INTRODUCING THE FAST NONLINEAR FOURIER TRANSFORM

Sander Wahls and H. Vincent Poor

Department of Electrical Engineering, Princeton University, Princeton, NJ 08544, USA

ABSTRACT

The nonlinear Fourier transform (NFT; also: direct scattering transform) is discussed with respect to the focusing nonlinear Schrödinger equation on the infinite line. It is shown that many of the current algorithms for numerical computation of the NFT can be interpreted in a polynomial framework. Finding the continuous spectrum corresponds to polynomial multipoint evaluation in this framework, while finding the discrete eigenvalues corresponds to polynomial root finding. Fast polynomial arithmetic is used in order to derive algorithms that are about an order of magnitude faster than current implementations. In particular, an N sample discretization of the continuous spectrum can be computed with only $O(N \log^2 N)$ flops. A finite eigenproblem for the discrete eigenvalues that can be solved in $O(N^2)$ is also presented. The feasibility of this approach is demonstrated in a numerical example.

Index Terms— Nonlinear Fourier Transform, Inverse Scattering Transform, Schrödinger Equation, Optical Fiber Communication

I. INTRODUCTION

The term *nonlinear Fourier transform (NFT)* commonly refers to a family of transforms that decompose a time-domain signal into nonlinearly interacting waves [1]. (Another common denotation is *direct scattering transform*.) In contrast to the common Fourier transform, which is restricted to superpositions of sinusoidal waves, the NFT decomposes a signal with respect to richer sets of nonlinearly interacting waves such as, e.g., cnoidal waves or solitons ("particle-like waves"). The details of the decomposition depend on the actual transform considered, and are specified by the evolution equation (a special type of differential equations) that governs the signal. In this paper, we consider the NFT with respect to the *focusing nonlinear Schrödinger equation (NLS) on the infinite line*,

$$\mathrm{i}\, q_x = q_{tt} + 2|q|^2 q. \tag{1}$$

Here, $q(x,t)$ is a real signal that depends on a real spatial coordinate $x \geq 0$ as well as a real temporal coordinate $t \in \mathbb{R}$. The subscripts denote partial derivatives. The boundary conditions are specified for $|t| \to \infty$. The focusing NLS is of special interest in optical communications because it describes the evolution of a complex waveform envelope in Raman-amplified optical fiber.

Yousef and Kschischang [2], [3], [4] have recently proposed a communications system based on the NFT. Their proposal is inspired by the *orthogonal frequency division multiplexing (OFDM)* technique [5], in which information is embedded in the phases and amplitudes of certain eigensignals of the linear channel. The Fourier transform is used at the receiver in order to extract the embedded information from the incoming signal. Note that the approach of Yousef and Kschischang is different from *optical OFDM* [5], in which the nonlinear effects of the fiber are considered to be nuisances that have to be suppressed. Instead, they propose to exploit the nonlinearities in order to transmit information. The success of OFDM is rooted to a large extend in the *fast Fourier transform (FFT)* algorithm [6], which can compute a discretization of the Fourier transform in N samples with only $O(N \log N)$ operations. The lack of a comparably *fast nonlinear Fourier transform* is a major open problem [3, Sec. VII], [7, p. 63], [8, Sec. V].

In this paper, we propose several fast NFTs. We only discuss the NFT with respect to the NLS on the infinite line, but it should be straightforward to apply our approach to other NFTs on the infinite line. Periodic boundary conditions are yet to be investigated.

This work was supported in part by the German Research Foundation (DFG) under Grant WA 3139/1-1, and in part by the U. S. Office of Naval Research Grant N00014-12-1-0767.

II. THE NONLINEAR FOURIER TRANSFORM

We will now sketch the derivation of the NFT and its most important properties. The mathematical foundations of the NFT are quite involved. Therefore, for the sake of brevity, we refer the reader to [2], [7], [9] and [10] for details.

Consider a signal $q(x,t)$ governed by the NLS (1). We shall fix x because the NFT of $q(x,t)$ is (up to some factors) independent of x, and use q to denote the function $t \mapsto q(x,t)$ henceforth. Let us associate q with the eigenproblem $\mathbf{L}\mathbf{v} = \lambda \mathbf{v}$, where

$$\mathbf{L} := \mathrm{i} \begin{bmatrix} \frac{\partial}{\partial t} & -q(t) \\ -q(t)^* & -\frac{\partial}{\partial t} \end{bmatrix}$$

is the *Lax operator* and $t \mapsto \mathbf{v}(t,\lambda)$ is an eigenfunction. The eigenproblem is equivalent to solving the linear differential equation

$$\mathbf{v}_t(t,\lambda) = \begin{bmatrix} -\mathrm{i}\,\lambda & q(t) \\ -q(t)^* & \mathrm{i}\,\lambda \end{bmatrix} \mathbf{v}(t,\lambda) \tag{2}$$

for $\mathbf{v}(\cdot,\lambda)$. The special case $q \equiv 0$ of (2) can be solved in closed form for any λ. Let $q(t) \to 0$ rapidly as $|t| \to \infty$. Then, we can use the exact solutions for the case $q \equiv 0$ as boundary conditions:

$$\mathbf{v}(t \to +\infty, \lambda) \to \begin{bmatrix} 0 \\ \mathrm{e}^{\mathrm{i}\,\lambda t} \end{bmatrix},\ \mathbf{v}(t \to -\infty, \lambda) \to \begin{bmatrix} \mathrm{e}^{-\mathrm{i}\,\lambda t} \\ 0 \end{bmatrix}. \tag{3}$$

Fix any eigenvalue $\lambda \in \mathbb{C}$. We consider the two unique solutions

$$\mathbf{v}^{\pm}(t,\lambda) = \begin{bmatrix} v_1^{\pm}(t,\lambda) \\ v_2^{\pm}(t,\lambda) \end{bmatrix},$$

of the differential equation (2) that correspond to the two boundary conditions (3) at $\pm\infty$. Their adjoints are defined by

$$\tilde{\mathbf{v}}^{\pm}(t,\lambda) := \begin{bmatrix} v_2^{\pm}(t,\lambda)^* \\ -v_1^{\pm}(t,\lambda)^* \end{bmatrix}.$$

The eigenspace E_λ of the Lax operator $\mathbf{L}$ is two-dimensional, and each of the two pairs $(\mathbf{v}^{\pm}(\cdot,\lambda), \tilde{\mathbf{v}}^{\pm}(\cdot,\lambda))$ forms a basis of it. Thus,

there exists a unique matrix

$$\boldsymbol{\mathcal{S}}(\lambda) = \begin{bmatrix} s_{11}(\lambda) & s_{12}(\lambda) \\ s_{21}(\lambda) & s_{22}(\lambda) \end{bmatrix},$$

the so-called *scattering matrix*, such that

$$\begin{bmatrix} v_i^-(t,\lambda) \\ \tilde{v}_i^-(t,\lambda) \end{bmatrix} = \boldsymbol{\mathcal{S}}(\lambda) \begin{bmatrix} \tilde{v}_i^+(t,\lambda) \\ v_i^+(t,\lambda) \end{bmatrix} \qquad (i = 1, 2).$$

In particular, $v_1^-(t,\lambda) = s_{11}(\lambda)\tilde{v}_1^+(t,\lambda) + s_{12}(\lambda)v_1^+(t,\lambda)$. By letting $t \to +\infty$ in (3), we find that

$$a(\lambda) := s_{11}(\lambda) \leftarrow v_1^-(t,\lambda)\,\mathrm{e}^{\mathrm{i}\,\lambda t} \qquad (t \to +\infty). \tag{4}$$

Similarly, we find that

$$b(\lambda) := s_{12}(\lambda) \leftarrow v_2^-(t,\lambda)\,\mathrm{e}^{-\mathrm{i}\,\lambda t} \qquad (t \to +\infty), \tag{5}$$

as well as $s_{21}(\lambda) = b(\lambda^*)^*$ and $s_{22}(\lambda) = -a(\lambda^*)^*$.

This construction can be carried out for any eigenvalue λ of the Lax operator $\mathbf{L}$. It can be shown that any real λ is an eigenvalue of $\mathbf{L}$. The scattering matrix $\lambda \mapsto \boldsymbol{\mathcal{S}}(\lambda)$ turns out to be analytic on the real line and can be extended analytically into the upper half-plane. The non-real spectrum of $\mathbf{L}$ is symmetric with respect to the real line, and the non-real eigenvalues of $\mathbf{L}$ correspond exactly to the roots of a. Since a is analytic as well, the non-real eigenvalues of $\mathbf{L}$ must be isolated and countable.

Now, the *nonlinear Fourier transform* of the signal q consists of two components. The first component is the *continuous spectrum*

$$\hat{q} : \mathbb{R} \to \mathbb{C}, \qquad \hat{q}(\lambda) := b(\lambda)/a(\lambda).$$

It corresponds to the radiation components. The continuous spectrum of αq, $\alpha > 0$, converges towards the usual Fourier transform as $\alpha \searrow 0$. The second component is the *discrete spectrum*:

$$\tilde{q} : \{\tilde{\lambda}_j \in \mathbb{C}^+ : a(\tilde{\lambda}_j) = 0\} \to \mathbb{C}, \qquad \tilde{q}(\lambda) := b(\lambda)/a_\lambda(\lambda).$$

Here, we are interested in the eigenvalues $\tilde{\lambda}_j$ as well as in the corresponding values $\tilde{q}(\tilde{\lambda}_j)$. The discrete spectrum represents the soliton components of the signal.

III. NUMERICAL COMPUTATION: STATE OF THE ART

In this section, we review the state of the art in the numerical computation of $a(\lambda)$ and $b(\lambda)$. We will not discuss computation of the derivative $a_\lambda(\lambda)$ because this can be achieved with a straightforward extension of the discussed methods [3, Sec. IV.A]. Any method for computing $a(\lambda)$ and $b(\lambda)$ can be used to evaluate the continuous spectrum $\hat{q}(\lambda) = b(\lambda)/a(\lambda)$. The discrete eigenvalues $\tilde{\lambda}_j$ can be found by applying iterative root-finding such as Newton's method to $a(\lambda)$. There are also matrix methods for finding the $\tilde{\lambda}_j$ based on finite-dimensional approximations of the eigenproblem $\mathbf{L}\mathbf{v} = \lambda\mathbf{v}$ [3, Sec. IV.B]. We do not discuss them here because that is not necessary for the derivation of the fast algorithms.

III-A. Basic Idea[1]

The basic idea is to exploit (4) and (5). We choose T_1 and T_2 sufficiently[2] close to $-\infty$ and $+\infty$, respectively. Then, for any λ:

1) Replace the second boundary condition in (3) with $v_1(T_1,\lambda) = \mathrm{e}^{-\mathrm{i}\,\lambda T_1}$, $v_2(T_1,\lambda) = 0$.
2) Solve the differential equation (2) numerically for $\mathbf{v}(T_2,\lambda)$.
3) Use $a(\lambda) :\approx v_1(T_2,\lambda)\,\mathrm{e}^{\mathrm{i}\,\lambda T_2}$, $b(\lambda) :\approx v_2(T_2,\lambda)\,\mathrm{e}^{-\mathrm{i}\,\lambda T_2}$.

Hence, the main task is to solve (2) numerically.

III-B. Solution of the Differential Equation (2)

Several approaches have been discussed for the numerical solution of Equation (2) in the literature. We begin with some notation:

$$\begin{aligned} \epsilon &:= (T_2 - T_1)/(N-1), \\ \mathbf{v}[n,\lambda] &:= \mathbf{v}(T_1 + \epsilon(n-1), \lambda), \\ q[n] &:= q(T_1 + \epsilon(n-1)), \\ \mathbf{P}[n,\lambda] &:= \begin{bmatrix} -\mathrm{i}\,\lambda & q[n] \\ -q[n]^* & \mathrm{i}\,\lambda \end{bmatrix}. \end{aligned}$$

With this notation, the goal becomes to find $\mathbf{v}[N,\lambda]$ from $\mathbf{v}[1,\lambda]$.

Forward Discretization [3], [13]: This is a finite difference method. The derivative in (2) is approximated by $(\mathbf{v}[n+1,\lambda] - \mathbf{v}[n,\lambda])/\epsilon$, and the right side by $\mathbf{P}[n,\lambda]\mathbf{v}[n,\lambda]$. One can then solve for $\mathbf{v}[n+1,\lambda]$:

$$\mathbf{v}[n+1,\lambda] = (\mathbf{I} + \epsilon\mathbf{P}[n,\lambda])\mathbf{v}[n,\lambda].$$

Crank-Nicolson Method [3]: This is another finite difference method. The derivative in (2) is again replaced with $(\mathbf{v}[n+1,\lambda] - \mathbf{v}[n,\lambda])/\epsilon$, but the right side now becomes $(\mathbf{P}[n+1,\lambda]\mathbf{v}[n+1,\lambda] + \mathbf{P}[n,\lambda]\mathbf{v}[n,\lambda])/2$. Solving for $\mathbf{v}[n+1,\lambda]$ gives

$$\mathbf{v}[n+1,\lambda] = \left(\mathbf{I} - \frac{\epsilon}{2}\mathbf{P}[n,\lambda]\right)^{-1}\left(\mathbf{I} + \frac{\epsilon}{2}\mathbf{P}[n,\lambda]\right)\mathbf{v}[n,\lambda].$$

Boffetta-Osborne Method [3], [9], [14]: Fix any interval $[T_1 + (n-1)\epsilon, T_1 + n\epsilon)$, and approximate $q(t)$ in (2) by $q[n]$. The result is a first order linear differential equation. With $\mathbf{expm}$ denoting the matrix exponential, its solution can be given in closed-form:

$$\mathbf{v}[n+1,\lambda] = (\mathbf{I} + \mathbf{expm}(\epsilon\mathbf{P}[n,\lambda]))\,\mathbf{v}[n,\lambda].$$

Ablowitz-Ladik Discretization [3], [13], [15]: The diagonal entries of the matrix $\mathbf{I} + \epsilon\mathbf{P}[n,\lambda]$ in the forward discretization are $1 \mp \epsilon\,\mathrm{i}\,\lambda$. They can be approximated by $z^\pm = \mathrm{e}^{\mp\,\mathrm{i}\,\epsilon\lambda}$. Then,

$$\mathbf{v}[n+1,\lambda] = \begin{bmatrix} z & \epsilon q[n] \\ -\epsilon q[n]^* & z^{-1} \end{bmatrix}\mathbf{v}[n,\lambda].$$

Typical properties of the continuous NLS like the emergence of solitons can be observed with this discretization for any ϵ (instead of sufficiently small ϵ only).

Normalized Methods: The transition matrices in these methods can be scaled by non-zero factors (independent of λ) without changing the approximations of the NFT. Normalization can improve numerical properties. For example, the factor $1/\sqrt{1+\epsilon^2|q[n]|^2}$ leads to a normalized Ablowitz-Ladik method [3, Sec. III.E].

Other Methods: We omit the central difference method [3] and Runge-Kutta methods [3], [9] because of the page limit.

IV. OUTLINE OF THE FAST POLYNOMIAL APPROACH

In this section, we outline a polynomial framework that unifies the numerical methods for the computation of $a(\lambda)$ and $b(\lambda)$ discussed in the previous section. Later, this will enable us to exploit fast polynomial arithmetic. From an abstract point of view, the numerical methods correspond to evaluating a mapping

[1] Almost all algorithms in the literature seem to make this ansatz. A notable exception is the recent paper of Olivier et al. [11], which replaces the boundary conditions at infinity with periodic boundary conditions for a very large period. The resulting problem then is to be solved using the Floquet-Fourier-Hill method of Deconinck and Kutz [12].

[2] What is sufficient here depends on the decay of q. We are currently not aware of any formal expression that quantifies the approximation error.

$\mathbf{v}[N,\lambda] = \mathbf{f}(\lambda)$ (which is the outcome of the chosen iteration). Our main observation is that for all but one of the methods, the mapping $\mathbf{f}$ is of a special form. As we shall see below, we have

$$\mathbf{f}(z) = \mathbf{v}[N,z] = \frac{\mathbf{S}(z)}{d(z)}\mathbf{v}[1,z], \tag{6}$$

where $z = \phi(\lambda)$ is a suitable coordinate transform, and

$$\mathbf{S}(z) := \prod_{n=1}^{N} \mathbf{S}[n,z], \quad d(z) := \prod_{n=1}^{N} d[n,z]$$

for matrix-valued polynomials $z \mapsto \mathbf{S}[n,z]$, and scalar polynomials $z \mapsto d[n,z]$. By construction, also $\mathbf{S}(z)$ and $d(z)$ are polynomials. Now, evaluation of $\hat{q}(\lambda)$ at N points is a multipoint evaluation problem, while finding the eigenvalues $\tilde{\lambda}_j$ is a root-finding problem.

We close this section with a summary of how the numerical methods in Sec. III-B fit into the polynomial formulation (6):

Euler's Method: $\phi(\lambda) = \lambda$, $d[n,z] = 1$, $\mathbf{S}[n,z] = \mathbf{I} + \epsilon\mathbf{P}[n,\lambda]$

Crank-Nicolson: $\phi(\lambda) = \lambda$,

$$d[n,z] = \det\left(\mathbf{I} - \frac{\epsilon}{2}\mathbf{P}[n,z]\right) = 1 - \frac{\epsilon^2}{4}(z^2 + |q[n]|^2),$$

$$\begin{aligned}\mathbf{S}[n,z] &= d[n,z]\left(\mathbf{I} - \frac{\epsilon}{2}\mathbf{P}[n,z]\right)^{-1}\left(\mathbf{I} + \frac{\epsilon}{2}\mathbf{P}[n,z]\right) \\ &= \begin{bmatrix} d[n,z] - \epsilon\,\mathrm{i}\,z & \epsilon q[n] \\ -\epsilon q[n]^* & d[n,z] + \epsilon\,\mathrm{i}\,z \end{bmatrix}.\end{aligned}$$

Boffetta-Osborne: This method does not fit into (6) because $\mathbf{S}(z)/d(z)$ is a rational matrix, but the matrix exponential is irrational. However, we can approximate $\mathbf{expm}$. A truncated Taylor series gives: $\phi(\lambda) = \lambda$, $d[n,z] = 1$, $\mathbf{S}[n,z] = \sum_{m=0}^{M} \frac{\epsilon^n}{n!}\mathbf{P}[n,z]^n$ [16, p. 9]. Another option is "scaling and squaring" of the first-order Taylor approximation: $\phi(\lambda) = \lambda$, $d[n,z] = 1$, $\mathbf{S}[n,z] = (\mathbf{I} + \frac{\epsilon}{M}\mathbf{P}[n,z])^M$ [16, p. 12]. In both cases, increasing M will improve the approximation at the cost of higher operation counts.

Ablowitz-Ladik: $\phi(\lambda) = \mathrm{i}\,\mathrm{e}^{-\mathrm{i}\,\lambda\epsilon}$, $d[n,z] = z$, and

$$\mathbf{S}[n,z] = \begin{bmatrix} z^2 & \epsilon q[n] z \\ -\epsilon q[n]^* z & 1 \end{bmatrix}.$$

The normalized Ablowitz-Ladik method is obtained if $\mathbf{S}[n,z]$ is replaced with $\mathbf{S}[n,z]/\sqrt{1+\epsilon^2|q[n]|^2}$.

V. FAST POLYNOMIAL ARITHMETIC

In this section, we review some results on fast polynomial arithmetic that are needed to implement our fast NFT as outlined in the previous section. We will always assume that polynomials are given with respect to the standard monomial basis.

Fast Product of Two Polynomials: The FFT can be used to compute the product $p = qr$ of two polynomials $q(z) = \sum_{n=0}^{D} q_n z^n$ and $r(z) = \sum_{n=0}^{D} r_n z^n$ in $O(D \log D)$ operations. As we are using a monomial basis representation, this means that the algorithm finds the p_n such that $p(z) = \sum_{n=0}^{2D-1} p_n z^n$ from the q_n and r_n.

Fast Product of N Polynomials: The product $\prod_{n=1}^{N} a_n$ of N numbers $a_1, \ldots, a_N$ can be found using only $O(\log N)$ multiplications with the following recursion: $\prod_{n=1}^{N} a_n = (\prod_{n=1}^{\lfloor N/2 \rfloor} a_n)(\prod_{n=\lfloor N/2 \rfloor + 1}^{N} a_n)$. The same idea can be used to find the product of N polynomials. See Alg. 1 for an explicit, non-recursive implementation. The product of two polynomials of degree at most D can be found with $O(D \log D)$ operations with the help of the FFT (see above). The degrees of the polynomials double at most in each iteration of the algorithm, but the number of products that we have to form halves at least. Thus, the cost is $O\left(\frac{N}{2} K \log K\right)$ for the first iteration, $O\left(\frac{N}{4} 2K \log(2K)\right)$ for the second, $O\left(\frac{N}{8} 4K \log(4K)\right)$ for the third, and so on. We perform at most $\log_2(N)$ iterations. Hence, the cost of each iteration is at most $O\left(NK \log(2^{\log_2 N} K)\right) = O\left(NK \log(NK)\right)$. We arrive at a total cost of $O\left(NK \log(NK) \log N\right)$.

Algorithm 1 Fast product of N polynomials

Input: N polynomials $p_1, \ldots, p_N$ of degree at most K
Outp.: $p = \prod_{n=1}^{N} p_n$

- while $N \geq 2$ do:
  - $N_m \leftarrow N \mod 2$
  - $N \leftarrow \frac{N - N_m}{2}$
  - for $n = 1, \ldots, N - N_m$ do: $p_n \leftarrow p_{2n-1} p_{2n}$
  - if $N_m \neq 0$ then: $p_N \leftarrow p_{2N-1}$
- $p \leftarrow p_1$

Fast Multipoint Evaluation: The fastest way to evaluate a polynomial p of degree D at only one point z is Horner's method, which requires $O(D)$ operations. Surprisingly, it is possible to evaluate p at D points $z_1, \ldots, z_D$ simultaneously with only $O(D \log^2 D)$ operations [17]. However, the naive approach requires repeated polynomial divisions which may cause numerical problems. Pan et al. [18] presented a Vandermonde matrix approach that avoids polynomial divisions, but numerical problems may still occur. Hence, specialized fast algorithms should be used whenever possible. For example, the FFT [6] can be used for equispaced grids on the complex unit circle. The chirp transform [19] can be used for certain arguments on a spiral in the complex plane, and the discrete Laplace transform [20], [21] can be used for arbitrary points on the open real interval $(0, 1)$.

Root-Finding Via Matrix Methods: In these methods, a companion matrix is constructed such that the eigenvalues of that matrix are equal to the roots of the polynomial. Then, e.g., the QR algorithm is used to find the eigenvalues. The advantage of this method is that we do not need any a-priori knowledge about the eigenvalues. The disadvantage used to be its computational costs of $O(D^3)$, where D is the degree of the polynomial. However, recently several fast algorithms that exploit the special structure of companion matrices have been proposed [22], [23], [24], [25]. Their computational costs are only $O(D^2)$. While the stability of these fast matrix methods has not been proven yet, numerical experiments look promising.

Root-Finding Via Search Methods: Search methods assume that we have initial guesses for the roots and then iteratively refine them. Their advantage is that a-priori knowledge on the number and approximate location of the roots can be integrated. Newton's method is a popular example. The cost of one iteration is usually $O(D)$, but giving an overall complexity estimate is difficult because it depends on the initial guesses and the number of iterations necessary. The best upper bound on the number of operations necessary to find all roots (assumed simple and sufficiently separated) using Newton's method without a-priori knowledge known to the authors is $O(D^3 \log^3 D)$ [26]. Note that fast polynomial multipoint evaluation is required to establish this bound. Hence, the numerical stability of the method is not assured. For the case with a-priori knowledge, very fast algorithms have recently been proposed in [27]. Their numerical stability is yet to be explored.

Algorithm 2 Fast computation of the continuous spectrum

Input: $q[1], \ldots, q[N]$, T_1, T_2
Outp.: $\hat{q}[1], \ldots, \hat{q}[N]$

- Use a fast method to find the monomial representation of $\mathbf{S}$
- Use a fast method to find the monomial representation of d
- For $n = 1, \ldots, N$ do: $\lambda_n \leftarrow T_1 + (n-1)\frac{T_2 - T_1}{N-1}$, $z_n \leftarrow \phi(\lambda_n)$
- Use fast multipoint evaluation to find $\mathbf{S}(z_1), \ldots, \mathbf{S}(z_N)$
- Use fast multipoint evaluation to find $d(z_1), \ldots, d(z_N)$
- For $n = 1, \ldots, N$ do:
  - $\mathbf{v}[N, \lambda_n] \leftarrow \frac{\mathbf{S}(z_n)}{d(z_n)} \mathbf{v}[1, \lambda_n]$
  - $\hat{q}[n] \leftarrow \mathrm{e}^{-2\,\mathrm{i}\,\lambda_n T_2} \frac{v_2[N,\lambda_n]}{v_1[N,\lambda_n]}$

Algorithm 3 Fast computation of the discrete eigenvalues

Input: $q[1], \ldots, q[N]$, T_1, T_2
Outp.: $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_M$

- Use a fast method to find the monomial representation of $\mathbf{S}$
- Use a fast method to find the roots $r_1, \ldots, r_L$ of $[\mathbf{S}]_{1,1}$ (i.e., the upper left element of $\mathbf{S}$)
- For $j = 1, \ldots, L$ do: $\tilde{\lambda}_j \leftarrow \phi^{-1}(r_j)$

VI. FAST COMPUTATION: CONTINUOUS SPECTRUM

Our general fast approach to computing the continuous spectrum is given in Alg. 2. It can be run for any choice of $\phi(\lambda)$, $d[n, z]$, and $\mathbf{S}[n, z]$ given in Sec. IV. A fast $O\left(NK \log(NK) \log N\right)$ method to perform the monomial representations of $\mathbf{S}$ and d has been given in Sec. V. Fast $O\left(NK \log^2(NK)\right)$ methods for multipoint evaluation have also been discussed. Note that the upper bound K on the degrees of $z \mapsto d[n, z]$ and $z \mapsto \mathbf{S}[n, z]$ can be chosen small for all methods.[3] Since the coordinate transform ϕ is $O(1)$ in all considered cases, the overall complexity of our meta-algorithm amounts to $O\left(NK \log^2(NK)\right)$ operations. This is about a magnitude faster than the naive implementation, which is $O(N^2)$.

VII. FAST COMPUTATION: DISCRETE SPECTRUM

Our general fast approach to computing the discrete eigenvalues $\tilde{\lambda}_j$ is given in Alg. 3.[4] We do not discuss finding the $\tilde{q}(\tilde{\lambda}_j)$ here. Once the eigenvalues are known, computation of the $\tilde{q}(\tilde{\lambda}_j)$ reduces again to polynomial multipoint evaluation. We omit the details because of the page limit. As already discussed in Sec. VI, we have a complexity of $O\left(NK \log(NK) \log N\right)$ for finding the monomial representation of $\mathbf{S}$. The complexities of various fast root finding procedures have been discussed in Sec. V. Using the algorithm of [25], we obtain a fast $O(N^2)$ matrix method. This is again a magnitude faster than naive matrix methods, which are $O(N^3)$ [3]. (A comparison with specialized search methods like [3, Alg. 1] is difficult (see Sec. V), and beyond the scope of this paper.)

VIII. NUMERICAL EXAMPLE

We have implemented two instances of our meta-algorithms, Alg. 2 and Alg. 3, for the normalized Ablowitz-Ladik method in MATLAB[5] in order to demonstrate the feasibility of our proposed polynomial approach. We implemented the fast product of N polynomials (Alg. 1) using the cconv routine from the Signal Processing Toolbox. The routine czt (a chirp transform) from the same toolbox was used for fast multipoint evaluation. Finally, we used a fast polynomial root finder as described in [25]. We compare these two algorithms with a naive MATLAB implementation of the normalized Ablowitz-Ladik method for the continuous spectrum as well as the Ablowitz-Ladik matrix method [3, IV.B.2] for the discrete spectrum, and apply them to a Satsuma-Yajima pulse [3], [28] for various numbers of samples N. The fast methods gave the same results (up to some different spurious, well-separated discrete eigenvalues) as the standard methods, which was confirmed by examining the relative differences (continuous spectrum) and by visual inspection (discrete spectrum), respectively. Fig. 1 shows the runtimes *per sample*. While the absolute values may not be representative because of implementation details, we see that the runtimes per sample of our fast algorithms are approximately constant (continuous spectrum) and linear (discrete spectrum), which corresponds to approximately linear and quadratic total runtimes, respectively. In contrast, the complexities of the standard methods have quadratic and cubic total runtimes, respectively.

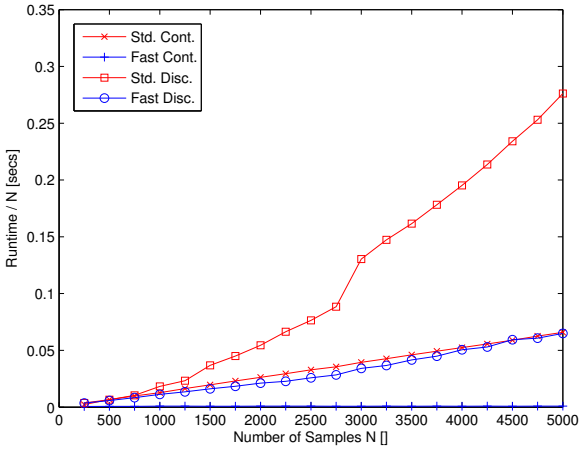

Fig. 1. Numerical Experiment

IX. CONCLUSION

A unified polynomial framework for the numerical computation of the NFT with respect to the focusing NLS on the infinite line has been presented. In this framework, computation of the continuous spectrum corresponds to polynomial multipoint evaluation, while finding the discrete eigenvalues corresponds to polynomial root finding. Fast polynomial arithmetic can be used to solve these problems about an order of magnitude faster than previous algorithms. The practical relevance of the framework has been demonstrated with an implementation of two fast Ablowitz-Ladik methods.

We are currently working on fast implementations for the other methods. First results indicate that more severe finite precision effects have to be taken care of with these methods because the polynomials are evaluated on the real line instead of the unit circle. We are also trying to extend our framework to other NFTs.

[3]Euler: 1; Crank-Nicolson, Ablowitz-Ladik: 2; Boffetta-Osborne: M

[4]Note that $d(z)$ has finite roots in the C.-N. method. In that case, any computed eigenvalue r_j that coincides with a root of $d(z)$ should be removed in an additional last step (counting multiplicities) because the approximation of $a(z)$ satisfies $\tilde{a}(z) = \mathrm{e}^{-\lambda(T_1+T_2)}[\mathbf{S}]_{1,1}(z)/d(z)$. The roots of $d(z)$ can be found in $O(NK^2)$ from the roots of the $d[n, z]$.

[5]MATLAB is a registered trademark of The Mathworks, Natick, MA.

X. REFERENCES

[1] M. J. Ablowitz, D. J. Kaup, A. C. Newell, and H. Segur, "The inverse scattering transform – Fourier analysis for nonlinear problems," *Stud. Appl. Math.*, vol. 53, pp. 249–315, 1974.

[2] M. I. Yousefi and F. R. Kschischang, "Information transmission using the nonlinear Fourier transform, Part I: Mathematical tools," Preprint, Feb. 2012, arXiv:1202.3653v1 [cs.IT].

[3] M. I. Yousefi and F. R. Kschischang, "Information transmission using the nonlinear Fourier transform, Part II: Numerical methods," Preprint, Apr. 2012, arXiv:1204.0830v1 [cs.IT].

[4] M. I. Yousefi and F. R. Kschischang, "Information transmission using the nonlinear Fourier transform, Part III: Spectrum modulation," Preprint, Feb. 2013, arXiv:1302.2875v1 [cs.IT].

[5] J. Armstrong, "OFDM for optical communications," *J. Lightwave Technol.*, vol. 27, no. 3, pp. 189–204, 2009.

[6] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comp.*, vol. 19, no. 90, pp. 297–301, Apr. 1965.

[7] A. R. Osborne, *Nonlinear Ocean Waves and the Inverse Scattering Transform*, Academic Press, 1st edition, 2010.

[8] E. Meron, M. Feder, and M. Shtaif, "On the achievable communication rates of generalized soliton transmission systems," Preprint, July 2012, arXiv:1207.0297v1 [cs.IT].

[9] S. Burtsev, R. Camassa, and I. Timofeyev, "Numerical algorithms for the direct spectral transform with applications to nonlinear Schrödinger type systems," *J. Comput. Phys.*, vol. 147, no. 1, pp. 166–186, Nov. 1998.

[10] M. J. Ablowitz, B. Prinari, and A. D. Trubatch, *Discrete and Continuous Nonlinear Schrödinger Systems*, vol. 302 of *London Math. Soc. Lect. Notes Ser.*, Cambridge Univ. Press, 2004.

[11] C. P. Olivier, B. M. Herbst, and M. A. Molchan, "A numerical study of the large-period limit of a Zakharov-Shabat eigenvalue problem with periodic potentials," *J. Phys. A: Math. Theor.*, vol. 45, no. 25, 2012.

[12] B. Deconinck and J. Nathan Kutz, "Computing spectra of linear operators using the Floquet-Fourier-Hill method," *J. Comput. Phys.*, vol. 219, no. 1, pp. 296–321, Nov. 2006.

[13] J. A. C. Weideman and B. M. Herbst, "Finite difference methods for an AKNS eigenproblem," *Math. Comput. Simul.*, vol. 43, no. 1, pp. 77–88, Jan. 1997.

[14] G. Boffetta and A. R. Osborne, "Computation of the direct scattering transform for the nonlinear Schroedinger equation," *J. Comput. Phys.*, vol. 102, no. 2, pp. 252–264, Oct. 1992.

[15] M. J. Ablowitz and J. F. Ladik, "Nonlinear differential-difference equations and Fourier analysis," *J. Math. Phys.*, vol. 17, no. 6, pp. 1011–1018, June 1976.

[16] C. Moler and C. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM Review*, vol. 45, no. 1, pp. 3–49, 2003.

[17] R. Moenck and A. Borodin, "Fast modular transforms via division," in *Proc. Annu. Symp. Switching Automata Theory (SWAT)*, College Park, MD, Oct. 1972.

[18] V. Pan, A. Sadikou, E. Landowne, and O. Tiga, "A new approach to fast polynomial interpolation and multipoint evaluation," ICSI Tech. Rep. TR-92-055, UC Berkeley, Aug. 1992, http://www.icsi.berkeley.edu/pubs/techreports/tr-92-055.pdf.

[19] L. Rabiner, R. Schafer, and C. Rader, "The chirp z-transform algorithm," *IEEE Trans. Audio Electroacoust.*, vol. 17, no. 2, pp. 86–92, 1969.

[20] V. Rokhlin, "A fast algorithm for the discrete Laplace transformation," *J. Complexity*, vol. 4, no. 1, pp. 12–32, 1988.

[21] J. Strain, "A fast Laplace transform based on Laguerre functions," *Math. Comp.*, vol. 58, no. 197, pp. 275–283, Jan. 1992.

[22] D. A. Bini, L. Gemignani, and V. Y. Pan, "Fast and stable QR eigenvalue algorithms for generalized companion matrices and secular equations," *Numer. Math.*, vol. 100, no. 3, pp. 373–408, 2005.

[23] S. Chandrasekaran, M. Gu, J. Xia, and J. Zhu, "A fast QR algorithm for companion matrices," *Oper. Theory Adv. Appl.*, vol. 179, pp. 111–143, 2007.

[24] M. Van Barel, R. Vandebril, P. Van Dooren, and K. Frederix, "Implicit double shift QR-algorithm for companion matrices," *Numer. Math.*, vol. 116, no. 2, pp. 177–212, Aug. 2010.

[25] P. Boito, Y. Eidelman, L. Gemignani, and I. Gohberg, "Implicit QR with compression," *Indag. Math.*, vol. 23, no. 4, pp. 733–761, Dec. 2012.

[26] D. Schleicher, "On the efficient global dynamics of Newton's method for complex polynomials," Preprint, Aug. 2011, arXiv:1108.5773v1 [math.DS].

[27] J. M. McNamee and V. Y. Pan, "Efficient polynomial root-refiners: A survey and new record efficiency estimates," *Comput. Math. Appl.*, vol. 63, no. 1, pp. 239–254, Jan. 2012.

[28] J. Satsuma and N. Yajima, "Initial value problems of one-dimensional self-modulation of nonlinear waves in dispersive media," *Prog. Theor. Phys. Suppl.*, vol. 55, pp. 284–306, 1974.