**Internship - WI5118**
Report

# Identification of Spines in Nonlinear Fourier Spectra for the Periodic Nonlinear Schrödinger Equation

Submitted by

**Christos Kitsios**
MSc Student Applied Mathematics
Delft University of Technology

Under the guidance of

**Sander Wahls**
Associate Professor
Delft Center for Systems & Control, 3mE
Delft University of Technology

Delft University of Technology

February 2022

**Abstract**

The nonlinear Fourier transform for the focusing periodic nonlinear Schrödinger equation is investigated. This paper is focused on the approximation of the spines in the nonlinear spectrum using results from Floquet theory. Algorithms for the numerical computation of the spines based on the Fourier collocation method are being examined and a new algorithm is presented. The new algorithm developed during the project computes the spines by tracking sign changes of the function $\Im(\Delta(\cdot))$ in the area $|\Re(\Delta(\cdot))| < 1$, where $\Delta$ is the Floquet discriminant. The new algorithm is successfully applied to examples where both the modified Fourier collocation method and the method implemented in the FNFT software library fail. In addition, the spine points that are numerically computed by the new algorithm are equally distributed along the curve, while using the other algorithms the computed points are clustered around the periodic eigenvalues. Finally, the algorithm provides information on which spectrum points belong to the same spine. The pseudocode and the MATLAB source code of the algorithm developed are provided.

## Acknowledgment

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Programme (grant agreement No 716669).

# Contents

# Chapter 1

# Introduction

Zakharov [1] initially derived the Nonlinear Schrödinger Equation (NSE)

$$i\partial_t q + \partial_{xx} q + 2\kappa|q|^2 q = 0, \tag{1.1}$$

where $i$ is the imaginary unit, $\partial$ is the partial derivative and $q : \mathbb{R} \times \mathbb{R}_{\geq 0} \to \mathbb{C}$ is a complex valued function of the location $x \in \mathbb{R}$ and time $t \geq 0$, as a model for waves in a deep fluid. He used Stokes' finding of an amplitude dependent correction to the dispersion of surface water waves, which determines the non-linear behaviour of the equation. The NSE has also been derived in various physical settings. Another physical phenomena for which the NSE (1.1) provides a model is the evolution of a waveform in an optical fiber [2]. We say that a signal $q : \mathbb{R} \times \mathbb{R}_{\geq 0} \to \mathbb{C}$ is a solution of the periodic NSE if it is governed by the NSE (1.1) and is subject to the periodic boundary condition

$$q(x, t) = q(x + \ell, t), \tag{1.2}$$

where $\ell > 0$ is the spatial period. If a signal $q : \mathbb{R} \times \mathbb{R}_{\geq 0} \to \mathbb{C}$ is governed by the NSE (1.1) and

$$|q(x, t)| \to 0, \tag{1.3}$$

fast enough for $|x| \to \infty$, then we say that $q$ is a solution of the vanishing NSE.

The real coefficient $\kappa \in \{\pm 1\}$ of the non-linear term of the NSE (1.1) determines the nature of the equation as focusing ($\kappa = +1$) or defocusing ($\kappa = -1$). The former case describes a fiber with anomalous dispersion and can be solved by bright solitons, localised waves that remain unchanged after interactions with other bright solitons, while the latter case corresponds to a fiber with normal dispersion with dark solitons as a solution, which are localised waves that cause a decrease in the wave amplitude [3]. Gardner et al. [4] introduced and developed a new method called inverse scattering method for the solution of the Korteweg-de Vries equation, which is another partial differential equation that models e.g. waves in shallow water. Extending the inverse scattering method, Zahkarov and Shabat solved the NSE for non-periodic signals with vanishing boundary conditions [5]. The NSE belongs to a class of evolution equations that are integrable by the inverse scattering method [6].

However, as a result of the mathematical complexity of the method, and the lack of efficient numerical methods for its computation, the inverse scattering method has not been widely used for the practical analysis of non-linear data in the past. In the last years, however several new highly efficient algorithms have been developed. The transformations used in the inverse scattering method can be thought of as a generalization of the Fourier transforms used in the analysis of linear evolution equations [6]. In the inverse scattering method, the signal is represented, by the forward scattering transform, in the form of scattering data which play the role of Fourier coefficients [6, 7]. Whenever the amplitude of the signal becomes small, the scattering data essentially reduces to the Fourier transform of the signal [8]. The inverse scattering method can therefore be interpreted as a nonlinear Fourier analysis and the scattering transforms as nonlinear Fourier transforms (NFT).

In Chapter 2 the underlying mathematical theory for the derivation of the nonlinear Fourier transform for the periodic nonlinear Schrödinger equation and significant lemmas regarding the computation of the spectrum will be presented. Algorithms that are found in the literature, as well as the algorithm developed during the project for the numerical computation of the spines are presented in Chapter 3. In the Appendix the pseudocodes and the MATLAB source code of different versions of the new algorithm can be found.

## 1.1 Problem Description and Objectives

During my second year following the MSc Programme Applied Mathematics, I worked as a project intern for three months on the project "Identification of Spines in Nonlinear Fourier Spectra for the Periodic Nonlinear Schrödinger Equation" under the supervision of Associate Professor Sander Wahls. The main goals of the project were: to compute spines, which are curves in the nonlinear spectrum in the spectrum faster than the current method used and to identify which points of the nonlinear spectrum are on the same spine.

## 1.2 Delft Center for Systems and Control

The Delft Center for Systems and Control (DCSC) in the faculty of Mechanical, Maritime and Materials Engineering (3mE) at Delft University of Technology was founded in 2003 and consists of four scientific sections, namely *Numerics for Control and Identification*, *Optimization and Learning for Control of Networks*, *Data-Driven Control*, and *Networked Cyber-Physical Systems* [9]. DCSC takes part in the national research school Dutch Institute of Systems and Control, contributing to research on a wide range of subjects within the spectrum of systems and control. In addition the center administrates the educational activities in the area of systems and control at the university, aiming to prepare students to become independent and highly-skilled professionals.

### 1.2.1 Nonlinear Fourier Transforms in Action

Since linear systems are well understood and can be solved analytically and numerically efficiently, engineers often treat nonlinear effects using linear approximation. However, there are engineering problems containing nonlinear effects that cannot be approximated linearly. Nonlinear Fourier transforms can be used to solve many of these problems in a similar way that the standard Fourier transform is used in the study of linear systems. Although nonlinear Fourier transforms have been extensively studied by mathematicians and physicists, since the initial use of a nonlinear Fourier transform by Gardner et al.

[4], nonlinear Fourier transforms are not yet widely used in engineering, with one reason being that their efficient numerical computation is a challenging problem [10].

Associate Professor Sander Wahls and his research group are currently working on the project Nonlinear Fourier Transforms in Action (NEUTRINO) [10], funded by a Starting Grant from the European Research Council. The goals of the NEUTRINO project are the development of efficient algorithms that are made available in an open source software library and the use of these algorithms in the areas of fiber-optic communications and water wave data analysis, in order to make nonlinear Fourier transforms a widespread and practical engineering tool.

# Chapter 2

# Review of the nonlinear Fourier transform for the periodic nonlinear Schödinger equation

In this chapter, we present the mathematical theory that was studied during the project, in order to have the necessary results for the numerical computation of the spines. Initially, the class of finite gap solutions for the NSE (1.1) and the scattering data are presented. After the derivation of the NFT, we use the Floquet theory to state several lemmas for the computation of the spectrum. Finally, we briefly discuss the FNFT library that was developed during the NEUTRINO project.

## 2.1 Finite Genus Solutions and the Nonlinear Fourier Transform

### 2.1.1 Scattering Data

Finite gap (or band) solutions are a special class of solutions of the NSE (1.1) [11]. Solutions that belong in this class depend on a finite number of parameters that constitute the scattering data, and can be compared to the linear Fourier series expansion with only finitely many non-zero terms. A sufficiently smooth periodic solution of the NSE (1.1) can be approximated over a finite space interval by finite gap solutions with an arbitrary small error [12]. Two different types of finite gap solutions can be found in the literature and were given by Kotlyarov and Its [13], and Ma and Ablowitz [14]. For the purpose of this paper, the former type of finite gap solutions will be presented.

A solution $q(x,t)$ to the NSE (1.1) is a finite gap solution, in the sense of Kotlyarov and Its [13], if there exist finitely many of the parameters [7]

$$\begin{aligned}
&\text{Main spectrum:} && \lambda_1,\, \lambda_2,\, \dots \lambda_{2N} \,\in\, \mathbb{C}, \\
&\text{Initial Auxiliary spectrum:} && \mu_j(x_0,t_0) \,\in\, \mathbb{C}\,,\; j=1,\,\dots,\,N-1, \\
&\text{Initial amplitude:} && q(x_0,t_0) \,\in\, \mathbb{R}_{\geq 0}, \\
&\text{Riemann sheet indices:} && \sigma_j(x_0,t_0) \,\in\, \{\pm 1\}\,,\; j=1,\,\dots,\,N-1
\end{aligned} \tag{2.1}$$

such that $q(x,t)$ is generated by the following system of coupled partial differential equations [7]

$$\partial_x \ln q = 2i\Big(\sum_{j=1}^{N-1}\mu_j - \frac{1}{2}\sum_{k=1}^{2N}\lambda_k\Big), \tag{2.2}$$

$$\begin{aligned}\partial_t \ln q = {} & 2i\Big(\sum_{j,k=1,j>k}^{2N}\lambda_j\lambda_k - \frac{3}{4}\Big(\sum_{k=1}^{2N}\lambda_k\Big)^2\Big)\\ & + 4i\Big(\frac{1}{2}\Big(\sum_{k=1}^{2N}\lambda_k\Big)\Big(\sum_{j=1}^{N-1}\mu_j\Big) - \sum_{j,k=1,j>k}^{N-1}\mu_j\mu_k\Big),\end{aligned} \tag{2.3}$$

$$\partial_x \ln \mu_j = \frac{-2i\sigma_j\sqrt{\prod_{k=1}^{2N}(\mu_j-\lambda_k)}}{\prod_{\substack{m=1\\m\neq j}}^{N-1}(\mu_j-\mu_m)} \quad ,\ j=1,\ ...,\ N-1, \tag{2.4}$$

$$\partial_t \ln \mu_j = -2\Big(\sum_{\substack{m=1\\m\neq j}}^{N-1}\mu_m - \frac{1}{2}\sum_{k=1}^{2N}\lambda_k\Big)\partial_x\mu_j \quad ,\ j=1,\ ...,\ N-1, \tag{2.5}$$

For the periodic NSE the parameters in (2.1) represent the scattering data. Note that $q(x,t)$ can be recovered completely from the scattering data by using it as initial conditions for (2.2)-(2.5). It should also be mentioned that not all combinations of (2.1) result in a periodic solution of the problem (1.1)-(1.2). The main spectrum $\{\lambda_k\}_{k=1}^{2N}$ is defined as the set of the periodic and anti-periodic eigenvalues and is independent of space $x$ and time $t$. On the other hand the auxiliary spectrum $\{\mu_j(x,t)\}_{j=1}^{N-1}$ depends on $x$ and $t$. Also, it can be shown that the main spectrum is symmetric to the real axis.

### 2.1.2 Derivation of Nonlinear Fourier Transform

Let $q(x,t)$ be a signal governed by the NSE (1.1) and fix a reference time $t_0$. The NFT is a transform that maps the signal $q(\cdot,t_0)$ to the scattering data. Before we explain how to compute the scattering data using the NFT, we introduce the notion of Lax pairs. Two operators $S$ and $T$ are said to form a Lax pair if they satisfy $\partial_t S = [T,S] = TS - ST$ and the equation $\partial_t S = [T,S]$ is equivalent to a partial differential equation of interest. It can be shown that the operator $S$ has the isospectral property, i.e. the spectrum of the operator $S$ is independent of $t$, and that the time evolution of any eigenfunction $\phi$ of $S$ is given by $\partial_t\phi = T\phi$ [15]. For the differential operator

$$L_{t_0} = i\begin{bmatrix}\frac{d}{dx} & q(.,t_0)\\ \kappa\bar{q}(.,t_0) & -\frac{d}{dx}\end{bmatrix}, \tag{2.6}$$

a suitable differential operator $B$ can be found such that $L_{t_0}$ and $B$ form a Lax pair [5] for the NSE (1.1).

Consider the eigenvalue problem

$$L_{t_0}v = \zeta v, \tag{2.7}$$

which is equivalent to

$$\frac{d}{dx}v = \begin{bmatrix} -i\zeta & -q(\cdot,t_0) \\ \kappa\bar{q}(\cdot,t_0) & i\zeta \end{bmatrix} v. \tag{2.8}$$

Note that the matrix in (2.7) acts on a two component wave function $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$. One can compute the quasi-periodic eigenvalues in the problem (2.8) with the use of Floquet Theory [16], which studies the analysis of differential equations with periodic coefficients. Suppose $\phi_{x_0,t_0,\zeta}$ and $\tilde{\phi}_{x_0,t_0,\zeta}$ are solutions of (2.8), subject to the canonical initial conditions

$$\phi_{x_0,t_0,\zeta}(x_0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tilde{\phi}_{x_0,t_0,\zeta}(x_0) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{2.9}$$

The monodromy matrix is defined as

$$\begin{aligned} M_{x_0,t_0}(\zeta) :&= \begin{bmatrix} \phi_{x_0,t_0,\zeta}(x_0+\ell) & \tilde{\phi}_{x_0,t_0,\zeta}(x_0+\ell) \end{bmatrix} & (2.10) \\ &= \begin{bmatrix} \phi^1_{x_0,t_0,\zeta}(x_0+\ell) & \tilde{\phi}^1_{x_0,t_0,\zeta}(x_0+\ell) \\ \phi^2_{x_0,t_0,\zeta}(x_0+\ell) & \tilde{\phi}^2_{x_0,t_0,\zeta}(x_0+\ell) \end{bmatrix} & (2.11) \end{aligned}$$

and represents the evolution of the solutions given by (2.9) over one period $\ell$ [7]. Using the monodromy matrix, the Floquet discriminant $\Delta$ is defined as

$$\Delta(\zeta) := \frac{1}{2}\operatorname{Tr}(M_{x_0,t_0}(\zeta)), \tag{2.12}$$

where $\operatorname{Tr}(\cdot)$ denotes the matrix trace. With the use of Floquet Theory, solutions of the eigenproblem (2.7) with the property

$$v(x+\ell,\zeta) = e^{ip(\zeta)}v(x,\zeta), \tag{2.13}$$

can be found. Such solutions are called Bloch solutions of (2.7) and $p(\zeta)$ is called the Floquet exponent or quasi-momentum [16]. Bloch solutions can be expressed as a linear combination of the standard solutions determined by (2.9)

$$v(x,\zeta) = A\phi_{x_0,t_0,\zeta}(x,\zeta) + B\tilde{\phi}_{x_0,t_0,\zeta}(x,\zeta). \tag{2.14}$$

Using the initial conditions (2.9) we have

$$v(x_0,\zeta) = \begin{bmatrix} A \\ B \end{bmatrix}. \tag{2.15}$$

Combining (2.15) and (2.13) at $x_0$ we have [16]:

$$m(\zeta)\begin{bmatrix} A \\ B \end{bmatrix} = M_{x_0,t_0}(\zeta)\begin{bmatrix} A \\ B \end{bmatrix}, \quad m(\zeta) = e^{ip(\zeta)}. \tag{2.16}$$

From the previous equation we can construct the Bloch functions and the Floquet exponents. The following lemma provides the connection between the Floquet exponents and the Floquet Discriminant.

**Lemma 2.1.1.** *[7] Fix $\zeta, m \in \mathbb{C}$. The eigenproblem (2.7) admits a quasi-periodic solution $v \neq 0$, i.e. $v(x+\ell) = mv(x)$ if and only if the Floquet discriminant $\Delta(\zeta)$ satisfies*

$$m^2 - 2m\Delta(\zeta) + 1 = 0 \tag{2.17}$$

Results that allow us to identify the main and auxiliary spectrum with the help of monodromy matrix and the Floquet discriminant are now presented. Regarding the main spectrum, the main result is Lemma 2.1.2.

**Lemma 2.1.2.** *[7] Assume that there exist a finite gap solution $q(x,t)$ and fix it. Let $\{\lambda_k\}_k$ be a main spectrum for $q(t,x)$ such that $\{\lambda_k\}_k$ does not contain any point more than once, i.e. $\lambda_j \neq \lambda_j$, $\forall\, i \neq j$. Moreover, assume that the roots of $1-\Delta^2$ are at most double. Then the main spectrum corresponds exactly to the simple roots of $1-\Delta^2$, i.e.*

$$\{\lambda_k\}_{k=1}^{2N} = \left\{\zeta \in \mathbb{C} : \Delta(\zeta) \in \{\pm 1\}, \frac{d\Delta}{dz}(\zeta) \neq 0\right\}$$

The main spectrum thus corresponds to the eigenvalues with periodic and anti-periodic eigenfunctions. It can furthermore been shown that the equation $1-\Delta^2$ has only finitely many non-simple non-real roots [16].

**Lemma 2.1.3.** *[7] Let $q(x_0,t_0) \neq 0$ and suppose we have the same assumptions of the previous lemma (2.1.2). Define*

$$\eta(\zeta) = \begin{cases} 1, & \text{if } \zeta \in \mathbb{C} \text{ is a double root of } 1-\Delta^2 \\ 0, & \text{otherwise.} \end{cases}$$

*Then the auxiliary spectrum is given by*

$$\{\mu_j(x_0,t_0)\}_{j=1}^{N-1} = \left\{\zeta \in \mathbb{C} : \lim_{z\to\zeta} \frac{[M_{x_0,t_0}]_{1,2}}{(z-\zeta)^{\eta(\zeta)}} = 0\right\}.$$

The points in the main spectrum form pairs that are connected by curves in the complex plane known as spines or bands in the literature [17]. The spines consist of quasi-periodic eigenvalues and usually connect two or more points of simple spectrum, i.e. complex points in the main spectrum, guaranteeing that the Bloch eigenfunctions are stable [18]. While spines are not required to reconstruct the signal $q(x,t)$, they can provide information about its behaviour. When a spine connects two points of the simple spectrum we call this combination of spectral information a non-linear mode, and we can further classify the mode based on whether the spines crosses the real axis or not [18, 19]. In the former case we have a stable sine wave or a Stokes wave, while in the latter case we have two phase-locked unstable NSE Stokes wave or a breather. Furthermore, in the rarer case of three or more points connected by the spine which does not cross the real axis, the mode is classified as a superbreather in the spectrum [19]. Using Lemma 2.1.1, it can be shown that the spines are the curves in the complex plane defined by the equations

$$\Im[\mathrm{Tr}(M_{x_0,t_0})/2] = 0\,, \qquad -1 \leq \Re[\mathrm{Tr}(M_{x_0,t_0})/2] \leq 1 \tag{2.18}$$

$$\text{or equivalently} \quad \Im[\Delta(\zeta)] = 0\,, \qquad -1 \leq \Re[\Delta(\zeta)] \leq 1 \tag{2.19}$$

Since the system of equations (2.19) provides the spines, a method to numerically compute the Floquet discriminant will now be discussed. Recall that the eigenvalue problem (2.7) is equivalent to the problem (2.8)

$$\frac{d}{dx}v = \begin{bmatrix} -i\zeta & -q(\cdot,t_0) \\ \kappa\bar{q}(\cdot,t_0) & i\zeta \end{bmatrix} v.$$

Divide the period $\ell$ into $M$ intervals of length $D_x = \ell/M$ and choose $x_n$ such that the intervals are given by $[x_n - \frac{D_x}{2}, x_n + \frac{D_x}{2}]$. Approximate $q(x,0)$ by a step function

$$q(x,0) \approx q_n := q(x_n,0),\ \forall x \in [x_n - \frac{D_x}{2}, x_n + \frac{D_x}{2}].$$

By direct integration of (2.8) we get

$$v(x_n + D_x,\zeta) = U(q_n)v(x_n,\zeta), \tag{2.20}$$

where $U(q_n) = \exp\left(D_x\left(\begin{bmatrix} -i\zeta & -q_n \\ \kappa\bar{q}_n & i\zeta \end{bmatrix}\right)\right)$ [8]. Now introducing $\Xi = \begin{bmatrix} v \\ \partial_\zeta v \end{bmatrix}$, we have the recursion relation $\Xi(x_n + D_x) = T(q_n)\Xi(x_n)$, where $T(q_n) = \begin{bmatrix} U(q_n) & 0 \\ \partial_\zeta U(q_n) & U(q_n) \end{bmatrix}$ and by the discretization of $q(x,0)$

$$\Xi(x_n) = \prod_{j=0}^{n} T(q_j)\Xi(x_0). \tag{2.21}$$

Then computing the matrix

$$S(\zeta) = \prod_{j=0}^{M-1} T(q_j) = \begin{bmatrix} \Sigma(\zeta) & 0 \\ \partial_\zeta\Sigma(\zeta) & \Sigma(\zeta) \end{bmatrix}, \tag{2.22}$$

we have that $\Sigma(\zeta) = \prod_{j=0}^{M-1} U(q_j)$ is a numerical approximation of the monodromy matrix [8]. Furthermore, it can be proved that $\partial_\zeta\Sigma$ approximates the derivative of the monodromy matrix with respect to $\zeta$ [8]. Hence, the Floquet discriminant can be approximated by $\Delta(\zeta) \approx \frac{1}{2}\operatorname{Tr}(\Sigma(\zeta))$ and the derivative of the Floquet Discriminant $\frac{\partial\Delta}{\partial\zeta}$ can be approximated by $\frac{\partial\Delta}{\partial\zeta}(\zeta) \approx \frac{1}{2}\operatorname{Tr}(\partial_\zeta\Sigma(\zeta))$.

## 2.2 FNFT library

The lack of an efficient and reliable software for the numerical computation of Nonlinear Fourier Transforms (NFT) has led Wahls et al. to publish the software library FNFT (Fast Nonlinear Fourier Transforms) during the NEUTRINO project [20]. The library is an open source software available on GitHub [21]. It is mostly focused on fast algorithms, but also provides nonfast algorithms. FNFT was the first publically available software library for the implementation of numerical NFT [20].

During the internship the algorithms that can be found in the library were studied, with focus on the computations of the main and auxiliary spectrum of the NSE with periodic boundary conditions (1.1). In addition, version 0.4.1 of the software [22] was used as a reference point while implementing new fast algorithms for the problem in Section 1.1.

# Chapter 3

# Numerical computation of spines

In this chapter, we initially present algorithms that are found in the literature for the numerical computation of the spines. Ultimately, the algorithm developed during the project is presented.

## 3.1 Existing algorithms

### 3.1.1 Numerical computation with the FNFT library

For the approximation of the spines the current method implemented in the FNFT initially subsamples the signal given by $q(x, t_0)$. Then, a rational approximation of the Floquet discriminant $\Delta$ is computed and a fast polynomial eigensolver is used in order to find initial guesses for the spine points. The initial guesses are refined with the use of the Newton Raphson method for the Floquet discriminant $\Delta$. The functions $\Delta$ and $\frac{\partial \Delta}{\partial \zeta}$ are numerically computed from the full signal using (2.22) or, alternatively, high-order variants for the refinement.

### 3.1.2 Fourier Collocation Method

The Fourier collocation method converts the Zakharov-Shabat problem for the periodic NSE (1.1), (1.2) to a matrix eigenvalue problem [23]. Hence solving the matrix eigenvalue problem provides the periodic spectrum. The eigenproblem for the first operator in the Lax pair that is equivalent to the problem (1.1) can be written as

$$L_{t_0} v = \zeta v, \; L_{t_0} = i \begin{bmatrix} \frac{d}{dx} & q(.,t_0) \\ \kappa \overline{q}(.,t_0) & -\frac{d}{dx} \end{bmatrix}, \; v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tag{3.1}$$

For the Fourier collocation method the components of the periodic eigenfunction $v$ as well as the signal $q(\cdot, t_0)$ are expanded as Fourier series (using the periodicity) [23]

$$v_1(x) = \sum_n \alpha_n e_n(x) \; , \; v_2(x) = \sum_n \beta_n e_n(x) \; , q(x, t_0) = \sum_n \gamma_n e_n(x) \; , \tag{3.2}$$

where $e_n$ denotes the exponential function $e_n(x) := e^{2n\pi x i/\ell}$. By truncating the number of

Fourier modes in (3.2) and substituting in (2.8), we obtain

$$\begin{aligned} & i\begin{bmatrix} \frac{\partial}{\partial x} & q(.,t_0) \\ \kappa\overline{q}(.,t_0) & -\frac{\partial}{\partial x}\end{bmatrix} v = \zeta v \\ \iff & i\begin{bmatrix} \frac{\frac{\partial}{\partial x}}{\kappa\sum_{n=-M}^{M}\gamma_n e_n(x)} & \sum_{n=-M}^{M}\gamma_n e_n(x) \\ & -\frac{\partial}{\partial x}\end{bmatrix}\begin{bmatrix}\sum_{n=-M}^{M}\alpha_n e_n(x) \\ \sum_{n=-M}^{M}\beta_n e_n(x)\end{bmatrix} = \zeta\begin{bmatrix}\sum_{n=-M}^{M}\alpha_n e_n(x) \\ \sum_{n=-M}^{M}\beta_n e_n(x)\end{bmatrix} \\ \iff & i\begin{bmatrix} \frac{\partial}{\partial x}\big(\sum_{n=-M}^{M}\alpha_n e_n(x)\big) + \kappa\big(\sum_{n=-M}^{M}\gamma_n e_n(x)\big)\big(\sum_{n=-M}^{M}\beta_n e_n(x)\big) \\ \kappa\big(\sum_{n=-M}^{M}\overline{\gamma_{-n}}e_n(x)\big)\big(\sum_{n=-M}^{M}\alpha_n e_n(x)\big) - \frac{\partial}{\partial x}\big(\sum_{n=-M}^{M}\alpha_n e_n(x)\big)\end{bmatrix} = \zeta\begin{bmatrix}\sum_{n=-M}^{M}\alpha_n e_n(x) \\ \sum_{n=-M}^{M}\beta_n e_n(x)\end{bmatrix}. \end{aligned} \tag{3.3}$$

From the equality of the first component of (3.3) we have

$$\begin{aligned} i\left(\frac{\partial}{\partial x}\left(\sum_{n=-M}^{M}\alpha_n e_n(x)\right) + \left(\sum_{n=-M}^{M}\gamma_n e_n(x)\right)\left(\sum_{n=-M}^{M}\beta_n e_n(x)\right)\right) &= \zeta\sum_{n=-M}^{M}\alpha_n e_n(x), \\ \Rightarrow\ i\left(\sum_{n=-M}^{M}\alpha_n\frac{2n\pi i}{\ell}e_n(x) + \sum_{n=-M}^{M}\sum_{j=-M}^{M}\gamma_n\beta_j e_n(x)e_j(x)\right) &= \zeta\sum_{n=-M}^{M}\alpha_n e_n(x), \\ \Rightarrow\ i\left(\sum_{n=-M}^{M}\alpha_n\frac{2n\pi i}{\ell}e_n(x) + \sum_{n=-M}^{M}\sum_{j=-M}^{M}\gamma_n\beta_j e_{n+j}(x)\right) &= \zeta\sum_{n=-M}^{M}\alpha_n e_n(x), \end{aligned}$$

and by comparing the coefficients of the basis $\{e_n\}_{n\in\mathbb{Z}}$ we have the equalities

$$-\frac{2n\pi}{\ell}\alpha_n + i\sum_{j=\max\{-M,-M+n\}}^{\min\{M,M+n\}}\gamma_{n-j}\beta_j = \zeta\alpha_n. \tag{3.4}$$

With a similar analysis for the second component of (3.3) we have

$$i\kappa\sum_{j=\max\{-M,-M+n\}}^{\min\{M,M+n\}}\overline{\gamma_{j-n}}\alpha_j + \frac{2n\pi}{\ell}\beta_n = \zeta\beta_n. \tag{3.5}$$

Hence the eigenproblem (2.8) reduces to the system [24]

$$\begin{cases} -\frac{2n\pi}{\ell}\alpha_n + i\sum_j\gamma_{n-j}\beta_j & = \zeta\alpha_n \\ i\kappa\sum_j\overline{\gamma_{n-j}}\alpha_j + \frac{2n\pi}{\ell}\beta_n & = \zeta\beta_n, \end{cases}$$

which is equivalent to the eigenproblem

$$\begin{bmatrix} C_1 & C_2 \\ -C_2^* & -C_1\end{bmatrix}\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = \zeta\begin{bmatrix}\alpha \\ \beta\end{bmatrix}, \tag{3.6}$$

where

$$\alpha = \begin{bmatrix}\alpha_{-M} & \alpha_{-M+1} & \dots & \alpha_0 & \dots & \alpha_M\end{bmatrix}^\intercal,$$
$$\beta = \begin{bmatrix}\beta_{-M} & \beta_{-M+1} & \dots & \beta_0 & \dots & \beta_M\end{bmatrix}^\intercal,$$
$$C_1 = \frac{-2\pi}{\ell}\operatorname{diag}\{-M,\ -M+1,\ ,\ \dots,\ M\} \ \in\ M_{2M+1}(\mathbb{C}), \tag{3.7}$$

$M_K(\mathbb{C})$ denotes the set of the $K \times K$ matrices with elements from $\mathbb{C}$,
$C_2$ is the Toeplitz matrix with first row

$$(i\begin{bmatrix}\gamma_0 & \gamma_{-1} & \gamma_{-2} & \dots & \gamma_{-M} & 0 & 0 & \dots & 0\end{bmatrix})$$

and first column

$$(i\begin{bmatrix}\gamma_0 & \gamma_1 & \gamma_2 & \dots & \gamma_M & 0 & 0 & \dots & 0\end{bmatrix}^\intercal). \tag{3.8}$$

Since the problem (3.6) approximates (3.1), we have that the eigenvalues of the matrix in (3.6) approximate the periodic eigenvalues of $L_{t_0}$. When the signal is smooth, it can be shown that the error of the discrete eigenvalues in the spectrum decays exponentially with the number of Fourier modes [23]. Now, replacing the period $\ell$ by $2\ell$ and applying the same analysis as before we can compute the eigenvalues of $L_{t_0}$ that are periodic over two periods, which are exactly the periodic and anti-periodic eigenvalues (i.e. those where $\Delta(\lambda) = \pm 1$) over the period $\ell$. Thus, using the Fourier Collocation method we can approximate the main spectrum. In general, applying the previous analysis with period $N_0\ell$ we compute the quasi-periodic eigenvalues of quasi-momentum $\frac{k\pi}{N_0}$ for every $k \in \{0, 1, 2, \dots, N_0\}$, which are points that belong in the spines. In order to compute multiple spine points we have to use a sufficient number of periods. However as $N_0$ increases, the complexity of the method increases rapidly.

### 3.1.3 Modified Fourier Collocation Method

Using the Fourier Collocation method for computing spine points becomes computationally expensive quickly, hence a modified version of the Fourier collocation method has been proposed in [25]. This modified version computes the quasi-periodic eigenvalues of the problem (2.7), which are spine points.

Suppose that $\tilde{v}$ is a periodic function of period $\ell$ such that $v = e^{ipx}\tilde{v}$ is a solution of the eigenproblem (2.7). It can be observed that $v$ is a quasi-periodic solution of quasi-momentum equal to $p\ell$ and noted that for $p = 0$ we have a periodic solution and for $p = \frac{\pi}{\ell}$ we have an anti-periodic solution. In the following we suppose that $p \ \in\ [\ 0,\ \frac{\pi}{\ell}\ ]$. Using the substitution $v = e^{ipx}\,\tilde{v}$ and the fact that $v$ satisfies equation (2.7), we have

$$
\begin{aligned}
L_{t_0} v = \zeta v \iff & \; L_{t_0}(e^{ipx}\,\tilde{v}) = \zeta(e^{ipx}\,\tilde{v}) \\
\iff & \; i\begin{bmatrix} \frac{d}{dx} & q \\ \kappa\overline{q} & -\frac{d}{dx} \end{bmatrix}(e^{ipx}\,\tilde{v}) = \zeta(e^{ipx}\,\tilde{v}) \\
\iff i\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}\frac{d}{dx}(e^{ipx}\tilde{v}) + i\begin{bmatrix} 0 & q \\ \kappa\overline{q} & 0 \end{bmatrix} & (e^{ipx}\,\tilde{v}) = \zeta(e^{ipx}\,\tilde{v}) \\
\iff (L_{t_0}\tilde{v})\,e^{ipx} - p\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} & (e^{ipx}\,\tilde{v}) = \zeta(e^{ipx}\,\tilde{v}) \\
\iff & \; L_{t_0}\,\tilde{v} - p\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}\tilde{v} = \zeta\,\tilde{v}. \qquad (3.9)
\end{aligned}
$$

Since $\tilde{v}$ is a periodic function, we can follow the same procedure as for the Fourier Collocation method in Section 3.1.2 in order to approximate the quasi-periodic eigenvalues of quasi-momentum $p\ell$ numerically. Details here are omitted (see the computations in the Fourier collocation method in Section 3.1.2). We arrive to the eigenproblem

$$
S(p)\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \zeta\begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \qquad S(p) := \begin{bmatrix} C_1' & C_2 \\ -C_2^* & -C_1' \end{bmatrix} \qquad (3.10)
$$

where

$$
\begin{aligned}
\alpha &= \begin{bmatrix} \alpha_{-M} & \alpha_{-M+1} & \dots & \alpha_0 & \dots & \alpha_M \end{bmatrix}^{\mathsf{T}}, \qquad (3.11) \\
\beta &= \begin{bmatrix} \beta_{-M} & \beta_{-M+1} & \dots & \beta_0 & \dots & \beta_M \end{bmatrix}^{\mathsf{T}}, \qquad (3.12) \\
C_1' &= C_1 - p\,\mathbb{I}_{2M+1} \in M_{2M+1}(\mathbb{C}), \qquad (3.13)
\end{aligned}
$$

where $\mathbb{I}_K$ is the $K \times K$ identity matrix and $C_1$ , $C_2$ are the matrices given by (3.7) and (3.8) respectively. It is clear now that setting $p = 0$ in the problem (3.10) we have the eigenproblem (3.6) and $S(0) = \begin{bmatrix} C_1 & C_2 \\ -C_2^* & -C_1 \end{bmatrix}$.

Standard algorithms from numerical linear algebra can be used to compute the eigenvalues of the matrix eigenproblem (3.10). In order to compute the spines let $N_0$ be a natural number, $N_0 \in \mathbb{N}$ , $N_0 > 0$, which corresponds to an upper bound to the number of points computed in each spine. The eigenvalues of the problem (3.10) for the cases $p = 0$ and $p = \pi/\ell$ approximate all the points in the main spectrum, while the rest of the spine points are approximated by the eigenvalues of $S(p)$ when $p = \frac{\pi j}{(N_0-1)\ell}$ , $j = 0, 1, 2, \dots, N_0 - 1$. Larger values of $N_0$ provide better resolution for the spines. The complexity of the modified method grows only linearly in $N_0$, while for the Fourier collocation method the complexity grows cubically with $N_0$ periods.

The method discussed in the previous paragraph is applied to a simple example that can be found in [17], where

$$
\begin{aligned}
q(x,t) &= 1 + 0.22e^{-0.822ix}, \qquad (3.14) \\
\ell &= \frac{2\pi}{0.822}, \qquad (3.15)
\end{aligned}
$$

for different choices for the number of points in the spines. The computed eigenvalues are plotted and are shown in Figure 3.1. From the plots in Figure 3.1 we can see the spines emerge as the number of points $N_0$ increases.

This algorithm for computing points in the spines has some drawbacks. The most important disadvantage is that it produces a list of points, without indicating which points belong in the same spine. After analysis of the points we can determine the spines visually, by plotting for example, but this will be time consuming and not always efficient. Another issue which makes this difficult is that the computed points are not equidistant, but instead are clustered around the periodic eigenvalues. The latter drawback can also be observed in Figure 3.1

## 3.2 New Algorithm: Spine tracking

During the project a new algorithm was developed that computes the spines on which a given main spectrum point lies, with accuracy $h$ chosen by the user in advance. The algorithm was inspired by the phase jump tracking algorithm proposed by Chekhovskoy et al. for the computation of the main spectrum [26]. Using Lemma 2.1.1 and taking $m = e^{ipx}$, for all $p \in [0,\, \pi/\ell]$, it can be shown that the spines are defined by the Equations (2.19), which are repeated here for convenience,

$$\Im[\Delta(\zeta)] = 0\ , -1 \leq \Re[\Delta(\zeta)] \leq 1, \tag{3.16}$$

where $\Delta(\zeta)$ is the Floquet Discriminant computed at the point $\zeta \in \mathbb{C}$.

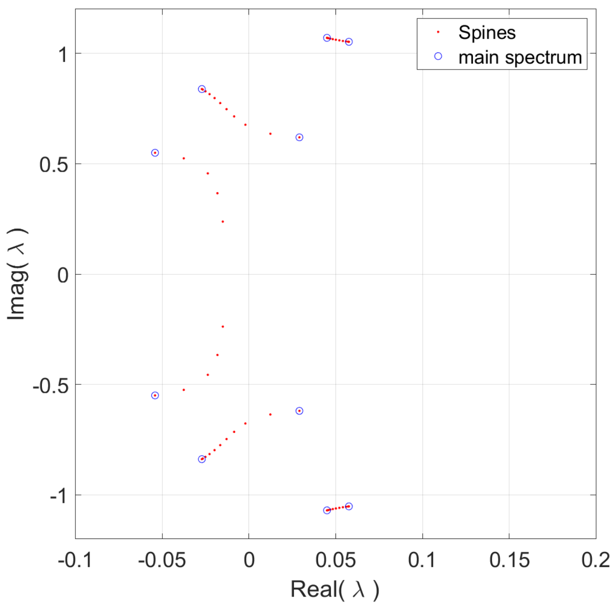
The following method is based on tracking points that lie on the curve $\Im[\Delta(\zeta)] = 0$ and do not exceed the boundaries in (3.16). Let $g$ be a main spectrum point and for simplicity suppose that $g$ is a periodic eigenvalue, i.e. $\Delta(g) = 1$ by Lemma 2.1.1. Note that the same algorithm can be used in the case that $g$ is an anti-periodic eigenvalue, i.e. $\Delta(g) = -1$. Then two points are located that belong in the two areas that are defined by the curve $\Im[\Delta(\zeta)] = 0$ in a small neighbourhood of $g$. Precisely, one point lies in $\Im[\Delta(\zeta)] > 0$ and one lies in $\Im[\Delta(\zeta)] < 0$. In addition, the midpoint of these two points should belong in the area $|\Re[\Delta(\zeta)]| < 1$, since the midpoint will approximate a new spine point, or equivalently a point that belongs in the curve defined by (3.16). In the plot in Figure 3.2 we can see an example of the different areas in a neighbourhood of $g$ and a possible choice of the two points $g_{left}$ and $g_{right}$. A part of the spine in the plot is given by the dashed curve that belongs in the area given by $\Re(\Delta(\zeta)) < 1$ in the neighbourhood of $g$. The points $g_{left}$ are located "left" and "right" of the spine respectively.

At each iteration the algorithm defines a new set of points $g_{left}$, $g_{right}$ for which the criterion

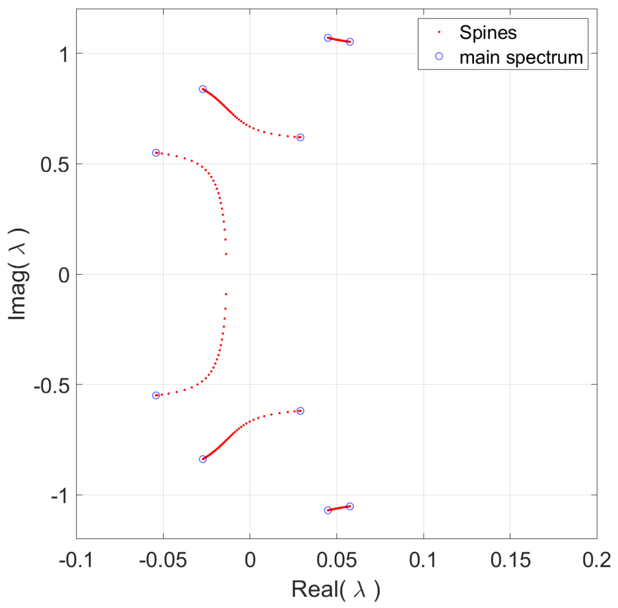
$$\Im(\Delta(g_{left}))\Im(\Delta(g_{right})) < 0, \quad \left|\frac{\Re(\Delta(g_{left}) + \Delta(g_{right}))}{2}\right| < 1 \tag{3.17}$$

is verified. This criterion checks whether the spine curve passes in between the points $g_{left}$ and $g_{right}$ In that case, $\frac{g_{left}+g_{right}}{2}$ is chosen as a point in the spine and it is used as the initial point $g$ in the next iteration. In order to avoid approximating a point that belongs in the part of the spine that was computed in previous iterations and to approximate a new point of the spine, we introduce an angle $\phi$ that is perpendicular to the direction in which the spine moves in the complex plane. The variable $\phi$ will be updated at each iteration. Methods to determine the initial value of the angle are examined later in Subsection 3.2.1.

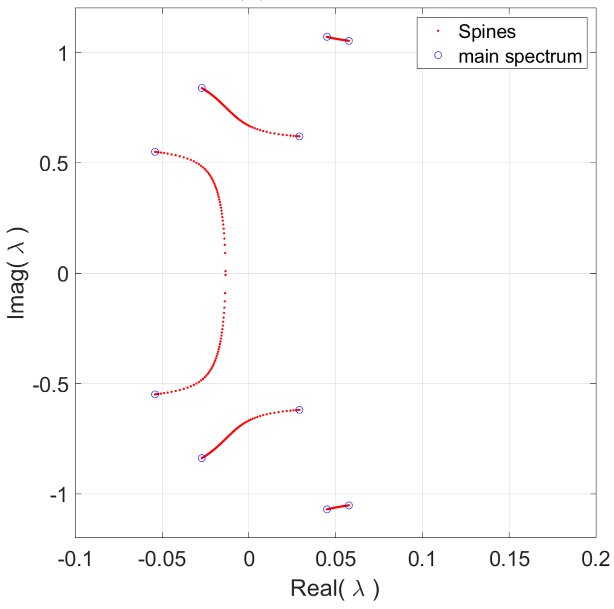
Two different variants for determining the pair $g_{left}$, $g_{right}$ will be presented. Let $h > 0$ small enough, which will be the bound of the step-size between the points in the curve



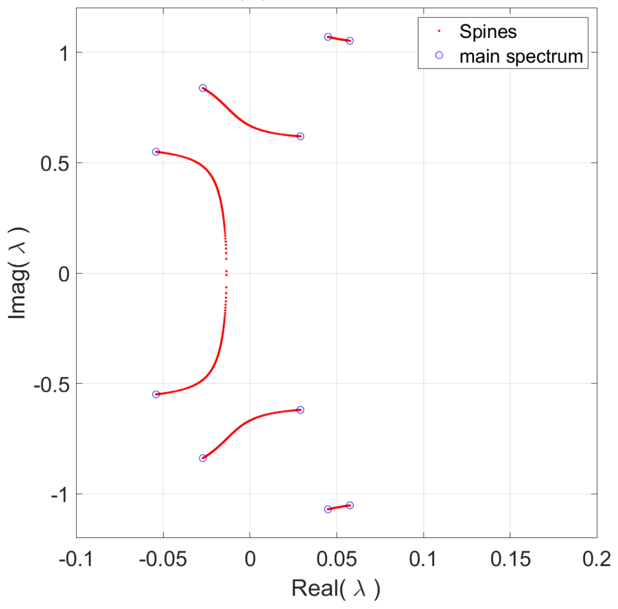
(a) $N_0 = 10$



(b) $N_0 = 50$



(c) $N_0 = 100$



(d) $N_0 = 200$

Figure 3.1: Plots of the non-linear spectrum for different choices of $N_0$

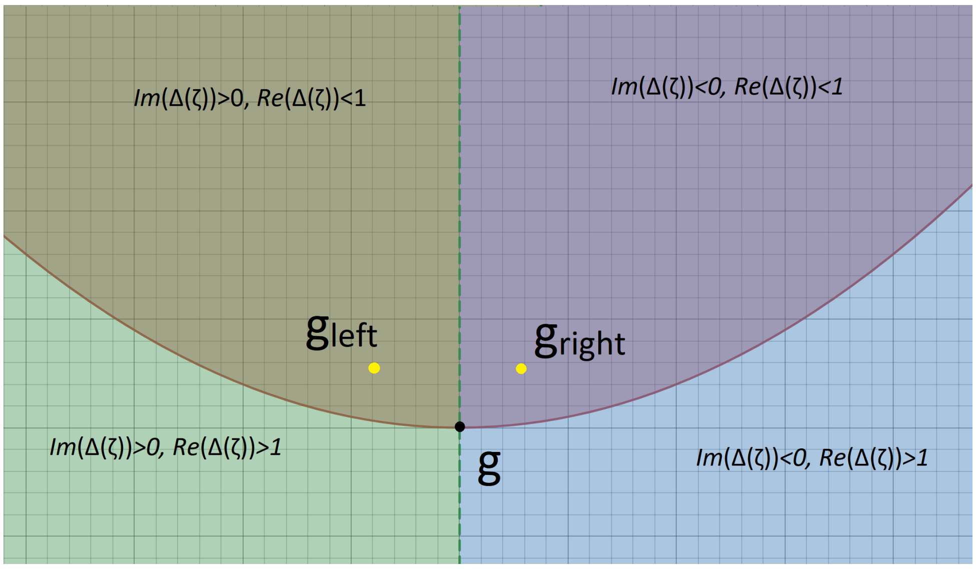


Figure 3.2: An example of the different areas defines by $\Delta$ and a possible choice of points $\{g_1, g_2\}$

and let $\phi$, be the angle that perpendicular to the direction of the spine. At each iteration of the first variation of the algorithm, we define the points

$$\begin{aligned} g_1 &= g + \frac{h}{2}e^{\phi i}, \\ g_2 &= g + \frac{h}{2}e^{\phi i} + shift, \\ g_3 &= g - \frac{h}{2}e^{\phi i} + shift, \\ g_4 &= g - \frac{h}{2}e^{\phi i}, \end{aligned} \tag{3.18}$$

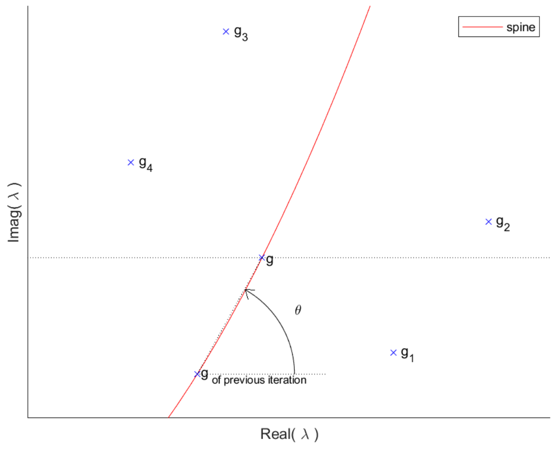


Figure 3.3: Plot of the points $\{g_i\}_{i=1}^{4}$ defined in (3.18). Note that the direction of the spine is given by $\theta$ and the angle used in (3.18) is computed by $\phi = \theta - \pi/2$.

where $shift = \frac{h}{2}e^{(\phi+\pi/2)i}$ and $g$ is the initial point. Afterwards, the criterion (3.17) is checked for the pairs $\{g_1, g_2\}$ , $\{g_2, g_3\}$ and $\{g_3, g_4\}$. When a pair $\{g_i, g_{i+1}\}$ verifies (3.17) then each point of the pair is stored in a pair of lists that correspond to curves which enclose the spine curve. An example of these curves can be seen in Figure 3.5. Also, in that case we set $g = \frac{g_i+g_j}{2}$, $\phi = Arg(g_i - g_{i+1})$ and move on to the next iteration. On the other hand, if the criterion is not met in either of the pairs checked, the algorithm stops and the final spine point is set as $\frac{1}{4}(g_1 + g_2 + g_3 + g_4)$.

Regarding the second variant of the algorithm, the pairs that potentially verify the

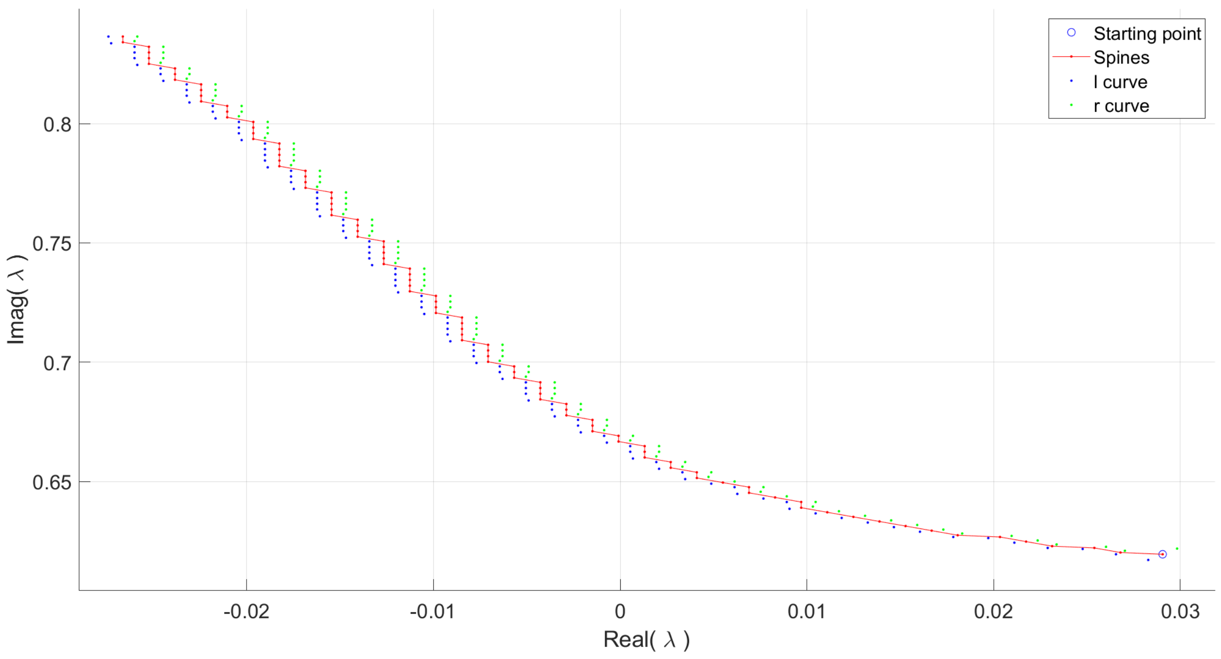


Figure 3.5: Curves and spine produce by the algorithm with initial eigenvalue g=0.02905785 + 0.61950993i , h=0.005 and signal q given by (3.14)

criterion (3.17) are formed by points that lie on the circle with center $g$ and radius $h$ are checked in each iteration. The potential pairs are the pairs $\{g_i, g_{i+1}\}$ of the set $\{g_i\}_{i=1}^{n+1}$ given by

$$
\begin{aligned}
g_1 &= g + he^{\phi i}, \\
g_2 &= g + he^{(\phi+\pi/n)i}, \\
&\dots, \\
g_n &= g + he^{(\phi+(n-1)\pi/n)i}, \\
g_{n+1} &= g + he^{(\phi+\pi)i}.
\end{aligned}
\tag{3.19}
$$

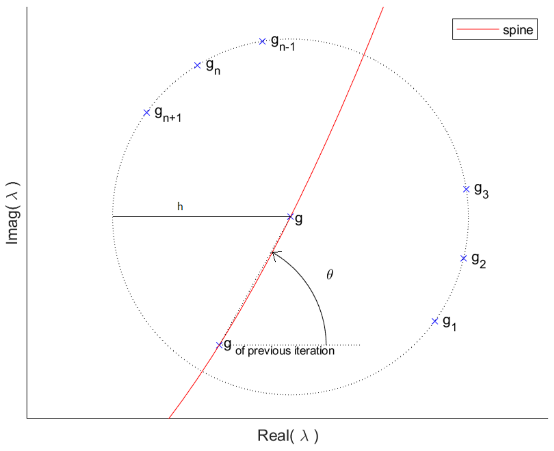


Figure 3.4: Plot of the points $\{g_i\}_{i=1}^{n+1}$ defined in (3.19). Note that the direction of the spine is given by $\theta$ and the angle used in (3.19) is computed by $\phi = \theta - \pi/2$.

The full form of the variations of the algorithm are presented in the appendix in Algorithm 2 and Algorithm 3. The second variation of the algorithm is presented for the case of four circle points (3.19), but can be modified for the case of $n$ points, $n \in \mathbb{N}$, by taking $shift = \pi/n$ and adding the appropriate subroutines in Algorithm 3. A more general form of the algorithm, which has as input the number of the points $\{g_i\}$ that are checked in each iteration can be found in Algorithm 4. Note that the spines can be extracted from the algorithms as the curves defined by the points

$$\left(spine(j) = \frac{l(j)+r(j)}{2}\right)_{j=1}^{s}, \text{where } s = length(r) = length(l)$$

In addition, observe that the Algorithms 2 and 3 may track curves that are not smooth. In order to overcome this problem, Algorithm 3 can be adjusted to only check if the condition (3.17) is met by pairs that lie on an arc of the circle $\gamma(z) := g + he^{zi}$. By choosing an appropriate arc, the pairs that provide not sufficiently smooth curves can be filtered out. For example, in the version of Algorithm 3 taking $shift = \frac{\pi}{7}$, we have subroutine checks for the pairs $\{g_i, g_{i+1}\}$, $\forall\, i = 1, 2, 3, 4, 5, 6, 7$, where $\{g_j\}_{j=1}^{8}$ are given by (3.19). The algorithm can be modified to only check the pairs $\{g_i, g_{i+1}\}$, $\forall\, i = 2, 3, 4, 5, 6$, or can be further modified to check the pairs $\{g_i, g_{i+1}\}$, $\forall\, i = 3, 4, 5$.

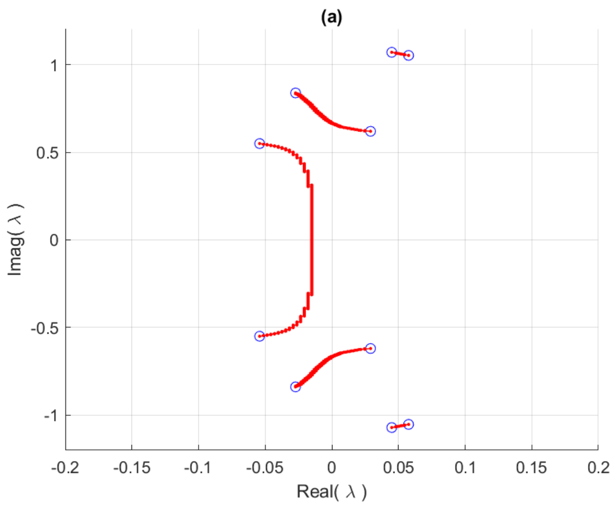
In order to overcome the problem of two or more spines intersecting the algorithm can be modified into a recursive algorithm that runs separately for every pair that verifies the criterion. A general form of the algorithm with filtering is presented in the Appendix in Algorithm 5. A MATLAB function implemented during the project for the new algorithm is presented in the Appendix in Listing 5. We demonstrate how to numerically compute the spines of the signal given in (3.14), using the script in Listing 5.

### 3.2.1 Initial angle

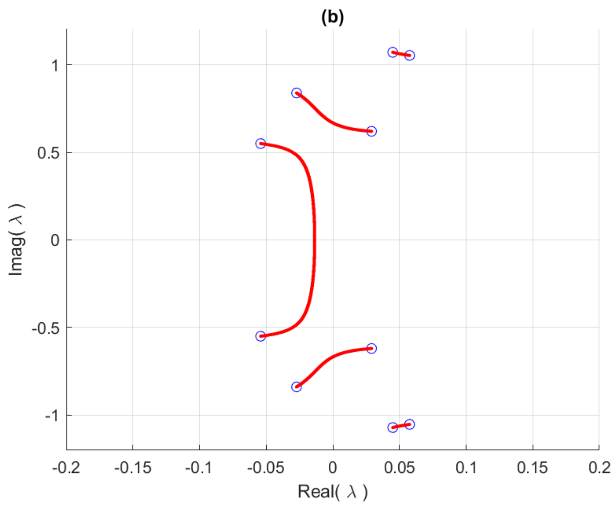
The algorithms presented in this chapter require an angle $\phi$ as an input, which depends on the direction of the spine in the complex plane. The modified Fourier collocation method (3.1.3) and Lemma 2.1.1 provide that the spines can be constructed by quasi-periodic eigenvalues of quasi-momentum $p \in [0, \pi]$, i.e. eigenvalues of the matrix $S(p)$ in (3.10). A method to identify the angle/direction of the spine is to find a point in the spine of quasi-momentum close to the one of the initial main spectrum point $g$. For example, if the point $g$ is a periodic (anti-periodic) eigenvalue, then we a point of quasi-momentum equal to $\epsilon$ $(\pi - \epsilon)$, for $\epsilon > 0$ small, should be searched. In order to determine the spectrum point on the spine, the initial main spectrum point can be refined, using e.g. Newton-Raphson method, to solve the equation $m^2 - 2m\Delta(\zeta) + 1 = 0$ (Lemma 2.1.1), for $m = e^{ip}$ and $p = \epsilon$ or $p = \pi - \epsilon$ depending on $g$. Let $g' \in \mathbb{C}$ be the new point in the spine, after the refinement of $g$. Then $\psi = Arg(g - g')$ is the angle/direction of the spine and $\phi = \psi - \frac{\pi}{2}$ is used as input for the Algorithms 2 and 3.

A drawback to this method is that if the quasi-periodic spectrum point computed is close to the initial main spectrum point, then it may result to an incorrect initial angle. Precisely, if the distance between the computed point and the initial main spectrum point $g$ is considerably less than the step-size $h$ then the possible pairs, that are placed in a distance of $h$ from $g$, may not verify the criterion and hence the tracking algorithm may break in the first iteration. A similar phenomenon might also occur when the distance between the computed point and the initial main spectrum point is significantly greater than the step-size.

Solving the mentioned problem leads us to another algorithm for computing the initial direction of the spine that uses the step-size $h$ and is similar to an iteration of Algorithm



(a) $h = 0.01$



(b) $h = 0.001$

Figure 3.6: Plots of the non-linear spectrum for different choices of $h$ using the tracking algorithm

3. Equidistant points on the circumference of the circle with center $g$ and radius $h$ are defined and the criterion (3.17) is checked for each pair of consecutive points. If the criterion is met for a pair $\{g_i, g_{i+1}\}$, then the angle $\phi = Arg(g_i - g_{i+1})$ is used in the Algorithms 2 and 3. The full form of this method is presented below in Algorithm 1. For the plots that appear in Section 3.3 Algorithm 1 was used.

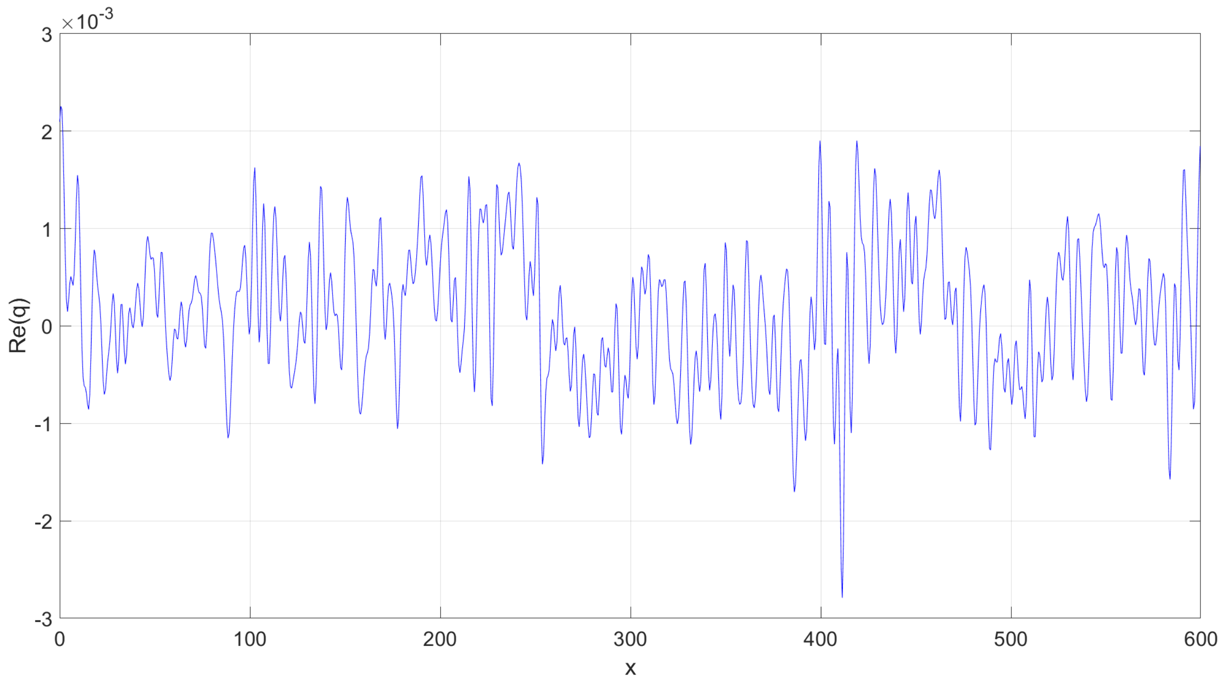
## 3.3 Results

The algorithm presented in Section 3.2 computes the spine curves with accuracy equal to the given step size $h$, and for each main spectrum point, a list can be extracted composed of points belonging in the spine passing through the point. Hence, the algorithm overcomes the problems that arise using Fourier Collocation method to compute the spines (Section 3.1.3). In detail, it can be observed that the step size $h$ in the tracking algorithm guarantees that the points computed are distributed equally along the curve. This phenomenon can also be observed by comparing the plots in Figures 3.1 and 3.6 in an example setting. Furthermore, it is obvious that by construction the algorithm in Section 3.2 provides a list of points for each spine and hence overcomes the other drawback presented in Section 3.1.3.

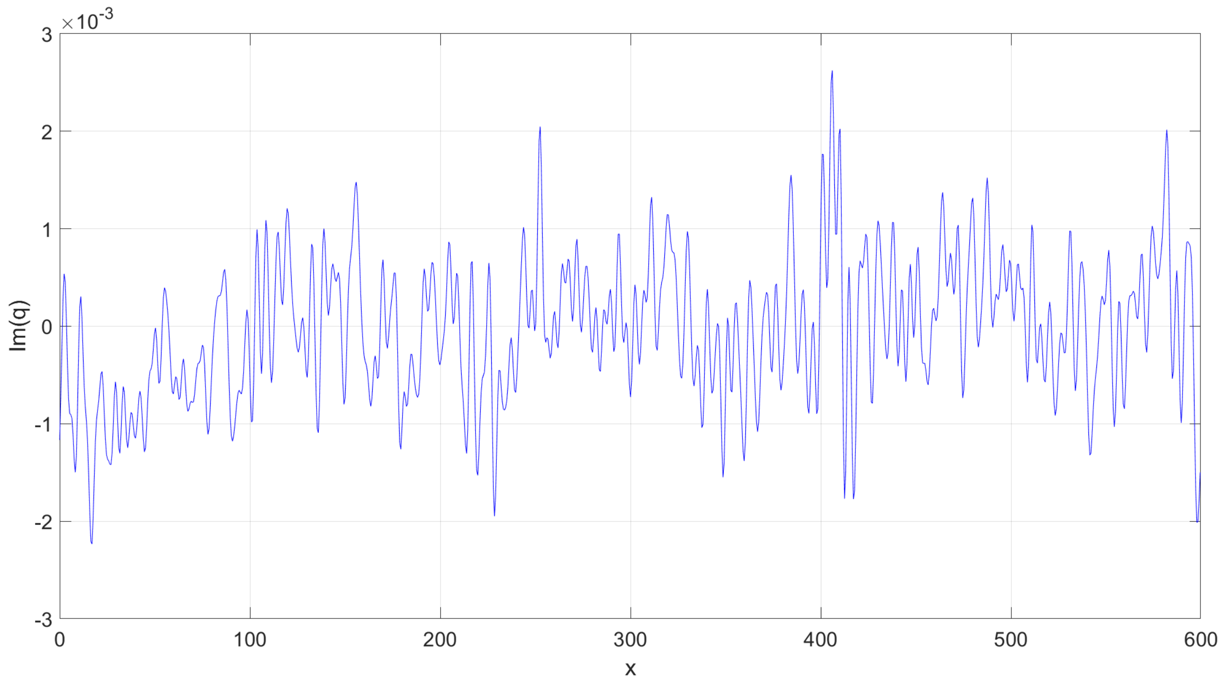
Additionally, the algorithm developed during the project has been tested for several signals for which the method based on the modified Fourier Collocation and the method implemented in FNFT fail. The structure of the algorithm can provide results even for complicated input data. For the plots in Figure 3.8 the modified Fourier collocation method was applied, for the plots in Figure 3.9 the method implemented in the FNFT library was applied, while for the plots in Figure 3.10 the MATLAB code in Listing 5 was used for the computation of the spines. The plots (a) in Figures 3.8, 3.9 and 3.10 show the computed main spectrum and spines of the signal $q$ given in Figure 3.7, after the application of the different methods. The signal used in the plots (b) in Figures 3.8, 3.9 and 3.10 is given by

$$q(x,t) = 1, \quad \text{for } x \in [0, 0.1], \tag{3.20}$$

with period $\ell = 0.1$. We can observe that in these examples the modified Fourier Collocation method and the method in the FNFT library fail, while the new algorithm computes the spines successfully.

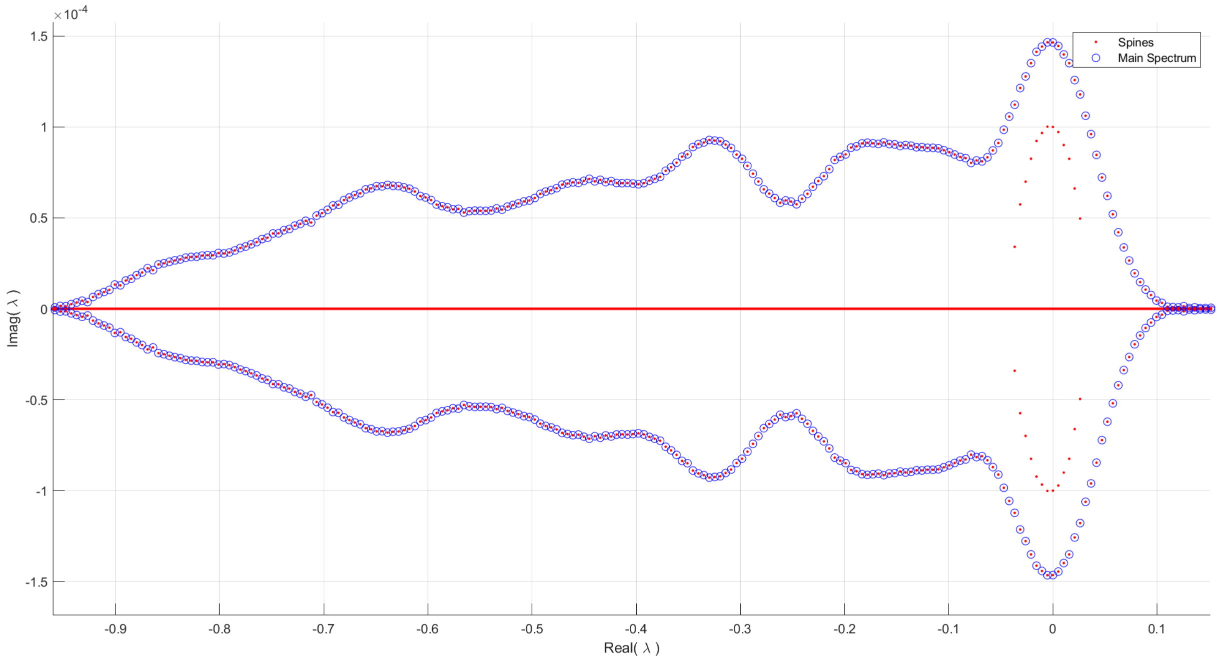


(a) Real part of the signal

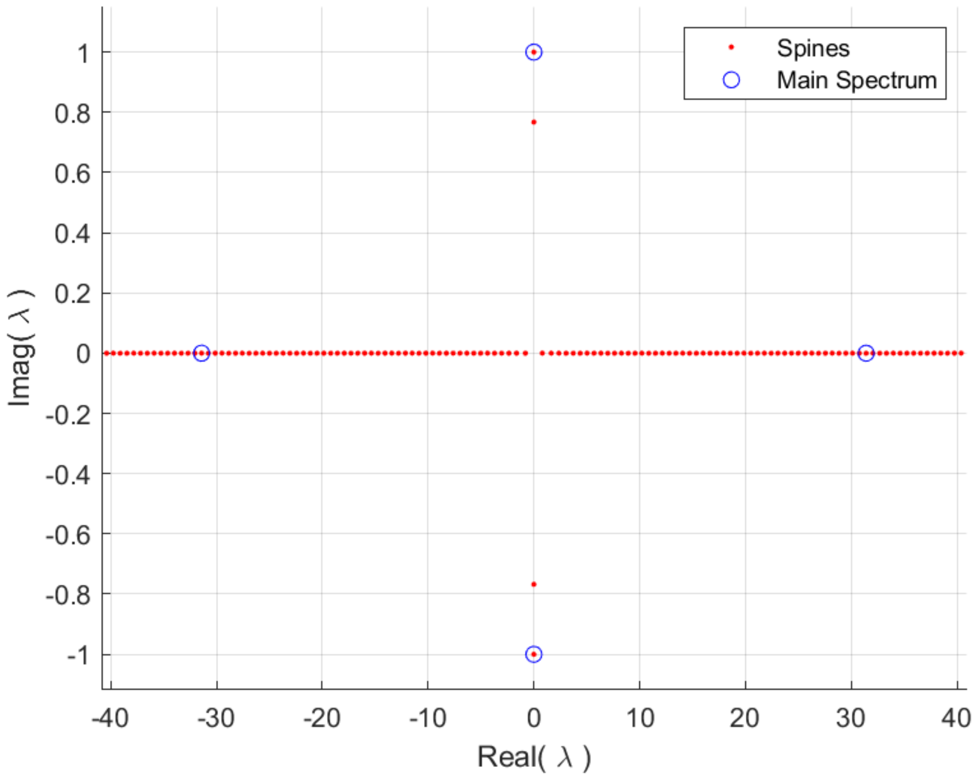


(b) Imaginary part of the signal

Figure 3.7: Sample collected from ocean waves

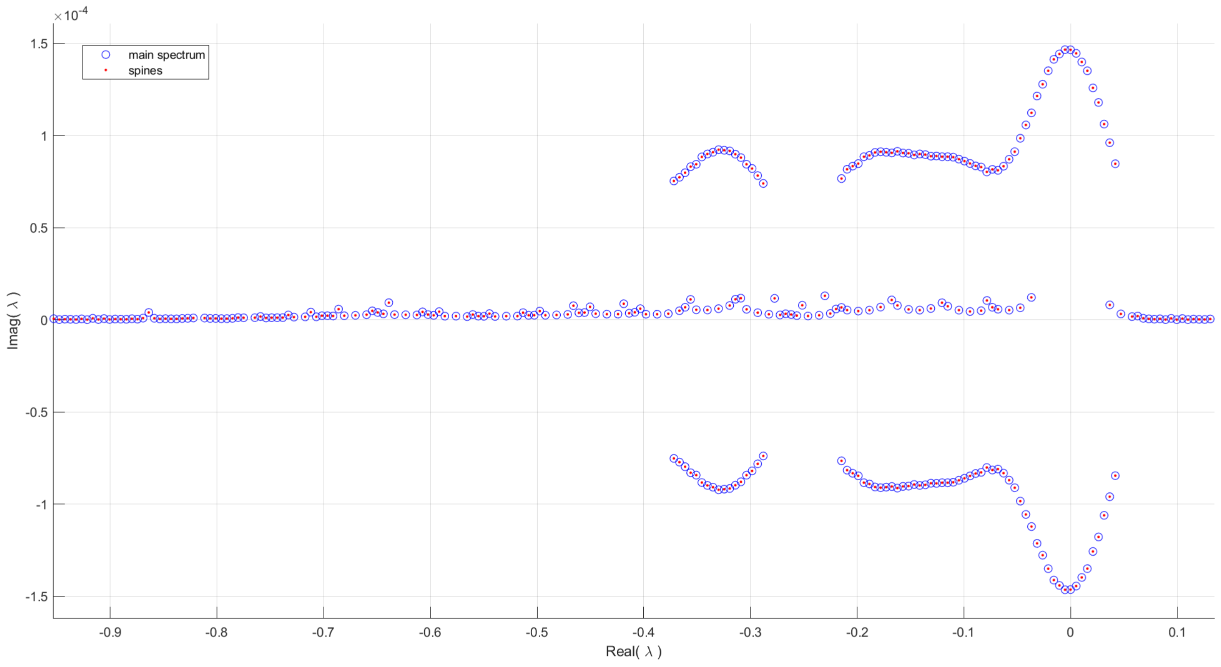


(a)

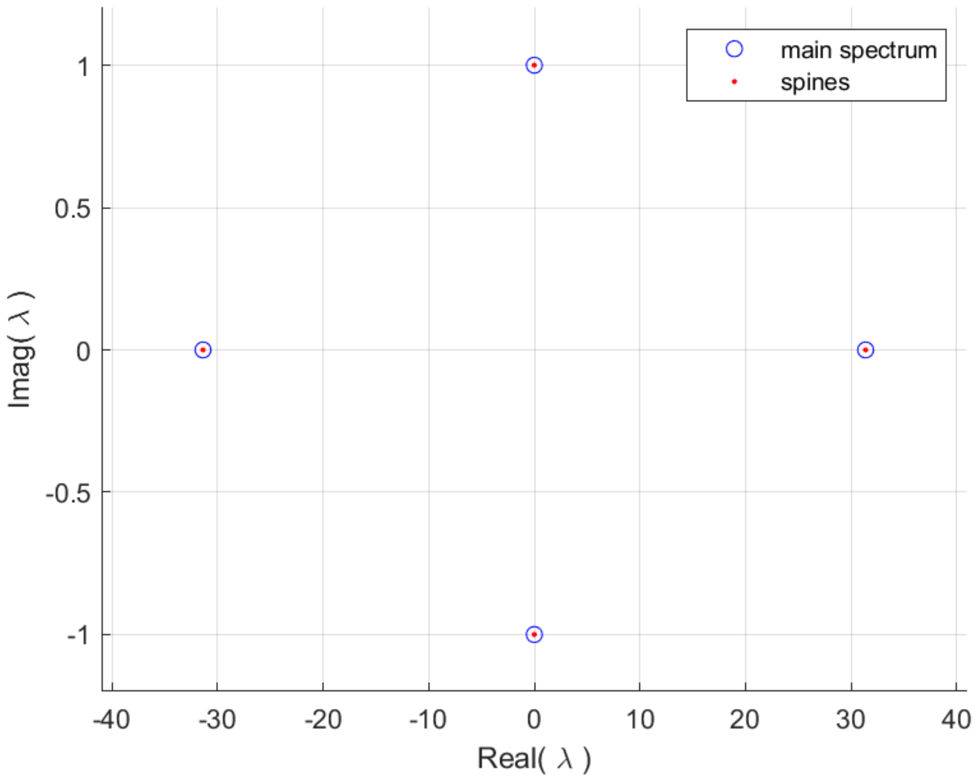


(b)

Figure 3.8: Plots of the non-linear spectrum for examples after the application of the modified Fourier Collocation method

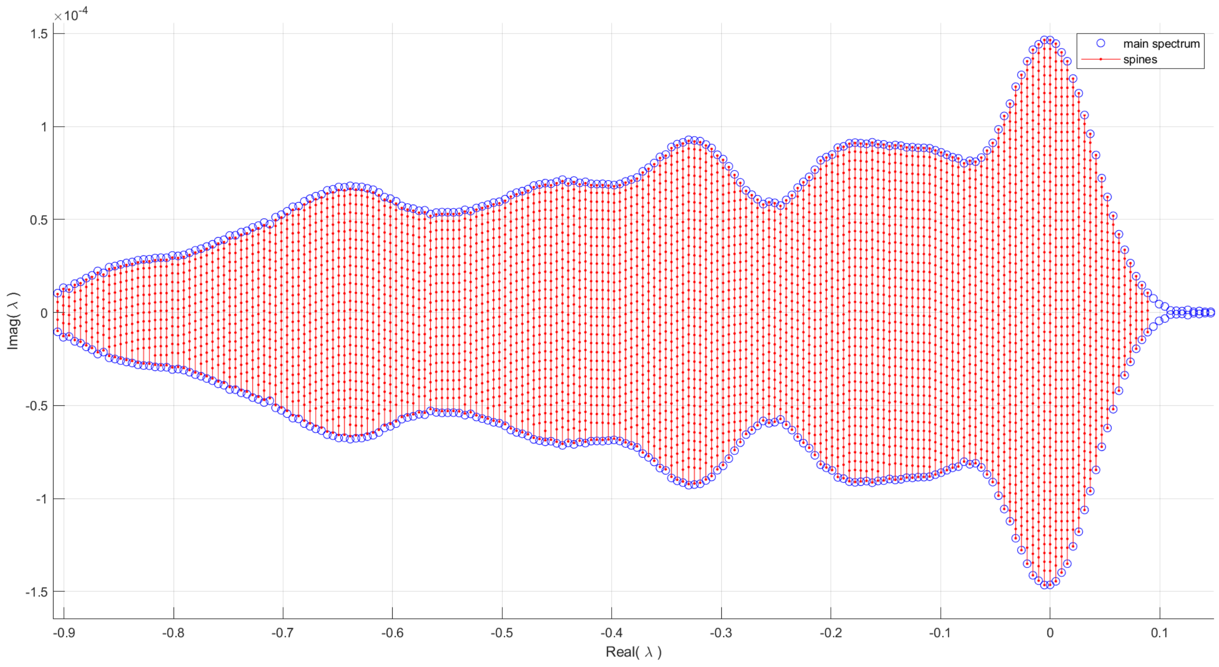


(a)

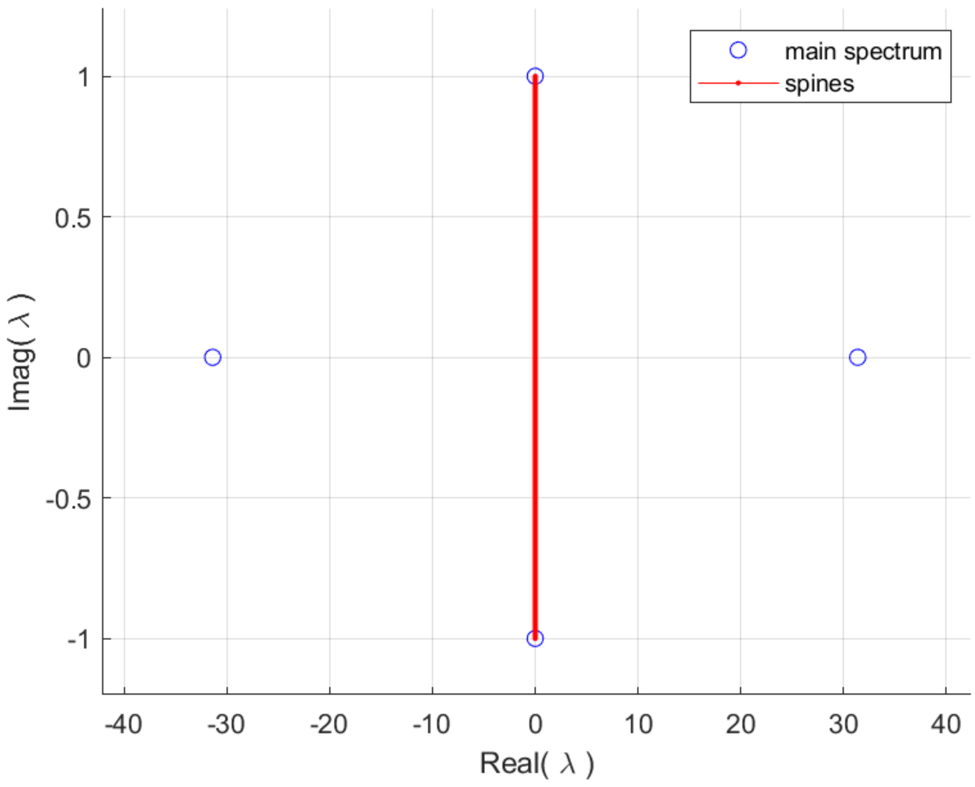


(b)

Figure 3.9: Plots of the non-linear spectrum for examples after the application of the method implemented in the FNFT library



(a)



(b)

Figure 3.10: Plots of the non-linear spectrum for examples after the application of the new algorithm

# Chapter 4

# Conclusion

The first part of the internship project was devoted on the study of the mathematical theory of nonlinear Fourier analysis. Emphasis was given to derivation of scattering data from the Monodromy matrix. In addition, several algorithms from the literature for the computation of the non-linear spectrum were examined. During the final part of the project, an algorithm for the solution of the Problem (1.1) was designed.

The algorithm in Section 3.2 determines the spines by tracking the sign changes of the imaginary part of the Floquet Discriminant, $\Im(\Delta)$, in the area $-1 \leq \Re(\Delta) \leq 1$. The algorithm tracks the spine curves that pass through a given main spectrum point and thus we can determine which other main spectrum points belong in the spines computed. In addition, the mentioned algorithm can overcome some of the problems observed when computing the spines with the use of the Fourier collocation method or the FNFT library.

The algorithm in Section 3.2 determines the spines by tracking the sign changes of the imaginary part of the Floquet Discriminant, $\Im(\Delta)$, in the area $-1 \leq \Re(\Delta) \leq 1$. The algorithm tracks the spine curves that pass through a given main spectrum point and thus we can determine which other main spectrum points belong in the spines computed. On the other hand the rest of the algorithms examined during the project did not provide any information on which points belong to the same spine. In addition, the points approximated by the new algorithm are equally distributed along the spines, while for the modified Fourier Collocation method and the FNFT this was not the case. The new algorithm approximated the spines in examples where the other methods fail. Finally, it was observed that the MATLAB function 5 computes numerically the spines faster than the modified Fourier Collocation method in experiments, however a formal complexity analysis was not carried out.

# Chapter 5

# Appendix

**Algorithm 1** Initial angle algorithm

**Input:** $q$ - signal, given by the signal samples $\{g_i\}_i$
$P$ - vector of period, i.e. $P(1)$ is the initial position , $P(2)$ is the final position,
$g$ - main spectrum point , $k$ - $2k$=number of points on the circle , $h$ - step-size
**Output:** $\phi$ - initial angle.
**Notes:** mod is the modulo operation. The operation mod $(\alpha, \beta)$ returns the remainder $r$ of the division $\alpha/\beta$, where $\alpha$, $\beta$ are integers.

```
for $j \leftarrow 1$ to $2k+1$ do
    $y(j) \leftarrow g + he^{(j-1)\frac{\pi}{k}i}$
    $a(j) \leftarrow \Delta(y(j)\,;\, q, P)$
end for
for $j \leftarrow 1$ to $2k$ do
    if $\Im((a(j))) \cdot \Im(a(\mod(j, 2k)+1)) < 0$
    and $|(\Re(a(j)) + \Re(a(\mod(j, 2k)+1))/2| < 1$ then
        $\phi \leftarrow (j - \frac{1}{2})\pi/k - \pi/2$
    end if
end for
```

**Algorithm 2** Tracking algorithm version 1

**Input:** $q$ - signal, given by the signal samples $\{g_i\}_i$
$P$ - vector of period, i.e. $P(1)$ is initial position , $P(2)$ is the final position,
$g$ - main spectrum point , $\phi$ - initial angle , $h$ - step-size
**Output:** $l, r$ - lists that enclose the spine , $eigvalue$ - final spine point.

```
$m \leftarrow 1$
$l(m) \leftarrow g - \frac{h}{2}e^{\phi i}$
$r(m) \leftarrow g + \frac{h}{2}e^{\phi i}$
while $true$ do
    $shift \leftarrow \frac{h}{2}e^{(\phi+\pi/2)i}$
    $r_{temp} \leftarrow r(m) + shift$
    $l_{temp} \leftarrow l(m) + shift$
    $a_1 \leftarrow \Delta(r_{temp}\,;\, q, P)$
    $a_2 \leftarrow \Delta(l_{temp}\,;\, q, P)$
    if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
        $l(m+1) \leftarrow l_{temp}$
        $r(m+1) \leftarrow r_{temp}$
    else
        $a_1 \leftarrow \Delta(r(m)\,;\, q, P)$
        $a_2 \leftarrow \Delta(r_{temp}\,;\, q, P)$
        if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
            $l(m+1) \leftarrow r_{temp}$
            $r(m+1) \leftarrow r(m)$
            $\phi \leftarrow \phi - \pi/2$
        else
            $a_1 \leftarrow \Delta(l(m)\,;\, q, P)$
            $a_2 \leftarrow \Delta(l_{temp}\,;\, q, P)$
            if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
                $l(m+1) \leftarrow l(m)$
                $r(m+1) \leftarrow l_{temp}$
                $\phi \leftarrow \phi + \pi/2$
            else
                $eigvalue \leftarrow (l(m) + r(m) + r_{temp} + l_{temp})/4$
                break
            end if
        end if
    end if
    $m \leftarrow m + 1$
end while
```

**Algorithm 3** Tracking algorithm version 2

**Input:**$q$ - signal, given by the signal samples $\{g_i\}_i$
$P$ - vector of period, i.e. $P(1)$ is initial position , $P(2)$ is the final position,
$g$ - main spectrum point , $\phi$ - initial angle , $h$ - step-size
**Output:** $l, r$ - lists that enclose the spine , $eigvalue$ - final spine point.

```
$m \leftarrow 1$
$l(m) \leftarrow g - he^{\phi i}$
$r(m) \leftarrow g + he^{\phi i}$
$shift \leftarrow pi/3$
while $true$ do
    $g \leftarrow (l(m) + r(m))/2$
    $r_{temp} \leftarrow g + he^{(\phi+shift)i}$
    $l_{temp} \leftarrow g + he^{(\phi+2shift)i}$
    $a_1 \leftarrow \Delta(r_{temp}\,;\, q, P)$
    $a_2 \leftarrow \Delta(l_{temp}\,;\, q, P)$
    if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
        $l(m+1) \leftarrow l_{temp}$
        $r(m+1) \leftarrow r_{temp}$
        $\phi \leftarrow \phi$
    else
        $r_{temp} \leftarrow g + he^{(\phi)i}$
        $l_{temp} \leftarrow g + he^{(\phi+shift)i}$
        $a_1 \leftarrow \Delta(r_{temp}\,;\, q, P)$
        $a_2 \leftarrow \Delta(l_{temp}\,;\, q, P)$
        if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
            $l(m+1) \leftarrow l_{temp}$
            $r(m+1) \leftarrow r_{temp}$
            $\phi \leftarrow \phi - shift$
        else
            $r_{temp} \leftarrow g + he^{(\phi+2shift)i}$
            $l_{temp} \leftarrow g + he^{(\phi+3shift)i}$
            $a_1 \leftarrow \Delta(r_{temp}\,;\, q, P)$
            $a_2 \leftarrow \Delta(l_{temp}\,;\, q, P)$
            if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
                $l(m+1) \leftarrow l_{temp}$
                $r(m+1) \leftarrow r_{temp}$
                $\phi \leftarrow \phi + shift$
            else
                $eigvalue \leftarrow g + (he^{(\phi)i} + he^{(\phi+shift)i} + he^{(\phi+2shift)i} + he^{(\phi+3shift)i})/4$
                break
            end if
        end if
    end if
    $m \leftarrow m + 1$
end while
```

**Algorithm 4** Tracking algorithm version 3

**Input:** $q$ - signal, given by the signal samples $\{g_i\}_i$
$P$ - vector of period, i.e. $P(1)$ is initial position , $P(2)$ is the final position,
$g$ - main spectrum point , $\phi$ - initial angle , $h$ - step-size, $k$ - k+1=points on the semicircle
**Output:** $l, r$ - lists that enclose the spine , $eigvalue$ - final spine point.

```
$m \leftarrow 1$
$l(m) \leftarrow g - he^{\phi i}$
$r(m) \leftarrow g + he^{\phi i}$
$shift \leftarrow pi/k$
while $true$ do
    $g \leftarrow (l(m) + r(m))/2$
    $new\_point = false$
    for $j \leftarrow 0$ to $k - 1$ do
        $r_{temp} \leftarrow g + he^{(\phi + j * shift)i}$
        $l_{temp} \leftarrow g + he^{(\phi + (j+1) * shift)i}$
        $a_1 \leftarrow \Delta(r_{temp}\,; q, P)$
        $a_2 \leftarrow \Delta(l_{temp}\,; q, P)$
        if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
            $l(m + 1) \leftarrow l_{temp}$
            $r(m + 1) \leftarrow r_{temp}$
            $\phi \leftarrow \phi + (j + \frac{1}{2}) * shift - \pi/2$
            $new\_point = true$
        end if
    end for
    if NOT($new\_point$) then
        $eigvalue \leftarrow g + \frac{1}{k+1} \sum_{j=0}^{k} he^{(\phi + j * shift)i}$
    end if
    $m \leftarrow m + 1$
end while
```

**Algorithm 5** General tracking algorithm for computing smooth spines

**Input:** $q$ - signal, given by the signal samples $\{g_i\}_i$
$P$ - vector of period, i.e. $P(1)$ is initial position , $P(2)$ is the final position,
$g$ - main spectrum point ,
$\phi$ - initial angle ,
$h$ - step-size,
$k$ - k+1=points on the semicircle
**Output:** $l, r$ - lists that enclose the spine , *eigvalue* - final spine point.
**Notes:** Regarding the filtering, let $y \in \mathbb{N},\ y > 2$, the algorithm checks only the pairs that belong in the arc $\gamma(s) = g + he^{(\phi+s)i}, s \in [filter * shift, \pi - filter * shift] \subseteq [\pi/(2y-1), \pi - \pi/(2y-1)]$, where $filter = \lfloor \frac{k}{y} \rfloor$.
In the algorithm the filtering variable $y$ is set $y = 3$

```
$m \leftarrow 1$
$l(m) \leftarrow g - he^{\phi i}$
$r(m) \leftarrow g + he^{\phi i}$
$shift \leftarrow pi/k$
$filter \leftarrow \lfloor \frac{k}{3} \rfloor$
while $true$ do
    $g \leftarrow (l(m) + r(m))/2$
    $new\_point = false$
    for $j \leftarrow filter$ to $k - 1 - filter$ do
        $r_{temp} \leftarrow g + he^{(\phi + j * shift)i}$
        $l_{temp} \leftarrow g + he^{(\phi + (j+1) * shift)i}$
        $a_1 \leftarrow \Delta(r_{temp}\,;\, q, P)$
        $a_2 \leftarrow \Delta(l_{temp}\,;\, q, P)$
        if $\Im(a_1) \cdot \Im(a_2) < 0$ and $|\frac{1}{2}(\Re(a_1) + \Re(a_2))| < 1$ then
            $l(m+1) \leftarrow l_{temp}$
            $r(m+1) \leftarrow r_{temp}$
            $\phi \leftarrow \phi + (j + \frac{1}{2}) * shift - \pi/2$
            $new\_point = true$
        end if
    end for
    if NOT($new\_point$) then
        $eigvalue \leftarrow g + \frac{1}{k+1} \sum_{j=0}^{k} he^{(\phi + j * shift)i}$
    end if
    $m \leftarrow m + 1$
end while
```

Listing 5.1: MATLAB source code for the new algorithm Spine tracking

```matlab
function [spine] = new_algorithm_spine_tracking(q,P,g,h)

%   Copyright <2021> <Christos Kitsios>
%
%   Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
%
%   The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
%
%   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
%
%   Author: Christos Kitsios
%
%
%Input:  q - signal,
%        P - vector of period, i.e. P(1) is initial position , P(2) is the final position,
%        g - main spectrum point ,
%        phi - initial angle ,
%        h - step-size,
%
%Output: spine - spine that passes through the spectrum point g



[phi] = initial_angle(q,P,g,h);
[eigvalue,l,r]=spine_track(q,P,g,phi,h);

spine=(l+r)/2;

```

```

function [phi] = initial_angle(q,P,g,h)
          k=5;
          y=g+h/2*exp(1i*pi/k*(0:1:2*k));
          [a]=floquet_points(q,P,y);

          phi=pi;
          for j=1:2*k
               if (imag(a(j))*imag(a(mod(j,2*k)+1))<0) && abs((real
                   (a(j))+real(a(mod(j,2*k)+1)))/2)<1
                    phi=j*pi/k-pi/(2*k)-pi/2;
               end
          end
end

function [eigvalues,l,r]=spine_track(q,P,g,phi,h)
%
%   Output: l, r - lists that enclose the spine ,
%             eigvalue - final spine point
%   NOTE: floquet_points is a function with input q, P and a
    point x and
%        output the floquet discriminant of x, \Delta(x), which
    is
%        computed numerically.


              m=1;

              l(m)=g-0.5*h*exp(1i*phi);
              r(m)=g+0.5*h*exp(1i*phi);
              shift=pi/5;

              while true

                   g=(l(m)+r(m))/2;
                   r_temp=g+h/2*exp(i*(phi+2*shift));
                   l_temp=g+h/2*exp(i*(phi+3*shift));
                   a1=floquet_points(q,P,l_temp);
                   a2=floquet_points(q,P,r_temp);
                   midp=(a1+a2)/2

                   cond1=(imag(a1)*imag(a2)<0) && (abs(real(midp))
                       <1);

                   if cond1
                        l(m+1)=l_temp;
```

```
                r(m+1)=r_temp;
                phi=phi;
            else
                r_temp=g+h/2*exp(i*(phi+shift));
                l_temp=g+h/2*exp(i*(phi+2*shift));
                a1=floquet_points(q,P,l_temp);
                a2=floquet_points(q,P,r_temp);
                midp=(a1+a2)/2

                cond1=(imag(a1)*imag(a2)<0) && (abs(real(midp))<1);
                if cond1
                    l(m+1)=l_temp;
                    r(m+1)=r_temp;
                    phi=phi-shift;
                else
                    r_temp=g+h/2*exp(i*(phi+3*shift));
                    l_temp=g+h/2*exp(i*(phi+4*shift));
                    a1=floquet_points(q,P,l_temp);
                    a2=floquet_points(q,P,r_temp);
                    midp=(a1+a2)/2

                    cond1=(imag(a1)*imag(a2)<0) && (abs(real(midp))<1);
                    if cond1
                            l(m+1)=l_temp;
                            r(m+1)=r_temp;
                            phi=phi+shift;
                    else
                            eigvalues=0.25*(g+h/2*exp(i*(phi))+...
                                          g+h/2*exp(i*(phi+shift))+...
                                          g+h/2*exp(i*(phi+2*shift))+...
                                          g+h/2*exp(i*(phi+3*shift)));
                            break
                    end
                end
            end

          m=m+1;
        end
end

```

```
function [FD]= floquet_points(q,P,z)

    N=length(q);
    l=P(2)-P(1);
    dT=(P(2)-P(1))/N;
    T=P(1)+dT/2:dT:P(2)-dT/2;
    kappa=1 ;

    %Computations for Monodromy matrix and Floquet discriminant
    for j=1:length(z)
        S=eye(2);
        clear U
        for n=1:N
            U(:,:,n)=(expm(dT*[-1i*z(j) -q(n) ; kappa*conj(q(n))
                1i*z(j)]));
            S=U(:,:,n)*S;
        end

    FD(j)=(1/2)*(S(1,1)+S(2,2));
    end
end

end
```

Listing 5.2: Demo script for the numerical computation of the spines of the signal given by (3.14)

```
clear all;
kappa=1;

%Define the signal
[q,~,P]=buildsignal;

%Accurancy / Step-size
h=1e-2;

%Computation of main spectrum, here it is computed using FNFT
[main_spectrum, ~] = mex_fnft_nsep(q, P, kappa);

%Computation of spines for the non-real main spectrum points
k=find(abs(imag(main_spectrum))>1e-2);
eigvalue=main_spectrum(k);
spine=[];
for j=1:length(eigvalue)
    clear temp ;
    x=eigvalue(j);
    [temp] = new_algorithm_spine_tracking(q,P,x,h);
    spine{j}=[temp];
end

%Plot
figure; hold on;
plot(main_spectrum,'bo')
for j=1:length(spine)
    plot(real(spine{j}),imag(spine{j}),'.-')
    axis([-0.6 0.6 -1.3 1.3]);  grid on;
end
xlabel('Real( \lambda )'); ylabel('Imag( \lambda )');
hold off;


function [q,T,P]=buildsignal
    l=2*pi/0.822;         %period
    P=[0, l];          %periodic vector
    N=2^9;             %number of samples
    dT=l/N        ;    %step size
    T=P(1)+dT/2:dT:P(2)-dT/2   ;   %grid

    q=1+0.22*exp(-1i*0.822*T);
end
```

# References

[1] V. E. Zakharov, "Stability of periodic waves of finite amplitude on the surface of a deep fluid," *Journal of Applied Mechanics and Technical Physics*, vol. 9, no. 2, pp. 190–194, 1968.

[2] S. Wahls and H. V. Poor, "Introducing the fast nonlinear Fourier transform," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing.* IEEE, 2013, pp. 5780–5784.

[3] A. Hasegawa and M. Matsumoto, "Optical solitons in fibers," in *Optical Solitons in Fibers.* Springer, 2003, pp. 41–59.

[4] C. S. Gardner, J. M. Greene, M. D. Kruskal, and R. M. Miura, "Method for solving the Korteweg-de Vries equation," *Physical review letters*, vol. 19, no. 19, p. 1095, 1967.

[5] A. Shabat and V. Zakharov, "Exact theory of two-dimensional self-focusing and one-dimensional self-modulation of waves in nonlinear media," *Soviet physics JETP*, vol. 34, no. 1, p. 62, 1972.

[6] M. J. Ablowitz, D. J. Kaup, A. C. Newell, and H. Segur, "The inverse scattering transform-Fourier analysis for nonlinear problems," *Studies in Applied Mathematics*, vol. 53, no. 4, pp. 249–315, 1974.

[7] S. Wahls and H. V. Poor, "Fast numerical nonlinear Fourier transforms," *IEEE Transactions on Information Theory*, vol. 61, no. 12, pp. 6957–6974, 2015.

[8] G. Boffetta and A. R. Osborne, "Computation of the direct scattering transform for the nonlinear Schrödinger equation," *Journal of computational physics*, vol. 102, no. 2, pp. 252–264, 1992.

[9] "Delft Center for Systems and Control," https://www.tudelft.nl/en/3me/about/departments/delft-center-for-systems-and-control, Accessed: 25/11/2021.

[10] CORDIS, "Nonlinear Fourier Transforms in Action," https://cordis.europa.eu/project/id/716669, Accessed: 27/11/2021.

[11] V. B. Matveev, "30 years of finite-gap integration theory," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1867, pp. 837–875, 2008.

[12] P. G. Grinevich, “Approximation theorem for the self-focusing nonlinear Schrödinger equation and for the periodic curves in r3,” *Physica D: Nonlinear Phenomena*, vol. 152, pp. 20–27, 2001.

[13] V. Kotlyarov and A. Its, “Periodic problem for the nonlinear Schrödinger equation,” *arXiv preprint arXiv:1401.4445*, 2014.

[14] Y. C. Ma and M. J. Ablowitz, “The periodic cubic Schrödinger equation,” *Studies in applied Mathematics*, vol. 65, no. 2, pp. 113–158, 1981.

[15] P. D. Lax, “Integrals of nonlinear equations of evolution and solitary waves,” *Communications on pure and applied mathematics*, vol. 21, no. 5, pp. 467–490, 1968.

[16] E. Tracy and H. Chen, “Nonlinear self-modulation: An exactly solvable model,” *Physical Review A*, vol. 37, no. 3, p. 815, 1988.

[17] S. Wahls, M. Bruehl, Y.-M. Fan, and C.-J. Huang, “Nonlinear Fourier analysis of free-surface buoy data using the software library FNFT,” in *International Conference on Offshore Mechanics and Arctic Engineering*, vol. 84386. American Society of Mechanical Engineers, 2020.

[18] A. R. Osborne, “Nonlinear ocean wave and the inverse scattering transform,” in *Scattering.* Elsevier, 2002, pp. 637–666.

[19] A. R. Osborne, D. T. Resio, A. Costa, S. P. de León, and E. Chirivì, “Highly nonlinear wind waves in Currituck Sound: dense breather turbulence in random ocean waves,” *Ocean Dynamics*, vol. 69, no. 2, pp. 187–219, 2019.

[20] S. Wahls, S. Chimmalgi, and P. Prins, “FNFT: A software library for computing nonlinear Fourier Transforms,” *The Journal of Open Source Software*, vol. 3, p. 597, 2018.

[21] “FNFT: Fast nonlinear Fourier transforms,” https://github.com/FastNFT/FNFT, Accessed: 11/10/2021.

[22] S. Wahls, S. Chimmalgi, P. J. Prins, and M. Brehler, “FastNFT/FNFT: Version 0.4.1,” 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3941976

[23] J. Yang, *Nonlinear waves in integrable and nonintegrable systems.* SIAM, 2010.

[24] M. I. Yousefi and F. R. Kschischang, “Information transmission using the nonlinear Fourier transform, Part II: Numerical methods,” *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 4329–4345, 2014.

[25] I. Mullyadzhanov, R. Mullyadzhanov, and A. Gelash, “Efficient Fourier-collocation method for full scattering data of Zakharov-Shabat periodic problem,” in *EGU General Assembly Conference Abstracts*, 2021, pp. EGU21–1720.

[26] I. Chekhovskoy, S. B. Medvedev, I. Vaseva, E. Sedov, and M. P. Fedoruk, “Introducing phase jump tracking-a fast method for eigenvalue evaluation of the direct Zakharov-Shabat problem,” *Communications in Nonlinear Science and Numerical Simulation*, vol. 96, p. 105718, 2021.