On the Practical Applications of Information Field Dynamics

Martin Dupont

Submitted for the master course in Theoretical and Mathematical Physics at the Ludwig–Maximilians–Universität Munich

Supervisor: Prof. Dr. Torsten Enßlin
Second referee: Prof. Dr. Ewald Müller
Defence date: 1st August 2017

Acknowledgements

First and foremost I am grateful to my friends Anthony and Max for sacrificing their weekend to proofread my thesis cover to cover. Thanks to Anthony, whose helpful commentary and LaTeX skills were invaluable to my thesis. Thanks to Max, whose unhelpful commentary and sarcasm were at least entertaining. Special thanks needs to go to my supervisor Torsten Enßlin for tolerating my pessimism, and my colleague Reimar Leike, for reading my numerous rambling manifestos.

Contents

Chapter 1

Introduction

As physicists, we use mathematical equations to describe the world around us. However, the unfortunate state of the world is that most physically interesting processes have equations of motion described by partial differential equations (PDEs) which have no analytical solutions. This necessitates the use of simulation schemes for differential equations, which can only approximate the behaviour of the true solution. Given a PDE, the solutions are functions, which contain an infinite number of degrees of freedom. Whereas for any implementable approximation, the number of degrees of freedom must be finite. Thus there is always a gap between any simulation and reality.

There is already a vast and well-known literature base of numerical methods for differential equations, with each method having its own advantages and drawbacks. One often-used approach is that of subgrid models. A field will typically be represented by a series of discrete samples of the field value at certain points, and the simulation scheme will generally assume that the field has some structure between those points. For example, subgrid models may often assume a linear interpolation of the field between the data points. This assumption is then applied somewhere in the scheme in the hope of obtaining more accurate results [1].

Information field dynamics (IFD)[2] is a new framework for developing numerical schemes, and can be thought of as an improvement on, or generalization of subgrid models. The main idea of IFD is that rather than making any concrete assumptions about the nature of the field, we use Bayesian statistics to infer the most likely continuous field configuration given some

data in computer memory, and this continuous reconstruction is used to inform the numerical simulation scheme. The continuous field reconstruction is achieved using a framework already developed by T. Enßlin known as Information Field Theory (IFT) [3]. The general mathematical framework of this scheme has indeed already been laid out in [4], but has yet to be practically implemented.

The original goal of this project was to develop the first real-world application of the IFD framework. The problem chosen for this was the simulation of cosmic ray transport in space. It was believed that IFD would be well-suited for this. However, throughout the course of this project, it became apparent that there were many unsolved problems on the general level which needed to be ironed out before a practical implementation could be carried out. Specifically speaking, an analysis of the stability, errors and convergence of codes in the IFD framework had not yet been performed. The early implementations of the cosmic ray codes were plagued by instabilities and numerical artefacts, which necessitated a theoretical analysis before moving forward.

The content of this work is as follows: after an introduction to the general framework of IFT and Bayesian statistics in chapter 2, we will present the construction of the IFD framework in chapter 3. In this chapter we will also present a number of small results and improvements to the framework, before moving on to discuss errors, stability and convergence. We will also introduce the trial problem for this project: cosmic ray simulations. Chapter 4 will then explore a broad class of IFD models whose stability and convergence properties can be analytically examined. In tandem, we will develop a toy model which serves as an illustrative example of the strengths and weaknesses of this general class of models. This example will then be extended to something which at least superficially resembles a cosmic ray evolution simulation.

A second class of models will also be presented in chapter 5, which is also based on IFD, and draws inspiration from so-called *Smooth Particle Hydrodynamics* algorithms. This model is unsuccessful, but draws into sharp focus some of the weaknesses presented by the previous class of models. While any numerical scheme has advantages and disadvantages, it is believed that the weaknesses uncovered during the course of this project will need to be addressed before attempting to simulate a truly nontrivial and scientifically

interesting system. This general weakness, and possible solutions for a way forward will be presented last. This will be followed in chapter 6 by a summary of all the results presented in this work.

Chapter 2

The IFT framework

2.1 Priors, posterior and Bayes Theorem

To understand IFT, one first needs a quick introduction to Bayesian statistics. In Bayesian probability theory, probabilities are assigned to events, and take the form of real numbers. These real numbers express a subjective belief about how likely a given even is to happen. The assigned probabilities for any event range between zero and one, with one implying absolute confidence in a result. The sum of probabilities (or integral) over all possible events within some set must equal one, i.e. *some* event must occur. In Bayesian statistics, a rational observers degree of belief about an event may change in response to new information. This updating of beliefs is the foundation of Bayesian inference, typically one has some prior belief about a system, which is updated to a new belief about the system after performing an experiment and obtaining new information.

The probability of an event a occurring is denoted by $\mathcal{P}(a)$. Typically, probabilities of events are dependent on some background condition. We write $\mathcal{P}(a|b)$ to denote the probability of a occurring given that we know b to be true. The product rule of probabilities states that given two events a and b, the joint probability of them both occurring is given by $\mathcal{P}(ab) = \mathcal{P}(a|b)\mathcal{P}(b)$. Two events are said to be mutually exclusive if they never occur in unison, and a set of events is said to be exhaustive if one element of the set must always occur. A further piece of terminology used in this thesis is that of marginalizing over probabilities; given a probability distribution $\mathcal{P}(a_i, b_j)$ in

two sets of variables $\{a_i\}$ and $\{b_j\}$ and the set of b's are mutually exclusive and exhaustive, then the probability distribution for just the first variable is given by

$$\mathcal{P}(a_i) = \sum_j \mathcal{P}(a_i, b_j) \tag{2.1}$$

In the case where the b's form a continuous variable, the sum will be replaced by an integral.

Some more terms need to be defined as well. Suppose one had a system whose state is governed by some variable θ, and one performs an experiment on the system yielding some data d, from which we want to infer the state of the system. In Bayesian statistics, we will have some preconceived probability on the state of the system $\mathcal{P}(\theta)$ before any measurement is carried out. This is known as the *prior distribution*. This prior may come from a variety of sources, such as past experiments, or symmetry principles (all states are equally likely, etc.) or even expert intuition. The *likelihood* is the probability of obtaining some data given a fixed state of the system, i.e. $\mathcal{P}(d|\theta)$. The *posterior* distribution is the probability of a given field configuration given some data, $\mathcal{P}(\theta|d)$. In this language, the posterior distribution is what we wish to obtain from an experiment. The posterior can be obtained from the prior and the likelihood by using Bayes theorem:

$$\mathcal{P}(\theta|d) = \frac{\mathcal{P}(d|\theta)\mathcal{P}(\theta)}{\mathcal{P}(d)} = \frac{\mathcal{P}(d,\theta)}{\mathcal{P}(d)} \tag{2.2}$$

which is a simple corollary of the product rule for probabilities. While this formula does depend on $\mathcal{P}(d)$, which is in general unknown, for a given trial it is constant, and so can be thrown out as an irrelevant normalization constant. The information in this section is well known was taken from a variety of sources, but may be found in any good probability textbook, like [5] for example.

2.2 Field inference

Now that the basic language of inference problems has been established, the goal of Information Field Theory can be stated. Suppose that the system

one was investigating was a continuous field $\phi(x)$ which is a function of the variable x in some set Ω. Suppose one also had experimental apparatus that measured that field, the goal is then to infer the value of the continuous field given a prior distribution and some data. To do this, we need to be able to write down probability distributions over fields, which means $\mathcal{P}(\phi)$ and $\mathcal{P}(\phi|d)$ become functionals.

Given that many operations on probability distributions such as normalization, expectation values etc. occur under an integral, we immediately see that we will have to commit the minor sin of resorting to the functional integral. If the probability distribution in question is everywhere greater than zero, then it can be rewritten in the form:

$$\mathcal{P}(\phi|d) = \exp(-H(\phi|d)) \quad \text{where} \quad H(\phi|d) = -\ln(\mathcal{P}(\phi|d)) \tag{2.3}$$

which, in our applications, we will always be able to do. We refer to $H(\phi|d)$ as the *information Hamiltonian* as a deliberate analogy to the physical Hamiltonians occurring in quantum/condensed matter field theories [3]. This direct analogy allows techniques developed in QFT to be applied to extracting relevant information such as expectation values and correlation functions from these formally ill-defined objects.

The goal is now to construct a posterior distribution given some prior $\mathcal{P}(\phi)$ on the field. In our framework we will generally assume that the measured data is a function of two things: a deterministic function of the field, R, known as the response function, plus some random measurement noise n which is independent of the field. We write this as $d = R(\phi) + n$. The data and noise will be assumed to live in $\mathbb{R}^m$ for some dimension m, although we will often extend to $\mathbb{C}^m$ in certain cases where it is mathematically convenient. The fields $\phi(x)$ defined on some set Ω must live in some subspace of $\mathcal{L}^2(\Omega)$, as we will shortly see that the IFT formalism requires the use of inner-products. These two spaces will be referred to as data space and signal space, respectively.

The full generality of information field theory can handle nonlinear responses, as well as prior distributions which are non-Gaussian in the fields, by expanding out the desired expectation values etc. in terms of Feynmann diagrams.

For the purposes of a short masters project however, we specialize to what we refer to as the linear case, analogous to the free theory in QFT. This means that we assume that the noise and the prior can be expressed as independent Gaussians:

$$\mathcal{P}(\phi) \propto \exp(-\frac{1}{2}\left\langle \phi-\psi\right| \Phi^{-1} \left|\phi-\psi\right\rangle_s) \quad \mathcal{P}(n) \propto \exp(-\frac{1}{2}\left\langle n\right| N^{-1} \left|n\right\rangle_d) \quad (2.4)$$

where N and Φ are some positive, self-adjoint operators on the data and field spaces respectively, and ψ is some assumed mean value of the field. The linear case also assumes that the response R is a linear map from signal space to data space. The spaces in which the inner-products are taken are denoted as subscripts s and d on the brackets for signal and data respectively.

The above Gaussians have the properties that $\mathbb{E}[|\phi-\psi\rangle\left\langle \phi-\psi\right|]_{\mathcal{P}(\phi)} = \Phi$ and $\mathbb{E}[|n\rangle\left\langle n\right|]_{\mathcal{P}(n)} = N$, these are often referred to as the correlation structures of the fields, or the covariance matrices, or simply covariances. $\mathbb{E}$ is used to denote an expectation value, with a subscript denoting which random variable the expectation is taken. The likelihood of the data given the signal is found by marginalizing over all possible values of the noise:

$$\mathcal{P}(d|\phi) \propto \int \delta(d-R\phi-n) \exp\Big(-\frac{1}{2}\left\langle n\right| N^{-1} \left|n\right\rangle\Big) dn \propto \mathcal{G}(d-R\phi, N) \quad (2.5)$$

where from now on we will use the notation $\mathcal{G}(a, A)$ to denote a zero-centred Gaussian in a with covariance A. To obtain the posterior from the likelihood, we use Bayes' theorem and multiply by the signal prior, i.e. we calculate $\mathcal{P}(d|\phi)\mathcal{P}(\phi)$. For the meantime, we assume that there is no prior mean field, $\psi = 0$. Given that all the involved terms are Gaussians, we can omit the exp terms and focus on the additive terms in the exponent:

$$\begin{aligned}
&-2\ln(\mathcal{P}(\phi|d)) = \left\langle \phi\right| \Phi^{-1} \left|\phi\right\rangle + \left\langle d-R\phi\right| N^{-1} \left|d-R\phi\right\rangle + C_0 \\
&= \left\langle \phi\right| \underbrace{(\Phi^{-1} + R^\dagger N^{-1} R)}_{D^{-1}} \left|\phi\right\rangle - \left\langle R^\dagger N^{-1} d|\phi\right\rangle - \left\langle \phi|R^\dagger N^{-1} d\right\rangle + \left\langle d\right| N^{-1} \left|d\right\rangle + C_0 \\
&= \left\langle \phi\right| D^{-1} \left|\phi\right\rangle - \left\langle D^{-1} D R^\dagger N^{-1} d|\phi\right\rangle - \left\langle \phi|D^{-1} D R^\dagger N^{-1} d\right\rangle + C_1 \\
&= \left\langle \phi - \underbrace{D R^\dagger N^{-1} d}_{m(d)}\right| D^{-1} \left|\phi - D R^\dagger N^{-1} d\right\rangle + C_2 \\
&= \left\langle \phi - m(d)\right| D^{-1} \left|\phi - m(d)\right\rangle + C_2
\end{aligned} \quad (2.6)$$

Note that we completed the square, and that any terms independent of ϕ (such as $\langle d| N^{-1} |d\rangle$) have been absorbed into the constants C_0, C_1 and C_2, which drop out as irrelevant normalization factors. One can immediately see that the result is again a Gaussian, so the expected mean field and variance can be simply read off: $\mathbb{E}[\phi]_{\mathcal{P}(\phi|d)} = m(d)$, $\mathbb{E}[(\phi - m(d))(\phi - m(d))^\dagger]_{\mathcal{P}(\phi|d)} = D$. We refer to this D as the *uncertainty variance*. The expected mean field $m(d)$ is a linear function of the data, and as such can be expressed by the action of a single linear operator, referred to as the *Wiener filter* [6]. We denote this object by W, such that $m(d) = Wd$,

$$W = (\Phi^{-1} + R^\dagger N^{-1} R)^{-1} R^\dagger N^{-1} = \Phi R^\dagger (R\Phi R^\dagger + N)^{-1} \tag{2.7}$$

The first equation is known as the signal space representation [4], and the second as the data space representation, after which space the inversion takes place in. We will rely on the second representation quite heavily throughout this text, as the inversion may be performed much more easily in data space, as well as being analytically easier to deal with. It is also stable in the limit of low-noise, i.e. $N \to 0$, which will often be assumed throughout the course of this thesis.

The derivation for the case of a nonzero prior mean ψ can be easily deduced, and is detailed in [4]. We will simply state that the posterior mean field $m(d)$ in the presence of a prior mean field is given by:

$$m(d) = \psi + W(d - R\psi) = D(R^\dagger N^{-1} d + \Phi^{-1}\psi) \tag{2.8}$$

with W and D the same as before.

Chapter 3

Dynamics

With the machinery for field inference now in-place, we can begin discussing dynamics. We assume that the field under consideration has an equation of motion of the form

$$\partial_t \phi(x,t) = f(\phi(x,t)) \tag{3.1}$$

for some function f. Many partial differential equations, including ones that are higher order in time, can be brought into this form.

Given this knowledge of the time evolution of the field, it should be possible to evolve the posterior probability distribution self-consistently. We take the view that if we assign a given field configuration a certain probability initially, if our views are consistent, then in the absence of outside factors the time-evolved field should have that same probability. We express this idea rigorously by defining ϕ_0 to be a value of the field at an initial time t_0, which undergoes evolution to a new configuration $\phi(t)$ at time t. We denote the operator taking ϕ_0 to $\phi(t)$ by $U(t)$ such that $\phi(t) = U(t)(\phi_0)$. Under these conditions, the time-evolved posterior distribution should be of the form:

$$\exp\left(-\frac{1}{2} \left\langle U^{-1}(\phi(t)) - Wd \right| D^{-1} \left| U^{-1}(\phi(t)) - Wd \right\rangle \right) \left| \det(J(U^{-1})) \right| \tag{3.2}$$

where we need some Jacobian volume factor $|\det(J(U^{-1}))|$ to account for

the changing probability mass[1], which in general will not be a constant if the time evolution is nonlinear in the fields. However, if the time evolution is nonlinear, then the original Gaussian posterior will evolve into a distribution which is non-Gaussian, which will in general be hard to deal with. In order to manage the scope of this project, we specialize to the case where the time evolution is linear, so that the equations of motion are given by $\partial_t \phi = L\phi$ for some linear operator $L(t)$. This ensures that the time evolution operator $U(t)$ will also be linear. Despite being denoted by U, this operator is not necessarily unitary. The evolved probability distribution can then be rewritten as:

$$\begin{aligned} &\exp\left(-\frac{1}{2} \langle U^{-1}(\phi(t) - UWd)| \, D^{-1} \, |U^{-1}(\phi(t) - UWd)\rangle \right) |\det(J(U^{-1}))| \\ &\propto \exp\left(-\frac{1}{2} \langle (\phi(t) - UWd)| \, {U^{-1}}^{\dagger} D^{-1} U^{-1} \, |(\phi(t) - UWd)\rangle \right) \end{aligned} \tag{3.3}$$

Where we note that the Jacobian dropped out because $|\det(J(U^{-1}))|$ is now a constant independent of the field values, and can be dropped as an irrelevant normalization factor. For this derivation to be correct however, it must still preserve information, i.e. $|\det(J(U))| \neq 0$.

Gaussians can be described by only two quantities, the mean field and the covariance. For the linear case, the mean field unsurprisingly evolves according to the equations of motion, and the uncertainty variance D evolves as $UDU^{\dagger}$. We are at the stage now where we at least have a self-consistent equation of motion for the posterior given some known time evolution of the fields. However, we have not yet arrived at a simulation scheme. Since, for any system that we need to simulate, we will not be able to compute the time evolution analytically, the previous equation will remain only a formal solution. Any construction of a practical simulation scheme will need to make a finite-dimensional approximation to this infinite-dimensional object.

[1]This Jacobian may formally be infinite-dimensional, however considering that we are already working with the functional integral, it doesn't really matter

3.1 Constructing the simulation scheme

To construct the actual simulation scheme, we break the time evolution up into a finite number of smaller timesteps, which we index by the variable i, $\{t_i\}$ for i going from 1 to N_{step}. Over these timesteps $[t_i, t_{i+1}]$ the field evolves according to some linear operator $U((t_{i+1}, t_i))$ which we assume can be represented as a matrix-valued Taylor series in $\Delta t = t_{i+1} - t_i$, i.e. $U(\Delta t) = \mathbb{1} + \Delta t L(t_i) + ...$ For notational convenience we set L to be constant in time (i.e. the system is linear *and* time-invariant), so $L(t) = L$. This assumption can be dropped at any time and does not affect the derivations. The formal solution for the time evolution is then $U(t) = \exp(tL)$.

We assume that we also have data d_i at timestep t_i that comes from some linear measurement of the field, as described in the previous sections. This data can result from a real-world experiment, or it can be a hypothetical measurement. In the latter case (which we will typically use), the response and prior simply define a rule for reconstructing the field given the data. This means that the response, prior and noise covariances are simply parameters which describe the nature of the "subgrid model" that we are using. We also declare that at the next timestep t_{i+1}, we will have some new response, prior and noise, which are considered fixed, and may in general be different[2] from those at t_i , and some yet-to-be determined data d_{i+1}. We distinguish the new and old responses etc. with subscripts i and $i + 1$.

Given the initial reconstruction at time t_i, the posterior probability distribution can be evolved to time t_{i+1}, where we also have a second posterior distribution from the hypothetical measurement at time t_{i+1}. The goal is now to select the data d_1 so that these two distributions match as well as possible.

This is done by picking the new data d_{i+1} such that the *Kullback-Leibler divergence* (a.k.a relative entropy) between the true evolved probability distribution from time t_i and the approximation at timestep t_{i+1} is minimized. For probability distributions P and Q over an arbitrary random variable x, it is given by:

[2]Of course not all choices of new responses and priors will give decent results, for example we will see later that the prior should be chosen so that it is consistent with the time evolution of the system.

$$\mathrm{KL}(P||Q) = \int P(x) \ln\left(\frac{P(x)}{Q(x)}\right) dx \tag{3.4}$$

It has the property of being always greater than zero, and has a minimum if and only if $P = Q$. The relative entropy measures the amount of information lost when Q is used to approximate P [7, p. 51], and crucially, this measure is asymmetric. Considering this asymmetry, it matters in which direction we take the divergence. It was pointed out by a member of the research group [8], that the previous publications on the subject of IFD [2] [4] have both taken the KL divergence in the wrong direction, so the IFD update equations will need to be rederived for this report. We have a true evolved probability distribution, which we are approximating by a Gaussian at the new timestep, so the Gaussian at timestep t_{i+1} will play the role of Q. For clarity and conceptual understanding, it is actually convenient to present the KL divergence for two arbitrary multivariate Gaussians, and then later insert the relevant terms. The proof is instructive, but tedious, and is not an original result [9]. It has therefore been relegated to the appendix.

Lemma 3.1.1. *For two Gaussians $\mathcal{G}(\phi - a, A)$ and $\mathcal{G}(\phi - b, B)$, the KL divergence between the two $\mathrm{KL}(\mathcal{G}(\phi - a, A)||\mathcal{G}(\phi - b, B))$is given by:*

$$\frac{1}{2}\Bigg(\mathrm{Tr}[\ln(BA^{-1}) - \mathbb{1} + B^{-1}A] + \langle b - a| \, B^{-1} \, |b - a\rangle \Bigg) \tag{3.5}$$

For our case, $\mathcal{G}(\phi - a, A)$ is the evolved probability distribution from t_i and $\mathcal{G}(\phi - b, B)$ is the new probability distribution at time t_{i+1}. We momentarily specialize to the case of no prior mean field, i.e. $\psi_i = \psi_{i+1} = 0$. This then allows us to set $B = D_{i+1}$, $A = UD_iU^\dagger$, $b = W_{i+1}d_{i+1}$, and $a = UW_id_i$, giving the full KL divergence:

$$\begin{aligned}\mathrm{KL} = \frac{1}{2}\Bigg(&\mathrm{Tr}[\ln\left(D_{i+1}(UD_iU^\dagger)^{-1}\right) - \mathbb{1} + D_{i+1}^{-1}(UD_iU^\dagger)] \\ &+ \langle W_{i+1}d_{i+1} - UW_id_i| \, D_{i+1}^{-1} \, |W_{i+1}d_{i+1} - UW_id_i\rangle \Bigg)\end{aligned}$$

We now pick the new data d_{i+1} such that the KL divergence is minimized. This is equivalent to saying *we pick the new data so that we retain the maximum possible amount of information.* A visual representation of this process

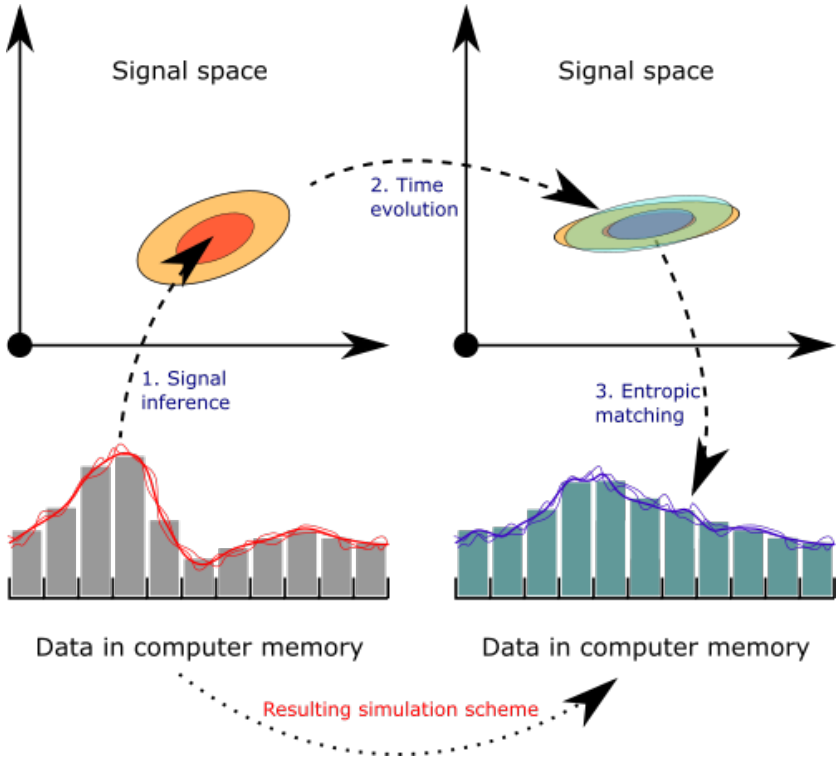

Figure 3.1: Schematic representation of the IFD approach. The ovals in the above picture represent level sets of multivariate Gaussians. Picture courtesy of Torsten Enßlin.

is shown in figure 3.1. To do this, first notice that there is only one term in the KL divergence which is dependent on the data, the term containing the inner product:

$$\langle UW_i d_i - W_{i+1} d_{i+1} | D_{i+1}^{-1} | UW_i d_i - W_{i+1} d_{i+1} \rangle \tag{3.6}$$

we take the derivative with respect to d_{i+1} and set it to zero, which yields

$$\begin{aligned} 0 =& W_{i+1}^\dagger D_{i+1}^{-1} W_{i+1} d_{i+1} - W_{i+1}^\dagger D_{i+1}^{-1} UW_i d_i \\ d_{i+1} =& (W_{i+1}^\dagger D_{i+1}^{-1} W_{i+1})^{-1} W_{i+1}^\dagger D_{i+1}^{-1} UW_i d_i \end{aligned}$$

The t_{i+1} terms in the above equation can be simplified using the the Wiener filter formula:

$$\begin{aligned}
&\left(W_{i+1}^{\dagger} D_{i+1}^{-1} W_{i+1}\right)^{-1} W_{i+1}^{\dagger} D_{i+1}^{-1} \\
&= \left(N_{i+1}^{-1} R_{i+1} D_{i+1} \underbrace{D_{i+1}^{-1} D_{i+1}}_{\mathbb{1}} R_{i+1}^{\dagger} N_{i+1}^{-1}\right)^{-1} N_{i+1}^{-1} R_{i+1} \underbrace{D_{i+1} D_{i+1}^{-1}}_{\mathbb{1}} \\
&= \left(R_{i+1} \underbrace{D_{i+1} R_{i+1}^{\dagger} N_{i+1}^{-1}}_{=W_{i+1}}\right)^{-1} N_{i+1} N_{i+1}^{-1} R_{i+1} = (R_{i+1} W_{i+1})^{-1} R_{i+1}
\end{aligned}$$

Thus giving a full update equation of:

$$\boxed{d_{i+1} = (R_{i+1} W_{i+1})^{-1} R_{i+1} U W_i d_i} \tag{3.7}$$

Because it will often be referred to, $(R_{i+1}W_{i+1})^{-1}R_{i+1}UW_i$ will be called the update operator, or transport operator (following the terminology of [1]), and we denote it by T_i.

To achieve a practical simulation scheme, the expansion of the operator U has to be truncated to some desired order in Δt. This is because finding the time evolution operator is equivalent to solving the system. So if U was known to arbitrary order, we would not need to do the simulation. From here on we denote the truncated expansion by $\bar{U} = \sum_{k=0}^{\alpha} (\Delta t L)^k / k!$ for some order α.

This update operator is actually computable, despite the fact that many of the involved matrices are infinite-dimensional, because the update operator as a whole is still finite. Given a prior, response and the equations of motion for the system, this operator can often be computed algebraically. When not, one can simply take a very-high resolution approximation to signal space, whose resolution is much higher than that of data space, precompute the update operator numerically at the start of the simulation, and save it in memory. In light of these considerations, it becomes apparent that the current IFD framework is best suited to time-invariant systems. If the update operator needs to be continually recomputed, then the time spent computing the operators may eclipse the time spent actually updating the data, because the approximation to signal space must always be of much higher resolution than that of data space.

To compute the data update equations for the case of a nonzero prior mean field, observe the general form for the KL in lemma 3.1.1. If the roles of the two Gaussians are swapped, the operator in the middle of the inner product changes from $B^{-1} \to A^{-1}$. For our case this would correspond to taking the divergence in the "wrong" direction, and thus switching the roles of the evolved posterior from t_i and the new posterior at t_{i+1}. Using this knowledge, for any incorrect formula in [4], we can fix it by sending $UD_iU^\dagger \to D_{i+1}$. For the case with the prior mean field, we copy the result presented in [4, p. 55] and fix the D terms, which yields:

$$d_{i+1} = (R_{i+1}W_{i+1})^{-1}R_{i+1}\Big(Um(d_i) - \psi_{i+1}\Big) + R_{i+1}\psi_{i+1} = \tag{3.8}$$

$$(R_{i+1}W_{i+1})^{-1}R_{i+1}\Big(U\Big[\psi_i + W_i(d_i - R_i\psi_i)\Big] - \psi_{i+1}\Big) + R_{i+1}\psi_{i+1} \tag{3.9}$$

3.2 Redundant parameters and simplifications

3.2.1 Prior mean field

The IFD formalism, as presented so far, has placed absolutely no restrictions on the form of the response, prior, noise and mean field at each timestep. This was indeed a deliberate choice, intended to maximize the generality in the derivations of [4]. It turns out however that the great freedom of choice for these parameters means that many of them are redundant, i.e. changing one is equivalent to changing another. Take for example the prior mean field, ψ. An assumed mean field makes sense in a pure inference problem, but it's role in a simulation scheme is not so clear. We expand eqn. 3.8 and rearrange it into a term dependent on the data, and one dependent on the prior means:

$$\begin{aligned} d_{i+1} =& (R_{i+1}W_{i+1})^{-1}R_{i+1}Ud_i \\ &+ (R_{i+1}W_{i+1})^{-1}R_{i+1}(U(\mathbb{1} - W_iR_i)\psi_i - \psi_{i+1}) + R_{i+1}\psi_{i+1} \end{aligned} \tag{3.10}$$

The first term is the one we want to keep, it is a linear update operator representing a linear equation, whereas the additional mean field terms introduce a drift at every timestep. The origin of this drift is easy to see: given the supposition of a prior mean field, the reconstruction of the data is an affine transformation, not a linear one, which moves the reconstruction towards the supposed mean. Repeatedly applying this operation will

introduce a persistent artificial shift in the equations. It could be asked if a consistency condition is missing; if one believes something about the mean field at a certain time, then they should update their beliefs self-consistently, i.e. one should set $\psi_{i+1} = U\psi_i$, or perhaps $\psi_{i+1} = \psi_i$. Neither of these identities however, when substituted into the previous equation, eliminate the drift.

To see that the parameter is also redundant, we write down the inner-product term from the KL divergence for the case with nonzero prior means, from [4], with the appropriate correction:

$$\langle Um(d_i) - m(d_{i+1})|D_{i+1}^{-1}|Um(d_i) - m(d_{i+1})\rangle \tag{3.11}$$

with $m(d_i) = \psi_i + W_i(d_i - R_i\psi_i)$ and likewise for $m(d_{i+1})$. This term is minimized by making $m(d_{i+1})$ as close as possible to $Um(d_i)$. The former depends lineary on the data and the prior mean ψ_{i+1}, thus shifting one simply introduces a compensatory shift in the data to attempt to return to the minimum KL. Given the redundancy, and the persistent drift for which we have not yet found a use, we discard it as a simulation parameter in this project.

3.2.2 Noise

We now focus on the next redundant parameter, which is the noise. The noise is a parameter which typically adds uncertainty to a measurement. However, at every timestep except the first (which may be the result of a real measurement), this noise is completely fictitious, as the simulation scheme is performing hypothetical measurements. One naturally asks why we should have update equations which adjust for a nonexistent uncertainty. The answer is that the noise term was initially kept in the equations in the hope that it may be a useful tunable parameter for the reconstructions, or it may help to describe the uncertainty in the simulation coming from numerical error. It turns out however, that the noise can simply be discarded as a parameter in IFD:

Lemma 3.2.1. *The equations of motion for linear IFD are independent of the noise up to a simple equivalence.*

Proof. For a simulation scheme with timesteps t_i for $i \in \{1, ...n\}$, responses $\{R_i\}$, priors $\{\Phi_i\}$, noises $\{N_i\}$, Wiener filters $\{W_i = \Phi_i R_i^\dagger (R_i \Phi_i R_i^\dagger + N_i)^{-1}\}$,

and linear time evolution operators $\bar{U}_i = 1 + \Delta t L_i + ...$, the data update equations are given by:

$$\begin{aligned} d_{i+1} &= (R_{i+1}W_{i+1})^{-1} R_{i+1}\bar{U}_i W_i d_i \\ &= \Big[R_{i+1}\Phi_{i+1}R_{i+1}^\dagger (R_{i+1}\Phi_{i+1}R_{i+1}^\dagger + N_{i+1})^{-1}\Big]^{-1} R_{i+1}\bar{U}_i \Phi_i R_i^\dagger (R_i\Phi_i R_i^\dagger + N_i)^{-1} \end{aligned} \tag{3.12}$$

The second line is obtained by inserting the definition of the Wiener filter. We rename the terms: $(R_i\Phi_i R_i^\dagger + N_i) = C_i$, $(R_i\Phi_i R_i^\dagger) = B_i$ and $R_{i+1}\bar{U}_i\Phi_i R_i^\dagger = A_i$, yielding:

$$d_{i+1} = (B_{i+1}C_{i+1}^{-1})^{-1} A_i C_i^{-1} d_i = C_{i+1}B_{i+1}^{-1}A_i C_i^{-1} d_i \tag{3.13}$$

The update equations are then iterated n times, yielding:

$$C_n B_n^{-1} A_{n-1} \underbrace{C_{n-1}^{-1} C_{n-1}}_{=\mathbb{1}} ... A_0 C_0^{-1} d_0 = C_n \Big(\prod_{i=0}^{n} B_{i+1}^{-1} A_i \Big) C_0^{-1} d_0 \tag{3.14}$$

The only terms with any dependence on the noise were the C terms and therefore, up to a change of basis at the beginning and end of the simulation, the equations of motion are independent of the noise. This holds even if the noise, response and prior change at every timestep. In the infinite-noise limit, $C \to N$, and in the zero noise limit $C \to B$.

□

Given the equivalence, from here on in we will always work in the no-noise limit, and the data update procedure becomes:

$$\prod_{i=0}^{n} A_i B_i^{-1} = \prod_{i=0}^{n} R_{i+1}\bar{U}_i \Phi_i R_i (R_i\Phi_i R_i^\dagger)^{-1} = \prod_{i=0}^{n} R_{i+1}\bar{U}_i W_i \tag{3.15}$$

with W_i now being the no-noise version of the Wiener filter. This means we can always write the transport operator as: $T_i = R_{i+1}\bar{U}_i W_i$. Furthermore, many times throughout this project, an unchanging response will often be used. We can then exploit the fact that in the no noise case, the Wiener filter has the property $R_i W_i = \mathbb{1}$, to rewrite the transport operator in the following useful form:

$$T_i = R(\mathbb{1} + \Delta t L + \cdots) W_i = \mathbb{1} + \Delta t RLW_i + \Delta t^2 RL^2 W_i + \cdots \tag{3.16}$$

This form is valid even if the prior changes at every timestep. The no-noise assumption allows us to free up the symbols N and n, which will now denote integers etc.

Worth mentioning is that the unsimplified transport operator found in 3.7, was of a similar form, but had a $(R_i W_i)^{-1}$ term out the front, whose purpose was unclear. Some readers may have also found the derivation of the update equations rather odd, since the entropic matching of two probability clouds, at first sight, has little to do with numerical simulation schemes. However the new form of the transport operator, $R_{i+1}\bar{U}_i W_i$, has a particularly simple interpretation: we guess the true field using the Wiener filter, evolve it, then measure the field again at the new timestep. All of this is without any reference to KL divergences.

3.2.3 Data/response equivalence

The next redundancy is the responses. The ability to select new responses at any time gives us the freedom to change to a more convenient coordinate system whenever we require. The responses can however be dynamically updated in a way that is equivalent to updating the data; by picking $R_{i+1} = R_i U^{-1}$. A time-evolved response should give an unchanging data output when acting on a time-evolved field: $d_{i+1} = R_{i+1}\phi(t_{i+1}) = R_i U^{-1} U\phi(t_i) = R_i\phi(t_i) = d_i$. This holds at least for a hypothetical measurement of the field. We should check that this behaviour is also reflected in the update equations:

$$d_{i+1} = R_{i+1}U\Phi_i R_i^\dagger (R_i\Phi_i R_i^\dagger)^{-1} d_i = R_i U^{-1} U\Phi_i R_i^\dagger (R_i\Phi_i R_i^\dagger)^{-1} d_i = d_i \quad (3.17)$$

which is consistent with our expectations. If the responses are updated according to the field evolution, then the data is static. Given a nontrivial time evolution, RU^{-1} will of course need to be simulated. This however, would mean that we have gone from attempting to model a field, to attempting to model an operator on the space of fields, and thus we have gone up a whole level of complexity. It will therefore in general be hard to exploit this equivalence.

As we will see later though, a code is developed in which the responses are partially updated using this equivalence, and in that specific case some data

processing is offloaded onto the response, where the computation is more convenient. It must be pointed out that in this process, we have swapped out an information-theoretic update on the data side (one that is constructed to minimize information loss) for a non information-theoretic update on the response side. This is because we have not yet prescribed *how* RU^{-1} is approximated. An information-theoretic response update would involve taking equation A.6, fixing R_i and minimizing w.r.t R_{i+1}, which should yield a set of responses which best capture the time evolved reconstruction and lose the least information.

However a short look at eqn. A.6, shows that inserting $D = (\Phi + R^\dagger N^{-1} R)^{-1}$, $W = \Phi R^\dagger (R\Phi R^\dagger + N)^{-1}$ etc. and attempting to minimize the KL will result in a highly nonlinear equation in the response matrices, objects which are themselves very high dimensional. For the purposes of this project, we discard the possibility of updating the responses information-theoretically; trying to model the linear evolution of a field by going through a nonlinear equation in operators *on* fields is probably suboptimal[3].

There was a point to this detour: the previously mentioned code in which we partially update the responses is not 100% information-theoretic. This should be kept in mind.

3.3 Errors, stability and convergence

Until now, it has not been proven that the IFD update equations actually converge to the true time evolution of the field. The previous publication [4] proved that the objects involved are mathematically well-defined, but it was not proved that schemes produced using this framework actually represent the equations of motion of the system. However, given the extremely general nature of the IFD framework, such a discussion is difficult. The IFD equations can represent sensible schemes, and they can also deliver rubbish. The reader will unfortunately have to wait a while before an example of a sensible scheme is presented, but we can present an illustrative example of

[3]The response/data equivalence was already discovered in [2], where it was shown that for a prior which is invariant under time evolution ($U\Phi U^\dagger = \Phi$), evolving the response as RU^{-1} perfectly preserves information. However if U is not analytically known, we are back at square one.

the latter case.

Example 3.3.1. In the worst case scenario, suppose one has a time-invariant system, and that U is expanded to first order $\bar{U} = \mathbb{1} + \Delta t L$. Suppose further that the responses and prior are static. It is possible to choose a response and prior that are so poorly designed, that for every vector v in the image of the Wiener filter, Lv lies in the kernel of the response. This would mean that

$$d_{i+1} = R(\mathbb{1} + \Delta t L)W d_i = d_i + \Delta t RLW d_i = d_i \tag{3.18}$$

Thus the data does not evolve in time, and the reconstructions do not evolve either, and the model achieves nothing. The kernel of the response is infinite dimensional, so we should expect the useless schemes to outnumber the useful ones.

3.3.1 Error

Given some intuition about the problem, we are now ready to derive some general formulas for the error. The previous example showed that we cannot expect to prove that in IFD the error is always less than some bound, or that the codes produced always converge. Therefore, errors/convergence must be checked specifically for each new model that we develop.

The error is defined as the difference between the true solution and the simulated solution. For a simulation on a discretized grid in time and space with locations $\{x_i\}$ and $\{t_j\}$, if the simulated solution is denoted by ϕ_i^j and the true solution is $\phi(t_j, x_i)$, then the global error at timestep j is typically defined as:

$$E_j = \phi_i^j - \phi(t_j, x_i) \tag{3.19}$$

The true analytic solution is in general unknowable, and the error is often estimated by looking at the *one-step error* (OSE) or *local error*, which is the error accumulated in a single timestep. We represent one step of the numerical simulation by the operator T. We then assume that at timestep t_j, there is no error: $\phi_i^j = \phi(t_j, x_i)$, and then define the one-step error by $\text{OSE} = T[\phi_i^j] - \phi(t_{j+1}, x_i)$. The global error is then bounded by summing the absolute values of the local error at every timestep ([10] p.593).

IFD was constructed with the explicit goal of minimizing the information lost at each timestep via the KL divergence. Thus in IFD, the criteria for success is to minimize the information theoretic error rather than the traditional error. Given that the KL divergence is an abstract distance between probability distributions, these two notions may have little to do with one-another. However, it turns out that in IFD there are three valid ways of interpreting error, and they are all roughly equivalent[4]. Observe the inner-product term in the KL divergence formula:

$$\langle W_{i+1}d_{i+1} - UW_i d_i)|\, D_{i+1}^{-1}\, |W_{i+1}d_{i+1} - UW_i d_i\rangle \tag{3.20}$$

this represents the information lost when passing from one timestep to the next, and is thus the local information-theoretic error, which will now be denoted by E_{KL}. One sees from inspection that E_{KL} goes to zero if and only if the local signal-space error E_s:

$$E_s \equiv ||W_{i+1}d_{i+1} - UW_i d_i|| \quad (\mathcal{L}^2 \text{ norm}) \tag{3.21}$$

goes to zero. The global signal space error is naturally $||W_i d_i - \phi(t_i)||$. We have the liberty of measuring the error in signal space because we have a formula for reconstructing the field given the data, in contrast to other numerical schemes. For the KL and signal space errors, if the error approaches zero in one norm, then it approaches zero in the other.

There is also a correspondence between the signal-space error and the typical notion of error, which we will refer to as data-space error, and denote by E_d. This notion must be slightly generalized, because IFD allows us to take arbitrary measurements of the data. We set $E_d = d_i - R_i\phi(t_i)$, and for a response which is a series of point-measurements of the field (i.e. a grid of delta-functions), this agrees with the old definition. We use the no-noise Wiener filter identity $R_i W_i = \mathbb{1}$ to write:

$$|E_d| = ||d_i - R_i\phi(t_i)|| = ||R_i(W_i d_i - \phi(t_i))|| \leq ||R_i|| \cdot ||(W_i d_i - \phi(t_i))|| \tag{3.22}$$

where $||R_i||$ denotes the operator norm. This equation shows that convergence in signal space implies convergence in data space, but not vice versa.

[4]It must be noted that this argumentation applies when the data is the only parameter being updated information-theoretically. If, for example, the responses are being updated information theoretically as discussed in subsection 3.2.3, then the full KL formula will need to be considered.

As long as the Wiener finter reconstructions are not perfect, there will be some signal-space error. When analysing error, since the true behaviour of the field is unknown, the error can typically only be expressed by an upper bound that has some dependence on the time and space resolutions Δt and Δx.[5] One then asks how the error scales in the limit $\Delta x, \Delta t \to 0$. In the limit of high resolutions, we expect the Wiener filter reconstructions to become perfect, i.e. $W_i R_i \to \mathbb{1}$, and in this limit, signal space and data space approach one another and the notions of error become equivalent[6].

Thus, there are three ways to analyse the error in IFD, all of which are roughly equivalent. Note however that the scaling of data/signal space errors differs from the KL error by a factor of a square. This is however a purely cosmetic factor that has to do with units, and does not represent an underlying higher accuracy in the IFD framework.

We now seek a general formula for the local data space error. We start by assuming zero accumulated error, i.e. $d_i = R_i \phi(t_i)$. We must then pick an expansion of U to some finite order α, $\bar{U} = \sum_{k=0}^{\alpha} (\Delta t L)^k / k!$. This gives the data update equation $d_{i+1} = R_{i+1} \bar{U} W_i d_i$, which can then be used to bring the local error into the following form:

$$\begin{aligned} E_d &= ||d_{i+1} - R_{i+1}\phi(t_{i+1})|| = ||R_{i+1}\bar{U}W_i d_i - R_{i+1}U\phi(t_i)|| \\ &\leq ||R_{i+1}\bar{U}(W_i R_i - \mathbb{1})\phi(t_i)|| + ||R_{i+1} \sum_{k=\alpha+1}^{\infty} \frac{(\Delta t L)^k}{k!}\phi(t_i)|| \end{aligned} \tag{3.23}$$

The second term is the expected truncation error from an order α time expansion. All other higher order terms in time are still present, but are modified by the term $(W_{i+1}R_{i+1} - \mathbb{1})$, which measures the accuracy of the Wiener filter reconstruction. Thus to bound the error to any order, a bound needs to be placed on the spatial part of the reconstruction.

This formula shows that for any IFD code, the one-step error is determined by only three factors: how much is lost by measuring then reconstructing

[5] Astute readers will note that a notion of Δx for IFD has not been constructed yet, this will be done in the next section.

[6] Note that $W_i R_i$ is an operation on signal space describing the act of measuring then reconstructing. This is different to the operator on data space, $R_i W_i$, which describes reconstructing then measuring, and is always $\mathbb{1}$ for our purposes.

$(W_{i+1}R_{i+1})$, the order of the expansion of U, and how much of the time-evolved reconstruction is captured by the new response R_{i+1}. Error will be analysed for only one class of models in this report, where we only observe the scaling behaviour, and thus are working in the limit where data and signal spaces approach each other. Nonetheless, it is useful to have a general formula for the error in IFD.

3.3.2 Stability

Stability is easy to describe intuitively. If there are certain solutions to a finite-difference scheme that grow without bound, when the actual physical solutions do not, then the code is unstable. The unbounded solutions will grow to dominate any simulation, no matter how small they are initially. To define stability rigorously, in the form that we need it, we first need to define a notion of spatial resolution in IFD, i.e. we need a Δx to match our Δt.

Now, the IFD framework technically doesn't need a notion of spatial resolution. After all, the responses can just be any arbitrary linear functions of the field, and don't need to be localized anywhere. However, there is already a significant wealth of theorems on stability, convergence and errors that exist in the literature, that rely on some notion of a spatial grid. Constructing a notion of Δx will give us access to these. Because we are also not extraordinarily creative, every system of responses used in this report will correspond to some sort of spatial grid anyway. For this project we can simply state that for a 1D simulation domain of length l, and an n dimensional data space, then $\Delta x = l/n$ as per usual. Higher dimensional domains are defined analogously.

For more creative systems of responses, a concept of resolution is still easy to define. Any simulation scheme for PDE's is a finite approximation to an infinite-dimensional object, so there is always a notion of resolution. Therefore it will always make sense to ask what happens in the limit of high resolutions. If in doubt, for any IFD scheme one can simply take the dimension n of data space, and set $\Delta x = C/n$ for some arbitrary constant C.

The *Lax-Richtmyer* definition of stability [11], is that given a transport operator T for the simulation, which defines $d_{i+1} = Td_i$, which is defined for

some Δx and Δt which both go to zero, and a total simulation time τ such that $\Delta t = \tau/N$ for a number of timesteps N, then the simulation is defined to be stable if the set:

$$\{T^n(\Delta t, \Delta x) | n \in \mathbb{Z}, 0 \leq n \leq N\} \tag{3.24}$$

is uniformly bounded in the operator norm in the limit $\Delta x, \Delta t \to 0$. If the set is not uniformly bounded, then that means there is a solution whose magnitude grows without bound.

There is a another notion of stability known as *Von Neumann stability*, which is far easier to compute. Von Neumann stability analysis assumes that the coefficients of the PDE do not change in space, the grid spacing is regular, and that the boundary conditions are nice enough such that the transport operator will be translation-invariant and will thus have a diagonal representation in Fourier space. The eigenvalues $\{\lambda_i\}$ of the transport matrix are then analysed, if the magnitude $|\lambda_i|$ of any of them is greater than one, then there is an exponentially growing mode and the simulation is unstable. Instability in the Von Neumann sense implies instability in the Lax-Richtmyer sense, but is only equivalent in certain cases [11]. Given that stability is hard to check analytically for numerical solvers with nonconstant spatial coefficients, grids etc., it is common practice to analyse a new numerical scheme on a constant-coefficient problem, and use it as an indicator of stability for the nontrivial problem [12, ch. 7].

3.3.3 Convergence

A notion closely related to error is that of convergence, which asks if the simulation scheme actually approaches the true behaviour of the field in the limit of high resolutions Δx and $\Delta t \to 0$. This is equivalent to asking if the global error goes to zero.
Of the many theorems on convergence that a notion of Δx gives access to, the absolute most important is the Lax-Richtmyer equivalence theorem [11]. For a differential equation of the form $\partial_t \phi(t, x) = L(t)\phi(t, x)$, and a set of well-posed boundary conditions, the theorem states:

Theorem 3.3.2. *Given a properly posed initial value problem, and a finite difference approximation $T(\Delta t)$ to it that satisfies the consistency condition,*

stability is a necessary and sufficient condition that $T(\Delta t)$ be a convergent approximation.

What is consistency? Consistency essentially states that in the limit of high resolutions, the approximated operator actually approaches the true differential operator. This may seem obvious, but often one can write down sensible-looking schemes which are in fact not consistent. In the notation of Lax and Richtmyer, they assume that the spatial and time resolutions are coupled through some function $\Delta x = g(\Delta t)$ which ensures that the simultaneous limit of $\Delta x, \Delta t \to 0$ is always taken. We follow their notation, but note that it is mostly a formality.

Definition 3.3.3 (Consistency)**.** For an operator $T(\Delta t)$ which approximates $U(t)$, with $U(t)$ being the analytic time evolution operator corresponding to $L(t)$, the approximation is said to be consistent, if for some set of solutions, Ω, to the differential equation, then for any $\phi \in \Omega$,

$$\lim_{\Delta t \to 0} \left\| \Big(T(\Delta t) - U(\Delta t) \Big) \phi(t, x) \right\| = 0 \tag{3.25}$$

uniformly in t.

The definition has been (harmlessly) paraphrased, with some technical details left out. It needs to be pointed out that the definition of consistency involves comparing operators which live in different spaces: $T(\Delta t)$ acts on a discrete space, yet $U(\Delta t)$ acts on a continuous space. The original paper by Lax et al. assumes that there is some sufficient level of smoothness such that Taylor series expansions or smooth interpolation etc. may be used to bridge the gap between spaces. We will not use the exact details here, and allow ourselves to freely make such statements as $(f_{i+1} - f_{i-1})/2\Delta x \to \partial_x f(x)$ as $\Delta x \to 0$.

Thus, to analyse consistency for any general model, we will need to specify a rule for comparing these operators on different spaces, which will depend on our choice of responses etc. So a general formula for consistency of IFD schemes is not presented here. The Lax equivalence theorem is extremely useful because convergence is often hard to check, as it is a global property. Stability and consistency however, are both easy-to-check local properties.

3.4 Implicit methods

IFD, has so far been constructed as an *explicit scheme*, the data at a new timestep is solved as an explicit function of the data at the previous timestep. The most basic example of a forward difference scheme is the Euler method. Given the DE $\partial_t \phi(x,t) = f(\phi(x,t))$, the update scheme is then $(\phi_{i+1} - \phi_i)/\Delta t = f(\phi(t_i)) \Rightarrow \phi_{i+1} = \phi_i + \Delta t f(\phi(t_i))$.

Forward Euler schemes can have drawbacks, for example, they are often unstable. One common remedy for this is to use an *implicit scheme*, such as the backward Euler method, which solves an implicit equation expressing the data at t_i in terms of t_{i+1}, i.e. $\phi_{i+1} = \phi_i + \Delta t f(\phi(t_{i+1}))$. For nonlinear equations, the implicit schemes are in general harder to solve [10, sec. 13]. We did attempt the use of a backward scheme during this project, but was not included in this report.

If one chooses constant responses and prior, and a first order time expansion, then eqn. 3.7 is brought into the form $d_{i+1} = d_i + \Delta t RLW d_i$, which is cosmetically identical to the forward Euler scheme. One is then tempted to interpret the above as a DE in continuous time: $d'(t) = RLW d(t)$, from which a backward Euler method is defined. However the data is not a continuous variable, and this construction doesn't generalize to nonconstant responses, which are only defined at discrete timesteps. Even if one could construct an $R(t)$, it wouldn't account for the possibility of the dimension of data space changing between timesteps, which is not a continuous transformation.

Implicit methods may still be constructed in IFD, they just need to be defined correctly. Start with a reconstructed posterior at a time t_{i+1} given some data d_{i+1}, and time evolve this distribution backwards, and then minimize the KL divergence w.r.t. the data at t_i, this gives:

$$\begin{aligned} d_i &= R_i \bar{U}^{-1} W_{i+1} d_{i+1} = R_i(\mathbb{1} - \Delta t L + \ldots) W_{i+1} d_{i+1} \\ &\Rightarrow d_{i+1} = [R_i(\mathbb{1} - \Delta t L + \ldots) W_{i+1}]^{-1} d_i \end{aligned} \tag{3.26}$$

The fix applied to the definition was scarcely worth mentioning, except for the fact there are a variety of creative timestepping schemes in existence such as Runge-Kutta methods etc. that future studies may want to apply to IFD.

The moral here is that often such schemes assume that one is solving an underlying continuous equation, which eqn. 3.7 is not. So, one should check the validity of such schemes in IFD before proceeding.

3.5 Boundary conditions

The astute reader will have noticed that boundary conditions have not been mentioned yet. This is because they aren't baked into the IFD formalism. To be able to compute the necessary Gaussian functional integrals, we needed to exploit the fact that we are working in a vector space of functions. Now for general boundary conditions, this isn't true. As a trivial example, take two functions f and g on an interval $[0, L]$, given that $f(0) = g(0) = a$ and $f(L) = g(L) = b$, but $f + g$ does not equal a or b on the boundaries. The addition of boundary conditions typically restricts us to an affine space. It would probably be easy to extend the IFD formalism to handle boundary conditions and affine spaces, but that is beyond the scope of this project.

The complicating factor with IFD is that the reconstructions of the field given the data are typically nonlocal. This is the desired behaviour, in that the spatial correlation structure of the field is used to get better inferences of the field at a location than one would get from a local reconstruction. The nonlocality comes from the $(R\Phi R^\dagger)^{-1}$ term in the Wiener filter. With local spatial correlations, the $R\Phi R^\dagger$ matrix is indeed local (i.e. almost diagonal in the spatial indices), however the matrix inversion depends on global properties of the system. Thus throughout this thesis, boundary conditions, when implemented at all, will be done in an ad-hoc manner, and only for some models.

3.6 Trial system: Cosmic Ray Simulations

The original goal of this project was to develop a first real-world application for IFD, and the problem chosen for this was the simulation of cosmic ray transport. Cosmic rays (CR's) are very high energy particles originating in space, and consist mainly of charged protons. On galactic scales they can be described by the equations of Magnetohydrodynamics, which describes the behaviour of charged plasmas by combining Maxwell's equations and the

Navier-stokes equations. In [13], the author derives an equation for cosmic ray propagation, in which the cosmic ray plasma is represented by a number density distribution $f(\vec{x}, \vec{p}, t)$ in the three-dimensional location $\vec{x}$ and momenta $\vec{p}$ and time t. For low energy cosmic rays, it is believed [14] that over galactic scales, the magnetic field of interstellar space has a far stronger effect on the movement of cosmic rays than the transport induced by their own momenta $\vec{p}$. In addition, the distribution of cosmic ray momenta is assumed to be nearly isotropic. This justifies replacing the vector momentum by a scalar, p. Under these assumptions, and some others, the cosmic ray transport equation in phase space is given by [14]:

$$\frac{\partial f}{\partial t} + \frac{\partial}{\partial x_i}(\nu_i f) + \frac{\partial}{\partial p}(\dot{p} f) = \frac{\partial}{\partial x_i}(\kappa_{ij} \frac{\partial}{\partial x_j} f) + q - \frac{f}{\tau_{loss}} \tag{3.27}$$

where $\nu_i(\vec{x})$ is the cosmic ray transport velocity which advects the particles along magnetic field lines. This term is an a For simplicity it is assumed to be a vector quantity independent of the momentum. $\kappa_{ij}(\vec{x})$, is the spatial diffusion tensor, which is highly anisotropic in directions parallel and perpendicular to the background magnetic field. $\dot{p}$ describes acceleration and deceleration of CR's in response to plasma waves and interaction processes. $q(\vec{x}, p, t)$ is an injection term, which corresponds to various processes in space producing cosmic rays. $\tau_{loss}(\vec{x}, t, p)$ is the catastrophic loss timescale, which describes CR's being removed from the population due to collisions with interstellar gas etc. As it is not relevant to this thesis, we neglected a momentum space diffusion term describing second order Fermi acceleration.

The typical regions of interest for simulating CR populations are on the scales of galaxies and galaxy clusters [15], the latter of which are the largest bound objects in the universe. A complete treatment of the CR evolution equation would require simulating the momentum component with equal resolution to that of the spatial component. This promotes the N^3 scaling of the memory requirements for the spatial components to an N^4 scaling, which becomes unmanageable at the high spatial resolutions required for resolving cluster-scale structures [16]. Thus practical simulations are often limited to momentum resolutions of the order of only a handful of bins, ≈ 10, and assume a subgrid structure in order to get satisfactory results [16, 17]. This made the CR evolution equation a promising candidate for an application of IFD, as it was hoped that it would offer an improvement over current subgrid models.

Developing a satisfactory simulation scheme was harder than initially expected, and there were many unexpected problems that needed to be solved, both theoretical and practical. Hence equation 3.27 wasn't solved in all its generality. It can be seen that if the injection and catastrophic loss terms are ignored, the equation is mostly just advection by a nonconstant velocity field and diffusion with a nonconstant diffusion tensor. The term $\frac{\partial}{\partial p}(\dot{p}f)$ is just advection in the momentum component by a velocity field $\dot{p}$. These difficulties required us to curb our expectations, and redefine "success" as simulating advection and diffusion in one dimension, with a nonconstant velocity field, and constant diffusion coefficient. The original problem was kept in mind, and thus a secondary goal for the project was developing a scheme which was viable at low resolutions.

Chapter 4

Translation-invariant schemes

4.1 Toy model: straight advection

One-dimensional advection on a finite interval will form the first trial problem for which we can develop a toy model. The development of this toy model will be presented side-by-side with some analytical results on a broader class of models for systems where the time evolution is diagonal in Fourier space. These analytic results will be used to analyse the error and convergence properties of the toy model. This model will then be extended to nonperiodic boundary conditions and nonconstant velocity fields. The analytic results presented here are for the one-dimensional case, but they trivially generalize to higher dimensions.

A disclaimer must be added at the start here, the error analysis for these models is carried out for the case of constant velocity and periodic boundary conditions. This case can already be solved analytically. However this practice is entirely normal in numerical methods, as many advanced schemes are too complicated to permit an analytic analysis. This is the case in IFD; as the codes are typically nonlocal, and the algebraic equations are always dependent on the geometry of the simulation domain. In this thesis, the best that we can do is prove convergence for the analytically solvable case, and then hope that these conclusions hold in the non-analytically solvable case. This should be kept in mind at all times.

The equation to be solved is the one-dimensional advection equation:

$$\partial_t \phi(t,x) = \partial_x (v(x)\phi(t,x)) \tag{4.1}$$

for $v(x)$ some velocity field defined on $x \in [0,l]$, which will at first be set to a constant v and brought in front of the partial derivative.

We start by selecting the signal and data spaces. The simulated space inside the computer must always be of finite extent. For this reason, we choose the signal space to be $\mathcal{L}^2([0,l])$. This is despite the fact that the true field itself may live on all of $\mathbb{R}$. The justification for this is that $\mathcal{L}^2([0,l])$ should contain all the relevant information for the simulation, given appropriate boundary conditions. Furthermore, most results will be derived in Fourier space, where the restriction of domain converts contour integrals in momentum space to infinite sums, which are much more manageable.

Now the prior must be selected. The system under consideration is fairly abstract, so there is a lot of room for choosing a prior which suits our goals. If one starts with the assumption that no point in the space is special[1], then the prior will be translation-invariant. This property means that the prior will always have a diagonal representation in Fourier space. The positivity and self-adjointness conditions on the prior ensure that the eigenvalues in momentum space will be everywhere positive and greater than zero, and symmetric about the origin. Priors of this form are generally referred to as smoothness priors. Using k to denote momentum, a prior $\Phi(k)$ whose values fall to zero as $k \to \infty$ essentially states that rapid oscillations in the signal are deemed unlikely; the field is smooth.

Common examples of a prior include power laws in momentum, i.e. $|k|^\beta$ for some integer β, often supplemented by a regularizing mass term: $\Phi(k) = 1/(|k|^\beta + m^\beta)$. For our toy example, we choose $\Phi(k) = k^{-4}$, although any prior with $\Phi(k) \to 0$ as $k \to \infty$ sufficiently fast would also work. The use of the Fourier transform tells us that we have already imposed periodic boundary conditions on the system. To extend to nonperiodic boundary conditions, we notice that in the case that $\Phi(k)$ has an inverse Fourier transform, then it's action on a function $\phi(x)$ takes the form of a convolution $\int \Phi(x-y)\phi(y)dy$.

[1] We will see later that for nonconstant velocity fields, this can cause problems, because some points *are* special.

This helps extend to nonperiodic boundary conditions later. Throughout this report, we use the convention that the (signal space) Fourier modes are normalized as: $\frac{1}{\sqrt{L}}e^{ikx}$ with $k = 2\pi n/l$, for n in the integers.

An expansion order for U must now be selected. For the toy model, we initially pick $\bar{U} = \mathbb{1} + v\Delta t\partial_x$, but stability requirements will later require going to second order in time.

Now to construct the responses. We assume that we have N points which will be labelled with the index j. The responses are chosen to be constant in time, and the subscripts R_j now denote spatial indices. The most natural and naive response is to choose that the index j labels a regular grid of positions, and that the response is an average of the field value around that position. We define $\Delta x = l/N$ and the box function:

$$B(x) = \begin{cases} 1/\Delta x & 0 \leq x \leq \Delta x \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

We then define the response operator from the signal space to the data space to be:

$$(R\phi)_j = \int_0^l dx B(x - x_j)\phi(x) \equiv \int_0^l dx B_j(x)\phi(x) \tag{4.3}$$

where x_j is the x-position of the j-th gridpoint, i.e. $x_j = \Delta x \cdot j$. This function simply averages the signal around a box centred at $\Delta x/2$. The B_j's form an orthogonal set, but they are not normalized since $\int B_j^2 dx = 1/\Delta x$, nor are they a basis.

It is quite easy to generalize this response to arbitrary functions $B(x)$ on $\mathcal{L}^2([0, l])$, simply set $(R\phi)_j = \int_0^l dx B_0(x - x_j)\phi(x)$. If the x_j's are evenly spaced, we refer to any response of this form as a *translation-invariant response*. The $B(x)$ functions will be referred to as the *response bins* or just *bins*.

We now want to begin to calculate the transport operator, starting with the computation of $(R\Phi R^\dagger)^{-1}$. Since both the responses and prior are invariant under translations of multiples of Δx, this allows us to actually make a very

general statement. Given that a translation-invariant prior will be diagonal in momentum space, we can state:

Lemma 4.1.1. *Given a signal space of the form $\mathcal{L}^2([0,l])$ with periodic boundary conditions, a translation invariant response R_j whose bin function $B(x)$ has a Fourier series representation, as well as a prior Φ which is diagonal in momentum space, $(R\Phi R^\dagger)_{jl}$ will be of the form:*

$$\sum_k \Phi(k)|\widehat{B}(k)|^2 e^{ik(x_j - x_l)}, \tag{4.4}$$

where $\widehat{B}(k)$ is the Fourier coefficient of $B(x)$.

Proof. By the shift property of the Fourier transform, $\widehat{R}_{j,k} = e^{-ikx_j}\widehat{R}_{0,k} = \widehat{B}(k)$. Therefore $(R\Phi R^\dagger)_{jl}$ is

$$(R\Phi R^\dagger)_{jl} = \sum_k \sum_q e^{ikx_j}\widehat{B}(k)\Phi(k)\delta_{kq}\widehat{B}^*(q)e^{-iqx_l} = \sum_k \Phi(k)|\widehat{B}(k)|^2 e^{ik(x_j - x_l)} \tag{4.5}$$

as desired. □

The generalization to higher dimensions takes x and k to vectors. It must be stressed here that this formula holds regardless of how the update matrices are actually computed. We are not necessarily solving the equations in Fourier space, but we know that the operators always have such a representation. From now on, we will refer to any simulation scheme which satisfies the criteria of the previous lemma as a *translation invariant scheme*.

For the toy model, we compute the Fourier transform of $B(x)$:

$$\begin{aligned}
\widehat{B}(k) &= \int_0^l dx B(x)\frac{1}{\sqrt{l}}e^{ikx} = \frac{1}{\sqrt{l}}\int_0^l dx B(x)e^{ikx} \\
&= \frac{1}{\Delta x\sqrt{l}}\int_0^{\Delta x} dx e^{ikx} = \frac{1}{\Delta x\sqrt{l}}\left[\frac{e^{ikx}}{ik}\right]_0^{\Delta x} \\
&= \frac{1}{\Delta x\sqrt{l}}\frac{\left(e^{ik\Delta x} - 1\right)}{ik}
\end{aligned}$$

Thus, using the concrete form of R and $\Phi(k) = 1/k^4$, we can immediately compute $(R\Phi R^\dagger)_{jl}$:

$$(R\Phi R^\dagger)_{jl} = \frac{1}{\Delta x^2 l}\sum_k \frac{e^{ik(x_j - x_l)}}{k^6}\Big(e^{ik\Delta x} - 1\Big)\Big(e^{-ik\Delta x} - 1\Big)$$
$$= \frac{2}{\Delta x^2 l}\sum_k \frac{1-\cos(k\Delta x)}{k^6} e^{ik(x_j - x_l)} \tag{4.6}$$

This matrix now needs to be inverted, but it is actually trickier than it looks, and the inverse is not equal to the inverse of the Fourier coefficients. Why? Observe that the spatial gridpoints are both finite *and* discrete, which means that terms like $\sum_j e^{ix_j(k-q)}$ do not form kroenecker deltas δ_{kq}. We need to study this behaviour more closely. Take our grid of N points and two momenta $k = 2\pi n/l$ and $q = 2\pi m/l$:

$$\sum_{j=0}^{N-1} e^{i(k-q)x_j} = \sum_{j=0}^{N-1} \exp\left[i(\frac{2\pi(n-m)}{l})\Delta x j\right] \overset{\Delta x/l = 1/N}{=} \sum_{j=0}^{N-1} \exp\left[i(\frac{2\pi(n-m)}{N})j\right] \tag{4.7}$$

Now the sum is equal to N when $n = m$ as expected, but also when $(n-m)/N$ is an integer, i.e. $n = m \mod (N)$, making it hard to algebraically invert.

What is going on here, is that data space is a discrete periodic interval, which has a discrete Fourier transform (DFT). For a DFT, the momentum values k are the same as those for the continuous interval, albeit with a highest uniquely resolvable frequency known as the *Nyquist frequency*, which is equal to half of the sampling frequency. In this case, the Nyquist is $\frac{\pi}{\Delta x}$ and is denoted by f_N. Given that the matrix is indeed translation-invariant in data space, it must have *some* diagonal representation in the discrete Fourier transform, i.e. some scalar function of k for k now less than the Nyquist frequency. It turns out this representation can be found by resumming over multiples of the Nyquist frequency.

Lemma 4.1.2. *Given a regular, discrete grid of points $\{x_j\}$ for $j \in \{1, \ldots, N\}$ on a periodic interval, and a matrix of the form:*

$$A_{lj} = \sum_{k=-\infty}^{\infty} f(k) e^{ik(x_l - x_j)} \tag{4.8}$$

for $f(k)$ some function of k, it has a diagonal representation in the DFT Fourier space, given by:

$$A_{lj} = \sum_{|k|}^{f_N} \left(\sum_{b \in 2f_N\mathbb{Z}} f(k+b) \right) e^{ik(x_l - x_j)} = \sum_{|k|}^{f_N} g(k) e^{ik(x_l - x_j)} \tag{4.9}$$

where the new diagonal function $g(k)$ denotes the sum $\sum_{b \in 2f_N\mathbb{Z}} f(k+b)$.

Proof. Observe what happens when the infinite sum over k is partitioned into smaller sums shifted by multiples of the Nyquist frequency. For any x_i and x_j separated by a multiple of Δx and $b = 2\pi n/\Delta x$, we have $(k+b)(x_i - x_j) = k(x_i - x_j) + 2\pi n$. This factor of 2π then disappears in the complex exponential:

$$A_{lj} = \sum_{|k|}^{<f_N} \sum_{b \in 2f_N\mathbb{Z}} f(k+b) e^{i(k+b)(x_l - x_j)} \overset{\text{Nyquist}}{=} \sum_{|k|}^{f_N} \left(\sum_{b \in 2f_N\mathbb{Z}} f(k+b) \right) e^{ik(x_l - x_j)} \tag{4.10}$$

This resummed function is a diagonal function of the DFT frequencies $k < f_N$, and so must be the operator we were looking for. □

Note that depending on whether the number of data points is even or odd, the domain of $|k| < f_N$ changes. For odd N we use the convention that $k \in [-(N-1)/2, (N-1)/2]$ and if it's even we use $k \in [-N/2, N/2 - 1]$. Due to the physical analogy with Brillouin zones, we refer to the procedure of summing over multiples of the Nyquist frequency as *the sum over Brillouin zones.*

Now that we have obtained a representation of the operator which is diagonal in the DFT space, inverting becomes rather easy:

$$(R\Phi R^\dagger)^{-1}_{lj} = \frac{1}{N} \sum_{|k|}^{f_N} \frac{1}{\sum_{b \in 2f_N\mathbb{Z}} \Phi(k+b) |\widehat{B}(k+b)|^2} e^{ik(x_l - x_j)} \tag{4.11}$$

The factor of N comes from the different normalizations of the DFT and the regular fourier transform. Fourier modes in the DFT are normalized as $\frac{1}{\sqrt{N}} e^{-ikx_j}$. For the example model, we can now write down the formula for $(R\Phi R^\dagger)^{-1}_{lj}$:

$$\begin{aligned}
&\frac{\Delta x^2 l}{2N} \sum_{|k|}^{<f_N} \frac{e^{ik(x_l - x_j)}}{\sum_b [1 - \cos((k+b)\Delta x)](k+b)^6} \\
&\overset{\text{Nyquist}}{=} \frac{\Delta x^2 l}{2N} \sum_{|k|}^{<f_N} \frac{e^{ik(x_l - x_j)}}{(1 - \cos(k\Delta x)) \sum_b (k+b)^6}
\end{aligned} \tag{4.12}$$

Now it is time to compute the second part of the transport operator, $R\bar{U}\Phi R^\dagger$. Given that $\bar{U}$ is assumed to be diagonal in Fourier space, the previous lemma 4.1.1 applies, and the operator will also be diagonal in the DFT space, with a sum over Brillouin zones. With this information, we may now write down the general form of the update operator $T = R\bar{U}\Phi R^\dagger (R\Phi R^\dagger)^{-1}$:

$$T_{lj} = \sum_{|k|}^{f_N} \frac{\left(\sum_{b \in 2f_n\mathbb{Z}} \bar{U}(k+b)\Phi(k+b)|\widehat{B}(k+b)|^2 \right)}{\sum_{\hat{b} \in 2f_n\mathbb{Z}} \Phi(k+\hat{b})|\widehat{B}(k+\hat{b})|^2} e^{ik(x_l - x_j)} \tag{4.13}$$

The factor of $1/N$ is cancelled by a factor of N coming from the sum over spatial indices. For the toy model, we know the analytic form in Fourier space of all the objects involved. So, the transport operator can be computed by substituting into the above equation, giving an infinite algebraic series in Fourier space. This is summed numerically on a computer until it converges to within some degree of accuracy, yielding a function of k. The position space operator is then obtained by taking the inverse DFT of this function. Simple. For more complex models, computing the transport operator is typically done solely in position space.

For the toy model, $L = ivk$, so the above formula applies. Since the responses are constant, we can use the expansion $T = \mathbb{1} + \Delta t RL\Phi R^\dagger + \cdots$. For the meantime, we take the first order expansion and calculate $RL\Phi R^\dagger$:

$$(RL\Phi R^\dagger)_{lj} = \frac{2v\Delta x^2}{l} \sum_{|k|}^{f_N} (1 - \cos(k\Delta x)) \sum_{b \in 2f_n\mathbb{Z}} \frac{i(k+b)}{(k+b)^6} e^{ik(x_l - x_j)} \tag{4.14}$$

This means that to first order, the Δt term in the update operator is:

$$v\frac{2}{\Delta x^2 l}\frac{\Delta x^2 l}{2}\sum_{|k|}^{f_N}(1-\cos(k\Delta x))\sum_{\hat{b}}\frac{i}{(k+\hat{b})^5}\left(\frac{1}{(1-\cos(k\Delta x))\sum_b(k+b)^6}\right)e^{iq(x_l-x_j)}$$

$$\Rightarrow T = \mathbb{1} + \Delta t v\sum_{|k|}^{f_N}\left(\sum_{\hat{b}}\frac{i}{(k+\hat{b})^5}\frac{1}{\sum_b\frac{1}{(k+b)^6}}\right)e^{ik(x_l-x_j)} \tag{4.15}$$

4.1.1 Results

When implementing the toy model, it becomes immediately apparent that a first order forward scheme is unstable. In fact, any first-order forward scheme for advection in this general class of models will be unstable. We show this via a Von Neumann stability analysis. We first insert $\bar{U} = \mathbb{1}+v\Delta t\partial_x = 1+iv\Delta tk$ into eqn. 4.13 to give:

$$T(k) = \mathbb{1} + iv\Delta t\frac{\sum_{b\in 2f_N\mathbb{Z}}(k+b)\Phi(k+b)|\widehat{B}(k+b)|^2}{\sum_{\hat{b}\in 2f_N\mathbb{Z}}\Phi(k+\hat{b})|\widehat{B}(k+\hat{b})|^2} \tag{4.16}$$

Since the transport operator T is diagonal in Fourier space, the magnitude of it's eigenvalues are simply $|T(k)|$. Noticing that the momentum-dependent term is purely imaginary, we get:

$$|T(k)|^2 = 1 + (v\Delta t)^2\left[\frac{\sum_b(k+b)\Phi(k+b)|\widehat{B}(k+b)|^2}{\sum_{\hat{b}}\Phi(k+\hat{b})|\widehat{B}(k+\hat{b})|^2}\right]^2 \tag{4.17}$$

which is everywhere greater than one, for all nonzero values of momentum. Thus all Fourier modes will undergo exponential growth, and the code is unstable. Thus, to stabilize the code, we go to second order in time.

The results for a second order advection code are shown in figure 4.1. The toy model was implemented in position space, in anticipation of the case of a nontrivial velocity field, where a Fourier space representation is not possible. The update operator was expanded to second order, $U = 1+iv\Delta tk-v^2\Delta t^2k^2$, so that $T(k)$ will have another term analogous to eqn. 4.15 with $iv\Delta t(k+b) \rightarrow -v^2\Delta t^2(k+b)^2$. $T(k)$ was then computed by summing over Brillouin zones numerically, until the sum converged. The resulting expression was then put through an inverse DFT to yield a position-space matrix representation of

the transport operator.

Even at relatively low resolutions, for a smooth initial pulse, the IFD scheme delivers results which are indistinguishable from the analytic solution by eye. This is in contrast to a basic first-order finite difference scheme, which suffers from the artefact of numerical diffusion; the simulated field spreads out despite the fact that there is no diffusion term in the equations.

There are other ways of potentially stabilizing this toy code, like a Backward Euler scheme. Such a scheme was attempted and did indeed stabilize the code, however other undesireable properties remained. The backward Euler code developed extra maxima and minima which, thanks to the stabilizing property of the backward Euler, did not grow in size. But they did remain at a finite size and generally made things look bad. This is due to a general flaw in these codes. Namely, that they handle shocks quite poorly. Loosely defined, in a numerical simulation a *shock* is a change in the field that is sharp relative to the grid spacing Δx. Given that our vague goal is to simulate systems at very low resolutions (≈ 10 bins), at these resolutions, everything looks like a shock. To analyse the propagation of shocks, we need to not just compute the magnitude error of our code, but also the *phase error*, which is the difference in propagation velocities for the true solution vs. the simulated solution, as a function of frequency.

4.1.2 Phase error

Suppose we just consider straight advection, and observe the time evolution of a plane-wave e^{-ikx} in this system. We know that the full analytic time evolution operator is given by $U(k) = e^{ivk\Delta t}$, which simply multiplies the plane wave by a phase factor. This phase, the term in the exponent, represents the velocity. We need to define an operation which extracts this phase, which we label ω:

$$\tan(\omega) = \mathrm{Im}(e^{ivk\Delta t})/\,\mathrm{Re}(e^{ivk\Delta t}) = \sin(vk\Delta t)/\cos(vk\Delta t) \Rightarrow \omega = vk\Delta t \tag{4.18}$$

gives the desired result. The phase error is defined to be the difference between the true phase (ω) and the numerical phase (ω_n) actually obtained in the simulation [12]. This can be calculated in the analytically solvable case

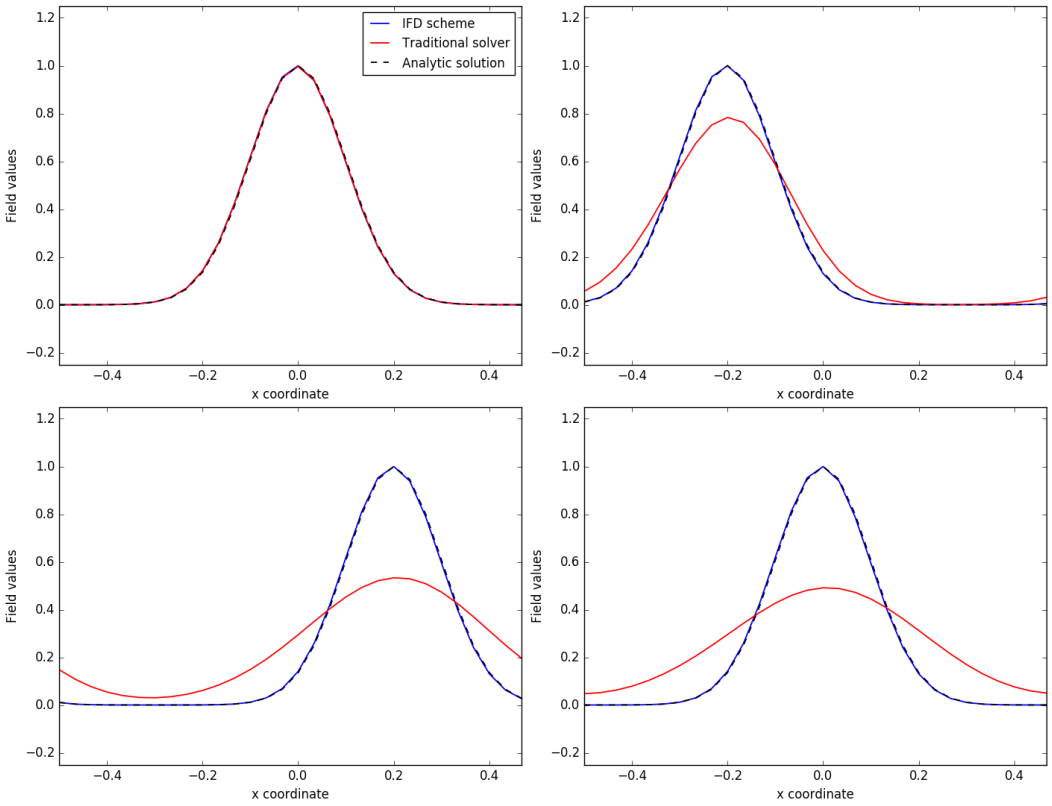

Figure 4.1: Simulated advection of an exponential pulse for a leftward directed velocity field, with periodic boundary conditions. The simulation space has 30 grid points and 500 timesteps, with a length, velocity and runtime of 1, in arbitrary units. The pictures detail one full cycle, at timesteps 0, 100, 400 and 499. The IFD code is rendered in blue, and the traditional first-order solver is shown in red. The analytic solution is rendered with a black dotted line. After one full cycle, the traditional solver has undergone significant broadening, whereas the IFD solution is indistinguishable from the analytic one.

by finding $\omega_n = \arctan(\mathrm{Im}(T(k))/\,\mathrm{Re}(T(k))$.

There is a mathematical subtlety that needs to be addressed here; in the generalized framework, the responses are simply integration over some regularly spaced grid of functions, whose form can be completely arbitrary. It has not yet been checked what the image in data space of a plane wave in signal space is. Pick a signal space plane wave $\phi(x) = e^{-ikx}$, and compute

$$R_j\phi = \int B(x - x_j)e^{-ikx}dx = e^{-ikx_j}\int B(x)e^{-ikx}dx = \widehat{B}(k)e^{-ikx_j} \tag{4.19}$$

This is neat; regardless of the shape of the response bins, plane waves in signal space show up as plane waves in data space, as long as the grid is translation-invariant. The plane wave will get scaled by a constant factor $\widehat{B}(-k)$ which depends on its momentum, but it is still a plane wave (remember that we are in the position-space representation here). Note that if k was above the Nyquist, the e^{ikx_j} term with the discrete x_j coordinates implies that the plane wave appears in the data with a frequency below the Nyquist, as expected. The above result allows us to perform a valid phase analysis by just looking at plane waves in data space.

We consider an expansion of U to arbitrary order, α, in time, because we can. For this model, $L = ivk$, so every odd power in the Taylor series will have a factor of i and every even power will not: this clearly forms truncated $\sin(v\Delta tk)$ and $\cos(v\Delta tk)$ series. The general phase error is then:

$$\begin{aligned}\omega_n = &\arctan\left(\frac{\sum_b\sum_{n=0}^{\alpha}\frac{(-1)^n[v\Delta t(k+b)]^{2n+1}}{(2n+1)!}\Phi(k+b)|\widehat{B}(k+b)|^2}{\sum_{\hat{b}}\Phi(k+\hat{b})|\widehat{B}(k+\hat{b})|^2}\right.\\ &\left.\times\frac{\sum_{\hat{b}}\Phi(k+\hat{b})|\widehat{B}(k+\hat{b})|^2}{\sum_b\sum_{n=0}^{\alpha-1}\frac{(-1)^n[v\Delta t(k+b)]^{2n}}{(2n)!}\Phi(k+b)|\widehat{B}(k+b)|^2}\right)\end{aligned} \tag{4.20}$$

The denominator of the first term cancels with the numerator of the second, giving:

$$\omega_n = \arctan\left(\frac{\sum_b\sum_{n=0}^{\alpha}\frac{(-1)^n[v\Delta t(k+b)]^{2n+1}}{(2n+1)!}\Phi(k+b)|\widehat{B}(k+b)|^2}{\sum_{\hat{b}}\sum_{n=0}^{\alpha-1}\frac{(-1)^n[v\Delta t(k+\hat{b})]^{2n}}{(2n)!}\Phi(k+\hat{b})|\widehat{B}(k+\hat{b})|^2}\right) \tag{4.21}$$

Note that this equation is for odd powers in α. For even powers, the term in the denominator has the higher power of α. This equation may seem rather intimidating, and is somewhat hard to interpret at first. Begin by noticing that due to the sum over Brillouin zones, it must be periodic in $k \in [-f_N, f_N]$. This expression should attempt to approximate the graph of vk, i.e. a straight line. Unless the prior and responses were chosen terribly, the code should be reasonable enough that waves don't propagate backwards. i.e. to the right of the origin ($k = 0$), the numerical phase will be everywhere positive, and to the left the phase will be everywhere negative. Periodicity implies that the phase must then drop to zero at $k = \pm f_N$, and the phase error approaches a maximum. An example of this behaviour is shown in the phase velocity plots for the toy model, figure 4.2.

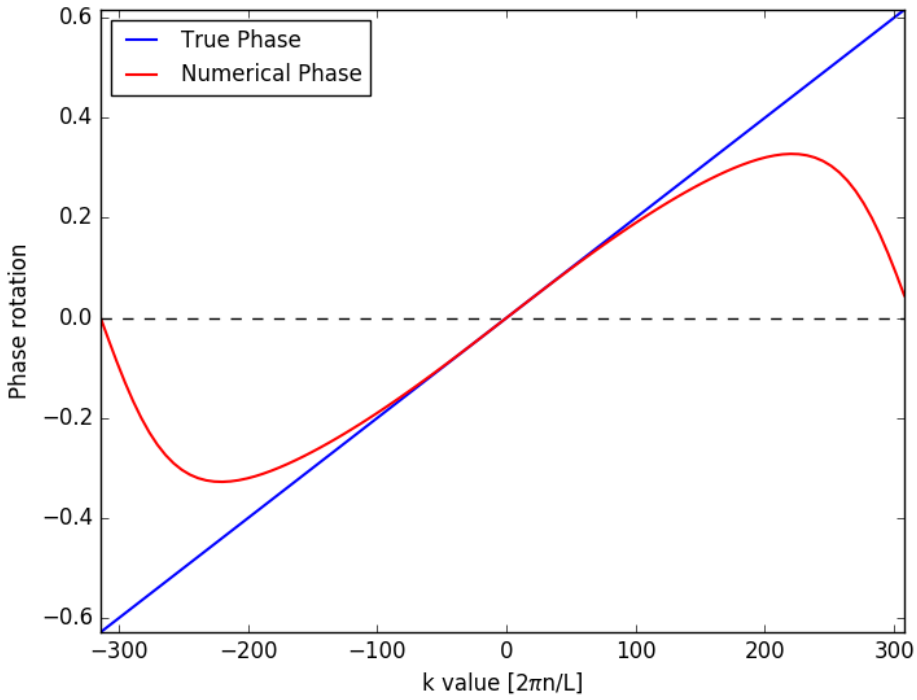

Figure 4.2: Phase error of the second-order toy model over the domain $[-f_N, f_N]$, performed on a 100 point grid with length and velocity equal to 1 in arbitrary units. The numerical phase diverges quite strongly as the momentum approaches the Nyquist.

It is this feature which is responsible for the poor handling of shocks in this class of models. A feature that is sharp on the scale of the grid length must be resolved by high k values, but at these high k values, the structures do not propagate at all. This leads to the development of unwanted oscillations in the simulation.

The argument that the phase error approaches a maximum at high momenta is just that: an argument. The k values are discrete, rather than continuous, so there's no reason to say that the phase error cannot be reduced arbitrarily by making the phase plot approach a sawtooth function centred at zero. The bad phase error is rather a general and somewhat persistent feature of these models. Though this in itself is no great tragedy, many models such as the Lax-Wendroff scheme, which is a second order scheme, also suffer from this problem [1]. In fact, modelling shocks effectively is one of the most challenging and interesting areas of numerical hydrodynamics.

There is a very relevant theorem here, called the *Godunov Theorem* ([12, p. 280]), which states that *any linear algorithm for solving partial differential equations, with the property of not producing new extrema, can be at most first order.*[2] Given that we are exclusively working with linear codes, we cannot expect to escape these spurious oscillations, but hopefully by analysing eqn. 4.21, they can be reduced as much as possible.

Given the formula for the expansion to arbitrary order in time, it should be compared to the hypothetical ideal behaviour of an order α expansion, $U = \sum_{n=0}^{\alpha} (\Delta t L)^n / n!$:

$$\omega_{\text{ideal}} = \arctan\left(\frac{\sum_{n=0}^{\alpha} \frac{(-1)^n [v\Delta t(k)]^{2n+1}}{(2n+1)!}}{\sum_{n=0}^{\alpha-1} \frac{(-1)^n [v\Delta t(k)]^{2n}}{(2n)!}}\right) \tag{4.22}$$

Comparing the numerical phase and ideal phase, the discrepancy is clearly caused by the sum over Brillouin zones, if higher frequency modes could be damped or eliminated, then the code would approach the ideal behaviour. The $B(x)$'s will very often be compactly supported, so by the uncertainty principle, their Fourier transforms will be nonzero all the way to infinity.

[2]Note that we have not yet figured out to which spatial order these codes are accurate. This will be discussed later.

Instead, imagine that the magnitude of the prior dropped off sharply for momenta above the Nyquist, this would cut out the influence of all the higher Brillouin zones. We conclude: *the lower the probability the prior assigns to modes above the Nyquist, the better the phase error.*[3] What is happening is that for any structure in the data below the Nyquist frequency, the prior associates a small but nonzero probability that it actually came from a structure above the Nyquist frequency, which has an entirely different propagation velocity, which then interferes with the simulation.

It is unfortunately time for a detour into matters of interpretation. We cannot just set the momentum-space representation of the transport operator to be $T(k) = \sum_{n=0}^{\alpha}(\Delta t L(k))^n/n!$ and get ideal results, because this will never generalize to nonconstant velocity fields. We remind the reader again that the simulations are not being carried out in Fourier space, it's just that for translation-invariant schemes they have such a representation, and an error analysis on this representation should give an idea as to what happens when nontrivial models are considered. The trickier problem however, is that of prior selection.

There is an unresolved ambiguity in IFD as to whether the prior represents our honest beliefs about the behaviour of the system, or is simply a tunable parameter for constructing a subgrid model. If the former is true, then the prior cannot be adjusted so that it drops off sharply above the Nyquist, as that would mean we are deliberately selecting our beliefs about the system depending on the resolution at which we are observing it; which is rather illogical. Consider instead that the prior is fixed, and drops off to below some desired level after some frequency k_0, then for best results, we must *pick* a resolution such that the Nyquist frequency is greater than k_0. Thus the phase error equation tells us something intuitively obvious: if we believe that the behaviour of a system is mostly described by structures above a certain length scale, then our simulations should have a resolution of at least that length scale. This assumption is implicit in normal finite-difference codes. The difference with IFD however, is that our assumptions are explicitly coded into the update equations, and we get poor results when the simulations vi-

[3]So, we can minimize the phase error *if* the prior kills frequencies above the Nyquist. Proving the converse statement, that the only way to take the phase error to zero is to assign zero probability to higher frequencies, will be very hard.

olate these assumptions. This will turn out to be a recurring theme.

If the prior is regarded as a tunable parameter, then all bets are off. It should be set to damp high-frequency modes as aggressively as possible, such that the reconstructions have little structure on scales above the below resolution. Intuition would say that a bit of higher structure should be kept when going to nonconstant velocity fields, as the advantage of subgrid models comes from the fact that they suppose the existence of extra structure between gridpoints.

It appears that if we want good results, we need to discuss the limit of high resolutions, which means it's time to discuss consistency and convergence.

4.1.3 Consistency

We suspect from equation 4.21, that in the limit of high resolutions, our simulations will approach reality. This is the same as proving that the transport operator is consistent. This can be shown in the translation-invariant case by only adding two extra assumptions: that the response bins $B(x)$ are compactly supported and have bounded Fourier transform, and that $U(k)\Phi(k) \to 0$ as $k \to \pm\infty$.

The bounded Fourier transform requirement will almost always be true for any reasonable response. It holds for all smooth, compactly-supported functions, by the *Paley-Wiener theorem* [18].

Theorem 4.1.3 (Paley-Wiener (weakened))**.** *If f is a smooth, compactly supported function on $\mathbb{R}$, then it's Fourier transform $\hat{f}(k)$ can be bounded by*

$$|\hat{f}(k)| \leq C_n(1 + |k|)^{-n} \tag{4.23}$$

for all $n \in \mathbb{Z}^+$, and some positive constant C_n

The box responses, despite not being smooth, also have bounded Fourier transform.

To prove consistency, we ask if $T(k) \to U(k)$ in the limit of high resolution. In the Fourier representation, comparing the action of T and U is easy, despite the fact that they technically act on different spaces. The definition of

consistency (def. 3.3.3) requires that the operators converge for any function in signal space that we pick, but not that they converge at the same rate for all functions[4]. Pick a basis of signal space consisting of Fourier modes, then pick out a single mode of frequency k. As the resolution increases, eventually the Nyquist frequency will be greater than k ($f_N = \pi/\Delta x$). Past this resolution, $T(k)$ and $U(k)$ can both be thought of as acting on the same space. $T(k)$ contains a time-order approximation $\bar{U}(k)$ to $U(k)$. If we can show that as $\Delta x \to 0$, $T(k) \to \bar{U}(k)$, then in the joint limit of time and space resolution going to infinity, then T approaches U.

Hence we need that for each fixed k, $T(k) \to \bar{U}(k)$ in Δx, but the convergence doesn't need to be uniform in k. For the translation-invariant response, we want to increase the number of bins while simultaneously decreasing their width. Given some initial resolution Δx_0 for which all the bins fit evenly inside the interval, we pick an integer λ that goes from 1 to infinity, then we set $\Delta x = \Delta x_0/\lambda$. This guarantees that the new set of scaled bins $B(\lambda x) = B_\lambda(x)$ fits evenly inside the interval.

The compact support property of the bins allows us to exploit the fact that up to a normalization constant, the coefficients $B(k)$ of the discrete values of k in the Fourier series of the bins are the same as the values at k in the continuous Fourier transform of B. This is because if a function is compact, it doesn't matter if it is integrated over a finite or infinite interval. This allows us to exploit the scaling property of the Fourier transform $\hat{B_\lambda}(k) = \frac{1}{\lambda}B(k/\lambda)$. The normalization constant λ and the constant from the differing normalizations of the Fourier transform cancel due to the division in eqn. 4.13.

Now observe the sum over the Brillouin zones. We sum over $b \in 2\mathbb{Z}f_N$ where $f_N = \pi/\Delta x$ and thus $f_N^\lambda = \pi\lambda/\Delta x_0$ and $b^\lambda = 2\pi n\lambda/\Delta x_0$ for $n \in \mathbb{Z}$. We don't scale the prior with λ (our beliefs about the system shouldn't change depending on the resolution of our equipment) and look at the upper term of eqn. 4.13:

$$\sum_{n\in\mathbb{Z}} \bar{U}(k + \frac{2\pi n\lambda}{\Delta x_0})\Phi(k + \frac{2\pi n\lambda}{\Delta x_0})|B(\frac{1}{\lambda}(k + \frac{2\pi n\lambda}{\Delta x_0}))|^2 \tag{4.24}$$

[4]This would be impossible, for *any* numerical scheme on a discretized grid, the Nyquist frequency dictates there is a function which the grid cannot resolve

The λ term inside B can be absorbed to give:

$$|B(\frac{1}{\lambda}(k+\frac{2\pi n\lambda}{\Delta x_0}))|^2 = |B(\frac{k}{\lambda}+\frac{2\pi n}{\Delta x_0}))|^2 \tag{4.25}$$

We want that in the limit of $\Delta x \to \infty$, the higher terms in the sum vanish, leaving only terms in the first Brillouin zone. The prior and bin terms in the numerator and denominator of 4.13 would then cancel, leaving just $\bar{U}$, i.e. we want

$$\begin{aligned}\frac{\lim_{\lambda\to\infty}\sum_{n\in\mathbb{Z}}\bar{U}(k+\frac{2\pi n\lambda}{\Delta x_0})\Phi(k+\frac{2\pi n\lambda}{\Delta x_0})|B(\frac{k}{\lambda}+\frac{2\pi n}{\Delta x_0})|^2}{\lim_{\lambda\to\infty}\sum_{m\in\mathbb{Z}}\Phi(k+\frac{2\pi m\lambda}{\Delta x_0})|B(\frac{k}{\lambda}+\frac{2\pi m}{\Delta x_0})|^2} \\ = \frac{\bar{U}(k)\Phi(k)|B(\frac{k}{\lambda})|^2}{\Phi(k)|B(\frac{k}{\lambda})|^2} = \bar{U}(k)\end{aligned} \tag{4.26}$$

We can expect that for all terms with $n \neq 0$, in the limit of $\lambda \to \infty$ each term goes to zero because $\bar{U}(k)\Phi(k)$ goes to zero at large $|k|$. Therefore we want to swap the limit and the infinite sum.

Given a sequence of functions $f_n(\lambda)$, swapping the limits $\lim_{\lambda\to\infty}\sum_{n=0}^{\infty} f_n(\lambda) = \sum_{n=0}^{\infty}\lim_{\lambda\to\infty} f_n(\lambda)$ is possible if and only if the sequence of functions converges uniformly. In our case, we throw out the $n = 0$ term, and consider the positive and negative n halves of the sum separately, but present only the case of positive n, as the working for negative n is nearly identical. Our goal is that the functions converge to zero, so we state: a function converges uniformly to zero if for any positive ϵ, there is an N such that $\forall n \geq N$, $|f_n(\lambda)| < \epsilon$ for all values of λ. The last requirement is the crucial part.

For our purposes, $f_n(\lambda) = \bar{U}(k+\frac{2\pi n\lambda}{\Delta x_0})\Phi(k+\frac{2\pi n\lambda}{\Delta x_0})|B(\frac{k}{\lambda}+\frac{2\pi n}{\Delta x_0})|^2$. We bound the whole function $|B(\frac{k}{\lambda}+\frac{2\pi n}{\Delta x_0})|^2 < C$ for some constant C, which we may do by assumption. The bin terms do not vanish in the limit of large λ, because as the bins become narrower, their Fourier transforms widen out, at the exact same rate as the Nyquist frequency is increasing. Hence we will need the property $\Phi(k)\bar{U}(k) \to 0$ to do all of the work.

This requirement means that for $|k|$ large enough $\bar{U}(k)\Phi(k)$ can be bounded by some monotonically decreasing function of $|k|$, call it $g(|k|)$. We start by finding a bound for $\lambda = 1$, and then show that this bound holds for all λ.

For $\lambda = 1$, and the desired ϵ bound, we can pick some n large enough such that we are in this decreasing regime, hence $|\bar{U}(k + \frac{2\pi n\lambda}{\Delta x_0})\Phi(k + \frac{2\pi n\lambda}{\Delta x_0})|C < g(k + \frac{2\pi n\lambda}{\Delta x_0}) < \epsilon$. For higher λ and large n, $|k + \frac{2\pi n}{\Delta x_0}| < |k + \frac{2\pi n\lambda}{\Delta x_0}|$, and since we have taken n to be large enough that we are in the decreasing regime, the $g(k)$ bound also holds. Thus the bound holds for all lambda.

Thus the sequence of functions is uniformly convergent, and equation 4.26 holds. We can now say

Theorem 4.1.4. *For a translationally-invariant scheme, whose response bins are compactly supported with bounded Fourier transform, and some time-order approximation $\bar{U}(k)$ to $U(k)$, where k denotes momentum, then the scheme is consistent provided* $\lim_{k\to\infty} \bar{U}(k)\Phi(k) = 0$.

Important to note is that we only need $\bar{U}(k)\Phi(k) \to 0$, not $U(k)\Phi(k)$. For derivative operators s.t. $U = \exp(\Delta t\partial_x) = \exp(i\Delta tk)$ or similar, this would require that the prior is smooth. Using the approximated time expansion, the prior, $\Phi(x)$, only needs to be as many-times differentiable as the order of the expansion dictates.

4.1.4 Error scaling

Using our formula for the transport operator, we can estimate the data space error. We stay in the exact same limit we were before, scaling with λ, and picking a fixed $\phi(x)$ that is some plane wave, and exploiting the fact that below the Nyquist, signal space and data space are comparable. This means that the one-step error in data space E_d is given by:

$$E_d = \left| \frac{\sum_{n\in\mathbb{Z}} \bar{U}(k + \frac{2\pi n\lambda}{\Delta x_0})\Phi(k + \frac{2\pi n\lambda}{\Delta x_0})|B(\frac{k}{\lambda} + \frac{2\pi n}{\Delta x_0})|^2}{\sum_{m\in\mathbb{Z}} \Phi(k + \frac{2\pi m\lambda}{\Delta x_0})|B(\frac{k}{\lambda} + \frac{2\pi m}{\Delta x_0})|^2} - U(k) \right| \tag{4.27}$$

We pick some order in our expansion $\bar{U} = \sum_{p=0}^{N} (\Delta tL)^p/p!$, and expand out the error in terms of powers of L, as in equation 3.23.

$$E_d \leq \sum_{p=0}^{N} \frac{\Delta t^p}{p!} \left| \frac{\sum_{n\in\mathbb{Z}} L(k)^p(k + \frac{2\pi n\lambda}{\Delta x_0})\Phi(k + \frac{2\pi n\lambda}{\Delta x_0})|B(\frac{k}{\lambda} + \frac{2\pi n}{\Delta x_0})|^2}{\sum_{m\in\mathbb{Z}} \Phi(k + \frac{2\pi m\lambda}{\Delta x_0})|B(\frac{k}{\lambda} + \frac{2\pi m}{\Delta x_0})|^2} - L^p(k) \right| \tag{4.28}$$

We analyse the scaling of each term individually, but start with the Δt term, and then generalize. In the limit of high resolutions, we expect the sum over the higher Brillouin zones to become small, so

$$\sum_b L(k+b)\Phi(k+b)|B(k+b)|^2 \approx L(k)\Phi(k)|B(0)|^2 + \epsilon(k) \tag{4.29}$$

The denominator should also admit such an expansion. We then seek to bound the whole fraction. The scaling of the error can only be estimated if we know the scaling behaviour of the prior and $L(k)$. So, suppose that as $|k|$ becomes large, $\Phi(k)$ can be bounded by some decreasing power law in k, $|k|^{-\alpha}$ for α positive. We also assume that $L(k)$ can be bounded by some $|k|^\beta$ for β positive, as L will typically be a derivative operator, with $\partial_x^n = (ik)^n$. There will be constants of proportionality, but they don't matter. We factorize the numerator as:

$$L(k)\Phi(k)|B(\frac{k}{\lambda})|^2 + \sum_{n\neq 0} L(k+\frac{2\pi n\lambda}{\Delta x_0})\Phi(k+\frac{2\pi n\lambda}{\Delta x_0})|B(\frac{k}{\lambda}+\frac{2\pi n}{\Delta x_0})|^2 \tag{4.30}$$

Once again, in the limit $\lambda \to \infty$, the bin terms just converge to $B(0+\frac{2\pi n}{\Delta x_0})$ which isn't useful for bounding anything so we replace it with the uniform bound C from before. We then bound

$$\begin{aligned}
&\left|\sum_{n\neq 0} L(k+\frac{2\pi n\lambda}{\Delta x_0})\Phi(k+\frac{2\pi n\lambda}{\Delta x_0})|B(\frac{k}{\lambda}+\frac{2\pi n}{\Delta x_0})|^2\right| \\
&\leq C^2 \sum_{n\neq 0}\left|L(k+\frac{2\pi n\lambda}{\Delta x_0})\Phi(k+\frac{2\pi n\lambda}{\Delta x_0})\right| \\
&\leq C^2 \sum_{n\neq 0}\left|\frac{2\pi n\lambda}{\Delta x_0}\right|^{\beta-\alpha} = \lambda^{\beta-\alpha}C^2\sum_{n\neq 0}\left|\frac{2\pi n}{\Delta x_0}\right|^{\beta-\alpha}
\end{aligned} \tag{4.31}$$

The term inside the sum is independent of the scaling. Therefore we have an object which scales as $\mathcal{O}(\lambda^{\beta-\alpha})$, which we identify with $\mathcal{O}(\Delta x^{\alpha-\beta})$, since $\Delta x = \Delta x_0/\lambda$. We repeat the argument with the denominator, and get a term proportional to $\mathcal{O}(\Delta x^\alpha)$. We use the Taylor expansion for $1/(1-\epsilon) \approx 1+\epsilon+\epsilon^2+\cdots$ to expand the denominator in eqn. 4.28 into something more useful:

$$\frac{1}{\Phi(k)|B(\frac{k}{\lambda})|^2 + \mathcal{O}(\Delta x^\alpha)} = \frac{1}{\Phi(k)|B(\frac{k}{\lambda})|^2} + \mathcal{O}(\Delta x^\alpha) \tag{4.32}$$

Thus the whole error expression scales as:

$$\left(L(k)\Phi(k)|B(\frac{k}{\lambda})|^2 + \mathcal{O}(\Delta x^{\alpha-\beta})\right)\left(\frac{1}{\Phi(k)|B(\frac{k}{\lambda})|^2} + \mathcal{O}(\Delta^\alpha)\right) - L(k) \quad (4.33)$$
$$= L(k) + \mathcal{O}(\Delta x^\alpha) + \mathcal{O}(\Delta x^{\alpha-\beta}) + \mathcal{O}(\Delta x^{2\alpha-\beta}) - L(k) = \boxed{\mathcal{O}(\Delta x^{\alpha-\beta})}$$

The other $\mathcal{O}$ terms cancel because only the term with the worst scaling (lowest power) matters. For a term of order Δt^p in eqn. 4.28, we repeat the argument and get

$$\boxed{E_d \propto \mathcal{O}(\Delta t^p \Delta x^{\alpha-p\beta})}. \quad (4.34)$$

The total error scaling is determined by the worst scaling of any of the individual terms.

We see from this formula that going to higher orders in Δt decreases the spatial order. This is fine for $L = \partial_x$ because the total order remains the same, but for higher derivatives, the spatial order decreases faster in p than the time order increases, decreasing the total order and making the scaling worse. This can be thought of in the following way: if the prior drops off as some power α, then it is only α times differentiable, so it is not smooth. In the limit of high resolutions, the bins approximate something non-smooth, and thus there is a maximum possible order in the expansion. This doesn't necessarily imply that higher orders in time give worse IFD simulations. We know this is not true for the case of straight advection, where second order codes are stable, and first order codes are not. Furthermore, we already know that for normal finite-difference derivatives, higher orders aren't better when the fields are rough.

This formula can be immediately used to get an error estimate on the toy model. Note that the boxes in this model are not smooth, and the Fourier transforms have a $1/k$ scaling independent of the box width, which shows up in the denominator of the transport operator (eqn. 4.15) in addition to the $1/k^4$ from the prior. Thus, for the second order implementation, the denominator on the transport operator scales as k^6 and $L = ik$, so

$$E_d = \mathcal{O}(\Delta t \Delta x^{-5}) + \mathcal{O}(\Delta t^2 \Delta x^{-4}) \quad (4.35)$$

We now know that the derivative that we have been implementing is fifth-order, and this confirms something that experience has shown in the codes;

they appear to be implementing something analogous to higher order derivatives. When we compare the first derivative operator for the toy code, vs the standard fifth order derivative, we see that they are quite similar, as shown in figure 4.3. This isn't to say that they are the same. The error scaling we just derived only kicks in in the limit in which the shape of the bins is lost. At lower resolutions, the structure of the bins will provide some additional subgrid structure, which should be advantageous, if the bins are intelligently chosen. Important to note also is that the operators produced in IFD still have nonlocal effects, i.e. it is not just the resolution, but the resolution compared to the size of the domain which has an effect on the resulting operators.

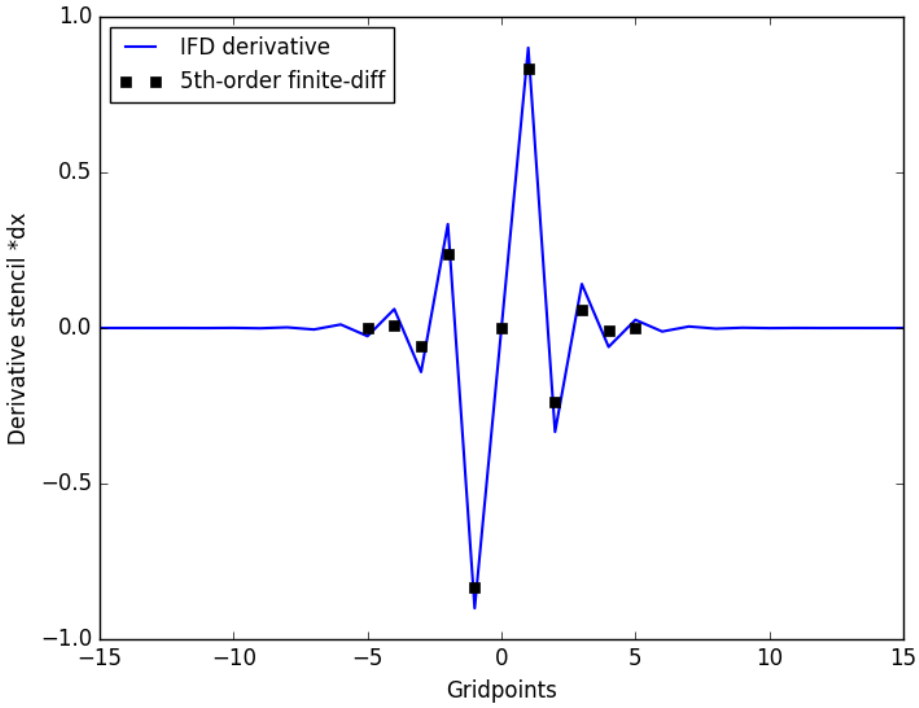

Figure 4.3: Comparison of the numerical approximation to the first derivative produced by IFD, and the standard numerical fifth-order derivative approximation. The two are quite similar, though the IFD derivative extends over a slightly larger distance. The IFD derivative was calculated on a periodic interval of 50 gridpoints, using the toy model, which has a $1/k^4$ prior.

4.2 Extension to nonconstant velocities

The code was extended to cover nonconstant velocity fields, and nonperiodic boundary conditions. This means that the transport operator had to be computed numerically. Adding a diffusion term to the system would have been a possible extension, but because diffusion simply destroys fine structure, the addition of diffusion actually makes numerical simulations *easier*. In fact, adding artificial diffusion is a common way to stabilize misbehaving codes [12].

We picked an approximation to signal space whose resolution was higher than that of data space by a factor of 100. A position space representation of the prior and responses was then chosen, and the $(R\Phi R^\dagger)_{ij}$ matrix was computed according to the convolutional action of the prior:

$$(R\Phi R^\dagger)_{ij} = \int B_i(x)\left(\int \Phi(x-y)B_j(y)dy\right)dx \tag{4.36}$$

The domain of definition of the convolution depends on the boundary conditions. Thus the inverse matrix $(R\Phi R^\dagger)^{-1}$ contains information about the geometry. To compute the second matrix $RL\Phi R^\dagger$, we constructed a matrix representation of $L = \partial_x v(x)$ on signal space by taking a standard central difference numerical derivative, i.e. $Lf(x) = (f_{i+1}v_{i+1} - f_{i-1}v_{i-1})/(100\Delta x)$. The rest of the computation was done with normal matrix algebra. For stability, a second order time expansion was used, $\bar{U} = \mathbb{1} + \Delta t L + \Delta t^2 L^2/2$.

We present two pairs of example outputs, one with periodic boundary conditions (fig.4.4), and one without (fig.4.5). The rest of the simulation parameters were kept as similar as possible to aid comparison. The prior was chosen to be a position-space representation of $1/(k^4 + m^4)$, with the width adjusted to that $\Phi(x)$ effectively drops to zero within the length of the simulation domain, so that convolution over $\Phi(x)$ was valid in both spaces. The mass term m was kept very small and was only used to regularize the integrals. The response bins were the square boxes from the previous toy model. A velocity field valid on both spaces was also chosen, $1 + c\sin(2\pi x/l)$ for a tuneable constant c, whose value was kept < 1 to ensure the velocity field has a consistent direction. The field flows from right to left, and the sinusoidal character means that there is a velocity minimum in the left-hand corner of

the box. This compresses the fields and causes shocks.

For the case with boundary conditions, the numerical convolution was computed by allowing $\Phi(x-y)$ to run off the edge of the simulation domain. Thus the $R\Phi R^\dagger$ matrix produced is not translation-invariant in data space, and shows boundary effects, though explicit boundary conditions have not yet been applied. This matrix is still invertible, and $RL\Phi R^\dagger$ was computed similarly. In this scheme, the matrix inversions carry information about the geometry, which means the transport operator automatically accounts for the existence of a boundary. The boundary conditions themselves were then imposed by fixing the values of the gridpoints on each end, as with a normal simulation. These boundary conditions injected an exponential pulse at one end, and let it pass through at the other. As stated earlier, this approach to boundary conditions is somewhat ad-hoc. Various other approaches were attempted, but this method functioned the best.

With both examples, there is no analytic solution to compare to, so they are compared to an ordinary finite-difference scheme carried out at 100 times resolution in time and space. This is regarded as the 'true' solution.

Both examples show the same weakness; a vulnerability to shocks as predicted by the analytic analysis. When the incoming pulses become compressed, the solutions develop extra maxima and minima. This can be expected from Godunov's theorem, now that the previous chapter has established that these IFD schemes are of very high order. Aside from the spurious oscillations, both examples retain the shape of the pulse better than the ordinary forward-difference code. Note that the periodic example was run with a factor of $c = 1/1.2$ (arbitrary units), whereas the nonperiodic code was run with $c = 1/1.5$. The second code is not able to handle compression as well, as edge effects coming from the $(R\Phi R^\dagger)^{-1}$ matrix in the transport operator induce an additional oscillation which originates on the boundary.

This poor handling of compression can also be seen in light of the Bayesian interpretation. We have picked a prior such that the field is correlated over some length scale that is on the order of multiple bins. During compression, any initial variation in the field is squeezed to within the scale of a few bins, and thus there is sharp variation over a length which the prior says should be smooth. This violates the prior, and delivers poor results. We have again

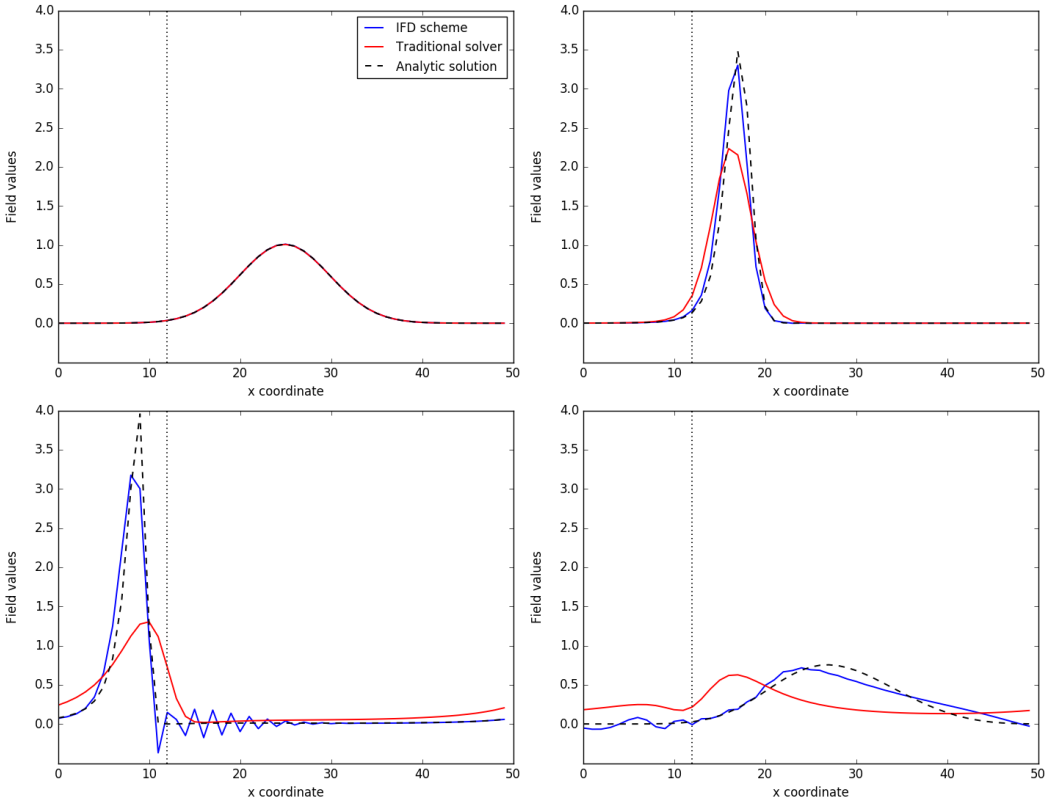

Figure 4.4: Simulated advection of an exponential pulse for a nonconstant leftward directed velocity field, with periodic boundary conditions. The simulation space has 50 grid points and 1000 timesteps, with a length and velocity of 1 and runtime of 2, in arbitrary units. The minimum in the velocity field is denoted by a dotted vertical black line. The first panel shows the initial conditions. The second panel shows the beginning of compression. The third panel shows the pulses after passing through the compression region; The IFD code has kept its shape better than the traditional solver, but has developed spurious oscillations due to the poor handling of shocks. The final panel shows the pulses after one full cycle; the IFD code still displays artefacts, but otherwise performs well.

touched on the theme encountered at the end of subsection 4.1.2.

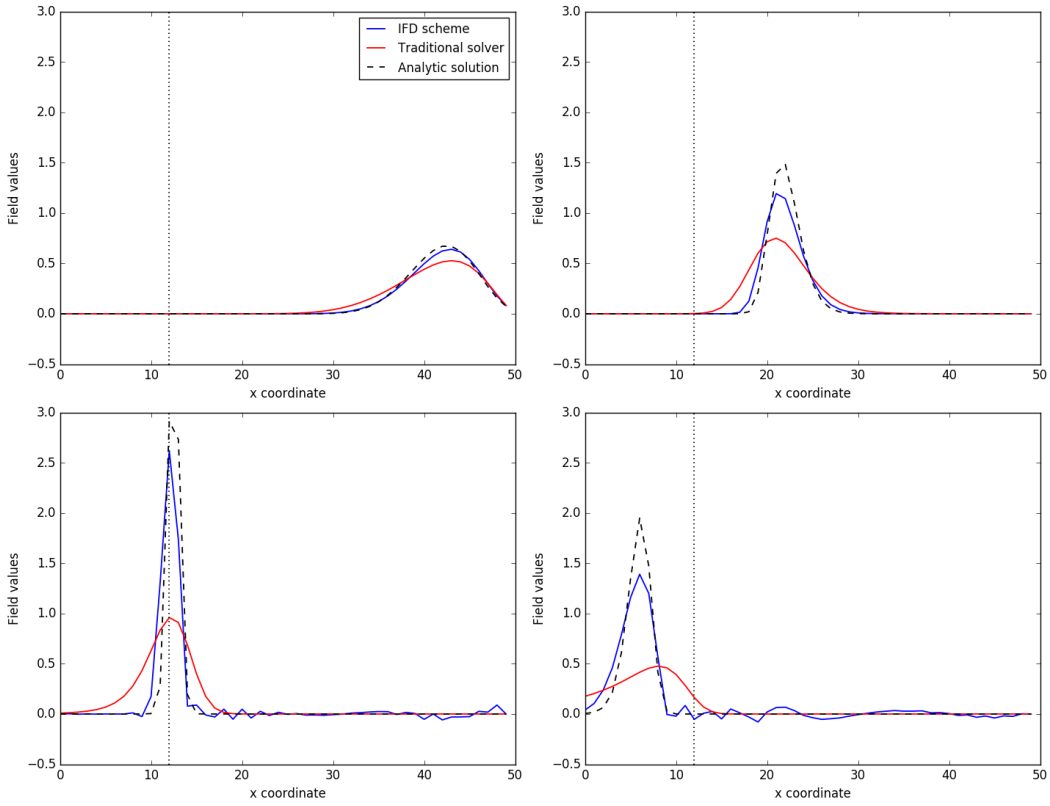

Figure 4.5: Simulated advection of an exponential pulse for a nonconstant leftward directed velocity field, with nonperiodic boundary conditions. The simulation space has 50 grid points and 1000 timesteps, with a length and velocity of 1 and runtime of 2, in arbitrary units. The minimum in the velocity field is denoted by a dotted vertical black line. The first panel shows the pulse being injected from the right hand side. The second panel shows the beginning of compression. The third panel shows the pulse at the minimum of the velocity field. As with the previous figure, the IFD has retained its shape better than the traditional solver, but has developed oscillations, as well as a new oscillation which collects on the boundary. The final panel shows the pulse leaving the compression region. Overall, the IFD solver performs better than the traditional solver, but its performance is much worse compared to the constant-velocity case.

Chapter 5

SPH-like schemes

5.1 A brief introduction to Smooth Particle Hydrodynamics

During the course of this project, a second type of model was attempted. Despite being unsuccessful, it is presented here for two reasons. The first is that it serves to highlight the importance of the theme discovered with the previous models: if the equations of motion contradict the prior, then the simulations begin to show errors. The second reason is that we believe it shows more promise than the translation-invariant schemes. For this problem, we once again attempt to solve the basic problem of 1D advection, but now with an added diffusion term: $\partial_t f(x,t) = \partial_x(v(x)f(x,t)) + K\partial_x^2 f(x,t)$, where K is some constant diffusion coefficient.

This model takes inspiration from so-called "smoothed particle hydrodynamics" (SPH) methods[19][20][21] [22].In such models, rather than solving the equations of motion on a fixed set of N gridpoints, the field in question (typically a fluid) is instead modelled by a set of N "particles" which represent a sampling of the true field, each posessing a location and velocity. At each timestep, an estimate of the continuous field is reconstructed via some *smoothing kernel* denoted by $W(x)$ which is positive, has a peak at zero, and drops off to 0 with increasing x. For a collection of particles at locations x_i with mass m, the reconstructed density is given by

$$\rho(x) = \frac{m}{N} \sum_i W(x - x_i) \tag{5.1}$$

Typical choices for the smoothing kernel are Gaussians, or polynomial splines which fall to zero inside some compact region [22]. The density reconstruction in an SPH code converges to the true density in the limit of $N \to \infty$ provided the width of the smoothing kernel $W(x)$ is scaled with $1/N$ [19]. From this reconstructed density field, using the equations of motion of the field, $\rho(x)$, and the particle velocities u_i, the force on the ith particle F_i is then calculated, which is used to update $u_i \to u_i + \Delta t F_i$. This new velocity is then used to update the positions $x_i \to x_i + \Delta t u_i$, and the cycle then repeats.

There are many advantages to this type of model, the most relevant of which is that it can be thought of as a model with an adaptive simulation grid. The variable particle locations automatically increase the spatial resolution of the code in regions where there is most activity. This makes them extremely useful for cosmological problems in which there are large density gradients, such as cosmic ray simulations.

5.2 The IFD approach

The analogy between IFD and SPH is immediately apparent. Both frameworks take a sampling from the field, from which a reconstruction is derived, which is then fed into the equations of motion. Typically the smoothness prior will take the form of a convolution over some integration kernel, so we can immediately make the comparison $\Phi(x-y) \Leftrightarrow W(x-y)$. The "particles" would then obviously take the form of a set of delta functions, representing point-measurements of the field i.e. $R_i = \int \delta(x - x_i)$. The locations of these delta functions would then be shifted in time according to the equations of motion of the field.

We call our new model the *SPH-like* code. According to the data/response equivalence as discussed in section 3.2, the responses can be updated in such a way that it is equivalent to updating the data. The model we are about to define is a 'hybrid' IFD model, in which the computation is split between the data and the response. The particles represent point measurements of the field, thus their mass can vary depending on the value of the field at that

location. Advection is modelled by moving the locations of the response delta functions, and diffusion is modelled by redistributing mass between the data points (particles). This is in contrast to an SPH code, where the particles have fixed masses, and diffusive effects are modelled by adding a repulsive force between them.

The SPH-like code needs to be expressed in equations. The responses will be dependent on time and space, so we write R_i^j with lower indices denoting spatial coordinates, and upper indices denoting time coordinates. The responses are then given by $R_i^j = \int \delta(x - x_i^j)$ for some finite set of locations $\{x_i^j\}$ at timestep t^j. The prior Φ will be constant in time, and will be given by a convolution over some kernel $\Phi(x)$. The time evolution operator is expanded to first order $\bar{U} = \mathbb{1} + \Delta t \partial_x v(x) + \Delta t K \partial_x^2$. Because the terms will be broken up, we denote $\partial_x v(x)$ by L_A for the advection part, and $K\partial_x^2$ by L_D for the diffusion part, with associated U_A and U_D time evolution operators. For dynamically changing responses, we write down the transport operator:

$$T^j = R^{j+1}(\underbrace{1 + \Delta t v(x)\partial_x}_{U_A} + \underbrace{\Delta t K \partial_x^2}_{L_D})W^j \tag{5.2}$$

Note that W^j is the Wiener filter at time t^j, $W^j = \Phi R^{j\dagger}(R^j \Phi R^{j\dagger})^{-1}$, and *not* the SPH smoothing kernel. We define $R^{j+1} = R^j U_A^{-1} = R^j(1 + \Delta t v(x)\partial_x)^{-1}$, so that the advection part is absorbed into the evolving responses. Now we do a little bit of lazy mathematics: the action of $R^j U_A^{-1}$ on a field corresponds to taking the field, advecting it backward in time, and then measuring it with the delta functions located at the positions x_i^j. This is the same as taking the locations of the delta functions, and advecting them forward in time: $x_i^{j+1} = x_i^j + \Delta t v(x_i^j)$. Now that we have defined the updating rule for the responses, we can write the transport operator as:

$$\begin{aligned} T^j &= \underbrace{R^{j+1}(1 + \Delta t v(x)\partial_x}_{=R^j U_A^{-1} U_A = R^j} + \Delta t K \partial_x^2)W^j \\ &= \Big(R^j + R^{j+1}\Delta t L_D\Big)W^j = \mathbb{1} + \Delta t R^{j+1} L_D W^j \end{aligned} \tag{5.3}$$

Where the last equality follows from the fact that $R^j W^j = \mathbb{1}$. One full cycle of the scheme is then as follows:

1. Construct the Wiener filter using the response locations at timestep j, $W^j = \Phi R^{j\dagger}(R^j \Phi R^{j\dagger})^{-1}$.

2. Reconstruct the field at timestep j based on the Wiener filter and the data: $\phi(x) = W^j d^j$.

3. Apply the diffusion operator $L_D = K\partial_x^2$ to the reconstructed field.

4. Advect the response locations $x^{j+1} = x_i^j + \Delta t v(x_i^j)$, such that $R_i^{j+1} = \int \delta(x - x_i^{j+1})$.

5. Apply the new response to the diffused field: $R^{j+1}\phi(x) = R^{j+1} L_D W^j d^j$.

6. Construct the new data: $d^{j+1} = T^j d^j = (\mathbb{1} + \Delta t R^{j+1} L_D W^j) d^j$.

This method should hopefully give an IFD scheme that has all the advantages of an SPH code. This scheme has a changing response at each timestep, which means the Wiener filter will need to be continually recomputed. This involves performing operations on objects in signal space, which should in general be computationally expensive; however the choice of responses and some clever computational tricks render this quite easy. Since the prior is static in time, we can store a single representation of $\Phi(x)$ at a very high resolution. We can exploit a handy property of convolution here: for two functions f and g, $\partial_x(f * g) = (\partial_x f) * g = f * (\partial_x g)$. So, to know the action of ∂_x^2 on $\Phi * \phi$, we only need to know $\partial_x^2 \Phi(x)$, which we also store a representation of. The Δt part of the transport operator is then computed in two parts,

$$R^{j+1} L_D W^j = \underbrace{R^{j+1} L_D \Phi R^{j\dagger}}_{\text{part one}} \underbrace{(R^j \Phi R^{j\dagger})^{-1}}_{\text{part two}} \tag{5.4}$$

Since the responses are delta functions, these two matrices can be computed extremely easily, using the fact that $(R\Phi R)^\dagger_{ik} = \Phi(x_i - x_k)$ and similarly, $(R\partial_x^2 \Phi R^\dagger)_{ik} = K\partial_x^2 \Phi|_{x_i - x_k}$. In the actual algorithm, this means that the two matrices are computed by just indexing into the stored copies of Φ and $\partial_x^2 \Phi(x)$ respectively, which is quite cheap in terms of computation time. This is however hard on memory, given that any approximation to signal space must have much higher resolution than that of data space in order to get good results.

For a cutting-edge simulation, we obviously would like that the actual data simulation space is at the limits of our available memory. This code uses a $50\times$ factor of resolution, and a translationally-invariant prior, which means the stored prior is 50 times larger than the actual data being simulated. This is at least better than the case with a general position-space prior, of the form $\int \Phi(x, y)dy$, which would be larger by a factor of 50^2. There are two ways around this scaling problem: the first is to simply use a prior whose analytic form, and that of its derivatives, is known. If that is not possible, we can anticipate eventually going to three dimensions, where we can pick an isotropic prior $\Phi(\vec{x}) = \Phi(|x|)$, and then wait until the simulation gets large enough such that $50N < N^3$. Since the code in this report is just a proof of concept, we store the full prior and simply tolerate the terrible memory performance.

We avoid the costly process of inverting $R\Phi R^\dagger$ at each timestep, as matrix inversion scales as $\mathcal{O}(N^3)$, and instead compute the cheaper (and more stable) problem of solving the linear system of equations $(R\Phi R^\dagger)v = d^j$ for the specific d^j at each timestep, and some unknown vector v. This only scales as $\mathcal{O}(N^2)$. We then apply $R_{i+1}L_D\Phi R_i^\dagger$ to this vector v, thus computing $R^{j+1}L_D W^j d^j$ without having to compute the full transport operator.

There are some additional differences between this code and an SPH code. For an SPH code, observation of eqn. 5.1 shows that the reconstruction is roughly analogous to $\Phi R^\dagger$ in the IFD framework, and is thus local. Our code differs by the factor of $(R\Phi R^\dagger)^{-1}$, thus rendering the reconstructions slightly nonlocal. The nonlocality is supposed to be advantageous, as it exploits the correlation structure of the field to give better reconstructions. Furthermore, in our code, the particles do not have their own velocities. This is due to the fact that we are modelling advection, so the particle velocity is entirely determined by the vector field which pushes it. If this code were to be extended to model a more complex fluid-dynamics problem, where the velocity of the fluid itself comes into the equations, then the inclusion of a particle velocity would become necessary.

5.3 Results and post-mortem

The SPH-like code was implemented, and is was seen to be extremely unstable. It diverges sharply when compressed, which was exactly what it was designed not to do. As with the previous examples, we took a simulation domain with periodic boundary conditions, and a sinusoidally-varying velocity field $1 + c\sin(2\pi x)$ for some tunable constant c, and arbitrary units. This model of course included a constant diffusion term. An example output is shown in figure 5.1, which was run with $c = 1/2.9$ and a diffusion coefficient of $K = 1/60$. The simulation used 20 test particles, and the initial conditions were a randomly sampled smooth field, whose correlation structure was the same as the prior. The prior was an exponential which drops to zero within the length of the simulation space. Various other priors were attempted, all with uniformly bad performance.

This code is stable as long as the particles remain evenly spread out, but as soon as multiple particles are compressed into a small space, the reconstruction diverges, and the particle values themselves then diverge. This behaviour is surprising, given that the no-noise Wiener filter always has the property that $RW = \mathbb{1}$, which means that the reconstruction always perfectly agrees with the data at the exact locations of the particles. However it can be seen that the reconstruction begins to oscillate wildly in the space *between* particles, and thus the gradients of the reconstruction at the particle locations become large, which bleeds into the data in the next timestep under the action of ∂_x^2.

In human terms, the failure of this code can be described as follows: In the IFT framework, the reconstruction of a field given some data typically gives very accurate results if the data is considered to be likely according to the prior. However, if the data is considered unlikely, then the reconstructions can become very, very bad. An example of this is shown in figure 5.2, which compares the reconstructions of a field given a set of point samples, for both a "likely" and "unlikely" data set. The reconstruction was performed using a basic Gaussian smoothness prior. The first result was random noise sampled with a very similar power spectrum to the prior, whereas the second was a localised bump of width much narrower than the correlation length of the prior.

We can now interpret this observed divergence under strong compression.

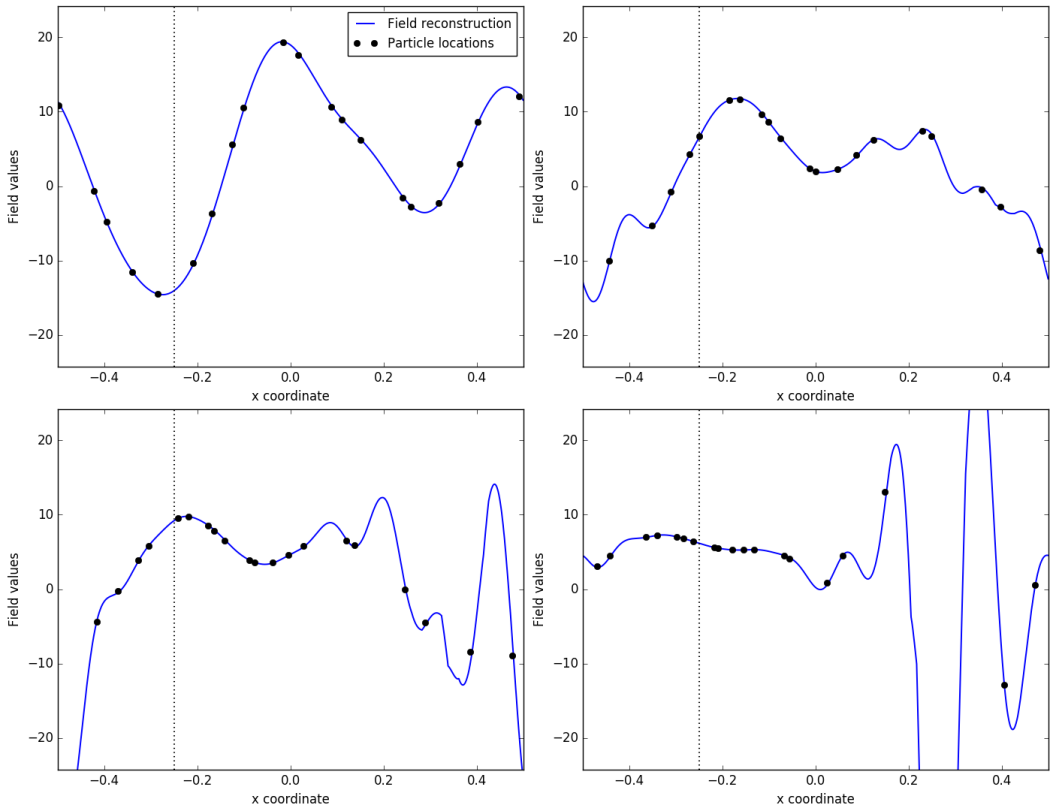

Figure 5.1: Output of the SPH-like code on box of length 1 and a runtime of 1 in arbitrary units, with periodic boundary conditions. The velocity field is sinusoidal and leftward-directed, with the minima denoted by a dotted vertical black line. Outputs shown are for timeteps 0, 106, 148, and 238. The figure shows the development of instabilities in the reconstruction as the initially well-spaced population of particles gets compressed in the left hand side of the box. Note that a violation of mass conservation can clearly be seen.

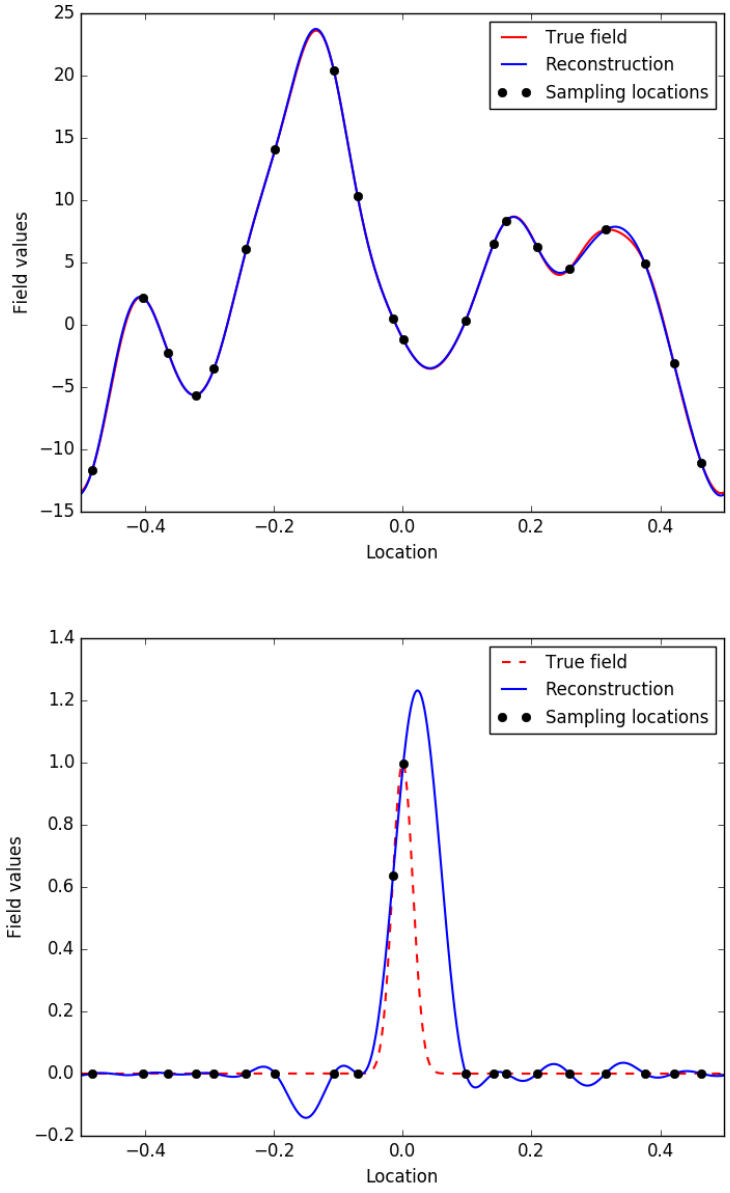

Figure 5.2: Comparison of reconstructions for two different fields given a set of irregular point measurements, reconstructed with the same prior. The prior has a Gaussian correlation structure. The plot on the left shows a randomly sampled field which also has Gaussian spatial correlations of width slightly smaller than that of the prior; the reconstruction is scarcely distinguishable from the original field. The plot on the right shows the reconstruction of a single narrow bump of width much smaller than the prior. This field is considered unlikely according to the prior, and displays numerous artefacts such as overshooting, and extra maxima and minima.

The simulation takes an initial thermally agitated field, with variations on the scale of the correlation length of the prior. This field is then compressed so that there is significant variation within the supposed correlation length. Thus the data is considered unlikely, and the reconstruction diverges. We have yet again touched on our common theme that the equations of motion must be consistent with the prior. In this case, we have supposed translation-invariance, which supposes that no locations are special, despite the fact that the compression region is clearly special.

In mathematical terms, the problem lies in the inversion of the $R\Phi R^\dagger$ matrix. Indeed, given almost any static prior, we can show that there will always be a scale at which the matrix $(R\Phi R^\dagger)^{-1}$, and thus the Wiener filter, diverges.

Theorem 5.3.1. *Given any prior which takes the form of an integration over a continuous, symmetric kernel $\Phi\phi(x) = \int \Phi(x,y)\phi(y)dy$ over some metric space, and a time-varying response of the form $R_i^j = \delta(x - x_i^j)$ for some finite set of locations $\{x_i^j\}$ at a set of timesteps $\{t_j\}$, then there will always be some compression scale for which the determinant of $(R\Phi R^\dagger)^{-1}$ becomes arbitrarily large.*

Proof. Suppose hypothetically that one had a pair of response locations that were identical, so that $x_1 = x_2$, then $(R\Phi R^\dagger)_{ik} = \Phi(x_i, x_k)$ will have two identical columns $(R\Phi R^\dagger)_{i1} = \Phi(x_i, x_1) = \Phi(x_i, x_2) = (R\Phi R^\dagger)_{i2}$, and thus its determinant will be zero.

The rest follows as a simple consequence of continuity. Suppose there is any sequence of response locations $\{x_i^j\}$ in time for $j \to \infty$ which converge to some locations x_i, such that one pair approaches each other: $||x_1^j - x_2^j|| \to 0$ as $j \to \infty$. Continuity of the kernel $\Phi(x,y)$ then implies that the sequence of matrices $\Phi(x_i^j, x_k^j)$ approaches some matrix M_{ij} in the euclidean matrix norm, which is defined as $||M|| = \sqrt{\sum_{ik} |M_{ik}|^2}$. That is to say $\sqrt{\sum_{ik}(\Phi(x_i^j, x_k^j) - M_{ij})} \to 0$. This is true because each entry of $\Phi(x_i^j, x_k^j)$ converges by continuity of $\Phi(x,y)$. M_{ij} has two equal columns and thus it's determinant is zero. Since the determinant function is also continuous in the euclidean norm, $\det(\Phi(x_i^j, x_k^j)) \to \det(M) = 0$, and thus $\det((\Phi(x_i^j, x_k^j)^{-1}) \to \infty$. Hence, the determinant of $(R\Phi R^\dagger)^{-1}$ can be made arbitrarily large by choosing the response locations close enough.

□

Note that in this limit of high compression, the matrix elements themselves do not become smaller. This shows that the matrix inversion is blowing up due to a true degeneracy, rather than a cosmetic scaling which could potentially be cancelled by the $\Phi R^\dagger$ term in the Wiener filter. This instability caused by the divergence of the Wiener filter can be reduced by introducing enough diffusion into the model such that structure is eliminated faster than it is compressed. However the amount of diffusion required is extremely large, and also does not counter the inaccuracy introduced into the equations by the unstable Wiener filter.

There is another flaw with the SPH-like code, caused by the use of delta functions. Observe that, for diffusion on a convolutional prior, the matrix elements on the main diagonal of $RL\Phi R^\dagger$ are given by $\partial_x^2 \Phi(x)|_{x=0}$. This means that the matrix entries depend entirely on the derivative of the prior at a single point, which is in contrast to box-responses or bump functions, where the matrix elements become averages over some region, and so are less dependent on the exact form of the prior. This did indeed have a noticeable effect on the simulations, where we experimented with various different reasonable-looking priors, and obtained very different diffusion speeds. This is in contrast to the toy code in the previous section, which was much more robust to small changes in the prior. Robustness is considered desirable, because often the correlation structure will be nothing more than an educated guess.

A mathematical justification for this is available, if we imagine a hypothetical case where the particles are evenly spaced out and the velocity field is constant. Then the system is translation-invariant and we can apply eqn. 4.13. Delta functions have a flat Fourier spectrum, so the transport operator becomes:

$$T(k) = \frac{\sum_{b \in 2f_N\mathbb{Z}} \bar{U}(k+b)\Phi(k+b)}{\sum_{\hat{b} \in 2f_N\mathbb{Z}} \Phi(k+\hat{b})} \tag{5.5}$$

Making small changes to the prior $\Phi(x)$ in a small region about the origin, by the uncertainty principle, corresponds to modifying the frequencies at infinity. Given that the derivative operators form polynomials in k, this drastically changes the convergence properties of the above sum. This is not the case when the bins are smooth and compactly supported, as they decay

faster than any polynomial by the Paley-Wiener theorem. This gives the other codes their robustness in response to a changing prior. An extreme example of this flaw is when one inserts a seemingly reasonable momentum space prior: $\Phi(k) = 1/(k^2 + m^2)$ which has a position-space representation of $\Phi(x) = e^{-m|x|}$; this is not differentiable and thus the code is unuseable. Once again, we expect the insight we gain in the translation-invariant case to hold in the non-invariant case.

The last problem with this code is that the no-noise Wiener filter on delta functions doesn't preserve positivity of the fields. This makes the reconstructions unsuitable for simulating everywhere positive quantities, as was the case for the original goal of simulating cosmic ray streaming. Suppose the prior $\Phi(x)$ is everywhere greater than zero (i.e. not just positive in the operator sense). If $R\Phi R^\dagger$ has any elements off the main diagonal, which it must in order to be useful, then these elements are all positive. Thus, the inverse matrix must have some negative entries in order to cancel the off-diagonal terms and yield the identity matrix. It is these negative elements which result in the fields not being everywhere positive. This of course is not a proof, rather an explanation of the observed behaviour. On a deeper level, this comes from the fact that the Gaussian regime of IFD requires signal space to be a vector field. Therefore for every possible field, it's negative must also be possible and is in fact equally likely to occur. A common approach in IFT is to enforce positivity by taking the logarithm of the fields, and supposing the statistics of the logged fields are Gaussian. However if we want to extend to more complex systems such as fluid dynamics, log fields become very inconvenient.

5.4 Suggested improvements

Despite the fact that the SPH-like code was the least successful of the two presented, it is the author's opinion that the second model shows the most promise. We know that the translation-invariant codes are implementing something analogous to very high-order finite difference schemes. The field of fixed-grid linear finite-difference solvers has been around for a very long time, and it is not suspected that there is much room for anything novel. SPH solvers are in contrast rather new, and already fit well with the IFD formalism. If the IFD reconstructions can be stabilized, then they could eas-

ily be inserted into existing SPH solvers, such as GADGET-2 [22], replacing the typical SPH reconstruction with a Bayesian one, and then proceeding as normal with the simulation.

We previously saw that in the SPH-like model, a static prior will always give a Wiener filter that diverges under enough compression. In regions of high compression, very often SPH codes will also have to dynamically resize the width of the smoothing kernel, so that the density estimates remain accurate [22]. Thus we know it is at least feasible to dynamically update basic features of the prior, such as its correlation width. This could be a potential fix for this problem. Alternatively, diffusive effects could be modelled by by a repulsive force between particles, thus preventing them from approaching each other too closely. The model would then need to be applied to a more complex fluid dynamics problem, where there are more forces than just advection and diffusion, so that the reconstructions have a purpose.

A more elegant alternative to dynamically changing the prior width, would be to derive a more sensible starting prior. Taking a translation-invariant prior on a system which is not translationally invariant is in hindsight a terrible idea. This project has shown that prior selection is a very important part of IFD, and the sensitivity of the delta grid models to changes in the prior indicates that priors should not be chosen arbitrarily, but rather after careful consideration of the system at hand. This is of course much harder, and it is not known what a properly-selected prior for a system with nonconstant velocity fields should look like.

Due to the delta-function responses, even with a more fitting prior, the simulations will still be very vulnerable to small changes in said prior. It is possible however to have responses which move under advection, but still have a volume. This would be something similar to a moving-mesh code. Instead of having delta function responses, one could take the moving set of point locations from before, and define a set of box responses $R = \int B_i(x)dx$ around the points via a Voroni cell decomposition. A similar process is carried out in [23], and we therefore know that recomputing the responses in this fashion at every timestep is at least computationally feasible.

However, even with better responses and priors, we still have to solve the problem that the reconstructions do not preserve positivity, which will for-

ever cause headaches whenever we try and model an everywhere-positive field. Under the assumption that going to log-fields is impractical, we may be required to leave the Gaussian regime of IFT, and consider more exotic reconstructions.

Chapter 6

Conclusions

6.1 Summary

This project did not produce a prototype cosmic ray simulation as intended. Many initial attempts at producing workable IFD codes were found to be unsuccessful. The source of the persistent problems with the attempted algorithms was not tracked down until a theoretical analysis of error and stability was undertaken. It took many months to realize that all the various unsuccessful codes could be united under a single unified framework of translation-invariant problems. This realization allowed us to develop the general form of the transport operator in momentum space, eqn. 4.13, from which all the major conclusions of this work followed. Along the way, a number of smaller results were proven, such as the noise-invariance, as well as implementing minor fixes, such as taking the KL divergence in the right direction.

Crucially, we showed that translation-invariant schemes in IFD are all consistent under a broad set of assumptions, which proves the basic validity of the IFD approach. A working prototype was then developed, which functions similarly to a higher-order linear finite difference scheme. The prototype model does indeed perform much better than basic finite-difference schemes. However the translationally-invariant schemes will generally perform poorly when simulating shocks and other numerically-interesting problems. Furthermore, given their similarity to existing linear codes, which are already extensively studied, it is not expected that this class of schemes will deliver

particularly novel results.

It is believed that the most promising code, despite its present flaws, is the SPH-like code. It can easily be integrated into current SPH algorithms, which are currently a very active field. It is also believed that there are techniques already used to stabilize SPH codes (adaptive smoothing lengths) which can be used to stabilize the prototype algorithm. It is hoped that the Bayesian nature of the reconstructions will provide an edge over the traditional field reconstructions in SPH codes.

Though both types of algorithms explored in this project displayed the same general flaw; if the behaviour of the system contradicts the prior, then the results are bad. In the case of the SPH-like code, they can be catastrophically bad. This topic warrants its own discussion.

6.2 Prior selection

Every simulation scheme relies on a set of assumptions. The difference between traditional simulation schemes and IFD is that our assumptions are hard-coded into the simulation by our choice of prior. The experience gained in this project shows that this is a double-edged sword; if the data to be simulated is considered likely given the prior, then the simulations perform well; if the equations of motion are inconsistent with the prior, then the results are bad, or even divergent.

We have learned that correct prior selection is probably the primary consideration when developing a simulation scheme using IFD. There is still an unresolved question as to whether the prior is simply a tunable parameter which can be arbitrarily adjusted to give good simulations, or if it is an honest Bayesian belief about the system under study. The challenge going forward, is how to select a sensible prior in advance of performing the simulation.

Given any initial prior, one can naively evolve it in time such that it is consistent with the equations of motion, by picking $\Phi(t) = U(t)\Phi_0 U(t)^\dagger$. This ensures that for any initial field ϕ_0, the evolved field has the same probability of occurring according to the prior probability distribution. Strictly speaking, this is the logical thing to do anyway for a dynamic system. How-

ever this naive approach is not computationally feasible. For any interesting system, the exact time evolution is unknown, so the prior evolution must be simulated. Any approximation to signal space must be of much higher dimension than that of data space in order to be useful. If m is the dimension of the approximation to signal space, and n the dimension of data space, then evolving the prior correctly in time would involve simulating an $m \times m$ dimensional object, for the sole purpose of getting better simulations of an n-dimensional object.

Cheap fixes for this problem may be found. As a suggested improvement for the SPH-like code, the correlation with of the prior could be scaled by some factor λ such that $\Phi(x) \to \Phi(\lambda x)$ in regions where the particle density is high. This is already known to be technically feasible. This fix is somewhat 'ad hoc' though, and means that the prior becomes more of a tunable parameter rather than expressing our beliefs about the system.

It appears that the best way forward is to derive a prior based on considerations of the equations of motion of the system itself. Philosophically speaking, whenever we perform a simulation, information theoretic or not, we assume that the mathematical equations of motion accurately describe the system under consideration, otherwise we wouldn't be doing the simulation. So, up to some factors such as boundary conditions, and the influence of random noise/thermal agitation, the prior should be able to be deduced from the equations of motion themselves.

For example, if we can associate an energy to a field configuration, $E[\phi]$, then we can describe our initial ignorance about the system by a thermodynamic prior: $\mathcal{P}(\phi) = \exp(-\beta E[\phi])$. For a typical Klein-Gordon field with mass, then $E[\phi] = \int \phi^\dagger(m^2 + \nabla)\phi dx$, yielding the familiar $\Phi(k) = 1/(k^2 + m^2)$ prior.

There is a danger here, we are taking the equations of motion and feeding them back into a simulation of the equations of motion, meaning that this process may simply be introducing redundant information. As an extreme example, for a linear system, the Green's functions give the correlation structure of the field (in time and space), but finding the Green's functions for a given system is equivalent to solving the system itself. So it appears that for nontrivial systems we should not expect to get the true correlation structure, but rather a very rough guess obtained by a consideration of symmetries,

boundary conditions, and the physical scales of the relevant formulae.

6.3 Final remarks

Information field dynamics is an extremely young theory; this thesis marks only the fourth publication on the topic. As with all new inventions, a significant amount of blood, sweat and tears will be required before it finally starts to generate novel results. In hindsight, attempting to develop a prototype simulation scheme for the challenging problem of cosmic ray transport was probably over-ambitious. Even on an extremely reduced problem, implementing a scheme that worked at all proved to be a significant challenge. We did not arrive at a scheme which worked better than an ordinary finite-difference scheme until a few months before the due date. However, future work can proceed armed with a set of theoretical results showing what works and what does not, and a general framework for analysing whatever problems may emerge. We also have a roadmap for the future, identifying a promising solution which doesn't work at present, but has a clearly prescribed set of fixes. The issue of prior selection has turned out to be both crucial and rather tricky. The limited time requirements of a masters thesis means that the job of solving this issue must fall on the shoulders of some other (un)fortunate student.

Bibliography

[1] C.P. Dullemond and H.H. Wang. Lecture numerical fluid dynamics. Course given at Heidelberg university, retrieved from `http://www.ita.uni-heidelberg.de/~dullemond/lectures/num_fluid_2009/Intro.pdf`, 2009.

[2] Torsten A. Enßlin. Information field dynamics for simulation scheme construction. *Phys. Rev. E*, 87:013308, Jan 2013.

[3] Torsten A. Enßlin, Mona Frommert, and Francisco S. Kitaura. Information field theory for cosmological perturbation reconstruction and nonlinear signal analysis. *Phys. Rev. D*, 80:105005, Nov 2009.

[4] Christian Münch. Mathematical foundation of Information Field dynamics. Master's thesis, Technische Universität München, Germany, 2014. Retrieved from: `http://wwwmpa.mpa-garching.mpg.de/~ensslin/research/research_IFT.html#Publications`.

[5] E.T. Jaynes and G.L. Bretthorst. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

[6] N. Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series: With Engineering Applications*. Massachusetts Institute of Technology : Paperback series. M.I.T. Press, 1964.

[7] Anderson D.R. Burnham K.P. *Model Selection and Multimodel Inference*. Springer-Verlag, New York, 2002.

[8] R.H. Leike and T.A. Enßlin. Optimal belief approximation. Unpublished paper, retrieved from `https://arxiv.org/abs/1610.09018`, 2017.

[9] John Duchi. Lecture notes for statistics 311/electrical engineering 377. Stanford University, retrieved from `https://stanford.edu/class/stats311/Lectures/full_notes.pdf`, 2016.

[10] R. Corless and N. Fillion. *A Graduate Introduction to Numerical Methods: From the Viewpoint of Backward Error Analysis*. SpringerLink : Bücher. Springer New York, 2013.

[11] P. D. Lax and R. D. Richtmyer. Survey of the stability of linear finite difference equations. *Communications on Pure and Applied Mathematics*, 9(2):267–293, 1956.

[12] C. Hirsch. *Numerical Computation of Internal and External Flows: Fundamentals of Computational Fluid Dynamics*. Butterworth-Heinemann. Elsevier/Butterworth-Heinemann, 2007.

[13] J. Skilling. Cosmic ray streaming—i effect of alfvén waves on particles. *Monthly Notices of the Royal Astronomical Society*, 172(3):557–566, 1975.

[14] Enßlin, T., Pfrommer, C., Miniati, F., and Subramanian, K. Cosmic ray transport in galaxy clusters: implications for radio halos, gamma-ray signatures, and cool core heating. *A & A*, 527:A99, 2011.

[15] Enßlin, T. A., Pfrommer, C., Springel, V., and Jubelgas, M. Cosmic ray physics in calculations of cosmological structure formation. *A & A*, 473(1):41–57, 2007.

[16] Francesco Miniati. Cosmocr: A numerical code for cosmic ray studies in computational cosmology. *Computer Physics Communications*, 141(1):17 – 38, 2001.

[17] Francesco Miniati, Dongsu Ryu, Hyesung Kang, and T. W. Jones. Cosmic-ray protons accelerated at cosmological shocks and their impact on groups and clusters of galaxies. *The Astrophysical Journal*, 559(1):59, 2001.

[18] M. Reed and B. Simon. *Methods of Modern Mathematical Physics*. Number v. 2 in Methods of Modern Mathematical Physics. Academic Press, 1975.

[19] Monaghan J.J. Gingold R.A. Smoothed particle hydrodynamics - theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181(11):375–389, 1977.

[20] Lucy L.B. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal*, 82(12):1013–1024, 1977.

[21] Volker Springel and Lars Hernquist. Cosmological smoothed particle hydrodynamics simulations: the entropy equation. *Monthly Notices of the Royal Astronomical Society*, 333(3):649–664, 2002.

[22] Volker Springel. The cosmological simulation code gadget-2. *Monthly Notices of the Royal Astronomical Society*, 364(4):1105–1134, 2005.

[23] Philip Mocz, Rüdiger Pakmor, Volker Springel, Mark Vogelsberger, Federico Marinacci, and Lars Hernquist. A moving mesh unstaggered constrained transport scheme for magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 463(1):477–488, 2016.

Appendix A

Derivation of the Gaussian KL divergence

Lemma. *For two Gaussians $\mathcal{G}(\phi - a, A)$ and $\mathcal{G}(\phi - b, B)$, the KL divergence between the two* $\mathrm{KL}(\mathcal{G}(\phi - a, A)||\mathcal{G}(\phi - b, B))$*is given by:*

$$\frac{1}{2}\Big(\mathrm{Tr}[\ln(BA^{-1}) - \mathbb{1} + B^{-1}A] + \langle b - a| \, B^{-1} \, |b - a\rangle \Big) \tag{A.1}$$

Proof.

$$\begin{aligned}
\mathrm{KL}(\mathcal{G}(\phi{-}a, A)||\mathcal{G}(\phi{-}b, B)) &= \int \mathcal{D}\phi \mathcal{G}(\phi{-}a, A)\Big[\ln(\mathcal{G}(\phi{-}a, A)){-}\ln\Big(\mathcal{G}(\phi{-}b, B)\Big)\Big] \\
= \frac{1}{|2\pi A|^{1/2}} \int \mathcal{D}\phi \exp\left(-\frac{1}{2}\langle \phi - a| \, A^{-1} \, |\phi - a\rangle\right) &\Bigg[\left(-\frac{1}{2}\langle \phi - a| \, A^{-1} \, |\phi - a\rangle {-} \frac{1}{2}\ln(2\pi \det A)\right) \\
&- \left(-\frac{1}{2}\langle \phi - b|^{\dagger} \, B^{-1} \, |\phi - b\rangle - \frac{1}{2}\ln(2\pi \det B)\right)\Bigg]
\end{aligned} \tag{A.2}$$

The $\ln(2\pi \det A)$ terms may be factored out of the integral as they are independent of the fields. This just leaves a multiple of the integral of the Gaussian $\mathcal{G}(\phi - a, A)$, which equals one. We also exploit the identity that for a positive matrix $\ln(\det A) = \mathrm{Tr}(\ln A)$, which simplifies the equation to:

$$= \frac{1}{2}[\mathrm{Tr}(\ln(B) - \ln(A))] + \tag{A.3}$$
$$\frac{1}{2|2\pi A|^{1/2}} \int \mathcal{D}\phi \exp\Big(- \langle \phi - a| \, A^{-1} \, |\phi - a\rangle \Big) \Bigg[\Bigg(\langle \phi - b| \, B^{-1} \, |\phi - b\rangle \Bigg) - \Bigg(\langle \phi - a| \, A^{-1} \, |\phi - a\rangle \Bigg) \Bigg]$$

For the $\langle \phi - a| \, A^{-1} \, |\phi - a\rangle$ term, we can shift to a new variable $\phi' = \phi - a$, which brings the integral into the form an expectation value:

$$\mathbb{E}\Big[\langle \phi'| \, A^{-1} \, |\phi'\rangle \Big]_{\mathcal{G}(\phi', A)} = \mathbb{E}\Big[\mathrm{Tr}(A^{-1} \, |\phi'\rangle \, \langle \phi'|) \Big]$$
$$= \mathrm{Tr}(A^{-1} \mathbb{E}\Big[\, |\phi'\rangle \, \langle \phi'| \, \Big]) = \mathrm{Tr}(A^{-1} A) = \mathrm{Tr}(\mathbb{1}) \tag{A.4}$$

Whereas for the $\langle \phi - b| \, B^{-1} \, |\phi - b\rangle$ term, we can perform a similar trick by decomposing it into $\langle \phi - a - (b - a)| \, B^{-1} \, |\phi - a - (b - a)\rangle$ and calling $(b - a) = y$, so we can rewrite it as $\langle \phi' - y| \, B^{-1} \, |\phi' - y\rangle$ with ϕ' as before. Expanding this quadratic out gives us:

$$\mathbb{E}\Big[\langle \phi' - y| \, B^{-1} \, |\phi' - y\rangle \Big]_{\mathcal{G}(\phi - a, A)} \tag{A.5}$$
$$= \mathbb{E}\Big[\langle \phi'| \, B^{-1} \, |\phi'\rangle \Big] + \mathbb{E}\Big[- \langle \phi'| \, B^{-1} \, |y\rangle \Big] + \mathbb{E}\Big[- \langle y| \, B^{-1} \, |\phi'\rangle \Big] + \mathbb{E}\Big[\langle y| \, B^{-1} \, |y\rangle \Big]$$
$$= \mathrm{Tr}\left(B^{-1} \underbrace{\mathbb{E}\Big[\, |\phi'\rangle \, \langle \phi'| \, \Big]}_{=A} \right) - \underbrace{\mathbb{E}\Big[\, \langle \phi'| \, \Big]}_{=0} B^{-1} \, |y\rangle - \langle y| \, B^{-1} \underbrace{\mathbb{E}\Big[\, |\phi'\rangle \, \Big]}_{=0} + \underbrace{\mathbb{E}\Big[\, \langle y| \, B^{-1} \, |y\rangle \, \Big]}_{= \langle y | B^{-1} | y \rangle}$$

Combining all the parts together yields

$$\frac{1}{2} \Bigg(\mathrm{Tr}[\ln(BA^{-1}) - \mathbb{1} + B^{-1} A] + \langle b - a| \, B^{-1} \, |b - a\rangle \Bigg) \tag{A.6}$$

as desired. □

Selbständigkeitserklärung

Hiermit erkläre ich, Martin Dupont, die beiliegende Arbeit “On the Practical Applications of Information Field Dynamics” selbständig und ohne unerlaubte fremde Hilfe angefertigt zu haben. Keine anderen als die von mir angegebenen Schriften und Hilfsmittel wurden benutzt. Die den benutzten Werken wörtlich und inhaltlich entnommenen Stellen sind kenntlich gemacht.

Datum, Unterschrift