# A SPARSE SPECTRAL METHOD FOR VOLTERRA INTEGRAL EQUATIONS USING ORTHOGONAL POLYNOMIALS ON THE TRIANGLE*

TIMON S. GUTLEB† AND SHEEHAN OLVER ‡

**Abstract.** We introduce and analyze a sparse spectral method for the solution of Volterra integral equations using bivariate orthogonal polynomials on a triangle domain. The sparsity of the Volterra operator on a weighted Jacobi basis is used to achieve high efficiency and exponential convergence. The discussion is followed by a demonstration of the method on example Volterra integral equations of the first and second kind with or without known analytic solutions as well as an application-oriented numerical experiment. We prove convergence for both first and second kind problems, where the former builds on connections with Toeplitz operators.

**Key words.** Volterra integral equations, spectral methods, sparse operators, orthogonal polynomials

**AMS subject classifications.** 65N35, 45D05

## 1. Introduction.

Define the *Volterra integral operator*

$$(\mathcal{V}_K u)(x) := \int_0^{l(x)} K(x,y)u(y)\mathrm{d}y, \tag{1.1}$$

where $K(x,y)$ is called the kernel, $u(y)$ is a given function of one variable and the limits of integration are either $l(x) = x$ or $l(x) = 1-x$. This paper concerns Volterra integral equations of the first and second kind, that is, to find $u$ satisfying

$$\mathcal{V}_K u = g \qquad \text{or} \qquad (\lambda I + \mathcal{V}_K)u = g.$$

Numerous applications and the fundamental nature of Volterra integral and integro-differential equations motivate research into efficient and accurate numerical solvers. Various forms of Volterra integral equations are analytically well-understood [13, 41, 51], have been the subject of various numerical approximation schemes [13, 12, 5, 33], and are encountered regularly in various scientific fields as well as engineering and finance applications [13, 41, 49, 51, 28, 29].

In this paper we present a method to compute Volterra integrals and solve Volterra integral equations by using orthogonal polynomials on a triangle domain [22, 39] to both resolve the kernel and to reduce the equations to banded linear systems. The method is in the same spirit as some previous contributions to the field of numerical Volterra, Fredholm, singular integral and differential equations based on operators and orthogonal polynomials such as [1, 27, 45, 26] but differs in choice of basis and domain, leading to operator bandedness properties which can be exploited for significantly increased efficiency. Notably the approach introduced in this paper can be used for a wider range of kernels than many other Volterra integral equation solvers such as the methods based on orthogonal polynomials due to Loureiro and Xu [32, 54], the recently developed ultraspherical spectral method in [26] or the Fourier extension method in [53] as it is not limited to convolution kernel cases, that is kernels of the

*Submitted to the editors 10/06/2019. Revised version submitted 04/02/2020.

†Department of Mathematics, Imperial College London, UK (t.gutleb18@imperial.ac.uk) and Fakultät für Mathematik, University of Vienna, Austria.

‡Department of Mathematics, Imperial College London, UK (s.olver@imperial.ac.uk).

form $K(x,y) = K(x-y)$, but works for a wider class of kernels. We prove the convergence of the proposed sparse spectral method for second kind Volterra integral equations with general kernels which are sufficiently smooth to be approximated by Jacobi polynomials as well as for first kind Volterra integral equations with sufficiently smooth kernels where $\forall x \in [0,1] : K(x,x) \neq 0$.

The sections in this paper are organized as follows: Section 2 introduces the required aspects of univariate and bivariate polynomial function approximation on a real interval and the triangle respectively. Section 3 introduces an efficient numerical method for Volterra integrals and integral equations and discusses how to approach kernel computations using a multivariate variant of Clenshaw's algorithm. In Section 4 we show the scheme in action in both toy and application-based examples. Proofs of convergence for well-posed problems are discussed in Section 5.

## 2. Function approximation with orthogonal polynomials.

**2.1. Jacobi polynomials on the real interval.** Multivariate orthogonal polynomials are ordered sets of polynomials satisfying a particular pair-wise and weighted orthogonality condition, often of the form

$$\langle P_{m,k}, P_{n,j} \rangle = \int_\Omega P_{m,k}(\mathbf{x}) P_{n,j}(\mathbf{x}) W(\mathbf{x}) \mathrm{d}A = C\delta_{mn}\delta_{jk}, \tag{2.1}$$

where $C \neq 0$ and $P_{m,k}$ are total degree $m$ polynomials. Many such sets of orthogonal polynomials are well-known and well-studied on various domains $\Omega$ such as $\mathbb{R}$, real intervals, simple 2D and 3D domains, as well as various higher dimensional spheres and polygons [22]. The relevant set of orthogonal polynomials for this paper are the Jacobi polynomials on the real line and on the triangle respectively. This section will thus give a quick overview of Jacobi polynomials aimed at equipping us with the tools needed to develop the Volterra integral equation solvers in later sections. We refer to [22, 23] for introductions with broader scope.

The Jacobi polynomials are orthogonal on $[-1,1]$:

$$\int_{-1}^{1} C_{(\alpha,\beta,m,n)} (1-x)^\alpha (1+x)^\beta P_m^{(\alpha,\beta)}(x) P_n^{(\alpha,\beta)}(x) \mathrm{d}x = \delta_{nm},$$

where $W_{(\alpha,\beta)}(x) = C_{(\alpha,\beta,m,n)} (1-x)^\alpha (1+x)^\beta$ acts as the weight function and $\delta_{nm}$ is the Kronecker delta. While the choice of $[-1,1]$ is natural, the Jacobi polynomials can be shifted to any real interval an application requires. For $\alpha = \beta = 0$ the Jacobi polynomials reduce to the Legendre polynomials [22].

One of the primary applications of interest for the study of orthogonal polynomials are their applications in the expansion of non-polynomial functions:

$$f(x) = \sum_{n=0}^{\infty} p_n(x) f_n = \mathbf{P}(x)^\mathsf{T} \mathbf{f},$$

where $f_n$ is the function-specific coefficient of the $n$-th polynomial $p_n$ and we use the notation

$$\mathbf{P}(x) := \begin{pmatrix} p_0(x) \\ p_1(x) \\ \vdots \end{pmatrix}, \qquad \mathbf{f} := \begin{pmatrix} f_0 \\ f_1 \\ \vdots \end{pmatrix}.$$

For numerical applications one uses finitely many terms in the above sum to obtain an approximation. If a distinction between different sets of polynomials and coefficient vectors on different domains is required we specify by indicating the type of polynomials using standard notation for the polynomials, such as $\mathbf{P}^{(\alpha,\beta)}(x)$ for the Jacobi polynomials on a real interval, and the domain using index notation, e.g. for the bivariate orthogonal polynomial coefficient vector of $g(x,y)$ on the triangle domain we write $\mathbf{g}_{\triangle}$.

To use function approximation of this type in a non-trivial numerical application one needs ways to do computations on functions represented as coefficient vectors. Basic computations such as addition and subtraction of functions have obvious element-wise implementations. Furthermore one can compute $xf(x)$ if $f(x)$ is already approximated as a coefficient vector: to do this one uses multiplication operators $\bar{\mathrm{X}}$ which act as

$$\mathbf{P}(x)^{\mathsf{T}}\bar{\mathrm{X}}\mathbf{f}_{[0,1]} = xf(x).$$

This is efficiently possible because the Jacobi polynomials satisfy a three-term recurrence relationship, making $\bar{\mathrm{X}}$ a tridiagonal operator, in fact it is the transpose of the Jacobi operator associated with $p_n$:

$$J = \bar{\mathrm{X}}^{\mathsf{T}} = \begin{pmatrix} a_0 & b_0 & & \\ c_0 & a_1 & b_1 & \\ & c_1 & a_2 & \ddots \\ & & \ddots & \ddots \end{pmatrix}. \tag{2.2}$$

Additionally, our approach to Volterra integral equations of the second kind will require explicit constructors for raising operators $\mathrm{S}^{(\alpha+1,\beta)}_{(\alpha,\beta)}, \mathrm{S}^{(\alpha,\beta+1)}_{(\alpha,\beta)}$ which are defined to increment from the Jacobi bases $\mathbf{P}^{(\alpha,\beta)}(x)$ to $\mathbf{P}^{(\alpha+1,\beta)}(x)$ and $\mathbf{P}^{(\alpha,\beta+1)}(x)$ respectively. Increments to $\alpha$ and $\beta$ can be computed using these operators but decrementing is generally only well-defined in the sense of *weighted* lowering operators:

$$xf(x) = \mathbf{P}^{(\alpha-1,\beta)}(x)^{\mathsf{T}}\mathrm{L}^{(\alpha-1,\beta)}_{(\alpha,\beta)}\mathbf{f},$$
$$(1-x)f(x) = \mathbf{P}^{(\alpha,\beta-1)}(x)^{\mathsf{T}}\mathrm{L}^{(\alpha,\beta-1)}_{(\alpha,\beta)}\mathbf{f}.$$

The explicit forms of the operators $\bar{\mathrm{X}}$, $\mathrm{S}^{(\alpha+1,\beta)}_{(\alpha,\beta)}$, $\mathrm{S}^{(\alpha,\beta+1)}_{(\alpha,\beta)}$, $\mathrm{L}^{(\alpha-1,\beta)}_{(\alpha,\beta)}$ and $\mathrm{L}^{(\alpha,\beta-1)}_{(\alpha,\beta)}$ are well known in the literature, see for example [35, 39, 22] and the references therein.

**2.2. Jacobi polynomials on the triangle.** We now briefly discuss how function approximation using bivariate orthogonal polynomials works in general and then move on to discuss the Jacobi polynomials on the canonical unit simplex

$$T^2 = \{(x,y) : 0 \leqslant x, 0 \leqslant y \leqslant 1-x\}.$$

We use a basis on this triangle in the following sections to compute Volterra integrals and solve integral equations. As in the univariate case, bivariate orthogonal polynomials are said to be orthogonal with respect to an inner product akin to (2.1).

Analogously to how functions of a single variable may be expanded into a basis of univariate orthogonal polynomials as $f(x) = \sum_{n=0}^{\infty} p_n(x) f_n$ we can expand a function

of two variables in a basis of bivariate polynomials as

$$f(x,y) = \sum_{n=0}^{\infty} \sum_{k=0}^{n} p_{n,k}(x,y) f_{n,k}.$$

Writing the bivariate polynomials of total degree $n$ as

$$\mathbb{P}_n(x,y) = \begin{pmatrix} p_{n,0}(x,y) \\ p_{n,1}(x,y) \\ \vdots \\ p_{n,n}(x,y) \end{pmatrix}$$

allows for the following compact notation for the infinite-dimensional polynomial basis:

$$\mathbf{P}(x,y) = \begin{pmatrix} \mathbb{P}_0(x,y) \\ \mathbb{P}_1(x,y) \\ \vdots \end{pmatrix}.$$

In this notation the expansion of a function of two variables in the bivariate polynomial basis becomes

$$f(x,y) = \sum_{n=0}^{\infty} \sum_{k=0}^{n} p_{n,k}(x,y) f_{n,k} = \mathbf{P}(x,y)^{\mathsf{T}} \mathbf{f}.$$

For function approximation one simply uses an appropriate finite cutoff of this expansion.

On the triangle $T^2$ we focus on the Jacobi weights $x^\alpha y^\beta (1-x-y)^\gamma$. One elegant way to define the corresponding Jacobi polynomials $\mathbf{P}^{(\alpha,\beta,\gamma)}(x,y)$ on the canonical triangle $T^2$ is by referring to the Jacobi polynomials $\mathbf{P}^{(\alpha,\beta)}(x)$ on the real interval $[-1,1]$ (compare [22, Proposition 2.4.1]):

$$P_{k,n}^{(\alpha,\beta,\gamma)}(x,y) = (1-x)^k \, P_{n-k}^{(2k+\beta+\gamma+1,\alpha)}(2x-1) \, P_k^{(\gamma,\beta)}\left(\frac{2y}{1-x} - 1\right). \tag{2.3}$$

Defined as such the triangle Jacobi polynomials are orthogonal with respect to a weighted integral over the canonical triangle domain $T^2$:

$$\int_0^1 \int_0^{1-x} x^\alpha y^\beta (1-x-y)^\gamma P_{k,n}^{(\alpha,\beta,\gamma)}(x,y) P_{j,m}^{(\alpha,\beta,\gamma)}(x,y) \mathrm{d}y \mathrm{d}x = C_{(\alpha,\beta,\gamma)} \delta_{jk} \delta_{mn}.$$

The detailed form of the constant $C_{(\alpha,\beta,\gamma)}$ is not important here but can for example be found in [22]. We will primarily use the Jacobi polynomials shifted to the $[0,1]$ interval and denote them by $\tilde{\mathbf{P}}^{(\alpha,\beta)}(x)$, which allows us to write the Jacobi polynomials on the triangle as:

$$P_{k,n}^{(\alpha,\beta,\gamma)}(x,y) = (1-x)^k \tilde{P}_{n-k}^{(2k+\beta+\gamma+1,\alpha)}(x) \, \tilde{P}_k^{(\gamma,\beta)}\left(\frac{y}{1-x}\right). \tag{2.4}$$

As in the 1-dimensional case we can define multiplication operators X and Y, one for each variable, which respectively act as

$$\mathbf{P}(x,y)^{\mathsf{T}} \mathrm{X} \mathbf{f}_\triangle = x f(x,y),$$
$$\mathbf{P}(x,y)^{\mathsf{T}} \mathrm{Y} \mathbf{f}_\triangle = y f(x,y),$$

for a given bivariate polynomial basis. Unlike the 1-dimensional Jacobi polynomial case these operators are not tridiagonal but block tridiagonal Jacobi operators for the triangle Jacobi polynomials [39]:

$$\mathrm{X}^\mathsf{T} = \begin{pmatrix} A_0^x & B_0^x & & \\ C_0^x & A_1^x & B_1^x & \\ & C_1^x & A_2^x & \ddots \\ & & \ddots & \ddots \end{pmatrix}, \quad \mathrm{Y}^\mathsf{T} = \begin{pmatrix} A_0^y & B_0^y & & \\ C_0^y & A_1^y & B_1^y & \\ & C_1^y & A_2^y & \ddots \\ & & \ddots & \ddots \end{pmatrix}, \tag{2.5}$$

where $A_n^x, A_n^y \in \mathbb{R}^{(n+1)\times(n+1)}$, $B_n^x, B_n^y \in \mathbb{R}^{(n+1)\times(n+2)}$ and $C_n^x, C_n^y \in \mathbb{R}^{(n+2)\times(n+1)}$. Analogous operators to the raising and lowering operators discussed for the real interval case can be constructed for the Jacobi polynomials on the triangle as well, see [38, 39], but we omit their discussion as we will not make direct use of them in this paper.

To make use of Jacobi polynomials for the approximation of functions on the triangle domain in a numerical context one requires efficient algorithms to determine the coefficient vector $\mathbf{f}_\triangle$ for a given function $f(x, y)$ of two variables. This can be done using an algorithm and its implementation in a C library by Slevinsky [42, 43, 44].

**2.3. Function evaluation using Clenshaw's algorithm.** Clenshaw's algorithm provides an efficient and direct method to evaluate functions expanded into orthogonal polynomial bases at given points, i.e. to evaluate $\sum_{n=0}^{N} p_n(\mathbf{x}) f_n$ at $\mathbf{x}_* \in \mathbb{R}^d$, cf. [16, 39]. The algorithm makes use of the polynomial basis' recurrence relationships to reduce function evaluation to the solution of an upper triangular linear system using backward substitution. In this section we give an outline of how this is done for Jacobi polynomials on the real interval and the triangle, which is discussed in more detail in [39]. An operator valued variant of what is discussed in this section will be used for efficient kernel computations for Volterra integrals in section 3.2. We mention a major benefit of Clenshaw's algorithm over building polynomials/operators via forward recurrences is that there is substantially less memory needed in the intermediary calculations.

For the case of Jacobi polynomials on a real interval, the three-term recurrence relationship seen in the Jacobi operator in (2.2) can be used to write

$$\mathcal{L}_N(x_*)\mathbf{P}_N^{(\alpha,\beta)}(x_*) = \mathbf{e}_0,$$

$$\mathcal{L}_N(x_*) = \begin{pmatrix} 1 & & & & & \\ a_0 - x_* & b_0 & & & & \\ c_0 & a_1 - x_* & b_1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & c_{N-2} & a_{N-1} - x_* & b_{N-1} \end{pmatrix},$$

where $\mathbf{e}_0$ is the first standard basis vector with 1 in its first component and of appropriate length. Solving this lower triangular system via forward substitution provides a way to recursively evaluate each component of $\mathbf{P}^{(\alpha,\beta)}(x)$ and thus also $\mathbf{P}^{(\alpha,\beta)}(x)^\mathsf{T}\mathbf{f}$ if the coefficients of $f(x)$ in this basis are known. Clenshaw's algorithm is conceptually similar but uses backward substitution on the system

$$f(x_*) = \mathbf{P}_N^{(\alpha,\beta)}(x_*)^\mathsf{T}\boldsymbol{a} = \mathbf{e}_0^\mathsf{T}\mathcal{L}_N(x_*)^{-\mathsf{T}}\boldsymbol{a}, \tag{2.6}$$

where $\boldsymbol{a}$ is the column vector collecting $a_0$ to $a_N$. The case for the Jacobi polynomials on the triangle was recently discussed in [39] and on the basis of the recurrence in (2.5) involves a block triangular system for evaluation at $\mathbf{x}_* = (x_*, y_*)$ instead:

$$\mathcal{L}_N(\mathbf{x}_*)\mathbf{P}_N^{(\alpha,\beta,\gamma)}(\mathbf{x}_*) = \begin{pmatrix} \mathbb{1}_1 & & & \\ A_0^x - x_*\mathbb{1}_1 & B_0^x & & \\ A_0^y - y_*\mathbb{1}_1 & B_0^y & & \\ C_0^x & A_1^x - x_*\mathbb{1}_2 & B_1^x & \\ C_0^y & A_1^y - y_*\mathbb{1}_2 & B_1^y & \\ & \ddots & \ddots & \ddots \end{pmatrix} \mathbf{P}_N^{(\alpha,\beta,\gamma)}(\mathbf{x}_*) = \mathbf{e}_0,$$

where $\mathbb{1}_k$ denotes the $k \times k$ identity matrix. As this is not a triangular but a block triangular matrix one cannot use forward substitution without first applying a preconditioner:

$$\begin{pmatrix} 1 & & & \\ & B_0^+ & & \\ & & B_1^+ & \\ & & & \ddots \end{pmatrix} \mathcal{L}_N(\mathbf{x}_*) = \tilde{\mathcal{L}}_N(\mathbf{x}_*).$$

$\tilde{\mathcal{L}}_N(\mathbf{x}_*)$ is then a proper lower triangular matrix and can be used in an analogous system to the ones above to evaluate the polynomials, and thus a function expanded into that polynomial basis, recursively via forward substitution. A preconditioner which satisfies these requirements is the block diagonal matrix whose elements are comprised of a left inverse of the blocks

$$B_n = \begin{pmatrix} B_n^x \\ B_n^y \end{pmatrix},$$

such that $B_n^+ B_n = \mathbb{1}_n$. Clenshaw's algorithm for the triangle Jacobi polynomials is thus

$$f(\mathbf{x}_*) = \mathbf{P}_N^{(\alpha,\beta,\gamma)}(\mathbf{x}_*)^\mathsf{T}\boldsymbol{A} = \mathbf{e}_0^\mathsf{T}\tilde{\mathcal{L}}_N(\mathbf{x}_*)^{-\mathsf{T}}\boldsymbol{A}.$$

This system can be solved via backward substitution in optimal $O(N^2)$ complexity if one chooses $B_n^+$ carefully, see [39].

## 3. A numerical method for Volterra integral equations.

**3.1. Volterra integrals on the triangle.** In this section we describe how to represent Volterra integrals using bivariate orthogonal polynomials on a triangle domain by moving to a view of operators acting on coefficient vectors. The following section extends this method to Volterra integral equations of the first and second kind.

We first describe the idea behind the relevant operators and their use before determining their entries in matrix representation. The first operator we need is the integration operator for a function given as the coefficients of orthogonal polynomials on a triangle. We label this operator $\mathrm{Q}_y$ and it acts as

$$\mathbf{P}(x)^\mathsf{T}\mathrm{W}_\mathrm{Q}\mathrm{Q}_y\mathbf{f}_\triangle = \int_0^{1-x} f(x,y)\mathrm{d}y,$$

where $\mathrm{W_Q}$ is a to-be-determined weight function which depends on the used basis. The reason for the limits of integration to be defined in this way for $\mathrm{Q}_y$ will become clear once we discuss the explicit form of these operators and how one can make optimal use of the triangle domain's symmetries. Second, we need an operator $\mathrm{E}_y$ which extends a one-dimensional function on $[0,1]$ to one on $T^2$, that is:

$$\mathbf{P}(x)^\mathsf{T}\mathbf{f}_{[0,1]} = \mathbf{P}(x,y)^\mathsf{T}\mathrm{E}_y\mathbf{f}_{[0,1]}$$

Together these two operators can be used to compute integrals of the form

$$\int_0^{1-x} f(y)\mathrm{d}y = \mathbf{P}(x)^\mathsf{T}\mathrm{W_Q}\mathrm{Q}_y\mathrm{E}_y\mathbf{f}_{[0,1]}$$

with function $f$ depending on a single variable. To instead integrate from 0 to $x$ we use a reflection operator. Due to symmetries of the polynomials, particular basis changes in a Jacobi basis obey the simple rule [35, 22]:

$$\tilde{P}_n^{(\alpha,\beta)}(x) = (-1)^n\tilde{P}_n^{(\beta,\alpha)}(1-x).$$

We use R to refer to the operator that uses the above property to reflect the function on the $[0,1]$ interval via an appropriate basis change. The operators X and Y have important commutation relations with the introduced $\mathrm{Q}_y$ and $\mathrm{E}_y$ operators. As the $\mathrm{Q}_y$ operator integrates with respect to $y$ and collapses a bivariate coefficient vector back to a univariate one the multiplication-with-$x$ operator changes from being multiplication-with-$x$ on the triangle (= X) to being multiplication-with-$x$ on the real interval (= $\bar{\mathrm{X}}$) when pulled through the $\mathrm{Q}_y$ operator. A similar relation holds for similar reasons for Y and $\mathrm{E}_y$:

$$\mathrm{Q}_y\mathrm{X}\mathbf{f}_\triangle = \bar{\mathrm{X}}\mathrm{Q}_y\mathbf{f}_\triangle, \tag{3.1}$$

$$\mathrm{Y}\mathrm{E}_y\mathbf{f}_{[0,1]} = \mathrm{E}_y\bar{\mathrm{X}}\mathbf{f}_{[0,1]}. \tag{3.2}$$

We now give the explicit matrix representations for the operators $\mathrm{Q}_y$ and $\mathrm{E}_y$ and discuss a sensible polynomial basis choice. The explicit form of the Jacobi operators on the real line is known in the literature (e.g. [22, 39]) and thus receives no further discussion here. To determine the explicit form of $\mathrm{Q}_y$ we begin by plugging in the polynomial expansion of $f(x,y)$ into the intended integral operation and using the Jacobi polynomials on the triangle domain as seen in (2.4) for our basis $p_{n,k}$ with $\alpha=\beta=\gamma=0$:

$$\begin{aligned}\mathbf{P}^{(1,0)}(x)^\mathsf{T}\mathrm{W_Q}\mathrm{Q}_y\mathbf{f}_\triangle &= \int_0^{1-x} f(x,y)\mathrm{d}y = \int_0^{1-x}\sum_{n=0}^{\infty}\sum_{k=0}^{n} p_{n,k}(x,y)f_{n,k}\mathrm{d}y \\ &= \sum_{n=0}^{\infty}\sum_{k=0}^{n} f_{n,k}(1-x)^k\tilde{P}_{n-k}^{(2k+1,0)}(x)\int_0^{1-x}\tilde{P}_k^{(0,0)}\left(\frac{y}{1-x}\right)\mathrm{d}y \\ &= \sum_{n=0}^{\infty}\sum_{k=0}^{n} f_{n,k}(1-x)^{k+1}\tilde{P}_{n-k}^{(2k+1,0)}(x)\int_0^{1}\tilde{P}_k^{(0,0)}(s)\,\mathrm{d}s,\end{aligned}$$

where a substitution of $\frac{y}{1-x} \to s$ was made in the last step. As $\tilde{P}_k^{(0,0)}$ are just the Legendre polynomials on $[0,1]$ we see that $\int_0^1\tilde{P}_k^{(0,0)}(s)\,\mathrm{d}s = 0, \forall k > 0$ and

$\int_0^1 \tilde{P}_0^{(0,0)}(s)\,\mathrm{d}s = 1$, resulting in

$$\mathbf{P}^{(1,0)}(x)^{\mathsf{T}}\mathrm{W_Q}\mathrm{Q}_y\mathbf{f}_\triangle = \sum_{n=0}^{\infty} f_{n,0}(1-x)\tilde{P}_n^{(1,0)}(x)$$

for integration from 0 to $1-x$. By using the reflection operation we obtain an analogous result for integration from 0 to $x$. This derivation shows that starting in the Jacobi polynomial basis on the triangle $T^2$ with $\alpha = \beta = \gamma = 0$ for the approximation of $f(x,y)$ results in the following block diagonal structure for the integration from 0 to $1-x$ operator with weight $\mathrm{W_Q} = (1-x)$:

$$\mathrm{Q}_y = \begin{pmatrix} \boxed{1} & & & & \\ & \boxed{1 & 0} & & & \\ & & \boxed{1 & 0 & 0} & & \\ & & & \boxed{\ddots & \ddots & \ddots & \ddots} \end{pmatrix}$$

where the $n$-th block is an $n$-dimensional row vector with 1 in the first element and 0 in all remaining elements. An additional $(-1)^n$ term and change of basis changes this integration to be from 0 to $x$ instead. The expansion operator $\mathrm{E}_y$ from the $\mathbf{P}^{(1,0)}(x)$ basis to the canonical triangle Jacobi polynomials where $\alpha = \beta = \gamma = 0$ has the block diagonal structure

$$\mathrm{E}_y = \begin{pmatrix} \boxed{\times} & & \\ & \boxed{\times} & \\ & \boxed{\times} & \\ & & \boxed{\ddots} \\ & & \ddots \\ & & \ddots \end{pmatrix}$$

where the $n$-th block is an $n$-dimensional column vector whose $j$-th entry is given by

$$\frac{(-1)^{j+n}(2j-1)}{n}.$$

Importantly, multiplication of $\mathrm{Q}_y$ and $\mathrm{E}_y$ yields a diagonal matrix whose $n$-th entry can be directly generated without any matrix multiplication being required (compare [35]):

$$(\mathrm{Q}_y\mathrm{E}_y)_{n,n} = (\mathrm{D}_y)_{n,n} = \frac{(-1)^{n+1}}{n}.$$

These observations justify the basis choices as well as the choice of the limits of integration for $\mathrm{Q}_y$ from the standpoint of computational efficiency. Defining $\mathrm{Q}_y$ as the integration operator from 0 to $x$ does not avoid the reflection step and only results in a less efficient or equivalent placement for it.

**3.2. Kernel computations using Clenshaw's algorithm.** Putting all the above observations together means one can save a significant amount of computation time by the use of a recurrence when simultaneously using an operator valued polynomial approximation for the kernel $K(\mathrm{X},\mathrm{Y})$ and then using the known commutation

relations in (3.1–3.2). To illustrate the idea behind this approach we first discuss how to do this for a monomial kernel (or equivalently a kernel approximated in a monomial basis) and then show how these ideas can be expanded to arbitrary polynomial bases for the kernel using a variant of Clenshaw's algorithm.
Assuming a monomial expansion for the kernel, i.e. $K(x,y) = \sum_{n=0}^{\infty}\sum_{j=0}^{n} k_{nj}x^{n-j}y^j$, the primary part of the Volterra integration operator has the form

$$\mathrm{Q}_y K(\mathrm{X},\mathrm{Y})\mathrm{E}_y = \mathrm{Q}_y\left(\sum_{n=0}^{\infty}\sum_{j=0}^{n} k_{nj}\mathrm{X}^{n-j}\mathrm{Y}^j\right)\mathrm{E}_y = \sum_{n=0}^{\infty}\sum_{j=0}^{n} k_{nj}\bar{\mathrm{X}}^{n-j}\mathrm{Q}_y\mathrm{E}_y\bar{\mathrm{X}}^j,$$

where we have used the commutation relations in (3.1–3.2) to rewrite the summation using the Jacobi operator for the interval Jacobi polynomials. Recalling that $\mathrm{Q}_y\mathrm{E}_y$ is a diagonal matrix which can be generated without any need to separately compute and multiply $\mathrm{Q}_y$ and $\mathrm{E}_y$, all that is left to compute are the required combinations of $\mathrm{Q}_y\mathrm{E}_y$ with the Jacobi operators, which can be built up recursively. This kind of recursive computation of all the required elements for the kernel can save significant computation cost if executed correctly. Since only the coefficients of $K(x,y)$ for this basis actually change across different problems one can in principle also store the basis elements $\bar{\mathrm{X}}^{n-j}\mathrm{Q}_y\mathrm{E}_y\bar{\mathrm{X}}^j$ and re-use them making this numerical evaluation of Volterra integrals even faster upon repeated use. This approach differs slightly depending on whether one intends to compute integrals from 0 to $1-x$ or to compute integrals from 0 to $x$. In the case of integrals from 0 to $x$, one is either required to supply $K(1-x,y)$ to the algorithm or alternatively the Jacobi operators on the left can be replaced by $(\mathbb{1}-\bar{\mathrm{X}})$ to account for the reflection, meaning that the basis elements become $(\mathbb{1}-\bar{\mathrm{X}})^{n-j}\mathrm{Q}_y\mathrm{E}_y\bar{\mathrm{X}}^j$. Taking the weight $\mathrm{W}_{\mathrm{Q}}$ into consideration the full Volterra integral operator is then

$$\mathrm{R}(\mathbb{1}-\bar{\mathrm{X}})\mathrm{Q}_y K(\mathrm{X},\mathrm{Y})\mathrm{E}_y = \mathrm{R}(\mathbb{1}-\bar{\mathrm{X}})\sum_{n=0}^{\infty}\sum_{j=0}^{n} k_{nj}(\mathbb{1}-\bar{\mathrm{X}})^{n-j}\mathrm{Q}_y\mathrm{E}_y\bar{\mathrm{X}}^j.$$

This straightforward approach evidently only works if the kernel is of a form that may sensibly be approximated using monomials but it inspires an analogous approach based on expanding the kernel in its own orthogonal polynomial basis which need not be the same as those used to expand the function $f$. We use a variant of the Clenshaw algorithm introduced in section 2.3 to build the kernel in terms of the Jacobi operators. In principle one could compute $K(\mathrm{X},\mathrm{Y})$ as a full multiplication operator acting on a triangle Jacobi coefficient vector using an operator-valued version of Clenshaw's algorithm as discussed in [39]. This is not the most efficient way to approach this problem, however, as it would mean losing the diagonal $\mathrm{Q}_y\mathrm{E}_y$ since for such an operator the multiplication with $K(\mathrm{X},\mathrm{Y})$ would need to happen between $\mathrm{Q}_y$ and $\mathrm{E}_y$. Nevertheless, we will briefly discuss how to generate this multiplication by $K(\mathrm{X},\mathrm{Y})$ operator in order to see which modifications one can make to this approach in order to respect the symmetries of the triangle and end up with recursive basis generation similar to the monomial kernel expansion case.

The multiplication by $K(x,y)$ operator, which we label $\mathrm{M}_K$, can be written in an operator Clenshaw approach as (see [39, 36, 50]):

$$\mathrm{M}_K = (\mathbf{e}_0 \otimes \mathbb{1})\mathcal{L}^{-\mathsf{T}}\mathbf{K}_{\triangle}, \tag{3.3}$$

where $\otimes$ denotes the Kronecker product and $\mathcal{L}$ is defined as

$$\mathcal{L} = \begin{pmatrix} (\mathbb{1}_1 \otimes \mathbb{1}) & & & & \\ (A_0^x \otimes \mathbb{1}) - (\mathbb{1}_1 \otimes \mathrm{X}) & (B_0^x \otimes \mathbb{1}) & & & \\ (A_0^y \otimes \mathbb{1}) - (\mathbb{1}_1 \otimes \mathrm{Y}) & (B_0^y \otimes \mathbb{1}) & & & \\ (C_0^x \otimes \mathbb{1}) & (A_1^x \otimes \mathbb{1}) - (\mathbb{1}_2 \otimes \mathrm{X}) & (B_1^x \otimes \mathbb{1}) & & \\ (C_0^y \otimes \mathbb{1}) & (A_1^x \otimes \mathbb{1}) - (\mathbb{1}_2 \otimes \mathrm{Y}) & (B_1^y \otimes \mathbb{1}) & & \\ & \ddots & & \ddots & \ddots \end{pmatrix}.$$

As discussed for the Clenshaw evaluation method in section 2.3 this system requires preconditioning to become solvable via backward substitution. For this case the preconditioner is

$$\begin{pmatrix} (\mathbb{1}_1 \otimes \mathbb{1}) & & & \\ & (B_0^+ \otimes \mathbb{1}) & & \\ & & (B_1^+ \otimes \mathbb{1}) & \\ & & & \ddots \end{pmatrix} \mathcal{L} = \tilde{\mathcal{L}},$$

with the $B_n^+$ defined as in section 2.3. Using such an operator valued Clenshaw algorithm one can compute $\mathrm{M}_K$ and thus obtain $\mathrm{Q}_y K(\mathrm{X}, \mathrm{Y})\mathrm{E}_y$ via $\mathrm{Q}_y \mathrm{M}_K \mathrm{E}_y$. However, as discussed above, for our purposes of Volterra integral operators this is computationally wasteful and misses the chance to take advantage of the triangle symmetries which allow for $\mathrm{Q}_y\mathrm{E}_y$ to be directly computable and diagonal. So instead we replace the $\mathbf{K}_\triangle$ in (3.3) by $(\mathbf{K}_\triangle \otimes \mathrm{Q}_y\mathrm{E}_y)$. The relations (3.1–3.2) then imply that all X operators may be replaced by a left multiplication with $\bar{\mathrm{X}}$ and all Y operators may be replaced by a right multiplication with $\bar{\mathrm{X}}$ (respectively denoted by a $\diamond$ on the appropriate side). The system to solve thus becomes

$$\mathrm{Q}_y K(\mathrm{X}, \mathrm{Y})\mathrm{E}_y = (\mathbf{e}_0 \otimes \mathbb{1})\mathcal{L}_V^{-\mathsf{T}}(\mathbf{K}_\triangle \otimes \mathrm{Q}_y\mathrm{E}_y),$$

with

$$\mathcal{L}_V = \begin{pmatrix} (\mathbb{1}_1 \otimes \mathbb{1}) & & & & \\ (A_0^x \otimes \mathbb{1}) - (\mathbb{1}_1 \otimes \bar{\mathrm{X}}\diamond) & (B_0^x \otimes \mathbb{1}) & & & \\ (A_0^y \otimes \mathbb{1}) - (\mathbb{1}_1 \otimes \diamond\bar{\mathrm{X}}) & (B_0^y \otimes \mathbb{1}) & & & \\ (C_0^x \otimes \mathbb{1}) & (A_1^x \otimes \mathbb{1}) - (\mathbb{1}_2 \otimes \bar{\mathrm{X}}\diamond) & (B_1^x \otimes \mathbb{1}) & & \\ (C_0^y \otimes \mathbb{1}) & (A_1^x \otimes \mathbb{1}) - (\mathbb{1}_2 \otimes \diamond\bar{\mathrm{X}}) & (B_1^y \otimes \mathbb{1}) & & \\ & \ddots & & \ddots & \ddots \end{pmatrix}.$$

After preconditioning as above, this allows the recursive and efficient computation of $\mathrm{Q}_y K(\mathrm{X}, \mathrm{Y})\mathrm{E}_y$ via an operator valued Clenshaw-type algorithm while at the same time taking advantage of the diagonal nature of $\mathrm{Q}_y\mathrm{E}_y$. As in the monomial case, this approach has to be modified when integrating from 0 to $x$ instead of from 0 to $1 - x$. In the 0 to $x$ case one needs to take the reflection into account, which ends up either replacing all the left multiplications with $\bar{\mathrm{X}}$ by left multiplications with $(\mathbb{1} - \bar{\mathrm{X}})$ for the same reasons as above, while the right multiplications corresponding to $y$ multiplication remain the same, or requiring that $K(1 - x, y)$ be supplied to the algorithm. Finally, this operator still requires left multiplication with the basis dependent weight $\mathrm{W}_\mathrm{Q}$ to represent the full Volterra integral operator for this approach.

**3.3. Numerical solutions to linear Volterra integral equations.** The computational method for Volterra integrals described above has a natural extension to solving Volterra integral equations, which we describe in this section. Most generally a Volterra integral equation is any equation in which the unknown appears at least once as the integrand of a Volterra integral as defined in (1.1) above. One usually distinguishes between at least two types of Volterra integral equations which are labeled Volterra integral equations of the first and second kind respectively. The Volterra integral equation of the first kind we will be interested in takes the following form:

$$\int_0^x K(x,y)u(y)\mathrm{d}y = g(x), \tag{3.4}$$

where $u(x)$ is the unknown function to be solved for, $K(x,y)$ is a given kernel and $g(x)$ is a given function. Volterra integral equations of the second kind we will be interested in take the following form[1]:

$$u(x) - \int_0^x K(x,y)u(y)\mathrm{d}y = g(x), \tag{3.5}$$

where once again $u(x)$ is the unknown function and $K(x,y)$ and $g(x)$ are given. While this is not further explored in this paper, there are natural extensions of these methods for other linear Volterra-type integral equations such as the third-kind equations discussed in [2, 3, 46].
Whenever we write $\mathrm{Q}_y K(\mathbb{1}-\mathrm{X},\mathrm{Y})\mathrm{E}_y$ in the coming sections, we mean to imply that this operator is computed using the Clenshaw approach detailed in section 3.2.

**3.3.1. Equations of the first kind.** Extending the above methods for Volterra integrals to Volterra integral equations is straightforward, though one needs to be mindful of the appropriate reflections. Using the above notation conventions, one way to write the Volterra integral equation of the first kind is

$$\begin{aligned}\tilde{\mathbf{P}}^{(1,0)}(x)^{\mathsf{T}}(\mathbb{1}-\bar{\mathrm{X}})\mathrm{Q}_y K(\mathbb{1}-\mathrm{X},\mathrm{Y})\mathrm{E}_y\mathbf{u} &= \tilde{\mathbf{P}}^{(1,0)}(x)^{\mathsf{T}}\bar{\mathbf{g}},\\ \Rightarrow \tilde{\mathbf{P}}^{(1,0)}(x)^{\mathsf{T}}\mathbf{u} &= \tilde{\mathbf{P}}^{(1,0)}(x)^{\mathsf{T}}\left((\mathbb{1}-\bar{\mathrm{X}})\mathrm{Q}_y K(\mathbb{1}-\mathrm{X},\mathrm{Y})\mathrm{E}_y\right)^{-1}\bar{\mathbf{g}}.\end{aligned}$$

The notation $\bar{\mathbf{g}}$ is used to indicate that we are directly supplying the coefficients of the reflected $g(1-x)$ to save an unnecessary additional reflection step, as formally we are solving the equivalent

$$\int_0^{1-t} K(1-t,y)u(y)\mathrm{d}y = g(1-t). \tag{3.6}$$

All function coefficient vectors in this section are initially expanded in the $\tilde{\mathbf{P}}^{(1,0)}(x)$ basis. This method works in numerical experiments but deriving convergence properties for it proves to be difficult (as is usual for Volterra equations of the first kind). However, under the condition that we can expand the function $q(x) = \frac{g(1-x)}{1-x}$ instead of $g(1-x)$ in $\tilde{\mathbf{P}}^{(1,0)}(x)$, one can find convergence conditions (see section 5 for details). Note that solvability of the Volterra integral equation of the first kind implies that both $g$ and $q$ must vanish when the upper limit of integration vanishes. When using

[1] For simplicity, we have divided through by $\lambda$ and incorporated into $K$ and $g$.

$\mathbf{q}$ to denote the coefficient vector of $q(x) = \frac{g(1-x)}{1-x}$ the method then becomes

$$\tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{q},$$
$$\Rightarrow \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \left(\mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y\right)^{-1} \mathbf{q}.$$

meaning that solving this type of equation for $u(x)$ is as simple as computing the coefficient vectors and operators (see the respective sections above for efficient ways to do so) and then solving a banded system of linear equations.

**3.3.2. Equations of the second kind.** Using the above-introduced weighted lowering operator $\mathrm{L}^{(0,0)}_{(1,0)}$ which shifts to the $\tilde{\mathbf{P}}^{(0,0)}(x)$ basis while multiplying with $(1-x)$, reflecting the result and then using a raising operator $\mathrm{S}^{(1,0)}_{(0,0)}$ to return to the $\tilde{\mathbf{P}}^{(1,0)}(x)$ basis we can write Volterra integral equations of the second kind as

$$\tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \left(\mathbb{1} - \mathrm{S}^{(1,0)}_{(0,0)} \mathrm{R} \mathrm{L}^{(0,0)}_{(1,0)} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y\right) \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{g},$$
$$\Rightarrow \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \left(\mathbb{1} - \mathrm{S}^{(1,0)}_{(0,0)} \mathrm{R} \mathrm{L}^{(0,0)}_{(1,0)} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y\right)^{-1} \mathbf{g},$$

which can once again be solved for $u(x)$ using any linear system of equations solver. Reflecting without the lowering and raising operator is not possible (although there are alternative ways to use such operators to accomplish the same goal) as this would result in an inconsistency between the bases used for the two appearances of $\mathbf{u}$.

**3.3.3. Different limits of integration.** As mentioned above, a similar derivation leads to an analogous method for Volterra integral equations of the first and second kind with different limits of integration:

$$\int_0^{1-x} K(x,y) u(y) \mathrm{d}y = g(x), \tag{3.7}$$
$$u(x) - \int_0^{1-x} K(x,y) u(y) \mathrm{d}y = g(x), \tag{3.8}$$

This results in an identity operator replacing the reflection and conversion operators in the above solution methods and in fact makes these types of equations even more efficient to solve but limits of integration of this sort are seen less often in applications. In particular, the operator version of Volterra integral equations of the first kind with limits of integration 0 to $1-x$ is:

$$\tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathrm{Q}_y K(\mathrm{X}, \mathrm{Y}) \mathrm{E}_y \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{q},$$
$$\Rightarrow \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \left(\mathrm{Q}_y K(\mathrm{X}, \mathrm{Y}) \mathrm{E}_y\right)^{-1} \mathbf{q}.$$

where now $\mathbf{q}$ is the coefficient vector of $q(x) = \frac{g(x)}{1-x}$. Equations of the second kind with these limits of integration can be written as:

$$\tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \left(\mathbb{1} - (\mathbb{1} - \bar{\mathrm{X}}) \mathrm{Q}_y K(\mathrm{X}, \mathrm{Y}) \mathrm{E}_y\right) \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{g},$$
$$\Rightarrow \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \mathbf{u} = \tilde{\mathbf{P}}^{(1,0)}(x)^\mathsf{T} \left(\mathbb{1} - (\mathbb{1} - \bar{\mathrm{X}}) \mathrm{Q}_y K(\mathrm{X}, \mathrm{Y}) \mathrm{E}_y\right)^{-1} \mathbf{g}.$$

We present an implementation of both options for the limits of integration in the next section.

**4. Numerical examples.** We present four sets of numerical examples to validate our implementation. The first set concerns itself with Volterra integral equations of the first kind and the second with Volterra integral equations of the second kind with kernels of varying oscillatory intensity. In the third set we study a parametrized set of Volterra integral equations requiring increasing orders of polynomials to accurately approximate and the fourth set discusses a singular Volterra integral equation stemming from a heat conduction problem with mixed boundary conditions. In the third set we also provide performance comparisons to the state-of-the-art collocation method package Chebfun [40, 8, 21] which introduced an option for Volterra integral equations in [20]. As oscillatory functions require high orders of polynomials to approximate accurately and the method was not designed for singular kernels, the second, third and fourth set are also designed to test the method's stability.
The computations presented in this section have been performed with an implementation of the scheme in the Julia programming language [9] in the framework of ApproxFun.jl and MultivariateOrthogonalPolynomials.jl [37, 36, 47]. The coefficients of the solution have relative accuracy with standard floating point arithmetic, even as they decay below machine precision. Values for absolute errors presented in this section converge beyond the precision of 64-bit floating point numbers because of the rapid convergence of the method and the way ApproxFun.jl implements function approximation (cf. [37, 36, 47])—the only time beyond 64-bit floating point precision numbers (via the inbuilt `BigFloat` type) were used is in the analytic solutions used as comparisons, as otherwise the convergence of the error would be capped by the precision at which the analytic solution is evaluated.

**4.1. Volterra integral equations of the first kind.** We investigate the numerical solution of the following two example Volterra integral equations of the first kind:

$$e^{-x} + e^x(-1+2x) = 4\int_0^x e^{y-x}u_1(y)\mathrm{d}y, \tag{4.1}$$

$$\frac{\sin(4\pi^2x^2)}{x} = \int_0^x e^{-10\left(x-\frac{1}{3}\right)^2-10\left(y-\frac{1}{3}\right)^2}u_2(y)\mathrm{d}y. \tag{4.2}$$

The analytic solution to the first equation can be found to be:

$$u_1(x) = xe^x.$$

We present the absolute error between the analytic and numerical solution for $u_1(x)$ using the orthogonal polynomial method introduced in this paper in Figure 1A for different matrix dimensions $n \times n$ and the absolute error between the numerical solution for $u_2(x)$ and a high degree solution computed with $n = 5050$ in Figure 1B.

**4.2. Volterra integral equations of the second kind with oscillatory kernels.** We seek numerical solutions $u_1$, $u_2$ and $u_3$ to the following three Volterra integral equations of the second kind with kernels of varying oscillatory intensity:

$$u_1(x) = \frac{e^{-10\pi x}(1+20\pi)-2+\cos(10\pi x)+\sin(10\pi x)}{20\pi} + \int_0^x 2\sin^2(5\pi(x-y))u_1(y)\mathrm{d}y \tag{4.3}$$

$$u_2(x) = \frac{e^{\frac{x}{2}}}{\pi} + \int_0^x \left(\sin(10\pi x) + \cos(10\pi y)\right)u_2(y)\mathrm{d}y \tag{4.4}$$

$$u_3(x) = e^{x^2-2x} + \int_0^{1-x}\left(-2x + y + \sin(25x^2 + 8\pi y)\right)u_3(y)\mathrm{d}y. \tag{4.5}$$

We include contour plots of the specified kernels on their natural triangle domains in Figure 2. One can find an analytic solution to the first equation:

$$u_1(x) = e^{-10\pi x}.$$

For the other two equations, we instead compare to a numerical solution of high degree ($n = 5050$). We plot the absolute error convergence of the numerical solutions in Figure 3. Due to the oscillatory character of these kernels and the number of coefficients involved, this can be considered a moderate stress test of the Clenshaw approach to computing Volterra integral operators.

**4.3. Performance comparison for high polynomial orders.** In order to visualize the performance improvements one gains from making use of the bandedness of the Volterra operator, we turn to the following parametrized example of a Volterra integral equation of second kind:

$$u_k(x) = g_k(x) + \int_0^x (x+y)u_k(y)\mathrm{d}y, \tag{4.6}$$

where $k \in \mathbb{N}$ and $g_k(x)$ is

$$g_k(x) = \frac{\cos\left(k^2x^2\right) + 2k^2\sin\left(k^2x^2\right) - 1}{2k^2} - \frac{x}{k}\sqrt{\frac{\pi}{2}}\int_0^{\sqrt{\frac{2}{\pi}}kx} \sin\left(\frac{\pi y^2}{2}\right)\mathrm{d}y.$$

While $g_k(x)$ in this example contains a so-called type-$S$ Fresnel integral [35, 7.2(iii)] which can be thought of as a special case Volterra integral with kernel $K(x, y) = 1$, there is little reason to compute Fresnel integrals using a Volterra operator approach, as accurate high performance code for these already exists in most programming languages. The analytic solution to the above integral equation can be found to be

$$u_k(x) = \sin(k^2x^2),$$

for all $k \in \mathbb{N}$. With increasing $k$ the solution to this integral equation rapidly becomes increasingly oscillatory and thus requires high orders to accurately approximate. This parametrized set of Volterra integral equations provides us with a structured way to capture performance improvements over dense collocation methods. We compare computation time and error compared to the analytic solution for the proposed sparse spectral method and Chebfun's implementation of Volterra integral equations [20] for different parameter values $k$ in Table 1 and present a visualization of convergence rate for the sparse method in Figure 4. Exponential convergence is observed once the polynomial order is high enough to resolve the frequency of the solution. For reasons discussed above we start benchmarking time *after* the approximation of the Fresnel integral with Julia and MATLAB internal tools to avoid Julia's faster Fresnel integral computation influencing the results. However, even taking the computation and approximation of the Fresnel integral into account for the benchmarking, the sparse method never exceeded $1s$ of CPU time.

**4.4. Singular Volterra integral equation of the second kind in heat conduction with mixed boundary conditions.** Finally we discuss a more application-oriented example discussed in a handful of different variations in [19, 18, 17, 52, 7]:

$$u(x) = g(x) + \int_0^x \frac{y^{\mu-1}}{x^\mu}u(y)\mathrm{d}y. \tag{4.7}$$

To see how equations of this type can result from heat conduction problems of the form $\frac{\partial^2 u}{\partial x} - \frac{1}{\alpha^2}\frac{\partial u}{\partial y} = 0$ with mixed boundary conditions, see for example [18]. This equation varies both in its singularity properties as well as its number of solutions depending on the parameter $\mu$. This example equation stemming from an application of Volterra integrals demonstrates that the method developed in this paper has a broader range of applicability and can in some cases extend to certain classes of singular problems as well, despite this not being part of the considerations during the development of the method. For testing purposes we choose the following for $g(x)$:

$$g_1(x) = (1 + x + x^2)$$
$$g_2(x) = \frac{(1 + 4\pi^2 x^2)\sinh(2\pi x) - 2\pi x \cosh(2\pi x)}{4\pi^2 x^2}.$$

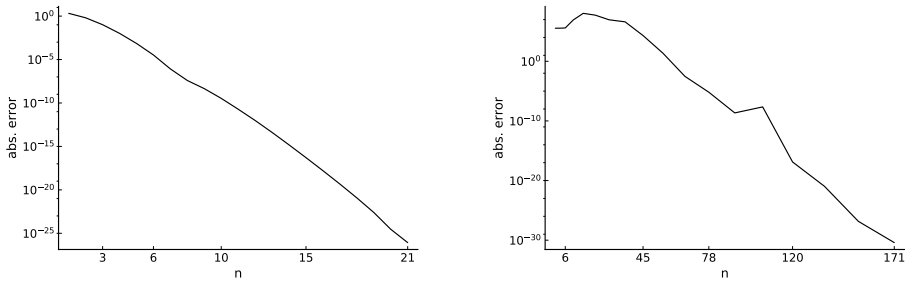
The following analytic solutions to these equations can be found for general $\mu$ for $g_1$ (e.g. in [52]) and for $\mu = 3$ for $g_2$:

$$u_1(x, \mu) = \frac{\mu}{\mu - 1} + \frac{\mu + 1}{\mu}x + \frac{\mu + 2}{\mu + 1}x^2,$$
$$u_2(x, \mu = 3) = \sinh(2\pi x).$$

As the kernel is separable, the problem can instead be treated as

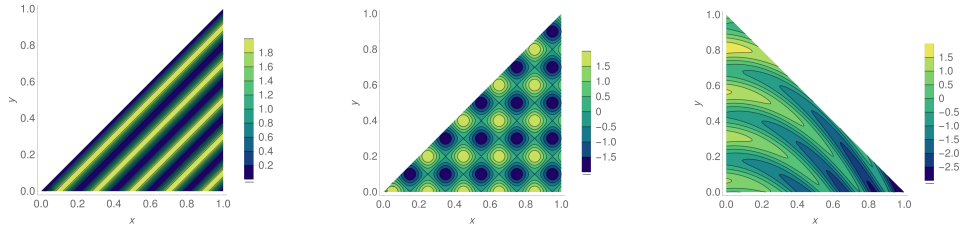
$$x^\mu u(x) = x^\mu g(x) + \int_0^x y^{\mu-1} u(y) \mathrm{d}y,$$

which can be solved by appropriately adding multiplications with Jacobi operators or altering the supplied $g(x)$ in the method to solve Volterra integral equations of the second kind. We plot numerical solutions obtained for $g_1(x)$ with $\mu = 7$ and $g_2(x)$ with $\mu = 3$ in Figure 5. The naturally more error prone neighborhood of the singularity can be well approximated arbitrarily close to the singularity (though not at the exact point of the singularity itself) using higher values of $n$ if needed. For $g_2(x)$ the method shows no instability at the singularity of the kernel.



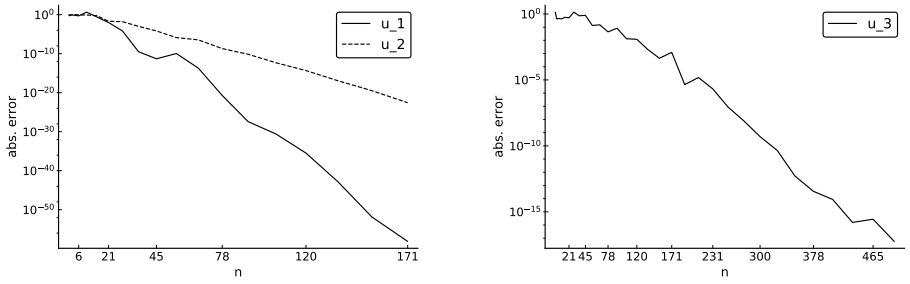

(a) (b)

Fig. 1: (a) shows absolute error between (4.1) and the known analytic solution while (b) compares (4.2) to a solution computed with $n = 5050$.



(a) $K_1(x, y)$ (b) $K_2(x, y)$ (c) $K_3(x, y)$

Fig. 2: Contour plots of oscillatory kernels for equations (4.3–4.5) on their natural triangle domains.




(a) (b)

Fig. 3: Absolute errors for equations (4.3–4.5). $u_1(x)$ is compared to the analytic solution, $u_2(x)$ and $u_3(x)$ are compared to a solution computed with $n = 5050$.

| **k (Sparse)** | **CPU time** | **approx. order** | **abs. error** |
|---|---|---|---|
| 1 | 0.001s | 19 | 7.8e-16 |
| 10 | 0.002s | 128 | 4.0e-14 |
| 50 | 0.08s | 2200 | 9.0e-13 |
| 75 | 0.29s | 3850 | 3.1e-12 |
| **k (Chebfun)** | **CPU time** | **autom. order** | **abs. error** |
| 1 | 0.18s | 17 | 1.0e-15 |
| 10 | 0.48s | 119 | 2.9e-14 |
| 50 | 27.0s | 1768 | 7.9e-13 |
| 75 | 163.5s | 4096 | 2.0e-12 |

Table 1: Quantitative performance comparison of sparse method and Chebfun for Equation (4.6). Chebfun's approximation order was automatically chosen, while the sparse method can generate results with similar accuracy in less time. CPU time measured on Intel Core i7-6700T CPU @ 2.80GHz.

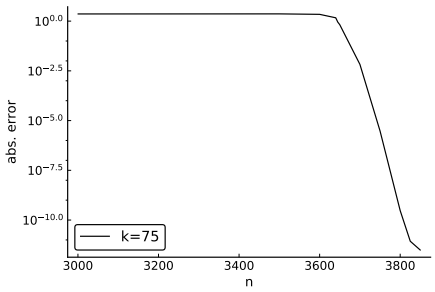
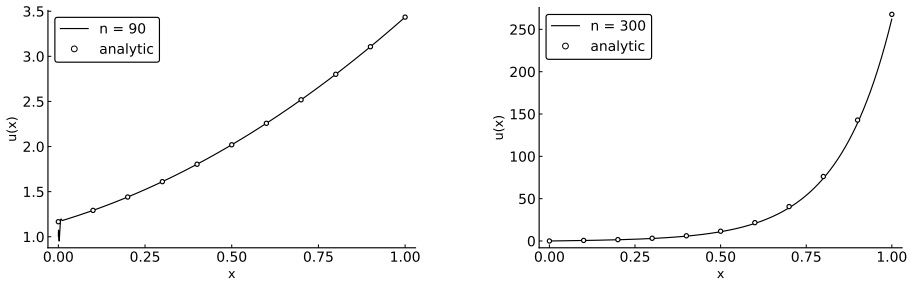



Fig. 4: Sparse method absolute errors for Equation (4.6) with $k = 75$.




(a) $g_1(x)$ with $\mu = 7$ (b) $g_2(x)$ with $\mu = 3$

Fig. 5: Numerical and analytic solutions to the problem in (4.7).

**5. Convergence of the method.** In this section we make use of the fact that the coefficient space of orthogonal polynomials is equivalent to an infinite-dimensional Banach space (in particular a sequence space). We prove convergence of the proposed method for second kind problems with general kernels which are sufficiently smooth to be approximated by Jacobi polynomials and for first kind problems with sufficiently smooth kernels given that $\forall x \in [0,1] : K(x,x) \neq 0$. The strategy for the analysis of the method is to show that the operators to be inverted for Volterra integral equations of the second kind can be written as compact perturbations of the identity (compare [36, 45, 31]), i.e. can be written as

$$(\mathbb{1} + \mathcal{K})u = g, \tag{5.1}$$

where $\mathcal{K}$ is compact. Operators of this form are either invertible or neither injective nor surjective by the Fredholm alternative, cf. [6, 30]. The assumption of well-posedness for the equation thus guarantees that an operator of this form is invertible and standard convergence results for finite section methods [11] then guarantee convergence. We begin by discussing the solver for Volterra integral equations of the second kind, as the analysis for first kind problems is more involved.

### 5.1. Equations of the second kind.

DEFINITION 5.1. *We define the projection operators $\mathcal{P}_n : \ell^2 \to \ell^2$ which map a given coefficient vector to a truncated version of itself with non-zero entries for the first $n$ coefficients only.*

DEFINITION 5.2. *The analysis operator $\mathcal{E} : L^2(0,1) \to \ell^2$ is the inclusion of a square integrable function into the $\ell^2$ coefficient space of the complete basis of orthogonal Jacobi polynomials and is a bounded operator. The synthesis operator is its inverse $\mathcal{E}^{-1} : \ell^2 \to L^2(0,1)$, which is also bounded. Note the terms analysis and synthesis are terminology in frame theory [14, 15].*

LEMMA 5.3. *The coefficient space Volterra integral operator $\mathrm{V}_K$ is compact, where $\mathrm{V}_K : \ell^2 \to \ell^2$ for a given kernel $K(x,y) \in L^2[T^2]$ with limits of integration 0 to $x$ acting on the coefficient vector Banach space $\ell^2$ of the Jacobi polynomials $\tilde{\mathbf{P}}^{(1,0)}(x)$ is of the form*

$$\mathrm{V}_K = \mathrm{L}^{(0,0)}_{(1,0)} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y,$$

*with the respective operators defined as in section 3.*

*Proof.* $\mathrm{V}_K = \mathrm{L}^{(0,0)}_{(1,0)} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y$ follows from the definition of the involved operators, see section 3. To see compactness of $\mathrm{V}_K$ we consider the following diagram of functions between Banach spaces which represents the formalized version of the method:

$$\begin{array}{ccc} L^2(0,1) & \xrightarrow{\mathcal{V}_K} & L^2(0,1) \\ \Big\downarrow{\scriptstyle \mathcal{E}} & & \Big\uparrow{\scriptstyle \mathcal{E}^{-1}} \\ \ell^2 & \xrightarrow[\mathrm{V}_K]{} & \ell^2 \end{array}$$

$\mathcal{V}_K$ for a kernel $K(x,y) \in L^2[T^2]$ is the Volterra integral operator for said kernel acting on $L^2(0,1)$. It is a classical result of functional analysis that such Volterra integral operators $\mathcal{V}_K$ are Hilbert–Schmidt operators and thus compact [34]. It follows that $\mathrm{V}_K = \mathcal{E} \circ \mathcal{V}_K \circ \mathcal{E}^{-1}$ is a finite composition of bounded and compact operators between Banach spaces and hence itself compact. ∎

LEMMA 5.4. *For $\mathrm{V}_K$ and $\mathcal{P}_n$ defined as above, we have*

$$\lim_{n\to\infty} \|\mathrm{V}_K - \mathcal{P}_n \mathrm{V}_K \mathcal{P}_n^{\mathsf{T}}\| = 0.$$

*Proof.* This follows directly from the compactness of $\mathrm{V}_K$ and the fact that $\ell^2$ is a Hilbert space and thus has the approximation property [30]. ∎

The above lemma justifies referring to the finite-dimensional projections $\mathcal{P}_n \mathrm{V}_K \mathcal{P}_n^{\mathsf{T}}$ of the Volterra operator as approximations.

LEMMA 5.5. $\mathrm{S}^{(1,0)}_{(0,0)} \mathrm{R} \mathrm{L}^{(0,0)}_{(1,0)} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y$ *is compact on $\ell^2$ and thus Volterra integral equations of the second kind can be written in the form $(\mathbb{1} + \mathcal{K})\mathbf{u} = \mathbf{g}$ with $\mathcal{K}$ compact.*

*Proof.* The operators $\mathrm{S}^{(1,0)}_{(0,0)}$ and R acting on the Banach space $\ell^2$ can both readily be seen to be bounded operators from their definitions from the Jacobi polynomial's recurrence relationships [35, 18.9.5]. The result then follows from the observation that the Volterra integral operator $\mathrm{L}^{(0,0)}_{(1,0)} \mathrm{Q}_y K(\mathbb{1} - \mathrm{X}, \mathrm{Y}) \mathrm{E}_y$ was shown to be compact

and composition of bounded operators with a compact operator yields a compact operator. ∎

An analogous chain of arguments immediately establishes:

LEMMA 5.6. *The Volterra integral operator for the limits* 0 *to* $1 - x$ *is compact and can be written as*

$$\mathrm{V}_K = (\mathbb{1} - \bar{\mathrm{X}})\mathrm{Q}_y K(\mathrm{X}, \mathrm{Y})\mathrm{E}_y.$$

*The method is thus also of the form in (5.1).*

COROLLARY 5.7. *The method described in section 3.3 converges like* $\|\mathbf{u} - \mathcal{P}_n\mathbf{u}\| \to 0$ *as* $n \to \infty$ *for well-posed Volterra integral equations of the second kind.*

*Proof.* As the method is of the form in (5.1), i.e. $(\mathbb{1} + \mathcal{K})\mathbf{u} = \mathbf{g}$ with $\mathcal{K}$ compact, the result is a corollary of the above results combined with the known invertibility and convergence properties for problems of this form in finite section methods, see e.g. [11]. ∎

**5.2. Equations of the first kind.** The Fredholm alternative and Neumann series arguments underlying the proofs above break down for first kind problems as the Volterra operator $\mathrm{V}_K : \ell^2 \to \ell^2$ is compact on the infinite dimensional Banach space $\ell^2$ and therefore is strictly singular, cf. [6]. Thus, while the finite dimensional approximations $\mathrm{V}_n$ of the Volterra operator may have an inverse $\mathrm{V}_n^{-1}$, it is not obvious that $\mathbf{u}_n = \mathrm{V}_n^{-1}\mathbf{q}$ converges to $\mathbf{u}$ in the limit. The problem can be made well-posed, however, if one considers the Volterra operator as a map between two different appropriately chosen Banach spaces. Under sufficient continuity assumptions as well as the assumption that a given Volterra integral equation of the first kind has a solution, this problem may then be salvaged by finding a preconditioner which allows us to rewrite it as a problem involving operators which are compact perturbations of Toeplitz operators. We begin by assuming a polynomial kernel from where an extension argument directly yields that it also applies for the non-polynomial case. Note that in this section we will prove convergence of the method only for the case of limits of integration 0 to $1 - x$. This is not a limitation for the case of integral equations of the first kind, since solving

$$\int_0^t K(t, y)u(y)\mathrm{d}y = g(t).$$

and

$$\int_0^{1-x} K(1 - x, y)u(y)\mathrm{d}y = g(1 - x).$$

are formally equivalent, as solving one automatically solves the other with $t = 1 - x$. The reason for the particular choice for our proofs is that some arguments are more clear in this variant. Furthermore, as the monomial expansion and Clenshaw algorithm based Volterra operators are the same for polynomial kernels the analysis will make use of the simpler structure of the former.

To discuss invertibility for equations of the first kind we need to reframe the Volterra operator as a map between two different Banach spaces, which are similar in spirit to Sobolev spaces.

DEFINITION 5.8. *Let $\ell^2_\lambda$ with $\lambda \geqslant 0$ denote the Banach space with norm*

$$\|\mathbf{u}\|_{\ell^2_\lambda} = \sqrt{\sum_{n=0}^{\infty} \left((1+n)^\lambda |u_n|\right)^2} < \infty.$$

Any $\mathbf{u} \in \ell^2_\lambda$ corresponds uniquely to a $\mathbf{u} \in \ell^2$ so we have $\ell^2_\lambda \subset \ell^2$ whereas the converse is clearly not the case.

LEMMA 5.9. *Let $\mathrm{V}_K : \ell^2 \to \ell^2_1$ denote the Volterra operator in coefficient space of $\tilde{\mathbf{P}}^{(1,0)}(x)$ with limits of integration 0 to $1-x$ for a given polynomial kernel*

$$K(x,y) = \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} x^{n-j} y^j.$$

*Then*

$$\mathrm{V}_K = (\mathbb{1} - \bar{\mathrm{X}})\mathrm{D}\left(\mathrm{D}^{-1} \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} \bar{\mathrm{X}}^{n-j} \mathrm{D} \bar{\mathrm{X}}^j\right),$$

*with $\mathrm{D} = \mathrm{Q}_y \mathrm{E}_y$, $\mathrm{D} : \ell^2 \to \ell^2_1$ and $\mathrm{D}^{-1} : \ell^2_1 \to \ell^2$.*

*Proof.* That $\mathrm{D} = \mathrm{Q}_y \mathrm{E}_y$ is diagonal with entries $\frac{(-1)^{n+1}}{n}$ is due to properties of the Jacobi polynomials, see section 3 as well as [35, 18.6.1 and 18.17.1]. The important observation to make is that D can be thought of as $\mathrm{D} : \ell^2 \to \ell^2_1$, which makes D a bounded and invertible operator with $\mathrm{D}^{-1} : \ell^2_1 \to \ell^2$. With $\mathrm{V}_K$ and $K(x,y)$ as above, we thus have

$$\mathrm{V}_K = (\mathbb{1} - \bar{\mathrm{X}}) \sum_{n=0}^{M} \sum_{j=0}^{n} \bar{\mathrm{X}}^{n-j} \mathrm{D} \bar{\mathrm{X}}^j = (\mathbb{1} - \bar{\mathrm{X}})\mathrm{D}\left(\mathrm{D}^{-1} \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} \bar{\mathrm{X}}^{n-j} \mathrm{D} \bar{\mathrm{X}}^j\right),$$

via Section 3.2. ∎

DEFINITION 5.10. *When solving Volterra integral equations of the first kind with the method described in Section 3.3, it is useful to distinguish the operator without the weight $(1-x)$ which is to be inverted from the full Volterra operator. We will denote this operator $\tilde{\mathrm{V}}_K : \ell^2 \to \ell^2_1$, where*

$$(\mathbb{1} - \bar{\mathrm{X}})\tilde{\mathrm{V}}_K = \mathrm{V}_K.$$

*We furthermore see that*

$$\tilde{\mathrm{V}}_K = \mathrm{D}\left(\mathrm{D}^{-1} \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} \bar{\mathrm{X}}^{n-j} \mathrm{D} \bar{\mathrm{X}}^j\right).$$

*as an immediate corollary of Lemma 5.9.*

LEMMA 5.11. *$\tilde{\mathrm{V}}_K$ may be written as*

$$\tilde{\mathrm{V}}_K = \mathrm{D}(\mathrm{T}[f] + \mathcal{K}),$$

*where $\mathrm{T}[f]$ is a Toeplitz operator with symbol $f$ and $\mathcal{K}$ is compact. Furthermore, the symbol is uniquely determined by the coefficients of the polynomial kernel $K(x,y) = \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} x^{n-j} y^j$ to be*

$$f(z) = \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} \cos^{2n}\left(\frac{\theta}{2}\right) \qquad \text{where} \qquad z = \mathrm{e}^{\mathrm{i}\theta}.$$

*Proof.* From the Lemma 5.9 we see that the first statement is equivalent to the claim that

$$\sum_{n=0}^{M}\sum_{j=0}^{n} k_{nj}\mathrm{D}^{-1}\bar{\mathrm{X}}^{n-j}\mathrm{D}\bar{\mathrm{X}}^{j}$$

is of the form $\mathrm{T}+\mathcal{K}$ and thus asymptotically Toeplitz. To show this we need two observations: First, under sufficient continuity assumptions for the kernel, which are satisfied due to the kernel being polynomial, we have that

$$T[a]T[b] = T[ab] - H[a]H[\bar{b}], \tag{5.2}$$

and in particular

$$T[a]T[a] = T[a^2] - H[a]H[\bar{a}],$$

where $H[a]$, $H[\bar{a}]$ and $H[\bar{b}]$ are compact Hankel operators [10]. Thus any asymptotically Toeplitz operator (of sufficiently continuous symbol) raised to a finite power is again an asymptotically Toeplitz operator, as $(T+\mathcal{K})^2 = T^2 + T\mathcal{K} + \mathcal{K}T + \mathcal{K}^2$ and $T^2$ is again Toeplitz plus something compact via the above relation. The composition of bounded operators with compact operators is compact making $T\mathcal{K} + \mathcal{K}T + \mathcal{K}^2$ compact. An induction argument demonstrates that this is true for any power $n \in \mathbb{N}$. In particular, since it is known that $\bar{\mathrm{X}}$ is a compact perturbation of a Toeplitz operator [35] we know that $\bar{\mathrm{X}}^j$ is a compact perturbation of a Toeplitz operator as well. The second observation is that for the banded operator $\bar{\mathrm{X}}^{n-j}$, the operator $\mathrm{D}^{-1}\bar{\mathrm{X}}^{n-j}\mathrm{D}$ is also a compact perturbation of a Toeplitz operator and in fact we have that $\bar{\mathrm{X}}^{n-j}$ and $\mathrm{D}^{-1}\bar{\mathrm{X}}^{n-j}\mathrm{D}$ differ only in their compact part, i.e. have the same Toeplitz component. Via (5.2) we thus have that $\sum_{n=0}^{M}\sum_{j=0}^{n} k_{nj}\mathrm{D}^{-1}\bar{\mathrm{X}}^{n-j}\mathrm{D}\bar{\mathrm{X}}^{j}$ is of the form $(\mathrm{T}+\mathcal{K})$ and thus asymptotically Toeplitz.

Along with the above observations, Equation (5.2) tells us that we can compute the symbol of the Toeplitz part of a product of operators which are compact perturbations of Toeplitz operators if we know the symbols of the individual Toeplitz components. Due to bandedness it is straightforward to confirm that the symbol of the Toeplitz part of the multiplication operator $\bar{\mathrm{X}}$ is $(\frac{1}{2}+\frac{z}{4}+\frac{\bar{z}}{4}) = \cos^2\left(\frac{\theta}{2}\right)$ for the Jacobi polynomials $\tilde{\mathbf{P}}^{(1,0)}(x)$, which is thus also the symbol of the Toeplitz part of $\mathrm{D}^{-1}\bar{\mathrm{X}}\mathrm{D}$. Note at this point that

$$\left(\mathrm{D}^{-1}\bar{\mathrm{X}}\mathrm{D}\right)^{n-j} = \mathrm{D}^{-1}\bar{\mathrm{X}}^{n-j}\mathrm{D}$$

due to the outer operators cancelling. Given these tools as well as the linearity of the Fourier series it follows that the symbol of the Toeplitz part of the Volterra operator $\tilde{\mathrm{V}}_K$ is the linear combination

$$f(z) = \sum_{n=0}^{M}\sum_{j=0}^{n} k_{nj}\cos^{2n}\left(\frac{\theta}{2}\right).$$

THEOREM 5.12. *The method described in Section 3.3 converges for well-posed Volterra integral equations of the first kind with limits of integration* 0 *to* $1-x$

$$\mathrm{V}_K\mathbf{u} = \mathbf{g},$$

*rewritten as*

$$\tilde{\mathrm{V}}_K\mathbf{u} = \mathbf{q},$$

*with* $q(x) = \frac{g(x)}{1-x}$ *for a polynomial kernel* $K(x,y) \in L^2[T^2]$ *and with* $\mathbf{q} \in \ell_1^2$*, subject to the symbol of the Toeplitz part of* $\tilde{\mathrm{V}}_K$ *not vanishing on the complex unit circle. This condition is fulfilled if and only if* $\forall x \in [0,1] : K(x,x) \neq 0$.

*Proof.* The requirement $\mathbf{q} \in \ell_1^2$ arises formally due to the need to first invert D and can be understood as stemming from the inverse integration being a differentiation. The invertibility conditions of asymptotically Toeplitz operators of the form $(\mathrm{T} + \mathcal{K})$ are known in the literature (see e.g. [25, 11] and the references therein): A compactly perturbed Toeplitz operator on $\ell^2$ is invertible if it is a Fredholm operator, its index is 0 and it has a trivial kernel [24, 11, 25]. Furthermore, a compactly perturbed Toeplitz operator is Fredholm if its symbol (which is just the symbol of the Toeplitz part) does not vanish anywhere on the complex unit circle.

In general, it holds that the index of a Toeplitz operator which is Fredholm is the sign-flipped winding number of its symbol on the complex unit disk [11]. Since the symbol of the Toeplitz part of the unweighted Volterra operator is real-valued and continuous its index is thus 0 if and only if it does not vanish anywhere on the complex unit circle, which is a necessary condition for it to be Fredholm in the first place. Since $\cos^2\left(\frac{\theta}{2}\right) \in [0,1]$, the symbol vanishes at some point $\theta \in [0, 2\pi]$, i.e.

$$\sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} \cos^{2n}\left(\frac{\theta}{2}\right) = 0,$$

if and only if for some $x \in [0,1]$ we have

$$\sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} x^n = 0.$$

This in turn is precisely the condition that $K(x,x) = 0$, since

$$K(x,y) = \sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} x^{n-j} y^j.$$

Conversely, if $\forall x \in [0,1] : K(x,x) \neq 0$ then the Volterra operator is Fredholm because the symbol of its Toeplitz part has no roots on the unit circle and as this symbol is real valued its winding number and thus index is 0. This necessary condition for invertibility of the operator becomes a sufficient condition if in addition to this we have $\ker(\mathrm{T} + \mathcal{K}) = \{0\}$, as this yields injectivity and via the index formula [11]:

$$\mathrm{ind}(\mathrm{T}) = \mathrm{ind}(\mathrm{T} + \mathcal{K}) := \dim(\ker(\mathrm{T} + \mathcal{K})) - \dim(\mathrm{coker}(\mathrm{T} + \mathcal{K})),$$

with $\mathrm{ind}(\mathrm{T} + \mathcal{K}) = 0$ also implies surjectivity. $\ker(\mathrm{T} + \mathcal{K}) = \{0\}$ is a consequence of the classical result that the Volterra integral operator has no non-zero eigenvalues. The convergence of the method is then a consequence of known results in the theory of finite section methods, see e.g. [25]. ∎

**Remark**: The motivation for solving $\tilde{\mathrm{V}}_K \mathbf{u} = \mathbf{q}$ with $q(x) = \frac{g(x)}{1-x}$ instead of $\mathrm{V}_K \mathbf{u} = \mathbf{g}$ directly can be understood at this point, since for $\mathrm{V}_K$ the symbol of the Toeplitz part is instead found to be

$$\sum_{n=0}^{M} \sum_{j=0}^{n} k_{nj} \sin\left(\frac{\theta}{2}\right) \cos^{2n}\left(\frac{\theta}{2}\right),$$

which always has a root on the complex unit circle at $\theta = 0$ and thus its induced Toeplitz operator is not Fredholm and not invertible. Therefore the presented proof strategy only succeeds if $q(x) = \frac{g(x)}{1-x}$ may be used instead to get rid of the additional

sine terms. The symbol of the Toeplitz part of $\tilde{\mathrm{V}}_K$ is comparably very well-behaved for a variety of kernels.

So far we have only been working with polynomial kernels of order $M$, henceforth denoted $K_M$, when it comes to Volterra equations of the first kind. We will need the following theorem (see [4, 48]) which we restate without proof for the extension of the above arguments to a non-polynomial kernel:

THEOREM 5.13. *Let $X$ and $Y$ be normed linear spaces with one or both being Banach spaces and let $\mathcal{T} : X \to Y$ be a bounded and invertible operator with $\mathcal{T}^{-1} : Y \to X$. Then if the bounded operator $\mathcal{M} : X \to Y$ satisfies*

$$\|\mathcal{M} - \mathcal{T}\| < \frac{1}{\|\mathcal{T}^{-1}\|},$$

*it follows that $\mathcal{M}$ is also invertible with bounded inverse operator $\mathcal{M}^{-1} : Y \to X$ and*

$$\|\mathcal{M}^{-1}\| \leqslant \frac{\|\mathcal{T}^{-1}\|}{1 - \|\mathcal{T}^{-1}\|\|\mathcal{T} - \mathcal{M}\|},$$

$$\|\mathcal{M}^{-1} - \mathcal{T}^{-1}\| \leqslant \frac{\|\mathcal{T}^{-1}\|^2\|\mathcal{T} - \mathcal{M}\|}{1 - \|\mathcal{T}^{-1}\|\|\mathcal{T} - \mathcal{M}\|}.$$

LEMMA 5.14. *Given that*

$$\|\tilde{\mathrm{V}}_{K_M} - \tilde{\mathrm{V}}_K\| \xrightarrow[M\to\infty]{} 0$$

*for a sequence of Volterra operators induced by polynomial kernels $K_M(x, y)$ and a not necessarily polynomial kernel $K(x, y)$, we have*

$$\|\mathbf{u}_M - \mathbf{u}\| \xrightarrow[M\to\infty]{} 0,$$

*where $\mathbf{u}_M$ is the solution to the approximated problem*

$$\tilde{\mathrm{V}}_{K_M}\mathbf{u}_M = \mathbf{q}.$$

*Proof.* The method can be extended to more general $K = K(x, y)$ if $K_M$ is interpreted as the polynomial approximation of order $M$ of the full kernel $K$. To show that the method can be extended sensibly to non-polynomial kernels what remains to be shown is that $\|\mathbf{u}_M - \mathbf{u}\| \xrightarrow[M\to\infty]{} 0$. This can be achieved by use of Theorem 5.13: The assumptions of the theorem are satisfied when setting $\mathcal{T} = \tilde{\mathrm{V}}_K$ and $\mathcal{M} = \tilde{\mathrm{V}}_{K_M}$ since if $\|\tilde{\mathrm{V}}_{K_M} - \tilde{\mathrm{V}}_K\| \xrightarrow[M\to\infty]{} 0$ then for some $M$ all subsequent $\tilde{\mathrm{V}}_{K_M}$ satisfy

$$\|\tilde{\mathrm{V}}_{K_M} - \tilde{\mathrm{V}}_K\| < \frac{1}{\|\tilde{\mathrm{V}}_K^{-1}\|}.$$

This immediately yields invertibility of $\tilde{\mathrm{V}}_{K_M}$ and more importantly the desired result that

$$\|\tilde{\mathrm{V}}_{K_M}^{-1} - \tilde{\mathrm{V}}_K^{-1}\| < \frac{\|\tilde{\mathrm{V}}^{-1}\|^2\|\tilde{\mathrm{V}}_{K_M} - \tilde{\mathrm{V}}_K\|}{1 - \|\tilde{\mathrm{V}}^{-1}\|\|\tilde{\mathrm{V}}_{K_M} - \tilde{\mathrm{V}}_K\|} \xrightarrow[M\to\infty]{} 0$$

which justifies calling the solution $\mathbf{u}_M = \tilde{\mathrm{V}}_{K_M}^{-1}\mathbf{q}$ an approximation to $\mathbf{u} = \tilde{\mathrm{V}}_K^{-1}\mathbf{q}$. ∎

**6. Discussion.** The method proposed in this paper can efficiently compute Volterra integrals as well as solve Volterra integral equations of the first and second kind with high accuracy using bivariate orthogonal polynomials to resolve the kernel along with an operator valued Clenshaw algorithm and is not restricted to convolution kernels. Numerical experiments suggest it can even be applicable to certain singular equations. Our approach takes advantage of the sparsity of the required integration and extension operators which are due to the symmetries of the Jacobi polynomial basis on the triangle domain. The method was shown to converge for well-posed Volterra integral equations of the first and second kind, using a link to compact perturbations of Toeplitz operators.

Extensions of this approach to various so-called integro-differential equations of Volterra-type, where both differentiation and Volterra operators act on the unknown function, as well as extensions to non-linear Volterra equations, where the unknown function can appear in non-linear fashion in the Volterra integral, while non-trivial are conceivable and will be addressed in future work.

**Acknowledgments.** We thank Nick Hale and Kuan Xu for crucial help on Volterra integral equations at the initial stages of this project, Mikael Slevinsky for thoroughly reading a draft and providing detailed comments and the anonymous reviewers for their useful comments and suggestions. The second author was supported by a Leverhulme Trust Research Project Grant RPG-2019-144.

## REFERENCES

[1] A. Akyüz-Daşcioğlu, *A Chebyshev polynomial approach for linear Fredholm–Volterra integro-differential equations in the most general form*, Appl. Math. Comput., 181 (2006), pp. 103–112, https://doi.org/10.1016/j.amc.2006.01.018.

[2] S. S. Allaei, Z. Yang, and H. Brunner, *Existence, uniqueness and regularity of solutions to a class of third-kind Volterra integral equations*, J. Integral Equ. Appl., 27 (2015), pp. 325–342, https://doi.org/10.1216/JIE-2015-27-3-325.

[3] S. S. Allaei, Z. Yang, and H. Brunner, *Collocation methods for third-kind VIEs*, IMA J. Num. Ana., 37 (2017), pp. 1104–1124, https://doi.org/10.1093/imanum/drw033.

[4] K. E. Atkinson and W. Han, *Theoretical Numerical analysis: A Functional Analysis Framework*, no. 39 in Texts in applied mathematics, Springer, Dordrecht ; New York, 3rd ed ed., 2009.

[5] E. Babolian and Z. Masouri, *Direct method to solve Volterra integral equation of the first kind using operational matrix with block-pulse functions*, J. Comput. Appl. Math., 220 (2008), pp. 51–57, https://doi.org/10.1016/j.cam.2007.07.029.

[6] G. Bachman and L. Narici, *Functional Analysis*, Dover Publications, Mineola, N.Y, 2000.

[7] P. Baratella, *A Nyström interpolant for some weakly singular linear Volterra integral equations*, J. Comput. Appl. Math., 231 (2009), pp. 725–734, https://doi.org/10.1016/j.cam.2009.04.007.

[8] Z. Battles and L. N. Trefethen, *An extension of MATLAB to continuous functions and operators*, SIAM J. Sci. Comput., 25 (2004), https://doi.org/10.1137/S1064827503430126.

[9] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM Rev., 59 (2017), pp. 65–98, https://doi.org/10.1137/141000671.

[10] A. Böttcher and B. Silbermann, *Introduction to Large Truncated Toeplitz Matrices*, Springer, New York, 1999.

[11] A. Böttcher, B. Silbermann, and A. Karlovich, *Analysis of Toeplitz Operators*, Springer monographs in mathematics, Springer, Berlin, 2. ed ed., 2006.

[12] H. Brunner, *Collocation Methods for Volterra Integral and Related Functional Differential Equations*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2004, https://doi.org/10.1017/CBO9780511543234.

[13] H. Brunner, *Volterra Integral Equations: An Introduction to Theory and Applications*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2017, https://doi.org/10.1017/9781316162491.

[14] P. G. Casazza and G. Kutyniok, eds., *Finite Frames: Theory and Applications*, Applied and

numerical harmonic analysis, Springer, New York, 2013.
[15] O. Christensen, *An Introduction to Frames and Riesz Bases*, Applied and Numerical Harmonic Analysis, Birkhäuser Boston, Boston, MA, 2003, https://doi.org/10.1007/978-0-8176-8224-8.
[16] C. W. Clenshaw, *A note on the summation of Chebyshev series*, Math. Comput., 9 (1955), pp. 118–118, https://doi.org/10.1090/S0025-5718-1955-0071856-0.
[17] T. Diogo, N. J. Ford, P. Lima, and S. Valtchev, *Numerical methods for a Volterra integral equation with non-smooth solutions*, J. Comput. Appl. Math., 189 (2006), pp. 412–423, https://doi.org/10.1016/j.cam.2005.10.019.
[18] T. Diogo, N. Franco, and P. Lima, *High order product integration methods for a Volterra integral equation with logarithmic singular kernel*, Comm. P. and Appl. Ana., 3 (2004), pp. 217–235, https://doi.org/10.3934/cpaa.2004.3.217.
[19] T. Diogo, S. Mckee, and T. Tang, *A Hermite-type collocation method for the solution of an integral equation with a certain weakly singular kernel*, IMA J. Num. Ana., 11 (1991), pp. 595–605, https://doi.org/10.1093/imanum/11.4.595.
[20] T. A. Driscoll, *Automatic spectral collocation for integral, integro-differential, and integrally reformulated differential equations*, J. Comput. Phys., 229 (2010), https://doi.org/10.1016/j.jcp.2010.04.029.
[21] T. A. Driscoll, F. Bornemann, and L. N. Trefethen, *The chebop system for automatic solution of differential equations*, BIT Numerical Mathematics, 48 (2008), https://doi.org/10.1007/s10543-008-0198-4.
[22] C. F. Dunkl and Y. Xu, *Orthogonal Polynomials of Several Variables*, no. 155 in Encyclopedia of mathematics and its applications, Cambridge University Press, Cambridge, second ed., 2014.
[23] W. Gautschi, *Orthogonal Polynomials: Computation and Approximation*, Numerical mathematics and scientific computation, Oxford University Press, Oxford ; New York, 2004.
[24] J. J. Grobler, L. E. Labuschagne, and M. Möller, eds., *Operator Algebras, Operator Theory and Applications*, Birkhäuser Basel, Basel, 2010, https://doi.org/10.1007/978-3-0346-0174-0.
[25] R. Hagen, S. Roch, and B. Silbermann, *C*-Algebras and Numerical Analysis*, no. 236 in Chapman & Hall/CRC Pure and Applied Mathematics, CRC Press, Taylor & Francis Group, New York, 2001.
[26] N. Hale, *An ultraspherical spectral method for linear Fredholm and Volterra integro-differential equations of convolution type*, IMA J. Num. Ana., (2018), https://doi.org/10.1093/imanum/dry042.
[27] H. Köroğlu, *Chebyshev series solution of linear Fredholm integrodifferential equations*, Int. J. Math. Educ. Sci. Technol., 29 (1998), pp. 489–500, https://doi.org/10.1080/0020739980290403.
[28] D. O. Krimer, S. Putz, J. Majer, and S. Rotter, *Non-Markovian dynamics of a single-mode cavity strongly coupled to an inhomogeneously broadened spin ensemble*, Phys. Rev. A, 90 (2014), https://doi.org/10.1103/PhysRevA.90.043852.
[29] D. O. Krimer, M. Zens, S. Putz, and S. Rotter, *Sustained photon pulse revivals from inhomogeneously broadened spin ensembles*, Laser Photonics Rev., 10 (2016), pp. 1023–1030, https://doi.org/10.1002/lpor.201600189.
[30] J. Lindenstrauss and L. Tzafriri, *Classical Banach Spaces*, Classics in mathematics, Springer, Berlin, 1996.
[31] S. K. Lintner and O. P. Bruno, *A generalized Calderón formula for open-arc diffraction problems: theoretical considerations*, P. Roy. Soc. Edinb. A, 145 (2015), pp. 331–364, https://doi.org/10.1017/S0308210512000807.
[32] A. Loureiro and K. Xu, *Volterra-type convolution of classical polynomials*, Math. Comput., (2019).
[33] K. Maleknejad and N. Aghazadeh, *Numerical solution of Volterra integral equations of the second kind with convolution kernel by using Taylor-series expansion method*, Appl. Math. Comput., 161 (2005), pp. 915–922, https://doi.org/10.1016/j.amc.2003.12.075.
[34] J. Muscat, *Functional Analysis An Introduction to Metric Spaces, Hilbert Spaces, and Banach Algebras*, Springer International Publishing, Cham, 2014.
[35] F. Olver, A. Daalhuis, D. Lozier, B. Schneider, R. Boisvert, C. Clark, B. Miller, and B. V. Saunders (eds.), *NIST Digital Library of Mathematical Functions*, Dec. 2018. http://dlmf.nist.gov.
[36] S. Olver and A. Townsend, *A fast and well-conditioned spectral method*, SIAM Rev., 55 (2013), pp. 462–489, https://doi.org/10.1137/120865458.
[37] S. Olver and A. Townsend, *A practical framework for infinite-dimensional linear algebra*, in

2014 First Workshop for High Performance Technical Computing in Dynamic Languages, LA, USA, Nov. 2014, IEEE, pp. 57–62, https://doi.org/10.1109/HPTCDL.2014.10.

[38] S. Olver, A. Townsend, and G. Vasil, *Recurrence relations for orthogonal polynomials on a triangle*, in ICOSAHOM 2018, 2019.

[39] S. Olver, A. Townsend, and G. Vasil, *A sparse spectral method on triangles*, SIAM J. Sci. Comput., 41 (2019), pp. A3728–A3756, https://doi.org/10.1137/19M1245888.

[40] R. Pachon, R. B. Platte, and L. N. Trefethen, *Piecewise-smooth chebfuns*, IMA J. Num. Ana., 30 (2010), https://doi.org/10.1093/imanum/drp008.

[41] J. Prüss, *Evolutionary Integral Equations and Applications*, Modern Birkhäuser classics, Springer, Basel ; New York, 2012.

[42] R. M. Slevinsky, *Conquering the pre-computation in two-dimensional harmonic polynomial transforms*, arXiv:1711.07866, (2017), http://arxiv.org/abs/1711.07866 (accessed 2019-01-11TZ).

[43] R. M. Slevinsky, *Fast and backward stable transforms between spherical harmonic expansions and bivariate Fourier series*, Appl. Comput. Harmon. Anal., (2017), https://doi.org/10.1016/j.acha.2017.11.001.

[44] R. M. Slevinsky, *FastTransforms v0.1.1*, Jan. 2019, https://github.com/MikaelSlevinsky/FastTransforms (accessed 2019-01-11TZ).

[45] R. M. Slevinsky and S. Olver, *A fast and well-conditioned spectral method for singular integral equations*, J. Comput. Phys., 332 (2017), pp. 290–315, https://doi.org/10.1016/j.jcp.2016.12.009.

[46] H. Song, Z. Yang, and H. Brunner, *Analysis of collocation methods for nonlinear Volterra integral equations of the third kind*, Calcolo, 56 (2019), p. 7, https://doi.org/10.1007/s10092-019-0304-9.

[47] A. Townsend and S. Olver, *The automatic solution of partial differential equations using a global spectral method*, J. Comput. Phys., 299 (2015), pp. 106–123, https://doi.org/10.1016/j.jcp.2015.06.031.

[48] T. Trogdon and S. Olver, *Riemann-Hilbert Problems, Their Numerical Solution, and the Computation of Nonlinear Special Functions*, no. 146 in Other titles in applied mathematics, SIAM, Society for Industrial and Applied Mathematics, Philadelphia, 2016.

[49] F. van den Bosch, J. A. J. Metz, and J. C. Zadoks, *Pandemics of focal plant disease, a model*, Phytopathology, 89 (1999), pp. 495–505, https://doi.org/10.1094/PHYTO.1999.89.6.495.

[50] G. M. Vasil, K. J. Burns, D. Lecoanet, S. Olver, B. P. Brown, and J. S. Oishi, *Tensor calculus in polar coordinates using Jacobi polynomials*, J. Comput. Phys., 325 (2016), pp. 53–73, https://doi.org/10.1016/j.jcp.2016.08.013.

[51] A. Wazwaz, *Linear and Nonlinear Integral Equations: Methods and Applications*, Higher Education Press; Springer, Beijing, Heidelberg, New York, 2011.

[52] A. Wazwaz and R. Rach, *Two reliable methods for solving the Volterra integral equation with a weakly singular kernel*, J. Comput. Appl. Math., 302 (2016), pp. 71–80, https://doi.org/10.1016/j.cam.2016.02.004.

[53] K. Xu, A. P. Austin, and K. Wei, *A fast algorithm for the convolution of functions with compact support using Fourier extensions*, SIAM J. Sci. Comput., 39 (2017), pp. A3089–A3106, https://doi.org/10.1137/17M1114764.

[54] K. Xu and A. Loureiro, *Spectral approximation of convolution operators*, SIAM J. Sci. Comput., 40 (2018), pp. A2336–A2355, https://doi.org/10.1137/17M1149249.