

A composite scheme for solving a differential equation driven by a system of time-varying vector fields.



Laura Campbell
Somerville College
University of Oxford

A thesis submitted for the degree of
Master of Science by Research

Trinity 2010

This thesis is dedicated to
my parents and all my friends
without whom I could not have done this.

Acknowledgements

I would like to thank my supervisors, Terry Lyons, Irene Moroz and John Norbury for all the help they have offered me over the course of my thesis. I would also like to thank the Met Office and EPSRC for funding me at the beginning of my thesis, and OCDEM and Oxford Brookes University for offering me part-time work towards the end of it. Ian Roulstone, Reason Machete and Chris Farmer offered me many valuable ideas and helped me refine my project. Some of the code I used for this project was adapted from that written by Rich Neidinger, Atakan Varol, Christophe Ladroue, Daniel Levin, and the Computational Rough Paths group (Terry Lyons, Stephen Buckley, Djalil Chafai, Greg Gyurk and Arend Janssen). I would also like to thank Mason Porter and Arend Janssen for many invaluable discussions and advice on developing my writing style, Gareth Jenkins-Jones for teaching me how to write and debug C++ code, Xan Ophis for proofreading, James Wyatt and Luke Pitcher for moral support, and my parents for their endless encouragement.

Abstract

Many problems in physics and chemistry are based around the interactions of large numbers of particles. Motivated by this, we consider numerical methods aimed towards the accurate modelling of a ‘particle’ under the influence of a large cloud of other particles. The trajectory of this ‘particle’ can be modelled by the solution to an ordinary differential equation driven by a high-dimensional system where the vector field defining the ordinary differential equation is given by the sum of a large number of time-varying vector fields. In this thesis, we shall describe and develop an algorithm for solving such equations.

To make progress, we need to find a way to summarise the large amount of information provided by the vector fields. We do this by performing a three-step simplification; first we localise the problem and expand the vector fields in space using Taylor series, thus projecting the trajectory onto the finite-dimensional subspace of polynomial vector fields of the infinite-dimensional space of vector fields that initially control the evolution of the ‘particle’ over time. We then separate the system into a series of space-dependent vector fields and a time-varying driving path. Finally, we approximate the system’s behaviour in time by finding the log-signatures of the driving path and contracting them with the appropriate vector fields, thus producing a flow map.

We consider three existing models and also construct and investigate two new test problems, the Interacting Harmonic Oscillators and Interacting Simple Pendula systems, which are two-dimensional dynamical systems driven by a large number of vector fields, in order to expose the weaknesses in our initial implementations. We then discuss how our implementation could be made faster and more accurate, and how it might be extended for use on higher-dimensional systems.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Statement of the problem	1
1.3	Results used in the algorithm	2
1.3.1	Taylor series	3
1.3.2	Interpolating with a partition of unity	4
1.3.3	Rough paths	6
1.3.4	Controlled systems, driving paths and signatures	7
1.3.5	The fourth-order explicit Runge-Kutta-Nyström method	11
1.4	Existing algorithms	16
1.4.1	Approximating linear differential equations with Magnus series	16
1.4.2	Fast Multipole Methods	18
1.5	A high-level description of the algorithm	19
1.6	Thesis outline	20
2	The test problems	21
2.1	Introduction	21
2.2	Simple harmonic motion	21
2.3	The simple pendulum	23
2.4	The Airy equations	23
2.5	The Interacting Pendula systems	25
2.5.1	Initial conditions	27
2.6	Investigation of the Interacting Pendula systems	29
2.6.1	A qualitative summary of the behaviour of the response particle	29
2.6.2	Mean, standard deviation, skewness and kurtosis	35
2.6.3	Periodicity	36
2.6.4	Energy	41
2.7	Summary	43

3	Implementing the algorithm	49
3.1	Description of the algorithm	49
3.2	Results used in applying the algorithm	52
3.2.1	Calculating coefficients for multivariate truncated Taylor series	52
3.2.2	Constructing the partition of unity	53
3.3	Lie brackets	54
3.3.1	The Hall basis and the log-signatures	57
3.3.2	Error control	58
3.3.2.1	Local error control	58
3.3.2.2	Placing triangulation points using a quadtree	60
3.4	Examples	60
3.4.1	The harmonic oscillator	60
3.4.2	The simple pendulum	62
3.4.3	The Airy equations	63
3.4.4	The Interacting Pendula systems	64
3.4.4.1	The Interacting Harmonic Oscillators	64
3.4.4.2	The Interacting Simple Pendula	65
3.5	Summary	65
4	Preliminary results	66
4.1	Properties of the algorithm and the test problems	66
4.1.1	Results	68
4.1.2	Simple harmonic motion	68
4.1.3	The Airy equations	70
4.1.4	The simple pendulum	73
4.1.5	The Interacting Simple Pendula system	75
4.1.6	The Interacting Harmonic Oscillators system	76
4.2	Summary	77
5	Conclusions	78
5.1	Introduction	78
5.2	The test problems	78
5.2.1	Physically plausible test problems	78
5.2.2	Higher dimensional test problems	79
5.2.3	Stochastic test problems	80
5.3	Improvements to the implementation of the algorithm	81

List of Figures

1.1	Comparison of five time series, with signatures given in the equations (1.19).	10
1.2	Clockwise from top left: phase plane, numerical solution plotted against time, energy plotted against time and exact solution plotted against time for the RKN method on the nonlinear simple pendulum given by $\dot{y} = p$, $\dot{p} = -\sin y$	12
1.3	Superimposed plots of the energy for RKN on the nonlinear simple pendulum with timestep 0.1×2^{-k} , with k going from 0 to 5 from top to bottom. Observe the quartic convergence.	13
2.1	Evolution of y with time, p with time, and the phase plane for simple harmonic motion, from top to bottom, with initial conditions $y(0) = 0$, $p(0) = 1$	22
2.2	Evolution of y with time, p with time, and the phase plane for the nonlinear simple pendulum, from top to bottom.	24
2.3	Behaviour of $A_i(t)$ and $B_i(t)$, showing the shift from oscillatory to exponential behaviour at $t = 0$	25
2.4	Plots of $x_i(0)$ and $p_i(0)$; the initial conditions for 100 control particles in two clouds, beginning at $x = 150$ and $x = -150$	27
2.5	Plots of y and p for the Interacting Harmonic Oscillators, where $G = 0.1$, with a single cloud of control particles centred around $x = 150$	28
2.6	Plots of y and p against time, and the phase plane for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$. We shall refer to this as Data Set 1.	30
2.7	Plots of y and p against time, and the phase plane for the the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$. We shall refer to this as Data Set 2.	31

2.8	Plots of y and p against time, and the phase plane for the Interacting Simple Pendula, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$. We shall refer to this as Data Set 3. . .	32
2.9	Plots of y and p against time, and the phase plane for the Interacting Simple Pendula, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$. We shall refer to this as Data Set 4. . .	33
2.10	Plots of y and p (above) compared to $\min x_i - y $ (below) for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$. Observe the difference in the maximum amplitude of the close transits: the amplitude of the oscillation increases as the minimal distance apart decreases. .	34
2.11	Plots of y and p against time for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$. We observe three different amplitude regimes within this diagram.	37
2.12	Plots of y and p against time for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$. We observe two different amplitude regimes within this diagram.	38
2.13	Plots of y and p against time for the Interacting Simple Pendula system, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$	39
2.14	Plots of y and p against time for the Interacting Simple Pendula system, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$. We observe two different amplitude regimes within this diagram.	40
2.15	Plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$	42
2.16	Detail of the plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$	43
2.17	Plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$	44

2.18	Detail of the plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$	45
2.19	Plot of energy against time for the Interacting Simple Pendula, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$	45
2.20	Detail of the plot of energy against time for the Interacting Simple Pendula, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$. We fit a linear polynomial $E = 0.001456t - 0.4937$ to the energy.	46
2.21	Plot of energy against time for the Interacting Simple Pendula, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$	47
2.22	Detail of the plot of energy against time for the Interacting Simple Pendula, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$. We fit a linear polynomial $E = 0.002045t - 0.4214$ to the energy.	48
3.1	Functions P_j for a 1-dimensional partition of unity, with triangulation points at $x = -1 + 0.5d$ for $d = 1, \dots, 6$, with radii $r = 1/\sqrt{2}$. We superimpose the bump functions (blue) on the partition of unity (red). Note that the partition of unity functions approach a constant value of 1 away from the triangulation points.	54
3.2	The two-dimensional Gaussian bump functions w_j with centres $(-0.5, 0.5)$, $(-0.5, -0.5)$, $(0.5, -0.5)$ and $(0.5, 0.5)$, each with an associated radius of $1/\sqrt{2}$	55
3.3	The partition of unity functions P_j generated from the bump functions shown in Figure 3.2.	56
3.4	Consider one of the functions P_j from the 1-dimensional partition of unity shown in Figure 3.1 (shown in red). We calculate the Taylor series for $f = -\sin y$ up to 5th order about the triangulation point, and then calculate the error (shown in blue). We observe that the error in the Taylor series becomes significantly non-zero before the partition of unity function fades away completely. This illustrates the importance of careful choice of radii for the partition of unity.	59

4.1	Plots of y and p against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$. We compare the results for the algorithm, for the exact solution, and for fourth-order RKN, using a time step of $\Delta t = 0.01$. We observe that there is very little difference.	68
4.2	Plots of d_1 and d_2 against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$, where d_1 is the distance between the exact solution and the solution given by the algorithm, and d_2 is the distance between the exact solution and the solution given by fourth-order RKN.	69
4.3	Plots of e_1 and e_2 against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$, where e_1 is the coordinate-wise distance between the exact solution and the solution given by the algorithm, and e_2 is the coordinate-wise distance between the exact solution and the solution given by fourth-order RKN.	70
4.4	Plots of y and p against time for the Airy equations, with initial conditions $y_0 = 0$, $p_0 = 1$. We compare the results for the algorithm and for fourth-order RKN, using a time step of $\Delta t = 0.01$. We observe that there is very little difference.	71
4.5	Plots of d and e against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$	72
4.6	Plots of y and p against time for the simple pendulum, with initial conditions $y_0 = 0$, $p_0 = 1$. We compare the results for the algorithm to those for the exact solution. We observe a spurious growth in amplitude for the algorithmic results.	73
4.7	Plots of d and e against time for the simple pendulum, with initial conditions $y_0 = 0$, $p_0 = 1$	74
4.8	Plots of p against time for the Interacting Harmonic Oscillators system, with initial conditions $y_0 = 0$, $p_0 = 1$. We observe that the error in the plot manifests dramatically even over short time scales.	75
4.9	Plots of p against time for the Interacting Harmonic Oscillators system, with initial conditions $y_0 = 0$, $p_0 = 1$. We observe that the error in the plot manifests dramatically even over short time scales.	76

Chapter 1

Introduction

1.1 Motivation

A large number of chemical and physical problems are based around the interactions of large numbers of particles, such as the N -body problem and related questions in celestial mechanics (see [13,25,27]), problems in weather and climate modelling involving the incorporation of sub-grid-scale perturbations into models for the large-scale balanced Navier-Stokes equations (see [21]), and the magnetic interactions within the disordered magnetic structure of a spin glass (see [24]).

We are interested in the specific sub-class of such problems in which we consider the effect of a large number of particles on an individual particle; for example following a single particle in a heat bath or a weather system, or a single star in a galaxy. We can model the trajectory of the individual particle in such a case as the solution to a differential equation driven by a high-dimensional system given by a sum of several vector fields. In this thesis, we discuss a numerical method for solving problems of this type.

1.2 Statement of the problem

We are interested in time inhomogeneous differential equations of the form

$$d\mathbf{y}_t = \sum_{i=1}^N \mathbf{F}^i(\mathbf{y}_t, t) dt, \quad (1.1)$$

where $\mathbf{F}^i$, for $i = 1, \dots, N$, are a large ($N \approx 100$) number of time-varying smooth vector fields and $\mathbf{y}_t$ is the response of the system over time. We intend to construct an algorithm which is capable in principle of describing the development of the trajectory $\mathbf{y}_t$ from some initial condition $\mathbf{y}_0$ at time t_0 , over a series of time steps of size Δt , to

a final value $\mathbf{y}_T$ at time T . We shall discuss the applications of this algorithm to test problems for which $\mathbf{F}^i$ are vector fields over $\mathbb{R}^n$, but in principle the algorithm can be extended to vector fields over any manifold.

In general, the sum of a large number of time-varying terms will vary with an increasingly high range of frequencies as more such terms are added. Because each of the terms $\mathbf{F}^i$ is time-varying, this means that $\sum_{i=1}^N \mathbf{F}^i$ will display very high frequency variation for large N . This means that $\mathbf{y}_t$ will have considerable oscillation on each time interval $[s, t]$, and we require that our algorithm be capable of taking this oscillation into account when describing the system's evolution. We shall also require that the algorithm be capable of dealing with a large number of vector fields, say $N = 10^2$.

In order to construct such an algorithm, we shall need a way of summarising the large amounts of information provided by this problem. Consider, for example, modelling the impact of multiple time-varying external forces on a non-rigid body; we would have to deal with very large amounts of data to solve this problem, and so summarising the data without losing too much accuracy would be very important here. Specifically, we need to summarise how the vector fields vary in both space and time, so we construct our algorithm using a two-step simplification; we expand the vector fields as power series in space using Taylor's theorem, and in time using the Lie brackets of the vector fields and the corresponding iterated integrals.

We shall then construct and describe two suitably challenging specific multi-dimensional dynamical systems driven by a large number of vector fields in Chapter Two, which we can consider as possible test problems for the implementation of the algorithm. In Chapter Three, we shall discuss the construction of the algorithm in detail, and in Chapter Four we shall consider the preliminary results from testing the algorithm on three well-known systems: the harmonic oscillator, the Airy equations, and the simple pendulum.

1.3 Results used in the algorithm

In order to construct this algorithm, we shall make use of several existing results, and combine them to perform a suitable two-step reduction in space and time. In the following section, we shall discuss these results, specifically the Taylor series expansion, the partition of unity, some results from rough path theory including the concept of a driving path and its signatures, and the Runge-Kutta-Nyström solver.

1.3.1 Taylor series

At the start of each individual time step, we need to consider the infinite dimensional space of vector fields that potentially control the trajectory $\mathbf{y}_t$, and to find some way to approximate it by a lower-dimensional space of vector fields. Provided that each vector field $\mathbf{F}^i$ is smooth, we can expand the vector fields $\mathbf{F}^i$ as Taylor series [32] at some time t_i around some point $\mathbf{a}$, in the neighbourhood of the corresponding point $\mathbf{y}_i = \mathbf{y}_{t_i}$. The resulting series for $\mathbf{y} \in \mathbb{R}^2$ with coordinates x_1, x_2 will be of the form

$$\mathbf{F}^i(\mathbf{y})|_{\mathbf{a}} = \sum_{k_1+k_2=j, j=0}^{\infty} \frac{\partial^{k_1}}{\partial x_1^{k_1}} \frac{\partial^{k_2}}{\partial x_2^{k_2}} \mathbf{F}^i(a_1, a_2) \left| \left(\frac{(x_1 - a_1)^{k_1} (x_2 - a_2)^{k_2}}{k_1! k_2!} \right) \right|. \quad (1.2)$$

We truncate the Taylor series after n terms, where we shall consider how best to choose a value of n later on. We note that to produce a truncated Taylor series with n terms, we only need all the $\mathbf{F}^i$ to be n times differentiable, as opposed to infinitely differentiable. In this case the upper bound on the remainder term $\mathbf{R}_{n+1}$ within a ball B centred on $\mathbf{a}$, where $\mathbf{F}^i$ has $n+1$ partial derivatives on the closure $\bar{B}$, is

$$|\mathbf{R}_{n+1}| \leq \sup_{\mathbf{x}^* \in \bar{B}} \sum_{k_1+k_2=n+1} \left| \frac{1}{k_1! k_2!} \frac{\partial^{k_1}}{\partial x_1^{k_1}} \frac{\partial^{k_2}}{\partial x_2^{k_2}} (x_1^* - a_1)^{k_1} (x_2^* - a_2)^{k_2} \mathbf{F}^i(\mathbf{x}^*, t) \right|. \quad (1.3)$$

We require that the magnitude of the remainder term $|\mathbf{R}_{n+1}| < \epsilon$ within a ball B centred on $\mathbf{a}$ of some radius a_n , where ϵ is some desired upper bound on the local error at each time step. We thus obtain the truncated Taylor series $\mathbf{F}_n^i$.

Since a Taylor series can be thought of as a linear map from a high-dimensional space of vector fields to a lower-dimensional space of polynomial vector fields, we can calculate the sum of all the Taylor series $G = \sum_{i=1}^N \mathbf{F}_n^i(\mathbf{y}_t, t)$, which will also be a vector field.

If we assume that $\mathbf{y}_t$ is a trajectory in $\mathbb{R}^d$, we can express the sum of all the Taylor series expanded about a point $\mathbf{y} = \mathbf{b}$ as

$$\begin{aligned} G(\mathbf{y}_t, t) &= \sum_{i=1}^N \mathbf{F}_n^i(\mathbf{y}_t, t)_{\mathbf{y}=\mathbf{b}}, \\ &= \sum_{i=1}^N \sum_{|\mathbf{k}| \leq n} \frac{1}{k_1! k_2!} \left(\left(\frac{\partial}{\partial x_1} \right)^{k_1} \left(\frac{\partial}{\partial x_2} \right)^{k_2} \right) F^i(\mathbf{b}, t) (\mathbf{y} - \mathbf{b})^{\mathbf{k}}, \end{aligned} \quad (1.4)$$

where $\mathbf{k} = (k_1, k_2)$ where $k_i \in \mathbb{Z} \setminus 0$ for all i , $|\mathbf{k}| = k_1 + k_2$, and $\mathbf{x}^{\mathbf{k}} = x_1^{k_1} x_2^{k_2}$.

1.3.2 Interpolating with a partition of unity

We define a *partition of unity* on a smooth manifold M as follows; first we define an *open cover* $\{U_j\}_{j=0}^J$ on M , which is a collection of J open subspaces of M such that their union is the whole of M . We refer to $\{U_j\}$ as *patches* (see [1]). The partition of unity is a set of continuous functions $\{P_j\}_{j=0}^J$ (see [1, 8, 22]), such that

- $P_j : M \rightarrow [0, 1] \forall j$,
- each P_j has compact support, i.e. it is zero outside a compact set,
- each patch U_j contains the support for the corresponding function P_j , i.e. the set closure of the non-zero arguments of the functions P_j lie in U_j ,
- there is some neighbourhood of points $x \in M$ where all but a finite number of the functions are identically zero,
- $\sum_{j \in J} P_j(x) = 1, \forall x \in M$.

Partitions of unity are often used to piece together locally defined objects to cover an entire manifold. In this case, we intend to do this with vector fields, as approximated by truncated Taylor series. Since a given truncated Taylor series of n terms centred around a point $\mathbf{b}$ is only a good approximation to the vector field on a ball of radius r given by $B(\mathbf{b}, r)$, being able to interpolate smoothly between the different Taylor series approximations in between their regions of accuracy is important.

We shall cover M with coordinate charts U_j , each one defined by a point b_j and a radius r_j , where $U_j = B(b_j, r_j)$, where we refer to b_j as a *triangulation point* and where r_j is chosen such that only finitely many overlap at any given point. We can use these charts as our patches to construct a partition of unity, by defining function $P_j(y)$ on each patch U_j such that it satisfies the conditions for the partition of unity stated above. On each patch U_j we define a local vector field $v_j = G(y)$, $y \in U_j$. Our global vector field is therefore $V = \sum_j v_j P_j$, which converges since at any y , only finitely many $P_j(y) \neq 0$. Since this sum V is actually a convolution, it has the same smoothness properties and the same approximation properties as the P_j (see [1, 8]). The composition of the Taylor polynomials with the functions of the partition of unity ($v_j P_j$ for $j \in J$) yields a basis for the global vector field. We also note that two vector fields v_i and v_j on disjoint patches will have a Lie bracket $[v_i, v_j] = 0$, which goes some way to eliminating the problem of controlling the magnitude of the error term in the Lie brackets.

For a function $F(y)$, $F(y) = \sum_j P_j(y)F(y)$ globally, since $\sum_{j \in J} P_j(y) = 1$, $\forall y \in M$. We define $\tilde{F}(y) = \sum_j P_j(y)F_n(y)|_{y=b_j}$, where $F_n(y)|_{y=b_j}$ is the Taylor series expansion of $F(y)$ to n terms centred about a point y_j . We shall now show that if $|F(y) - F_n(y)|_{y=b_j}| < \epsilon \forall y \in U_j$, then $|F(y) - \tilde{F}(y)| < \epsilon$ over the whole manifold M . Consider

$$\begin{aligned} |F(y) - \tilde{F}(y)| &= \left| \sum_j P_j(y)F(y) - \sum_j P_j(y)F_n(y)|_{y=b_j} \right| \\ &= \left| \sum_j P_j(y)(F(y) - F_n(y)|_{y=b_j}) \right|. \end{aligned} \quad (1.5)$$

Now for any given y and j , if $y \notin U_j$ then $P_j(y) = 0$, and therefore $P_j(y)|(F(y) - F_n(y)|_{y=b_j})| < P_j(y)\epsilon$. If $y \in U_j$, then $0 \leq P_j(y) \leq 1$, and therefore $P_j(y)|(F(y) - F_n(y)|_{y=b_j})| < P_j(y)\epsilon$. Hence

$$\begin{aligned} |F(y) - \tilde{F}(y)| &= \left| \sum_j P_j(y)(F(y) - F_n(y)|_{y=b_j}) \right|, \\ &< \left| \sum_j P_j(y)\epsilon \right|, \\ &< \epsilon \left| \sum_j P_j(y) \right|, \\ &\leq \epsilon. \end{aligned} \quad (1.6)$$

Hence, if each $F_n(y)|_{y=b_j}$ is a good local approximation to $F(y)$ in the patch U_j , then the product of these local approximations with the partition of unity provides a good global approximation over the entire manifold.

A good local approximation in the patches U_j can be achieved either by using a large number of small patches (which is referred to as the ‘ h version’ of the method) or by using higher-accuracy versions of v_j , which in this case would be more terms in the Taylor expansion (referred to as the ‘ p version’). These terms are taken from analogy with classical finite element methods; in that context, we approximate the vector field by a polynomial on each polygonal element, and we can improve the accuracy by decreasing the mesh size (h version), by increasing the number of terms of the polynomial (p version) or by some flexible trade-off between the two ($h-p$ versions). In classical finite element methods, the jumps across inter-element boundaries are resolved by polynomial corrections. The advantage of the partition of unity method is that it gives us the very precise knowledge about the local behaviour of the vector

field that allows us to make the trade-offs involved in the $h-p$ method in an optimal way (see [1]).

Let $\nu_j(y_t, t) = G_{\tilde{\mathbf{y}}}(y_t, t)$ for $\tilde{\mathbf{y}} \in U_j$. The global vector field will be given by

$$\begin{aligned}
V(y_t, t) &= \sum_{j=1}^J \nu_j(y_t, t) P_j(y), \\
&= \sum_{j=1}^J \sum_{i=1}^N F_n(y)|_{b_j} P_j(y), \\
&= \sum_{j=1}^J \sum_{i=1}^N \sum_{|\mathbf{k}| \leq 2} \frac{1}{k_1! k_2!} \left(\left(\frac{\partial}{\partial x_1} \right)^{k_1} \left(\frac{\partial}{\partial x_2} \right)^{k_2} \right) F^i(\mathbf{b}_j) (\mathbf{y} - b_j)^{\mathbf{k}} P_j(y), \\
&= \sum_{|\mathbf{k}| \leq n} C_{\mathbf{k}} y^{\mathbf{k}}, \tag{1.7}
\end{aligned}$$

where $C_{\mathbf{k}}(y) = (\sum_{j=1}^J c_{\mathbf{k}}(\mathbf{b}_j) P_j(y))$. Hence, we now have a multivariate truncated power series in $\mathbf{y}$.

1.3.3 Rough paths

Let $(V, \|\cdot\|_V)$ be a Banach space (i.e. a vector space V such that every Cauchy sequence in V with respect to that norm has a limit in V) where $V^{\otimes k}$ is its k -tensor product (i.e. the tensor product of the vector space V with itself k times, or the space of all k^{th} order tensors over V). Let $T^{(n)}(V) = \bigoplus_{k=0}^n V^{\otimes k}$ be the truncated tensor algebra, which is a *Lie algebra* with the *Lie bracket* $[a, b] = a \otimes b - b \otimes a$.

We define a Lie algebra by a vector field V over a field F along with a binary operation $[\cdot, \cdot]$ which we refer to as a Lie bracket, which is bilinear (i.e. $[ax + by, z] = a[x, z] + b[y, z]$ and $[z, ax + by] = a[z, x] + b[z, y]$), skew-symmetric ($[x, y] = -[y, x]$ and also $[x, x] = 0$ if F has characteristic 2) and satisfies the Jacobi identity $[x, [y, z]] + [y, [z, x]] + [z, [x, y]] = 0 \ \forall x, y, z \in V$.

We then define a *multiplicative functional* (see [29]) as a map from pairs of reals $s \leq t$ to $\mathbf{x}_{s,t} = (1, x_{s,t}^1, \dots, x_{s,t}^n) \in T^{(n)}(V)$, where $x_{s,t}^k$ is a k^{th} order tensor, such that

$$\mathbf{x}_{r,t} = \mathbf{x}_{r,s} \otimes \mathbf{x}_{s,t}, \tag{1.8}$$

for all $r \leq s \leq t$.

A smooth path x in V has a natural extension $\mathbf{x}_{s,t} = (x_{s,t}^0, x_{s,t}^1, \dots, x_{s,t}^n)$ to $T^{(n)}(V)$ where $x_{s,t}^k$ is the k^{th} iterated integral of the path: $x_{s,t}^0 = 1$, and $x_{s,t}^{k+1} = \int_t^s x_{s,u}^k \otimes dx_u$, which is a vector field.

We define the p -variation of a continuous function $x : [0, 1] \rightarrow \mathbb{R}^d$, for some $p \geq 1$,

$$\sup_{k \geq 1, t_0 \leq t_1 \leq \dots \leq t_k \leq 1} \sum_{i=0}^{k-1} |x_{t_i} - x_{t_{i+1}}|^p < \infty. \quad (1.9)$$

Consider the free Lie algebra $T^{(\lfloor p \rfloor)}(V)$, where $\lfloor p \rfloor$ is the integer part of p , and the operator $d_p(\mathbf{x}, \mathbf{y})$, which we refer to as the p -variation distance, where

$$d_p(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{\lfloor p \rfloor} \sup_D \sum_{t_1} \|x_{t_i, t_{i+1}} - y_{t_i, t_{i+1}}\|_{V^{\otimes k}}^{i/p} + \sup_t \|x_t - y_t\|_V, \quad (1.10)$$

where D is the set of all *partitions* $t_0 \leq t_1 \leq \dots \leq t_k \leq 1$ of $[0, 1]$. This operator defines a distance between two points x and y in $T^{(\lfloor p \rfloor)}(V)$. We define the closure of $T^{(\lfloor p \rfloor)}(V)$ under $d_p(x, y)$ as the smallest closed set containing $T^{(\lfloor p \rfloor)}(V)$ and also the limit points under $d_p(x, y)$, where the limit points x are the points for some $\epsilon > 0$ such that there exists a point y in $T^{(\lfloor p \rfloor)}(V)$ different from x with $d_p(x, y) < \epsilon$. The closure of the set of extensions of smooth paths in $T^{(\lfloor p \rfloor)}(V)$ under the p -variation distance is the set of *geometric rough paths* $G\Omega_P(V)$. We say that $\mathbf{x}$, which is a path in the Lie algebra $T^{(\lfloor p \rfloor)}(V)$ whose elements are vector spaces, is a rough path *lying above* the path x in V whose elements are vectors (as per [29]).

To summarise, a rough path is a path in a space of vector spaces. It may be highly oscillatory, and continuous but not differentiable over a large scale. It is specified by a series of k -tensors (i.e. points and associated iterated integrals). The rough path can be thought of as a path in a Lie algebra (which will come in useful when we consider how to approximate such a path numerically).

1.3.4 Controlled systems, driving paths and signatures

A *controlled system* is a dynamical system subject to an external control process; consider, for example, the motion of a bridge during an earthquake, or the vibrations of a human ear drum when subjected to sound waves (see [16, 17]). Generally, we are chiefly interested in a low-dimensional process $X(t)$ for the effect it has when used as an input for a controlled higher-dimensional system, whose response is not linear in the input data stream, and where the input data $X(t)$ is rapidly fluctuating and rich in information and so is difficult to summarise easily. We say that the system is *driven* by the process $X(t)$, and describe $X(t)$ as a *driving path*. In general, we relate the state $Y(t)$ of the differential system to the state $X(t)$ of the control process via a

system of differential equations of the form

$$dY(t) = f(Y(t))dt + \sum_{i=1}^n g_i(Y(t))dX_i(t), \quad (1.11)$$

for some functions $f(Y)$ and $g_i(Y)$. Control functions are generally not smooth on normal time scales, even if they are continuous (see [17]).

We have constructed a general global time-dependent vector field

$$V(\mathbf{y}_t, t) = \sum_{|\mathbf{k}| \leq n} C_{\mathbf{k}}(\mathbf{y}_t, t) y^{\mathbf{k}}, \quad (1.12)$$

which is driving the differential systems (1.1). We now need to consider how best to summarise the manner in which this vector field changes with time, in order to take into account the fact that the oscillations over each time interval will be of very high frequency for large N .

Consider an integral curve y on a vector field $\mathbf{G}$. We can map this onto a path Y , using the identity map in this case. We can then define a functional, which we can use to lift the path Y on the vector field $\mathbf{G}$ to a unique path $\mathcal{Y}$ based at the initial value y_0 in a manifold M . This functional, which we refer to as the Itô map (see [14]), takes the path Y_t on a vector field to a path $\mathcal{Y}_t$ on a space of vector fields.

For a geometric rough path $\mathcal{X}_t$ obtained from a path X_t on a vector field $\mathbf{V}$ over an interval $[s, t]$, we can use the following iterative definition to obtain iterated integrals:

$$\mathcal{X}_{s,t}^{k+1} = \int_s^t \mathcal{X}_{s,u}^k \otimes d\mathcal{X}_u \quad (1.13)$$

for $k \geq 1$, where the zeroth-order iterated integral is

$$\mathcal{X}_{s,t}^0 = 1. \quad (1.14)$$

We refer to the sum of these iterated integrals

$$S_{s,t}(\mathcal{X}) = 1 \oplus \mathcal{X}_{s,t}^1 \oplus \dots \oplus \mathcal{X}_{s,t}^m + \dots \quad (1.15)$$

as the *signature* of the path $\mathcal{X}$ over the time interval $[s, t]$, and to

$$S_{s,t}^m(\mathcal{X}) = 1 \oplus \mathcal{X}_{s,t}^1 \oplus \dots \oplus \mathcal{X}_{s,t}^m \quad (1.16)$$

as the *truncated signature at level m* . The signature $S_{s,t}(\mathcal{X}_t)$ is an element of the closure $T((\mathbb{R}^d))$ of the tensor algebra $T(\mathbb{R}^d) \bigoplus_{k=0}^{\infty} (\mathbb{R}^d)^{\otimes k}$, where d is the dimension of the vector space in which $\mathcal{X}_t$ exists, and so $S_{s,t}^m(\mathcal{X})$ belongs to the truncated tensor algebra $T^{[m]}(\mathbb{R}^d)$ (see [17]).

We can let $\mathbf{G}(\mathbf{y}_t, t) = \sum_{i=1}^N \mathbf{F}^i(\mathbf{y}_t, t)$, as the sum of several vector fields is a vector field itself. The truncated tensor signature $S_{s,t}^m(\mathcal{Y}_t)$ is a path in the finite-dimensional subspace spanned by polynomial vector fields of degree $\leq m$ in a way that is similar to the use of the Taylor series approximations to project the trajectory $\mathbf{y}_t$ in space down onto a finite-dimensional subspace spanned by polynomial vector fields of degree $\leq n$.

We define the *log-signature* and *truncated log-signature* of $\mathcal{Y}_t$ as

$$\log S_{s,t}(\mathcal{Y}_t) = \log(S_{s,t}(\mathcal{Y}_t)), \quad (1.17)$$

$$\log S_{s,t}^m(\mathcal{Y}_t) = \log_m(S_{s,t}^m(\mathcal{Y}_t)), \quad (1.18)$$

where $\log$ denotes the usual logarithm on the tensor algebra $T(\mathbb{R}^m)$, and $\log_m$ denotes the corresponding truncated logarithm (i.e. m terms of the power series for the logarithm). Since there are many dependencies between the iterated integrals $\mathcal{Y}_{s,t}^m$, there is a lot of redundancy in the whole series $S_{s,t}(\mathcal{Y})$, which can be avoided by considering the log-signatures instead (see [17]). The first order term of the signature $\mathcal{S}_{s,t}^1(\mathcal{Y}_t)$ is the total increment of the path (given by the sum of the individual increments), and the second order term $\mathcal{S}_{s,t}^2(\mathcal{Y}_t)$ is the Lévy area i.e. the sum of the areas in between the path and the increments; higher order terms are less easy to understand geometrically.

In Figure 1.1 we see a comparison of five time series, with the following signatures:

$$\begin{aligned} S(y_1)_{0,T} &= 1 + 4X^1 + 5X^{22}, \\ S(y_2)_{0,T} &= 5 + 4X^1 + 5X^{22}, \\ S(y_3)_{0,T} &= 5 + 4X^1 + 2X^{21} + 5X^{22} + 1X^{111}, \\ S(y_4)_{0,T} &= 1 + 4X^1 + 2X^{21} + 5X^{22} + 1X^{111} + 10X^{112} + 1X^{122}, \\ S(y_5)_{0,T} &= 1 + 4X^1 + 2X^{21} + 5X^{22} + 1X^{111} + 0.1X^{112} + 0.5X^{122}. \end{aligned} \quad (1.19)$$

The time series were generated using the Computational Rough Paths Matlab code written by Christophe Ladrone (see [7]) to create objects in terms of the specified combination in terms of coefficients and one-dimensional iterated integrals, and then to integrate them using a Brownian path as the stochastic driver. As we can see, loosely speaking, the lower-order terms determine the general direction and large-scale behaviour of the time series, and the higher-order terms affect its level of roughness and small-scale behaviour, which we might call ‘flavour’.

The solution to an ODE of the form of equation (1.1) can then be found using the signatures of the path $\mathcal{Y}_t$ found from an integral curve of the vector field $\mathbf{G}$, with

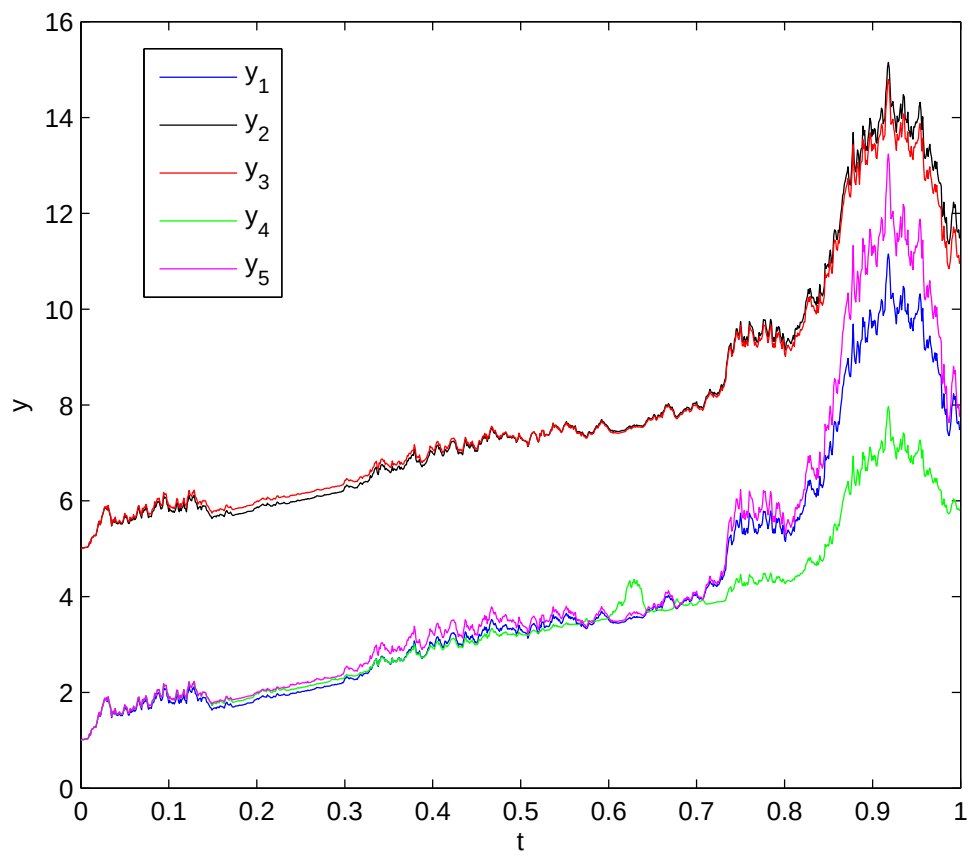


Figure 1.1: Comparison of five time series, with signatures given in the equations (1.19).

iterated integrals $\mathcal{Y}_{s,t}^{k,(i_1,\dots,i_k)}$. Consider the power series

$$\phi([s,t], \mathcal{Y}_t) = 1 + \sum_{k \geq 1, i_1, \dots, i_k \in \{1, \dots, d\}^k} \dots \sum X^{i_1} \dots X^{i_k} \mathcal{Y}_{s,t}^{k,(i_1,\dots,i_k)}, \quad (1.20)$$

for some indeterminates $X^1, \dots, X^d$. The Chen series $\phi([s,t], \mathcal{G})$ completely characterises the path $\mathcal{Y}_t$ (as shown in [3]), and for all $u \in [s,t]$ we have

$$\phi([s,u], \mathcal{Y}_t) \phi([u,t], x) = \phi([s,t], \mathcal{Y}_t). \quad (1.21)$$

If we let ‘exp’ denote the non-commutative exponential, then there exists a power series $\psi([s,t], \mathcal{Y}_t)$ such that $\phi([s,t], \mathcal{Y}_t) = \exp \psi([s,t], x)$, and

$$\psi([s,t], \mathcal{Y}_t) = 1 + \sum_{k \geq 1, i_1, \dots, i_k \in \{1, \dots, d\}^k} \dots \sum H_{i_1, \dots, i_d}(X^{i_1} \dots X^{i_k}) \mathcal{Y}_{s,t}^{k,(i_1,\dots,i_k)}, \quad (1.22)$$

where $H_{i_1, \dots, i_d}(X^{i_1} \dots X^{i_k})$ belongs to the Lie algebra generated by the indeterminates $X^1, \dots, X^d$.

The solution to the ODE (1.1) can then be given by

$$\mathbf{y}_t = \exp(\tilde{\psi}([s,t], \mathcal{Y}_t)) \mathbf{y}_s, \quad (1.23)$$

where we substitute the vector fields F^i and their Lie brackets for the indeterminates X^i into $\psi([s,t], \mathcal{Y}_t)$ to obtain $\tilde{\psi}([s,t], \mathcal{Y}_t)$, and in this case we define the exponential ‘exp’ as the map defining a left-invariant vector field from a vector in the Lie algebra (see [9])

1.3.5 The fourth-order explicit Runge-Kutta-Nyström method

Runge-Kutta-Nyström methods were designed for use on Hamiltonian systems, where we define a *Hamiltonian System* as a system of differential equations in $2n$ variables x_j, p_j for $j = 1, \dots, n$, which can be thought of in physical terms as a position vector $\mathbf{x}_j = (x_1, \dots, x_n)$ and corresponding conjugate momentum $\mathbf{p}_j = (p_1, \dots, p_n)$, where we can define a Hamiltonian function $H(x_1, \dots, x_n, p_1, \dots, p_n, t)$ such that if we know the system’s state at two times t_1 and t_2 , then the system’s phase-space trajectory (as given by $\mathbf{x}(t), \mathbf{p}(t)$) obeys Hamilton’s variation principle

$$\delta \int_{t_1}^{t_2} \left[\sum_j p_j(t) \frac{dx_j(t)}{dt} - H(\mathbf{p}(t), \mathbf{x}(t), t) \right] dt = 0, \quad (1.24)$$

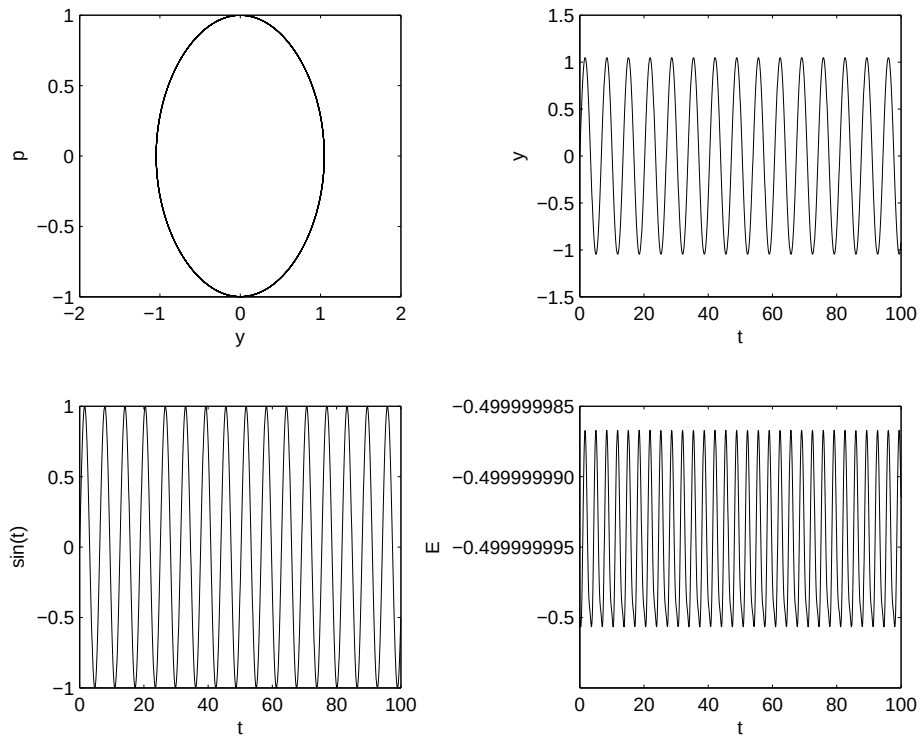


Figure 1.2: Clockwise from top left: phase plane, numerical solution plotted against time, energy plotted against time and exact solution plotted against time for the RKN method on the nonlinear simple pendulum given by $\dot{y} = p$, $\dot{p} = -\sin y$.

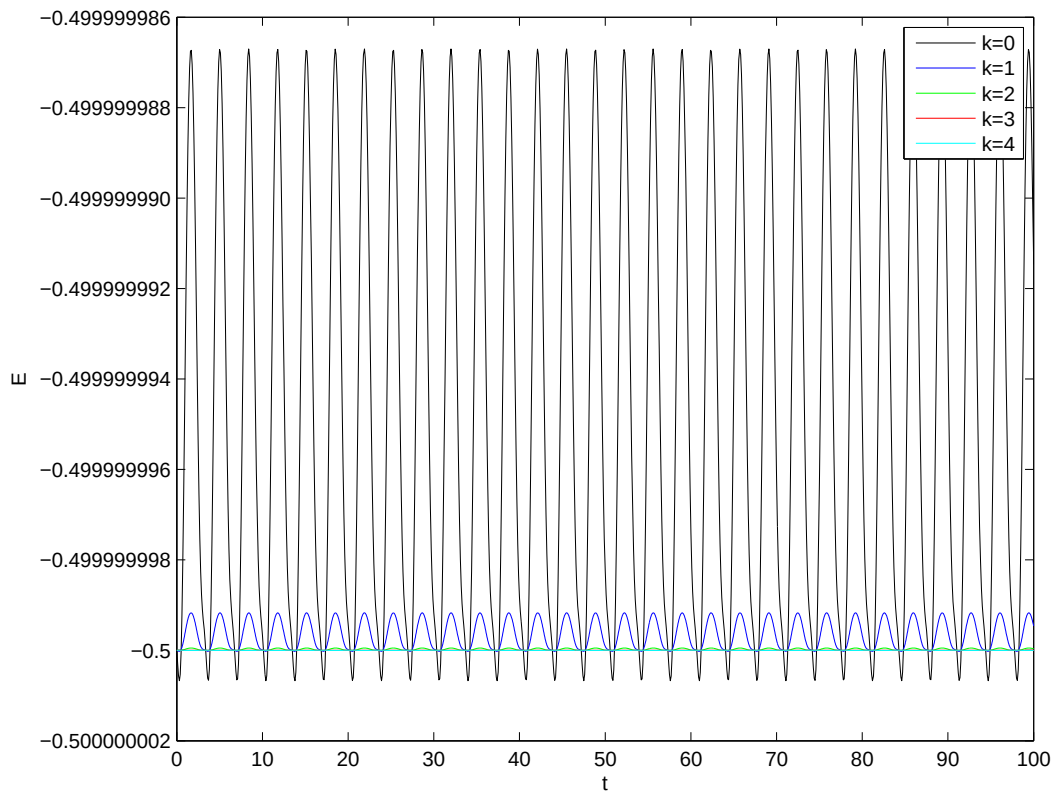


Figure 1.3: Superimposed plots of the energy for RKN on the nonlinear simple pendulum with timestep 0.1×2^{-k} , with k going from 0 to 5 from top to bottom. Observe the quartic convergence.

as defined in [26]. In this case, t acts as an external parameter in both the trajectories of the system and the Hamiltonian function. This variational integral (1.24) is exactly satisfied if the trajectories satisfy the Hamiltonian equations

$$\begin{aligned}\frac{dx_j}{dt} &= \frac{\partial H}{\partial p_j}, j = 1, \dots, n, \\ \frac{dp_j}{dt} &= -\frac{\partial H}{\partial x_j}, j = 1, \dots, n.\end{aligned}\tag{1.25}$$

In the case of an explicitly time-dependent (non-autonomous) system, we can introduce a new evolution parameter s and so parametrise the system's time evolution $t = t(s)$ (see [26]), thus rewriting the variation principle as

$$\delta \int \left[\sum_j p_j(s) \frac{dx_j(s)}{ds} - H(\mathbf{p}(s), \mathbf{x}(s), t(s)) \frac{dt(s)}{ds} \right] ds = 0.\tag{1.26}$$

The RKN method is designed to be explicit, symplectic, and capable of producing a reasonable order of accuracy (a k^{th} order RKN method with stepsize Δt will yield accuracy of $O(\Delta t^k)$). We prefer explicit schemes as they are easier to implement. In general, we cannot expect a fixed-step method to preserve energy, momentum and symplectic structure at the same time (see [10]). We can, however, expect an accurate value for the angular momentum (see [5]), and a value for the energy which is accurate to within $O(\Delta t^2)$ and is stable over the long term (i.e. will not display spurious growth or dissipation even after time scales of $t = O(10^3)$ or further). Adding a forcing term with multiple controls will mean that the system is dissipative or increasing in energy, even if the unforced system was Hamiltonian, and so will not have a symplectic structure, but we can still use a symplectic integrator in order to benefit from the long-term stability and accuracy (to within the margin of error mentioned above) of the invariants (see [10]). This method may, however, develop small-scale inaccuracies in phase over the longer time scales ($t \geq 1000$).

A general s stage Runge-Kutta-Nyström method for a system of equations of the form

$$\begin{aligned}\dot{x} &= p, \\ \dot{p} &= F(x, t),\end{aligned}\tag{1.27}$$

is specified by the tableau of real constants γ_i , α_{ij} , b_i and β_i for $i, j = 1, \dots, s$, and is

of the form

$$\begin{aligned}
X_i &= x_n + \Delta t \gamma_i p_n + \frac{\Delta t^2}{2} \sum_j \alpha_{ij} F(X_j), \\
p_{n+1} &= p_n + \Delta t \sum_i b_i F(X_i), \\
x_{n+1} &= x_n + \Delta t p_n + \frac{\Delta t^2}{2} \sum_i \beta_i F(X_i).
\end{aligned} \tag{1.28}$$

We shall need to impose conditions on the tableaux in order to produce an explicit, symplectic and consistent method of fourth-order accuracy. For the method to be explicit, we need

$$\alpha_{ij} = 0, \tag{1.29}$$

for $i \leq j$. For consistency, we require

$$\sum b_i = 1, \tag{1.30}$$

and the accuracy conditions for fourth order we also impose are

$$\begin{aligned}
\sum_i b_i \gamma_i &= \frac{1}{2}, \\
\sum_i b_i \gamma_{ij}^2 &= \frac{1}{3}, \\
\sum_i b_i \alpha_{ij} &= \frac{1}{6}.
\end{aligned} \tag{1.31}$$

To make the method symplectic, we need

$$b_i(\beta_j - \alpha_{ij}) = b_j(\beta_j - \alpha_{ji}). \tag{1.32}$$

We use a five-stage ($s = 5$) method with the tableaux computed by Sanz-Serna and Calvo (see [23]) to satisfy the conditions given in (1.29)-(1.32).

$$\begin{aligned}
\gamma &= \begin{pmatrix} 0 \\ 0.20517766 \\ 0.60819894 \\ 0.48727806 \\ 1 \end{pmatrix}, \\
b &= \begin{pmatrix} 0.06175885 \\ 0.33897803 \\ 0.61479131 \\ -0.14054801 \\ 0.12501982 \end{pmatrix},
\end{aligned}$$

$$\alpha_{ij} = b_j(\gamma_i - \gamma_j), \quad (1.33)$$

for $1 \leq i, j \leq 5$, for the nonzero α_{ij} elements, and

$$\beta_i = b_i(1 - \gamma_i), \quad (1.34)$$

$1 \leq i, j \leq 5$. This gives us a fourth-order accurate explicit symplectic method.

1.4 Existing algorithms

There already exist algorithms that make use of similar results and ideas to solve ODEs. We conduct a brief survey of the literature on these ideas, and consider two existing algorithms that are related to the one we shall go on to describe in Section 1.5, and then to construct mathematically in Chapter 3. We shall also consider how our algorithm differs from the existing algorithms, and what these differences imply for its applications and effectiveness.

1.4.1 Approximating linear differential equations with Magnus series

Consider a linear differential equation of the form

$$dy = a(t)ydt, \quad (1.35)$$

whose solution is of the form $y(t) = e^{\sigma(t)}y$. Specifically, we consider the case where $y_0 \in G$, where G is a Lie group, and $a : \mathbb{R}^+ \rightarrow \mathfrak{g}$, where $\mathfrak{g}$ is the Lie algebra of G . In this case, $\sigma(t) \in \mathfrak{g}$, and so $y(t) \in G$. This case is worth considering for the production of methods that preserve the Lie group structure by solving for σ .

An example of such a method is described in [6], using Magnus's expression for $\sigma(t)$ as a series expansion in terms of iterated integrals of the commutators (see [31]). The Magnus series for $\sigma(t)$ is given by

$$\begin{aligned} \sigma(t) = & \int_0^t a(\kappa)d\kappa + \frac{1}{2} \int_0^t [a(\kappa), \int_0^\kappa a(\xi)d\xi]d\kappa \\ & + \frac{1}{4} \int_0^t [a(\kappa), \int_0^\kappa [a(\xi), \int_0^\nu a(\nu)d\nu]d\xi]d\kappa + \dots, \end{aligned} \quad (1.36)$$

which also satisfies the equation

$$\sum_{k=0}^{\infty} \frac{(-1)^k}{(k+1)!} a d^k(\sigma', \sigma) = a, \quad (1.37)$$

for $t \geq 0$, $\sigma(0) = 0$, where we define

$$\begin{aligned} ad^0(p, q) &= p, \\ ad^k(p, q) &= [ad^{k-1}(p, q), q], \end{aligned} \quad (1.38)$$

and

$$\sigma' = a + \frac{1}{2}ad(a, \sigma) + \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!} ad^{2k}(a, \sigma), \quad (1.39)$$

where B_m is the m^{th} Bernoulli number.

There remains the issue of where to truncate the series in order to produce an efficient expansion. We can identify the expansion terms with rooted trees, and then truncate the series to obtain

$$\sigma_p(t) = \sum_{k=0}^{p-1} \sum_{\omega \in T_{3k+1}} a(\omega) \int_0^t H_\omega(\kappa) d\kappa, \quad (1.40)$$

where T_m is the set of all rooted trees with m vertices, H_ω is the function associated with the tree ω , and

$$a(\omega) = f_r \prod_{t=1}^r a(\omega^{[i]}), \quad (1.41)$$

where $\omega^{[i]}$ is a sub-tree of ω with i vertices and $f_r = B_r/r!$.

By combining the Magnus series with the use of quadrature to approximate the integrals, Iserles *et al* obtain the following 4th order scheme:

$$\begin{aligned} a_1 &= a(t_n + (\frac{1}{2} - \frac{\sqrt{3}}{6})n), \\ a_2 &= a(t_n + (\frac{1}{2} + \frac{\sqrt{3}}{6})n), \\ \sigma^n &= \frac{1}{2}(a_1 + a_2) + \frac{\sqrt{3}}{12}h^2[a_2, a_1], \\ y_{n+1} &= e^{\sigma^n} y_n \end{aligned} \quad (1.42)$$

Iserles notes that this method is significantly better than standard Runge-Kutta methods, in terms of accuracy and stability. It does, however, rely on $a(t)$ being smooth, whereas the scheme outlined in this thesis draws on ideas from rough path theory and should therefore be suitable for use with non-differentiable driving paths.

1.4.2 Fast Multipole Methods

Greengard and Beatson (see [2]) outline a group of Fast Multipole Methods (FMM) for solution of n -body problems. Specifically, they address means of computing sums of the form

$$u(x, y) = \sum_{i=1}^n w_i K(x, y_i), \quad (1.43)$$

for example the gravitational potential

$$\phi(y_j) = \sum_{i=1, i \neq j}^n \frac{m_i}{||x_i - y_j||} \quad (1.44)$$

given by n bodies at positions x_i for a large number of target points y_j .

Fast Multipole Methods require the following:

- A specified accuracy ϵ ;
- A hierarchical subdivision of space;
- A far field expansion of $K(x, y)$ in which we can separate the influence of source and evaluation points;
- A way to convert the far field expansion to a local expansion.

We choose the division of space with a number of levels such that on average there are s particles per box on the finest level. We begin at the finest level and create a multipole expansion for the source positions and strengths, then move up to higher levels by merging the lower-level boxes. Having reached the coarsest level, we convert the multipole expansion into a local expansion about the centre of all boxes in the interaction list for the box containing the evaluation point (choosing the $(2n + 1)^2$ nearest boxes gives an error of $O(10^{-4})$ for a two-dimensional problem with $n = 6$).

The algorithm we are constructing in this thesis has certain qualities in common with the FMM scheme; we shall choose a maximum acceptable local error, and then place the triangulation points of the partition of unity in order to produce the desired level of accuracy. Instead of using a hierarchical grid, we can fine-tune the level of accuracy by careful positioning of the triangulation points for the partition of unity. We are also less concerned with the near field/far field distinction than with the evolution of the vector fields over time.

1.5 A high-level description of the algorithm

We shall construct the algorithm mathematically in Chapter 3, but here we begin by giving a high-level description of the algorithm, and explaining how the results we have mentioned in this chapter are used in it.

We perform a two-step reduction of the problem given by equation (1.1). First, we project the solution trajectory down from the infinite-dimensional space of vector fields controlling it to a finite-dimensional subspace, using the Taylor series expansions along with the partition of unity to provide a basis for the global vector field on that subspace. Secondly, we model the highly oscillatory data over each given time step as a rough path, and then use the lower-order terms of the signature over each time interval in order to find the endpoint of the trajectory at the end of that time interval.

At the beginning of each time interval $(\mathbf{y}_i, t_i)$, we need to calculate the truncated Taylor series approximations $\mathbf{F}_n^i$ (for $i = 1, \dots, N$). We can then find the sum of all N of these Taylor series.

We need to bear in mind that the vector fields we have just calculated depend on the truncated Taylor series $\mathbf{F}_n^i$ we computed above, and so will only be accurate within the ball $B(\mathbf{a}_j, r_j)$. This will be problematic if the trajectory $\mathbf{y}_t$ passes outside this radius before the end of the time step, so we need to take account of the influence of the vector fields outside $B(\mathbf{a}_j, r_j)$. To that end, we shall use a partition of unity to interpolate smoothly between the values of the log-signatures in the different regions.

The approximated vector fields on the patches that the trajectory is passing should commute (although the approximations on different patches will not commute in general) as they are approximations to the continuously varying global vector field over which the trajectory moves. If the approximated vector fields do not commute, then the patch size is too large, the approximations are not accurate or the smoothing using the partition of unity is not sufficient. Bearing this in mind, we need to be careful when choosing the size of the patches.

The global vector field is of the form

$$\mathbf{V}(\mathbf{y}_t, t) = \sum_{j=1}^J \sum_{i=1}^N \mathbf{F}_{n,j}^i(\mathbf{y}_t, y) P_j(\mathbf{y}). \quad (1.45)$$

We can now separate variables, and consider V as the contraction of a series of vector fields $G_{j,\mathbf{k}} = (\mathbf{y} - \mathbf{a}_j)^{\mathbf{k}} P_j$ and a driving path $\{X_t^{j,\mathbf{k}}\}_{j=1, \mathbf{k} \geq 0}^{J, |\mathbf{k}|=n}$, for which $dX_t^{j,\mathbf{k}} = c_{j,\mathbf{k}}(t)dt$ where $c_{j,\mathbf{k}}$ is the $\mathbf{k}^{th}$ Taylor coefficient for the truncated Taylor series evaluated around the point $\mathbf{a}_j$.

We calculate the path's first m terms of the signatures and the corresponding truncated log-signatures, where m is chosen in order to provide enough terms of the signature to produce an approximation to the detailed behaviour of the system which falls within desired bounds of accuracy (we shall discuss how this choice is made in Chapter Three). We can then contract the truncated log-signatures $S_{t_i, t_{i+1}}^m(X_t)$ with the vector fields $G_{j, \mathbf{k}} = (\mathbf{y} - a_j)^{\mathbf{k}} P_j$ in order to obtain a flow map that will give us an ODE we can solve to find the trajectory over this time interval.

We repeat this procedure until we reach the time T , the end of our final time interval.

1.6 Thesis outline

In this thesis, we shall bring together the results we have discussed in this chapter to construct an algorithm which we can use to solve systems of ordinary differential equations driven by multiple controls.

As part of this process, we need to consider what would be suitable test problems for the algorithm. We have chosen three well-studied problems: simple harmonic motion, which we have chosen for its simple exact closed-form solution and can be used to study basic issues of accuracy and stability; the simple pendulum, which has a well-known if more complicated exact closed-form solution, but which is nonlinear; and the Airy equations, which we could use to compare the accuracy and stability of signature-based methods versus conventional Runge-Kutta methods. We shall also construct two new problems which could be used as further tests, and in Chapter Two, we shall discuss the the Interacting Harmonic Oscillators and Interacting Simple Pendula systems and investigate their properties.

In Chapter Three, we shall discuss the details of implementation of the full algorithm, and the various computational methods we can use for each of the steps. In Chapter Four, we shall consider how the algorithm can be practically implemented, and discuss some preliminary numerical results for the three well-studied test problems. In our final chapter, we shall consider what the next steps in the study, implementation and uses of the algorithm could be.

Chapter 2

The test problems

2.1 Introduction

We shall consider five test problems which could be used to test implementations of the our algorithm, or similar methods. Three of these problems are well-known and well-studied problems with closed form solutions: the harmonic oscillator, the nonlinear simple pendulum and the Airy equations. Each of these problems is of interest to us for its specific properties, which we shall discuss below.

We shall also construct and investigate two new test problems in this chapter. Both of these will be designed to provide a simple two-dimensional dynamical system driven by a large number of vector fields, of the form of a single oscillator driven by its interactions with a large number of other oscillators. The two problems differ only in the form of the equations for the oscillators in question; in one we shall use simple harmonic motion, in the other the simple pendulum.

2.2 Simple harmonic motion

The most basic test problem which we shall use for our methods is that of linear simple harmonic motion, as given by

$$\begin{aligned}\dot{y} &= p, \\ \dot{p} &= -y.\end{aligned}\tag{2.1}$$

The advantage of this as a test problem is that the analytic solution is already well-known, specifically

$$\begin{aligned}y(t) &= A \sin t + B \cos t, \\ p(t) &= A \cos t - B \sin t,\end{aligned}\tag{2.2}$$

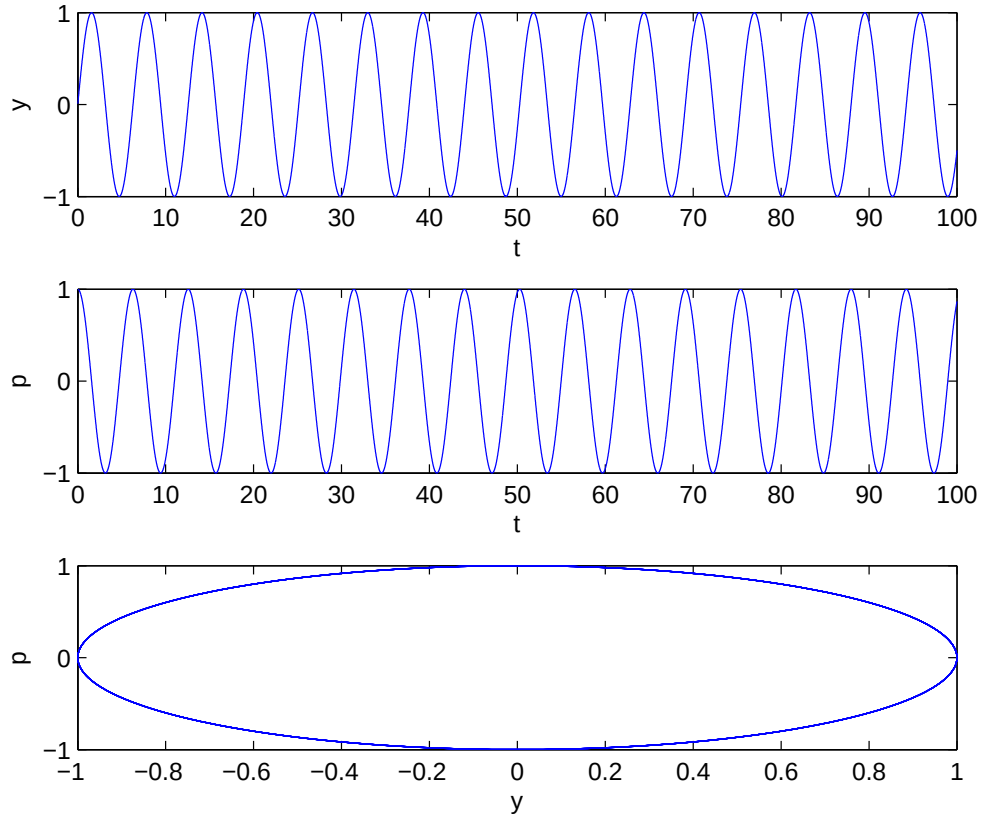


Figure 2.1: Evolution of y with time, p with time, and the phase plane for simple harmonic motion, from top to bottom, with initial conditions $y(0) = 0$, $p(0) = 1$.

where our initial data determines the values of the constant coefficients. We can therefore use this system as a test to estimate the accuracy of our methods, by comparing the results we get from them to the analytic solution (as shown in Figure 2.1) and considering the behaviour of the error.

The other interesting property of this system is its simple linear *flow map*, where we define a flow map as a map from the phase space into itself of the form $\mathbf{y}_t = \pi_t(\mathbf{y}_0)$. In this case, we have

$$\pi_t = \begin{pmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{pmatrix} \quad (2.3)$$

This will become relevant when we attempt to use the decomposition of the vector space using the Lie basis and the signatures to reconstruct the flow map for the algorithm.

2.3 The simple pendulum

Secondly, we consider the nonlinear test problem given by the nonlinear simple pendulum

$$\begin{aligned}\dot{y} &= p, \\ \dot{p} &= -\sin y,\end{aligned}\tag{2.4}$$

with Hamiltonian $H = \frac{1}{2}p^2 - \cos y$. We use the same initial conditions as for the linear problem.

The analytic solution for the nonlinear simple pendulum is a little more complicated: from the equations (2.4), we have

$$\begin{aligned}p\dot{p} &= -\sin y \dot{y} \\ \frac{1}{2}p^2 &= \cos y + C \\ \frac{1}{2}\dot{y}^2 &= \cos y + C \\ \dot{y} &= \pm\sqrt{2\cos y + 2C} \\ \pm t + t_0 &= \int \frac{dy}{\sqrt{2\cos y + 2C}};\end{aligned}\tag{2.5}$$

i.e. the relationship between t and y is given by an elliptic integral (see Figure 2.2), and hence can be evaluated independently, either symbolically or using quadrature. The Matlab quadrature routines are accurate to $O(10^{-6})$, whereas evaluating it symbolically and then substituting the numbers in can produce arbitrary accuracy up to machine precision but is slower than quadrature. In this case, we used the symbolic expression to evaluate the integral up to double precision.

2.4 The Airy equations

Consider the Airy equations

$$\begin{aligned}dy &= udt, \\ du &= \pm K^2 \frac{y}{2} dt,\end{aligned}\tag{2.6}$$

for some constant K . The general solution of this equation, in the case where the minus sign is present, is

$$y(t) = A' Ai(K^{2/3}t) + B' Bi(K^{2/3}t),\tag{2.7}$$

where A' and B' are constants, and $Ai(t)$ and $Bi(t)$ are the Airy functions, which are given by

$$\begin{aligned}Ai(t) &= \frac{1}{\pi} \int_0^\infty \cos\left(\frac{1}{3}s^3 + st\right) ds, \\ Bi(t) &= \frac{1}{\pi} \int_0^\infty \exp\left(-\frac{1}{3}s^3 + st\right) \sin\left(\frac{1}{3}s^3 + st\right) ds.\end{aligned}\tag{2.8}$$

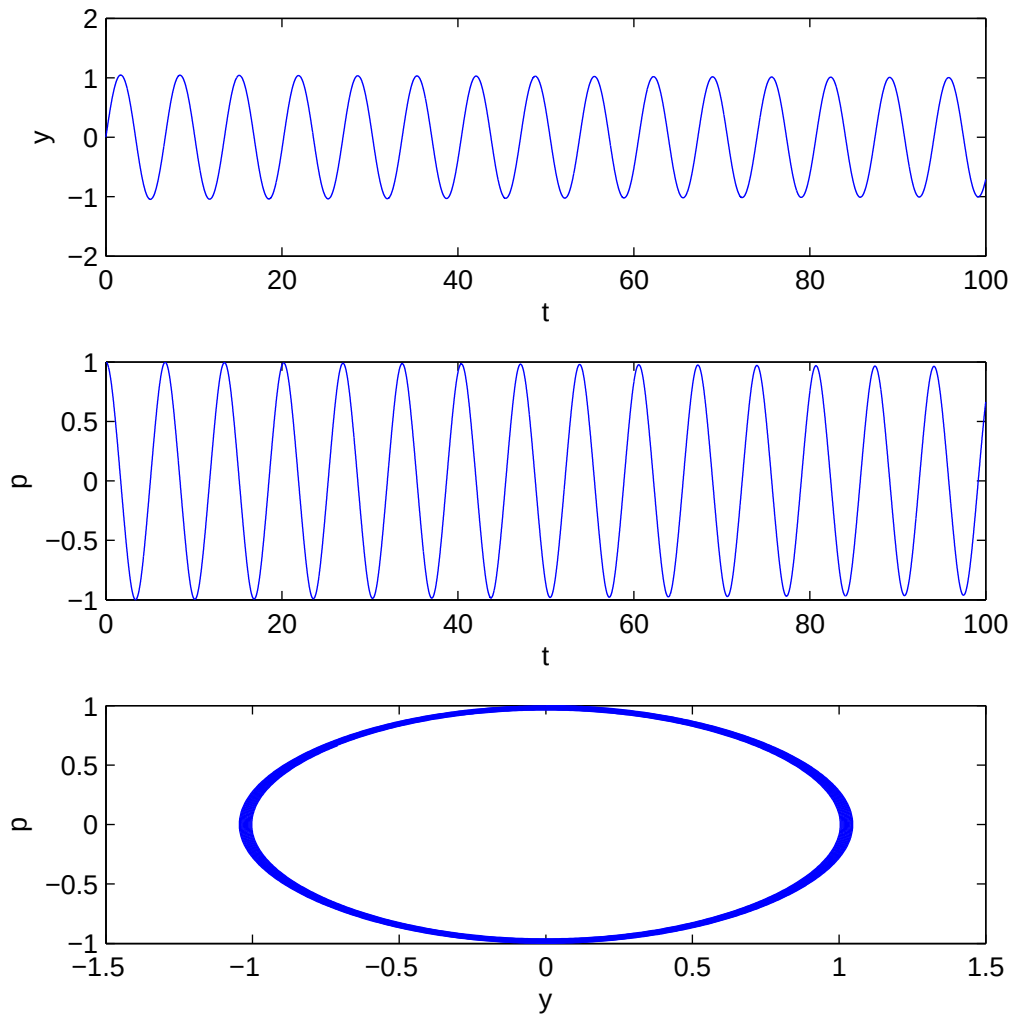


Figure 2.2: Evolution of y with time, p with time, and the phase plane for the nonlinear simple pendulum, from top to bottom.

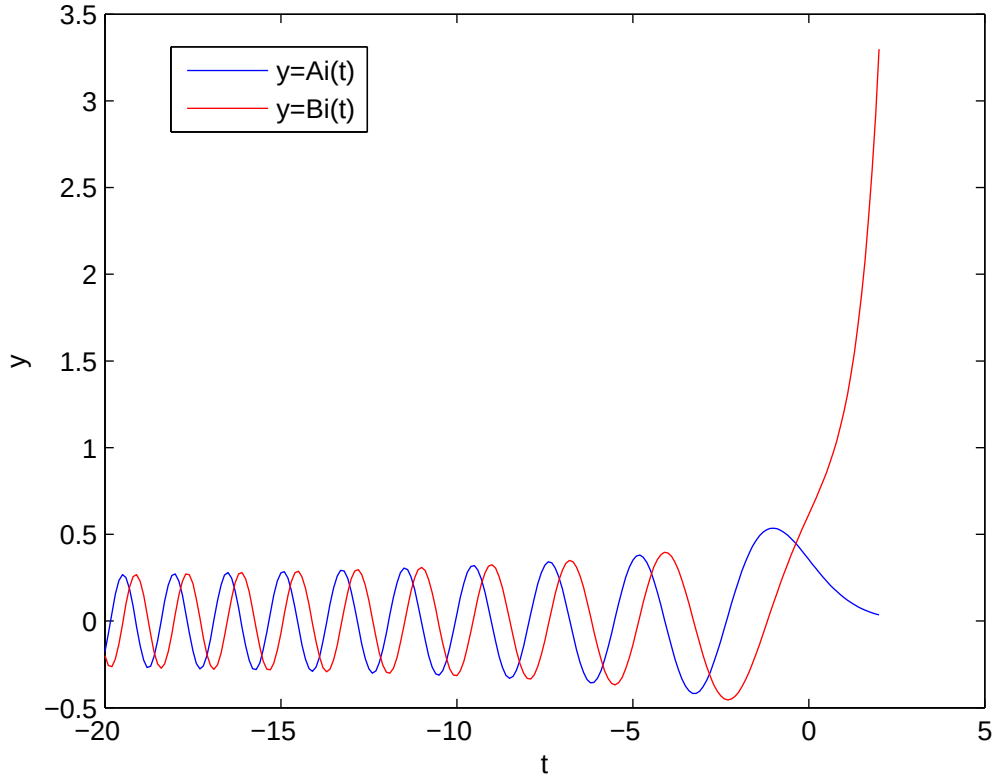


Figure 2.3: Behaviour of $Ai(t)$ and $Bi(t)$, showing the shift from oscillatory to exponential behaviour at $t = 0$.

We are principally interested in the Airy equations since they are used as a test problem by Iserles for the Magnus series based algorithm we discussed in Chapter 1 (see [6]). These equations are the simplest linear differential system with a turning point where the solutions change from oscillatory to exponential. For positive t , $Ai(t)$ is positive, convex and decreasing to zero, whilst $Bi(t)$ is positive, convex and exponentially increasing, whereas for negative t both solutions are oscillatory, with frequency increasing with the magnitude of t and amplitude decreasing (see Figure 2.3). As a test problem, we would want our algorithms to be successful in reproducing the oscillatory behaviour, even for the higher-frequency oscillations for $|-t|$ large.

2.5 The Interacting Pendula systems

We shall construct a specific example of a dynamical system with a large number of controls, which we shall call the Interacting Pendulum systems. This can be thought

of as a highly simplified model of a mechanical system where a large number of particles (which we shall refer to as *control* particles, because their influence provides an input or control for the system) interact with a single particle (which we shall refer to as the *response* particle, because its dynamics provide an output or response from the system). Our fundamental simplification when considering this model comes from the assumption that the cloud of control particles exerts a small but non-negligible influence on the motion of the response particle, but that the control particles do not affect each other in the same way, nor are they affected by the response particle.

In this case, the trajectories of the N control particles (given by $(\mathbf{x}_i, \mathbf{p}_i)$ for $i = 1, \dots, N$) obey a Hamiltonian set of equations, and the response particle's trajectory (y, p) is given by similar equations but with an additional forcing term given by the sum of its interactions with the control particles. We shall stipulate that both the cloud of control particles and the single response particle are moving on the integral curves of the vector field which specifies the solutions of either the simple harmonic motion or nonlinear simple pendulum equations.

We shall now construct the Interacting Pendula systems. First, we let the N control particles have position $x_i(t)$ and corresponding momenta $p_i(t)$ for $i = 1, \dots, N$, where $x_i(t), p_i(t)$ are given by the solutions of either the SHM or simple pendulum equations. The response particle's position $y(t)$ and momentum $p(t)$ are also given by the SHM or simple pendulum equations, with the addition of a forcing term given by the influence of the control particles, as follows:

$$dy = p dt, \tag{2.9}$$

$$dp = (F^0(y) + \sum_i^N \Gamma_i(y, p, x_i, p_i, t)) dt. \tag{2.10}$$

Each component Γ_i of the forcing term is given by the interactions between the i^{th} control particle and the response particle. We assume that the attraction between each control particle and the response particle takes the form of a vector field of a form similar to a gravitational attractive force, governed by a similar inverse square law, and so we have

$$\Gamma_i = G \frac{x_i - y}{|x_i - y|^3}, \tag{2.11}$$

where x_i is the position of the i th control particle and $G \ll 1$ is a parameter (analogous to the gravitational constant).

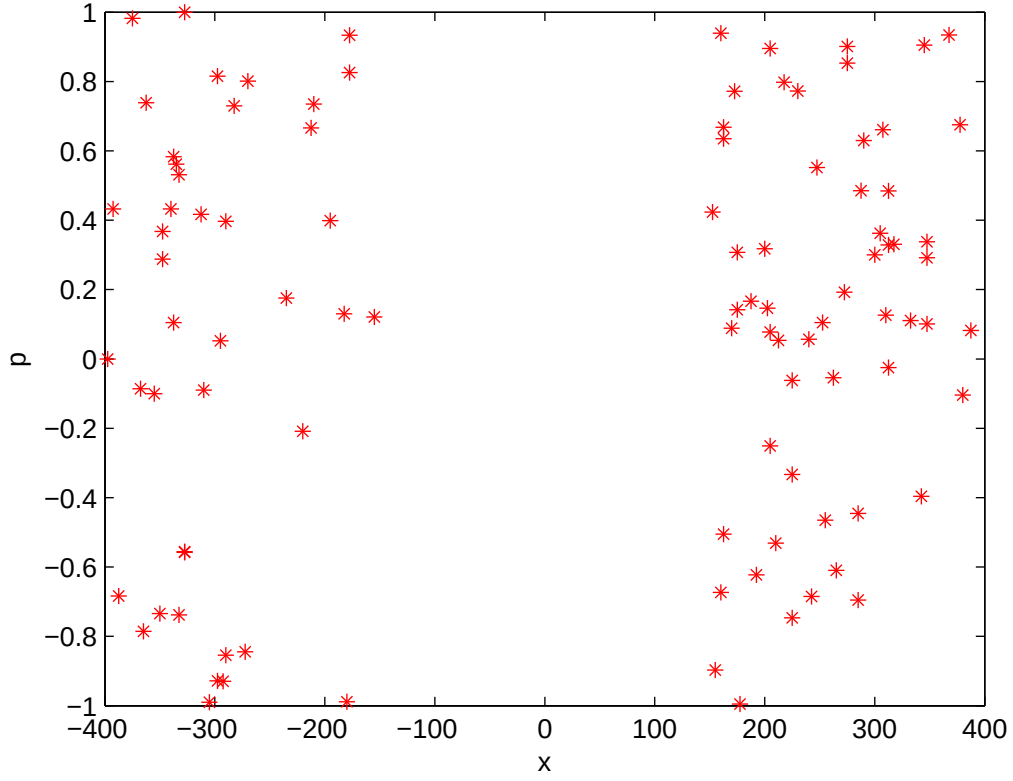


Figure 2.4: Plots of $x_i(0)$ and $p_i(0)$; the initial conditions for 100 control particles in two clouds, beginning at $x = 150$ and $x = -150$.

2.5.1 Initial conditions

In order to minimise the problems associated with very close transits between the control and response particles, specifically large jumps in the energy and angular momentum of the response particle, we wish to keep the control particles well away from the response particle. We need to place the response particle in such a way that its distance of closest approach to the control particles is greater than $O(10^{-4})$, in order to keep the magnitude of the jumps in the energy at a smaller order of magnitude than the energy itself. We will discuss the energy concerns in more detail later on. By placing it at a sufficiently large distance from the control particles, we ensure that their influence acts to perturb its orbit but is not strong enough to pull the response particle in to the cloud of control particles, as intersections between the control and response particle will lead to singularities.

We place the response particle initially at the point $y_0 = 0$, $p_0 = 1$, so that its behaviour is comparable to the unperturbed pendula we considered earlier. We then

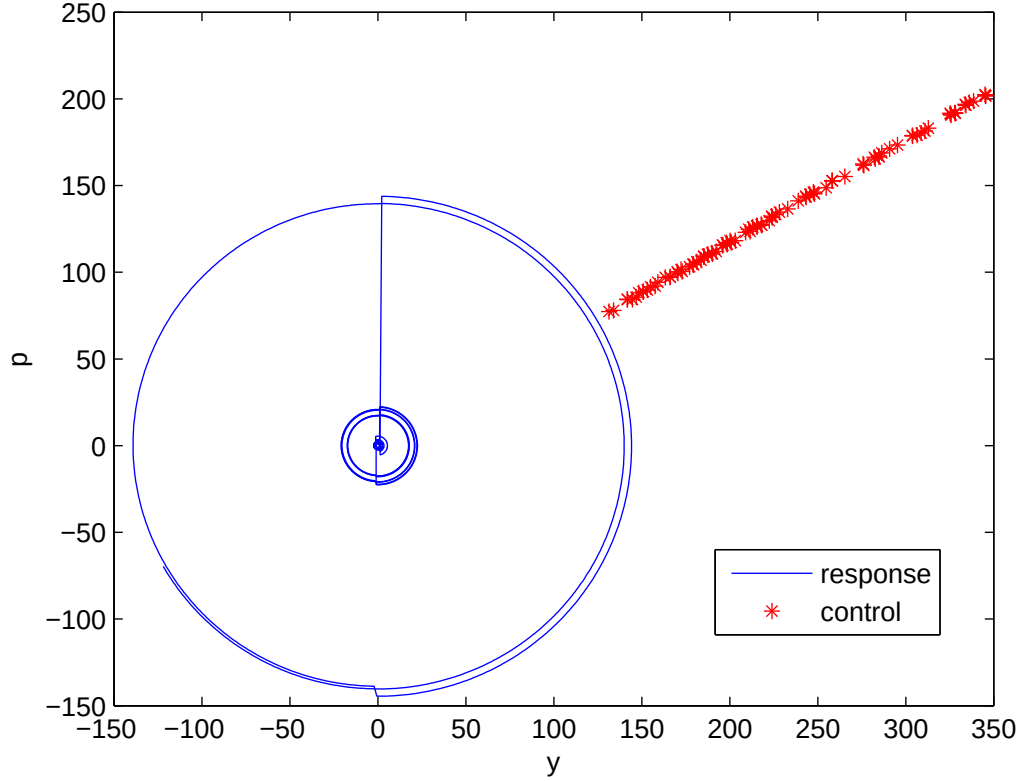


Figure 2.5: Plots of y and p for the Interacting Harmonic Oscillators, where $G = 0.1$, with a single cloud of control particles centred around $x = 150$.

randomly distribute the N control particles in between two control clouds, for which $250 + r_c > x_i > r_c$ and $-r_c > x_i > -r_c - 250$ and $-1 < p_j < 1$, for some value r_c . We assign each control particle to a randomly chosen cell of width 2.5 in the x direction. The choice of r_c depends on the strength of the influence of the control particles, in order to deal with the close transits problem.

We observe from Figure 2.5 that using a single cloud of control particles results in the response particle being pulled into the control cloud, whereas two clouds of control particles prevent there from being an overwhelming resultant force in either direction on the response particle.

We shall solve the equations for the Interacting Pendula systems using a standard adaptive step fourth order Runge-Kutta-Nyström solver as described in Chapter 1, with base step size $\Delta t = 0.01$. We shall thus generate two sets of data sets, one for which $G = 0.01$ and one for which $G = 0.1$.

2.6 Investigation of the Interacting Pendula systems

We shall define those of the models' features that we may wish to use as descriptive statistics in order to compare the results produced by the algorithm to the results from solving the equations using a fourth order symplectic Runge-Kutta-Nyström solver. First, we shall give a qualitative description of the behaviour of the response particles. Secondly, we define basic features of the time series, such as their mean value, amplitude, whether they are periodic and if so of what period, skewness and kurtosis, and we shall also consider the energy and angular momentum of the response particle. Finally, we shall estimate the first few terms of the signature of the time series from these data sets.

2.6.1 A qualitative summary of the behaviour of the response particle

We note that for the IHO system (see Figures 2.6-2.7), we have periodic oscillatory behaviour in both the y and p directions, whose amplitude increases sharply at certain points and grows slowly in between those jumps, but whose frequency is not greatly affected. Both variables increase in magnitude simultaneously; the sharp increases correspond to the response particle jumping to a wider elliptical orbit, whereas the slow growth in amplitude corresponds to the orbit spiralling outward between the jumps. For the ISP system (see Figures 2.8-2.9), we observe the slow growth without the same sharp increases.

Due to the simplifications we have made in designing this model, energy can be transferred from the control particles to the response particle, but not vice versa. Particularly close transits between the control and response particles result in large transfers of energy to the response particle, pushing it to a larger orbit. We note from Figure 2.10 that the increases in amplitude in y correspond to closer transits between the control and response particles.

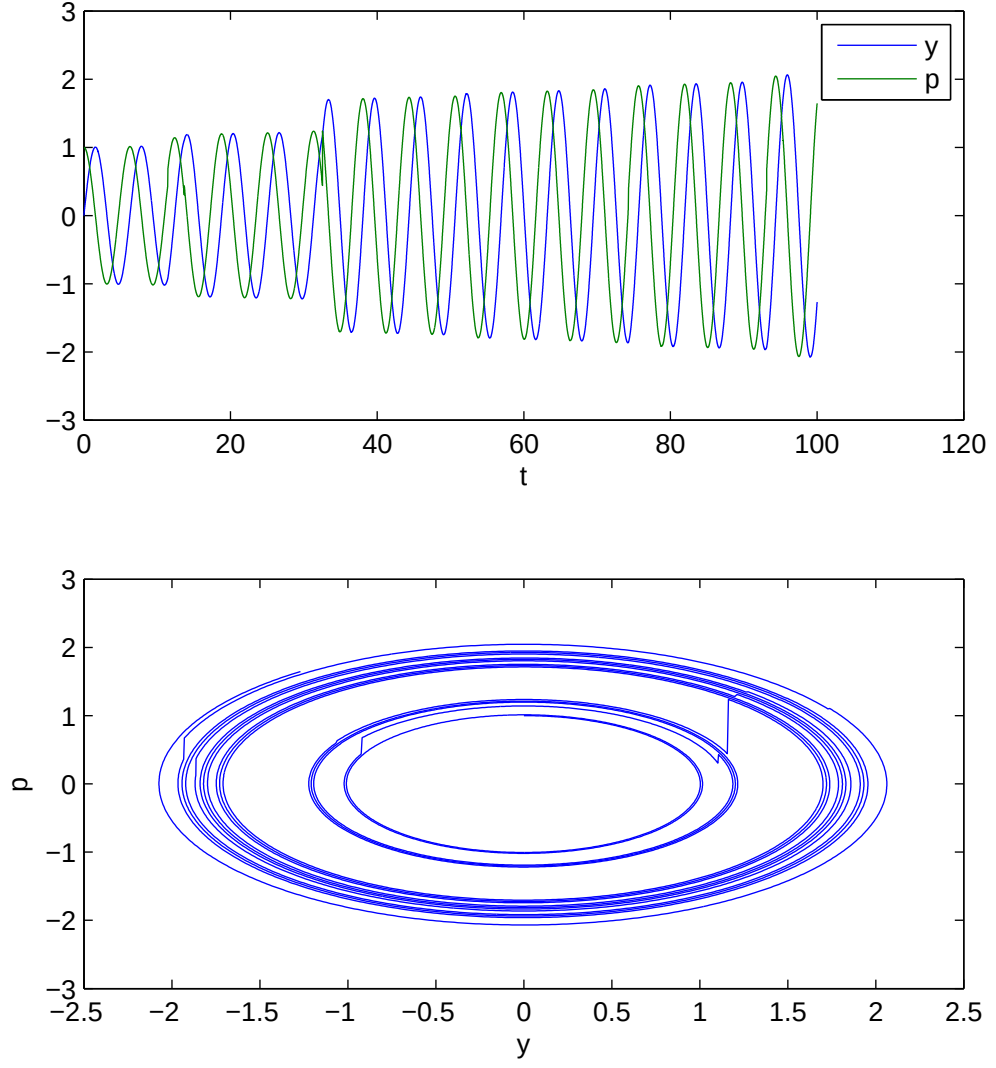


Figure 2.6: Plots of y and p against time, and the phase plane for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$. We shall refer to this as Data Set 1.

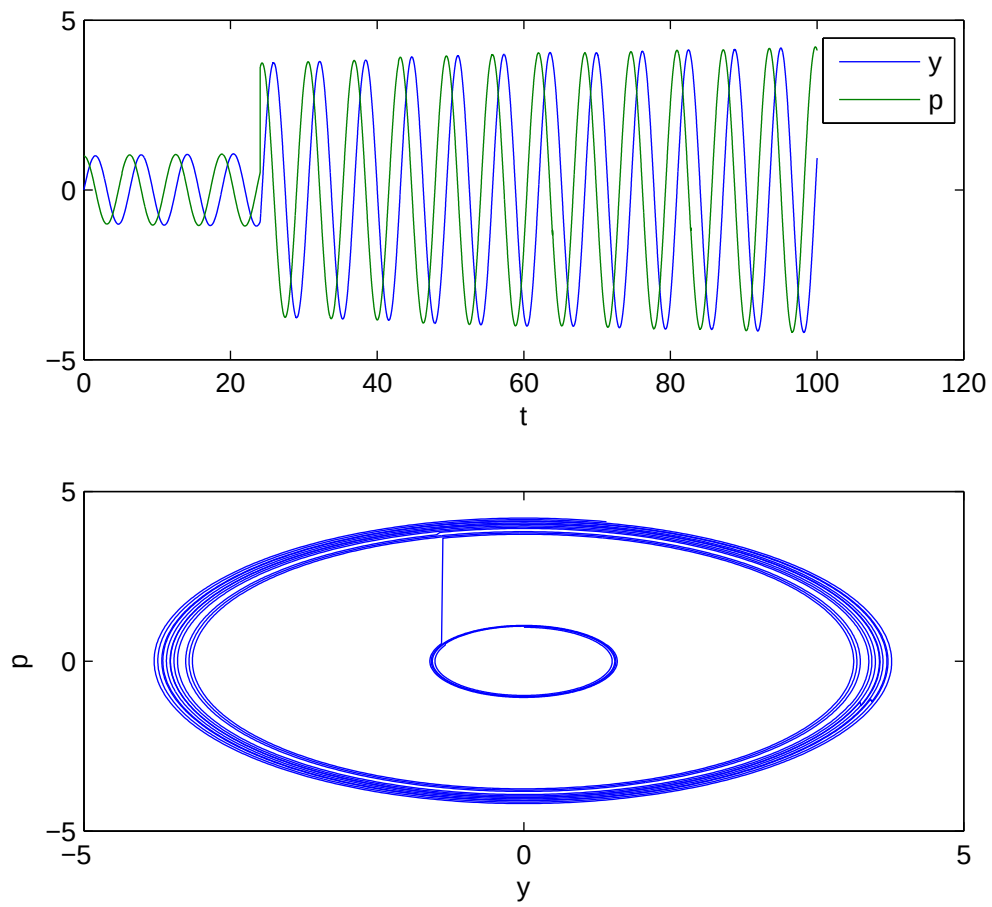


Figure 2.7: Plots of y and p against time, and the phase plane for the the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$. We shall refer to this as Data Set 2.

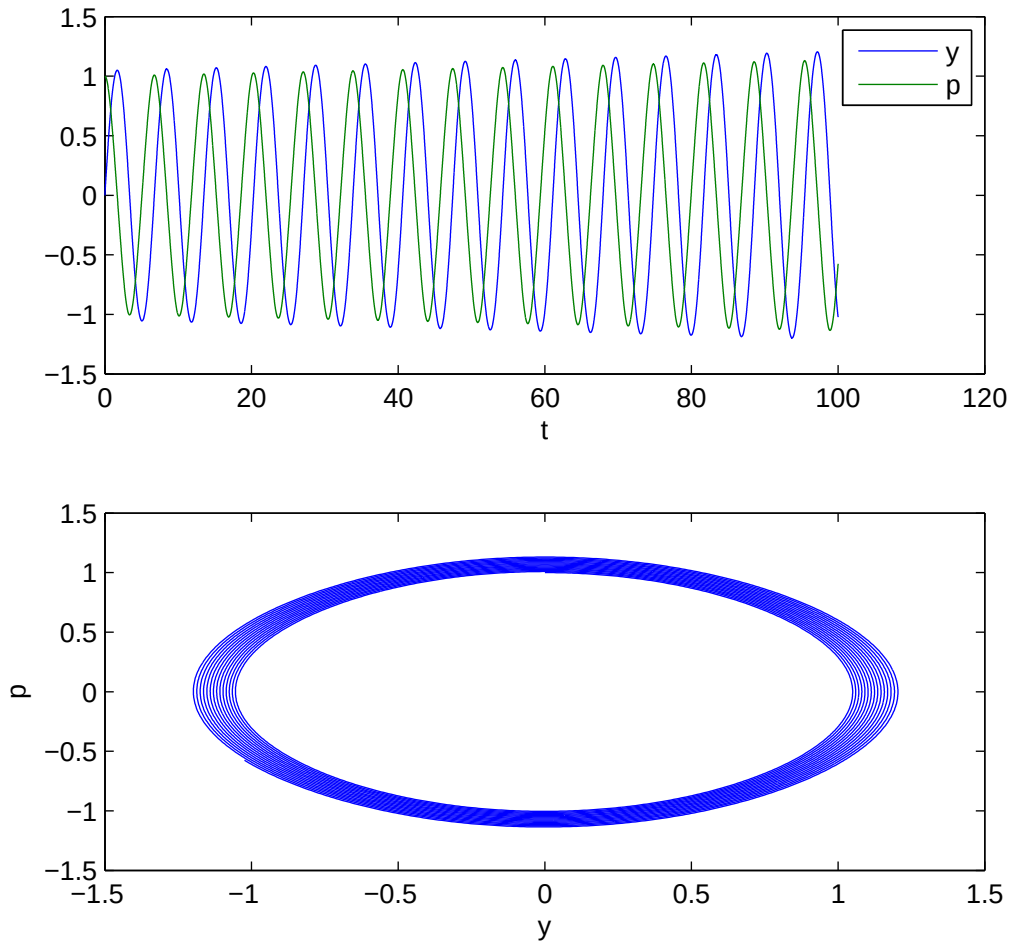


Figure 2.8: Plots of y and p against time, and the phase plane for the Interacting Simple Pendula, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$. We shall refer to this as Data Set 3.

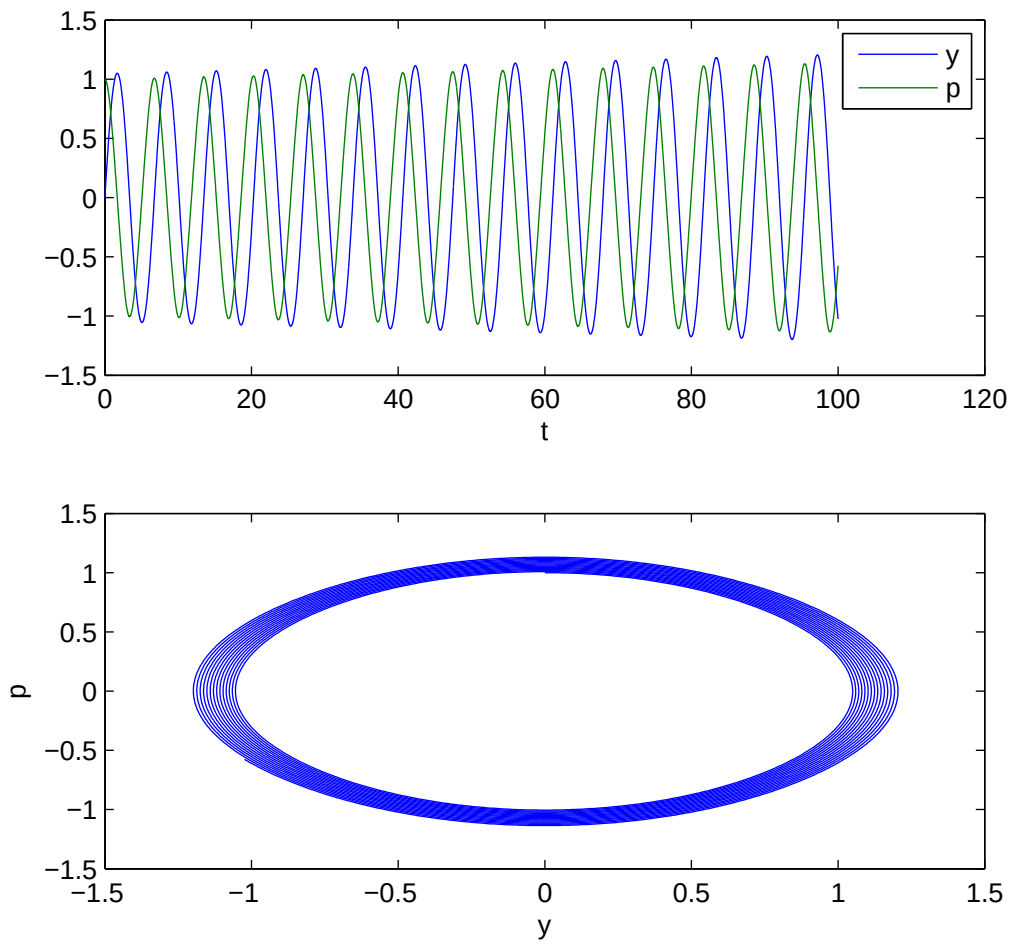


Figure 2.9: Plots of y and p against time, and the phase plane for the Interacting Simple Pendula, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$. We shall refer to this as Data Set 4.

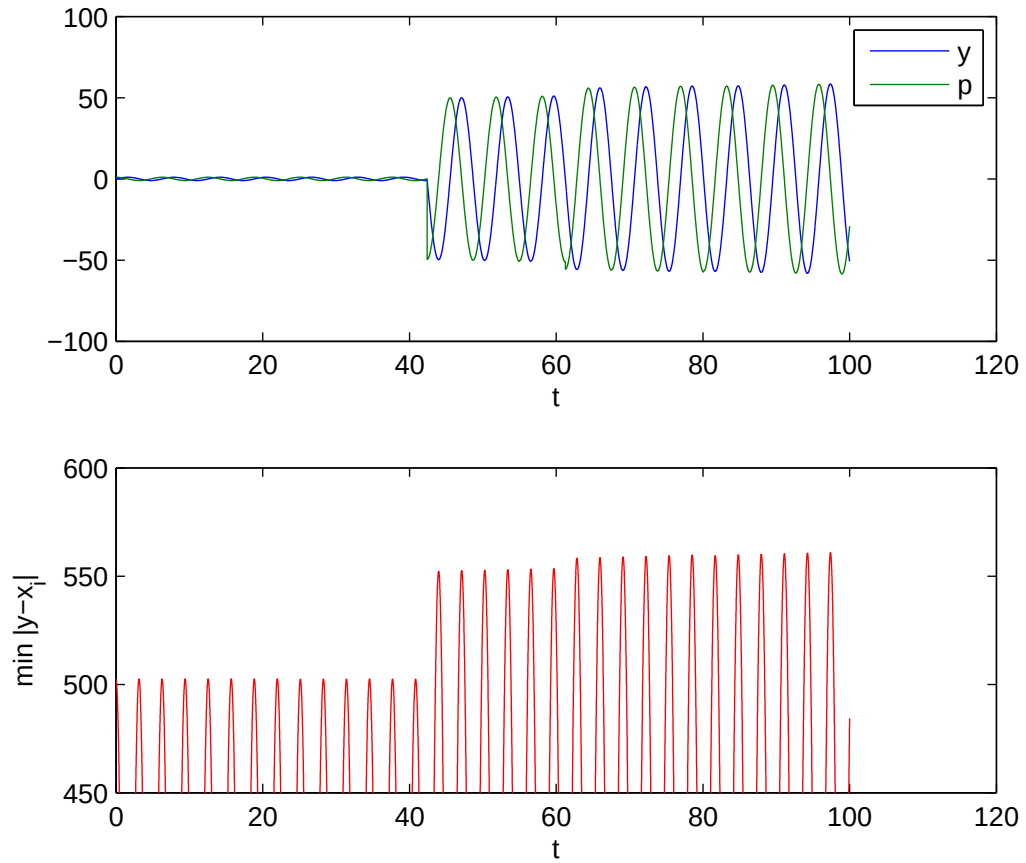


Figure 2.10: Plots of y and p (above) compared to $\min |x_i - y|$ (below) for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$. Observe the difference in the maximum amplitude of the close transits: the amplitude of the oscillation increases as the minimal distance apart decreases.

Table 2.1: Mean and standard deviation (accurate to 4 decimal places).

Dataset	$\mu(y)$	$\mu(p)$	$\sigma(y)$	$\sigma(p)$
1	0.0125	-0.0125	1.1751	1.1667
2	0.0047	0.0097	2.4822	2.4938
3	0.0185	-0.0103	0.8005	0.7414
4	0.0186	-0.0103	0.8005	0.7414

Table 2.2: Skewness and kurtosis (accurate to 4 decimal places).

Dataset	$g(y)$	$g(p)$	$K(y)$	$K(p)$
1	-0.0273	-0.0036	1.7199	1.7223
2	-0.0097	0.0094	1.9136	1.9035
3	-0.0346	0.0173	1.5004	1.5468
4	-0.0346	0.0173	1.5005	1.5468

2.6.2 Mean, standard deviation, skewness and kurtosis

We begin by considering four basic statistics for each time series: mean μ , standard deviation σ , skewness g and kurtosis K , where

$$g = \frac{\mathbb{E}(x - \mu)^3}{\sigma^3}, \quad (2.12)$$

$$K = \frac{\mathbb{E}(x - \mu)^4}{\sigma^4}, \quad (2.13)$$

i.e. skewness and kurtosis are the third and fourth moments of the distribution, scaled in terms of appropriate powers of the standard deviation. The skewness of the distribution can be thought of as a measure of its asymmetry: negative skewness means that the left tail is longer, and the bulk of the data is concentrated on the left, whereas positive skewness conversely means that the right tail is longer, and the data is concentrated on the right. Kurtosis can be thought of as a measure of the sharpness of the peak of the distribution; high kurtosis means that more of the variation is due to infrequent extreme deviations. The kurtosis of the normal distribution is 3; distributions that are more outlier-prone than the normal distribution have kurtosis greater than 3, whereas distributions that are less outlier-prone than the normal distribution have kurtosis less than 3.

We note from Table 2.1 that for most of the data sets, both $\mu(y) \approx 0$ and $\mu(p) \approx 0$; these values are close to the mean values for the unperturbed problems, but with an offset that is positive in the y direction but whose direction varies in the p direction. We also note that for all the data sets, $\sigma(y) \approx \sigma(p)$, and that increasing G by a factor

of 10 produces a difference of magnitude 10^{-2} in the mean of the IHO system, but one of 10^{-4} in the mean of the ISP system.

From Table 2.2, we note that all data sets have small negative skewness in the y direction, and all but the first have a small positive skewness in the p direction. This implies that all the data sets are mainly centrally weighted, with a slight tendency towards a longer left tail in the y direction, and a slight tendency towards a longer right tail in the p direction for all but the first. This effect is caused by the increasing amplitude over time, thus meaning that larger values become more likely for later time values. We also note that, for all data sets, the kurtosis is less than 3 and $K(y) \approx K(p)$, implying that the time series in both coordinates is not significantly outlier prone (again caused by the large jumps in amplitude at later time values). The kurtosis for both coordinates is higher for the IHO data sets than for the ISP ones, and the two ISP data sets display very similar kurtosis; this is due to the fact that the first two data sets display jumps in amplitude, whereas the latter two merely have the slow growth.

2.6.3 Periodicity

From Figures 2.11-2.14, we observe the time series in y and p in more detail. We note that, for the IHO data sets, the period of oscillation in both y and p is ≈ 6.2 , and that the period remains unchanged as the amplitude changes. We also note that the oscillation in y is smooth, whereas the oscillation in p displays more small jumps and irregularities brought on by the close transits, and that the jumps manifest in p before they affect y ; we can tell from this that the changes in energy affect the momentum before influencing the position. This occurs because the gravitational forces affect the equation for the first derivative of p , and will therefore manifest in p before y . The jumps in amplitude occur at $t = 32.52$ for Data Set 1 and $t = 24.15$ for Data Set 2.

Similarly, for the ISP data sets, the period of oscillation in both y and p is ≈ 6.7 . We note that these periods are slightly shorter than the period of oscillation for simple harmonic motion, which is ≈ 7.3 , and the period for the simple pendulum, which is ≈ 7.7 - approximately 1s shorter, respectively. We also observe that the oscillation in both variables is smooth.

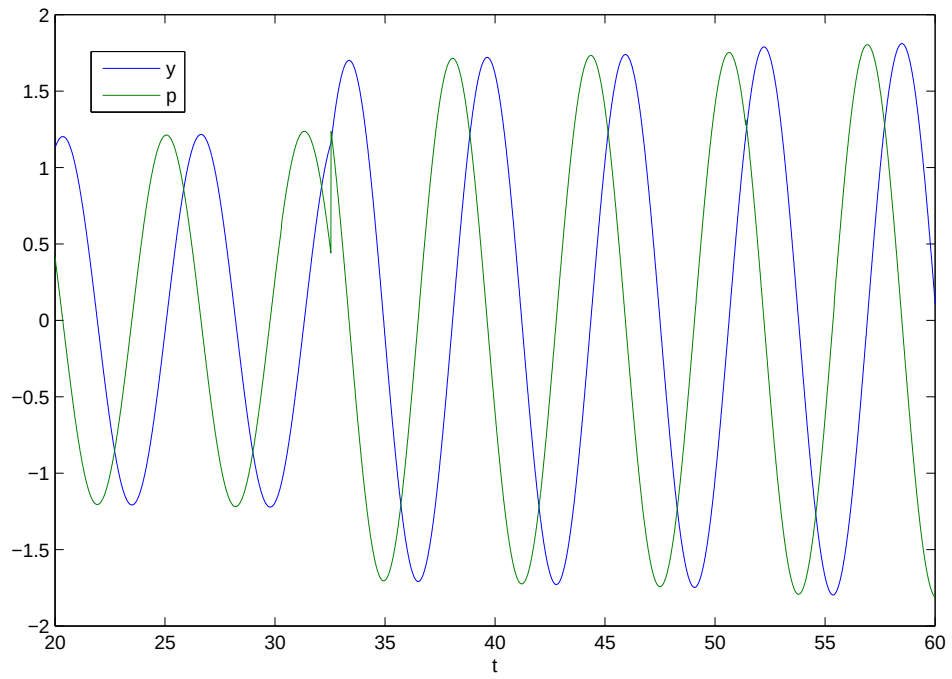


Figure 2.11: Plots of y and p against time for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$. We observe three different amplitude regimes within this diagram.

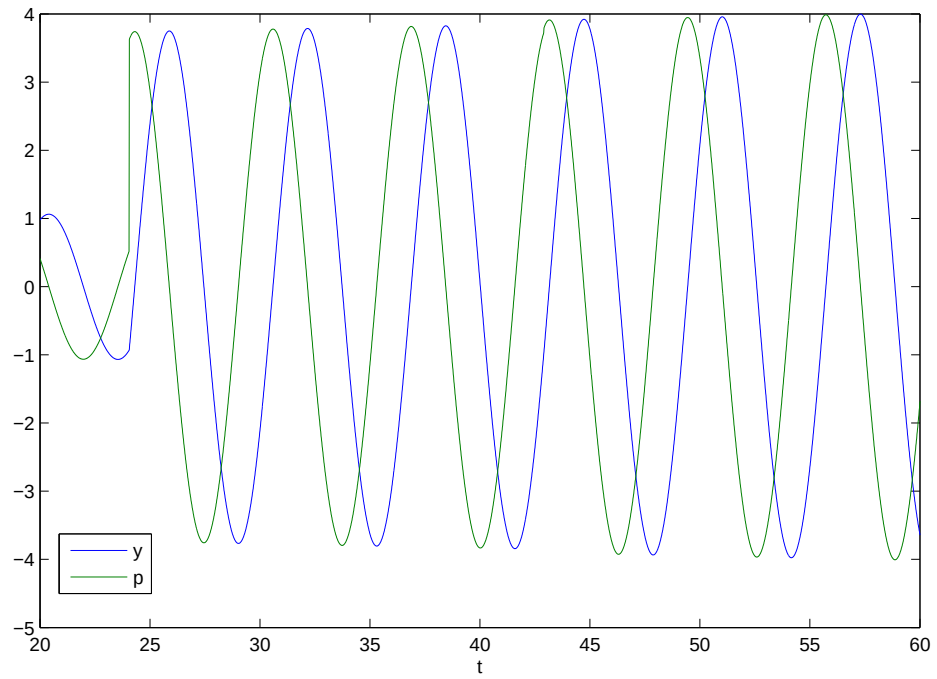


Figure 2.12: Plots of y and p against time for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$. We observe two different amplitude regimes within this diagram.

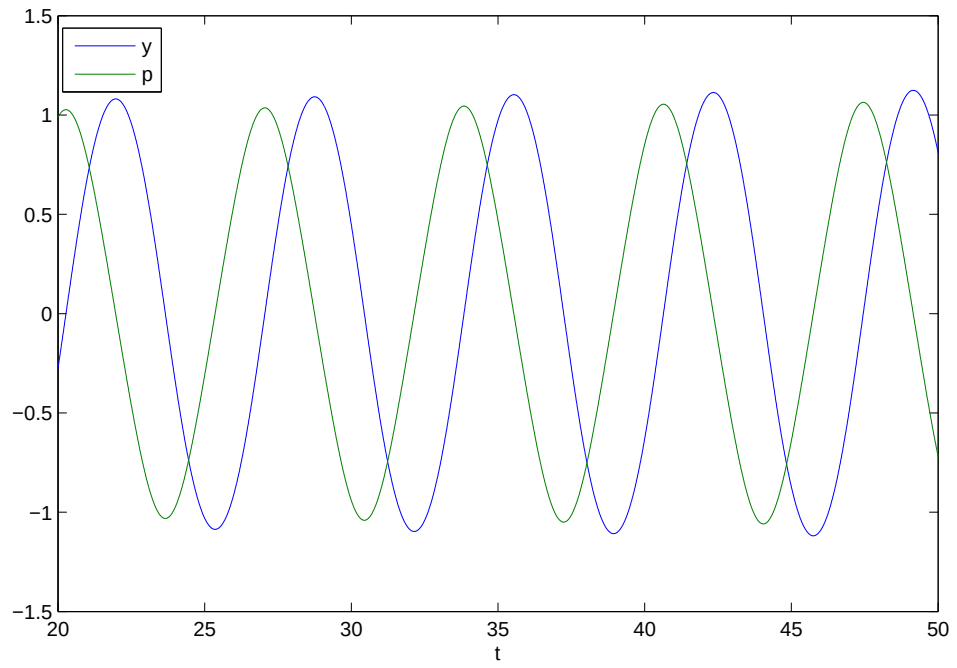


Figure 2.13: Plots of y and p against time for the Interacting Simple Pendula system, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$.

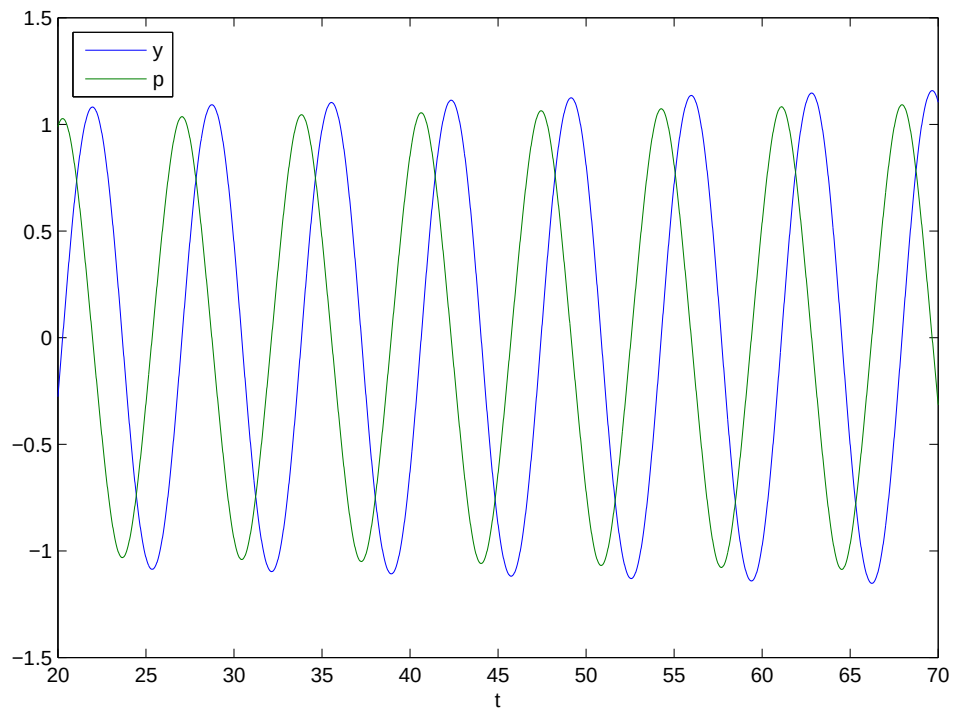


Figure 2.14: Plots of y and p against time for the Interacting Simple Pendula system, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$. We observe two different amplitude regimes within this diagram.

2.6.4 Energy

Both simple harmonic motion and the simple pendulum are Hamiltonian systems: the Hamiltonian for simple harmonic motion is given by

$$H = \frac{1}{2}p^2 + \frac{1}{2}y^2, \quad (2.14)$$

and the Hamiltonian for the simple pendulum is given by

$$H = \frac{1}{2}p^2 - \cos y. \quad (2.15)$$

The gravitational potential energy change given by the interaction of the response particle with a single control particle is

$$E_p = \frac{1}{|y(t) - x_i(t)|}, \quad (2.16)$$

and so the Hamiltonian for the Interacting Harmonic Oscillators system is given by

$$H = \frac{1}{2}p^2 + \frac{1}{2}y^2 + \sum_{i=1}^N \frac{G}{|y(t) - x_i(t)|}, \quad (2.17)$$

and the Hamiltonian for the Interacting Simple Pendula system is given by

$$H = \frac{1}{2}p^2 - \cos y + \sum_{i=1}^N \frac{G}{|y(t) - x_i(t)|}. \quad (2.18)$$

From Figures 2.15-2.18, we observe that the energy of the response particle for the IHO system is generally increasing through a series of steady levels, with frequent and irregular spikes of variable amplitude (of maximum $O(10)$ for Dataset 1, and $O(10^2)$ for Dataset 2). As we might have expected, the largest jumps to a higher energy level occurs immediately after the large jumps in amplitude of the oscillation of the response particle (i.e at $t = 32.58$ for Dataset 1, and $t = 24.05$ for Dataset 2), whereas the slow upward trend corresponds to the intermediate periods of slow growth in amplitude. We also note that the upward trend in the energy is not in fact completely steady, but contains other small jumps in energy which are not immediately noticeable in the oscillation.

From Figures 2.19-2.22, we observe that energy for the ISP system behaves in a similar way, but with fewer of the large irregular spikes. We can fit linear polynomials to the general upward trend, and the spikes do not seem to influence the amplitude of the response particle. In general, it appears that the spikes are caused by closer transits between the control and response particles and their effect on the $\sum_{i=1}^N G/|y(t) - x_i(t)|$ term, whereas the slow growth in energy is caused by the simplification that energy can be passed from the control particles to the response particle, but not vice versa.

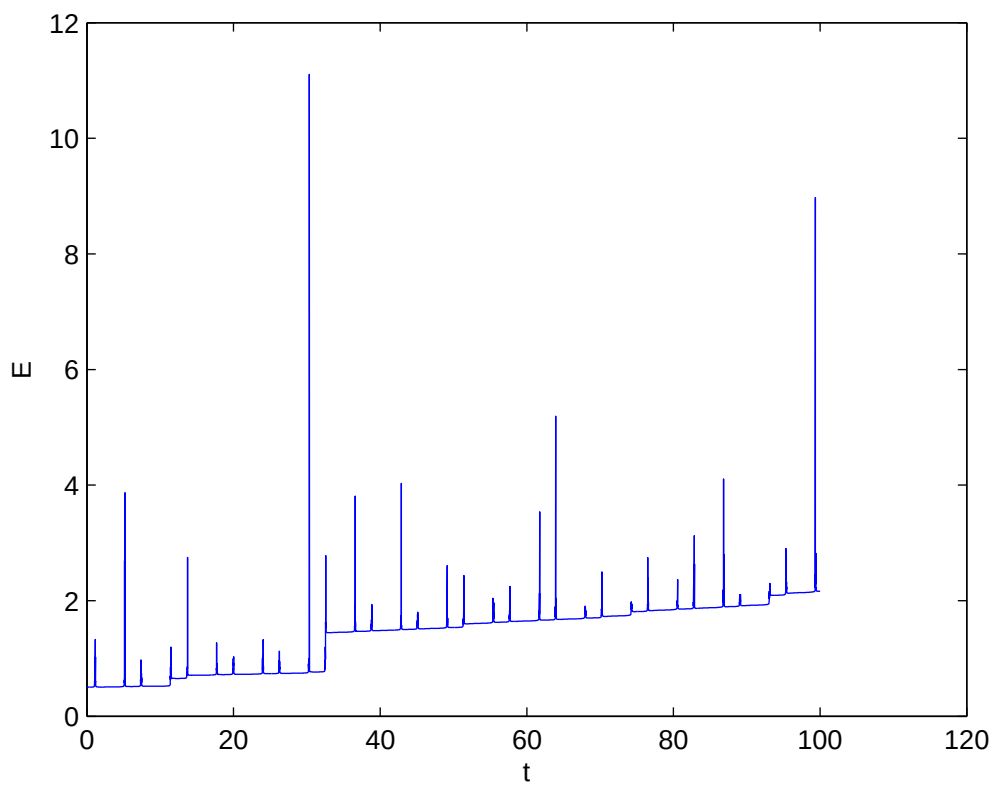


Figure 2.15: Plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$.

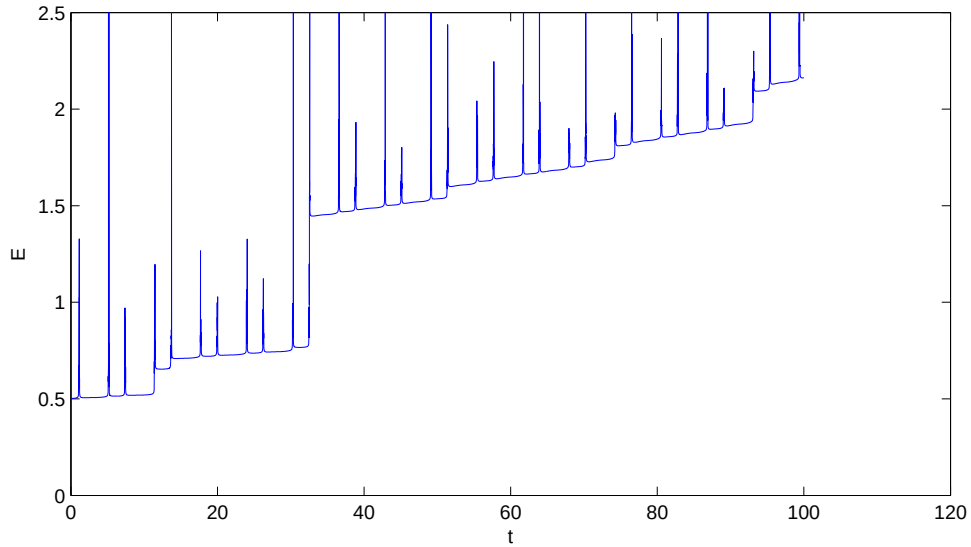


Figure 2.16: Detail of the plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.01$, with clouds of control particles centred around $x = -350$ and $x = 350$.

2.7 Summary

We now have five test problems to consider, two of which have associated exact closed-form solutions, and we have produced test statistics for all of them. We can go on to consider how we would apply the algorithm to these test problems in Chapter 3, and then look at some preliminary results for the implementation of the algorithm in Chapter 4.

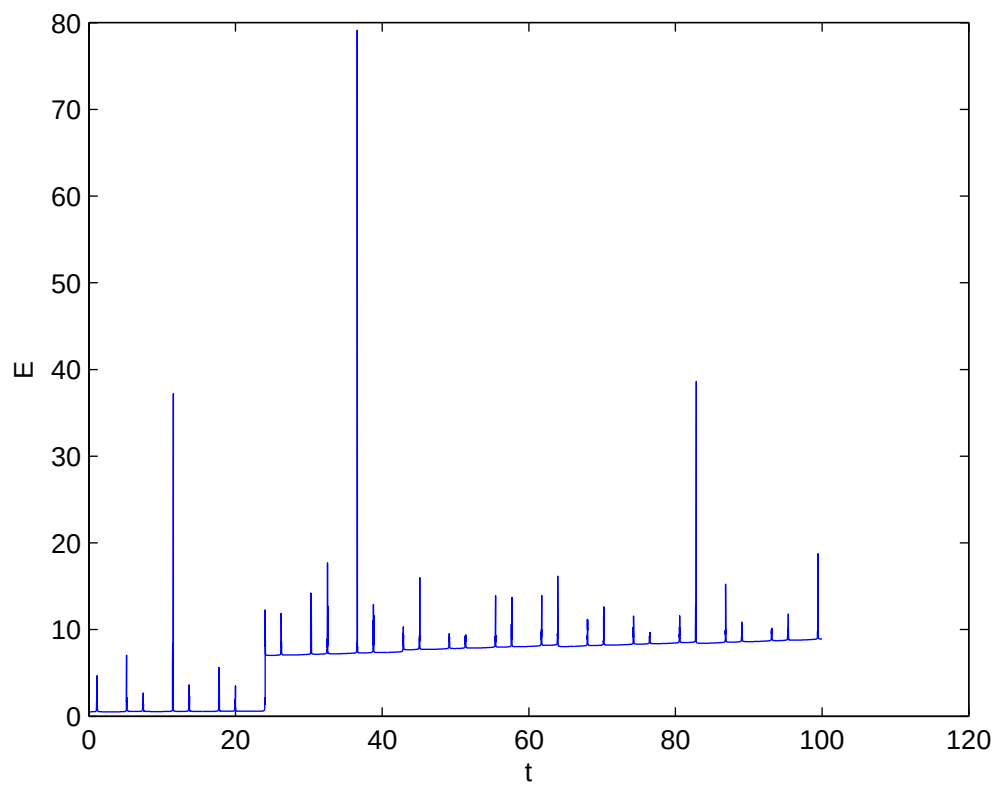


Figure 2.17: Plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$.

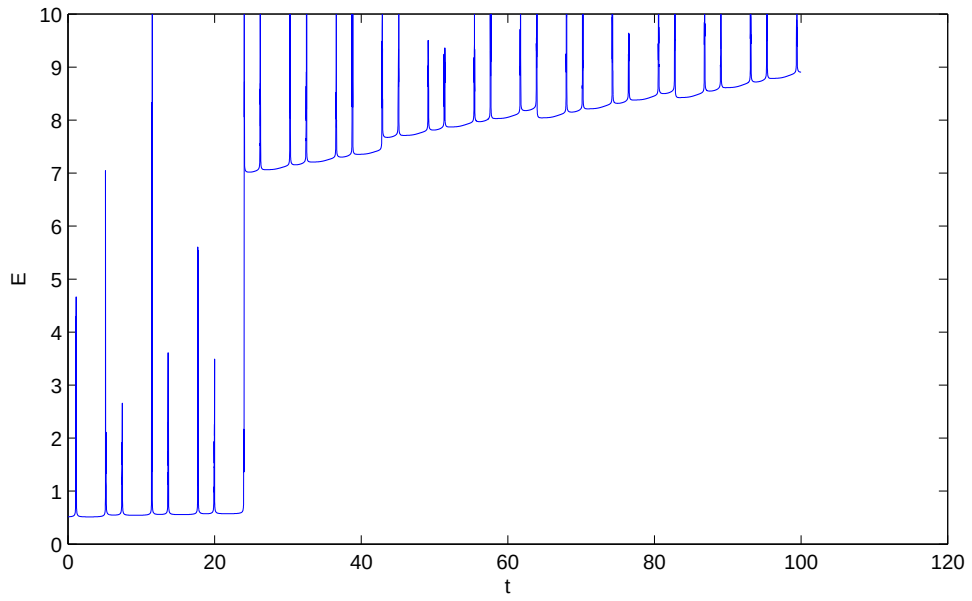


Figure 2.18: Detail of the plot of energy against time for the Interacting Harmonic Oscillators, with $G=0.1$, with clouds of control particles centred around $x = -500$ and $x = 500$.

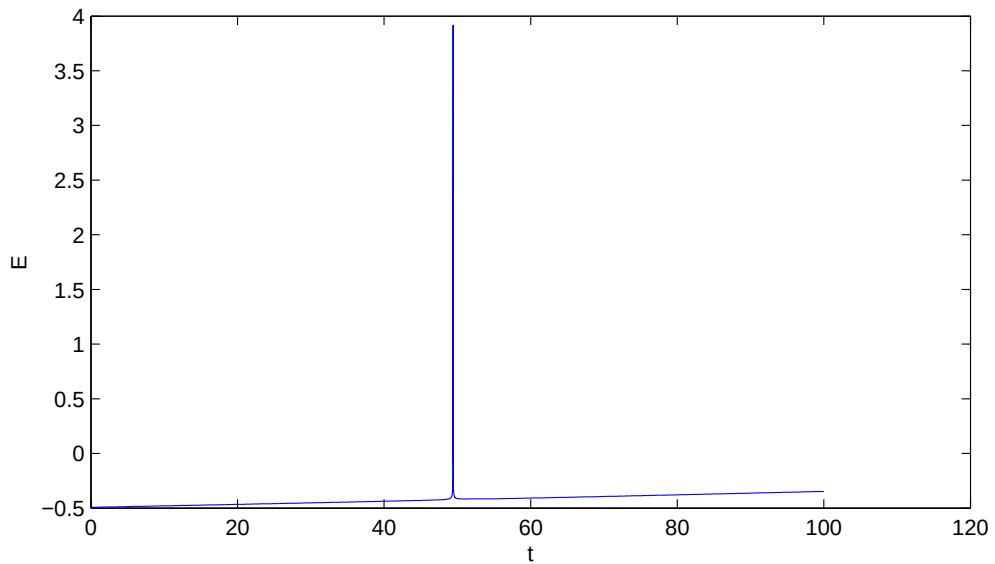


Figure 2.19: Plot of energy against time for the Interacting Simple Pendula, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$.

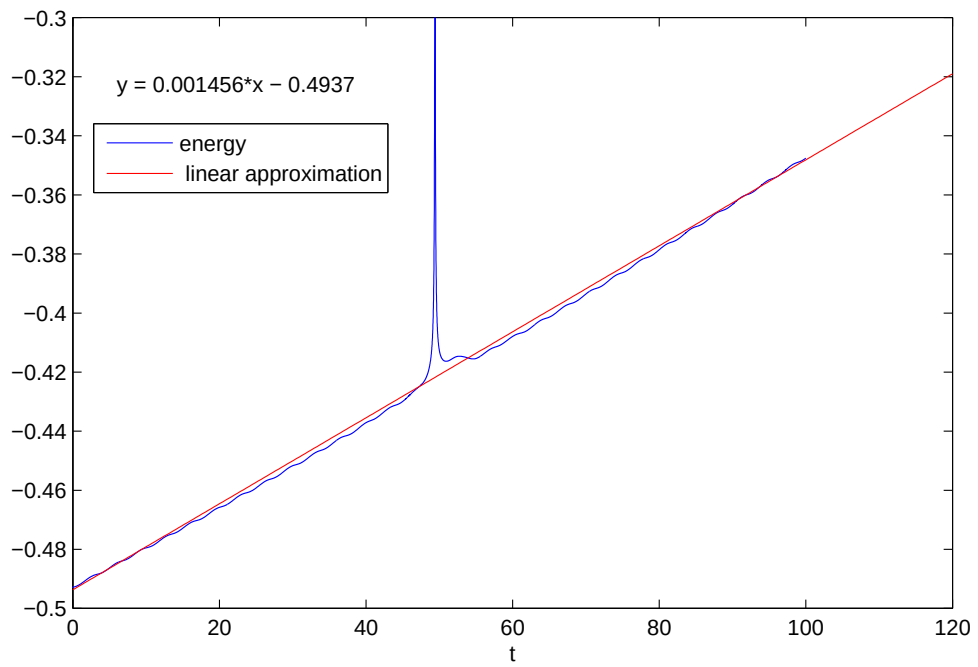


Figure 2.20: Detail of the plot of energy against time for the Interacting Simple Pendula, with $G=0.01$, with clouds of control particles centred around $x = -50$ and $x = 50$. We fit a linear polynomial $E = 0.001456t - 0.4937$ to the energy.

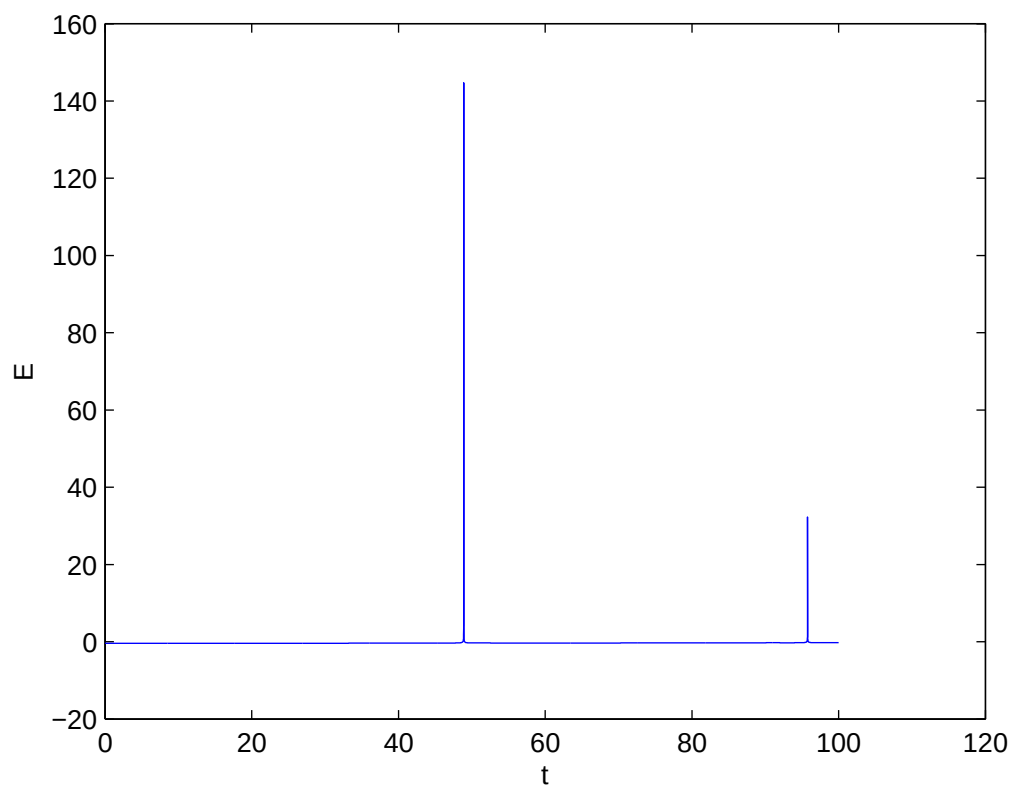


Figure 2.21: Plot of energy against time for the Interacting Simple Pendula, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$.

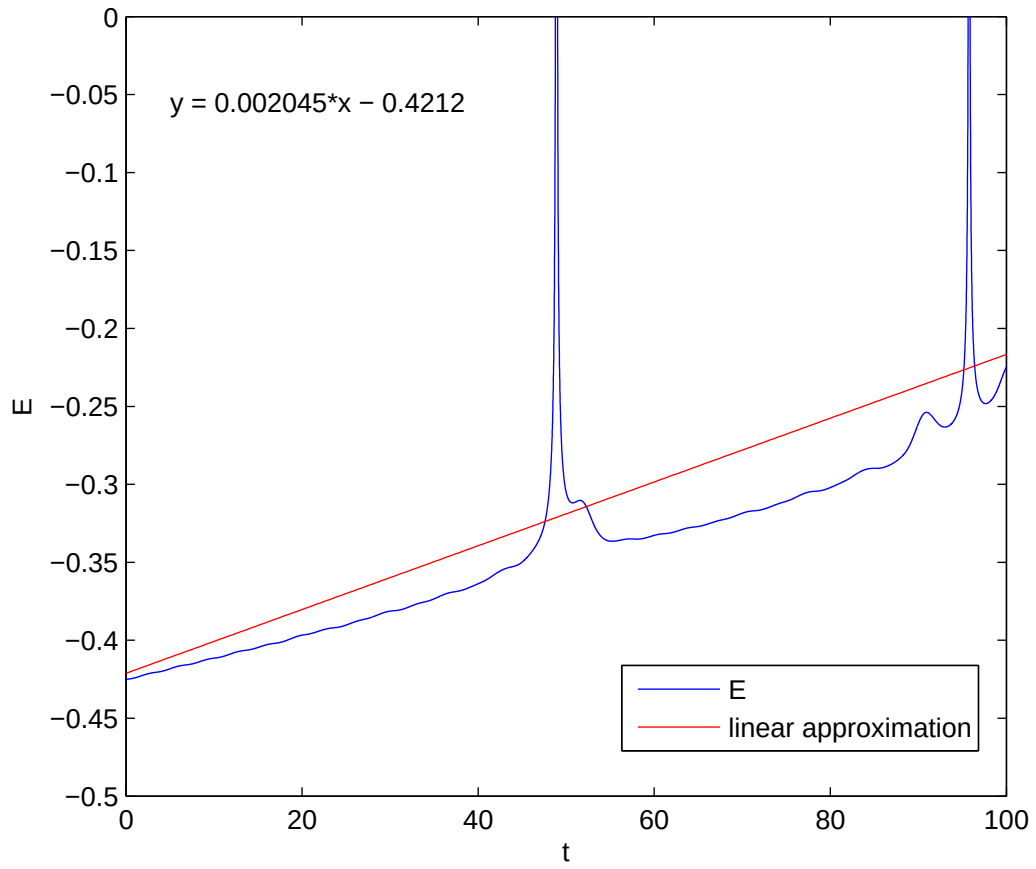


Figure 2.22: Detail of the plot of energy against time for the Interacting Simple Pendula, with $G=0.1$, with clouds of control particles centred around $x = -50$ and $x = 50$. We fit a linear polynomial $E = 0.002045t - 0.4214$ to the energy.

Chapter 3

Implementing the algorithm

3.1 Description of the algorithm

We are attempting to solve an ODE of the form

$$dy_t = \sum_{i=1}^N F_i(t, y_t) dt, \quad (3.1)$$

where N is large, and $y_t \in \mathbb{R}^2$.

We assume that all the vector fields F_i are at least n times continuously differentiable, for some value of n . This enables us to consider the two-dimensional Taylor expansion for F_i in y about a point b , such that

$$F_i(t, y_t)|_b = \sum_{|\mathbf{k}|=0}^{\infty} \frac{1}{k_1!k_2!} \left[\left(\frac{\partial^{k_1}}{\partial x_1^{k_1}} \right) \left(\frac{\partial^{k_2}}{\partial x_2^{k_2}} \right) F_i(t, \cdot) \right]_b (y_1 - b_1)^{k_1} (y_2 - b_2)^{k_2}. \quad (3.2)$$

We truncate this after n terms, where

$$F_i^n(t, y_t)|_b = F_i(t, y_t)|_b - R_i^n(t, b). \quad (3.3)$$

In this case, there exists a radius r_b such that

$$\left\| \sum_{i=1}^n (F_i(t, y_t)|_b - F_i^n(t, y_t)|_b) \right\|_{B(b, r_b)} < \epsilon, \quad (3.4)$$

for some appropriate Lipschitz norm $\|\cdot\|_{B(b, r_b)}$ on the ball $B(b, r_b)$.

Next, we need to choose a set of functions P_j such that they make up a partition of unity, and so that there exist points $\{a_j\}_{j=1}^J$ and associated radii r_j , such that $\bigcup_j B(a_j, r_j)$ covers any plausible trajectory $y(t)$, and P_j is compactly supported on the ball $B(a_j, r_j)$. Let

$$M_p = \sup_y |\{j : P_j(y) > 0\}|, \quad (3.5)$$

i.e. M_p is the maximum number of partition of unity functions that are non-zero at the point y . Hence

$$\begin{aligned} dy_t &= \sum_{i=1}^N \sum_{j=1}^J (F_i^n(t, y_t)|_{a_j} + R_j^n(t, a_j)) P_j(y_t) dt, \\ &= \left(\sum_{i=1}^N \sum_{j=1}^J (F_i^n(t, y_t)|_{a_j}) P_j(y_t) dt \right) + Q(y_t, t) dt, \end{aligned} \quad (3.6)$$

where

$$\|Q\| \leq \epsilon M_p C \quad (3.7)$$

for some constant C and a suitable global Lipschitz norm. We can now use results from rough path theory to control the error; if two driving noises are close in the Lipschitz norm, their effects will be similar (if they are both p -rough paths). The rougher the path, the more derivatives are needed to be taken into account when calculating the Lipschitz norm.

Hence, (3.1) can be approximated by the system

$$dy_t = \sum_{i=1}^N \sum_{j=1}^J \sum_{|\mathbf{k}|=0}^n \frac{1}{\mathbf{k}!} \left(\frac{\partial^{\mathbf{k}}}{\partial \mathbf{x}^{\mathbf{k}}} \right) F_i(t, \cdot)|_{\mathbf{a}_j} ((\mathbf{y} - \mathbf{a}_j)^{\mathbf{k}} P_j(y)) dt. \quad (3.8)$$

This enables us to re-express (3.1) in the form

$$dy_t = \sum_{j=1}^J \sum_{|\mathbf{k}|=0}^n G_{j,\mathbf{k}}(y) dX_t^{j,\mathbf{k}}, \quad (3.9)$$

where

$$\begin{aligned} dX_t^{j,\mathbf{k}} &= \left\{ \frac{1}{\mathbf{k}!} \left(\frac{\partial^{\mathbf{k}}}{\partial \mathbf{x}^{\mathbf{k}}} \right) \sum_{i=1}^N F_i(t, \cdot)|_{\mathbf{a}_j} dt \right\}_{j=1}^J, \\ &= c_{j,\mathbf{k}}(t) dt \end{aligned} \quad (3.10)$$

which is a vector of two-dimensional time-varying scalar coefficients $c_{j,\mathbf{k}}(t)$ indexed by the vector $\mathbf{k}$ and the scalar j , and

$$G_{j,\mathbf{k}}(y) = (\mathbf{y} - \mathbf{a}_j)^{\mathbf{k}} P_j(y). \quad (3.11)$$

The function G is a map into a sequence of multi-dimensional vector fields, also indexed by j and $\mathbf{k}$. We have thus produced a finite dimensional approximation for (3.1), which is separable into a space component and a time component.

We now consider how to solve an equation of the form

$$dy_t = G(y)dX_t, \quad (3.12)$$

where $G(y)$ is a continuous vector field and X_t is a path of bounded variation. Let $\mathbf{e}_1, \dots, \mathbf{e}_m$ be the basis vectors for the vector space in which X lies. The basis for the Lie algebra is given by

$$\begin{aligned} \{\mathbf{e}_1, \dots, \mathbf{e}_m, [\mathbf{e}_{i_1}, \mathbf{e}_{i_2}]_{i_1 \leq i_2}, [e_{i_1}, [e_{i_2}, e_{i_3}]], \dots\} &= H \\ &= \{h_1, h_2, \dots\}. \end{aligned} \quad (3.13)$$

We refer to the set H as the *Hall basis* for the Lie algebra.

Having computed a Hall basis H for the path X , we can then find its log-signatures, which we can express as

$$L(X_{0,T}) = \sum_h \alpha_h, \quad (3.14)$$

for a series of coefficients α_h .

We can now extend G into a unique Lie map G^* such that

$$G^*[v_1, [v_2, v_3]] = [G(v_1), [G(v_2), G(v_3)]], \quad (3.15)$$

as described in [15]. We now consider the flow π_t corresponding to the solution y_t . The logarithm of this flow is given by $G^*(L(X_{0,T}))$. We can now contract the log-signature of X with the Lie brackets of the vector fields $G(y)$ in order to find an approximation to the global flow:

$$\log \pi = \sum_h \alpha_h \mathcal{G}_h(y), \quad (3.16)$$

where

$$\begin{aligned} \mathcal{G}[h_1, h_2] &= [\mathcal{G}_{h_1}, \mathcal{G}_{h_2}], \\ \mathcal{G}[\mathbf{e}_i] &= G_{j,\mathbf{k}}, \end{aligned} \quad (3.17)$$

if $i \leq m$, where m is the total number of indices j and $\mathbf{k}$. This is true no matter what sequence of vectors h we use; since G_h are all vector fields, they can be used in exactly the same way as the basis vectors h .

Since we do not know π_t at the end of the time step, we shall instead solve the ODE given by

$$d\mathbf{y} = \sum_h \alpha_h \mathcal{G}_h(y) dt, \quad (3.18)$$

which we can solve with a Runge-Kutta type algorithm.

We shall now discuss in more detail how we can implement each step of the algorithm computationally.

3.2 Results used in applying the algorithm

3.2.1 Calculating coefficients for multivariate truncated Taylor series

We have adapted Rich Neidinger's code which implements his algorithm for computing these coefficients quickly and efficiently (see [18, 19]).

Instead of directly computing the coefficients for a continuous multivariate function $f(\mathbf{x})$ evaluated at a point $\mathbf{a}$, this method uses univariate directional functions of the form $u_{\mathbf{k}}(s) = f(s\mathbf{e}_{\mathbf{k}})$ for some parameter s and a basis vector $\mathbf{e}_{\mathbf{k}}$, computes the univariate lists of coefficients, and then "interpolates" the multivariate array of coefficients. With suitable choices of the directions, we can use the univariate coefficient lists $\{\{u_k^{(0)}, \dots, u_k^{(n)}\} : |\mathbf{k}| = n\}$ to construct the entire pyramid of multivariate Taylor coefficients $c_{\mathbf{k}} = \frac{1}{k_1! \dots k_d!} \left[\left(\frac{\partial}{\partial x_1} \right)^{k_1} \dots \left(\frac{\partial}{\partial x_d} \right)^{k_d} f \right](\mathbf{a})$.

The "interpolation" step (which is referred to as such for conceptual reasons and is not a process of polynomial fitting) makes use of the following relationship between $u_{\mathbf{k}}$ and $c_{\mathbf{k}}$

$$\begin{aligned} u_{\mathbf{k}}(s) &= f(s\mathbf{e}_{\mathbf{k}}), \\ &= \sum_{\mathbf{k}} c_{\mathbf{k}} (s\mathbf{e}_{\mathbf{k}})^{\mathbf{k}}, \\ &= \sum_{\mathbf{k}} c_{\mathbf{k}} (\mathbf{e}_{\mathbf{k}})^{\mathbf{k}} s^{|\mathbf{k}|}. \end{aligned} \tag{3.19}$$

Since $u_{\mathbf{k}}(s) = \sum_{i=0}^{\infty} u_{\mathbf{k}}^{(i)} s^i$, by comparing coefficients

$$u_{\mathbf{k}}^{(i)} = \sum_{|\mathbf{k}|=i} c_{\mathbf{k}} (\mathbf{e}_{\mathbf{k}})^{\mathbf{k}}. \tag{3.20}$$

By taking a number of different directions, we can therefore produce a set of simultaneous equations for the various multivariate coefficients.

Let $(\omega_i : i = 0, 1, 2, \dots)$ be a sequence of distinct real numbers, where for the sake of simplicity we assume $\omega_0 = 0$. For some $\alpha = (\alpha_1, \dots, \alpha_{d-1})$, let the direction vector $\mathbf{e}_{\alpha} = (1, \omega_{\alpha_1}, \dots, \omega_{\alpha_d})$. We also define a map from $\mathbf{k}$ (where $|\mathbf{k}| = k$) to $\beta = (k_2, \dots, k_d)$ which simply drops the first entry of $\mathbf{k}$, and an inverse map $T_k(\beta) = (k - |\beta|, \beta_1, \dots, \beta_{d-1})$. For general values of n and d , we choose directions such that $\{e_{\alpha} : |\alpha| \leq n\}$, and where for each axis $D \in \{1, 2, \dots, d-1\}$ we choose any sequence of distinct real numbers $\omega_0^{[m]}, \omega_1^{[m]}, \dots$ as an arbitrarily spaced set of grid marks labeled in arbitrary

order. This yields a "wedge" $\{\omega_{\alpha_1}^{[1]}, \dots, \omega_{\alpha_{d-1}}^{[d-1]} : |\alpha| \leq n\}$ of "node" points from which we can compute our directions $\mathbf{e}_\alpha$. Hence

$$\sum_{\beta} = c_{T_k(\beta)}(\omega_\alpha)^\beta = u_\alpha^{(k)}. \quad (3.21)$$

We then define a square matrix

$$M_k = [(\omega_\alpha)^\beta], \quad (3.22)$$

with row index α and columns indexed by β . Given that M_k is invertible, the "interpolation" step simply becomes

$$(c_{T_k(\beta)}) = M_k^{-1}(u_\alpha^{(k)}). \quad (3.23)$$

3.2.2 Constructing the partition of unity

We assume the manifold we are working on is at least locally embedded in $\mathbb{R}^n$, in order for a metric function to be sufficiently well-defined for an idea of a ball with a given radius to be meaningful. The partition of unity we will be using is one for which each $P_j = 1$ at the triangulation point a_j , and then falls off smoothly such that it becomes zero outside the patch of radius r_j centred about the triangulation point a_j .

We need to work out a way to tile the local area of the manifold with balls of the form $B(a_j, r_j)$ for $j \in J$, where the Taylor series truncated after n terms about the local triangulation point a_j has a radius of accuracy r_j . We shall adopt a heuristic method, taking advantage of the properties of a regular tessellation of the manifold. Specifically, we shall construct a regular hypercubic honeycomb in order to make sure that the local neighbourhood of the manifold is covered with no overlaps and no gaps. We begin by constructing a hypercube whose inscribed hypersphere is of radius $r_1/3$ (whose side is therefore $2r_1/3$) and whose centre is at the first triangulation point a_1 . We also construct a second hypercube whose inscribed hypersphere is of radius r_1 (and therefore whose side is of length $2r_1$), and whose centre is also at a_1 . We tile the local area using hypercubes of side $r_j/3$ whose inscribed hypersphere is of radius $2r_j/3$, surrounding the original small hypercube, and then construct around them larger hypercubes of side $2r_j$. The triangulation points are then the centres of these larger hypercubes. We shall address the question of how we choose the radii r_j and how we position the triangulation points a_j later on in this chapter.

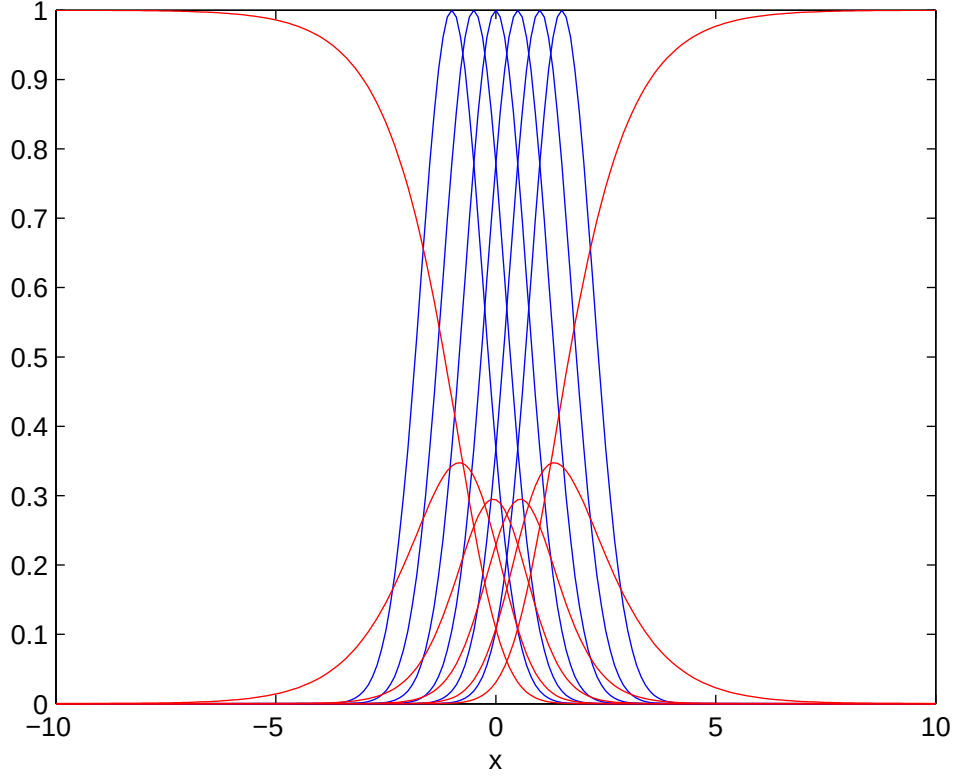


Figure 3.1: Functions P_j for a 1-dimensional partition of unity, with triangulation points at $x = -1 + 0.5d$ for $d = 1, \dots, 6$, with radii $r = 1/\sqrt{2}$. We superimpose the bump functions (blue) on the partition of unity (red). Note that the partition of unity functions approach a constant value of 1 away from the triangulation points.

We then construct a partition of unity whose functions P_j are such that

$$P_j(\mathbf{y}) = \frac{w_j(\mathbf{y})}{\sum_k w_k(\mathbf{y})} \forall y \in B(A_j, a_n), \quad (3.24)$$

and $P_j(y) = 0 \forall y \notin B(A_j, a_n)$, where $w_j(\mathbf{y})$ are decaying functions of $\mathbf{y}$ from the triangulation points A_j (see [1]). In this case, the $w_j(\mathbf{y})$ are Gaussian functions with means at our specified triangulation points, of the form

$$w_j(\mathbf{y}) = e^{-(\mathbf{y}-A_j)^2/(2r_j^2)}. \quad (3.25)$$

3.3 Lie brackets

We will need to consider how this approximation by truncated Taylor series affects the accuracy of computations involving the Lie brackets and hence the Lie derivatives

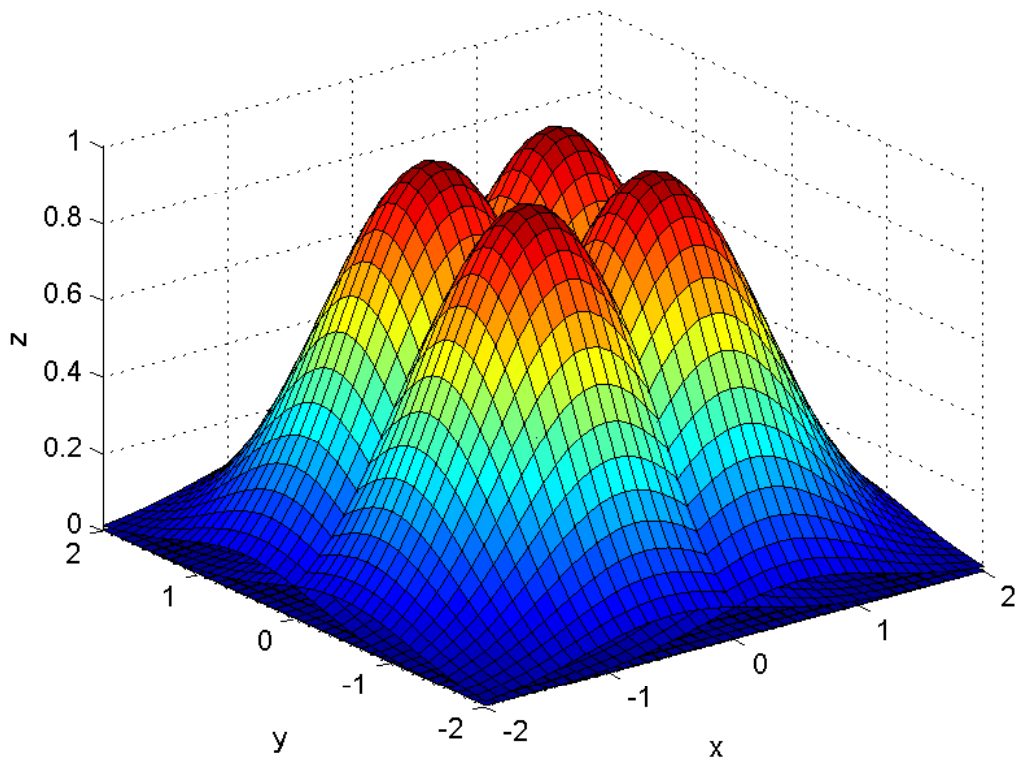


Figure 3.2: The two-dimensional Gaussian bump functions w_j with centres $(-0.5, 0.5)$, $(-0.5, -0.5)$, $(0.5, -0.5)$ and $(0.5, 0.5)$, each with an associated radius of $1/\sqrt{2}$.

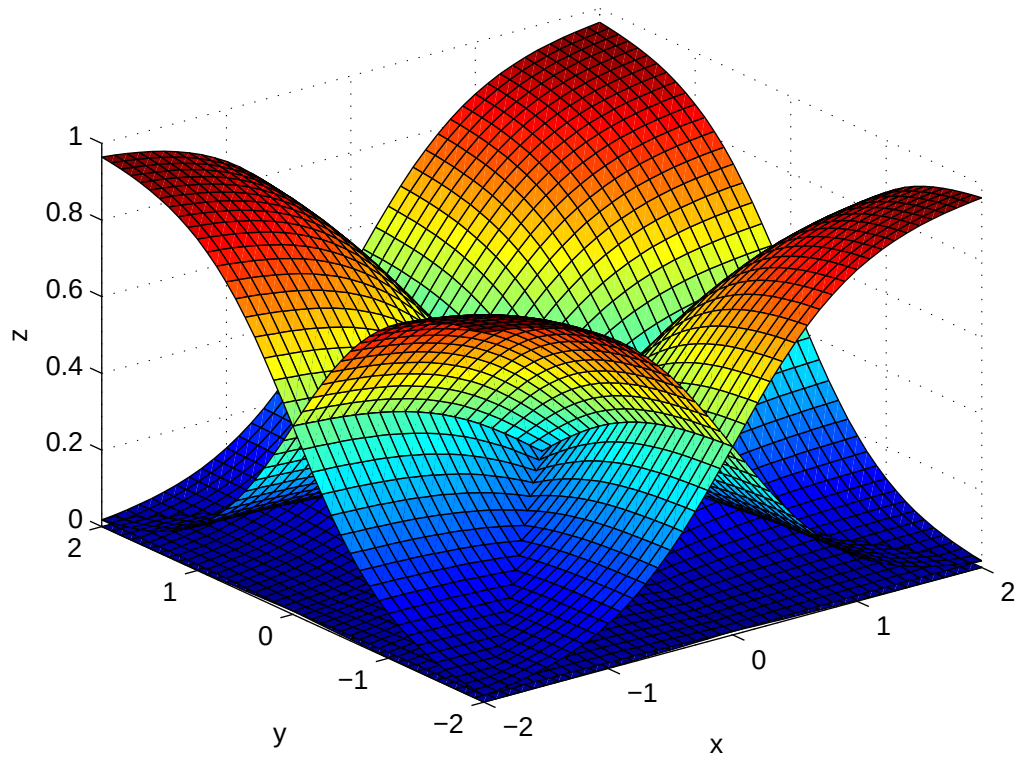


Figure 3.3: The partition of unity functions P_j generated from the bump functions shown in Figure 3.2.

of the approximated vector fields. The Lie bracket of two vector fields F^i and F^j , considered as operators on a trajectory $y(t)$, will be given by

$$\begin{aligned} [\mathbf{F}^i, \mathbf{F}^j] &= F^i F^j - F^j F^i, \\ &= L_{\mathbf{F}^i} F^j, \end{aligned} \quad (3.26)$$

where $L_{\mathbf{F}^i}$ is the Lie derivative with respect to the vector field $\mathbf{F}^i$. Given that $\mathbf{F}^i = \mathbf{F}_n^i + \mathbf{R}_{n+1}^i$, and similarly for $\mathbf{F}^j$,

$$\begin{aligned} [\mathbf{F}^i, \mathbf{F}^j] &= [\mathbf{F}_n^i + \mathbf{R}_{n+1}^i, \mathbf{F}_n^j + \mathbf{R}_{n+1}^j], \\ &= [\mathbf{F}_n^i, \mathbf{F}_n^j] + [\mathbf{R}_{n+1}^i, \mathbf{R}_{n+1}^j] + [\mathbf{F}_n^i, \mathbf{R}_{n+1}^j] + [\mathbf{R}_{n+1}^i, \mathbf{F}_n^j]. \end{aligned} \quad (3.27)$$

The magnitude of the correction term given by $[\mathbf{R}_{n+1}^i, \mathbf{R}_{n+1}^j] + [\mathbf{F}_n^i, \mathbf{R}_{n+1}^j] + [\mathbf{R}_{n+1}^i, \mathbf{F}_n^j]$ will affect the accuracy of the Lie brackets of the vector fields, and hence that of any calculations involving them. We therefore require the first few Lie derivatives of the remainder be near zero, as well as the remainder term being near zero.

We will need to compute the Lie brackets of the component vector fields $V^i(x)$ in order to compute the trajectory. Since Lie brackets have the properties that $[X_1 + X_2, Y] = [X_1, Y] + [X_2, Y]$ and $[cX, Y] = c[X, Y]$ for constant c , then we can precompute the Lie brackets for the products of the monomial terms of the Taylor expansion and the bump functions. This means that we only need to calculate the values of $[(x - x_j)^{i_1} P_j(x), (x - x_k)^{i_2} P_k(x)]$ for $1 \leq i_1 \leq i_2 \leq n$, $1 \leq j \leq k \leq J$. We can then find all possible Lie brackets of component vector fields from linear combinations of these precomputed Lie brackets with the appropriate coefficients, which we can find from appropriate products of the Taylor coefficients.

We use Atakan Varol's symbolic Matlab code for computing the Lie brackets of symbolically given vector fields (see [28]), which computes the iterated Lie brackets of two symbolic vector fields $f(\mathbf{y})$ and $g(\mathbf{y})$, where we take the 0^{th} Lie bracket to be $g(\mathbf{y})$, and then iteratively compute the further Lie brackets using the Jacobian matrix J , where

$$[f, g]^{l+1} = J([f, g]^l) f - J(f)[f, g]^l. \quad (3.28)$$

As this is a symbolic computation in Matlab, it is accurate but time-consuming.

3.3.1 The Hall basis and the log-signatures

Determining which combinations of Lie brackets of basis vectors will produce independent basis vectors, so that we can obtain a true basis and not just a spanning set, can be difficult and time-consuming. The Hall basis provides a way of performing a

systematic search by iteratively generating new basis elements from combinations of the first m basis vectors. The construction of the basis follows a width-first search, where the depth d of a Lie product is defined recursively: we begin by letting $d(h_i) = 1$ for $i = 1, \dots, m$, then let $d([h_{i_1}, h_{i_2}]) = d(h_{i_1}) + d(h_{i_2})$.

In addition to the width-first search, we also need to prune the basis to eliminate redundancy. We require that the Hall basis $H = \{h_1, h_2, \dots\}$ satisfies the following conditions:

- The first m elements of H are the vector fields $h_1, \dots, h_m$ which provide a basis for the vector space that the system's trajectory lies in.
- If $d(h_{i_1}) < d(h_{i_2})$, then $i_1 < i_2$.
- Each bracket $[h_{i_1}, h_{i_2}] \in H$ if and only if $h_{i_1}, h_{i_2} \in H$, and either $h_{i_2} \in H$ or $h_{i_2} = [h_{i_3}, h_{i_4}]$ for some $h_{i_3}, h_{i_4} \in H$ such that $i_3 \leq i_1$.

The algorithm for computing the log-signatures is relatively simple. We first compute the increments $I_{j,\mathbf{k}}$ of the path X_t over the m points in the vector $\{c_{j,k}(t)\}_{j=1, \mathbf{k} \geq 0}^{J, |\mathbf{k}|=n}$, where $I_i = c_{i+1} - c_i$ for $i = 1, \dots, m-1$ and each i maps to a given combination of j and $\mathbf{k}$. We then compute

$$S(X_{0T}) = \prod_h e^{I_h h_i}. \quad (3.29)$$

The log-signature can then be computed, as $L(X_{0T}) = \log(S(X_{0T}))$.

We use the code produced by the Computational Rough Paths project (see [4]) to calculate the Hall basis, contract the vector fields with the signatures, and then solve one step forward for the resulting flow.

3.3.2 Error control

3.3.2.1 Local error control

If the maximum tolerable local error is ϵ_L , that means that we need the total error in the sum of the local vector fields to be less than ϵ_L , i.e.

$$Q(y_t, t) \leq \epsilon_L. \quad (3.30)$$

We need each partition of unity function P_j to be close to zero outside the ball $B(a_j, r_j)$. Problems will arise if the partition of unity functions remain significantly greater than zero whilst the error in the Taylor series approximation is also not negligibly small (see Figure 3.4). We need $P_j R^{n+1}(y, t)|_{a_j} \leq \epsilon$ even outside the ball $B(a_j, r_j)$.

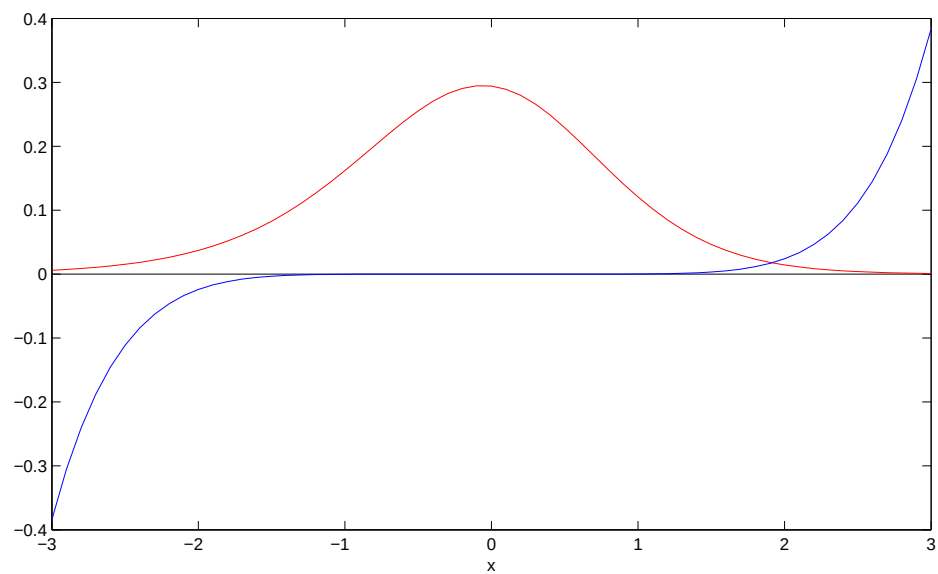


Figure 3.4: Consider one of the functions P_j from the 1-dimensional partition of unity shown in Figure 3.1 (shown in red). We calculate the Taylor series for $f = -\sin y$ up to 5th order about the triangulation point, and then calculate the error (shown in blue). We observe that the error in the Taylor series becomes significantly non-zero before the partition of unity function fades away completely. This illustrates the importance of careful choice of radii for the partition of unity.

3.3.2.2 Placing triangulation points using a quadtree

We now need to consider how to place the triangulation points for the partition of unity in order to obtain a suitable level of accuracy. We begin by considering a region of space that we expect to cover the trajectories of the system, and apply *quadtree decomposition* to this area. A quadtree is a tree data structure in which every node has exactly four children. It is often used to partition a two-dimensional region into sub-regions by recursively dividing it into quadrants.

In this case, we pick the square of side $2w_0$ that we expect to cover the trajectories, and refer to its centre as a_0 and the radius of the partition of unity patch around it as r_0 , where $r_0 = 3\sqrt{2}w_0$. We refer to the region specified as the inscribed square of the circle with centre a_0 and radius r_0 as the root of the quadtree. We then divide this region into a fine grid, then calculate the remainder function $R^{n+1}(y, t)|_{a'_j}$ at the centre a'_j of each grid cell. If $R^{n+1}(y, t)|_{a'_j} < \epsilon \forall j$, then we take a_0 and r_0 as the triangulation point and patch radius for the partition of unity. If not, we split the region into four quadrants, each one a square of side $w_0/2$, with centres $a_1, \dots, a_4$ and associated radii $r_j = \frac{3\sqrt{2}}{2}w_0$, and repeat the procedure for each quadrant.

After ν steps of the quadtree algorithm, we have $J = 4^\nu$ triangulation points a_j , each the centre of a square of side $w_j = 2w_0/4^\nu$ and a partition of unity patch of radius $r_j = 3\sqrt{2}w_j$.

This procedure allows us to select a partition of unity that will yield the desired level of accuracy for a specified number of terms of the Taylor series. We shall truncate the Taylor series after four terms, for the sake of speed and simplicity.

3.4 Examples

3.4.1 The harmonic oscillator

As mentioned in Chapter 2, the harmonic oscillator is given by the equations:

$$\begin{aligned}\dot{y} &= p, \\ \dot{p} &= -y.\end{aligned}\tag{3.31}$$

The analytic solution of these equations is

$$\begin{aligned}y(t) &= A \sin t + B \cos t, \\ p(t) &= A \cos t - B \sin t,\end{aligned}\tag{3.32}$$

where our initial data determines the values of the constant coefficients.

We are looking at a simple two-dimensional problem driven by a single vector field here. In this case, we represent the system as

$$d\mathbf{y} = F_1(\mathbf{y}, t)dt \quad (3.33)$$

where $\mathbf{y} = (y, p)$ and

$$F_1(\mathbf{y}, t) = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \mathbf{y}. \quad (3.34)$$

In this case, since $F_1(y, t)$ is linear, we do not need to expand it as a Taylor series.

We now need to construct a partition of unity starting around the initial value $y_0 = 0, p_0 = 1$. Since we know from Chapter 2 the expected amplitude of the motion, we choose a root region with centre $(0, 1)$, width 2 and radius 6. Since we are not using a Taylor series approximation, we shall divide this region into four quadrants, thus yielding a partition of unity with triangulation points at $(-1, 2)$, $(-1, 0)$, $(1, 2)$ and $(1, 0)$, each with an associated radius of $3\sqrt{2}/2$.

The partition of unity functions in this case will be given by

$$w_l(\mathbf{x}) = \frac{1}{2\pi r_l} e^{-\frac{1}{2r_l^2}((y-a_{l1})^2 + (p-a_{l2})^2)}, \quad P_j = \frac{w_j}{\sum_{l=1}^J w_l}. \quad (3.35)$$

At this point, we can calculate a global vector field V , where,

$$V(\mathbf{y}_t, t) = \sum_{i=1}^N \sum_{j=1}^J \sum_{|\mathbf{k}|=n} F_i|_{a_j} (\mathbf{y} - a_j)^{\mathbf{k}} P_j, \quad (3.36)$$

which means that in this case the first step of simplification yields the equation

$$d\mathbf{y} = \sum_{j=1}^J \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \mathbf{y} P_j dt, \quad (3.37)$$

which is effectively the equation we began with.

The series of coefficients for which we are finding the signatures is given by

$$d\mathbf{X} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \end{pmatrix} dt, \quad (3.38)$$

and the relevant vector fields in this case are simply

$$\begin{aligned} G_{j,0,1}(y) &= pP_j(y), \\ G_{j,1,0}(y) &= yP_j(y). \end{aligned} \quad (3.39)$$

The flow map equation is simply given by

$$d\mathbf{y} = \sum_{j=1}^J \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \mathbf{y} P_j dt, \quad (3.40)$$

which means that in this case the full algorithm yields exactly the same results as simply solving forward for the vector field.

3.4.2 The simple pendulum

The simple pendulum's governing equations are given by

$$\begin{aligned} \dot{y} &= p, \\ \dot{p} &= -\sin y, \end{aligned} \quad (3.41)$$

where $y_0 = 0$ and $p_0 = 1$.

This system is nonlinear, and therefore requires Taylor series expansion; we shall take terms up to fourth order in the interest of simplicity and speed. Hence, we have the expansions evaluated around a point $\mathbf{b} = (b_1, b_2)$

$$\begin{aligned} \dot{y} &\approx p, \\ \dot{p} &\approx -\sin b_1 - \cos b_1(y - b_1) + \sin b_1 \frac{(y - b_1)^2}{2} + \cos b_1 \frac{(y - b_1)^3}{6}. \end{aligned} \quad (3.42)$$

Since we are using a Taylor expansion, we shall need to run the quadtree decomposition code in order to find the triangulation points and associated radii for the partition of unity. We shall use a root region with centre $(0, 1)$, width 2 and associated radius 6, and require a level of accuracy $\epsilon = 10^{-5}$ (in order to obtain second order accurate solutions for a time step $\delta t = 0.01$). The

The vector field we establish here is going to be of the form

$$d\mathbf{y} = \sum_{j=1}^J \begin{pmatrix} p \\ -(y - a_j) + \frac{(y - a_j)^3}{6} \end{pmatrix} P_j dt, \quad (3.43)$$

which we can solve using a Runge-Kutta method in order to obtain an estimate of the solution.

The coefficients which we need to find the signature of will be given by

$$\begin{aligned} dX_1 &= 1dt, \\ dX_2 &= \left\{ -\sin b_1^j, -\cos b_1^j, \frac{\sin b_1^j}{2}, \frac{\cos b_1^j}{6} \right\}_{j=1}^J dt \end{aligned} \quad (3.44)$$

The vector fields in this case will be given by

$$G_{j,k_1,k_2}(y) = (y - b_{j1})^{k_1}(p - b_{j2})^{k_2}P_j(y), \quad (3.45)$$

We can now proceed to find the relevant Hall basis vectors and log-signatures, and thence construct the flow map.

3.4.3 The Airy equations

The Airy equations are given by

$$\begin{aligned} dy &= p dt, \\ dp &= -K^2 y t dt, \end{aligned} \quad (3.46)$$

where $y_0 = 1$ and $p_0 = 0$.

We write $\mathbf{y} = (y, p)$, and then we can express the system as

$$d\mathbf{y} = \begin{pmatrix} 0 & 1 \\ -K^2 t & 0 \end{pmatrix} \mathbf{y} dt, \quad (3.47)$$

where in this case we shall take $K = 1$. Again, this equation is linear in $\mathbf{y}$, and so we do not need to expand it as a Taylor series. Since we are using an exact expression for the functions, we shall once again divide the root region with its centre at $(0, 1)$ and width 4 into four quadrants, thus yielding a partition of unity with triangulation points at $(-2, 3)$, $(-2, -1)$, $(2, 3)$ and $(2, -1)$, each with an associated radius of $12\sqrt{2}$.

Again, we shall calculate a vector field V which will yield an equation which we can solve using the simplified algorithm. In this case, the first step of simplification yields the equation

$$d\mathbf{y} = \sum_{j=1}^J \begin{pmatrix} 0 & 1 \\ -K^2 t & 0 \end{pmatrix} \mathbf{a}_j P_j dt, \quad (3.48)$$

which we can solve using a Runge-Kutta method in order to obtain an estimate of the solution.

The series of coefficients for which we are finding the signatures is given by

$$d\mathbf{X} = \begin{pmatrix} 0 & 1 \\ -K^2 t & 0 \end{pmatrix} dt, \quad (3.49)$$

where we instantiate this at each given time step t . The vector fields in this case are simply

$$G_{j,\mathbf{k}}(y) = \sum_{j=1}^J (\mathbf{y} - \mathbf{a}_j) P_j(y). \quad (3.50)$$

We can therefore construct a flow map, in which case we will once again be reconstructing the original equations.

3.4.4 The Interacting Pendula systems

3.4.4.1 The Interacting Harmonic Oscillators

The equations for this system are given by

$$\begin{aligned} dy &= p dt \\ dp &= -y + \sum_{i=1}^N G \frac{x_i - y}{|x_i - y|^3} dt. \end{aligned} \quad (3.51)$$

The second equation is nonlinear, and so we need to produce a Taylor series expansion for each of the gravitational interaction terms. Around a point (b_1, b_2) , where we let $r_i = x_i - b_j$, we have

$$\begin{aligned} \Gamma_i &= G \frac{x_i - y}{|x_i - y|^3}, \\ &\approx G \left(\frac{r_i}{|r_i|^3} + \left(3 \text{sign}(r_i) \left(\frac{-r_i}{|r_i|^4} - \frac{1}{|r_i|^4} \right) \right) (y - b_1) + \dots \right) \\ &= \Gamma_i^4 \\ &= \sum_{k=0}^4 c_{b,k} y^k \end{aligned} \quad (3.52)$$

for the first four terms of the Taylor series.

Since we are using a Taylor expansion, we shall need to run the quadtree decomposition code in order to find the triangulation points and associated radii for the partition of unity. We shall use a root region with centre $(0, 1)$, width 150 and associated radius $150/\sqrt{2}$, and require a level of accuracy $\epsilon = 10^{-5}$ (in order to obtain second order accurate solutions for a time step $\delta t = 0.01$).

The vector field we establish here is going to be of the form

$$d\mathbf{y} = \sum_{j=1}^J \begin{pmatrix} p \\ -y + \sum_{i=1}^N \Gamma_i^4 \end{pmatrix} P_j dt, \quad (3.53)$$

which we can solve using a Runge-Kutta method in order to obtain an estimate of the solution.

The vector field in this case is given by

$$G_{j,k} = (\mathbf{y} - a_j)^{\mathbf{k}} P_j(y), \quad (3.54)$$

but the list of coefficients that we need to find the signature of will be rather more complicated.

3.4.4.2 The Interacting Simple Pendula

The equations for this system are given by

$$\begin{aligned} dy &= p dt \\ dp &= -\sin y + \sum_{i=1}^N G \frac{x_i - y}{|x_i - y|^3} dt. \end{aligned} \quad (3.55)$$

The second equation is nonlinear, and so we need to produce a Taylor series expansion. We have already found the expansion for each of the gravitational interaction terms above, and we also know the expansion for the unperturbed simple pendulum. We simply need to add the expansion in order to obtain

$$dp = -(y - b_1) + \frac{(y - b_1)^3}{6} + \sum_{i=1}^N \Gamma_i^4 dt. \quad (3.56)$$

The vector field approximation, and the vector fields and coefficients that we will use to find the path-based approximation, are of the same form as for the Interacting Harmonic Oscillators.

3.5 Summary

We have now described in detail the results we use to obtain the two-step simplification of the set of equations; using the Taylor series and partition of unity to simplify with regard to space, and using the log-signatures to simplify with regard to time. We have also described the error control heuristics we intend to use when computing the number of terms of the Taylor series to use and the patch size of the partition of unity.

Finally, we discussed the effects of applying these simplifications to the five test problems we discussed earlier. In the next chapter, we shall consider some preliminary results from our implementation of the algorithm.

Chapter 4

Preliminary results

4.1 Properties of the algorithm and the test problems

The algorithm can, effectively, be split up into four steps: the spatial decomposition of the global vector field into locally accurate Taylor series; the reconstruction of a global vector field using the partition of unity; and then the decomposition of the vector field in time using the log-signatures and Lie basis, and its reconstruction via the contraction of the log-signatures with the vector fields. We shall attempt to use the test problems we have already discussed to test as many of these steps as possible for accuracy and efficiency. The third and fourth steps are difficult to separate, but we can test the first and second steps separately, and then the third and fourth together.

The harmonic oscillator has the advantage of a known exact closed-form solution, which makes it a good choice for tests of accuracy. However, it is a linear problem, and so we do not need to use the Taylor series expansions, and therefore the full versions of the algorithm will, of necessity, be highly simplified. We will not be able to test the accuracy of the Taylor series expansion for this problem, which means that we also cannot test the suitability of the placement of the triangulation points for the partition of unity. The main use of this problem is as a basic test that the partition of unity works as a way of interpolating between local vector fields, and that the vector field decomposition using the signatures and Hall basis is producing the right flow map.

The Airy equations also have known exact analytical solutions, and also display two different modes of behaviour; exponential behaviour for positive t , and oscillatory behaviour for t negative. However, as they are linear problems, they will not require the Taylor series expansions. The main use of the Airy equations as a test problem

will be to test the vector field decomposition, and to see if it will produce a better approximation to the oscillatory section of the problem than the Runge-Kutta-Nyström based solution. Both the Airy equations and the harmonic oscillator have the advantage that the algorithm should exactly recreate the original linear set of ODEs, and so we can expect the algorithm's results to be as accurate as the fourth-order Runge-Kutta solver (since the map reconstruction is exact, and the least accurate step is the solving of the ODE for the flow using a Runge-Kutta method).

The simple pendulum is a slightly more advanced test problem. It also has an exact closed-form solution, which we can use to check the accuracy of the solutions. Since this problem is nonlinear, it is a good first test for the accuracy of the Taylor expansions and the choice of the placement of the triangulation points. Given that the problem is nonlinear, we can also use this as a slightly more advanced test of the vector field decomposition.

The Interacting Simple Pendula system is the first problem to test the method's robustness with a large number of vector fields. At this point, we test the Taylor series, the partition of unity, and the vector field decomposition simultaneously, as all three will be required to solve this problem. In this case, we expect the solution to be able to reproduce the slow growth in energy due to the interactions between the control and response particles. The Interacting Harmonic Oscillators system will be the most demanding test yet, as we shall require the solution to reproduce the jumps in energy caused by the close transits, as well as the oscillations in the energy, and to remain stable for large N .

There are various ways in which we could measure the accuracy of the approximations, but the most obvious way is to define a distance function d between the time series $\mathbf{y}_1(t)$ given by the algorithm, and the time series $\mathbf{y}_2(t)$ given by either the exact closed-form solution or the Runge-Kutta-Nyström based method. In this case, we simply let

$$d(t) = |\mathbf{y}_1(t) - \mathbf{y}_2(t)|, \quad (4.1)$$

where $|\cdot|$ denotes the ℓ^2 norm. We also define a second coordinate-wise distance function

$$e(t) = (y_1(t) - y_2(t), p_1(t) - p_2(t)). \quad (4.2)$$

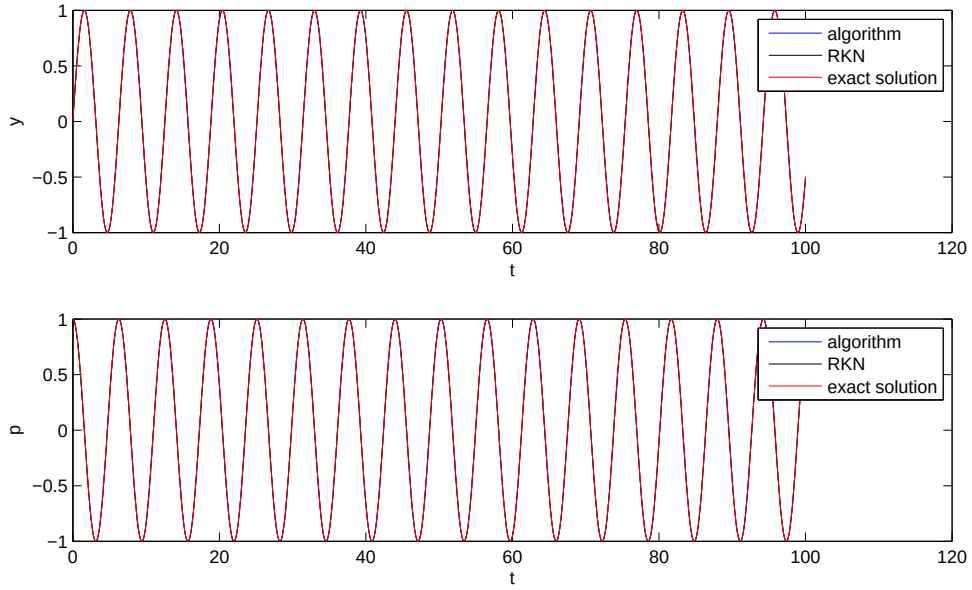


Figure 4.1: Plots of y and p against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$. We compare the results for the algorithm, for the exact solution, and for fourth-order RKN, using a time step of $\Delta t = 0.01$. We observe that there is very little difference.

4.1.1 Results

4.1.2 Simple harmonic motion

We shall be solving the equations for simple harmonic motion, as given by Equation (3.31) in the previous chapter, with the initial conditions $y_0 = 0$, $p_0 = 1$ and a grid of four triangulation points a distance 2 apart. We compare the solutions for the algorithm with the exact closed-form solution and with fourth-order RKN.

We observe from Figure 4.1 that the difference between the solutions is small, and almost imperceptible on the scale of the full amplitude. From Figures 4.2-4.3, we observe that the solution produced by the algorithm oscillates about the exact solution with amplitude 10^{-2} , with a mean error of $O(10^{-4})$. We also observe that the difference between the algorithm and the RKN-based solution is $O(10^{-8})$, but slowly growing in magnitude.

As we had expected, the error for this case was close to the error found by using an RKN method, since the algorithm effectively decomposes the vector fields then re-composes them to regain the same linear vector field, which we can then solve using a fourth order Runge-Kutta method. The Runge-Kutta solver we used for the

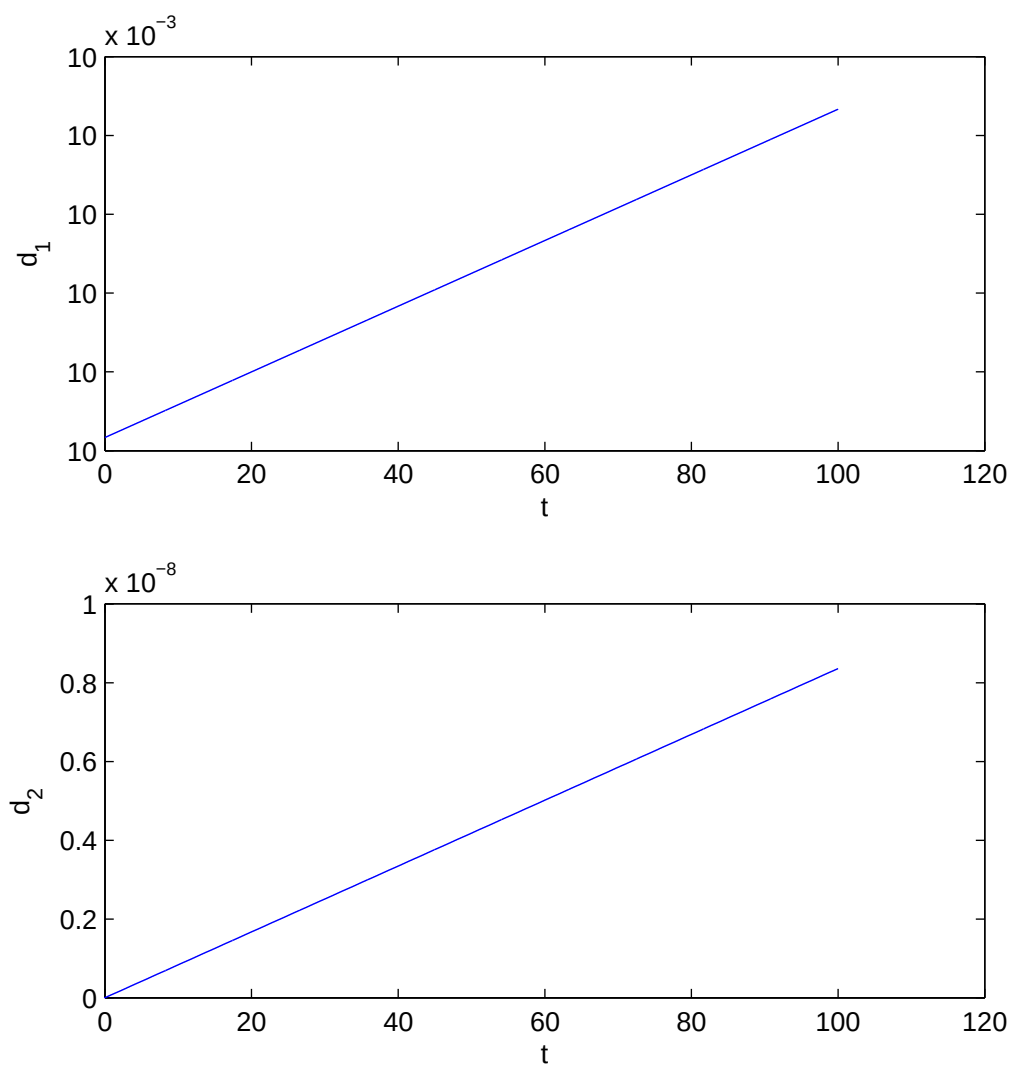


Figure 4.2: Plots of d_1 and d_2 against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$, where d_1 is the distance between the exact solution and the solution given by the algorithm, and d_2 is the distance between the exact solution and the solution given by fourth-order RKN.

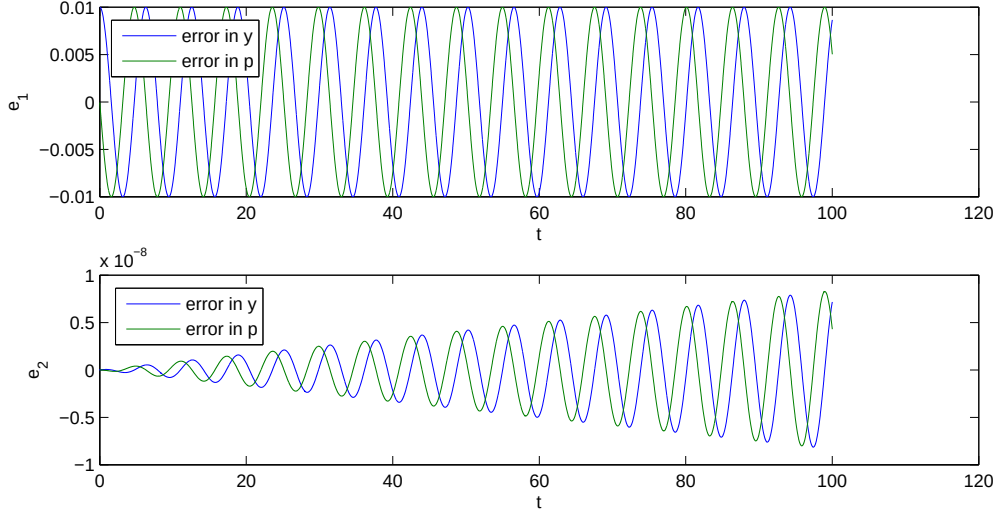


Figure 4.3: Plots of e_1 and e_2 against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$, where e_1 is the coordinate-wise distance between the exact solution and the solution given by the algorithm, and e_2 is the coordinate-wise distance between the exact solution and the solution given by fourth-order RKN.

flow map is not a symplectic one, and since the harmonic oscillator is a Hamiltonian system it is to be expected that the symplectic RKN method will yield more accurate results.

4.1.3 The Airy equations

We shall be solving the Airy equations, as given in the previous chapter by the equations (3.46), where we set the parameter $K = 1$ and with the initial conditions $y_0 = 1$, $p_0 = 0$. The analytic solution in this case will be

$$y_2 = A' Ai(-t) + B' Bi(-t). \quad (4.3)$$

We shall focus in particular on solving the Airy equation for a negative time case, from $t = 0$ to $t = -4$, and observing how well the algorithms deals with the oscillatory behaviour. Once again set, we up a grid of four evenly spaced triangulation points.

As we can see from Figure 4.4, the simplified algorithm provides a good approximation for the solutions of the Airy equations for the negative time case, as we might have expected given that we are once again working with a linear system of ODEs. From Figure 4.5, we can see that the difference between the solution given by the algorithm and the solution given by RKN is $O(10^{-16})$ and grows irregularly with

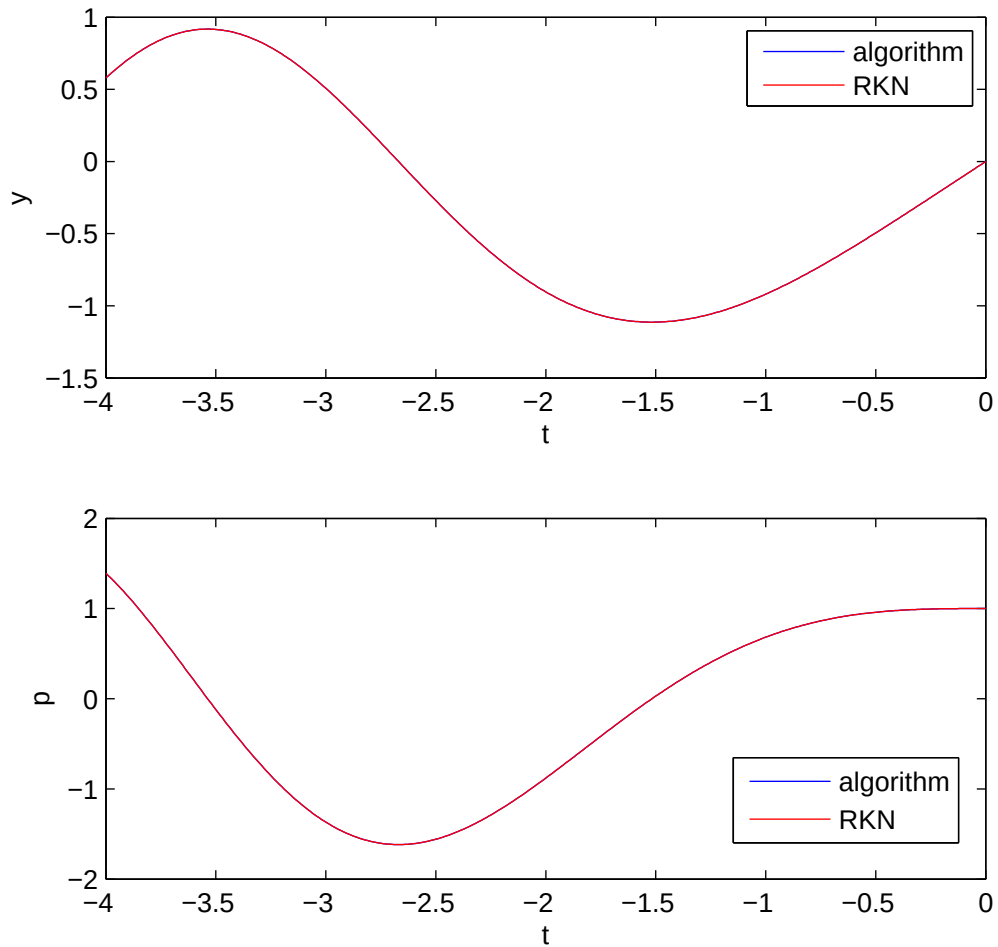


Figure 4.4: Plots of y and p against time for the Airy equations, with initial conditions $y_0 = 0$, $p_0 = 1$. We compare the results for the algorithm and for fourth-order RKN, using a time step of $\Delta t = 0.01$. We observe that there is very little difference.

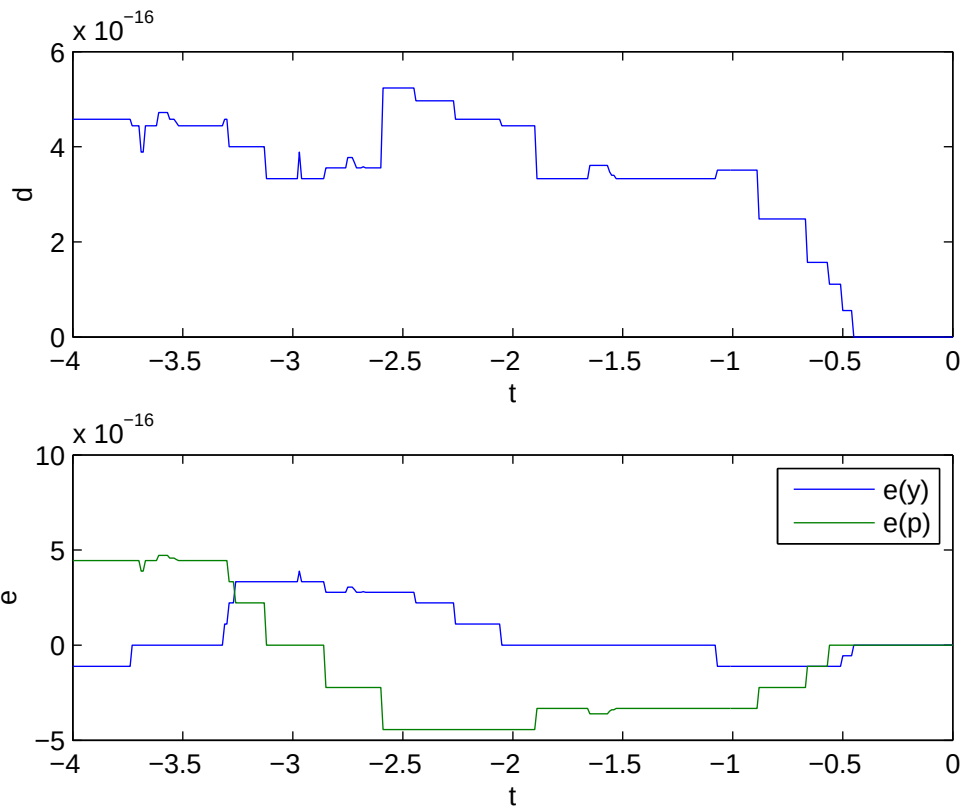


Figure 4.5: Plots of d and e against time for the harmonic oscillator, with initial conditions $y_0 = 0$, $p_0 = 1$.

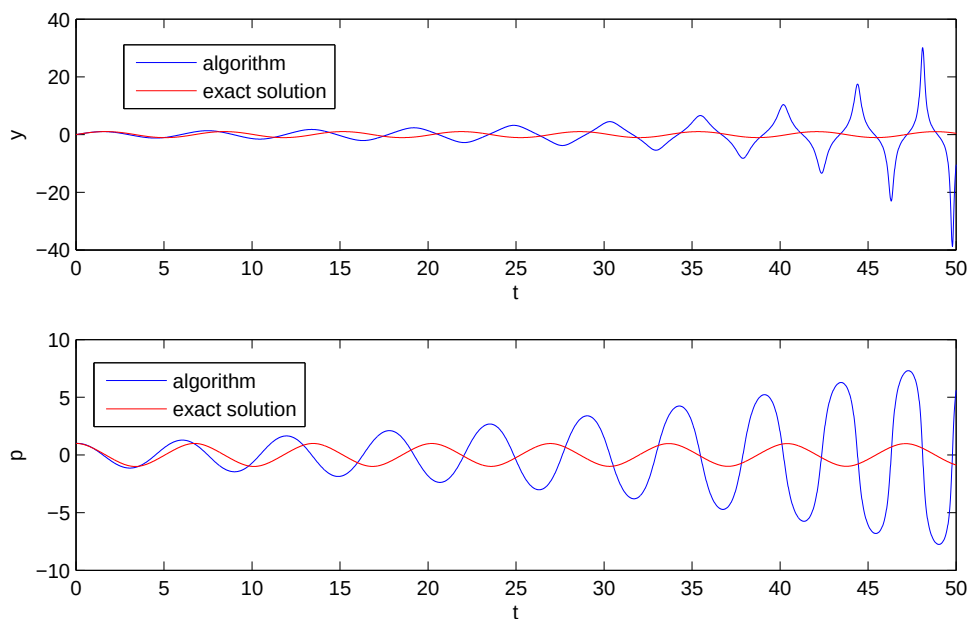


Figure 4.6: Plots of y and p against time for the simple pendulum, with initial conditions $y_0 = 0$, $p_0 = 1$. We compare the results for the algorithm to those for the exact solution. We observe a spurious growth in amplitude for the algorithmic results.

(negative) time. We have a closer similarity in this case as the Airy equations are not Hamiltonian, and so the symplectic solver does not have an advantage here.

4.1.4 The simple pendulum

We shall be solving the equations for the simple pendulum, given in the previous chapter by the equations (3.41), with the initial conditions $y_0 = 0$, $p_0 = 1$. We have constructed the partition of unity using the quadtree method described in the previous chapter, requiring that the sum of the Taylor series remainders be less than $O(10^{-5})$. This provides a grid of 256 triangulation points with spacing 0.25.

We observe from Figures 4.6 and 4.7 that the algorithm is not, in fact, providing an accurate solution for the simple pendulum; introducing the Taylor series expansion, the accuracy concerns of the partition of unity, and the accuracy concerns of taking enough terms of the signature has introduced an error term which is steadily growing with time, and also a growing phase shift error.

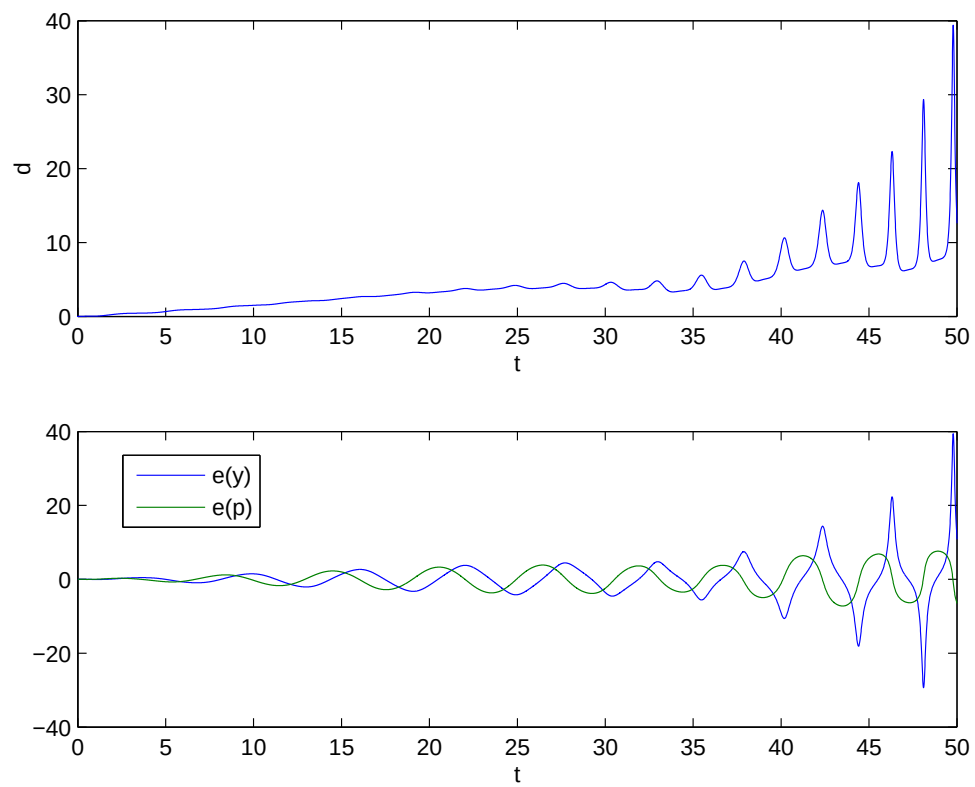


Figure 4.7: Plots of d and e against time for the simple pendulum, with initial conditions $y_0 = 0$, $p_0 = 1$.

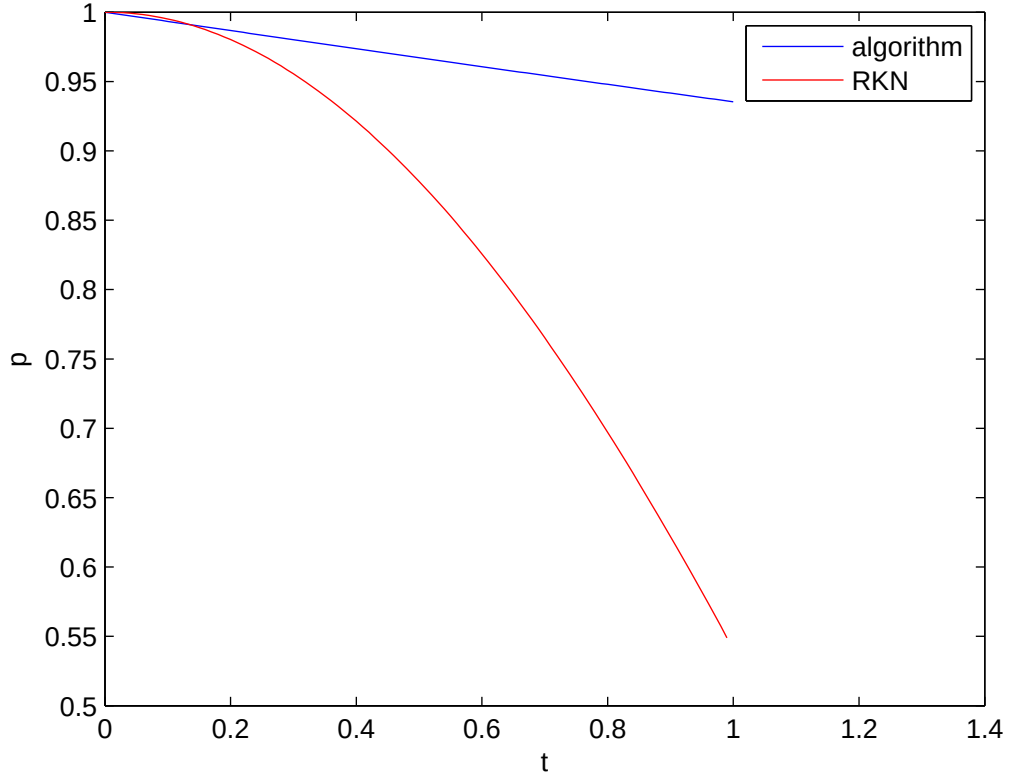


Figure 4.8: Plots of p against time for the Interacting Harmonic Oscillators system, with initial conditions $y_0 = 0$, $p_0 = 1$. We observe that the error in the plot manifests dramatically even over short time scales.

4.1.5 The Interacting Simple Pendula system

We shall be solving the equations for the ISP system, as described in Chapter 2, with the initial conditions $y_0 = 0$, $p_0 = 1$. We have constructed the partition of unity using the quadtree method described in the previous chapter, requiring in this case that the sum of the Taylor series remainders be less than $O(10^{-4})$. This yields a grid of 64 triangulation points with spacing 0.5.

In this case, the error growth has become even more drastic than in the simple pendulum case; we observe that the solutions provided by the algorithm are rapidly becoming inaccurate over a small number of time steps.

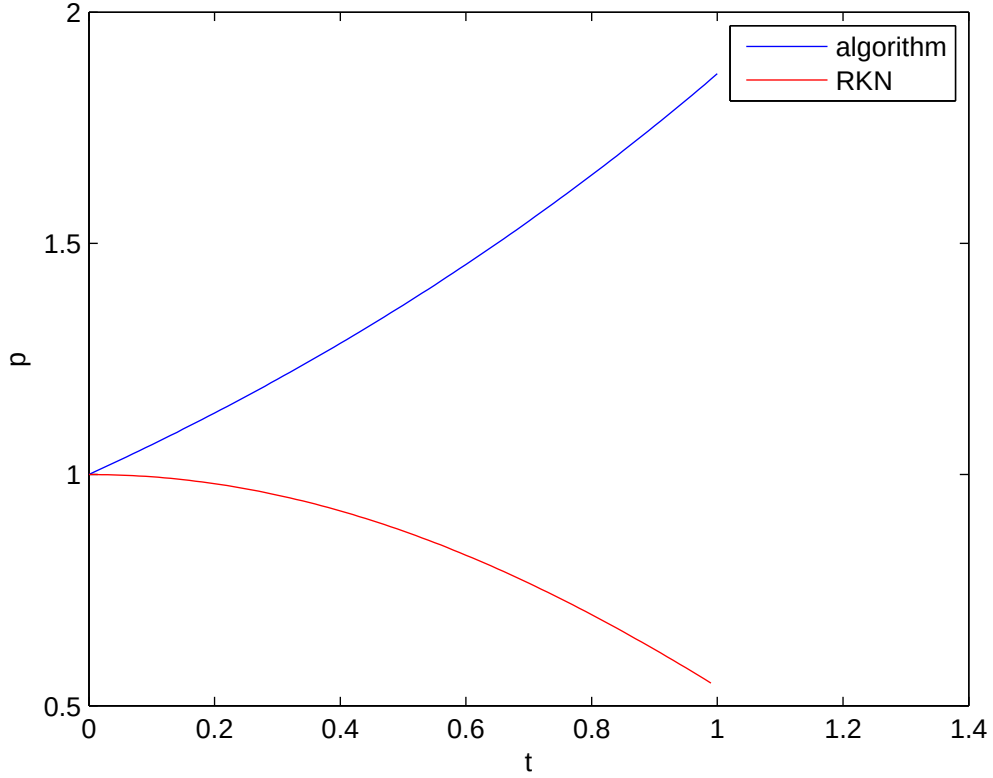


Figure 4.9: Plots of p against time for the Interacting Harmonic Oscillators system, with initial conditions $y_0 = 0$, $p_0 = 1$. We observe that the error in the plot manifests dramatically even over short time scales.

4.1.6 The Interacting Harmonic Oscillators system

We shall be solving the equations for the IHO system, as described in Chapter 2, with the initial conditions $y_0 = 0$, $p_0 = 1$. Again, we have constructed the partition of unity using the quadtree method described in the previous chapter, requiring in this case that the sum of the Taylor series remainders be less than $O(10^{-3})$. This provides a grid of 4 triangulation points with spacing 1.

As we can see from Figure 4.9, the implementation of the algorithm produces results which begin to deviate from the RKN-based solutions very early on. The IHO system was expected to be the most demanding test problem, due to its sensitivity to the influence of the control particles, and it appears that the algorithmic implementation has broken down badly with regard to this problem. In particular, the quadtree method seems incapable of producing any greater level of accuracy than the four point partition we have employed without requiring in excess of 2^{100} triangula-

tion points, since the effect of the control particles produces very large fluctuations in the strength of the vector field on a small space scale. Addressing this problem will require considerable improvements to the implementation of the algorithm.

4.2 Summary

On linear systems of ODEs, the implementation of the algorithm produces a level of accuracy comparable to fourth-order Runge-Kutta; this is to be expected, as we are in effect using the signatures and vector fields to reconstruct the original equations and then solving them forward using a fourth-order Runge-Kutta solver. When used on even a simple nonlinear system such as the simple pendulum, however, the accuracy of the implementation decreases rapidly with time, as both amplitude errors and phase shifts enter into the solution.

Chapter 5

Conclusions

5.1 Introduction

We have, in this thesis, described a new algorithm that can be used to solve differential equations driven by large numbers of vector fields. We have also discussed ways of implementing the algorithm computationally, although it is clear from the preliminary results that this implementation is flawed. We have shown that the decomposition in space and time is a viable method with a bound on its maximum error, but we have some distance to go before the computational implementation can be used on a large-scale physical problem.

5.2 The test problems

The Interacting Simple Pendula and Interacting Harmonic Oscillators systems are interesting in their own right, and could reward further study. They are both complicated dynamical systems which may display many interesting forms of behaviour. For example, we have carefully chosen the positioning of the control particles in order to prevent the response particle from becoming too close to them, in order to avoid singularities in the equations. If we could find a way to deal with such singularities, we could explore a wider variation in the types of distribution of the control particles, and how these might affect the behaviour of the response particle.

5.2.1 Physically plausible test problems

The next step in the development of these two systems would be to address the question of making the interactions between the control particle and the response particles mutual, in order to eliminate the slow growth of energy we see in the ISP

problem. It is worth noting that whilst both the simple pendulum and harmonic oscillator are Hamiltonian problems, neither the ISP or IHO systems are, since energy can be transferred from the control particles to the response particle but not vice versa. A fully mutually interacting test problem would be globally Hamiltonian, in that the total energy of the system (over all control and response particles) would be conserved. Such a system could then be used as an example of a complicated test problem driven by multiple vector fields that was also Hamiltonian, and which could then be used as a test for symplectic algorithms for solving this type of problem.

5.2.2 Higher dimensional test problems

We can also design more interesting systems by taking the same idea as the ISP and IHO systems and applying it to higher-dimensional problems. For example, consider a system in which we consider multiple interacting pendula operating under the equations of the Swinging Spring system (see [11,12,20,30], which we shall refer to as the Weakly Interacting Swinging Springs system. The WISS system has equations of motion for the response particle of the form:

$$d\mathbf{x} = \mathbf{p}dt, \quad (5.1)$$

$$d\mathbf{p} = \mathbf{F}(\mathbf{x})dt + \boldsymbol{\gamma}dt, \quad (5.2)$$

where

$$\boldsymbol{\gamma} \in \mathbb{R}^n = \sum_{i=1}^N \mathbf{G}_i(\mathbf{x}, \mathbf{p}, t), \quad (5.3)$$

is a (small) forcing term given by a superposition of N small-magnitude functions $\mathbf{G}_i(\mathbf{x}, \mathbf{p}, t)$, where N is large, and the unperturbed system of equations which describe the Hamiltonian external influences on the response particle are given by the swinging spring equations (see [11]):

$$\ddot{x} = -\frac{(r-1)}{r}x, \quad (5.4)$$

$$\ddot{y} = -\frac{(r-1)}{r}y, \quad (5.5)$$

$$\ddot{z} = -\frac{(r-1)}{r}z - L, \quad (5.6)$$

where $\mathbf{y} = (x, y, z)$, $\mathbf{p} = (p_x, p_y, p_z) \in \mathbb{R}^3$ specify the trajectory of the response particle, and L is a parameter.

The external influences on the i th control particle are also given by the swinging spring equations;

$$d\mathbf{x}_i = \mathbf{p}_i dt, \quad (5.7)$$

$$d\mathbf{p}_i = \left(-\frac{|\mathbf{x}_i| - 1}{|\mathbf{x}_i|} \mathbf{x}_i + \begin{pmatrix} 0 \\ 0 \\ -L' \end{pmatrix} \right) dt, \quad (5.8)$$

where the i th control particle has a trajectory given by $\mathbf{x}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$, $\mathbf{p}_i = (p_{x_i}, p_{y_i}, p_{z_i}) \in \mathbb{R}^3$, and L' is a parameter which is the same for all the control particles.

The forcing term γ is given by the interactions between the cloud of control particles and the response particle. As in the IHO and ISP systems, we assume that the attraction between each control particle and the response particle takes the form of a vector field of a form similar to a gravitational attractive force, governed by a similar inverse square law, and so we have

$$\gamma = \sum_{i=1}^N G \frac{\mathbf{x}_i - \mathbf{x}}{|\mathbf{x}_i - \mathbf{x}|^3}, \quad (5.9)$$

where x_i is the position vector of the i th control particle and $G \ll 1$ is a parameter.

Hence, the equations of motion of the response particle under the total resultant influences would be

$$\dot{\mathbf{x}} = \mathbf{p}, \quad (5.10)$$

$$\dot{\mathbf{p}} = -\frac{|\mathbf{x}| - 1}{|\mathbf{x}|} \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ -L \end{pmatrix} + \sum_i^N G \frac{\mathbf{x}_i - \mathbf{x}}{|\mathbf{x}_i - \mathbf{x}|^3}. \quad (5.11)$$

Here we have an example of a test problem in 6 dimensions driven by a large number of vector fields, which we can investigate in the same way as the IHO and ISP systems. We could also use this a test for a higher-dimensional version of the algorithm we constructed in this thesis.

5.2.3 Stochastic test problems

It is also worth noting that the test problems we have studied so far are not stochastic, and do not produce a very rough driving path. The algorithm should, in theory, be able to deal with a rough driving path, so long as it is of bounded p -variation, but the question of how many terms of the signature we would need to take in order to obtain the same level of accuracy that we have managed for the previous algorithms remains to be addressed.

We could consider the construction of a test problem based on one of the Interacting Pendula systems; in this case the response particle would be governed by the pendulum equations, but the control particles would be subject to small random perturbations. It would, in this case, be less straightforward to guarantee the absence of close transits based on initial positioning (unless the distance between the control and response particles was very large and the maximum magnitude of the random perturbations very small), so it might be best only to deal with this problem once we have worked out a way of coping with singularities in the system (as we discussed above).

5.3 Improvements to the implementation of the algorithm

In implementing the algorithm for our current test problems, we have had to sacrifice accuracy for the sake of speed and computational simplicity. There are several areas in which we have had to compromise on the level of accuracy we can expect from the algorithm: the placing of the triangulation points; the number of terms of Taylor series expansion we take; and the number of terms of the signature that we consider.

If we choose a finer grid of triangulation points, we will have to consider a larger value of J . This will increase the number of vector fields involved when computing the Hall basis, although this section in the code is capable of quickly finding basis vectors for a large alphabet size. Taking more terms of the Taylor series is unlikely to slow the algorithm down considerably, as Neidinger's code for computing the coefficients is reasonably fast and efficient. It would, however, also increase the number of basis vectors involved whilst computing the Hall basis. Taking more terms of the signature will require us to find the Hall basis to a greater depth, which will slow the code down considerably. It will also require us to find further Lie brackets, which is a symbolic operation in Matlab and hence considerably slower than the numerical operations. In general, the code for implementing the algorithm can also be made faster by using numerical rather than symbolic computations wherever possible. Looking into these options, and finding out which one will be the most viable, will be the next step on the way to producing a robust implementation for the algorithm that is capable of producing a higher level of accuracy.

Bibliography

- [1] I. Babuska and J. M. Melenk. The partition of unity method. *International Journal of Numerical Methods in Engineering*, 40:727–758, 1996.
- [2] Rick Beatson and Leslie Greengard. A short course on fast multipole methods. In *Wavelets, Multilevel Methods and Elliptic PDEs*, pages 1–37. Oxford University Press, 1997.
- [3] Kuo-Tsai Chen. Integration of paths—a faithful representation of paths by non-commutative formal power series. *Trans. Amer. Math. Soc.*, 89:395–407, 1958.
- [4] Terry Lyons, Stephen Buckley, Djalil Chafai, Greg Gyurkó and Arend Janssen. Computational rough paths libraries. <http://coropa.sourceforge.net/>, 2010.
- [5] Ernst Hairer, Christian Lubich, and Gerhard Wanner. Geometric numerical integration illustrated by the Störmer Verlet method. *Acta Numerica*, 12:399–450, 2003.
- [6] A. Iserles, A. Marthinsen, and S. P. Nørsett. On the implementation of the method of Magnus series for linear differential equations, 1998.
- [7] Christophe Ladroue. Iixp, Iixpzt and Ilsym matlab objects. <http://coropa.sourceforge.net/Docs/MatlabSource/>, 2007.
- [8] Marcos Lage, Fabiano Petronetto, Afonso Paiva, Hlio Lopes, and Thomas Lewiner. Vector field reconstruction from sparse samples with applications. In *In Sibgrapi*, pages 297–304, 2006.
- [9] Antoine Lejay. An introduction to rough paths. Preprint at <http://www.iecn.u-nancy.fr/~lejay/rough.html>, 2002.
- [10] Adrian Lew, Jerrold Marsden, Michael Ortiz, and Matthew West. An overview of variational integrators. In *Finite Element Methods: 1970s and Beyond*.

- [11] Peter Lynch. Resonant motions of the three-dimensional elastic pendulum. *Int J. Nonlin. Mech.*, 37:345–367 (electronic), 2002.
- [12] Peter Lynch and Darryl Holm. Stepwise precession of the resonant swinging spring. *SIAM J. APPLIED DYNAMICAL SYSTEMS*, 1(1):44–64 (electronic), 2002.
- [13] D. Lynden-Bell. From quasars to extraordinary N-body problems. *Annals of the New York Academy of Sciences*, 867:3–13, 1998.
- [14] Terry Lyons. Differential equations driven by rough signals. *Rev. Mat. Iberoamericana*, 14(2):215–310, 1998.
- [15] Terry Lyons. Differential equations driven by rough signals. *Rev. Mat. Iberoamericana*, 14(2):215–310, 1998.
- [16] Terry Lyons and Zhongmin Qian. *System Control and Rough Paths*. Oxford University Press, Oxford, 2002. Oxford Mathematical Monographs.
- [17] Terry J. Lyons and Nadia Sidorova. Sound compression: a rough path approach. In *WISICT '05: Proceedings of the 4th international symposium on information and communication technologies*, pages 223–228. Trinity College Dublin, 2005.
- [18] Richard D. Neidinger. Computing multivariable Taylor series to arbitrary order. *SIGAPL APL Quote Quad*, 25(4):134–144, 1995.
- [19] Richard D. Neidinger. Directions for computing truncated multivariate Taylor series. *Mathematics of Computation*, 74(249):321–340, 2005.
- [20] John Norbury and Ian Roulstone, editors. *Large-scale atmosphere-ocean dynamics. Vol. I*. Cambridge University Press, Cambridge, 2002.
- [21] John Norbury and Ian Roulstone, editors. *Large-scale atmosphere-ocean dynamics. Vol. II*. Cambridge University Press, Cambridge, 2002.
- [22] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, Hans-Peter Seidel, and Mpi Saarbrcken. Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22:463–470, 2003.
- [23] J. M. Sanz-Serna and M. P. Calvo. *Numerical Hamiltonian problems*, volume 7 of *Applied Mathematics and Mathematical Computation*. Chapman & Hall, London, 1994.

- [24] David Sherrington and Scott Kirkpatrick. Solvable model of a spin-glass. *Phys. Rev. Lett.*, 35(26):1792–1796, Dec 1975.
- [25] L. Solokov. Families of integrable and stochastic trajectories in the N-body problem. *Astronomical and Astrophysical Transactions*, 7:275–276, 1995.
- [26] Jürgen Struckmeier and Claus Riedel. Invariants for time-dependent Hamiltonian systems. *Phys. Rev. E*, 64(2):026503, Jul 2001.
- [27] F. Varadi, M. Ghil, and W.M. Kaula. Mass-weighted symplectic forms for the N-body problem. *Celestial Mechanics and Dynamical Astronomy*, 72:187–199, 1999.
- [28] Atakan Varol. Iterated Lie bracket. <http://www.mathworks.co.uk/matlabcentral/fileexchange/10512-iterated-lie-bracket>, 2006.
- [29] Nicolas Victoir. *From cubature to rough paths*. PhD thesis, University of Oxford, 2003.
- [30] A Vitt and G Gorelik. Oscillations of an elastic pendulum as an example of the oscillations of two parametrically coupled linear systems. *Historical Note No. 3, Met Éireann, Dublin (1999)*, 1933. Translated by Lisa Shields, with an Introduction by Peter Lynch.
- [31] Magnus W. On the exponential solution of differential equations for a linear operator. *Comm. Pure. Appl. Math.*, 7:649673, 1954.
- [32] Eric W. Weisstein. Taylor series. <http://mathworld.wolfram.com/TaylorSeries.html>.