Marchi, R. Schaback, and H. Wendland. Near-optimal data-independent point locations for radial basis function interpolation. *Adv. Comput. Math.*, 23(3), 2003.

[27] L. Devroye. *Non-Uniform Random Variate Generation*. New York: Springer-Verlag, 1986.

[28] F. J. Doblas-Reyes and et al. Decadal hindcasts with an initialised coupled atmosphere-ocean model: First results from the ecmwf ensembles contribution. *EMS 2008, Amsterdam*, 2008.

[29] F. J. Doblas-Reyes and et al. Addressing model uncertainty in seasonal and annual dynamical ensemble forecasts. *Q. J. R. Meteorol. Soc.*, 135:1538–1559, 2009.

[30] F. J. Doblas-Reyes, R. Hagedorn, and T. N. Palmer. The rationale behind the success of multi-model ensembles in seasonal forecasting. part II: Calibration and combination. *Tellus, Ser. A*.

[31] F. J. Doblas-Reyes, A. Weisheimer, M. Déqué, N. Keenlyside, M. McVean, Murphy J. M., P. Rogel, D. Smithd, and T. N. Palmer. Addressing model uncertainty in seasonal and annual dynamical ensemble forecasts. *Q. J. R. Meteorol. Soc.*, 135:1538–1559, 2009.

[32] N. Draper and H. Smith. *Applied Regression Analysis*. Wiley, 1998.

[33] H. Du. *Combining Statistical Methods with Dynamical Insight to Improve Nonlinear Estimation*. PhD thesis, London School of Economics, 2009.

[34] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley, New York, 2 edition, 2001.

[35] I. Dvorak and J. Klaschka. Modification of the grassberger-procaccia algorithm for estimating the correlation exponent of chaotic systems with high embedding dimension. *Physics letters A*, 145:225–231, 1989.

[36] ENSEMBLES. Ensemles project website. http://www.ensembles-eu.org/.

[37] V. A. Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability and its Applications*, 14:153–158, 1969.

[38] J. D. Farmer and J. J. Sidorowich. Exploiting chaos to predict the future and reduce noise. *In: Y.C. Lee (Ed.), Evolution, Learning and Cognition*, 127, 1988.

[39] G. Fasshauer. *Meshfree Approximation Methods with MATLAB*. World Scientific Publishers, 2007.

[40] A. J. M. Ferreira, E. J. Kansa, G. E. Fasshauer, and V. M. A. (Eds.) Leitao. *Progress on Meshless Methods*. Springer, 2009.

[41] B. Fornberg, T. A. Driscolli, G. B. Wright, and R. Charles. Observations on the behaviour of radial basis functions near boundaries. *Comput. Math. Appl.*, (43):473–490, 2002.

[42] R. Franke. A critical comparison of some methods for interpolation of scattered data. *Naval Postgraduate School Tech.Rep.*, 1979.

[43] R. Franke. Scattered data interpolation: tests of some methods. *Math. Comp.*, (48):181–200, 1982.

[44] R. Franke. Thin plate splines with tension. *Comput. Aided Geom. Design*, (2):87–95, 1985.

[45] J. Franta, M. adn Barunik, R. Horvath, and K. Smidkova. Are bayesian fan charts useful for central banks? uncertainty, forecasting, and financial stability stress tests. *(Working Paper), Czech National Bank*, 2011.

[46] A. M. Fraser and H. L. Swinney. Independent coordinates for strange attractors from mutual information. *Math. Comp.*, (33):1134–1140, 1986.

[47] J. G. Freire, C. Bonatto, and C. C. DaCamarra. Multistability, phase diagrams, and intransitivity in the lorenz-84 low-rder atmospheric circulation model. *Chaos*, 18:1705–1717, 2008.

[48] T. Gneiting, A. Raftery, A. H. Westveld, and T. Goldman. Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation. *Monthly Weather Review*, 133, 2004.

[49] I. J. Good. Rational decisions. *Journal of the Royal Statistical Society*, XIV(1):107–114, 1952.

[50] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F, Radar and Signal Processing*, 140:107–113, 1993.

[51] P. Grassberger and I. Procaccia. Estimation of the kolmogorov entropy from a chaotic signal. *Physical Review A*, 28(4):2591–2593, 1983.

[52] P. Hall, J. S. Marron, and B. U. Park. Smoothed cross-validation. *Probability Theory and Related Fields*, 92:1–20, 1992.

[53] T. M. Hamill and J. Juras. Measuring forecast skill: is it real or is it the varying climatology? *Q. J. R. Meteorol. Soc.*, 132:2905–2923, 2005.

[54] T. M. Hamill, J. S. Whitaker, and X. Wei. Smoothed cross-validation. *Monthly Weather Review*, 132:1434–1447, 2004.

[55] L. Hardy, R. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.*, (76):1905 – 1915, 1971.

[56] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[57] C. D. Hewitt. The ensembles project: Providing ensemble-based predictions of climate changes and their impacts. *EGGS newsletter*, 13:22–25, 2005.

[58] S. Jewson. Comparing the ensemble mean and the ensemble standard deviation as inputs for probabilistic medium range temperature forecasts. *arXiv:physics*, 2004.

[59] K. Judd. Nonlinear state estimation, indistinguishable states, and the extended kalman filter. *PD*, 183(3–4):273–281, May 2003.

[60] K. Judd, C. Reynolds, and T. Rosmond. Toward shadowing in operational weather prediction. Technical report, Naval Research Laboratory, 2004. NRL/MR/7530-04018.

[61] K. Judd, C. A. Reynolds, T. E. Rosmond, and L. A. Smith. The geometry of model error. *JAS*, 65:1749–1772, 2008.

[62] K. Judd and M. Small. On selecting models for nonlinear time series. *Physica D*, (82):426–444, 1995.

[63] K. Judd and M. Small. Towards long-term prediction. *Physica D*, (136):31–44, 1999.

[64] K. Judd, L. Smith, and A. Weisheimer. Gradient free descent: shadowing, and state estimation using limited derivative information. *PD*, 190(3–4):153–166, April 2004.

[65] K. Judd and L. A. Smith. Indistinguishable states I. perfect model scenario. *Physica D*, 151:2–4, 2001.

[66] K. Judd and L. A. Smith. Indistinguishable states II: the imperfect model scenario. *Physica D: nonlinear phenomena*, 196:224–242, 2004.

[67] K. Judd and T. Stemler. Forecasting: It's not about statistics, it's about dynamics. *Phil. Trans. of Royal Soc.*, 2009.

[68] J. Juras. Comments on probabilistic predictions of precipitation using the ecmwf ensemble prediction system. *Weather and Forecasting*, 15:365–366, 2000.

[69] E. Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, 2007.

[70] M. H. Kalos and P. A. Whitlock. *Monte Carlo Methods*. Wiley-VCH, 2008.

[71] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, 1997.

[72] J. L. Kelly. A new interpretation of information rate. *Bell System Technical Journal*, 35:917–926, 1956.

[73] T. N. Krishnamurti, C. M. Kishtawal, D. R. LaRow, T. E.and Bachiochi, Z. Zhang, S. Williford, E. C. Gadgil, and S. Surendran. Improved weather and seasonal climate forecasts from multi-model superensemble. *Science*, 22(285):1548–1550, 1999.

[74] Leibler R. A. Kullback, S. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79 – 86, 1951.

[75] S. Kullback. Letter to the editor: The kullback-leibler distance. *The American Statistician*, 41(4):340 – 341, 1987.

[76] W. Kutta. Beitrag zur naherungsweisen integration totaler differentialgleichungen. *Z. Math. Phys.*, 46:435–453, 1901.

[77] L. Y. Leung and G. R. North. Information theory and climate prediction. *Journal of Climate*, 3:5–14, 1990.

[78] M. Leutbecher and T. N. Palmer. Ensemble forecasting. *Journal of Computational Physics*, 227, 2007.

[79] T. Y. Li and J. A. Yorke. Period three implies chaos. *American Mathematical Monthly*, 82:985–992, 1975.

[80] C. Loader. *Local Regression and Likelihood.* Springer, 1999.

[81] E. N. Lorenz. Deterministic nonperiodic flow. *JAS*, 20:130–141, 1963.

[82] E. N. Lorenz. Atmospheric predictability as revealed by naturally occuring analogues. *J. Atmos. Sci.*, 26:636–646, 1969.

[83] E. N. Lorenz. Irregularity: a fundamental property of the atmosphere. *Tellus*, 36A:98–110, 1984b.

[84] E. N. Lorenz. Can chaos and intransitivity lead to interannual variability? *Tellus*, 42A:378–389, 1990.

[85] Uppala S. M., Køallberg P.W., Simmons A.J., Andrae U., Da Costa B.V., Fiorino M., Gibson J.K., Haseler J., Hernandez A., Kelly G.A., Li X., Onogi K., Saarinen S., Sokka N., Allan R.P., Andersson E., Arpe K., Balmaseda M.A., Beljaars A.C.M., van deBerg L., Bidlot J., Bormann N., Caires S., Chevallier F., Dethof A., Dragosavac M., Fisher M., Fuentes M., Hagemann S., Holm E., Hoskins B.J., Isaksen L., Janssen P.A.E.M., Jenne R., McNally A.P., Mahfouf J.F., Morcrette J.J., Rayner N.A., Saunders R.W., Simon P., Sterl A., Trenberth K.E., Untch A., Vasiljevic D., Viterbo P., and Woollen J. The era-40 reanalysis. *Q. J. R. Meteorol. Soc.*, 131:2961–3012, 2005.

[86] H. MacKay. *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, 2003.

[87] R. Madden. A quantitative approach to long-range prediction. *J. Geophys. Res.*, 86:9817–9825, 1981.

[88] J. C. Mairhuber. On haar's theorem concerning chebyshev approximation problems having unique solutions. *Proc. Am. Math. Soc.*, 7:609–615, 1956.

[89] I. B. Mason. Dependence of the critical success index on sample climate and threshold probability. *Aust. Meteorol. Mag.*, 37:75–81, 1989.

[90] J. Matlis. Sidebar: The linpack benchmark. *Computer World*, May 2005.

[91] M. R. May. Simple mathematical models with very complicated dynamics. *Nature*, 261:459–467, 1976.

[92] K. McGuffie and A. Henderson-Sellers. *A climate modelling primer.* Wiley, 2005.

[93] F. Molteni, R. Buizza, T. N. Palmer, and T. Petroliagis. Ecmwf ensemble prediction system: Methodology and validation. *Quarterly Journal of the Royal Meteorological Society*, 122:73–119, 1976.

[94] A. P. Morse, F. J. Doblas-Reyes, M. B. Hoshen, R. Hagedorn, and T. N. Palmer. A forecast quality assessment of an end-to-end probabilistic multi-model seasonal forecast system using a malaria model. *Tellus*, 57A:464–475, 2004.

[95] Allan H. Murphy. The early history of probability forecasts: Some extensions and clarifications. *Weather and Forecasting*, (13):5–15, 1998.

[96] S. Y. Novak. *Extreme value methods with applications to finance.* Chapman Hall,CRC Press, London, 2011.

[97] Intergovernmental Panel on Climate Change (IPCC). Climate change 2007. *Cambridge Univ. Press*, 2007.

[98] T. N. Palmer. Predicting uncertainty in forecasts of weather and climate. *Rep. Prog. Phys*, 63, 2000.

[99] T. N. Palmer. A nonlinear dynamical perspective on model error: A proposal for non-local stochastic-dynamic parametrization in weather and climate prediction models. *Q. J. R. Meteorol. Soc.*, 127, 2001.

[100] T. N. Palmer, F. J. Doblas-Reyes, R. Hagedorn, and A. Weisheimer. Probabilistic prediction of climate using multi-model ensembles: from basics to applications. *Phil. Trans. R. Soc. B*, 10, 2009.

[101] T. N. Palmer, F. J. Doblas-Reyes, A. Weisheimer, and M. Rodwell. Seasonal forecast datasets: A resource for calibrating regional climate change projections? *CLIVAR Exchanges*, 43:6–7, 2007.

[102] T. N. Palmer, F. J. Doblas-Reyes, A. Weisheimer, and M. Rodwell. Towards seamless prediction: Calibration of climate-change projections using seasonal forecasts. *American Meteorological Society*, 89:459–470, 2008.

[103] T. N. Palmer, G. J. Shutts, R. Hagedorn, F. J. Doblas-Reyes, T. Jung, and M. Leutbecher. Representing model uncertainty in weather and climate prediction. *Annual Review of Earth and Planetary Sciences*, 33, 2005.

[104] T. N. et al. Palmer. Development of a european multimodel ensemble system for seasonal-to-interannual prediction (demeter). *Bull. Am. Meteorol. Soc.*, 85:853–872, 2004.

[105] E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[106] R. Penrose. A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society*, 51:406–413, 1955.

[107] Picard, Richard, Cook, and Dennis. Cross-validation of regression models. *Journal of the American Statistical Association*, 79:575–583, 1984.

[108] M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94:590–591, 1999.

[109] M. J. D. Powell. Radial basis functions for multivariate interpolation: a review. *in Proc. IMA Conf. on Algorithms for the Approximation of Functions and Data (RMCS Shrivenham, 1985)*, 79:575–583, 1985.

[110] A. Raftery, F. Balabdaoui, T. Gneiting, and M. Polakowski. Using bayesian model averaging to calibrate forecast ensembles. *Monthly Weather Review*, 133:1155–1174, 2005.

[111] P. L. Read, M. J. Bell, D. W. Johnson, and R. M. Small. Quasi-periodic and chaotic flow regimes in a thermally driven, rotating fluid annulus. *Journal of Fluid Mechanics*, 238:599–632, 1992.

[112] D. S. Richardson. Skill and relative economic value of the ecmwf ensemble prediction system. *Quart. J. Roy. Meteor. Soc.*, 126:649–667, 2000.

[113] D. S. Richardson. Measures of skill and value of ensemble prediction systems, their interrelationship and the effect of ensemble size. *Quart. J. Roy. Meteor. Soc.*, 127:2473–2489, 2001.

[114] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *Annals of Mathematical Statistics*, 27:832–837, 1956.

[115] M. S. Roulston and L. A. Smith. Evaluating probabilistic forecasts using information theory. *Monthly Weather Review*, 130:1653–1660, 2002.

[116] M. S. Roulston and L. A. Smith. Combining dynamical and statistical ensembles. *Tellus*, 55A:16–30, 2003.

[117] T. Sauer, J. A. Yorke, and M. Casdagli. Embedology. *Journal of Statistical Physics*, 65:579–616, 1991.

[118] R. Schaback. *Adaptive greedy techniques for approximate solution of large RBF systems.* 1997. *num.math.uni-goettingen.de/schaback/teaching/texte/rbfbook.ps.*

[119] R. Schaback. *Reconstruction of Multivariate Functions from Scattered Data.* 1997. *http://www.num.math.uni-goettingen.de/schaback/research/papers/rbfbook.ps.*

[120] W. Scott, D. *Multivariate Density Estimation. Theory, Practice and Visualization.* Wiley, 1992.

[121] G.A.F. Seber. *Multivariate Observations.* John Wiley and Sons, Inc., 1984.

[122] L. F. Shampine. *Numerical Solution of Ordinary Differential Equations.* Chapman and Hall, New York, 1994.

[123] Lawrence F. Shampine and Herman A. Watts. The art of writing a runge-kutta code. ii. *Applied Mathematics and Computation*, 5(2):93 – 121, 1979.

[124] C. E. Shannon. A mathematical theory of communication. *Proc. Institute of Radio Engineers*, 27(1):379–423, 1948.

[125] C. E. Shannon. Communication in the presence of noise. *Proc. Institute of Radio Engineers*, 37(1):10 – 21, 1949.

[126] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30:50–64, 1951.

[127] D. Shepard. A two dimensional interpolation function for irregularly spaced data. *Proc. 23rd Nat. Conf. ACM*, pages 517–524, 1968.

[128] L. Shukla. Dynamical predictability of monthly means.. *J. Atmos. Sci.*, 32:2547–2572, 1981.

[129] B.W. Silverman. *Density Estimation.* London: Chapman and Hall, 1986.

[130] L. A. Smith. Identification and prediction of low dimensional dynamics. *Physica D*, 58:50–76, 1992.

[131] L. A. Smith. The maintenance of uncertainty. *In Proc. International School of Physics "Enrico Fermi", Course CXXXIII*, pages 177–246, 2003.

[132] L. A. Smith, C. Ziehman, and K. Fraedrich. Uncertainty dynamics and predictability in chaotic systems. *Quart. J. Royal Meteorological Soc.*, 125:2855–2886, 1997.

[133] Leonard A. Smith. Disentangling uncertainty and error: On the predictability of nonlinear systems. In Alistair I. Mees, editor, *Nonlinear Dynamics and Statistics*, chapter 2, pages 31–64. Birkhäuser Boston, 2000.

[134] D. J. Spiegelhalter, N. G. Best, B. P. Carlin, and A. van der Linde. Bayesian measures of model complexity and fit. *J. R. Statist. Soc. B, Series B*, 64:583–639, 2002.

[135] J. C. Sprott. *Chaos and Time-Series Analysis*. 2003.

[136] G. Steinbrecher and W. T. Shaw. Quantile mechanics. *European Journal of Applied Mathematics*, 19:87–112, 2008.

[137] T. Stemler and K. Judd. A guide to using shadowing filters for forecasting and state estimation. *Physica D*, 238(14):1260–1273, 2009.

[138] F. Takens. Detecting strange attractors in turbulence. *Dynamical Systems and Turbulence*, 898:366 – 381, 1981.

[139] O. Talagrand and P. Courties. Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory. *Quart. J. Roy. Meteor. Soc.*, 113:1311–1328, 1987.

[140] J. Teixeira, C. A. Reynolds, and K. Judd. Time step sensitivity of nonlinear atmospheric models: numerical convergence, truncation error growth, and ensemble design. *J. Atmos. Sci.*, 64:175–189, 2007.

[141] I. Tetko, D. J. Livingstone, and A. I. Luik. Neural network studies. 1. comparison of overfitting and overtraining. *J. Chem. Inf. Comput. Sci.*, 113:826–833, 1995.

[142] J. N. Thepaut and P. Courties. Four-dimensional data assimilation using the adjoint of a multi-level primitive-equation model. *Quart. J. Roy. Meteor. Soc.*, 117:1225–1254, 1991.

[143] Zoltan Toth and Eugenia Kalnay. Ensemble Forecasting at NMC: The Generation of Perturbations. *Bulletin of the American Meteorological Society*, 74(12):2317–2330, 1993.

[144] K. E. Treberth. The definition of el niño. *American Meteor. Soc.*, 78:2771–2777, 1997.

[145] L. N. Trefethen. *Spectral Methods in Matlab.* SIAM, 2000.

[146] K. E. Trenberth. Evolution of El Niño Southern Oscillation and global atmospheric surface temperatures. *Journal of Geophysical Research*, 107, 2002.

[147] K. E. Trenberth and J. Timoth. The 19900-1995 El Niño Southern Oscillation Event: Longest on record. *Geophysical Research Letters*, 23:57–60, 1996.

[148] H. J. Van den Dool. *Empirical Methods in Short-Term Climate Prediction.* Oxford University Press, 2007.

[149] H. J. Van den Dool, J. Huang, and Y. Fan. Performance and analysis of the constructed analogue method applied to us soil moisture over 1981-2001. *J. Geophys. Res.*, 108(D16), 2003.

[150] J. von Neumann and O. Morgenstern. *Theory of games and economic behaviour.* Princeton Univ. Press, 1947.

[151] A. P. Weigel, M. A. Liniger, and C. Appenzeller. Can multi-model combination really enhance the prediction skill of probabilistic ensemble forecasts? *Quaterly Journal of the Royal Meteorological Society*, 134:241–260, 2008.

[152] A. Weisheimer, F. J. Doblas-Reyes, T. N Palmer, A. Alessandri, A. Arribas, M. Déqué, N. Keenlyside, M. MacVean, A. Navarra, and P. Rogel. Ensembles: A new multi-model ensemble for seasonal-to-annual predictions - skill and progress beyond demeter in forecasting tropical pacific ssts. *Geophysical Research Letters*, 36, 2009.

[153] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Adva. in Comput. Math.*, 2:389–396, 1995.

[154] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005.

[155] Wikipedia. Dynamical system. http://en.wikipedia.org.

[156] D. S. Wilks. Smoothing forecast ensembles with fitted probability distributions. *Quaterly Journal of the Royal Meteorological Society*, 128:2821–2836, 2002.

[157] D. S. Wilks. Comparison of ensemble-mos methods in the lorenz'96 setting. *Meteorol. Appl.*, 13:243–256, 2006.

[158] D. S. Wilks. *Statistical Methods in the Atmospheric Sciences*. ELSEVIER, 2011.

[159] Z. Wu and R. Schaback. Local error estimates for radial basis function interpolation of scattered data. *IMA J. Numer. Anal.*, 13:13–27, 1993.

[160] J. F. Yates, L. S. McDaniel, and E. S. Brown. Probabilistic forecasts of stock prices and earnings: The hazards of nascent expertise. *Organizational Behaviour And Human Decision Processes*, 49:60–79, 1991.

[161] R. M. B. Young and P. L. Read. Flow transitions resembling bifurcations of the logistic map in simulations of the baroclinic rotating annulus. *Physica D: Nonlinear Phenomena*, 237(18):2251 – 2262, 2008.

[162] C. Ziehmann, L. A. Smith, and Kurths J. Localized lyapunov exponents and the prediction of predictability. *Phys. Lett. A*, 271 (4):237–251, 2000.