國立臺灣大學
National Taiwan University

Inverse Nonlinear Fourier Transform: Applications in QSP

Wen Perng, Po-Chieh Liu

Special Topics in Quantum Information Science

2025 May 27

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

4 Conclusion

About this Tutorial

This tutorial is about linking the similarity between the field of quantum signal processing (QSP) and nonlinear Fourier analysis (NLFA) on $\mathrm{SU}(2)$.

From the lecture, we have learned the *layer-stripping* method [T05; GSLW19] for obtaining the phase of QSP protocols. It is reported that such method is numerically unstable. Henceforth, many different optimization methods were proposed. Of them, we focus especially on those that were inspired by the relationship between QSP and NLFA [AMT24].

We demonstrate the (1) Riemann–Hilbert–Weiss (RHW) algorithm [ALMTW24], (2) the half-Cholesky (HC) method [NY24] which speeds up RHW, and (3) the inverse nonlinear FFT (iNLFFT) [NSYL25], which is the state-of-the-art.

Table of Contents

1 Introduction

- QSP & NLFT
- Prior Works

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

4 Conclusion

Table of Contents

1 Introduction

- QSP & NLFT
- Prior Works

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

4 Conclusion

Let us recap what QSP is: let

$$W(x)=\begin{bmatrix} x & \mathrm{i}\sqrt{1-x^2} \\ \mathrm{i}\sqrt{1-x^2} & x \end{bmatrix} = \mathrm{e}^{\mathrm{i}(\arccos x)X} \tag{1}$$

and $\Phi=(\phi_0,\phi_1,\ldots,\phi_n)$, we have

$$V_\Phi(x)=\mathrm{e}^{\mathrm{i}\phi_0 Z}W(x)\mathrm{e}^{\mathrm{i}\phi_1 Z}W(x)\cdots W(x)\mathrm{e}^{\mathrm{i}\phi_n Z}=\begin{bmatrix} P(x) & * \\ * & * \end{bmatrix}. \tag{2}$$

The block encoding we wish to retrieve is the degree n polynomial with definite parity:

$$P(x). \tag{3}$$

Nonlinear Fourier Transform

Def. (Nonlinear Fourier Transform)

The NLFT of a sequence $F : \mathbb{Z} \to \mathbb{C}$, or simply $F = (\ldots, F_{-1}, F_0, F_1, \ldots)$, is the product $\mathcal{F}(z)$ defined by: let $a^*(z) = \overline{a(\overline{z}^{-1})}$,

$$\mathcal{F}(z) = \begin{bmatrix} a(z) & b(z) \\ -b^*(z) & a^*(z) \end{bmatrix} := \lim_{k \to \infty} \mathcal{F}_k(z), \quad \mathcal{F}_{-\infty}(z) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{4}$$

$$\mathcal{F}_k(z) = \begin{bmatrix} a_k(z) & b_k(z) \\ -b_k^*(z) & a_k^*(z) \end{bmatrix} = \mathcal{F}_{k-1}(z) \cdot \frac{1}{\sqrt{1 + |F_k|^2}} \begin{bmatrix} 1 & F_k z^k \\ -\overline{F}_k z^{-k} & 1 \end{bmatrix}. \tag{5}$$

Notice that for $z \in \mathbb{T}$ (unit circle), we have $\mathcal{F}_k(z) \in \mathrm{SU}(2)$ for all k. We often simply write $\mathcal{F} = (a, b)$ in place of the whole matrix.

Conjugation of Functions

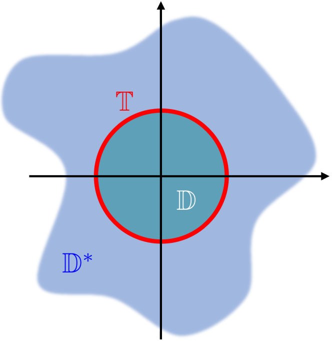
For any function $a : \mathbb{D} \to \mathbb{C}$, let us define the following map that preserves analyticity

$$a^*(z) = \overline{a(\overline{z}^{-1})}, \tag{6}$$

where $a^* : \mathbb{D}^* \to \mathbb{C}$. The action of the operator * is basically:

$$cz^n \overset{*}{\mapsto} \overline{c}z^{-n}. \tag{7}$$

Notice that on $\mathbb{T}$, $a^*(z) = (a(z))^*$.



Immediately, let us link the two disparate mathematical fields: let $x = \cos\theta$, $z = \mathrm{e}^{\mathrm{i}2\theta}$, and $F_k = \mathrm{i}\tan\phi_k$. Then since

$$W(x) = \mathrm{e}^{i\theta X}, \qquad \frac{1}{\sqrt{1+|F_k|^2}}\begin{bmatrix} 1 & F_k z^k \\ -\overline{F}_k z^{-k} & 1 \end{bmatrix} = \mathrm{e}^{\mathrm{i}k\theta Z}\mathrm{e}^{\mathrm{i}\phi_k X}\mathrm{e}^{-\mathrm{i}k\theta Z}, \tag{8}$$

we have, for $F = (F_0, F_1, \ldots, F_n)$,

$$\begin{aligned} HV_\Phi(x)H &= H\mathrm{e}^{\mathrm{i}\phi_0 Z}\mathrm{e}^{i\theta X}\mathrm{e}^{\mathrm{i}\phi_1 Z}\mathrm{e}^{i\theta X}\cdots\mathrm{e}^{i\theta X}\mathrm{e}^{\mathrm{i}\phi_n Z}H \\ &= H\mathrm{e}^{\mathrm{i}\phi_0 Z}HH\mathrm{e}^{i\theta X}HH\mathrm{e}^{\mathrm{i}\phi_1 Z}HH\mathrm{e}^{i\theta X}H\cdots H\mathrm{e}^{i\theta X}HH\mathrm{e}^{\mathrm{i}\phi_n Z}H \\ &= \mathrm{e}^{\mathrm{i}\phi_0 X}\mathrm{e}^{i\theta Z}\mathrm{e}^{\mathrm{i}\phi_1 X}\mathrm{e}^{i\theta Z}\cdots\mathrm{e}^{i\theta Z}\mathrm{e}^{\mathrm{i}\phi_n X} \\ &= \mathrm{e}^{\mathrm{i}\phi_0 X}\left(\mathrm{e}^{\mathrm{i}\theta Z}\mathrm{e}^{\mathrm{i}\phi_1 X}\mathrm{e}^{-\mathrm{i}\theta Z}\right)\left(\mathrm{e}^{\mathrm{i}2\theta Z}\cdots\right)\left(\mathrm{e}^{\mathrm{i}n\theta Z}\mathrm{e}^{\mathrm{i}\phi_n X}\mathrm{e}^{-\mathrm{i}n\theta Z}\right)\mathrm{e}^{\mathrm{i}n\theta Z} \\ &= \mathcal{F}(z)\mathrm{e}^{\mathrm{i}n\theta Z}. \qquad \blacksquare \end{aligned}$$

Table of Contents

1 Introduction

- QSP & NLFT
- Prior Works

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

4 Conclusion

Prior Works – Layer-Stripping

Before we discuss how NLFT aids in our goal of QSP, let us review some history in the development of phase finding in QSP.

We have seen the *layer-stripping method* as introduced in [GSLW19] in course:

$$\begin{bmatrix} P'(x) & * \\ \mathrm{i}\sqrt{1-x^2}Q'^*(x) & * \end{bmatrix} = \begin{bmatrix} P(x) & * \\ \mathrm{i}\sqrt{1-x^2}Q^*(x) & * \end{bmatrix} \mathrm{e}^{-\mathrm{i}\phi_n Z} W^\dagger(x),$$

then

$$P'(x) = \mathrm{e}^{-\mathrm{i}\phi_n} x \underbrace{P(x)}_{p_n x^n + \cdots} + \mathrm{e}^{\mathrm{i}\phi_n}(1-x^2) \underbrace{Q(x)}_{q_{n-1}x^{n-1}+\cdots}$$

reduces its order if and only if $\mathrm{e}^{\mathrm{i}2\phi_n} = p_n/q_{n-1}$.

However, this method is numerically unstable! (Or is it?)

Prior Works – Layer-Stripping

Let us analyze the complexity to the layer-stripping method:

- The amount of FLOPs required is

$$\underbrace{O(n)}_{n \text{ phase terms}} \cdot \underbrace{O(n)}_{P \text{ to } P'} = O(n^2).$$

- For each computation, $\Theta(n\log(n/\varepsilon))$ bits of precision is required [H19] such that $\left\|P-\hat{P}\right\|_\infty < \varepsilon$.

The definition to a *numerically stable* algorithm is its required number of bits is of order $O(\mathrm{polylog}(n/\varepsilon))$.

In [NSYL25], however, it is shown that stability can be achieved for certain a's (outer).

Prior Works

Other methods were developed, such as: root finding, Prony's method [Y22], iterative optimization methods.

But before any of that, we should ask ourselves whether the solution is unique? Many weird behaviors in the phase finding of QSP are discussed in [WDL22]. Consider requiring f to be *real*[1]. Due to the parity constraint on P being a real degree $n = 2d$ even polynomial, we only have $d + 1$ degrees of freedom. Hence, we should expect

$$\Phi = (\psi_d, \ldots, \psi_1, \underbrace{\psi_0, \psi_1, \ldots, \psi_d}_{\Psi}) \tag{9}$$

to be a symmetric phase Ψ. In the case of symmetric phase, we denote $V_\Phi = V_\Psi$, and this becomes the study of *symmetric QSP*.

[1] This constraint can be lifted.

Prior Works – Iteration Methods

Iterative methods were proposed: consider the following optimization problem

$$\Psi^* = \arg\min_{\Psi} \left\| \hat{P}_\Psi - P \right\|_2^2 = \arg\min_{\Psi} \sum_{k=1}^{d} \left| \hat{P}_\Psi(x_k) - P(x_k) \right|^2 , \tag{10}$$

where P is the function obtained from the QSP protocol with symmetric phase Ψ, and x_k are the Chebyshev nodes. We can optimize via

1. Fixed-point iteration [DLNW24a]:

$$\Psi \mapsto \Psi - \frac{1}{2}(\hat{P}_\Psi - P).$$

Note that $\mathrm{D}\hat{P}_\Psi(0) = 2\mathbb{1}$.

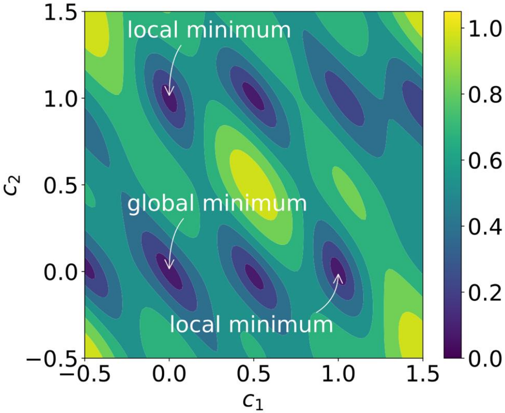
2. Newton's method [DLNW24b]:

$$\Psi \mapsto \Psi - \underbrace{\mathrm{D}\hat{P}_\Psi^{-1}}_{\mathrm{Hess}^{-1}} \underbrace{(\hat{P}_\Psi - P)}_{\mathrm{grad}} .$$

Prior Works – Maximal Solution

Even with the phase reduced, we cannot ensure our iterations converge. In fact, in [WDL22], it is observed that there are multiple local minimas!

There exists a global minima near the special initial condition $\Psi_0 = (0, \dots, 0)$. This solution is termed the *maximal solution*.



Prior Works – Complexity Comparison

A summary of existing algorithms for inverse NLFT is as below:

Algorithm	Time Complexity[2]	Coherence[3]	Stability
FPI [DLNW24a]	$\tilde{O}(n^2 \log \varepsilon^{-1})$	$1-\eta \ll 1$	Y
FFPI [NY24]	$O(n \log^2 n \log \varepsilon^{-1})$	$1-\eta \ll 1$	Y
Layer-Stripping [T05]	$O(n^2)^*$	Full	N/Y
RHW [ALMTW24]	$O(n^4)^*$	Full	Y
HC [NY24]	$O(n^2)^*$	Full	Y
iNLFFT [NSYL25]	$O(n \log^2 n)^*$	Full	Y

[2] The algorithms with * requires the Weiss algorithm, costing an additional $O(n \,\mathrm{polylog}(n/\varepsilon))$.

[3] $\|f\|_\infty = 1-\eta$. If $\eta \sim 0$, it is fully-coherent.

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

- GQSP & NLFT
- Properties

3 Inverse Nonlinear Fourier Transform

4 Conclusion

History

Here we give some more information on NLFA.

Just like linear Fourier transform is invented to deal with the heat diffusion equation, which is a linear PDE; the nonlinear Fourier transform is invented to solve nonlinear PDEs, such as the *scattering equations* or the *nonlinear Schrödinger's equation*:

$$\mathrm{i}\psi_t + \psi_{xx} + 2\left|\psi\right|^2 \psi = 0.$$

This works with continuous NLFT on $\mathrm{SU}(2)$. In this talk, however, we especially work with the discrete NLFT on $\mathrm{SU}(2)$. Other variants of NLFT also exist, such as working on $\mathrm{SU}(1,1)$:

$$\mathcal{F}_k(z) = \mathcal{F}_{k-1}(z) \cdot \frac{1}{\sqrt{1 - \left|F_k\right|^2}} \begin{bmatrix} 1 & F_k z^k \\ \overline{F}_k z^{-k} & 1 \end{bmatrix}.$$

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

- GQSP & NLFT
- Properties

3 Inverse Nonlinear Fourier Transform

4 Conclusion

Here we introduce a modified version of QSP – the generalized QSP (GQSP): let

$$W(z) = \begin{bmatrix} z & 0 \\ 0 & 1 \end{bmatrix} \text{ and } R(\psi,\phi) = \begin{bmatrix} \cos\psi & \mathrm{e}^{\mathrm{i}\phi}\sin\psi \\ -\mathrm{e}^{-\mathrm{i}\phi}\sin\psi & \cos\psi \end{bmatrix} \tag{11}$$

be the *signal* and *control* unitaries. Given a target polynomial $f(z) \in \mathbb{C}[z]$, find the GQSP phase sequences $\{\phi_k\}_{k=0}^n$ and $\{\psi_k\}_{k=0}^n$ such that

$$R(\psi_0,\phi_0) \cdot \overrightarrow{\prod_{k=1}^{n}} W(x) R(\psi_k,\phi_k) = \begin{bmatrix} * & f(z) \\ * & * \end{bmatrix}. \tag{12}$$

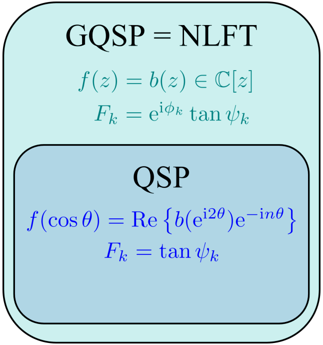
The $\overrightarrow{\prod}$ represents the ordered multiplication.

Generalized QSP

The GQSP allows more freedom to the control unitaries in comparison to the usual QSP [L25], allowing us to generate functions not restricted to being real / imaginary and odd / even.

$$\begin{bmatrix} \cos\psi & e^{i\phi}\sin\psi \\ -e^{-i\phi}\sin\psi & \cos\psi \end{bmatrix} = \frac{1}{\sqrt{1+|F|^2}} \begin{bmatrix} 1 & F \\ -\overline{F} & 1 \end{bmatrix}, \tag{13}$$

where $F = e^{i\phi}\tan\psi$.



GQSP = NLFT

$f(z) = b(z) \in \mathbb{C}[z]$

$F_k = e^{i\phi_k}\tan\psi_k$

QSP

$f(\cos\theta) = \mathrm{Re}\left\{b(e^{i2\theta})e^{-in\theta}\right\}$

$F_k = \tan\psi_k$

NLFT

Def. (Nonlinear Fourier Transform)

Let $F \in \ell^0(\mathbb{Z}, \mathbb{C})$ be a compactly supported sequence on $[m : n]$, its $\mathrm{SU}(2)$ *nonlinear Fourier transform* on $\mathbb{C} \cup \{\infty\}$ is defined as

$$\mathcal{F}(z) = (a, b) := \prod_{k=m}^{\overrightarrow{n}} \frac{1}{\sqrt{1 + |F_k|^2}} \begin{bmatrix} 1 & F_k z^k \\ -\overline{F}_k z^{-k} & 1 \end{bmatrix}. \tag{14}$$

Sometimes, we often see $\mathcal{F}$ be denoted by

$$\overbrace{F} \text{ or } \overbrace{[F_m, \dots, F_n]}.$$

But these are too space-consuming for this presentation, so we will not use them.

GQSP and NLFT

GQSP and NLFT are connected by the following substitution:

Thm. (GQSP $\equiv$ NLFT)

Consider $f \in \mathbb{C}[z]$. By setting

$$b(z) = f(z) \tag{15}$$

$$F_k = \mathrm{e}^{\mathrm{i}\phi_k} \tan \psi_k \quad (k \in [0:n]), \tag{16}$$

we have

$$\begin{bmatrix} * & f(z) \\ * & * \end{bmatrix} = \begin{bmatrix} a(z) & b(z) \\ -b^*(z) & a^*(z) \end{bmatrix} \begin{bmatrix} z^n & \\ & 1 \end{bmatrix}. \tag{17}$$

Tasks

Task (QSP):
Given an even or odd polynomial $f(x) \in \mathbb{R}[x]$ of degree n satisfying $\|f\|_\infty \leq 1$, find a sequence Ψ such that $f(x) = \mathrm{Im}\{\langle 0|V_\Psi(x)|0\rangle\}$.

↓

Task (inverse NLFT):
Given a Laurent polynomial $b(z)$, determine a complementary Laurent polynomial $a(z)$ and a compactly supported sequence of Fourier coefficients F such that $\mathcal{F} = (a, b)$.

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

- GQSP & NLFT
- Properties

3 Inverse Nonlinear Fourier Transform

4 Conclusion

Properties of NLFT – Shifting Property

Thm. (Laurent Polynomials)

If the sequence $F = (\dots, 0, F_m, F_{m+1}, \dots, F_n)$, then for $\mathcal{F} = (a(z), b(z))$,

$$a(z) = a_{m-n} z^{m-n} + \cdots + a_0 \qquad b(z) = b_m z^m + \cdots + b_n z^n.$$

Thm. (Shifting Property)

Consider the sequence $\{F_k\}$ having the NLFT $\mathcal{F} = (a(z), b(z))$. Then for the sequence $\{G_k\}$ satisfying $G_k = F_{k-1}$,

$$\mathcal{G} = (a(z), zb(z))\,. \tag{18}$$

From the above two results, we can set F to be *supported* on $[0:n]$ without loss of generality. Henceforth, both a^* and b are degree n polynomials in z.

Consider $\mathcal{F} = (a(z), b(z))$. Since for z on $\mathbb{T}$, $\mathcal{F} \in \mathrm{SU}(2)$, we have

$$|a(z)|^2 + |b(z)|^2 = 1.$$

Thus, for all $z \in \mathbb{C} \cup \{\infty\}$,

$$a(z)a^*(z) + b(z)b^*(z) = 1. \tag{19}$$

A simple example will be $F = (F_0, F_1, F_2) = (0, 1, 1)$:

$$\mathcal{F} = \frac{1}{2}\begin{bmatrix} 1 & z \\ -z^{-1} & 1 \end{bmatrix}\begin{bmatrix} 1 & z^2 \\ -z^{-2} & 1 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 - z^{-1} & z + z^2 \\ -(z^{-1} + z^{-2}) & 1 - z \end{bmatrix},$$

$$\Rightarrow aa^* + bb^* = \frac{1}{4}\left((-z + 1 + 1 - z^{-1}) + (z + 1 + 1 + z^{-1})\right) = 1. \qquad \square$$

Properties of NLFT

We note that

$$a^*(0) = \prod_{k=0}^{n} \frac{1}{\sqrt{1+|F_k|^2}} > 0.$$

Thus we introduce the following lemma (Lemma 2.2 in [NSYL25]):

Lem. (NLFT Bijection)

The NLFT is a bijection from $\ell_0(\mathbb{Z}, \mathbb{C})$ onto the space

$$\mathbf{S} := \{ (a,b) \mid a, b \text{ are Laurent polynomials, } aa^* + bb^* = 1,\ 0 < a^*(0) < \infty \} . \tag{20}$$

The polynomials a and b are *complementary* to each other. As a corollary, if a valid b is given, then there exists a unique outer a^* such that $(a,b) \in \mathbf{S}$. Constructing the polynomial a^* requires the *Weiss algorithm*.

Outer Functions

Def. (Outer Function)

An outer function is a function that has no zeros in $\mathbb{D}$.

Lem. (Outer Function)

A function $g \in L^{\infty}(\mathbb{T})$ is outer if

$$\log|g| \in L^1(\mathbb{T}),\ g = \mathrm{e}^{G},\ \text{and } G = \log|g| + \mathrm{i}\mathcal{H}(\log|g|). \tag{21}$$

The function $\mathcal{H}$ is the Hilbert transform:

$$\mathcal{H}(c) = 0,\ \mathcal{H}(z^n) = -\mathrm{i}z^n,\ \mathcal{H}(z^{-n}) = \mathrm{i}z^{-n}. \tag{22}$$

Properties of NLFT – Szegő Condition

Given a function b, can we find another function a such that (a, b) is the NLFT of a sequence?

The answer is positive! See theorem 4 in [ALMTW24]:

Lem. (Szegő Condition)

Consider a measurable function b on $\mathbb{T}$ with $\|b\|_\infty < 1$. If b satisfies the Szegő condition

$$\int_{\mathbb{T}} \log(1 - |b|^2)\, \mathrm{d}z > -\infty, \tag{23}$$

then there exists a *unique* measurable outer function a^* on $\mathbb{T}$ such that $(a, b) \in \mathbf{S}$.

This condition is natural: $|a(z)|^2 = 1 - |b(z)|^2$ is > 0 almost everywhere on $\mathbb{T}$.

Properties of NLFT – Riemann–Hilbert Factorization

We can decompose F into the two disjoint sequences $F_{<k} = (F_0, \ldots, F_{k-1}, 0, \ldots)$ and $F_{\geq k} = (0, \ldots, 0, F_k, \ldots, F_n)$. Then, the corresponding NLFT will have the factorization

$$\underbrace{\begin{bmatrix} a(z) & b(z) \\ -b^*(z) & a^*(z) \end{bmatrix}}_{\mathcal{F}} = \underbrace{\begin{bmatrix} a_{<k}(z) & b_{<k}(z) \\ -b^*_{<k}(z) & a^*_{<k}(z) \end{bmatrix}}_{\mathcal{F}_{<k}} \underbrace{\begin{bmatrix} a_{\geq k}(z) & b_{\geq k}(z) \\ -b^*_{\geq k}(z) & a^*_{\geq k}(z) \end{bmatrix}}_{\mathcal{F}_{\geq k}}. \tag{24}$$

This is the Riemann–Hilbert factorization of $\mathcal{F}$. The resulting polynomials are parameterized by

$$a^*_{\geq k}(z) = a_{k,0} + a_{k,1} z + \cdots + a_{k,n-k} z^{n-k}, \tag{25}$$

$$b_{\geq k}(z) = z^k \left(b_{k,0} + b_{k,1} z + \cdots + b_{k,n-k} z^{n-k} \right). \tag{26}$$

Properties of NLFT – Layer-Stripping

Consider the sequence $F = (F_0, F_1, F_2, \ldots)$ factorized into $F_{<1}$ and $F_{\geq 1}$. Then

$$\mathcal{F}_{\geq 1} = \begin{bmatrix} a_{\geq 1}(z) & b_{\geq 1}(z) \\ -b^*_{\geq 1}(z) & a^*_{\geq 1}(z) \end{bmatrix} = \frac{1}{\sqrt{1 + |F_0|^2}} \begin{bmatrix} 1 & -F_0 \\ \overline{F}_0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a(z) & b(z) \\ -b^*(z) & a^*(z) \end{bmatrix},$$

with $d(z)$ a polynomial having smallest power 1. Henceforth,

$$\begin{aligned} b_{\geq 1}(z) &\propto b(z) - F_0 a^*(z) \\ 0 = b_{\geq 1}(0) &= b(0) - F_0 a^*(0) \\ \Rightarrow F_0 &= \frac{b(0)}{a^*(0)} = \frac{b_{\geq 0}(0)}{a^*_{\geq 0}(0)} = \frac{b_{0,0}}{a_{0,0}}. \end{aligned}$$

Properties of NLFT – Layer-Stripping

Thm. (Layer-Stripping)

Consider $\mathcal{F} = (a, b)$ as the NLFT of $F = (F_0, F_1, \ldots, F_n)$ with RH factorization $\mathcal{F} = \mathcal{F}_{<k}\mathcal{F}_{\geq k}$, then

$$F_k = \left.\frac{b_{\geq k}(z)z^{-k}}{a^*_{\geq k}(z)}\right|_{z=0} = \frac{b_{k,0}}{a_{k,0}}. \tag{27}$$

Proof. Proof by combining the shifting property, $F_0 = b_{\geq 0}(0)/a^*_{\geq 0}(0)$, and the parameterization of the polynomials $a^*_{\geq k}$ and $b_{\geq k}$. ■

The layer-stripping method in NLFT is EXACTLY the SAME as the layer-stripping method developed in [T05; GSLW19].

Properties of NLFT – Riemann–Hilbert Factorization

Thm. (Riemann–Hilbert Factorization)

Let $P_{\mathbb{D}}$ be the projection of a Laurent polynomial onto its non-negative powers, and $P_{\mathbb{D}^*}$ the projection onto the non-positive powers:

$$\begin{bmatrix} 1 & P_{\mathbb{D}^*}\frac{b^*}{a^*} \\ -z^k P_{\mathbb{D}} z^{-k}\frac{b}{a} & 1 \end{bmatrix} \begin{bmatrix} a_{\geq k} \\ b_{\geq k} \end{bmatrix} \propto \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{28}$$

Proof. By observing the order to the polynomials, we have

$$(a_{<k}, b_{<k}) = (a, b)\left(a^*_{\geq k}, -b_{\geq k}\right) = \left(aa^*_{\geq k} + bb^*_{\geq k}, -ab_{\geq k} + a_{\geq k}b\right)$$

$$\Rightarrow \begin{cases} b_{\geq k} = \frac{ba_{\geq k}}{a} - \frac{b_{<k}}{a} \Rightarrow b_{\geq k} = z^k P_{\mathbb{D}}\left(z^{-k}\frac{b}{a}\right) a_{\geq k} \\ a^*_{\geq k} = \frac{1}{a_{\geq k}(0)} - P_{\mathbb{D}}\left(\frac{b}{a}\right) b^*_{\geq k} \end{cases}.$$

Thus it is shown, with the proportionality constant being $a^*_{\geq k}(0)$. ■

Table of Contents

1 Introduction

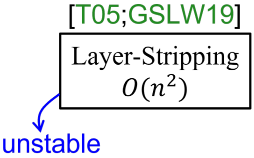
2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

▪ Riemann–Hilbert–Weiss Algorithm ▪ Half-Cholesky Method ▪ Layer-Stripping Revisited ▪ Inverse Nonlinear FFT

4 Conclusion

Roadmap

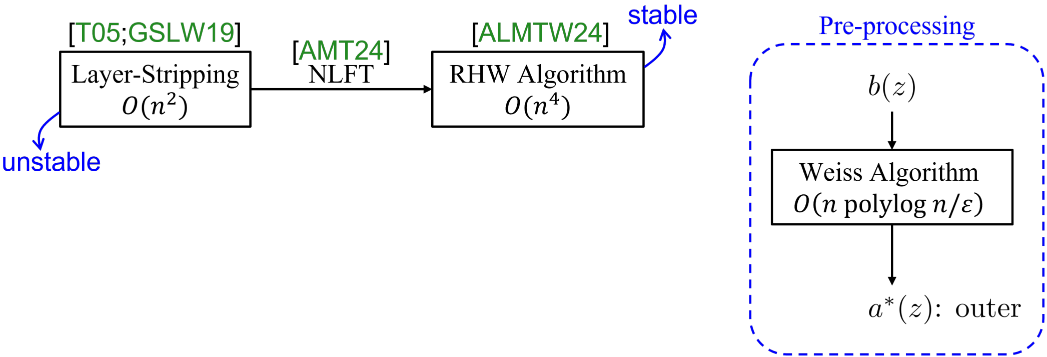


[T05;GSLW19]

Layer-Stripping
$O(n^2)$

unstable

Roadmap



[T05;GSLW19]

Layer-Stripping $O(n^2)$

unstable

[AMT24] NLFT

[ALMTW24]

RHW Algorithm $O(n^4)$

stable

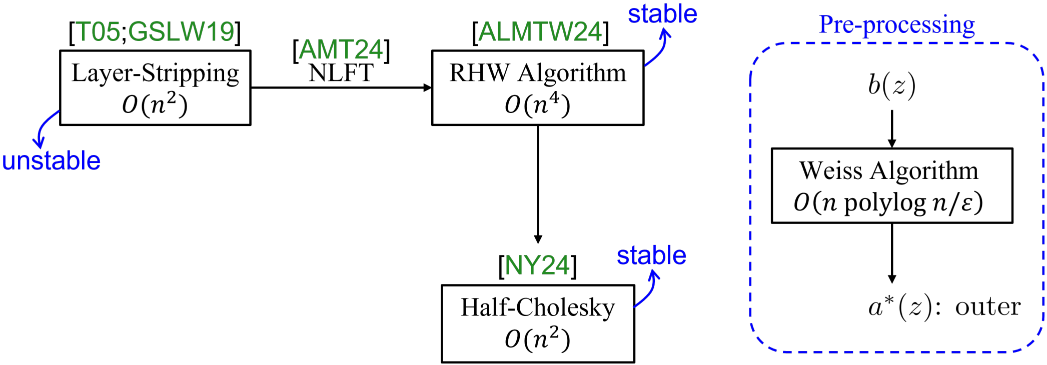
Pre-processing

$b(z)$

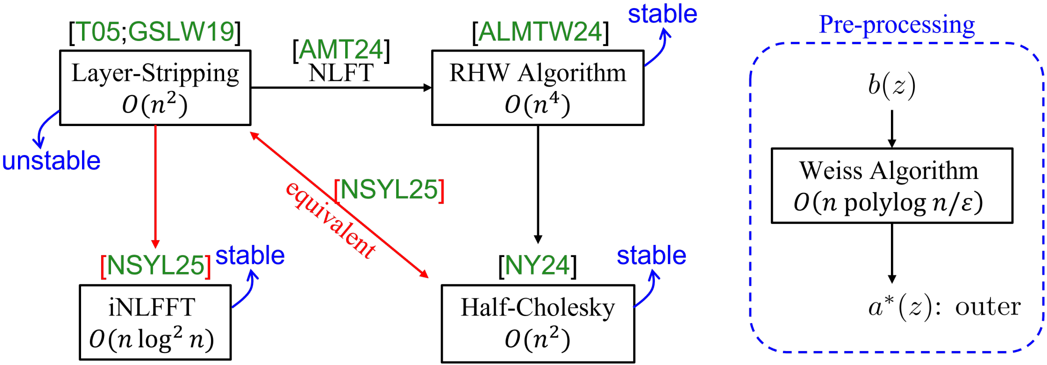
Weiss Algorithm $O(n \text{ polylog } n/\varepsilon)$

$a^*(z)$: outer

Roadmap


[T05;GSLW19]
Layer-Stripping
$O(n^2)$
unstable
[AMT24]
NLFT
[ALMTW24]
RHW Algorithm
$O(n^4)$
stable
[NY24]
Half-Cholesky
$O(n^2)$
stable
Pre-processing
$b(z)$
Weiss Algorithm
$O(n \text{ polylog } n/\varepsilon)$
$a^*(z)$: outer

Roadmap


[T05;GSLW19]
Layer-Stripping
$O(n^2)$
unstable
[AMT24]
NLFT
[ALMTW24]
stable
RHW Algorithm
$O(n^4)$
[NSYL25]
equivalent
[NSYL25]
stable
iNLFFT
$O(n\log^2 n)$
[NY24]
stable
Half-Cholesky
$O(n^2)$
Pre-processing
$b(z)$
Weiss Algorithm
$O(n \text{ polylog } n/\varepsilon)$
$a^*(z)$: outer

Roadmap

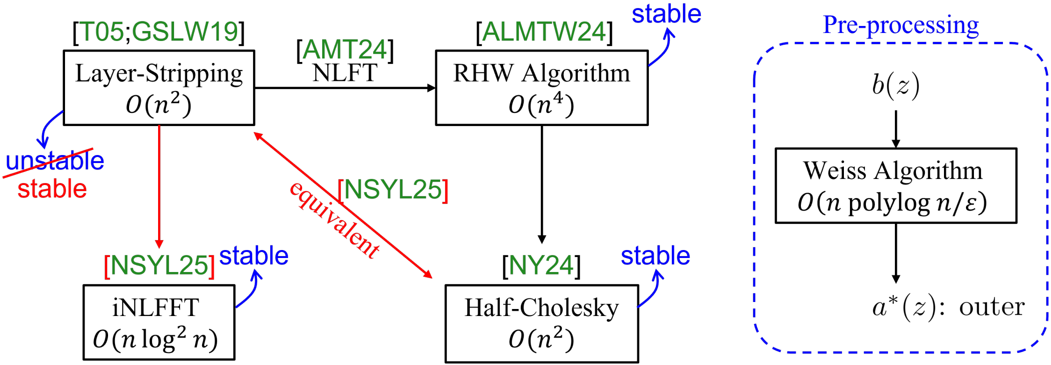

[T05;GSLW19]
Layer-Stripping
$O(n^2)$
~~unstable~~
stable
[AMT24]
NLFT
[ALMTW24]
stable
RHW Algorithm
$O(n^4)$
[NSYL25]
equivalent
[NSYL25]
stable
iNLFFT
$O(n \log^2 n)$
[NY24]
stable
Half-Cholesky
$O(n^2)$
Pre-processing
$b(z)$
Weiss Algorithm
$O(n \text{ polylog } n/\varepsilon)$
$a^*(z)$: outer

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

- Riemann–Hilbert–Weiss Algorithm
- Half-Cholesky Method
- Layer-Stripping Revisited
- Inverse Nonlinear FFT

4 Conclusion

The Scheme

This method was proposed in [ALMTW24] under QSP and later generalized to GQSP in [L25].

We have seen the layer-stripping method:

$$F_k = \left.\frac{b_{\geq k}(z) z^{-k}}{a^*_{\geq k}(z)}\right|_{z=0},$$

requiring the Riemann–Hilbert factorization

$$\begin{bmatrix} 1 & P_{\mathbb{D}^*} \frac{b^*}{a^*} \\ -z^k P_{\mathbb{D}} z^{-k} \frac{b}{a} & 1 \end{bmatrix} \begin{bmatrix} a_{\geq k} \\ b_{\geq k} \end{bmatrix} \propto \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{29}$$

The matrix in the front requires a series representation to $b(z)/a(z)$. This task is accomplished by the Weiss algorithm.

Weiss Algorithm

Given b that satisfies the *Szegő condition*, the Weiss algorithm constructs a unique outer a^* such that $(a, b) \in \mathbf{S}$.

Let $R(z) = \log \sqrt{1 - |b(z)|^2}$, $G = R + \mathrm{i}\mathcal{H}(R)$, then $a^* = \exp G$.

In numerical computation, however, we obtain the Laurent coefficients to a^* by $N = O(n \log(n/\varepsilon))$-point FFT on R and $\exp G$, finally truncating the obtained length-N sequence down to the coefficients representing z^0 to z^n.

The complexity of the Weiss algorithm is

$$O(N \log N) = O\left(n \operatorname{polylog}(n/\varepsilon)\right).$$

Riemann–Hilbert Factorization

Given $c(z) := b(z)/a(z) = \sum_{k=-\infty}^{n} c_k z^k$, we can rewrite the system of linear equations as

$$\begin{bmatrix} \mathbb{1}_{n-k+1} & -T_k^\dagger \\ T_k & \mathbb{1}_{n-k+1} \end{bmatrix} \begin{bmatrix} \boldsymbol{b}_k \\ \mathrm{rev}(\boldsymbol{a}_k^*) \end{bmatrix} \propto \mathrm{rev}(\boldsymbol{e}_0), \tag{30}$$

with vectors $\boldsymbol{e}_0 = (1, 0, \ldots, 0)^\mathsf{T}$, $\boldsymbol{a}_k^* = (a_{k,0}^*, \ldots, a_{k,n-k}^*)^\mathsf{T}$, $\boldsymbol{b}_k = (b_{k,0}, \ldots, b_{k,n-k})^\mathsf{T}$, the operator $\mathrm{rev}(\cdot)$ reversing the order of the vector, and the Toeplitz

$$T_k = \begin{bmatrix} c_n^* & & & \\ \vdots & \ddots & & \\ c_{k+1}^* & & \ddots & \\ c_k^* & c_{k+1}^* & \cdots & c_n^* \end{bmatrix}. \tag{31}$$

Riemann–Hilbert Factorization

Each phase term can be computed in parallel, requiring $O(n^3)$ for inverting the matrix! The total complexity will be $O(n^4)$. Costly! However, we can rewrite it: let $T := T_0$,

$$\underbrace{\left[\begin{array}{c:c} \mathbb{1}_{n+1} & -T^\dagger \\ \hdashline T & \mathbb{1}_{n+1} \end{array}\right]}_{A} \underbrace{\left[\begin{array}{c:cccc} & b_{n,0} & b_{n-1,0} & \cdots & b_{0,0} \\ \mathbb{1}_{n+1} & & b_{n-1,1} & & \vdots \\ & & & \ddots & b_{0,n} \\ \hdashline & a^*_{n,0} & a^*_{n-1,1} & \cdots & a^*_{0,n} \\ 0 & & a^*_{n-1,0} & & \vdots \\ & & & \ddots & a^*_{0,0} \end{array}\right]}_{S} = \underbrace{\left[\begin{array}{c:ccc} \mathbb{1}_{n+1} & & 0 & \\ \hdashline & * & & \\ T & \vdots & \ddots & \\ & * & \cdots & * \end{array}\right]}_{L}.$$

This is the LU decomposition of $A = LS^{-1}$. The complexity is down to $O(n^3)$.

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

■ Riemann–Hilbert–Weiss Algorithm ■ Half-Cholesky Method ■ Layer-Stripping Revisited ■ Inverse Nonlinear FFT

4 Conclusion

Half-Cholesky Method

The LU decomposition of A can be written as

$$\begin{bmatrix} \mathbb{1} & -T^\dagger \\ T & \mathbb{1} \end{bmatrix} = \begin{bmatrix} \mathbb{1} & \\ T & \mathbb{1} \end{bmatrix} \begin{bmatrix} \mathbb{1} & \\ & \mathbb{1} + TT^\dagger \end{bmatrix} \begin{bmatrix} \mathbb{1} & -T^\dagger \\ & \mathbb{1} \end{bmatrix}.$$

If we can obtain the Cholesky decomposition of $\mathbb{1} + TT^\dagger = LDL^\dagger$, then

$$A = \begin{bmatrix} \mathbb{1} & \\ T & L \end{bmatrix} \cdot \begin{bmatrix} \mathbb{1} & \\ & DL^\dagger \end{bmatrix} \begin{bmatrix} \mathbb{1} & -T^\dagger \\ & \mathbb{1} \end{bmatrix} \Rightarrow S = \begin{bmatrix} \mathbb{1} & T^\dagger (DL^\dagger)^{-1} \\ 0 & (DL^\dagger)^{-1} \end{bmatrix}. \tag{32}$$

The general time complexity required for Cholesky decomposition is $O(n^3)$.

Half-Cholesky Method

We don't need to do the whole Cholesky decomposition, however, as what we need is

$$\begin{bmatrix} a_{n,0} \\ \vdots \\ a_{0,0} \end{bmatrix} \propto D^{-1} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \text{ and } \begin{bmatrix} b_{n,0} \\ \vdots \\ b_{0,0} \end{bmatrix} \propto D^{-1} L^{-*} \boldsymbol{c}_0, \text{ with } \boldsymbol{c}_0 = \begin{bmatrix} c_n \\ \vdots \\ c_0 \end{bmatrix}, \tag{33}$$

as $\mathbb{1} + TT^\dagger \succeq 0$, D is real. The proportionality constant for the two equations above are the same. The Fourier coefficients will be

$$\begin{bmatrix} F_n \\ \vdots \\ F_0 \end{bmatrix} = \begin{bmatrix} b_{n,0}/a_{n,0} \\ \vdots \\ b_{0,0}/a_{0,0} \end{bmatrix} = L^{-*} \boldsymbol{c}_0. \tag{34}$$

Half-Cholesky Method

For the Cholesky decomposition of $\mathbb{1} + TT^\dagger$, we can do better than $O(n^3)$! Observe that T is a Toeplitz, it can be fully characterized by $n \times 2$ numbers. Hence, we can expect that the LDL decomposition of $\mathbb{1} + TT^\dagger$ should be faster!

Rmk. (Displacement)

If the *displacement rank* of an $n \times n$ matrix is r, then the Cholesky decomposition has complexity $O(n^2 r)$.

Hence, the final complexity for the half-Cholesky method is $O(n^2)$

$$Z := \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix} \tag{35}$$

For $K = \mathbb{1} + TT^\dagger$,

$$\Delta K := K - ZKZ^\mathsf{T} = \boldsymbol{e}_0 \boldsymbol{e}_0^\dagger + \boldsymbol{c}_0 \boldsymbol{c}_0^\dagger \tag{36}$$

$$= \begin{bmatrix} \boldsymbol{e}_0 & \boldsymbol{c}_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{e}_0^\dagger \\ \boldsymbol{c}_0^\dagger \end{bmatrix} = G_0 G_0^\dagger. \tag{37}$$

The displacement rank $r = 2$ [SK95].

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

- Riemann–Hilbert–Weiss Algorithm
- Half-Cholesky Method
- Layer-Stripping Revisited
- Inverse Nonlinear FFT

4 Conclusion

Layer-Stripping Revisited

Let us follow [NSYL25] and rewrite layer-stripping in a matrix notation:

$$(a_{\geq k}, b_{\geq k}) = \frac{1}{\sqrt{1+|F_k|^2}} \left(1, F_k z^k\right) (a_{\geq k+1}, b_{\geq k+1})$$

$$\Rightarrow (a_{\geq k+1}, b_{\geq k+1}) = \frac{1}{\sqrt{1+|F_k|^2}} \left(1, -F_k z^k\right) (a_{\geq k}, b_{\geq k})$$

Thus, we have

$$a^*_{\geq k+1} = \frac{1}{\sqrt{1+|F_k|^2}} \left(a^*_{\geq k} + \overline{F}_k z^{-k} b_{\geq k}\right), \tag{38}$$

$$z^{-(k+1)} b_{\geq k+1} = \frac{1}{z\sqrt{1+|F_k|^2}} \left(-F_k a^*_{\geq k} + z^{-k} b_{\geq k}\right). \tag{39}$$

Layer-Stripping Revisited

Through the parameterization $a^*_{\geq k} = \sum_{j=0}^{n-k} a_{k,j} z^j$ and $z^{-k} b_{\geq k} = \sum_{j=0}^{n-k} b_{k,j} z^j$, we have

$$\begin{bmatrix} a_{k+1,0} & 0 \\ a_{k+1,1} & b_{k+1,0} \\ \vdots & \vdots \\ a_{k+1,n-k-1} & b_{k+1,n-k-2} \\ 0 & b_{k+1,n-k-1} \end{bmatrix} = \underbrace{\begin{bmatrix} a_{k,0} & b_{k,0} \\ a_{k,1} & b_{k,1} \\ \vdots & \vdots \\ a_{k,n-k-1} & b_{k,n-k-1} \\ a_{k,n-k} & b_{k,n-k} \end{bmatrix}}_{G_k = [\boldsymbol{a}_k, \boldsymbol{b}_k]} \frac{1}{\sqrt{1+|F_k|^2}} \begin{bmatrix} 1 & -F_k \\ \overline{F}_k & 1 \end{bmatrix} \quad (40)$$

The phase $F_k = b_{k,0}/a_{k,0}$ is obtained by the QR decomposition of G_k, requiring $O(n)$ complexity each. The total complexity will be $O(n^2)$.

Layer-Stripping Revisited

However, through induction, we see that for any $f(z)$, the following sequence has the same layer-stripped Fourier coefficients:

$$(a(z)/f(z), b(z)/f(z))\,.$$

The layer-stripping method is closely related to the half-Cholesky method previously mentioned when setting $f = a$, then the half-Cholesky method is just applying layer-stripping on

$$(1, b(z)/a(z)) =: (1, c(z))\,.$$

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

- Riemann–Hilbert–Weiss Algorithm
- Half-Cholesky Method
- Layer-Stripping Revisited
- Inverse Nonlinear FFT

4 Conclusion

This is the state-of-the-art proposed by [NSYL25] on May 19th, 2025. It essentially reduces the redundancy in computation of the layer-stripping method (or equivalently, the half-Cholesky method).

Though inspired by [AG89], it's essence is still, nevertheless, layer-stripping:

$$\begin{bmatrix} a_{k+1,0} & 0 \\ a_{k+1,1} & b_{k+1,0} \\ \vdots & \vdots \\ a_{k+1,n-k-1} & b_{k+1,n-k-2} \\ 0 & b_{k+1,n-k-1} \end{bmatrix} = \underbrace{\begin{bmatrix} a_{k,0} & b_{k,0} \\ a_{k,1} & b_{k,1} \\ \vdots & \vdots \\ a_{k,n-k-1} & b_{k,n-k-1} \\ a_{k,n-k} & b_{k,n-k} \end{bmatrix}}_{G_k=[\boldsymbol{a}_k,\boldsymbol{b}_k]} \frac{1}{\sqrt{1+|F_k|^2}} \begin{bmatrix} 1 & -F_k \\ \overline{F}_k & 1 \end{bmatrix}. \tag{41}$$

Inverse Nonlinear FFT

The previous algorithm (layer-stripping, or equivalently, half-Cholesky) follows:

$$G_0 = [\boldsymbol{a}_0, \boldsymbol{b}_0] \longmapsto G_1 = [\boldsymbol{a}_1, \boldsymbol{b}_1] \longmapsto \cdots \longmapsto G_{\lceil n/2 \rceil} \longmapsto \cdots \longmapsto G_n,$$

where each arrow represents a QR decomposition and shifting.

Note that the calculation to $(F_{\lceil n/2 \rceil}, \dots, F_n)$ fully depends on $G_{\lceil n/2 \rceil}$, with $G_{\lceil n/2 \rceil}$ depending on $(F_0, \dots, F_{\lceil n/2 \rceil - 1})$.

Similarly, we can continue this partitioning process.

Inverse Nonlinear FFT

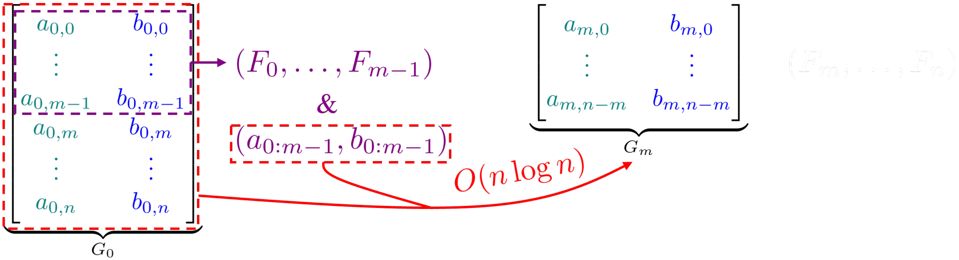
$$\underbrace{\begin{bmatrix} a_{0,0} & b_{0,0} \\ \vdots & \vdots \\ a_{0,m-1} & b_{0,m-1} \\ a_{0,m} & b_{0,m} \\ \vdots & \vdots \\ a_{0,n} & b_{0,n} \end{bmatrix}}_{G_0} \rightarrow (F_0, \ldots, F_{m-1})$$

$$\&$$

$$(a_{0:m-1}, b_{0:m-1})$$

$$(F_m, \ldots, F_n)$$

Inverse Nonlinear FFT



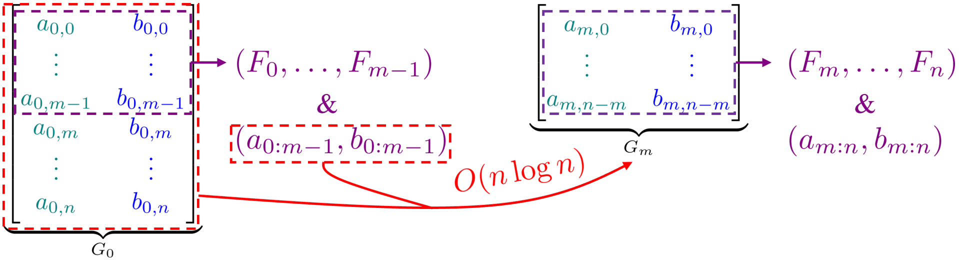
$a_{0,0}$ $b_{0,0}$ $\vdots$ $\vdots$ $a_{0,m-1}$ $b_{0,m-1}$ $a_{0,m}$ $b_{0,m}$ $\vdots$ $\vdots$ $a_{0,n}$ $b_{0,n}$
G_0
$(F_0, \ldots, F_{m-1})$
&
$(a_{0:m-1}, b_{0:m-1})$
$O(n \log n)$
$a_{m,0}$ $b_{m,0}$ $\vdots$ $\vdots$ $a_{m,n-m}$ $b_{m,n-m}$
G_m
$(F_m, \ldots, F_n)$

The time complexity required for calculating

$$G_m = (a_{m:n}, b_{m:n}) = (a_{0:m-1}, b_{0:m-1})^{-1}(a_{0:n}, b_{0:n}) = (a_{0:m-1}, b_{0:m-1})^{-1}G_0,$$

where $m = \lceil n/2 \rceil$, is $O(n \log n)$.

Inverse Nonlinear FFT



$$G_0 = \begin{bmatrix} a_{0,0} & b_{0,0} \\ \vdots & \vdots \\ a_{0,m-1} & b_{0,m-1} \\ a_{0,m} & b_{0,m} \\ \vdots & \vdots \\ a_{0,n} & b_{0,n} \end{bmatrix} \to (F_0, \ldots, F_{m-1}) \ \& \ (a_{0:m-1}, b_{0:m-1})$$

$$O(n \log n)$$

$$G_m = \begin{bmatrix} a_{m,0} & b_{m,0} \\ \vdots & \vdots \\ a_{m,n-m} & b_{m,n-m} \end{bmatrix} \to (F_m, \ldots, F_n) \ \& \ (a_{m:n}, b_{m:n})$$

The time complexity required for calculating

$$G_m = (a_{m:n}, b_{m:n}) = (a_{0:m-1}, b_{0:m-1})^{-1}(a_{0:n}, b_{0:n}) = (a_{0:m-1}, b_{0:m-1})^{-1} G_0,$$

where $m = \lceil n/2 \rceil$, is $O(n \log n)$. The total time complexity is

$$n \log n + 2\left(\frac{n}{2} \log \frac{n}{2}\right) + 4\left(\frac{n}{4} \log \frac{n}{4}\right) + \cdots = O(n \log^2 n). \qquad (44)$$

Table of Contents

1 Introduction

2 Nonlinear Fourier Analysis

3 Inverse Nonlinear Fourier Transform

4 Conclusion

Conclusion

Thm. (GQSP $\equiv$ NLFT)

Given a degree n polynomial $b \in \mathbb{C}[z]$ that satisfies the Szegő condition

$$\int_{\mathbb{T}} \log(1 - |b(z)|^2)\,\mathrm{d}z > -\infty, \tag{45}$$

we can construct a unique outer complementary polynomial a using the Weiss algorithm.

Then, by the iNLFFT algorithm, we can retrieve the Fourier coefficients $F : [0, n]$ to error within ε such that $\mathcal{F} = (a, b)$ with time complexity

$$O(n \log^2 n + n \operatorname{polylog}(n/\varepsilon)).$$

Lastly, by a simple change of variable $F_k = \mathrm{e}^{\mathrm{i}\phi_k} \tan \psi_k$, we can transform the NLFT into a GQSP protocol.

Stability

It is shown in [NSYL25], that under the constraint that

$$a^* \text{ has no zeros in } \overline{\mathbb{D}}, \tag{46}$$

all the algorithms satisfies numerical stability through the *backward error estimation*.

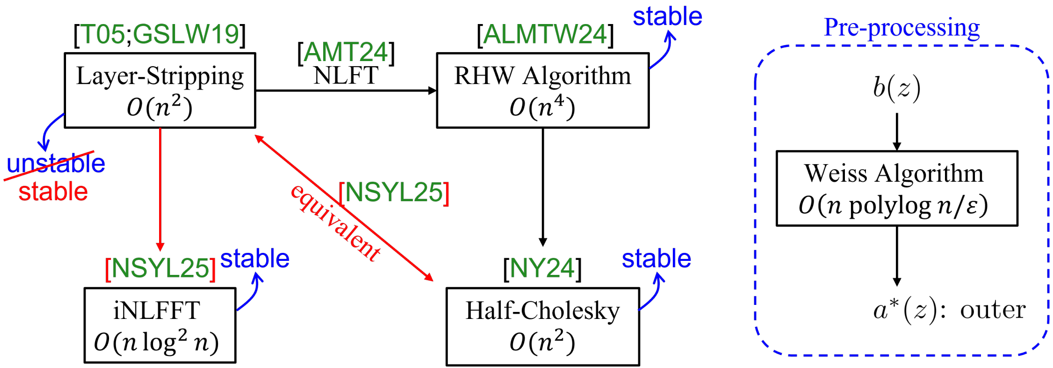
- Forward error measure:

  The deviation of the computed $\hat{x}$ to the exact solution x.

- Backward error measure:

  The smallest perturbation to the input that would make $\hat{x}$ an exact solution.

Roadmap


[T05;GSLW19]
Layer-Stripping
$O(n^2)$
[AMT24]
NLFT
[ALMTW24]
RHW Algorithm
$O(n^4)$
stable
Pre-processing
$b(z)$
Weiss Algorithm
$O(n \text{ polylog } n/\varepsilon)$
$a^*(z)$: outer
~~unstable~~
stable
[NSYL25]
equivalent
[NSYL25]
stable
iNLFFT
$O(n \log^2 n)$
[NY24]
stable
Half-Cholesky
$O(n^2)$

Future Directions

It is shown in [ALMTW24], [L25], and [NSYL25] that the above methodology still holds in the case of infinite quantum signal processing, where the length of the phase sequence goes to infinity.

Some future directions suggested includes:

- Is there a stable algorithm for *non-outer* a^*?
- A stable algorithm for fully-coherent case: $\|b\|_\infty = 1$.
- Are all the behaviors of QSP in [WDL22] explained? Especially from the point of view of NLFT.
- M-QSP [RC22]: Is there any better choice of unitaries to span the whole space of multivariate functions?
- Is there a *theoretical lower bound* to the complexity and bits required? Up to constants?
- ...

References I

[T05] Ya-Ju Tsai. “SU(2) nonlinear fourier transform”. PhD thesis. UCLA, 2005. URL: https://www.proquest.com/dissertations-theses/i-su-2-nonlinear-fourier-transform/docview/305033060/se-2.

[GSLW19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. “Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. STOC '19. ACM, June 2019, pp. 193–204. DOI: 10.1145/3313276.3316366. URL: http://dx.doi.org/10.1145/3313276.3316366.

[AMT24] Michel Alexis, Gevorg Mnatsakanyan, and Christoph Thiele. *Quantum signal processing and nonlinear Fourier analysis*. 2024. arXiv: 2310.12683 [quant-ph]. URL: https://arxiv.org/abs/2310.12683.

References II

[ALMTW24] Michel Alexis, Lin Lin, Gevorg Mnatsakanyan, Christoph Thiele, and Jiasu Wang. *Infinite quantum signal processing for arbitrary Szegő functions*. 2024. arXiv: 2407.05634 [quant-ph]. URL: https://arxiv.org/abs/2407.05634.

[NY24] Hongkang Ni and Lexing Ying. *Fast Phase Factor Finding for Quantum Signal Processing*. 2024. arXiv: 2410.06409 [quant-ph]. URL: https://arxiv.org/abs/2410.06409.

[NSYL25] Hongkang Ni, Rahul Sarkar, Lexing Ying, and Lin Lin. *Inverse nonlinear fast Fourier transform on SU(2) with applications to quantum signal processing*. 2025. arXiv: 2505.12615 [quant-ph]. URL: https://arxiv.org/abs/2505.12615.

[H19] Jeongwan Haah. “Product Decomposition of Periodic Functions in Quantum Signal Processing”. In: *Quantum* 3 (Oct. 2019), p. 190. ISSN: 2521-327X. DOI: 10.22331/q-2019-10-07-190. URL: http://dx.doi.org/10.22331/q-2019-10-07-190.

References III

[Y22] Lexing Ying. "Stable factorization for phase factors of quantum signal processing". In: *Quantum* 6 (Oct. 2022), p. 842. ISSN: 2521-327X. DOI: 10.22331/q-2022-10-20-842. URL: http://dx.doi.org/10.22331/q-2022-10-20-842.

[WDL22] Jiasu Wang, Yulong Dong, and Lin Lin. "On the energy landscape of symmetric quantum signal processing". In: *Quantum* 6 (Nov. 2022), p. 850. ISSN: 2521-327X. DOI: 10.22331/q-2022-11-03-850. URL: http://dx.doi.org/10.22331/q-2022-11-03-850.

[DLNW24a] Yulong Dong, Lin Lin, Hongkang Ni, and Jiasu Wang. "Infinite quantum signal processing". In: *Quantum* 8 (Dec. 2024), p. 1558. ISSN: 2521-327X. DOI: 10.22331/q-2024-12-10-1558. URL: https://doi.org/10.22331/q-2024-12-10-1558.

References IV

[DLNW24b] Yulong Dong, Lin Lin, Hongkang Ni, and Jiasu Wang. “Robust Iterative Method for Symmetric Quantum Signal Processing in All Parameter Regimes”. In: *SIAM Journal on Scientific Computing* 46.5 (2024), A2951–A2971. DOI: 10.1137/23M1598192. URL: https://doi.org/10.1137/23M1598192.

[L25] Lorenzo Laneve. *Generalized Quantum Signal Processing and Non-Linear Fourier Transform are equivalent*. 2025. arXiv: 2503.03026 [quant-ph]. URL: https://arxiv.org/abs/2503.03026.

[SK95] Ali H. Sayed and Thomas Kailath. “Fast algorithms for generalized displacement structures and lossless systems”. In: *Linear Algebra and its Applications* 219 (1995), pp. 49–78. ISSN: 0024-3795. DOI: https://doi.org/10.1016/0024-3795(93)00193-4. URL: https://www.sciencedirect.com/science/article/pii/0024379593001934.

References V

[AG89] Gregory S. Ammar and William B. Gragg. "Numerical experience with a superfast real Toeplitz solver". In: *Linear Algebra and its Applications* 121 (1989), pp. 185–206. ISSN: 0024-3795. DOI: https://doi.org/10.1016/0024-3795(89)90701-5. URL: https://www.sciencedirect.com/science/article/pii/0024379589907015.

[RC22] Zane M. Rossi and Isaac L. Chuang. "Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle". In: *Quantum* 6 (Sept. 2022), p. 811. ISSN: 2521-327X. DOI: 10.22331/q-2022-09-20-811. URL: http://dx.doi.org/10.22331/q-2022-09-20-811.