Unstable Space of Discretized Partial Differential Equations

Mats Veldhuizen
3971740

June 16, 2016

1 Introduction

1.1 Data assimilation

Data assimilation is the process of continuously gathering data, calculating predictions based on dynamical models and updating these predictions with new data. The method of data assimilation for a model with multiple degrees of freedom is as follows.
First a measurement is taken of a predetermined number of the degrees of freedom. These measurements are used to approximate the current state of the entire system. Based on this approximation, a prediction is made for the system after one time-step using the model. After one time-step has passed, a new measurement is taken in the same manner as the first one. Also a new approximation is made. This new approximation is compared to the predictions made at the previous time-steps to generate a better approximation of the current state and a better prediction of the coming states. This process is continuously repeated to generate better predictions and approximations of the system then would normally have been possible using either the data or the model alone.
An example of where data assimilation is used is in weather forecasting [1], where the current state of the system is not accurately known, because weather models have a very high number of degrees of freedom. The weather models model the entire atmosphere so any reasonable discretization of these states are bound to have far more degrees of freedom than the number of weather stations that are available. So data assimilation is an ideal method to make weather predictions more accurate.
Another example of where data assimilation is used is ocean flows, which have almost identical complications to the weather forecasting system.
In this paper we will focus on the stability of such discretized partial differential equations and in particular their Lyapunov spectrum. With the Lyapunov spectrum we will calculate the Kaplan-Yorke dimension (more on this in section 2.5). The Lyapunov spectrum is a valuable asset in data assimilation, because "[...] the requisite number of tracking observations is closely related to the number and magnitude of the system's positive Lyapunov exponents." ([2] section 1B). We have created a program that we used to test how the unstable space (and Kaplan-Yorke dimension) of one dimensional partial differential equations

scales with the resolution (more on this in the next section).

1.2 The program

Alongside this paper we created a program, based on numerical techniques found in e.g. [3] and [10], that can be used to estimate the Lyapunov exponents of one dimensional models and used this program to analyze three (1+1)-dimensional partial differential equations. A (1+1)-dimensional equation is an equation that has one spatial and one temporal dimension. The test equations we used were the damped forced Burgers-Hopf equation, the stochastic Burgers equation and the Kuramoto-Sivashinsky equation (more on these equations in section 4). To create this program we studied Lyapunov exponents, in particular numerical methods for approximating these exponents. We also needed to use a special integration technique (implicit-explicit Runge-Kutta method) in order to accurately and efficiently integrate the Kuramoto-Sivashinsky equation. In this paper we will extensively discuss both these subjects.

1.3 Outline

This paper is organized as follows.
In section 2 we will discuss the Lyapunov exponents. In the corresponding subsections we will discuss the following. In section 2.1 we will start by explaining what Lyapunov exponents are and how to derive a definition for them, that will be useful to approximate them numerically. Section 2.2 will be about the numerical method we used to approximate the Lyapunov exponents. This method is called the discrete QR-method and relies on the QR-decomposition of matrices. The Modified Gram-Schmidt method is the decomposition method we used and is explained in section 2.3. As seen in section 2.2 we will need to find the Jacobian matrix corresponding to the partial differential equations. Because these Jacobian matrices will be mostly empty, explicitly calculating them dramatically decreases efficiency. Section 2.4 will explain the method we used to avoid this problem. To approximate the dimension of the attractor for our test equations we used the Kaplan-Yorke conjecture. Section 2.5 will discuss this subject.
Section 3 is about the implicit-explicit Runge-Kutta (IMEX RK) integration methods, which are schemes used for partial differential equations where one part is stiff and linear, but which also have a non-stiff (and non-linear) part. This section is made up of seven subsections. A subsection on the explicit Runge-Kutta 4 method, which we used for the equations that did not require an IMEX method (section 3.1). This section is followed by an introduction to IMEX RK schemes, explaining the advantages and disadvantages of these methods compared to a regular explicit (or implicit) method (section 3.2). After the introduction we will show how these methods can be created by taking an implicit method and coupling it with a corresponding explicit method (section 3.3). Then we will show how this combination of methods can be applied to an equation by using the implicit scheme for the stiff part and the explicit scheme for the non-stiff part (section 3.4). Some examples of IMEX RK combinations are given section 3.5. The stability of those examples will be reviewed in section 3.6. We conclude this section with section 3.7, a section about more efficiently

computing the solution to the matrix equation that arises when applying an IMEX RK method, by using a fast Fourier transform.
In section 4 we will discuss the three test equations we analyzed with our program. In the subsections 4.1, 4.2 and 4.3 we discuss the damped forced Burgers-Hopf equation, the stochastic Burgers equation and the Kuramoto-Sivashinsky equation respectively.
The numerical results of our program are reported in section 5. We start this section off by outlining the discretizations of the test equations we used in section 5.1 and the different values for e.g. the number of degrees of freedom we used in section 5.2. The actual results are reported and discussed in section 5.3.

2 Lyapunov exponents

2.1 Introduction to Lyapunov exponents

In this section we shall give the definition of Lyapunov exponents and derive a definition that will be of use when numerically approximating them. Readers who wish to directly look at how Lyapunov exponents are calculated may skip to the next section.
Lyapunov exponents are numbers associated with the stability of dynamical systems. In a multidimensional system there is one Lyapunov exponent per dimension. Each of these Lyapunov exponents defines the exponential growth of an infinitesimal initial separation between 2 trajectories in phase space in its respective dimension. In this paper we will be looking at m-dimensional discretizations of continues partial differential equations in the form

$$\dot{u} = f(u), \quad u(0) = u_0 \tag{1}$$

With $u \in \mathbb{R}^m$. Following the introduction of [3] the Lyapunov exponents can be approximated by looking at the linearized problem

$$\dot{y} = A(t)y, \quad y(0) = y_0 \tag{2}$$

Where $y \in \mathbb{R}^m$ and $A(t) \in \mathbb{R}^{m\times m}$ is the Jacobian matrix of f in $u(t)$. The definition we will use to approximate the Lyapunov exponents is given in [5]: "If we have a regular system with upper triangular coefficient matrix $B(t)$:

$$\dot{x} = B(t)x \tag{3}$$

Then, for $i = 1, ..., n$, its Lyapunov exponents are given by

$$\lambda_i = \lim_{t\to\infty} \frac{1}{t} \int_0^t B_{ii}(s)ds" \tag{4}$$

Since $A(t)$ is not necessarily in upper triangular form this requires us to apply a Lyapunov transformation to (2) (more on this in section 2.2).

2.2 Discrete QR method

In this section we will discuss the discrete QR method we used to numerically approximate the Lyapunov exponents.

In order to find the Lyapunov exponents using (4) we will need to apply a change of variables that will transform $A(t)$ to an upper triangular matrix. This change of variables also has to preserve the Lyapunov exponents and therefore has to be a Lyapunov transformation as defined in [4] (definition 2.4): "A smooth invertible change of variables $T^{-1}y \to x$ is called a Lyapunov transformation if T, T^{-1} and $\dot{T}$, are bounded". In this paper we used the discrete QR method as described in [3], that will achieve this change of variables through a QR decomposition. In the discrete QR method we will look at the following system.

$$\dot{Y} = A(t)Y, \quad Y(0) = Y_0 \tag{5}$$

With $Y \in \mathbb{R}^{m\times n}$ ($n \leqslant m$ is the number of Lyapunov exponents we want to compute). We will now show why a QR decomposition of Y will result in a system with an upper triangular matrix $B(t)$. Filling $Y = QR$, with $Q \in \mathbb{R}^{m\times n}, R \in \mathbb{R}^{n\times n}$, into (5) we get

$$\dot{Q}R + Q\dot{R} = A(t)QR \tag{6}$$

Which will give us an equation for R

$$\dot{R} = (Q^T A(t)Q - Q^T\dot{Q})R, \quad R(0) = R_0 \tag{7}$$

Since both R and $\dot{R}$ are upper triangular matrices, the matrix $(Q^T A(t)Q - Q^T\dot{Q}) = B(t) \in \mathbb{R}^{n\times n}$ is also upper triangular. Because the QR decomposition is a Lyapunov transformation and the larger Lyapunov exponents are more dominant, the Lyapunov exponents of (7) are the n largest Lyapunov exponents of (2).
For the QR-decomposition we used the modified Gram-Schmidt process (more on this in section 2.3), because the modified Gram-Schmidt process preserves the ordering of the matrices involved. Therefore it ensures that the Lyapunov exponents come out ordered and we can compute the n largest exponents.
The iterative process for the discrete QR method is as follows.
First we pick a Y_0 randomly in order "[...]to guarantee that all possible growth behavior is represented in the growth of the columns of $Y(t)$, and that the n most dominant exponents will emerge, ordered, during integration." ([3] page 8) and compute the QR decomposition $Y_0 = Q_0\hat{R}_0$. (We will do 1000 iterations on the system before we will start to calculate the Lyapunov exponent in order to eliminate spikes created by the randomly chosen Y_0.)
Next we let $t_j = jh$ with h the time-step and for $j = 0, ..., k$ we will define $Z_{j+1}(t_j) = Q_j$ and simultaneously integrate (1) and

$$\dot{Z}_{j+1} = A(t)Z_{j+1} \tag{8}$$

With $Z : t \in [t_j, t_{j+1}] \to \mathbb{R}^{m\times n}$ over one time-step (from t_j to t_{j+1}) and do another QR decomposition on $Z_{j+1}(t_{j+1})$

$$Z_{j+1}(t_{j+1}) = Q_{j+1}\hat{R}_{j+1} \tag{9}$$

This will define the QR decomposition of $Y(t_{k+1})$ as

$$Y(t_{k+1}) = Q_{k+1}[\hat{R}_{k+1}\hat{R}_k...\hat{R}_1\hat{R}_0] \tag{10}$$

The system (8) we used is slightly different from the system suggested in [3], but our system ensures the validity of (10), because $Y(0) = Y_0 = Q_0\hat{R}_0$ and $Z_1(0) = Q_0$ and if

$$Y(t_l) = Q_l[\hat{R}_l\hat{R}_{l-1}...\hat{R}_1\hat{R}_0] \tag{11}$$

And

$$Z_{l+1}(t_l) = Q_l \tag{12}$$

For any value of l then

$$\begin{aligned} \dot{Y}(t_l) &= A(t_l)Q_l[\hat{R}_l\hat{R}_{l-1}...\hat{R}_1\hat{R}_0] \\ \dot{Z}_{l+1}(t_l) &= A(t_l)Q_l \end{aligned} \tag{13}$$

And because both are linear systems this will give

$$\begin{aligned} Y(t_{l+1}) &= C(t_l)Q_l[\hat{R}_l\hat{R}_{l-1}...\hat{R}_1\hat{R}_0] \\ Z_{l+1}(t_{l+1}) &= C(t_l)Q_l \end{aligned} \tag{14}$$

With $C(t_l) \in \mathbb{R}^{m\times m}$ a matrix that depends on $A(t_l)$ and the integration scheme used. Using (14) and (9) we get (10) through induction.
After the first 1000 iterations we will start forming a one dimensional array of length n by adding $\dfrac{1}{t_{max}}\log(\hat{R}_j)_{ii}$ to the ith element of this originally empty array after each time-step. With $\dfrac{t_{max}}{h} = k \gg 0$ and k the number of iterations after the first 1000. This will define the Lyapunov exponents based on the relation.

$$\lambda_i = \lim_{k\to\infty}\frac{1}{t_k}\sum_{j=1}^{k}\log(\hat{R}_j)_{ii}, \quad i = 1, ..., n \tag{15}$$

Which is equivalent to (4), because $\dot{R} = BR$. (15) is once more slightly different from the equation given in [3], but as we will show it is equivalent to (4). Because both B and R are upper triangular matrices, the diagonal elements of R can be described by a system of n separate one dimensional differential equations

$$\dot{R}_{ii} = B_{ii}R_{ii}, \quad i = 1, ..., n \tag{16}$$

With as solutions

$$R_{ii}(t_k) = (R_0)_{ii}e^{\int_0^{t_k} B_{ii}(s)ds}, \quad i = 1, ..., n \tag{17}$$

$R_{ii}(t_k)$ is defined as the product $\prod_{j=0}^{k}(\hat{R}_j)_{ii}$ (from (10)). Taking the natural logarithm of (17) and filling in this product will give

$$\sum_{j=0}^{k}\log(\hat{R}_j)_{ii} = \log(\hat{R}_0) + \int_0^{t_k} B_{ii}(s)ds, \quad i = 1, ..., n \tag{18}$$

Which combined with (15) will give (4).

2.3 Modified Gram-Schmidt QR-decomposition

In this section we will show how the modified Gram-Schmidt QR-decomposition is computed.
The modified Gram-Schmidt process is a more numerically stable variant of the regular Gram-Schmidt process and is a method used for orthonormalising a set of vectors. This process is also applicable to the QR-decomposition of matrices by combining this orthonormalized set in a matrix Q and defining R by $V = QR$ where $V \in \mathbb{R}^{m \times n}$ is the matrix to be decomposed. To find the QR-decomposition we first need to define the projection operator as.

$$\text{proj}_u(v) = \frac{v^T u}{u^T u} u \tag{19}$$

With this projection operator we define the following vector. Referring to the columns of V as vectors $V = [v_1, ..., v_n]$.

$$u_k^{(0)} = v_k \tag{20}$$

Next for $k = 2, ..., n$ we compute u_k by progressively evaluating the following equation.

$$u_k^{(i)} = u_k^{(i-1)} - \text{proj}_{u_i}(u_k^{(i-1)}) \tag{21}$$

u_k will now be defined as $u_k^{(k-1)}$. We will also define

$$e_i = \frac{u_i}{||u_i||} \tag{22}$$

After computing all u_k and all e_i we can compute the matrices Q and R in the following way. Q will be the matrix defined by

$$Q = [e_1, ..., e_n] \tag{23}$$

And R will be

$$R = \begin{bmatrix} e_1^T v_1 & e_1^T v_2 & e_1^T v_3 & \cdots & e_1^T v_n \\ 0 & e_2^T v_2 & e_2^T v_3 & \cdots & e_2^T v_n \\ 0 & 0 & e_3^T v_3 & \cdots & e_3^T v_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & e_n^T v_n \end{bmatrix} \tag{24}$$

2.4 Avoiding the Jacobian matrix

In this section we will discuss why it is more efficient to avoid explicitly forming the Jacobian matrix and how we implemented functions for the Jacobian matrix for each of the analyzed partial differential equations in this paper.
In section 2.2 we used a $Y(t) \in \mathbb{R}^{m \times n}$ and solution to (5) to approximate only the n largest Lyapunov exponents. Since we want to find the Kaplan-Yorke dimension of the test equations (more on this in section 2.5), it is sufficient to only look for the $n \leqslant m$ largest Lyapunov exponents and in some cases even the $n \ll m$ largest Lyapunov exponents.
As indicated in [3] this makes explicitly calculating the Jacobian matrix at every time-step inherently inefficient, because calculating the Jacobian matrix will

take at least $O(m^2)$ operations and calculating AV, with $A \in \mathbb{R}^{m\times m}$ the Jacobian and V element of $\mathbb{R}^{m\times n}$ will take $O(m^2 n)$. Formulating an equation that directly calculates AV will do this in $O(mn)$, so for the discretization of each of the partial differential equations implemented in our program we formed a function that would calculate AV in $O(mn)$.
As an example we will show how we did this with the finite difference discretization of the damped forced Burgers-Hopf equation

$$u_t = uu_x - u + F \tag{25}$$

The discretization is as indicated in [6] assuming periodic boundary conditions.

$$\frac{du_i}{dt} = f(u_i) = \frac{u_{i+1} - u_{i-2}}{3\Delta x} u_{i-1} - u_i + F \tag{26}$$

It is clear to see that when forming a Jacobian matrix of this discretization there will only be four non-zero terms per row. Namely for the ith row the terms at positions $i-2, ..., i+1$. Therefore the equation for AV will turn into

$$(AV)_{i,j} = V_{i-2,j}\frac{\partial f(u_i)}{\partial u_{i-2}} + V_{i-1,j}\frac{\partial f(u_i)}{\partial u_{i-1}} + V_{i,j}\frac{\partial f(u_i)}{\partial u_i} + V_{i+1,j}\frac{\partial f(u_i)}{\partial u_{i+1}} \tag{27}$$

Or

$$(AV)_{i,j} = -V_{i-2,j}\frac{u_{i-1}}{3\Delta x} + V_{i-1,j}\frac{u_{i+1} - u_{i-2}}{3\Delta x} - V_{i,j} + V_{i+1,j}\frac{u_{i-1}}{3\Delta x} \tag{28}$$

A similar method was applied to the other equations.

2.5 Kaplan-Yorke dimension

In this section we will discuss what the Kaplan-Yorke dimension is and how to calculate it using the Lyapunov spectrum.
A Lyapunov spectrum $\lambda_1 \leqslant \lambda_2 \leqslant ... \leqslant \lambda_{n-1} \leqslant \lambda_n$ can be used to calculate the Kaplan-Yorke dimension, which is an estimate for the dimension of the chaotic attractor of the system. An attractor is a multidimensional space that attracts initial conditions, close enough to it, towards it. These initial conditions will be bounded to this attractor after transients have died out. In this attractor the bounded initial conditions can either exibit periodic or chaotic behaviour. The dimension of such an attractor can be approximated by the Kaplan-York conjecture [8].

$$D_{KY} = k + \frac{\sum_{i=1}^{k} \lambda_i}{|\lambda_{k+1}|} \tag{29}$$

Where k is the index of the Lyapunov exponent for which $\sum_{i=n}^{k} \lambda_i > 0$ and $\sum_{i=n}^{k+1} \lambda_i < 0$.

3 Implicit explicit Runge-Kutta methods

3.1 Runge-Kutta 4

For the damped forced Burgers-Hopf equation we used the famous Runge-Kutta four (RK4) method. This method has the following Butcher tableau.

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

This method is fourth order accurate and is used to integrate a differential equation in the form of (1) for one time-step (from t_n till $t_{n+1} = t_n + h$ by computing

$$\begin{cases} K_1 = f(y_n) \\ K_2 = f(y_n + \frac{h}{2}K_1) \\ K_3 = f(y_n + \frac{h}{2}K_2) \\ K_4 = f(y_n + hK_3) \\ y_{n+1} = y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \end{cases} \tag{30}$$

Since this is an explicit method it was not suitable to integrate the Kuramoto-Sivashinky equation, because it has a very stiff term. In order to integrate over this equation we needed an implicit-explicit (IMEX) differentiation scheme.

3.2 Introduction to IMEX RK schemes

In this section we will review what the applications of IMEX RK methods are and what advantages they have over regular implicit or explicit RK schemes. Reader who wish to directly look at how IMEX RK schemes are created may skip to the next section.

IMEX RK methods are associated with discretizised partial differential equations in the form

$$\dot{u} = Gu + f(u) \tag{31}$$

With $u \in \mathbb{R}^m$. Where G is a stiff and linear matrix and f is non-stiff and nonlinear. "An IMEX scheme consists of applying an implicit discretization for G and an explicit one for f" ([10] section 1). An IMEX scheme is needed in a situation like this, because integrating G explicitly will result in stability issues unless a unacceptable small time-step is used. Trying to do the entire integration implicitly, however, requires solving an expensive nonlinear algebraic system in each time-step. Although stiffness is not well-defined it has to do with the necessity of picking much smaller time-steps then normally expected, in relation to the smoothness of the solution, in order to achieve stable integration over the system. Explicit methods are especially susceptible to this phenomenon, while implicit methods generally perform reasonably well even for larger time-steps. There are special cases of implicit methods (A-stable and L-stable methods) which have the best stability properties for stiff systems. The disadvantage of implicit methods is that, when applied to (31), they require a system of the following form to be solved for $z \in \mathbb{R}^m$.

$$z = G(a + z) + f(a + z) \tag{32}$$

With $a \in \mathbb{R}^m$. This will be difficult to solve if f is not a linear function. Because (31) has both stiff (linear) and non-stiff (nonlinear) terms a solution seems to

be to use a combination of both implicit and explicit schemes.

3.3 Creating an IMEX RK scheme

In this section we will show the technique for creating an IMEX RK scheme. Following [10] we will show how to create an IMEX RK scheme. We start with an s-stage diagonally implicit Runge-Kutta (DIRK) scheme for G in the Butcher notation with coefficients $A \in \mathbb{R}^{s\times s}$, $c, b \in \mathbb{R}^s$. Next we pick an $(s+1)$-stage explicit Runge-Kutta scheme and define $\sigma = s+1$. We pick the explicit scheme so that its abscissae satisfies $\hat{c} = \begin{pmatrix} 0 \\ c \end{pmatrix}$ with coefficients $\hat{A} \in \mathbb{R}^{\sigma\times\sigma}$, $\hat{b} \in \mathbb{R}^\sigma$. We now pad the DIRK scheme with zeros to obtain the IMEX RK scheme

0	0	0	0	$\cdots$	0
c_1	0	a_{11}	0	$\cdots$	0
c_2	0	a_{21}	a_{22}	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_s	0	$a_{s,1}$	$a_{s,2}$	$\cdots$	$a_{s,s}$
	0	b_1	b_2	$\cdots$	b_s

Implicit

0	0	0	$\cdots$	0	0
c_1	$\hat{a}_{21}$	0	$\cdots$	0	0
c_2	$\hat{a}_{31}$	$\hat{a}_{32}$	$\cdots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
c_s	$\hat{a}_{\sigma,1}$	$\hat{a}_{\sigma,2}$	$\cdots$	$\hat{a}_{\sigma,s}$	0
	$\hat{b}_1$	$\hat{b}_2$	$\cdots$	$\hat{b}_s$	$\hat{b}_\sigma$

Explicit

We will refer to the coefficients of the implicit tableau padded with zeros as $\tilde{A} \in \mathbb{R}^{\sigma\times\sigma}$, $\tilde{b}, \tilde{c} \in \mathbb{R}^\sigma$.

3.4 Applying an IMEX RK scheme

In this section we will show how to integrate for one time-step using an IMEX RK scheme as shown in [10]. We will show how to integrate a function in the form of (31) for one step from t_{n-1} to $t_n = t_{n-1} + h$. To do so we will first define

$$\hat{K}_1 = f(u_{n-1}) \tag{33}$$

Next for $i = 1, ..., s$ we will solve

$$K_i = Gu_i \tag{34}$$

For K_i, with

$$u_i = u_{n-1} + h\sum_{j=1}^{i} a_{i,j}K_j + h\sum_{j=1}^{i} \hat{a}_{i+1,j}\hat{K}_j \tag{35}$$

And define

$$\hat{K}_{i+1} = f(u_i) \tag{36}$$

After doing the above two steps for all i we will evaluate

$$u_n = u_{n-1} + h\sum_{j=1}^{s} b_jK_j + h\sum_{j=1}^{\sigma} \hat{b}_j\hat{K}_j \tag{37}$$

When forming IMEX RK methods it helps to pick them so that they fall into two strongly different families of methods as also shown in [10], namely those methods in which $\tilde{b} = \hat{b}$ and thus $\hat{b}_1 = 0$ which will simplify (37) to

$$u_n = u_{n-1} + h\sum_{j=1}^{s} b_j(K_j + \hat{K}_{j+1}) \tag{38}$$

And those methods in which $\hat{b}_{s+1} = 0$ so that $\hat{K}_{s+1}$ does not need to be calculated. For these methods we also require

$$b_j = a_{s,j}, \quad \hat{b}_j = \hat{a}_{s+1,j}, \quad j = 1, ..., s \tag{39}$$

Substituting (39) into (35) for $i = s$ we get (37) implying that

$$u_n = u_s \tag{40}$$

Which is useful for very stiff equations. Since we do not need to calculate $\hat{K}_{s+1}$ anymore we can simplify the explicit scheme to an s-stage scheme

$$\begin{array}{c|ccccc} 0 & 0 & 0 & \cdots & 0 & 0 \\ c_1 & \hat{a}_{21} & 0 & \cdots & 0 & 0 \\ c_2 & \hat{a}_{31} & \hat{a}_{32} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{s-1} & \hat{a}_{s,1} & \hat{a}_{s,2} & \cdots & \hat{a}_{s,s} & 0 \\ \hline & \hat{b}_1 & \hat{b}_2 & \cdots & \hat{b}_{s-1} & \hat{b}_s \end{array}$$

When solving (34) we need to find the inverse of

$$I - ha_{i,i}G, \quad i = 1, ..., s \tag{41}$$

Something that could decrease efficiency is picking a DIRK scheme that has different values at $a_{i,i}$ for different i. Therefore all the schemes mentioned in the next section will have the same values for each $a_{i,i}$. Since the matrix G is a cyclic matrix in our application, we could avoid inverting this matrix altogether by using a fast Fourier transform (more on this in section 3.7).

3.5 Examples of IMEX RK schemes

In this section we will show some examples of IMEX RK schemes from [10]. In order to refer to these schemes we shall use the following notation (s, σ, p) to identify a combination of an s-stage implicit scheme with a σ-stage explicit scheme. So if $\sigma = s + 1$ it is part of the family for which (38) holds and if $\sigma = s$ it is part of the famlily for which (40) holds. p is the combined order of the IMEX RK scheme.

3.5.1 Forward backward Euler (1,1,1) and (1,2,1)

The forward Euler and backward Euler can be combined in two different IMEX RK schemes (one scheme for each of the families). The forward-backward Euler

(1,1,1)

0	0	0
1	0	1
	0	1

0	0	0
1	1	0
	1	0

Implicit Explicit

And the forward-backward Euler (1,2,1)

0	0	0
1	0	1
	0	1

0	0	0
1	1	0
	0	1

Implicit Explicit

Both are first order accurate, but because of the relation (40) the forward-backward Euler (1,1,1) method needs only one computation of f whereas the (1,2,1) method needs two.

3.5.2 Implicit-explicit midpoint (1,2,2)

The implicit-explicit midpoint is a combination of two second order accurate integration schemes and therefore it is itself second order accurate. The IMEX RK scheme that corresponds to this method is as follows

0	0	0
$\frac{1}{2}$	0	$\frac{1}{2}$
	0	1

0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0
	0	1

Implicit Explicit

This scheme has some good symmetry properties and performs reasonably well considering it is still just a (1,2)-stage scheme.

3.5.3 A third order combination (2,3,3)

The (2,3)-stage third order accurate scheme with the best damping properties is

0	0	0	0
γ	0	γ	0
$1-\gamma$	0	$1-2\gamma$	γ
	0	$\frac{1}{2}$	$\frac{1}{2}$

0	0	0	0
γ	γ	0	0
$1-\gamma$	$\gamma-1$	$2(1-\gamma)$	0
	0	$\frac{1}{2}$	$\frac{1}{2}$

Implicit Explicit

With $\gamma = \dfrac{3+\sqrt{3}}{6}$. This scheme is both third order accurate and also has some attenuation at the stiffness limit of ∞ and is therefore quite suitable for

stiff problems.

3.5.4 L-stable, two-stage, second-order DIRK (2,3,2)

For very stiff problems the attenuation of the previous scheme might not be sufficient to guarantee stability therefore we used the following scheme based on an L-stable, two-stage, second-order DIRK (diagonally-implicit Runge-Kutta) for the integration of the Kuramoto-Sivashinsky equation.

0	0	0	0
γ	0	γ	0
1	0	$1-\gamma$	γ
	0	$1-\gamma$	γ

Implicit

0	0	0	0
γ	γ	0	0
1	δ	$1-\delta$	0
	0	$1-\gamma$	γ

Explicit

With $\gamma = \dfrac{2-\sqrt{2}}{2}$ and $\delta = \dfrac{-2\sqrt{2}}{3}$. Because of L-stability, this scheme ensured accurate integration over the Kuramoto-Sivashinsky equation. Even though this system is only second order accurate.

3.5.5 L-stable, two-stage, second-order DIRK (2,2,2)

Another second order accurate scheme we considered is the following scheme based on the same L-stable, two-stage, second-order DIRK with the same γ as the previous scheme. The difference is that this scheme has been chosen to satisfy (40) in stead of (38), which would help in the case of an extremely stiff G and requires one less evaluation of $f(u_i)$ which makes it easier implementable.

0	0	0	0
γ	0	γ	0
1	0	$1-\gamma$	γ
	0	$1-\gamma$	γ

Implicit

0	0	0	0
γ	γ	0	0
1	δ	$1-\delta$	0
	δ	$1-\delta$	0

Explicit

With $\delta = 1 - \dfrac{1}{2\gamma}$. This scheme turned out to not differ that much in stability compared to the (2,3,2) scheme, in the case of the Kuramoto-Sivashinsky equation. Furthermore it required more time-steps to reach the same Lyapunov exponents, so we chose to use the (2,3,2) scheme instead.

3.5.6 Higher order schemes (3,4,3) and (4,4,3)

For completion we shall also include the higher order schemes given in [10]. Both these schemes are third order accurate and based on an L-stable DIRK

$(\mathbf{3}, \mathbf{4}, \mathbf{3})$

0	0	0	0	0
0.4358665215	0	0.4358665215	0	0
0.7179332608	0	0.2820667392	0.4358665215	0
1	0	1.208496649	-0.644363171	0.4358665215
	0	1.208496649	-0.644363171	0.4358665215

Implicit

0	0	0	0	0
0.4358665215	0.4358665215	0	0	0
0.7179332608	0.3212788860	0.3966543747	0	0
1	-0.105858296	0.5529291479	0.5529291479	0
	0	1.208496649	-0.644363171	0.4358665215

Explicit

$(\mathbf{4}, \mathbf{4}, \mathbf{3})$

0	0	0	0	0	0
$\frac{1}{2}$	0	$\frac{1}{2}$	0	0	0
$\frac{2}{3}$	0	$\frac{1}{6}$	$\frac{1}{2}$	0	0
$\frac{1}{2}$	0	$-\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0
1	0	$\frac{3}{2}$	$-\frac{3}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
	0	$\frac{3}{2}$	$-\frac{3}{2}$	$\frac{1}{2}$	$\frac{1}{2}$

Implicit

0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0
$\frac{2}{3}$	$\frac{11}{18}$	$\frac{1}{18}$	0	0	0
$\frac{1}{2}$	$\frac{5}{6}$	$-\frac{5}{6}$	$\frac{1}{2}$	0	0
1	$\frac{1}{4}$	$\frac{7}{4}$	$\frac{3}{4}$	$-\frac{7}{4}$	0
	$\frac{1}{4}$	$\frac{7}{4}$	$\frac{3}{4}$	$-\frac{7}{4}$	0

Explicit

The first scheme is part of the family that satisfies (40) while the second scheme satisfies (38). Even though these schemes are much more accurate than the second order schemes, for us it was not necessary to use any of them.

3.6 Stability

In this section we will present some graphs from [10]. These graphs will show the stability of the above mentioned IMEX RK schemes and help justify our choice of the (2,3,2) scheme for the Kuramoto-Sivashinsky equation.
Using the simple test equation

$$f = i\beta u, \quad g = \alpha u \tag{42}$$

With $\alpha, \beta \in \mathbb{R}$, $\alpha \leqslant 0, b > 0$ and i the imaginary number. The paper [10] devised some graphs regarding the relation between the time-step restriction and the fraction $\frac{\alpha}{\beta}$(Figures 1 and 2). They made these graphs for the IMEX RK schemes mentioned in the last section.
In these graphs we can see that when the function of g dominates over the function of f the IMEX RK schemes that belong to the family that satisfies

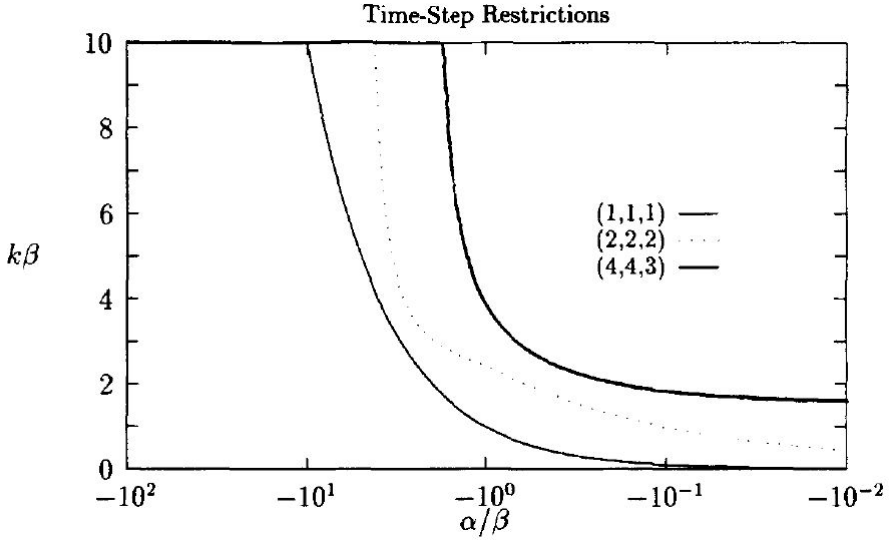


Figure 1: *From [10] the time-step restrictions for the family of IMEX RK schemes that satisfy (40), $k = h$ is the time-step size.*

(40) perform much better than the functions of the family that satisfies (38), but this family in turn has better overall stability. In the case of the Kuramoto-Sivashinsky equation the function of g is the sum of a second and fourth order spatial derivative and f is the multiplication of a first order derivative with the current value of u. For limited values of u both are of about the same order as u and $\frac{\alpha}{\beta} \approx 1$. Therefore we can use both families to integrate this problem, slightly favoring the family that satisfies (38). When comparing second order schemes from both families the scheme from the family that satisfies (38) performed the best so we chose to use the (2,3,2) scheme. We also considered using a higher order scheme like the (3,4,3) scheme but this proved to be unnecessary and therefore redundant.

3.7 Circulant matrix

In this section we will discuss how we used the discrete Fourier transform to more efficiently solve (34).
We can rewrite (34), by filling in (35) to read

$$(I - hG)K_i = G(u_{n-1} + h\sum_{j=1}^{i-1} a_{i,j}K_j + h\sum_{j=1}^{i} \hat{a}_{i+1,j}\hat{K}_j) \tag{43}$$

Where $I \in \mathbb{R}^m \times m$ is the identity matrix. In order to solve this problem we would ordinarily need to compute the inverse of $I - ha_{i,i}G$. This should, in order to maintain efficiency turn into a similar function as the one we composed for the Jacobian matrix in section 2.4. This is because, even though this inverse needs to be computed only once, computing a full matrix multiplication of $(I - ha_{i,i}G)^{-1}v$ with $(I - ha_{i,i}G)^{-1} \in \mathbb{R}^{m\times m}$ and $v \in \mathbb{R}^m$ would take $O(m^2)$ operations. For the Kuramoto-Sivashinsky equation and the (2,3,2) scheme the

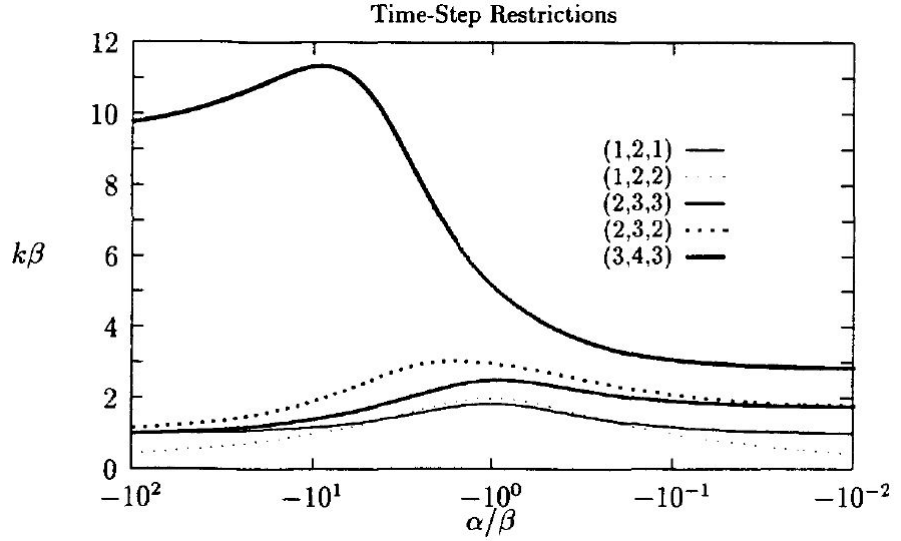


Figure 2: *From [10] the time-step restrictions for the family of IMEX RK schemes that satisfy (38), $k = h$ is the time-step size.*

matrix to be inverted would turn into

$$\begin{bmatrix} q & p & r & 0 & \cdots & r & p \\ p & q & p & r & 0 & \cdots & r \\ r & p & q & p & r & \ddots & 0 \\ 0 & r & p & q & \ddots & \ddots & \vdots \\ \vdots & 0 & r & \ddots & \ddots & p & r \\ r & \vdots & \ddots & \ddots & p & q & p \\ p & r & 0 & \cdots & r & p & q \end{bmatrix} \tag{44}$$

Where $q = \dfrac{6\gamma h}{\Delta x^4} - \dfrac{2\gamma h}{\Delta x^2} + 1$, $p = \dfrac{-4\gamma h}{\Delta x^4} + \dfrac{\gamma h}{\Delta x^2}$ and $r = \dfrac{\gamma h}{\Delta x^2}$. This matrix is a circulant matrix which means that we can rewrite the problem (43) in circulant form

$$c \star K_i = b \tag{45}$$

Where $b = G(u_{n-1} + h\sum_{j=1}^{i-1} a_{i,j}K_j + h\sum_{j=1}^{i} \hat{a}_{i+1,j}\hat{K}_j)$ and $c \in \mathbb{R}^m$ is the vector that composes the first column of the matrix $(I - ha_{i,i}G)$. To solve this equation we can use the discrete Fourier transform as shown in [7] to rewrite (45) to

$$\mathcal{F}_n(c \star K_i) = \mathcal{F}_n(c)\mathcal{F}_n(K_i) = \mathcal{F}_n(b) \tag{46}$$

Which makes it a component-wise multiplication so we can find the solution to (43) directly via

$$K_1 = \mathcal{F}_n^{-1}\left(\frac{\mathcal{F}_n(b)}{\mathcal{F}_n(c)}\right) \tag{47}$$

With the division a component-wise division. This combined with the fast Fourier transform algorithms from [17] implemented using [16] reduces this from $O(m^2)$ to $O(m \log m)$.

4 The test equations

In this section we will discuss the partial differential equations we analyzed with our program. We will discuss the properties these equations have and their physical relevance.

4.1 The damped forced Burgers-Hopf equation (L96 model)

The first equation we used is the damped forced Burgers-Hopf equation assuming periodic boundary conditions from [6].

$$u_t = uu_x - u + F \tag{48}$$

This is a special case of the Burgers-Hopf equation with a force term F and a damping term $-u$. We chose this equation because of the nontraditional finite difference discretization seen in section 5.1. This equation has a nontraditional discretization that is equal to the Lorenz 96 model for a specific Δx. The Lorenz 96 model is given by

$$\dot{u}_i = -u_{i-2}u_{i-1} + u_{i-1}u_{i+1} - u_i + F \tag{49}$$

Using the following approximations to u_{i-1}, u_{i+1} and u_{i-2}

$$\begin{cases} u_{i-1} = u - \Delta x u_x + O(\Delta x^2) \\ u_{i+1} = u + \Delta x u_x + \frac{\Delta x^2}{2} u_{xx} + O(\Delta x^3) \\ u_{i-2} = u - 2\Delta x u_x - 2\Delta x^2 u_{xx} + O(\Delta x^3) \end{cases} \tag{50}$$

We can rewrite (49) as

$$u_t = 3\Delta x u u_x + \frac{\Delta x^2}{2}(5uu_{xx} + 6u_x^2) + O(\Delta x^3) - u + F \tag{51}$$

Looking at this equation we can see that, when dividing the first two terms of (49) by $3\Delta x$ (as we will do for our discretization), we get something resembling (48). We will not be looking at the actual damped forced Burgers-Hopf equation, as we are using a nonstandard discretization. The Lorenz 96 model is known to have chaotic behavior for $F = 8$ and is a relatively simple model originally introduced by Edward Lorenz to study predictability in weather forecasting. The variables u_i are designed to describe some atmospheric quantity in m sectors of a latitude circle, with m the system size. It does not contain the real physics of the atmosphere, it is only correlated because it has external forcing (F), internal dissipation (u_i) and advection ($u_{i-2}u_{i-1}$ and $u_{i-1}u_{i+1}$) conserving the total energy. Since this model has been shown to have extensive chaos [14] we expect to see the same in our finite difference discretization of the Burgers-Hopf equation.

4.2 The stochastic Burgers equation

The second equation we used is the stochastic Burgers equation with periodic boundary conditions as seen in [13]

$$u_t = -uu_x + vu_{xx} + a\dot{W}(t, x) \tag{52}$$

Where a and v are positive scalars, with v the viscosity, and $\dot{W}(t,x)$ represents space-time white noise. We used a standard Gaussian distribution for this noise, with $\mu = 0$ and $\sigma = 1$. The original Burgers equation (in the case where $a = 0$) was introduced by J.M. Burgers as an "[...]extremely simplified model describing the interaction of dissipative and non-linear inertial terms in the motion of the fluid" ([15] section 1). Adding random forcing to this equation is also an idea from Burgers and is referred to as Burgers turbulence, because it is a way to model and investigate turbulence [13].

4.3 The Kuramoto-Sivashinsky equation

The third equation we used is the Kuramoto-Sivashinsky equation in derivative form.

$$u_t = -uu_x - u_{xx} - vu_{xxxx} \tag{53}$$

Also assuming periodic boundary conditions. This equation is characterized by a destabilizing diffusion (u_{xx}), a stabilizing dissipation (u_{xxxx}) and a coupling term (uu_x). v is a positive scalar denoting the viscosity of the system. The main use for this equation is the investigation of global attractors and inertial manifolds. "An inertial manifold is a finite-dimensional Lipschitz manifold which attracts all orbits exponentially, and is positively invariant under the flow" ([12] section 1). This means that if an equation has inertial manifolds we can compose a finite dimensional dynamic system, which contains all the asymptotic dynamics even if the original system itself has infinite dimension. Since it has been shown in [12] that the Kuramoto-Sivashinsky equation has inertial manifolds, this indicates that it has a finite number of positive Lyapunov exponents and finite Kaplan-Yorke dimension scaling only with period length and not with degrees of freedom, as all the asymptotic dynamics can be described by a finite dimensional system independent of the dimension of the discretization.

5 Numerical results

5.1 The implementations of the test equations

In this section we will discuss the discretizations we used as well as the direct functions for the multiplication of the Jacobian matrix as explained in section 2.4.

5.1.1 The damped forced Burgers-Hopf equation

Our discretization of the damped forced Burgers-Hopf equation from section 4.1 is as indicated in [6]

$$f(u_i) = \frac{u_{i+1} - u_{i-2}}{3\Delta x} u_{i-1} - u_i + F \tag{54}$$

This discretization is equal to (49) for $\Delta x = \frac{1}{3}$. To integrate this discretization we used the standard Runge-Kutta 4 (RK4) method as described in section 3.1.

Since there are no stiff terms in this discretization this method performed well within expectations.

$$(AV)_{i,j} = -V_{i-2,j}\frac{u_{i-1}}{3\Delta x} + V_{i-1,j}\frac{u_{i+1}-u_{i-2}}{3\Delta x} - V_{i,j} + V_{i+1,j}\frac{u_{i-1}}{3\Delta x} \tag{55}$$

We also tried to use a more conventional finite difference discretization of the same system (one that is not equivalent to the Lorenz 96 model for any Δx) to investigate how a difference in discretization might influence the stable directions. Following is the alternative discretization we used.

$$f(u_i) = \frac{u_{i+1}-u_{i-1}}{2\Delta x}u_i - u_i + F \tag{56}$$

With Jacobian matrix multiplication function.

$$(AV)_{i,j} = -V_{i-1,j}\frac{u_i}{2\Delta x} + V_{i,j}\frac{u_{i+1}-u_{i-1}}{2\Delta x} - V_{i,j} + V_{i+1,j}\frac{u_i}{2\Delta x} \tag{57}$$

This discretization, however, did not present chaos and grew very quickly to the stable solution of $u_i = F$. We have therefore excluded this discretization from the final results.

5.1.2 The stochastic Burgers equation

Since the stochastic Burgers equation has a stiff term (u_{xx}) we considered using an IMEX RK method for the integration, but upon testing this equation with the standard RK4 method we achieved sufficiently efficient and accurate results. Therefore we did not have to split this equation in two parts and the discretization simply reads

$$f(u_i) = -u_i\frac{u_{i+1}-u_{i-1}}{2\Delta x} + v\frac{u_{i+1}-2u_i+u_{i-1}}{\Delta x^2} + a\sqrt{-2\log(r_1)}\sin(2\pi r_2) \tag{58}$$

With r_1 and r_2 machine generated random doubles on the interval $[0,1]$. The Jacobian matrix multiplication function for this discretization is

$$(AV)_{i,j} = -V_{i-1,j}\frac{u_i}{2\Delta x} + V_{i,j}\frac{u_{i+1}-u_{i-1}}{2\Delta x} + V_{i+1,j}\frac{u_i}{2\Delta x} + v\frac{V_{i-1,j}-2V_{i,j}+V_{i+1,j}}{\Delta x^2} \tag{59}$$

5.1.3 The Kuramoto-Sivashinsky equation

For this equation we needed the IMEX RK methods discussed in section 3 because both the second and fourth order derivative terms are stiff. The fourth order derivative being so stiff that in order to get a stable approximation by using only a explicit integration scheme, we would require a time-step so small that we could not reasonably integrate to large t. By putting both of these stiff terms in G (from (31) we managed to ensure more stable integration for larger time-steps and smaller spatial-steps. This lead to two discretized equations. One for f and one for Gu_i.

$$\begin{cases} f(u_i) = -u_i\dfrac{u_{i+1}-u_{i-1}}{2\Delta x} \\ Gu_i = -\dfrac{u_{i+1}-2u_i+u_{i-1}}{\Delta x^2} - v\dfrac{u_{i+2}-4u_{i+1}+6u_i-4u_{i-1}+u_{i-2}}{\Delta x^4} \end{cases} \tag{60}$$

Since this discretization contains two equations and both are used in a different integration scheme, we have to generate two Jacobian matrix multiplication functions as well.

$$\begin{cases} (AV)_{i,j} = -V_{i-1,j}\dfrac{u_i}{2\Delta x} + V_{i,j}\dfrac{u_{i+1}-u_{i-1}}{2\Delta x} + V_{i+1,j}\dfrac{u_i}{2\Delta x} \\ (GV)_{i,j} = -v\dfrac{V_{i-2,j}-4V_{i-1,j}+6V_{i,j}-4V_{i+1,j}+V_{i+2,j}}{\Delta x^4} - \dfrac{V_{i-1,j}-2V_{i,j}+V_{i+1,j}}{\Delta x^2} \end{cases} \tag{61}$$

Where A is the Jacobian matrix of f.

5.2 Setup of the numerical approximations

In this section we will discuss what initial conditions we used for the test equations and what values we used for e.g. the time-step and period lengths.
Since all of our test equations assumed periodic boundary conditions we decided to use the following periodic initial condition for each equation.

$$u_i = \cos\left(\frac{2\pi\Delta x i}{L}\right)\left(1-\sin\left(\frac{2\pi\Delta x i}{L}\right)\right) \tag{62}$$

With L the period length and Δx the step size. Taking $m = \frac{L}{\Delta x}$ the dimension of the system. For the Burgers-Hopf and the stochastic Burgers equations we chose the first $m = 50$, but for the Kuramoto-Sivashinsky equation we chose the first $m = 150$. Next we picked the number of Lyapunov exponents to be calculated n. The value we picked for n was a value we had previously seen to be larger than the number of Lyapunov exponents needed for the Kaplan-Yorke dimension for the first m. After the program had approximated the Lyapunov exponents we let it decrease m by one and picked n to be 5 larger than the number of Lyapunov exponents needed for the previous m and repeated the program. This ensured we found all the Lyapunov exponents needed for all m. We continued this until we had 40 results or until the system became unstable. We experimented some with the time-step h as well as with the integration length t_{max}. We found $h = \frac{1}{64}$ to give accurate and stable results without being too time-consuming for all three equations. $t_{max} = 1000$ gave the same results as $t_{max} = 10000$ for numerous test-cases and was therefore accepted as a correct value. Finally we experimented some with different L for both systems. For the Kuramoto-Sivashinsky equation we used $L = 22, 50, 100$ and for the Burgers-Hopf equation we used $L = 5, 9, 15$ all of these results are included in section 5.3. For the stochastic Burgers equation we used $L = 25, 50, 100$. For the Burgers-Hopf equation we also tried some different values for the force term F. We used $F = 5, 8, 10$. For both the Kuramoto-Sivashinsky equation and the stochastic Burgers equation we experimented some with different values of v. We used $v = 1, 2$. For the stochastic Burgers equation we used $a = 0.1$ for the scaling in the white noise.

5.3 Results

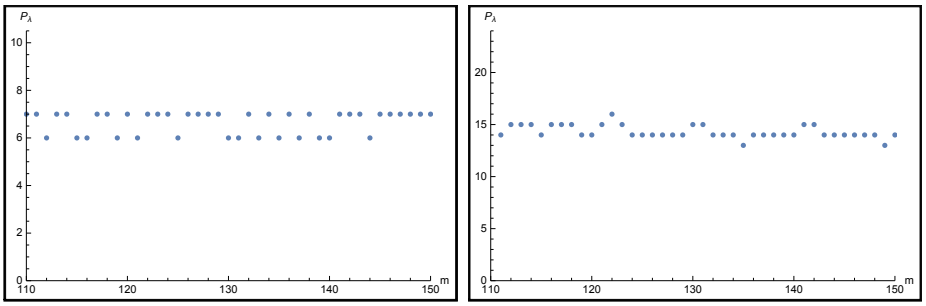
In this section we will show and discuss the graphs we generated with the data obtained from our program.
We can see from figures 3, 4, 5 and 6 that for the Kuramoto-Sivashinsky equation both the Kaplan-York dimension and the dimension of the unstable space

is invariant under the resolution. We do however see that they both change with the period length L and the viscosity v. In figures 3 and 4 it is visible that, when L is doubled, both the dimension of the unstable space and the Kaplan-Yorke dimension double as well. From figures 5 and 6 we can see that if the viscosity is doubled the Kaplan-Yorke dimension and the dimension of the unstable space decreases.
For the damped forced Burgers-Hopf equation we expected extensive chaos based on the resemblances between our nonstandard discretization and the Lorenz 96 model. Furthermore we expected to see almost purely extensive chaos for $F \geqslant 8$ and mostly extensive chaos with certain values for m where there was periodic behavior for $F = 5$ (based on [14]). Contrarily we found the behavior that for the real Lorenz 96 model was only found for $F = 5$ (in [14]) for all values of F, and only for a small enough resolution. As is visible in figures 7, 8, 9 and 10 for a certain value of $m = m_0$ the fluctuating chaotic/periodic behavior transforms to purely extensive chaos. The value m_0 where this happens depends on L and F. The equation for m_0 seems to be (based on the graphs)

$$m_0 \approx \frac{15L}{F} \tag{63}$$

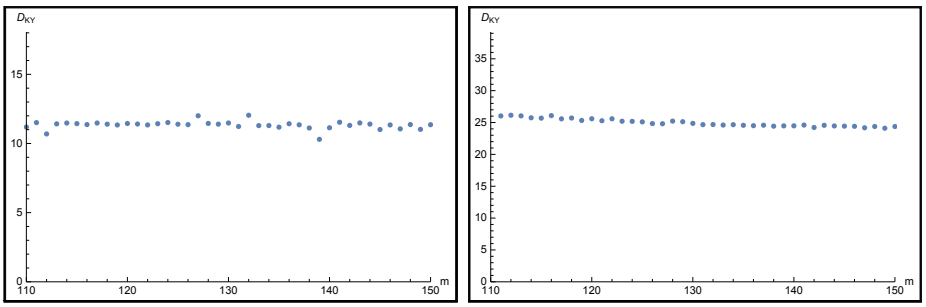
For the stochastic Burgers equation we found purely extensive chaos that only scaled with m. As illustrated in figures 11, 12, 13 and 14 both the Kaplan-Yorke dimension and the dimension of the unstable space did not change when either the viscosity or the period length changed (the plotted linear model in all of these graphs is $\lambda_p = 0.42m$ for the dimension of the unstable space and $D_{KY} = 0.79m$ for the Kaplan-Yorke dimension). The maximum Lyapunov exponent however did change in these cases, as seen in figures 15 and 16. By halving the period length the maximum Lyapunov exponent increased fourfold and by doubling the viscosity the maximum Lyapunov exponent also doubled.

5.3.1 Graphs



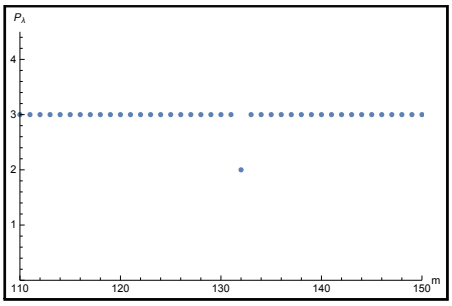
(a) $L = 50$ and $v = 1$ (b) $L = 100$ and $v = 1$

Figure 3: *Dimension of the unstable space of the Kuramoto-Sivashinsky equation for different period lengths.*

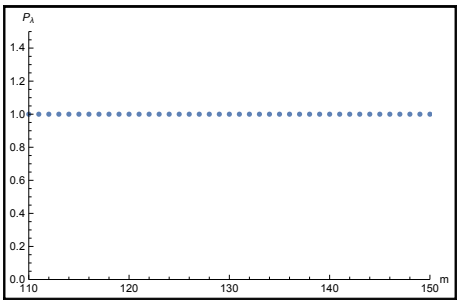


(a) $L = 50$, $v = 1$ (b) $L = 100$, $v = 1$

Figure 4: *Kaplan-Yorke dimension of the Kuramoto-Sivashinsky equation for different period lengths.*

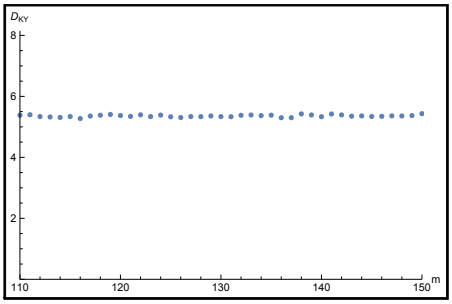


(a) $L = 22$, $v = 1$

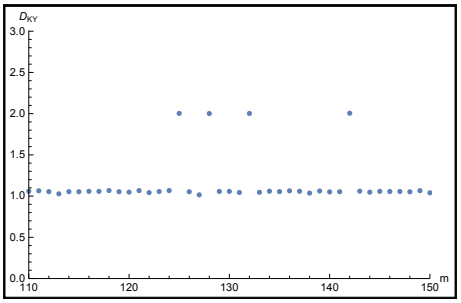


(b) $L = 22$, $v = 2$

Figure 5: *Dimension of the unstable space of the Kuramoto-Sivashinsky equation for different viscosities.*

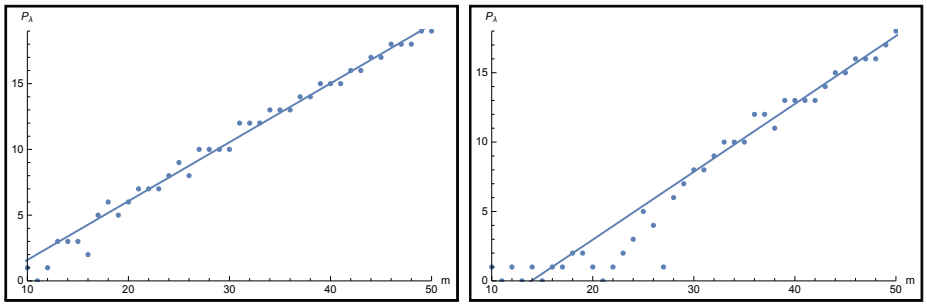


(a) $L = 22$, $v = 1$



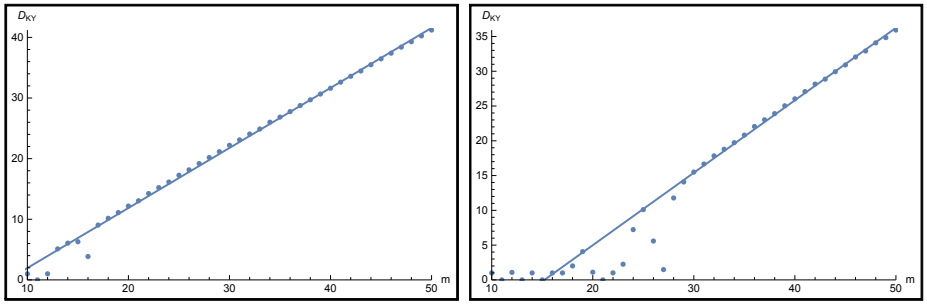
(b) $L = 22$, $v = 2$

Figure 6: *Kaplan-Yorke dimension of the Kuramoto-Sivashinsky equation for different viscosities.*



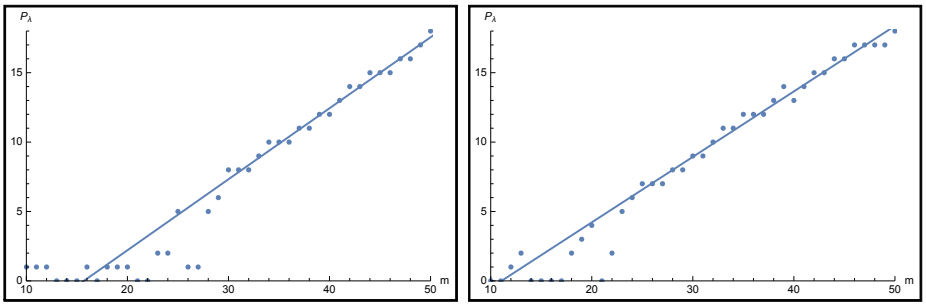
(a) $L = 5$ and $F = 5$ (b) $L = 9$ and $F = 5$

Figure 7: *Dimension of the unstable space of the damped forced Burgers-Hopf equation for different period lengths. The blue line is a linear model based on the values where purely extensive chaos occurs.*



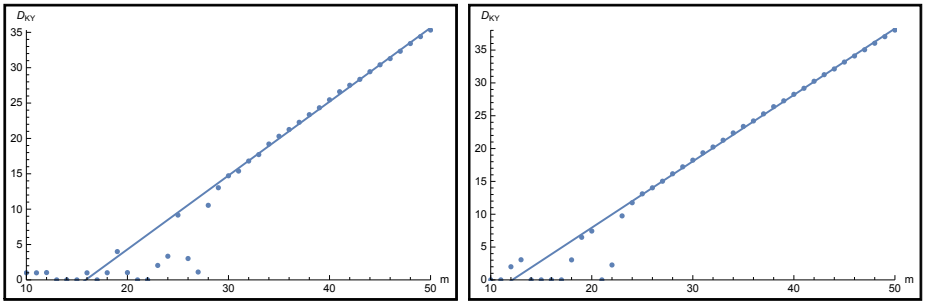
(a) $L = 5$, $F = 5$ (b) $L = 9$, $F = 5$

Figure 8: *Kaplan-Yorke dimension of the damped forced Burgers-Hopf equation for different period lengths. The blue line is a linear model based on the values where purely extensive chaos occurs.*



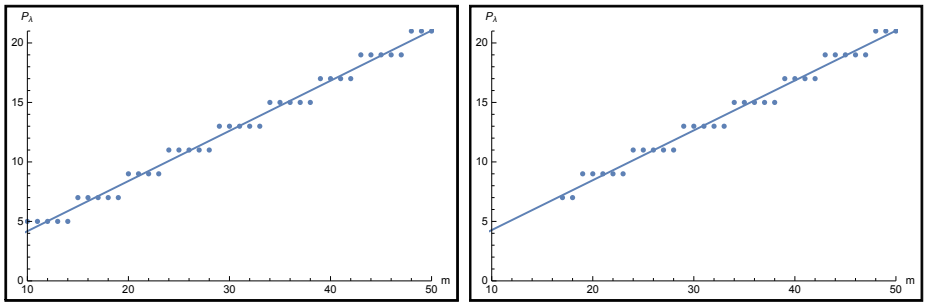
(a) $L = 15$, $F = 8$ (b) $L = 15$, $F = 10$

Figure 9: *Dimension of the unstable space of the damped forced Burgers-Hopf equation for different forcing. The blue line is a linear model based on the values where purely extensive chaos occurs.*



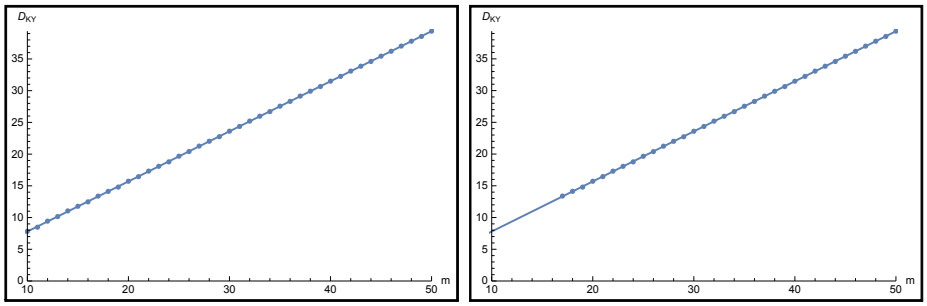
(a) $L = 15$, $F = 8$ (b) $L = 15$, $F = 10$

Figure 10: *Kaplan-Yorke dimension of the damped forced Burgers-Hopf equation for different forcing. The blue line is a linear model based on the values where purely extensive chaos occurs.*



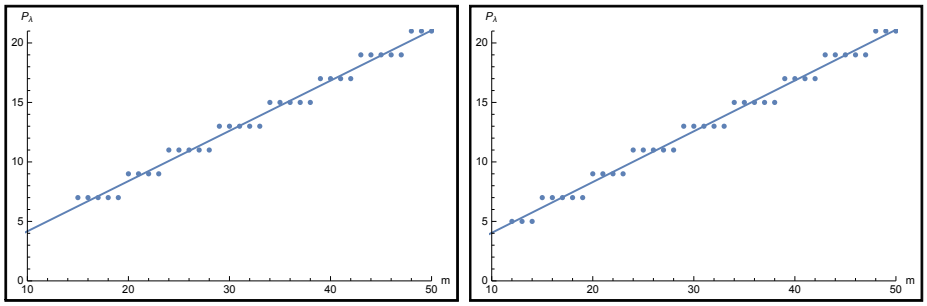
(a) $L = 25$, $v = 2$ (b) $L = 100$, $v = 2$

Figure 11: *Dimension of the unstable space of the stochastic Burgers equation for different period lengths. The blue line is a linear model of the values.*



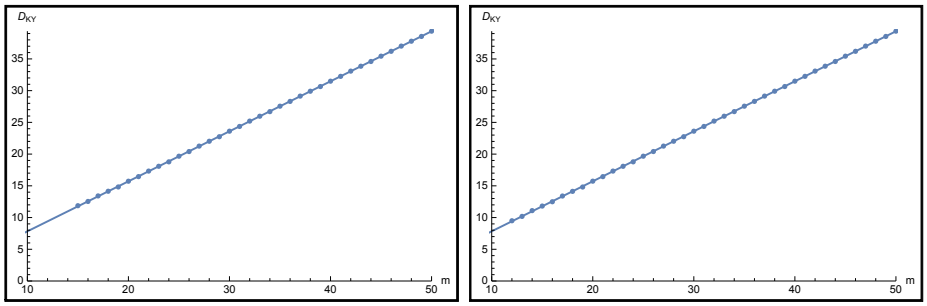
(a) $L = 25$, $v = 2$ (b) $L = 100$, $v = 2$

Figure 12: *Kaplan-Yorke dimension of the stochastic Burgers equation for different period lengths. The blue line is a linear model of the values.*



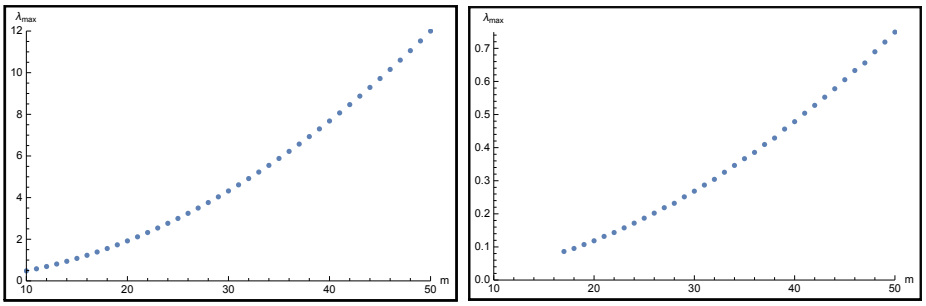
(a) $L = 50$, $v = 1$ (b) $L = 50$, $v = 2$

Figure 13: *Dimension of the unstable space of the stochastic Burgers equation for different viscosities. The blue line is a linear model of the values.*



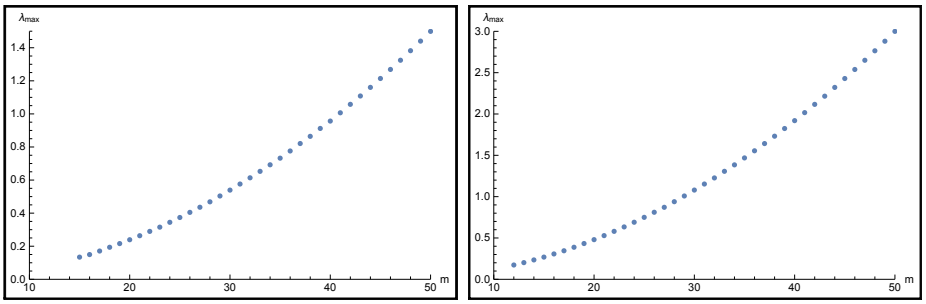
(a) $L = 50$, $v = 1$ (b) $L = 50$, $v = 2$

Figure 14: *Kaplan-Yorke dimension of the stochastic Burgers equation for different viscosities. The blue line is a linear model of the values.*



(a) $L = 25$, $v = 2$ (b) $L = 100$, $v = 2$

Figure 15: *Maximum Lyapunov exponent of the stochastic Burgers equation for different period lengths.*



(a) $L = 50$, $v = 1$ (b) $L = 50$, $v = 2$

Figure 16: *Maximum Lyapunov exponent of the stochastic Burgers equation for different viscosities.*

References

[1] C. Gonzales-Tokman and B.R. Hunt, "Ensemble data assimilation for hyperbolic systems" (2013)

[2] A. Carrassi et al., "Data assimilation as a nonlinear dynamical systems problem: Stability and convergence of the prediction-assimilation system" (2008)

[3] L. Dieci, M.S. Jolly and E.S Van Vleck, "Numerical Techniques for Approximating Lyapunov Exponents and Their Implementation"

[4] L. Dieci AND E.S. van Vleck, "Lyapunov and Sacker-Sell Spectral Intervals"

[5] A. Lyapunov, "Problem general de la stabilite du mouvement", Int. J. Control 53 (1992), pp. 531–773.

[6] A. Majda and X. Wang, "Nonlinear Dynamics and Statistical Theories for Basic Geophysical Flows", page 239

[7] P.J. Davis, "Circulant Matrices" (1994)

[8] P. Fredrickson et al., "The Liapunov Dimension of Strange Attractors" (1982)

[9] E. Ott, "Chaos in dynamical systems" (1993)

[10] U.M. Ascher, S.J. Ruuth and R.J. Spiteri, "Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations" (1997)

[11] E.N. Lorenz, "Predictability-a problem partly solved" (1996)

[12] J.C. Robinson, "Inertial manifolds for the Kuramoto-Sivashinsky equation" (1993)

[13] S. Kaligotla and S.V. Lototsky, "Wick product in the stochastic burgers equation: a curse or a cure?"

[14] A. Karimi and M.R. Paul, "Extensive Chaos in the Lorenz-96 Model" (2010)

[15] L. Bertini, N. Cancrini1 and G. Jona-Lasinio, "The Stochastic Burgers Equation" (1993)

[16] https://github.com/tszalay/FFTWSharp

[17] http://www.fftw.org/