A wavelet-based solution of the Kuramoto-Sivashinsky equation

by

Gloria B. Ruiz Valle

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER IN SCIENCES
in
APPLIED MATHEMATICS

UNIVERSITY OF PUERTO RICO
MAYAGUEZ CAMPUS
2007

Approved by:

_____	_____
Krzysztof Rozga, Ph.D. Member, Graduate Committee	Date
_____	_____
Uroyoan Walker, Ph.D. Member, Graduate Committee	Date
_____	_____
Alexander Urintsev, Ph.D. President, Graduate Committee	Date
_____	_____
Nelson Cardona Martínez, Ph. D. Representative Graduate School	Date
_____	_____
Julio Quintana, Ph.D. Department Director	Date

Abstract

The scope of this thesis investigation is to obtain an approximate numerical solution to the nonlinear one-dimensional Kuramoto-Sivashinsky partial differential equation. The Gaussian wave has been successfully applied to convert this equation by means of a wavelet transform into a nonlinear integro-differential equation for the transformant. A Cauchy problem was formulated. The wavelet coefficients were expanded by means of basis functions based on the classical Laguerre and Hermite orthogonal polynomials, and then the Galerkin method was used to get a system of ordinary differential equations that was solved numerically with the Mathematica system.

Resumen

Esta tesis describe cómo obtener una solución numérica para la ecuación diferencial parcial no lineal de Kuramoto–Sivashinsky en una dimensión. La ondita (wavelet) “Gaussian wave” ha sido aplicada con éxito para convertir esta ecuación en una ecuación no lineal integro-diferencial para el transformante a través de una transformación de ondita. Se formuló un problema de Cauchy. Los coeficientes de ondita fueron expandidos a través de funciones bases fundamentadas en los polinomios ortogonales clásicos de Hermite y de Laguerre y entonces el método de Galerkin se usó para obtener un sistema de ecuaciones diferenciales ordinarias que fue resuelto numéricamente utilizando el sistema Mathematica.

Dedicatory

To my parents: Ismenio Ruiz Cedeño and Aurora Valle Rivera.

Acknowledgements

During the development of my graduate studies at the University of Puerto Rico several persons and institutions collaborated directly and indirectly in my research. Their support have being of great value. That is why I wish to dedicate this section to recognize their support.

I would like to start expressing a sincere acknowledgement to my advisor, Dr. Alexander Urintsev for his constant support to my thesis work. I received motivation; encouragement and support from him during my graduated studies. I learned from him how to write Mathematica programs. I want to recognize my father for helping me with the documents translation from Spanish into English and my mother for taking care of my children during my studies and investigation.

Table of Contents

Table List

Figure List

Appendix List

1. Introduction

This thesis investigation analyzes how to solve the nonlinear one-dimensional Kuramoto-Sivashinsky equation by means of a continuous wavelet transform. Some progress has been made in this regard in recent years, including constructing a long wavelength, long time theory of this differential equation [Xiong et al., 2006]. Assuming strong surface tension effects and long waves, a weakly nonlinear expansion of the Navier-Stokes equations and free surface boundary conditions yields this partial differential equation.

The formula for the Kuramoto-Sivashinsky equation [Trevelyan et al., 2005] is the following one:

$$u_t + u_{xx} + u_{xxxx} + 4uu_x = 0. \tag{1.1}$$

In equation (1.1) u, x, and t are appropriately dimensionless film thickness, streamwise coordinate, and time respectively. This model retains the fundamental elements of any nonlinear process that involves wave evolution: a simple nonlinearity $-uu_x$, instability and energy production $-u_{xx}$, and stability and energy dissipation $-u_{xxxx}$.

The nonlinearity of uu_x is due to the movement associated with mean flow for thin films. The nonlinearity arises from the nonlinear correction to the phase speed; a nonlinear kinematic effect that captures how large waves move faster than smaller ones. This is a weakly nonlinear process though rather different in their physical nature, such as: wavy liquid layer falling by its gravity down a vertical plane; hydrodynamic instability of flame fronts, concentrational waves in Belousov-Zhabotinsky reactions [Eremenko, 2005] occurring far from the thermodynamic equilibrium point, thermocapillary convection in horizontal thin layers, and wavy processes in plasma [Demekhin et al., 1993]. The last two applications are used in laboratories related to medicine and environmental analysis.

1.1 Motivation

The Kuramoto-Sivashinsky equation is the simplest one to characterize nonlinear waves in media with energy supply and dissipation [Demekhin et al., 1993]. The main objective of this investigation is to study the solution of the equation by means of a numerical technique based on wavelets. Wavelets are useful for this purpose because they are smoother than the original function. The solution of Kuramoto-Sivashinsky equation is represented in a graphic form.

Mexican Hat [Daubechies, 1992] is a well known continuous wavelet used with success to model equations of this type [Lewalle, 1993]. After integrating a Mexican Hat wavelet, a new one results. This wavelet is known in literature but it is not used as often as the Mexican Hat is. It is proportional to the Gaussian first derivative. The wavelet is named the Gaussian wave [Addison, 2005]. It is the one used in this thesis investigation.

1.2 Literature Review

There are some scientists who previously investigated the use of wavelets to solve differential equations. The first one was V. Perrier in 1989 [Farge et al., 1996]. She introduced the technique of solving partial differential equations using wavelet bases. In 1995, V. Perrier and C. Basdevant [Monasse et al., 1995] presented a family of new wavelets which can correctly measure any power law energy spectrum. Further, in 2005 she used wavelets for the numerical representation of the solution of the three-dimensional Navier-Stokes equations.

J. Elezgaray and collaborators used Perrier-Basdevant wavelets [Dahmen et al., 1997] in 1996 to investigate the ability of local models of the one-dimensional Kuramoto-Sivashinsky equation to reproduce coherent events typical of the solution of the same equation on long intervals. They also showed that an effective equation preserving the statistics of the large scales of this equation can be obtained from a coarse graining procedure based on its wavelet decomposition. Before this investigation [Elezgaray et al., 1996], they studied the Kuramoto-Sivashinsky equation from a statistical point of view. The investigators showed that the statistical properties of the large

scales of this equation in the extended system limit can be understood in terms of the dynamical behavior of the same equation in a small finite domain.

C. Meneveau proposed the dual spectra and mixed energy cascade algorithm to model turbulence in the wavelet representation [Meneveau, 1991]. This algorithm is efficient for computer implementation because it reduces the number of operations for each step. He also developed a new class of sub grid closures for large eddy simulation and applied his results to both steady and freely isotropic turbulence. This same year, he showed that the wavelet transformation can be applied to the Navier-Stokes equations.

E. A. Demekhin studied the instability and nonlinear waves in thin films of viscous liquid. The study was based on the Kuramoto-Sivashinsky equation. He performed a quantitative comparison between the theory developed and the experimental results. He wrote a book with the collaboration of H. - C. Chang where he described the complex wave evolution from nearly harmonic waves to complex spatio-temporal patterns involving solitary waves downstream [Demekhin et al., 1993].

In 1992 J. Lewalle [Lewalle, 1993] introduced the use of Hermitian wavelets for solving differential equations. These wavelets are obtained recursively as derivatives of the Gaussian bell curve. In addition, he explained the relation between the fundamental laws and the emergence of large-scale coherent eddies from differential interactions in a random background [Lewalle, 2002a]. This same year, he also combined the recursive filtering of renormalization theory and the powerful analytical tool provided by wavelet transforms to express solutions for Navier-Stokes equations. Dr. Lewalle also [Lewalle, 2002b] showed that the filtered flexion is identical to the Mexican-hat wavelet transform of velocity. He included the formulation of nonlinear terms in the framework of complex systems dynamics and emergence of structures.

M. Farge and her collaborators showed the efficiency of wavelets for the numerical simulation of turbulence, and suggested novel methods for modeling the flow in two dimensions [Farge et al., 1995]. In 1998 she extended her results for the three-dimensional case and compared different wavelet type approaches applied to instantaneous velocity, vorticity, and

pressure fields. In addition, the team of investigators led by Dr. Farge applied orthogonal wavelets to the Coherent Vortex Simulation filtering of turbulent mixing layers and compared it with the method of Proper Orthogonal Decomposition. Both methods were applied to data of three-dimensional forced homogeneous isotropic turbulence. She also showed this same year that, using a wavelet basis, a turbulent signal can be decomposed into coherent and incoherent components which are orthogonal and whose properties can thus be studied independently [Schneider et al., 2000].

G. Beylkin et al. introduced a class of numerical algorithms based on the fast wavelet transform [Beylkin et al., 1996]. They applied an adaptive pseudo wavelet approach for solving nonlinear partial differential equations with initial condition and periodic boundary condition numerically [Beylkin et al., 1997]. In addition, [Beylkin et al., 2002] they explained how to work with nonlinear terms in differential equations. Beylkin's team also developed numerical examples for the Heat equation, Burgers equation, reaction-diffusion equations, and Korteweg-de Vries equation.

M. Griebel et al. described an algorithm using the Petrov-Galerkin scheme to solve the Navier-Stokes equations [Griebel et al., 2000]. Their approach can be adapted to evaluate nonlinear terms of other differential equations. Most recently, X. Zhou and Y. He used the Galerkin's wavelet scheme to solve the two-dimensional stationary Navier-Stokes equations. Their scheme was based on constructing the scaling generator of the wavelets. They enumerated ideas to improve the approximation's accuracy.

A. N. Fedorova and her collaborators applied a technique based on the methods of local nonlinear harmonic analysis to nonlinear wave dynamics problems such as the Kuramoto-Sivashinsky equation [Fedorova et al., 2002]. S. Saprykin et al. presented the most up to date developments for the problem of a falling film and outlined issues which have not yet been resolved [Saprykin et al., 2005]. They also used the generalized Kuramoto-Sivashinsky equation in two dimensions to model the stationary solitary pulses in a falling film and obtained an analytical estimate for the speed of those waves. Finally, G. Xiong et al. presented the results of a wavelet based approach to the study of the chaotic dynamics [Xiong et al., 2006] of a one-

dimensional model of the Kuramoto-Sivashinsky equation and extended the results to a similar model for the Nikolaevskii equation.

1.3 Summary of the following chapters

First, in Chapter 2 the necessary background theory is developed. Chapter 3 deals with the materials and methods. These methods were applied to a known function and then in the case of the Kuramoto-Sivashinsky equation. The third chapter presents the results in a table and graphic form and explains them in detail. Discussion of the results and conclusions are presented in Chapter 5. In the Appendix, there are all of the Mathematica programs written to obtain the results presented in this work.

2. Theoretical background

This chapter has five sections outlined as follows. The first section describes the applications and history of the Kuramoto-Sivashinsky equation. The second one is about wavelet transforms. In this part the wavelet and Fourier transform were compared and details about continuous wavelets were enumerated. The third part is about basis functions based on Hermite and Laguerre classical orthogonal polynomials. The fourth section describes the integro-differential equation constructed. The fifth section deals with the Galerkin method. This last section is of crucial importance for understanding the formulation of the problem.

2.1 The origin of the Kuramoto-Sivashinsky equation

The Kuramoto-Sivashinsky equation [Demekhin et al., 1993] serves to represent one class of pattern formation equation and is a good model of bifurcation and chaos. It was first derived by Kuramoto in 1976 [Liu et al., 2001] for the study of phase turbulence in the Belousov-Zhabotinsky reaction. This is a reaction formed by acidified bromate, ceric salt, and citric acid or malonic acid [Peterson, 1998]. The novel thing about this reaction is that, instead of forming only one, as is usual in chemistry, there are a series of oscillating reactions. This phenomenon may explain the chemical reactions with heart rate variations [Addison, 2005] and other biochemical body reactions [Eremenko, 2005].

An extension to two or more spatial dimensions was then given by Sivashinsky in 1980 for the study of the propagation of a flame front for the case of mild combustion, describing the combined influence of diffusion and thermal conduction of the gas on stability of a plane flame front [Liu et al., 2001]. The pioneering works with the problem of large-time behavior of the Kuramoto-Sivashinsky equation appears to be due to Foias et al. in 1988 and Nicolaenko et al. since 1986, who described the global attractors and inertial manifolds of this nonlinear fourth order dissipative equation. Since then, there has been progress on analysis of the equation by scientifics such as Demekhin, Elezgaray, and Saprykin.

This equation is useful to model solitary pulses in a falling thin film [Saprykin et al., 2005]. Previous numerical results indicate that this equation admits three classes of non-periodic traveling wave solutions, namely: regular shocks, oscillatory shocks, and solitary waves. Under certain conditions, [Elezgaray et al., 1996] the growing waves can link a nearly antisymmetric oscillatory shock with a radiant regular shock to form a solitary wave.

2.2 Wavelet transform

Wavelets have been developed to analyze frequency components according to a scale. A wavelet is a localized function $\Psi(x)$ of mean zero [Starck et al., 1998]. The corresponding integral is the following:

$$\int_{-\infty}^{\infty} \Psi(x)\, dx = 0. \tag{2.1}$$

Wavelets [Beylkin, 1996] have been applied with considerable success to equations which model turbulence. A wavelet transform is a generalization of the Fourier transform in the sense that it gives information about the behavior of a function in the frequency and time domains. Wavelet transforms are also computationally efficient, and part of the reason for this is that the wavelet function is often of compact support, i.e. defined on a limited and finite domain.

When the wavelet transform is carried out on successive time intervals, the contributions of the various frequencies to the signal can be followed in time [Chui, 1992]. Therefore a wavelet is not only a function of frequency, but also of time. A wavelet transform is usually expressed in terms of the scale of the analyzing wavelet, rather than referring to its frequency. The scale can be understood to be proportional to the inverse of the frequency.

2.2.1 Comparison with the Fourier transform

Wavelet analysis [Meyer, 1992] can be seen as a generalization of the Fourier analysis and in many cases permits a similar interpretation, but amplifies it by adding time resolution in a more fundamental way. Wavelet transforms provide a simpler and more efficient way to analyze those

functions and distributions [Chui, 1992; Daubechies, 1992] that have been studied by means of Fourier series and integrals. Fourier algebraic formalism is used to construct the wavelets. Once the wavelets have been constructed, they perform well in situations where Fourier series and integrals involve subtle mathematics or heavy numerical calculations.

Fourier and wavelet analyses [Meyer, 92] are thus complementary rather than competing. Wavelet analysis takes as its analyzing basis functions wavelets, i.e., oscillating functions that decay rapidly with time, instead of sines and cosines that do not have such decay [Chui, 1992]. Thus, a wavelet transform at a given point is similar to a Fourier transform in the sense that it exhibits the contribution of the different frequencies to a signal, but due to the decay of the wavelet this information only pertains to a certain short time interval.

2.2.2 The continuous wavelet transform

The Fourier transform of any wavelet is given by the following formula:

$$\hat{\Psi}(\alpha) = \int_{-\infty}^{\infty} \Psi(x) e^{-i\alpha x} dx . \tag{2.2}$$

Not all the functions can be a mother wavelet function. There is a condition that should be satisfied; it is named the admissibility condition for the wavelet. It is the following:

$$\int_0^{\infty} \frac{|\hat{\Psi}(\alpha)|^2}{\alpha} d\alpha < \infty ; \tag{2.3}$$

where α is a real number and $\hat{\Psi}(\alpha)$ is the Fourier transform of the wavelet $\Psi(x)$. The admissibility condition is required, so that the original function can be reconstructed from its wavelet transform. This condition implies that $\hat{\Psi}(0) = 0$, so that a wavelet cannot contain a constant term [Starck et al., 1998]. The result of integrating the admissibility condition yields the admissibility constant for the continuous wavelet. It is denoted by C_{Ψ} as follows:

$$C_{\Psi} = \int_0^{\infty} \frac{|\hat{\Psi}(\alpha)|^2}{\alpha} d\alpha . \tag{2.4}$$

One difficult part of this work is to find a wavelet suitable for symbolic calculations in our case. For this reason a continuous wavelet transform was chosen as a model wavelet.

The Morlet-Grossmann formula of the continuous wavelet transform for the one-dimensional case [Starck et al., 1998; Van Milligen et al., 1995], applied to the function $u(x,t) \in L^2(\mathbb{R})$ is given by the formula

$$W_{k,b}(t) = \sqrt{k} \int_{-\infty}^{\infty} u(x,t)\, \overline{\Psi_{k,b}(x)}\, dx, \tag{2.5}$$

where $k > 0$ is the scale parameter similar to a wave number, b is the position parameter, $\Psi(x)$ is the analyzing (mother) wavelet, and the bar "$\bar{}$" on the top of $\Psi_{k,b}(x) = \Psi(k(x-b))$ denotes a complex conjugate. The scaled and shifted variants of that function are called daughter wavelets. They are of the form:

$$\Psi_{k,b}(x) = \Psi(k(x-b)). \tag{2.6}$$

The following properties [Starck et al., 1998] characterize continuous wavelet transforms (CWT):

1. CWT is a linear transformation:

$$f(x) = a_1 f_1(x) + a_2 f_2(x) \Rightarrow W_{k,b}[f] = a_1 W_{k,b}[f_1] + a_2 W_{k,b}[f_2] \tag{2.7}$$

2. If the function is translated or dilated, then its CWT can be easily found:

$$f_0(x) = f(x - x_0) \Rightarrow W_{k,b}[f_0] = W_{k,b-x_0}[f] \tag{2.8}$$

$$f_s(x) = f(sx) \Rightarrow W_{k,b}[f_s] = \sqrt{s}\, W_{sk,sb}[f].$$

Continuous wavelets are effective in the analysis of partial differential equations, particularly in the absence of boundaries [Lewalle, 2002a]. For the continuous wavelet transform, the wavelets are not orthogonal and the wavelet transform contains some redundant information. There is no analytical penalty for the redundancy, as noted by I. Daubechies et al. in the article entitled "*Painless non-orthogonal expansions"* [Lewalle, 2002b]. The corresponding inverse

wavelet transform [Starck et al., 1998; Van Milligen et al., 1995; Lewalle, 2002a] can be calculated by means of the formula:

$$u(x,t)=\frac{1}{C_{\Psi}}\int_{k=0}^{\infty}\int_{-\infty}^{\infty} W_{k,b}(t)\,\Psi_{k,b}(x)\sqrt{k}\,db\,dk\,. \tag{2.9}$$

With the formula above, the original function *u(x, t)* can be restored if we know its wavelet transform, $W_{k,b}(t)$.

When continuous wavelets are used in integral transforms, they meet the requirements of the existence of an inverse transform (no loss of information) and of Parserval's theorem about $L^2(\mathbb{R})$-space normalization [Lewalle, 2002b]. In Lebesgue theory [Chui, 1992], $L^2(\mathbb{R})$ denotes the collection of all the measurable functions f defined in $\mathbb{R}$ with:

$$\int_{-\infty}^{\infty}|f(x)|^2\,dx<\infty\,. \tag{2.10}$$

The wavelet is normalized, so the following condition is satisfied:

$$\int_{-\infty}^{\infty}[\Psi(x)]^2\,dx=1. \tag{2.11}$$

The corresponding inner product for this space is:

$$<f,g>=\int_{-\infty}^{\infty}f(x)g(x)\,dx. \tag{2.12}$$

All the derivatives of the Gaussian function $e^{-\frac{x^2}{2}}$ may be employed as continuous wavelets [Addison, 2005]. The family of the derivatives of the Gaussian function has the name of Hermitian wavelets [Lewalle, 2002b]. There are two wavelets from this family useful for this thesis. They correspond to the first and second derivatives of the Gaussian function. They are the *Gaussian wave* and the *Mexican Hat*. The graph and the formula of the normalized Gaussian wave are presented below.

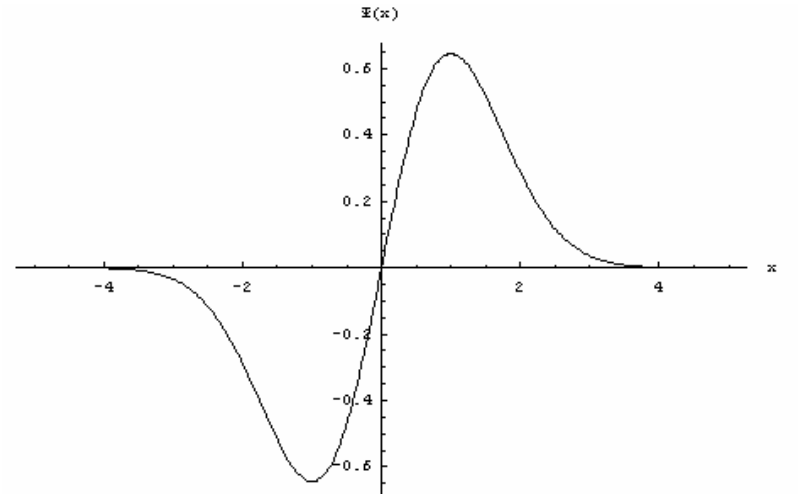

Figure 2.1 The graph of $\Psi(x) = \frac{\sqrt{2}}{\sqrt[4]{\pi}} x e^{-\frac{x^2}{2}}$.

The graph and the formula of the Mexican Hat wavelet are presented below:

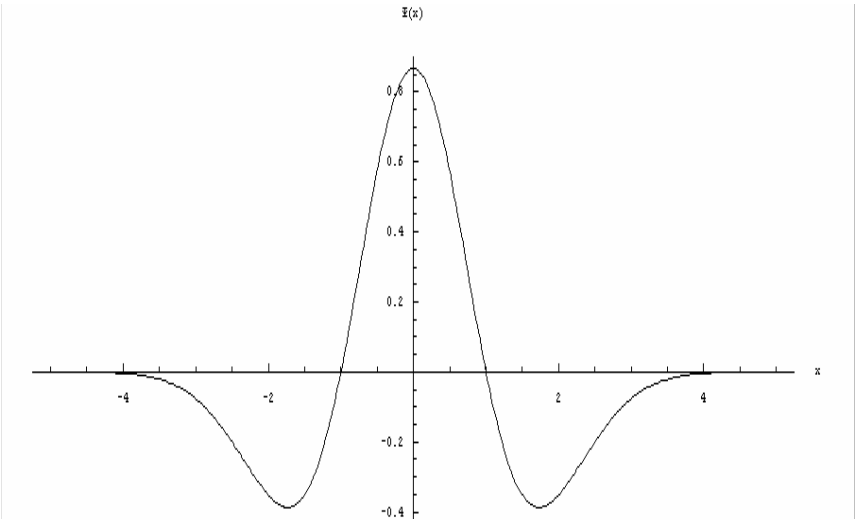

Figure 2.2 The graph of $\Psi(x) = \frac{2}{\sqrt{3}\sqrt[4]{\pi}} (1 - x^2) e^{-\frac{x^2}{2}}$.

2.3 Basis functions

There are different types of orthogonal polynomials. Two of them are useful in this investigation: the Hermite and the Laguerre polynomials. The Rodríguez's formulas for the Hermite and Laguerre polynomials are as follows:

$$H_q(b)=(-1)^q e^{\frac{b^2}{2}}\frac{d^q}{db^q}(e^{-\frac{b^2}{2}}) \text{ and}$$

$$L_p^{\alpha}(k)=\frac{e^{\frac{k}{2}}}{k^{\alpha}p!}\frac{d^p}{dk^p}(k^{p+\alpha}e^{-\frac{k}{2}}) \tag{2.13}$$

where $p,q=0,1,2,\ldots$. The parameter α is chosen depending on the degree of the polynomial part of the function to be expanded. It is easier to calculate these special polynomials using one of the following recurrence relations:

$$H_{q+1}(b)=2bH_q(b)-2qH_{q-1}(b) \text{ and}$$

$$L_{p+1}^{\alpha}(k)=\frac{(2p+1+\alpha-k)L_p^{\alpha}(k)-(p+\alpha)L_{p-1}^{\alpha}(k)}{p+1}. \tag{2.14}$$

A basis makes it possible [Book, 1981] to expand an arbitrary function $f(k,b)$ as a linear combination of the basis functions in this way:

$$f(k,b)=\sum_{p,q=0} g_{p,q}\, l_p(k)\, h_q(b). \tag{2.15}$$

In this case the basis functions are the following:

$$l_p(k)=\frac{e^{-\frac{k}{2}}k^{\frac{\alpha}{2}}L_p^{\alpha}(k)}{\sqrt{(p+\alpha)!/\,p!}} \text{ and } h_q(b)=\frac{e^{-\frac{x^2}{2}}H_q(b)}{\sqrt{2^q\,q!\sqrt{\pi}}}. \tag{2.16}$$

This basis consisting of the product $l_p(k)h_q(b)$ is complete for the space $L^2(\mathbb{R}^+\times\mathbb{R})$ [Isaacson et al., 1994]. These basis functions were chosen because the parameter *k* is defined in the interval $(0,\infty)$ and the parameter *b* is defined in the interval $(-\infty,\infty)$.The graphs of the basis functions (2.16) were plotted below for several *p* and *q* values and $\alpha=3$.

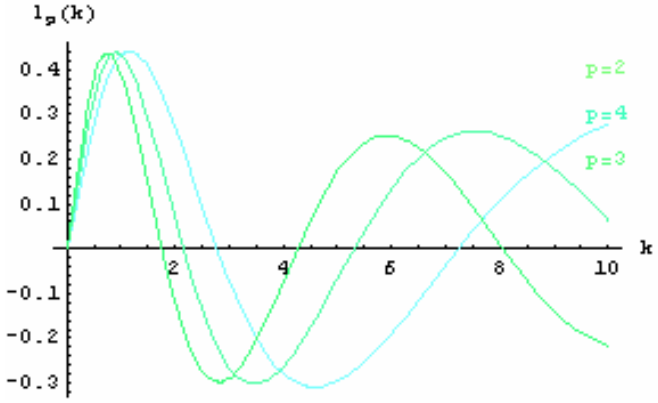

Figure 2.3 The graph of the basis function $l_p(k)$ for $p = 2, 3, 4$ and $\alpha = 3$.

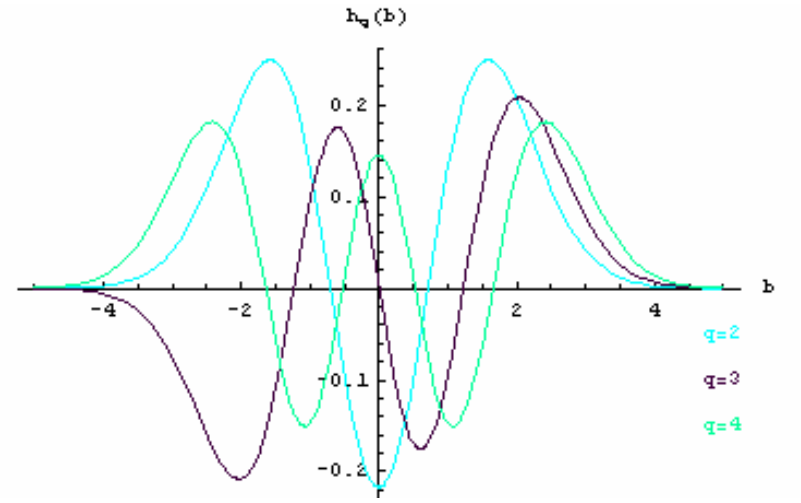

Figure 2.4 The graph of the basis function $h_q(b)$ for $q = 2, 3, 4$.

The basis functions, $h_p(b)$ ($p = 0, 1, 2,\ldots$) are orthonormal in pairs. The same is true for the set of basis functions $l_q(k)$ ($q = 0, 1, 2,\ldots$). Their orthogonality and completeness can be used to expand an arbitrary function in a series. The orthonormality of this basis can be checked by using the following integral relations, for each one:

$$\int_{-\infty}^{\infty} h_p(b)h_q(b)\,db = \delta_{pq} \quad \text{and} \quad \int_0^{\infty} l_p(k)l_q(k)\,dk = \delta_{pq}\,. \tag{2.17}$$

Here δ_{pq} is the Kronecker delta, which equals unity when p=q and zero otherwise.

2.4 The integro-differential equation for the wavelet transformant

Let us consider the following initial value problem with a boundary condition for the Kuramoto-Sivashinsky equation

$$u_t = -u_{xx} - u_{xxxx} - 4uu_x,\ u|_{t=0} = f(x),\ \lim_{x\to\pm\infty} u(x,t) = 0; \tag{2.18}$$

where *f(x)* is a given function of the coordinate *x* (an initial condition) and the solution *u* straights to zero as *x* straights to plus or minus infinity. As it will be shown later, it is possible to convert (2.18) to an equation in terms of $W_{k,b}(t)$, the wavelet transformation defined by the formula (2.5). It is a nonlinear integro-differential equation for the wavelet transformant $W_{k,b}(t)$:

$$\begin{aligned}\frac{d}{dt}W_{k,b}(t) = &\int_{-\infty}^{\infty}\int_{K=0}^{\infty}\lambda(k,K,b,B)W_{K,B}(t)\,dK\,dB + \\ &\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{K=0}^{\infty}\int_{\kappa=0}^{\infty}\nu(k,K,\kappa,b,B,\beta)W_{\kappa,\beta}(t)W_{K,B}(t)\,d\kappa\,dK\,d\beta\,dB\end{aligned} \tag{2.19}$$

In the formula above, $\lambda(k,K,b,B)$ and $\nu(k,\kappa,K,b,\beta,B)$ are the following improper definite integrals with respect to *x*:

$$\lambda(k,K,b,B) = -\frac{\sqrt{kK}}{C_\Psi}\int_{-\infty}^{\infty}\left(\frac{d^2\Psi_{K,B}(x)}{dx^2} + \frac{d^4\Psi_{K,B}(x)}{dx^4}\right)\Psi_{k,b}(x)dx \tag{2.20}$$

and

$$\nu(k,K,\kappa,b,B,\beta) = 2\frac{\sqrt{kK\kappa}}{C_\Psi^2}\int_{-\infty}^{\infty}\Psi_{K,B}(x)\Psi_{\kappa,\beta}(x)\frac{d}{dx}\Psi_{k,b}(x)dx.$$

The parameters *k*, *K*, and *κ* are scale parameters. The parameters *b*, *B*, and *β* are position parameters. To solve the integro-differential equation the method of Galerkin was used.

2.5 Galerkin's method

Galerkin's method was invented by the Russian mathematician Boris Grigoryevich Galerkin. For this method, orthogonal polynomials are usually used as the basis functions [Book, 1981]. This method employs projections onto basis functions to obtain ordinary differential equations that describe the evolution of the system when time t changes.

According to this method, the value of $W_{k,b}(t)$ is searched in the form of a linear combination of the basis functions:

$$W_{k,b}(t)=\sum_{i=0}^{M}\sum_{j=0}^{M} c_{i,j}(t)\,l_i(k)\,h_j(b), \tag{2.21}$$

where $l_i\,(k)$ and $h_j\,(b)$ are the basis functions described by the formulas (2.16); and $c_{i,j}\,(t)$ are unknown coefficients that depend on time t. The number M is a natural number that is chosen to be equal to *4 (M=4)*. Of course, it would be better to have bigger M, but a reasonably small value of M had to be chosen because the amount of calculations grows dramatically when M grows.

To convert the equation (2.19) to a system of ordinary differential equations, the representation (2.21) was substituted into (2.19) and then the right-hand side and left-hand side were multiplied by $l_p(k)h_q(b)$ and integrated for k from zero to infinity and for b from minus infinity to infinity. Doing so, the property of orthonormality for the basis functions was used. It gave the following nonlinear system of ordinary differential equations:

$$\frac{dc_{p,q}(t)}{dt}=\sum_{i,j}^{\infty} L_{p,q}^{i,j}c_{i,j}(t)+\sum_{i,j}^{\infty}\sum_{r,s}^{\infty} N_{p,q}^{i,j,r,s}c_{i,j}(t)c_{r,s}(t) \tag{2.22}$$

where

$$L_{p,q}^{i,j}=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{K=0}^{\infty}\int_{k=0}^{\infty}\lambda(k,K,b,B)l_p(k)l_i\,(K)h_q(b)h_j(B)\;dkdKdbdB \tag{2.23}$$

and

$$N_{p,q}^{i,j,r,s}=\int_{-\infty}^{\infty}\;\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{K=0}^{\infty}\int_{\kappa=0}^{\infty}\;\int_{k=0}^{\infty}\nu(k,\kappa,K,b,\beta,B)l_p(k)l_i\,(K)l_r\,(\kappa)h_q(b)h_j(B)h_s(\beta)\,dkdKd\kappa dbdBd\beta$$

are numerical coefficients corresponding to the linear and nonlinear terms, respectively. Formulas (2.21) to (2.23) constitute the Galerkin approximation. Further details about the application of Galerkin's method are explained in chapter 3 of this investigation.

3. Materials and Methods

In this chapter, first the materials are described, the software used and the computer's minimum specifications needed to perform this investigation. Thereafter, the methods to perform the wavelet transformation to a specific function and to the Kuramoto-Sivashinsky equation are explained.

3.1 Materials

The investigation of this thesis utilized the computer algebra system Mathematica 5.2 created by Wolfram Research, Inc. to solve and analyze the systems of equations. Mathematica system has a powerful programming language which makes the process user- friendly and in this way it is very useful to the people with all levels of knowledge in programming [Ruskeepää, 1998]. To create and run the programs written to solve the Kuramoto-Sivashinsky equation, a computer with at least 2 GB of RAM and a Pentium IV dual-core processor of 3.4 GHz was needed.

3.2 Methods

In this part it is explained how to transform and restore a function by means of wavelets. It was explained by a specific example. With the example, it was verified that the formulas developed in this work were correct. This example helps understand the application of the wavelet transform. The parts corresponding to the orthogonal polynomials and the Galerkin method were not illustrated here because we wanted to emphasize the use of wavelets in this investigation. Then a detailed explanation of how the solution to Kuramoto-Sivashinsky equation was found is given.

3.2.1 Example for a known function

The first step was to check out how one can obtain the wavelet transformation of a function f and how this function can be restored by means of wavelets. The function chosen

was $f(x) = sin(x)$. Two types of continuous wavelets were selected: the Gaussian wave and the Mexican Hat.

The wavelet transformant, the restored function, and the absolute errors of these approximate transformations were calculated for each type of wavelets used. Two graphs, one for each wavelet, of the transformed function were plotted. In addition, the performances of both wavelets were compared. The corresponding Mathematica program for this part of the procedure is the first one in the Appendix.

3.2.2 Application to the Kuramoto-Sivashinsky equation

To find a wavelet-based solution to the Kuramoto-Sivashinsky equation, it was assumed that the unknown function $u(x,t)$ can be described by means of the following formula:

$$u(x,t) = \frac{1}{C_\Psi} \int_{-\infty}^{\infty} \int_{k=0}^{\infty} W_{k,b}(t) \Psi_{k,b}(x) \sqrt{k}\, dk\, db; \tag{3.1}$$

where C_Ψ is the admissibility constant and $W_{k,b}(t)$ is the continuous wavelet transform of the function $u(x,t)$. In the formulas above, the positive number k is the scale parameter similar to a wave number (the reciprocal of the wavelength), b is the position parameter, and $\Psi(x)$ is a known function that is named "mother wavelet" or "analyzing wavelet". In addition, it can be assumed that the solution $u(x,t)$ has mean zero; in formula, $\int_{-\infty}^{\infty} u(x,t)\, dx = 0,$ as it is true for the continuous wavelet transform (see the formula (2.1)). For the objective of this work, in formula (2.5), we can use Ψ instead of $\overline{\Psi}$ because of the choice of a real-valued wavelet Ψ.

The differentiation of both sides of the equation (3.1) with respect to *x*, results in the following formulas:

$$u_{xx}(x,t) = \frac{1}{C_\Psi} \int_{-\infty}^{\infty} \int_{k=0}^{\infty} \sqrt{k} W_{k,b}(t) \frac{d^2}{dx^2} \Psi_{k,b}(x)\, dk\, db, \qquad \text{and}$$

$$u_{xxxx}(x,t) = \frac{1}{C_\Psi} \int_{-\infty}^{\infty} \int_{k=0}^{\infty} \sqrt{k} W_{k,b}(t) \frac{d^4}{dx^4} \Psi_{k,b}(x)\, dk\, db\,. \tag{3.3}$$

These two formulas are useful for the linear term $-u_{xx} - u_{xxxx}$. Following Beylkin's approach [Beylkin et al., 1996; 1997; 2002], the Kuramoto-Sivashinsky differential equation can be written in the form $u_t = Lu + Nf(u)$ where L and N are the differential operators

$$L = -\frac{\partial^2}{\partial x^2} - \frac{\partial^4}{\partial x^4},\ N = -\frac{\partial}{\partial x}, \text{ and } f(u) = 2u^2.$$

For this reason, before writing the corresponding formula for the nonlinear term, $4uu_x$, we had to perform a substitution in it. We replaced $4uu_x$ by $2(u^2)_x$. Thereafter, for the nonlinear term the corresponding formula (3.3) yields:

$$2\frac{\partial}{\partial x}[u(x,t)]^2 = \frac{2}{C^2_{\Psi}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{k=0}^{\infty}\int_{K=0}^{\infty} \sqrt{kK}\, W_{k,b}(t) W_{K,B}(t) \frac{d}{dx}[\Psi_{k,b}(x)\Psi_{K,B}(x)]\, dK\, dk\, dB\, db. \tag{3.4}$$

Then it was multiplied the left-hand side and right-hand side of the Kuramoto-Sivashinsky equation by the daughter wavelet $\Psi_{k,b}(x)$ and integrated with respect to x from $-\infty$ to $+\infty$. It means that the wavelet transform (2.5) was applied to each term of the equation. When it was doing so, the formulas (3.1)-(3.4) were used to eliminate $u(x, t)$ and use $W_{k,b}(t)$ instead. The order of integrations was changed in such a way to perform first the integration with respect to x. It gave the formulas (2.20) for the functions λ and ν. The formulas were simplified by means of integrating by parts. Eventually, it yielded the integro-differential equation (2.19).

The values of λ and ν were found in symbolic form using the Mathematica program named *intx* explained in the Appendix. Analyzing the expressions for λ and ν, it was noticed that they both are proportional to k^3. This is why we decided to choose the value of the parameter α in the basis function based in the Laguerre polynomial equal to three ($\alpha = 3$).

In order to solve a Cauchy problem for the system (2.22), an initial condition was needed. It was chosen an initial condition suitable for the illustration of the method used in this investigation: $u(x,0) = \sin(0.1x)\, e^{-\frac{x^2}{2}}$. The proper graph is shown below.

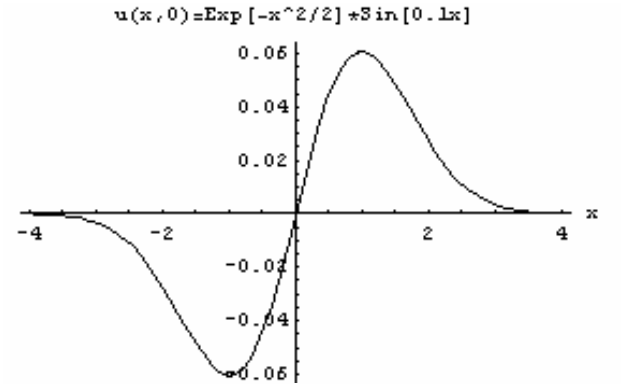

Figure 3.1 The plot of the initial condition $u(x,0) = \sin(0.1x)\, e^{-\frac{x^2}{2}}$.

Then this initial condition was transformed by the wavelet based on formula (2.5). In addition, it had to be expanded in terms of the basis functions (2.16).

The Mathematica function **TrigToExp** converts functions like the initial condition into exponential functions which make integration calculations faster. For this reason, I used **TrigToExp** in this case. Mathematica calculated the initial condition's wavelet transformation in terms of complex numbers, but the imaginary part of the complex numbers vanishes when one evaluates the integrals.

For this initial condition the integrations were conducted symbolically for the parameter *b* and numerically for the parameter *k*. For this purpose a Mathematica program named ***simpson*** was created. It can be found in the Appendix. Here is the formula developed for this purpose:

$$c_{p,q}(t)|_{t=0} = \int_{-\infty}^{\infty}\int_{k=0}^{\infty}\int_{-\infty}^{\infty} u(x,0)\Psi_{k,b}(x) l_p(k) h_q(b)\sqrt{k}\, dx dk db\,. \tag{3.5}$$

In order to restore back the solution in terms of *x* and *t*, the inverse transform of the product of the basis functions had to be calculated. It gives us the following result:

$$u(x,t) = \sum_{p,q} c_{p,q}(t)\varphi_{p,q}(x)\,, \tag{3.6}$$

where

$$\varphi_{p,q}(x) = \frac{1}{C_\Psi}\int_{-\infty}^{\infty}\int_{k=0}^{\infty} l_p(k) h_q(b)\Psi_{k,b}(x)\sqrt{k}\; dk\, db. \tag{3.7}$$

The integration with respect to *b* was done symbolically, but for the parameter *k*, numerical integration was needed. The calculations for this part were performed with the programs named ***tifb*** and ***simpson*** explained in the Appendix.

As written before, the value of *M* chosen for formula (2.18) was ***4***. It yields ***25*** inverse transforms and ***25*** coefficients for the initial condition, ***625*** coefficients for the linear term and ***15,625*** coefficients for the nonlinear term. The magnitude of *M* can be increased if a more powerful computer is available.

The steps to solve the Kuramoto-Sivashinsky equation by means of the wavelet transform can be summarized as follows:

1. Apply the wavelet transform given by the formula (2.5) to each term of the Kuramoto-Sivashinsky equation and integrate by parts to eliminate partial derivatives of *u* with respect to *x*. It gives the following equation:

$$\frac{d}{dt}W_{k,b}(t)=\int_{-\infty}^{\infty}u(x,t)[-\frac{d^2\Psi_{k,b}(x)}{dx^2}-\frac{d^4\Psi_{k,b}(x)}{dx^4}]\sqrt{k}dx+2\int_{-\infty}^{\infty}u^2(x,t)\frac{d\ \Psi_{k,b}(x)}{dx}\sqrt{k}dx\,. \quad (3.8)$$

2. Use formulas (3.3) and (3.4) to eliminate *u(x, t)*. It results in the nonlinear integro-differential equation (2.19) where the values of $\lambda(k,K,b,B)$ and $\nu(k,K,\kappa,b,B,\beta)$ are to be calculated by formulas (2.20).

3. Use the Galerkin method: assume that $W_{k,b}(t)$ is given as the linear combination of the basis functions as it is described by the formula (2.21) and then project the left-hand side and the right-hand side of the equation (3.8) onto the basis functions $l_p(k)h_q(b)$ ($p, q = 0,1,2,\ldots$) and obtain the system (2.22) of ordinary differential equations for the unknown coefficients $c_{p,q}(t)$.

4. Calculate numerically numerous coefficients of the system of ODEs (2.23).

5. Apply wavelet transform to *u(x, 0)*; project the value of the transformant onto basis functions $l_p(k)h_q(b)$ ($p, q = 0, 1, 2, \ldots$) and obtain initial conditions for the coefficients $c_{p,q}(t)$ for initial moment $t=0$ as it is given in the formula (3.5).

6. Integrate the system of ODEs for the coefficients by means of a numerical procedure.

7. Use the formulas (3.6) and (3.7) to restore the solution *u(x, t)*.

4. Results

In this chapter the results of my investigation are enumerated. In the first part I described the output in the case of the function *f(x) = sin(x)* using the wavelets named Mexican Hat and the Gaussian wave. The second part is devoted to the results obtained with the Gaussian wave when it was applied to the Kuramoto-Sivashinsky equation. The Mathematica programs used to find the results of this part are explained in the Appendix below.

4.1 Comparison of two wavelets

For the function $f(x) = sin(x)$, two three-dimensional plots for k from 0.1 to 3.1 and b from -6.0 to 6.0 were built; one for each wavelet type. It is necessary to remark here that for the parameter k the graphs starts with $k = 0.1$ because of the condition $k > 0$. The first one is described by means of the Gaussian wave and the second one is based on the Mexican Hat.

In each plot, the corresponding formula for the wavelet transformation was written. The graphs for each case are the following:

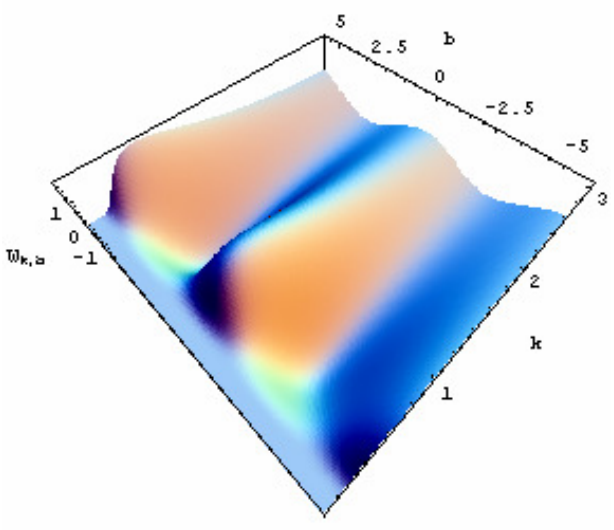

Figure 4.1 Graph of $W_{k,b} = 2\pi^{\frac{1}{4}} k^{-\frac{3}{2}} e^{-\frac{1}{2k^2}} \cos(b)$.

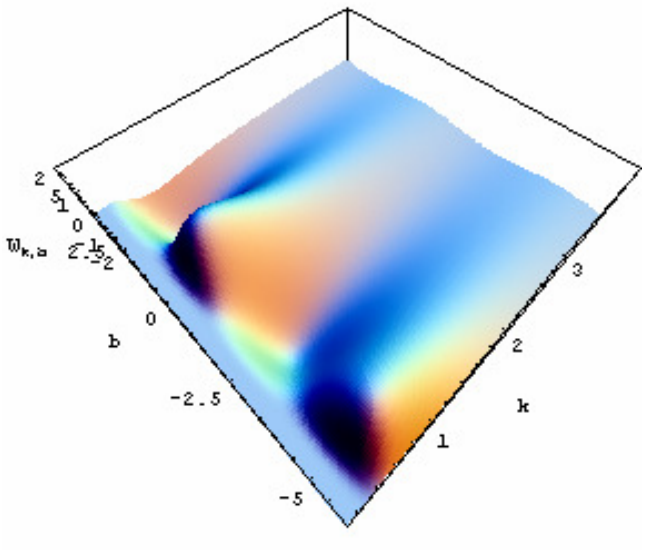

Figure 4.2 Graph of $W_{k,b} = 2\sqrt{\frac{2}{3}}\pi^{\frac{1}{4}}k^{-\frac{5}{2}}e^{-\frac{1}{2k^2}}\sin(b)$.

In addition, there is a table, which illustrates the error for some values of x. It can be seen that the mean error is less when using the Mexican Hat wavelet. The table is the following:

x	fOriginal	Gaussian Wave		Mexican Hat	
		fRestored	Absolute error	fRestored	Absolute error
-0.785	-0.707	-0.707	$1.53\cdot10^{-7}$	-0.707	$6.61\cdot10^{-9}$
0.00	0.00	$6.53\cdot10^{-10}$	$6.53\cdot10^{-10}$	$-3.90\cdot10^{-9}$	$-3.90\cdot10^{-9}$
0.785	0.707	0.707	$1.53\cdot10^{-7}$	0.707107	$6.61\cdot10^{-9}$
1.57	1.00	1.00	$2.37\cdot10^{-7}$	1.00	$2.57\cdot10^{-5}$
2.36	0.707	0.707539	$4.32\cdot10^{-4}$	0.707	$5.82\cdot10^{-8}$
3.14	0.00	$1.46\cdot10^{-5}$	$1.46\cdot10^{-5}$	$1.70\cdot10^{-7}$	$1.70\cdot10^{-7}$

Table 4.1 Comparison of original and restored function and its corresponding error for some x values using the Gaussian Wave and the Mexican Hat wavelets.

Absolute errors are very small numbers, as can be seen from the table above. It means that the quality of the approximation is very good.

4.2 **Results in case of the Kuramoto-Sivashinsky equation**

The solutions for the integral formulas λ and ν of equation (2.20) are the following:

$$\lambda = -\sqrt{\frac{2(kK)^7}{\pi}}\frac{e^{-\frac{((b-B)kK)^2}{2(k^2+K^2)}}}{\sqrt{(k^2+K^2)^{13}}}((3K^8 - 3k^2K^6(K^2(5+2b^2-4bB+2B^2)-4)$$
$$+(kK)^4(18-9(5+2b^2-4bB+2B^2)K^2+(b-B)^2K^4(45+b^2-2bB+B^2))-15(b-B)^4K^8$$
$$+k^6(12K^2-9(5+2b^2-4bB+2B^2)K^4+2(b-B)^2K^6(45+b^2-2bB+B^2)$$
$$+k^8(3-3(5+2b^2-4bB+2B^2)K^2+(b-B)^2K^4(45+b^2-2bB+B^2)-15(b-B)^4K^6+(b-B)^6K^8$$

and

$$\nu = \frac{2}{\pi^{\frac{5}{4}}}\sqrt{\frac{(kK\kappa)^3}{(k^2+K^2+\kappa^2)^9}}e^{-\frac{(k^2+K^2)(\beta\kappa)^2+(k^2+\kappa^2)(BK)^2(K^2+\kappa^2)(bk)^2-2(BK^2\beta\kappa^2+bk^2(BK^2+\beta\kappa^2))}{2(k^2+K^2+\kappa^2)}}.$$
$$(k^6(2+(b-B)K^2(2B+b((K(b-B))^2-5)+3\beta-(b-B)^2K^2\beta)+(b-\beta)(3B+b(2((b-B)K)^2$$
$$-5+2\beta-2(b-B)^2K^2\beta)\kappa^2+(b-B)^3\kappa^4)-k^2(B-b)K^6(b-4B+3\beta)+(b-B)K^4(4B+b(-3$$
$$+2(K(B-\beta))^2)-B(1+2K^2(B-\beta)^2)+4\beta)\kappa^4+(B-\beta)(4\beta-3B-K^2(B-\beta)^2\beta+b(BK-1$$
$$-K\beta)(1+(B-\beta)K)\kappa^6)+(K^2+\kappa^2)^2(K^2(((B-\beta)\kappa)^2-1)-\kappa^2)+k^4((b-B)^3K^6(B-\beta)+3\kappa^2$$
$$-2(b-\beta)(2b-3B+\beta)\kappa^4+(b-\beta)^3(\beta-B)\kappa^6+(b-B)K^4(6\beta-4b-2B+(b-B)(b-\beta)(B-$$
$$\beta)\kappa^2)+K^2(3-(8b^2+3B^2+2B\beta+3\beta^2-8b(B+\beta))\kappa^2+(b-B)(b-\beta)^2(\beta-B)\kappa^4)))$$
(4.1)

The following graphs were built with the Mathematica function **ListPlot**. The first one corresponds to the coefficients $L_{p,q}^{i,j}$ calculated for the linear term. The second one is about the coefficients $N_{p,q}^{i,j,r,s}$ corresponding to the nonlinear term. The third one is related with the coefficients for the initial condition. The colors in the graphs depend on the position of each number in each list.

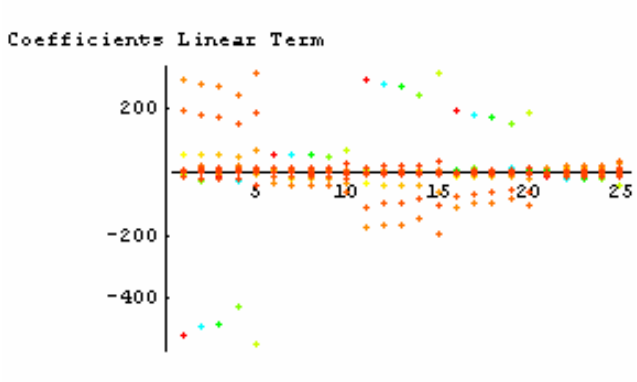

Figure 4.3 The 625 coefficients of the linear term of the Kuramoto-Sivashinsky equation

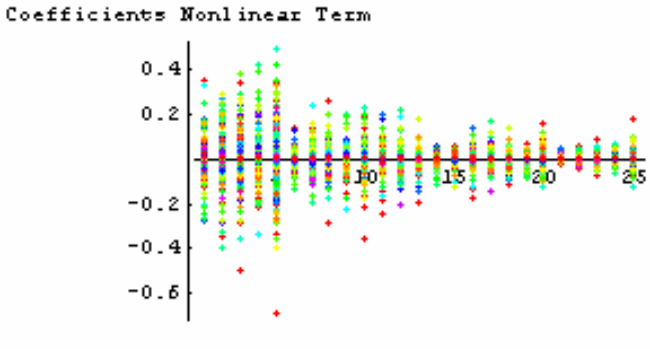

Figure 4.4 The 15,625 coefficients of the nonlinear term of the Kuramoto-Sivashinsky equation

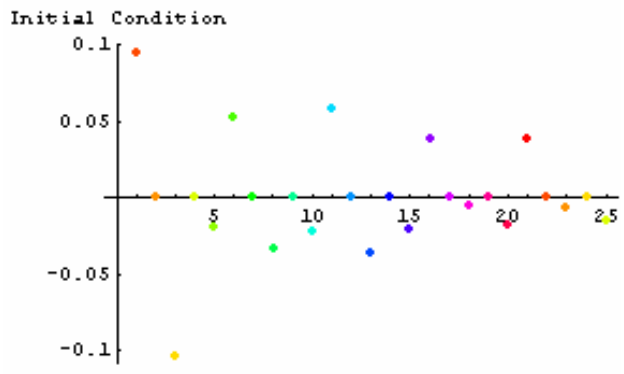

Figure 4.5 The 25 coefficients of the initial condition wavelet transformation

The initial condition chosen has the following wavelet transform formula:

$$W_{k,b}\,|_{t=0} = \frac{\pi^{\frac{1}{4}}}{5}\sqrt{\left(\frac{k}{1+k^2}\right)^3}\,\frac{\sqrt{e^{(bk)^2}}}{e^{\frac{1+100(bk)^2(2+k^2)}{200(1+k^2)}}}\left(\cos\left(\frac{bk^2}{10(1+k^2)}\right)-10b\sin\left(\frac{bk^2}{10(1+k^2)}\right)\right). \qquad (4.2)$$

which has this graph:

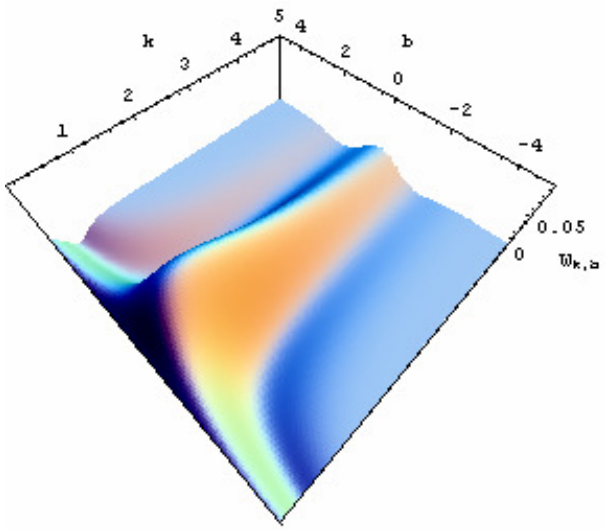

Figure 4.6 Graph of the initial condition's wavelet transformation

The admissibility constant of the Gaussian wave is $C_\Psi = 2\sqrt{\pi}$. It was obtained with formula (2.4). As it can be seen from the table presented below, the solution obtained satisfies the initial condition with a good accuracy.

x	Initial Condition	Solution for $t=0$	Absolute Error
0.000	0.000	0.000	0.000
0.200	$1.960\cdot10^{-2}$	$1.961\cdot10^{-2}$	$9.708\cdot10^{-6}$
0.400	$3.691\cdot10^{-2}$	$3.695\cdot10^{-2}$	$3.050\cdot10^{-5}$
0.600	$5.009\cdot10^{-2}$	$5.012\cdot10^{-2}$	$3.823\cdot10^{-5}$
0.800	$5.803\cdot10^{-2}$	$5.799\cdot10^{-2}$	$4.255\cdot10^{-5}$
1.000	$6.055\cdot10^{-2}$	$6.023\cdot10^{-2}$	$3.256\cdot10^{-4}$
1.200	$5.827\cdot10^{-2}$	$5.735\cdot10^{-2}$	$9.249\cdot10^{-4}$
1.400	$5.237\cdot10^{-2}$	$5.046\cdot10^{-2}$	$1.907\cdot10^{-3}$
1.600	$4.430\cdot10^{-2}$	$4.104\cdot10^{-2}$	$3.256\cdot10^{-3}$
1.800	$3.543\cdot10^{-2}$	$3.057\cdot10^{-2}$	$4.860\cdot10^{-3}$
2.000	$2.689\cdot10^{-2}$	$2.034\cdot10^{-2}$	$6.535\cdot10^{-3}$

Table 4.2 Verification that the initial condition is satisfied for eleven x values

There were 25 inverse transforms of the product of basis functions $\varphi_{p,q}$. What remained were the expressions in terms of x. For different p and q values three plots are shown in figure 4.6.

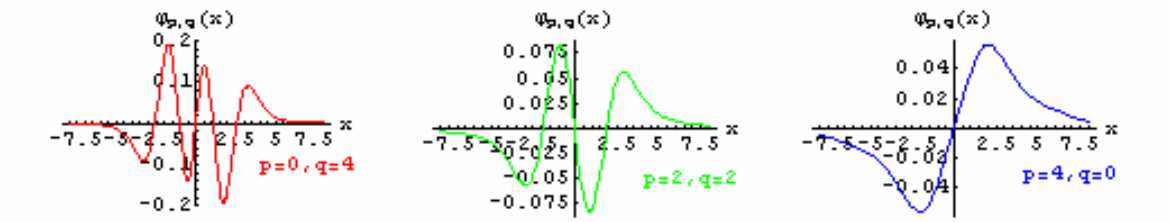

Figure 4.7 Plot of $\varphi_{p,q}(x)$ for different values of p and q

The following graph illustrates the solution $u(x,t)$ for different moments of time t.

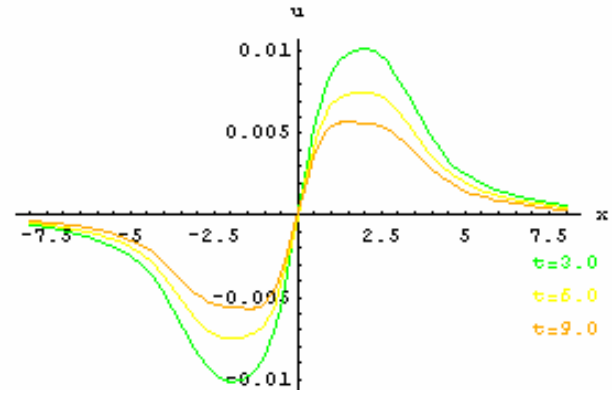

Figure 4.8 Plot of the solution for different values of time t

The following graph illustrates the solution $u(x,t)$ for different for different values of x.

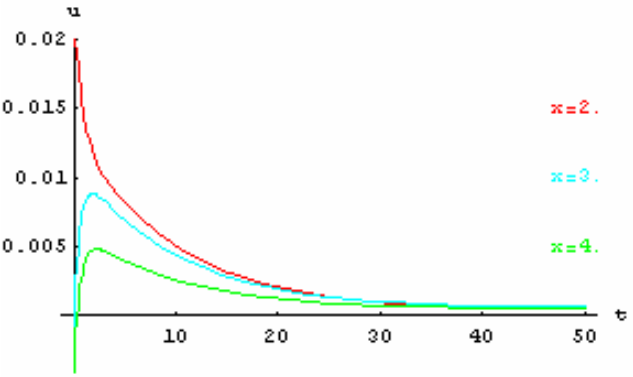

Figure 4.9 Plot of the solution for different values of x

This last figure is a three-dimensional plot of the solution *u(x,t)* for *x* from -4.0 to 4.0 and *t* from 0.0 to 2.0.

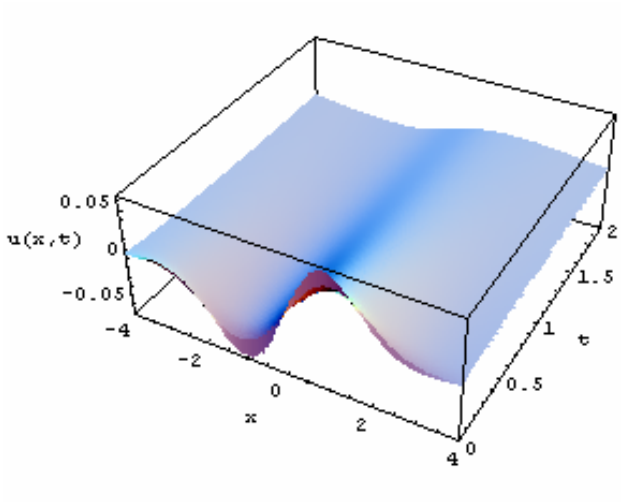

Figure 4.10 Plot of the solution *u(x,t)*

5. Conclusions

In this chapter, firstly the results of the investigation were discussed for a known function and the Kuramoto-Sivashinsky equation. Thereafter the conclusions based on the results obtained were discussed.

5.1 Discussion

In the first part of the chapter about the results the Mexican Hat and the Gaussian wave were employed to calculate the wavelet transform of a sample function. It was shown that in the case of the Mexican Hat wavelet the error is less than in case of the Gaussian wave. It means that the Mathematica program **NIntegrate** used to calculate improper definite integrals gave better results in the case of Mexican Hat. From this point of view Mexican Hat is the best of both wavelets.

In terms of computer efficiency, the Mathematica programs runs faster when the Gaussian wave is employed. It means that less time is needed to conduct the calculations when this wavelet is used. From this point of view, the Gaussian wave is a better choice.

After this comparison was done, it was decided to apply the wavelet transform based on the Gaussian wave to the Kuramoto-Sivashinsky equation because of the time consuming experience with the Mexican Hat wavelet. The Gaussian wave was a better choice also because it is not as popular as the Mexican Hat. This statement can be supported by referring to some of the previous investigations where this wavelet was employed [Farge et al., 1996; Lewalle, 1993, 2002; Starck et al. 1998]. On the contrary, the wavelet used is not used so often, and this investigation can serve as a way to let other people to develop a detailed knowledge about it. In addition, there was more simplicity in the formulas obtained when the Gaussian wave was used.

The results obtained in section 4.1 motivated to use the Gaussian wave in the case of the Kuramoto-Sivashinsky equation. Mathematica is powerful enough to allow me to obtain exact

formulas for the values of the functions λ and ν. The formulas were not so easy to handle, even when they were simplified.

The most difficult part was to calculate the coefficients $L_{p,q}^{i,j}$ and $N_{p,q}^{i,j,r,s}$ because the formulas for them require the calculation of multidimensional improper integrals and the total amount of coefficients is 625 for $L_{p,q}^{i,j}$ and 15625 for $N_{p,q}^{i,j,r,s}$ (if M=4). These coefficients were calculated numerically because the amount of parameters for which integration had to be done was significant: four for the linear term and six for the nonlinear term. A more powerful computer, maybe, will give output in a symbolically form. The inability to find access to a powerful computer was the biggest limitation in this thesis investigation.

The magnitude of the coefficients of the linear term was bigger than those of the nonlinear term, as can be seen from the figures (4.3) and (4.4). In the case of the linear term, the coefficients of the term with the fourth derivative were larger than the coefficients of the term with second derivative. The reason was that, the polynomial part of result (4.2) had higher degrees in k's and b's in this term. In this part the larger contribution to the solution was done by the term with the fourth derivative.

In addition, the initial condition chosen was adequate to illustrate how to combine it with the equation transformed. The results of the table (4.2) served to validate in terms of the accuracy the solution obtained. The accuracy increases as the value of M grows. The solution decreases slowly as time t passes. A solution can be found, even for big values of time t. The calculations conducted allowed to create a short movie that demonstrates the evolution of the solution in time.

5.2 Conclusion

The procedure based on the continuous wavelet transform and the Galerkin method was implemented; and proper Mathematica code was created. A numerical solution of the nonstationary Kuramoto-Sivashinsky equation was found. Firstly, by means of the wavelet

transform, the nonlinear integro-differential equation was obtained. Then it was transformed to a nonlinear system of ordinary differential equations by means of the Galerkin procedure based on the use of classical orthogonal polynomials introduced by Hermite and Laguerre. Numerical integration was used to find the solution of the system of ordinary differential equations. The power of Mathematica system was demonstrated.

6. Bibliography

[1] Addison, P. S. (2005). *Wavelet transforms and the ECG: a review*. Retrieved on January 20, 2006 from www.stacks.iop.org/PM/26/R155.

[2] Beylkin, G. and Keiser, J. M. (1996). *On the Adaptive Numerical Solution of Nonlinear Partial Differential Equations in Wavelet Bases*. Retrieved on February 1, 2006 from http://amath.colorado.edu/pub/wavelets/papers/wavelets_for_PDEs.pdf.

[3] Beylkin, G. and Keiser, J. M. (1997). *An Adaptive Pseudo-Wavelet Approach for Solving Nonlinear Partial Differential Equations*. Wavelet Analysis and Applications. Volume 6. Academic Press, California.

[4] Beylkin, G., Alpert, B., Gines, D., and Vozovoi, L. (2002). *Adaptive Solution of Partial Differential Equations in Multiwavelet Bases*. Journal of Computational Physics. Volume 182, Pages 149 -190.

[5] Book, D. L. (1981). *Finite-difference techniques for vectorized fluid dynamics calculations*. Springer series in computational physics, New York, New York.

[6] Chui, C. K. (1992). *An Introduction to Wavelets*. Academic Press, San Diego, California.

[7] Dahmen, W., Kurdilla, A., and Oswald, P. (1997). *Multiscale Wavelet Methods for Partial Differential Equations (Wavelet Analysis and Its Applications)*. Second Edition. Academic Press, California.

[8] Daubechies, I. (1992). *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

[9] Demekhin, E. A., Chang, H. –C., and Kopelevich, D. I. (1993*).Construction of stationary waves in a falling film*. Computational Mechanics. Volume 11, Pages 313-322.

[10] Elezgaray, J., Berkooz, G., and Holmes, P. (1996). *Large-scale statistics of the Kuramoto-Sivashinsky equation: A wavelet-based approach*. Physical Review E. Volume 54, Number 1. Pages 224-230.

[11] Eremenko, A. (2005). *Meromorphic traveling wave solutions of the Kuramoto-Sivashinsky equation*. Retrieved on April 23, 2006 from http://www.math.purdue.edu/~eremenko/dvi/conte.pdf.

[12] Farge, M., Kevlahan, N., Perrier, V., and Goirand, E. (1996). *Wavelets and Turbulence*. Proceedings of the IEEE. Volume 84. Number 4. Pages 639-668.

[13] Fedorova, A. N. and Zeitlin, M. G. (2002). *Localization and Coherent Structures in Wave Dynamics via Multiresolution.* Retrieved on April 3, 2006 from http://arxiv.org/PS_cache/nlin/pdf/0206/0206024.pdf.

[14] Griebel, M. and Koster, F. (2000). *Adaptive Wavelet Solvers for the Unsteady Incompressible Navier-Stokes Equations*. Retrieved on February 14, 2006 from http://wissrech.ins.uni-bonn.de/research/projects/koster/Navier.html.

[15] Isaacson, E. and Keller, H. B. (1994). *Analysis of numerical methods*. Dover Publications. Mineola, New York.

[16] Lewalle, J. (1993). *Energy dissipation in the wavelet - transformed Navier - Stokes equations*. Phys. Fluids A. Volume 5. Number 6. Pages 1512 – 1513.

[17] Lewalle, J. (2002). *Self-organization in Navier-Stokes turbulence*. Retrieved on March 2, 2006 from http://www.ecs.syr.edu/faculty/lewalle/papers/necsi.pdf.

[18] Lewalle, J. (2002). *Wavelet Dynamics at Turbulent/Non-Turbulent Interfaces*. Retrieved on March 2, 2006 from http://www.ecs.syr.edu/faculty/lewalle/papers/PoF.pdf.

[19] Liu, W.-J. and Krstić. (2001). *Stability enhancement by boundary control in the Kuramoto-Sivashinsky equation*. Nonlinear Analysis, Pages 485 – 507.

[20] Meneveau, C. (1991). *Dual Spectra and Mixed Energy Cascade of Turbulence in the Wavelet Representation.* Physical Review Letters, Volume 66, Number 11, Pages 1450-1453.

[21] Meyer, Y. (1992). *Wavelets and Operators*. Cambridge University Press, New York, New York.

[22] Monasse, P. and Perrier, V. (1995). *Orthonormal Wavelet Bases adapted for Partial Differential Equations with boundary conditions*. SIAM Journal on Mathematical Analysis, Volume 29, Issue 4, pp. 1040-1065.

[23] Peterson, G. (1998). The Belousov-Zhabotinsky Reaction. Retrieved on October 24, 2006 from http://online.redwoods.cc.ca.us/instruct/darnold/deproj/Sp98/Gabe/

[24] Ruskeepää, H. (1998). *Mathematica Navigator Graphics and Methods of Applied Mathematics*. Academic Press, San Diego, California.

[25] Saprykin, S., Demekhin, E. A., and Kalliadasis, S. (2005). *Two-dimensional wave dynamics in thin films. I. Stationary solitary pulses.* Journal of Physics of Fluids, Volume 17, Article 117105.

[26] Schneider, K., Farge, M., Pellegrino, G., and Rogers, M. (2000). *CVS filtering of 3D turbulent mixing layers using orthogonal wavelets*. Retrieved on February 20, 2006 from http://wavelets.ens.fr/PUBLICATIONS/ARTICLES/PDF/135.pdf.

[27] Starck, J. L., Murtagh, F., and Bijaoui, A. (1998). *Image processing and data analysis: The multiscale approach*. Retrieved on September 12, 2006 from http://www.multiresolution.com/cupbook.pdf.

[28] Trevelyan, P. M. J., Saprykin, S., and Kalliadasis, S. (2005). *Falling films under complicated conditions*. Retrieved on March 27, 2006 from http://www3.imperial.ac.uk/pls/portallive/docs/1/3493901.PDF.

[29] Van Milligen, B., Sánchez, E., Estrada, T., Hidalgo, C., Brañas, B., Carreras, B., and García, L. (1995). *Wavelet bicoherence: A new turbulence analysis tool*. Retrieved on February 17, 2006 from http://bacterio.uc3m.es/investigacion/fusion/papers/php_luis_95.pdf.

[30] Xiong, G., Gunton, J. D., Xi, H., and Toral, R. (2006). *Space and Scale Localization in a Model of a Direct Transition to Spatiotemporal Chaos*. Retrieved on April 3, 2006 from http://arxiv.org/PS_cache/nlin/pdf/0105/0105036.pdf.

7. Appendix

In this part of the thesis some of the programs used and the calculations are shown and explained.

7.1 Brief descriptions of the Mathematica programs

The following programs are useful in computing the results for the formulas in the thesis. This appendix ends with a table summarizing the information about the programs I wrote.

wave

With this program I calculated the wavelet transform for the initial condition.

INPUT :
- wav - the wavelet
- fun - the function to be transformed by the wavelet
- x - variable with respect the integration will be done
- xNum - numerical value for the x variable
- α - variable for the integration in the Fourier Transform
- k - translation parameter
- k1 - minimum value for *k*
- k2 - maximum value for *k*
- b - dilation parameter
- b1 - minimum value for *b*
- b2 - maximum value for *b*

OUTPUT :
- Cψ - the normalization constant
- Wkb - the function transformed by the continuous wavelet
- grafkb - three - dimensional plot of $W_{k,b}$
- fRestored - the function restored for one numerical value

```
wave[wav_, fun_, x_, xNum_, α_, {k_, k1_, k2_}, {b_, b1_, b2_}] :=
    Module[{fou, sh, wkb, wavekb, wavkbN, grafkb, fRest},
            fou = Integrate[wav*Exp[-I*α*x], {x, -Infinity, Infinity}];
            sh = Integrate[Refine[Abs[fou], α ∈ Reals]^2/α, {α, 0, Infinity}];
            wavekb = wav /. x -> k*(x - b);
            wkb = Sqrt[k]*Integrate[fun*wavekb, {x, -Infinity, Infinity},
```

```
Assumptions -> {k > 0, b ∈ Reals}];
grafkb = Plot3D[wkb, {k, k1, k2}, {b, b1, b2}, PlotRange -> All, PlotPoints -> 200,
Mesh -> False, HiddenSurface -> True, ViewPoint -> {-0.5, -0.5, 1.0} ,
AxesLabel -> {"k", "b", "Wkb"}];
wavkbN = wav /. x -> k*(xNum - b);
mult = Simplify[wavkbN*Sqrt[k]*wkb/sh];
fRest = Integrate[mult, {k, 0, ∞}, {b, -∞, 0, ∞}];
Print["Cψ = ", sh];
Print["Wkb = ", wkb];
Print["fRestored = ", fRest];
grafkb]
```

intx

This is the function to calculate the integration with respect to x for λ and v of the wavelet transformation of the derivatives of K - S equation.

INPUT:
- k, K – translation parameters
- b, B – dilation parameters
- x – variable for which the integration is to be done
- n – amount of derivatives
- wav – the wavelet
- cf – normalization constant

OUTPUT:
- the integration with respect to x

```
intx[k_, K_, b_, B_, x_, n_, wav_, cf_] := Module[ {w, wpd, wd, mw, inte},
    w = wav /. x -> k*(x - b);
    wpd = wav /. x -> K*(x - B);
    wd = D[wpd, {x, n}];
    mw = Sqrt[k]*wd*w/cf;
    inte = Integrate[mw, {x, -∞, ∞}, Assumptions -> {k > 0, K > 0, b ∈ Reals, B ∈ Reals}]]
```

h and **L**

These are the basis functions that use Hermite and Laguerre polynomials.

INPUT:
n – degree of the polynomial of the basis functions
x – variable for the basis function
α – parameter for the Laguerre polynomial

OUTPUT:
basis function

```
h[n_, x_] := Exp[-x^2/2]*HermiteH[n, x]/Sqrt[(2^n*n!*Sqrt[Pi])];

L[n_, α_, x_] := Exp[-x/2]*x^(α/2)*LaguerreL[n, α, x]/Sqrt[(n + α)!/n!];
```

CoefsLin

This is the program to calculate the Galerkin's coefficients of the linear term.

INPUT:
intx – output obtained in the second program, λ
in, jn, pn, qn – numerical indices {i, j, p, q}

OUTPUT:
numerical coefficients of the linear term

```
CoefsLin[int_, {in_, jn_, pn_, qn_}] :=
  Module[{tabla},
      tabla = int/.{i->in, j->jn, p->pn, q->qn};
      ListIntegrate[tabla, 1]]
```

CoefsNonLin

This is the program to calculate the Galerkin's coefficients of the nonlinear term. As with the program explained before, to run this program it is necessary to download first the Mathematica package **NumericalMath** by means of the statement: Needs["NumericalMath`ListIntegrate`"].

INPUT:
tnl – expression for the nonlinear term, v
in, jn, pn, qn, rn, sn – numerical indices {i, j, p, q, r, s}

OUTPUT:
numerical coefficients of the nonlinear term

```
CoefsNonLin[tnl_, {in_, jn_, pn_, qn_,rn _, sn_}] :=
    Module[{tabla},
        tabla = int/.{i->in, j->jn, p->pn, q->qn, r->rn, s->sn};
        ListIntegrate[tabla, 1]]
```

tifb

This program integrates with respect to *b* the inverse transform of bases functions.

INPUT:
mn – degree of Laguerre polynomial
nn – degree of Hermite plolynomial

OUTPUT:
symbolical integration with respect to b of the inverse transform of bases functions

```
tifb[{mn_, nn_}] :=
  Module[{cf = 2 Sqrt[Pi], wx},
        wx = (x*Exp[-x^2/2]*Sqrt[2]*Pi^(-1/4)) /. x -> (k*(x - b));
        mu = ((Sqrt[k]/cf)*L[m, α, k]*h[n, b]*wx) /. {m -> mn, n -> nn};
        Integrate[mu, {b, -∞, ∞}, Assumptions -> {x ∈ Reals, k>0}]]
```

MyIntegral

This program finds the value of definite integrals by means of a procedure of numerical integration. It was used to find the expression for the inverse transform of bases functions in terms of *x*.

INPUT:
tk – output generated with the program *tifb* for each pair of indices
z – variable for which doing the integration
zm – minimum value for *z*
zM - maximum value for *z*
dz – step for variable *z*

OUTPUT:
The approximation of an integral.

```
MyIntegral[{z_, zm_, zM_, dz_}, f_] := Module[{tf, X, i, lt, si},
        tf = Table[f /. z -> X, {X, zm, zM, dz}];
        lt = Length[tf];
        ti = Table[(tf[[i]] + tf[[i + 1]])*dz/2, {i, 1, lt - 1}];
        si = Plus @@ ti;
        si]
```

simpson

This program is an alternative approach based on a program created to follow Simpson's algorithm for numerical integration. The parameters not explained are the same for both programs. The output is a solved integration too. It runs slower but the accuracy is bigger. It was used to integrate the parameter *k* of the initial condition.

INPUT:
n – amount of iterations, should be an even number
f – function to be integrated numerically
z – variable for which the integration is done
zm, zM – minimum and maximum values to do the integral

OUTPUT:
An approximate value for the integral.

```
simpson[n_, zm_, zM_, f_, z_] := Module[{s, h, k, y, m},
        m = n - 1;
        s = N[(f /. z -> zm) + (f /. z -> zM)];
        y = zm;
        k = 2;
        h = N[(zM - zm)/n];
        Do[y = y + h;
            k = 6 - k;
            s = s + k*N[f /. z -> y], {m}];
            s*h/3]
```

ODEs

This program was created to solve the system of ordinary differential equations.

INPUT:
n – maximum degree of the orthogonal polynomials
c – denotes the variable for the coefficients
lt – list of coefficients of the linear term
nlt – list of coefficients of the nonlinear term
cci - list of coefficients of the initial condition
st – list of indices
t – denotes the time
tm – minimum value for *t*
tM – maximum value for *t*

OUTPUT:
system's solution in terms of an interpolating function

```
ODEs[n_, c_, lt_, nlt_, cci_,st_,{t_, tm_, tM_}] :=
    Block[{vars = { }, indices = { }, ci = { }, difeqs = { },
      dimension = (n+1)^2,
      sol, g, k, a},
     Do[
       indices = Append[indices, st[[k]] /. st[[k]] -> k];
       vars = Append[vars, c[k]];
       ci = Append[ci, c[k][0] == cci[[k]]];
       difeqs = Append[difeqs,
       c[k]'[t] == Sum[lt[[k, g]]*c[g][t], {g, dimension}] +
                   Sum[nlt[[k, g, a]]*c[g][t]*c[a][t], {g, dimension}, {a, dimension}]],
      {k, dimension}];
     ecs = Chop[Join[difeqs, ci], 10^-6];
     sol = NDSolve[ecs, vars, {t, tm, tM}, MaxSteps -> 5000,
     StartingStepSize -> 1/100]]
```

sumasMP

This program was intended to create the movie plots of the solution for *t* values in a specified range. To run this program it is necessary to load firstly the Mathematica package **Animate**. It can be done by means of the statement Needs["Graphics`Animate`"]

INPUT:
n - maximum degree of the orthogonal polynomials

sol – the solution set generated with the program *ODEs*
t, x – variables in terms of which the solution is expressed
xm – minimum value for *x*
xM – maximum value for *x*
tm – minimum value for *t*
tM – maximum value for *t*
todas – set of inverse transform of bases functions

OUTPUT:
a movie plot of *u(x, t)*

```
sumasMP[n_, sol_, todas_, {x_, xm_, xM_}, {t_, tm_, tM_}] :=
        Module[{dimension, k, suma, tot},
                dimension = (n + 1)^2;
                suma = Table[(c[k][t] /. sol[[1]])*todas[[k]], {k, dimension}];
                tot = (Plus @@ suma);
                MoviePlot[Evaluate[tot], {x, xm, xM}, {t, tm, tM} , PlotRange -> {-1.5, 1.75},
                AxesLabel -> {"x", "u"}]]
```

GraficasX

This program does a plot of the solution for a specified *t* value.

INPUT:
n - maximum degree of the orthogonal polynomials
t, x – variables in terms of which the solution is expressed
tn – numerical *t* value
xm – minimum value for *x*
xM – maximum value for *x*
todas – set of inverse transform of bases functions
sol – the solution set generated with the program *ODEs*

OUTPUT:
a plot of *u(x, tn)*

```
GraficasX[n_, {t_, tn_}, {x_, xm_, xM_}, todas_, sol_] :=
        Block[{dimension, sum, tot},
                dimension = (n + 1)^2;
                sum = Table[(c[k][t] /. sol[[1]])*todas[[k]], {k, dimension}];
                tot = (Plus @@ sum) /. t -> tn;
                Plot[Evaluate[tot], {x, xm, xM}, PlotRange -> All]]
```

GraficasT

This program gives the plot of the solution for a specified *x* value.

INPUT:

n - maximum degree of the orthogonal polynomials
t, x – variables in terms of which the solution is expressed
xn – numerical *x* value
tm – minimum value for *t*
tM – maximum value for *t*
todas – set of inverse transform of bases functions
sol – the solution set generated with the program *ODEs*

OUTPUT:

a plot of *u(xn, t)*

```
GraficasT[n_, {x_, xn_}, {t_, tm_, tM_}, todas_, sol_] :=
    Block[{dimension, sum, tot},
    dimension = (n + 1)^2;
    sum = Table[(c[k][t] /. sol[[1]])*todas[[k]], {k, dimension}];
    tot = (Plus @@ sum) /. x -> xn];
    Plot[Evaluate[tot], {t, tm tM}, PlotRange->All]
```

This is the table which summarizes the information about the programs:

Program's name	What performs	Amount of lines
wave	C_Ψ , $W_{k,b}$, and f restoring. Plots $W_{k,b.}$	17
intx	formulas (3.10)	6
h and L	basis functions	2
CoefsLin	coefficients for the linear terms.	4
CoefsNonLin	coefficients for the nonlinear term.	4
tifb	integration with respect to *b* for the inverse transform	5
MyIntegral	integration with respect to *k* for the inverse transform	5
simpson	integration with respect to *k* for the inverse transform	10
ODEs	solve the system of ordinary differential equations	11
sumasMP	the movie plots of the solution	6
GraficasX	plots the solution for a *t* value	6
GraficasT	plots the solution for a *x* value	6

Table 6.1 Name, description and amount of lines of the Mathematica programs used.

7.2 The listings of the Mathematica programs

```
(*Mathematica packages to be loaded.*)

Needs["NumericalMath`ListIntegrate`"];
Needs["Graphics`Animation`"];

(*Input data*)
a=3; n=4; n2=25; n3=625; n=4; tStart=0.; α=a; tEnd=10.; tm=tStart; tM=tEnd;  xm=-8.;
tM2=50.; xM=8.; tM3=2.0;  xm1=-5.0;  xM1=5.; ind=st; km=0.1; Km=0.1;
kM=6.1; KM=6.1; dk=0.1; dK=0.1; n4=15625;
cf=2 Sqrt[Pi]; (*admissibility constant*)
wx=(x*Exp[-x^2/2]*Sqrt[2]*Pi^(-1/4));(*the Gaussian wave*)


(*Values that was already calculated and that can be loaded before running any other program
that needs this information.*)

<<"C:\\Tesis\\ind.txt";(*The 25 indices of the inverse transform of bases functions.*)

<<"C:\\Tesis\\ind4.txt";(*The 625 indices for the linear term's coefficients.*)

<<"C:\\Tesis\\ind6.txt";(*The 15625 indices for the nonlinear term's coefficients.*)

<<"C:\\Tesis\\sd24.txt";(*λ*)

<<"C:\\Tesis\\itnln.txt";(*ν*)

<<"C:\\Tesis\\ttt.txt";(*The 15625 coefficients of the nonlinear term.*)

<<"C:\\Tesis\\cintk.txt";(*The 25 inverse transform of bases functions.*)

<<"C:\\Tesis\\pctftit.txt";(*The 625 coefficients of the linear term.*)

<<"C:\\Tesis\\coni.txt";(*The 25 coefficients for the initial condition.*)

<<"C:\\Tesis\\solu.txt";(*The result of NDSolve in terms of InterpolatingFunction.*)


(*Short program to create a table of value which compares the solution when t=0, with the initial
condition.*)

TablaX[n_, {t_,tn_},{x_,xm_,xM_,dx_},todas_,sol_]:=Block[{dimension,sum,tot,ic},
```

```
dimension=(n+1)^2;sum=Table[(c[k][t]/.sol[[1]])*todas[[k]],{k,dimension}];
ic=Exp[-x^2/2]*Sin[x]//N;tot=(Plus@@sum)/.t→tn;
TableForm[Table[Chop[{x,ic,tot,Abs[ic-tot]},10^-6],{x,xm,xM,dx}],
TableHeadings→{None,{"x","Initial Condition","Solution for t=0","Absolute Error"}}]]

(*Short program to create the three-dimensional plot of the last figure of the thesis.*)

sumas[n_,sol_, {x_,xm_,xM_},{t_,tm_,tM_}]:=Module[{dimension,k,suma,tot},
   dimension=(n+1)^2;suma=Table[(c[k][t]/.sol[[1]])*cintk[[k]],{k,dimension}];

tot=(Plus@@suma);Plot3D[Evaluate[tot],{x,xm,xM},{t,tm,tM},PlotRange→All,Mesh→False,A
xesLabel→{"x","t","u(x,t)"}]]
(*Short program to expand the initial condition into the basis functions.*)

expan[{m_,n_}]:=Integrate[wkbsen L[m,3,k] h[n,b],{b,-∞,∞},Assumptions→k>0];

(*Running the programs and calculations.*)

(*Work done with the indices*)
ind=Flatten[Table[{i,j},{i,0,n},{j,0,n}],1];
ind4=Flatten[Table[{i,j,p,q},{i,0,n},{j,0,n},{p,0,n},{q,0,n}],1];
ind6=Flatten[Table[{i,j,p,q,r,s},{i,0,n},{j,0,n},{p,0,n},{q,0,n},{r,0,n},{s,0,n}],1];

(*Work done with the inverse transform of basis functions*)
intb=Table[tifb[ind[[p]]],{p,n2}];
intK=Table[simpson[50,0,15,intb[[p]],x],{p,n2}];
cintk=Chop[intK,10^-6];

(*Work done with the initial condition*)
wav=Sqrt[2] x Exp[-x^2/2]/Pi^(1/4); fun=TrigToExp[Exp[-x^2/2] Sin[x/10]];
wave[wav, fun, x, 0, α, {k, 0.1, 5.0}, {b, -5.0, 5.0}]
Plot[fun, , {x, -1.25 Pi, 1.25 Pi}], AxesLabel->{“x”, ”u(x, 0) = Exp[-x^2/2] Sin[x/10]”},
PlotRange->All]; tex=Table[expan[ind[[g]]], {g, 25}];
sol=Refine[ComplexExpand[wkbsen], (k>0 && b Є Reals)]
 Simplify[sol]
coni=Table[simpson[100,0,50,tex[[g]], k], {g, 1, 25}]
Table[Graphics[{Hue[i/21], PointSize[0.02], Point[{i, coni[[i]]}]}, Axes->True,
AxesLabel->{None, “InitialCondition”}], {i, 25}]//Show

(*Work done to find λ*)
sd2=intx[k,K,b,B,x,2,wx,cf];
sd4=intx[k,K,b,B,x,4,wx,cf];
sd24=sd2+sd4;
```

```
(*Work done to find the coefficients of the linear term.*)
m=sd24 L[i,a,k] h[j,b] L[p,a,K] h[q,B];
mk=MyIntegral[{k,km,kM,dk},m];
mK=MyIntegral[{K,Km,KM,dK},mk];
tm=Table[mK,{b,-5,5},{B,-5,5}];
ftm=Flatten[tm];fc1625=Table[CoefsLin[ind4[[d]],ftm],{d,n3}];
cfc1625=Chop[-fc1626,10^-6];

(*Work done to find ν*)
w1=wx/.x→k*(x-b);
w2=wx/.x→κ*(x-β);
w3=wx/.x→K*(x-B);
constnl=(2*Sqrt[k]/cf^2);
tnl=w2*w3*D[w1,x]*constnl;
itnln=Integrate[tnl,{x, -∞,∞},Assumptions→{k>0,κ>0,K>0,b ε Reals, β ε Reals, B ε Reals}];

(*Work done to find the coefficients of the nonlinear term*)
m=itnln L[i,a,k] h[j,b] L[p,a,K] h[q,B]*L[r,α,κ]*h[s,β];
mk=MyIntegral[{k,km,kM,dk},m];tk=Table[mk,{b,-5,5},{B,-5,5},{β,-
5,5}];tnk=Flatten[tk];tκK=Flatten[Table[{κ,K},{κ,0.1,5.1,1.0},{K,0.1,5.1,1.0}],1];
la=Length[tκK];
se[{κn_,Kn_}]:=tnk/.{κ→κn,K→Kn}
s1=se[tκK[[1]]]; s2=se[tκK[[2]]];s3=se[tκK[[3]]];s4=se[tκK[[4]]];s5=se[tκK[[5]]];
s8=se[tκK[[8]]];s6=se[tκK[[6]]];s7=se[tκK[[7]]];s9=se[tκK[[9]]];s10=se[tκK[[10]]];
s11=se[tκK[[11]]];s12=se[tκK[[12]]];s13=se[tκK[[13]]];s14=se[tκK[[14]]];s15=se[tκK[[15]]];
s16=se[tκK[[16]]];s17=se[tκK[[17]]];s18=se[tκK[[18]]];s19=se[tκK[[19]]];s20=se[tκK[[20]]];
s21=se[tκK[[21]]];s22=se[tκK[[22]]];s23=se[tκK[[23]]];s24=se[tκK[[24]]];s25=se[tκK[[25]]];
s26=se[tκK[[26]]];s27=se[tκK[[27]]];s28=se[tκK[[28]]];s29=se[tκK[[29]]];s30=se[tκK[[30]]];
s31=se[tκK[[31]]];s32=se[tκK[[32]]];s33=se[tκK[[33]]];s34=se[tκK[[34]]];s35=se[tκK[[35]]];
s36=se[tκK[[36]]];
ctotnl={s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19,s20,s21,s22,s23,s24,s
25,s26,s27,s28,s29,s30,s31,s32,s33,s34,s35,s36};
cjtotnl=Flatten[ctotnl];fc115625=Table[CoefsNonLin[ind6[[d]],cjtotnl],{d,n4}];
fc115625c=Chop[fc115625,10^-
6];tt=Partition[fc115625c,625];ttt=Table[Partition[tt[[o]],25],{o,25}];

(*Work done to organize all the calculations done before in order to find the solution and
represent it in graphs and a table.*)

sol1=ODEs[n,c,pctftit,ttt,coni,st,{t,tm,tM}];
(*This is for the movie plot.*)
sumasMP[n,solf, {x,-8,8},{t,tm,tM}];

(*This is to do the graph for different t values.*)
```

```
sol2=ODEs[n,c, pctftit,ttt,coni,st,{t,tm,tM2}];
txn=Table[GraficasT[4, {x, nn}, {t, 0, 50}, cintk, sol2, nn-1], {nn, 2, 4, 1}]
te1=Graphics[{Hue[1], Text[“x=2.0”,{50, 0.015}]}]
te5=Graphics[{Hue[1/2], Text[“x=3.0”, {50, 0.010}]}]
te10=Graphics[{Hue[1/3], Text[“x=4.0”, {50, 0.005}]}]
Show[{txn, te1, te5, te10}]
(*This is to create the table which serves to validate the results.*)
TablaX[n,{t,0},{x,0,2,0.2},cintk,sol1];

(*This is to do the plot for different values of time x.*)
ts=Table[GraficasX[4, {t, tn}, {x, -8,8}, cintk, sol1, tn], {tn, 3, 9, 3}]
te1= Graphics[{Hue[1/3], Text[“t=3.0”, {8, -0.003}]}]
te5=Graphics[{Hue[1/6], Text[“t=6.0”, {8, -0.005}]}]
te10=Graphics[{Hue[1/9], Text[“t=9.0”, {8, -0.007}]}]
Show[{ts, te1, te5, te10}]

(*This is to do the last three-dimensional plot.*)
sumas[n,sol1,{x,xm1,xm2},{t,tm,tM3}]

(*Important parts to be saved.*)

Save["C:\\Tesis\\ind.txt",ind];
Save["C:\\Tesis\\ind4.txt",ind4];
Save["C:\\Tesis\\ind6.txt",ind6];
Save["C:\\Tesis\\sd24.txt",sd24];
Save["C:\\Tesis\\itnln.txt",itnln];
Save["C:\\Tesis\\coni.txt",coni];
Save["C:\\Tesis\\cintk.txt",cintk];
Save["C:\\Tesis\\ pctftit.txt", pctftit];
Save["C:\\Tesis\\ttt.txt",ttt];
Save["C:\\Tesis\\solu.txt",solu];
```

7.3 A CD-ROM with information

In the attached CD-ROM, there are Mathematica programs, articles, and some web links related to the thesis investigation.