Numerical simulation of the non-linear Schrödinger equation
A comparative study based on the summation-by-parts method in space and time

Ibrohim Hamoud
Abstract

In this work, we present the numerical implementation of various time marching methods based on the summation-by-parts (SBP) method for the time integration of nonlinear dispersive systems. More specifically, we consider the nonlinear Schrödinger (NLS) equation that accepts soliton or soliton-like solution as a primary case study. The spatial discretization is performed via the SBP method with the projection technique to enforce the boundary conditions strongly. We show that the method in combination with the projection technique represents a robust, accurate, and highly stable numerical method for the treatment of Nonlinear Schrödinger-type systems. The imposition of the initial data for the SBP-based time marching methods can, equivalently to SBP-spatial discretization, be combined with either the simultaneous approximation term (SAT) or the projection technique. In contrast to using the SAT for the imposition of the initial data, the projection technique represents a part of recent progress in the field and has not been studied in the literature. To the best of our knowledge, no investigation of the numerical efficiency of the SBP-based time marching methods has been conducted yet in the context of the nonlinear initial boundary value problem (IBVP). Therefore, we present in this work a thorough numerical efficiency analysis for the NLS-type systems using both the SAT and the projection technique with the classical fourth-order Runge-Kutta as a standard of comparison.
Popular Scientific Summary

Nonlinear wave theory, characterized by phenomena modeled through nonlinear partial differential equations (PDEs), has been a cornerstone in the study of diverse scientific domains, particularly in soliton theory. The nonlinear Schrödinger (NLS) equation, a prominent system arising in various physical settings, holds extensive applications in quantum mechanics, solid-state physics, condensed matter physics, nonlinear optics, and more. Therefore, we consider NLS-type systems as a primary test case study for the implementation of Summation-by-parts (SBP) in space and time.

This project is centered around the numerical implementation of NLS-type equations using the SBP-projection approximation method for spatial discretization. This method is known for its high accuracy and optimal stability properties. In addition, we employ SBP-based time marching methods as time integrators. NLS is a weakly dispersive system that requires very accurate time integration and will lead to very small time steps when using traditional explicit Runge-Kutta (RK) methods. Therefore, SBP in time is a good candidate for an efficient time-discretization for such systems. Therefore, we present in this study a detailed exploration of different SBP-time marching methods in the context of numerical efficiency. In addition, we consider two ways of imposing the initial boundary condition; one by combining the SBP-based time marching method with the simultaneous approximation term (SAT) and the other by using the projection technique.

The numerical results in this work show that the SBP-projection approximation method for spatial discretization represents a robust, accurate, and highly stable numerical method for solving NLS-type systems. A comparative study of the numerical efficiency of the SBP-time marching methods shows that these methods are generally far more numerically efficient compared to the classical fourth-order RK. The Legendre-Gauss-Lobatto-based time marching method in combination with the projection technique was found to be by far the most numerically efficient time integrator. These findings are supported by numerical results in both one and two-dimensional test cases.

One of the key findings was that the weak nonlinearity in the NLS-type systems can be exploited for further approximating the the large nonlinear system of equations that arises from the temporal discretization. This approximation adds a considerable improvement to the numerical efficiency of the various SBP-based time marching methods employed in this work.
Acknowledgment

I would like to thank my supervisor Ken Mattsson for providing me with the opportunity to pursue such an interesting work. Thanks for all the guidance, editing, proofreading, providing the necessary material, and directing my work in the correct direction.

I would also thank Jörn Zimmerling for the collaboration and for providing me with valuable discussions that helped me to go through my work, especially in the part related to iterative solvers.

I would like also to thank all my new friends I met during my thesis time; Jiaying, Golam, Axel, George, and Mikolai who made my thesis semester a happy and fun time. Special thanks to George for all the intellectually stimulating "fika breaks".
Acronyms

PDEPartial differential equation
ODEordinary differential equation
IBVPinitial boundary value problem
NLSNonlinear Schrödinger
KdVKorteweg-de Vries
BECBose-Einstein condensates
RKRunge-Kutta
RK4Classical Runge-Kutta
IRKImplicit Runge-Kutta
ESDIRKExplicit singly diagonally implicit Runge–Kutta
ERKExplicit Runge–Kutta
BDFbackward differentiation
IMEXImplicit-explicit time integration
SBPSummation by parts
SBPn n-order Summation by parts
SATsimultaneous approximation term
GSBPGeneralized SBP
SBP-PSBP with the projection technique
SBP-SATSBP with SAT technique
GMRESGeneralized Minimal Residual Method
FGMRESFlexible GMRES
DCFDefect correction method
LGLLegendre-Gauss-Lobatto quadrature
LGLegendre-Gauss quadrature
DI SBP-time marching method based on diagonally implicit Runge-Kutta
MMSMethod of manufactured solution
Notations

For clarity and avoiding confusion, the following labeling system is adopted throughout this work,

\( A \) denotes a scalar labeled by one letter.

\([AB]\) denotes a scalar labeled by two letters while \( AB \) denotes multiplication of two scalars \( A \) and \( B \).

\( \mathbf{A} \) denotes a vector \([A_1, A_2, \ldots]^T\) labeled by one letter.

\([\mathbf{AB}]\) denotes a vector \([\mathbf{AB}_1, \mathbf{AB}_2, \ldots]\) labeled by two letters while \( \mathbf{AB} \) denotes vector-vector multiplication between the vectors \( \mathbf{A} \) and \( \mathbf{B} \).

\( \mathbf{A} \) denotes a matrix \([A_1, A_2, \ldots]^T\) labeled by one letter.

\([\mathbf{A}\mathbf{B}]\) denotes a matrix \([\mathbf{AB}_1, \mathbf{AB}_2, \ldots]\) labeled by two letters while \( \mathbf{A}\mathbf{B} \) denotes matrix-matrix multiplication between the matrices \( \mathbf{A} \) and \( \mathbf{B} \).
1 Introduction

Nonlinear wave theory evoked substantial interest over the years and has been a key element in the study of various nonlinear phenomena such as in the field of soliton theory. Many of these phenomena can be described by nonlinear partial differential equations (PDEs). Among the most studied typical nonlinear equations are the Korteweg-de Vries (KdV) equation, the Sasa-Satsuma equation, and the nonlinear Schrödinger (NLS) equation, which can describe various nonlinear wave phenomena in dispersive physical media. The ubiquity of these equations is demonstrated by the prevalence of the soliton concept in diverse scientific domains. The NLS equation is an example of integrable systems of nonlinear PDE that arise in a vast number of physical settings. It admits a wide range of applications and appears in many branches of present-day physics and mathematics, e.g. quantum mechanics, solid state physics, condensed matter physics, quantum chemistry, nonlinear optics, wave propagation, optical communication, protein folding and bending, semiconductor industry, laser propagation, and nanotechnology. To some extent, most systems that are characterized by weak nonlinearity, dispersivity, and energy preservation can lead, in an appropriate limit, to the NLS equation [1]

The scope and extent of influence of the NLS equation have grown tremendously in the past decades as well as the vast amount of literature on the topic. The Schrödinger equation was first proposed as a model for many-body problems in physics characterized by high dimensionality rendering it impossible to solve. After the assumption of Hatree-Fork ansatz, the nonlinear Schrödinger equation emerged as a dimensionality reduction of the linear version [2]. NLS models the dynamic of slowly varying envelop of a weakly non-linear wave packet in a dispersive media. The NLS can be used to model different aspects of wave packet dynamics depending on how the variables are interpreted physically.

One of the most relevant areas where the NLS equation arises is Bose-Einstein condensate (BEC) and plasma physics [2]. Condensate atoms’ interaction can be modeled by a mean-field where the atom feels the additional potential of all other atoms’ presence proportionally to its local atomic density. This potential can be included in the Schrödinger equation to account for atomic one-to-one interaction. The outcome is an NLS-type equation, dubbed, Gross-Pitaevskii equation [3]. BEC has long been considered a pillar in the study of macroscopic quantum phenomena e.g. in the theory of superconductivity [4] and the theory of superfluidity [5]. Thus, the realization of BEC has opened since then new avenues for the study of NLS-type equations [6, 7].

Recently, a renewed interest and attention in NLS emerged, mainly due to the latest developments in nonlinear optics [8, 9] and soft-condensed matter physics [10]. The NLS equation has shown to be of particular significance from the point of view of nonlinear optics with a strong presence in the study of beam self-cleaning phenomenon [11] e.g. self-channeling of an intense ultrashort laser pulse in a plasma [12], electromagnetic fields [13] and electric field in optical fibers.

2
In condensed matter theory, the NLS equation is involved in the study of various phenomena, to mention a few, ferromagnets [16], magnons [17], and modeling lattice dynamics [18, 19].

NLS-type equations possess soliton solutions in various forms. Considering the proliferation of NLS-type equations in various scientific disciplines, the solutions of NLS equations brought a massive investigation into soliton behavior and the understanding of it. In one dimension, the soliton can be thought of as an undeformed propagating localized wave pulse brought by a nonlinear effect that completely compensates for the dispersion. For a brief introduction to solitons and their presence in various fields of science consult [20, 21, 22]. NLS equation admits a variety of solitons’ solutions i.e. Bose-Einstein soliton [23], optical solitons [24, 25, 26] and lattice soliton [27]. An optical soliton solution of the NLS has even been observed experimentally [28].

A major area of application of the NLS lies in modulating the behavior of water waves. Historically, the Korteweg-de Vries (KdV) equation, an equation for unidirectional, long, nonlinear dispersive wave, was initially derived to describe shallow water waves [29] and later found relevance in the description of plasma waves [30]. Interestingly, the connection between the NLS and the KdV equation has been established and investigated for a long time, cf. [31]. The NLS can be derived from the KdV equation as a modulation equation and its solution as an approximation for the one of the KdV [32, 33]. The resultant NLS equation describes small-amplitude, and narrow bandwidth wave pocket’s motion in shallow water [34, 35].

In addition to the NLS equation and the context of shallow water waves, well-known weakly non-linear models include the fifth-order KdV equation and the Kadomtsev-Petviashvili equation [36] in the presence of surface tension, and the Boussinesq-type equations in the absence of surface tension [37]. Further, the NLS is present in the behavior of deep water waves and freak waves in the ocean [38]. Water wave modulation is associated with the wave-induced current, and the NLS equation is thereby most accurately applicable for certain cases e.g. the study of modulation instability and the mechanism of rogue wave generation [39, 40], and deep water wavetrains [41]. Notably, rogue waves were also reported in other physical models that are affiliated with the NLS equation such as in optical systems [42, 43].

This work aims to solve NLS-type equations numerically using the Summation-by-parts (SBP)-projection approximation method. High accuracy and optimal stability properties characterize this technique. The method is based on previous work for spatial discretizations using high-order finite difference formulae on SBP. For derivations and elaborated introduction refer to e.g. [44, 45, 46, 47, 48, 49]. Stable boundary treatment is obtained by combining SBP approximation with the projection approach to impose the boundary conditions, cf. [50, 51, 52]. To guarantee a stable and highly accurate SBP approximation scheme, it is necessary to enforce the boundary condition such that the solution is bound by the initial data. SBP property [53]
The focus of this work is to deploy the SBP schemes as a time integrator method for initial boundary value problems. In the recent decade, extensive work was made in the literature concerning the SBP-time marching methods, for an elaborate introduction cf. [54, 55, 56, 57]. These methods are global, i.e. they require solving a large system of equations for all time points simultaneously. The initial condition can be imposed weekly using the simultaneous approximation term (SAT) method [58, 59, 60]. SBP-SAT in time methods are fully implicit time-marching methods and can be understood as a subclass of IRK methods and constructed as IRK schemes. On the contrary, not every IRK method can be constructed as an SBP-time marching scheme [61, 62, 63]. Owning to the association with IRK schemes, SBP-time marching methods possess profitable stability properties e.g. A-, B-, L- algebraic and energy stability [64]. These methods are especially favorable when accommodated with an energy-stable semi-discrete approximation ( spatial discretization via the SBP projection, for example). The combination elicits an optimally sharp fully discrete energy estimate and thus the primal justification for the SBP-time approach. Recently, a new SBP-time integration scheme was proposed, accompanied by a projection method to impose the initial boundary strongly [65]. The new scheme is A-stable and can be as well characterized by the IRK method. However, the scheme is not necessarily L- or B-stable.

The objective of this study is to present SBP-based time integrators for the NLS equation and NLS-type equation with soliton solutions. We investigate the numerical efficiency of the new methods in comparison to popular time integration schemes such as the classical RK4 or IMEX as a benchmark for numerical efficiency. In Section 2 we present the NLS-type equation as a case study with concepts and characteristics that are relevant to time integration. In Section 3, a set of definitions and concepts are introduced in the context of the SBP-projection spatial discretization of the NLS equation with a proof of energy stability on the semi-discrete problem. We closely examine the treatment of boundary conditions using the projection method which is the focus of the spatial discretization in this work. In Section 4, the SBP-based time integration methods are introduced and derived. A brief overview of the implementation of the SBP-based time integration schemes is given in Section 5. In Section 6, the integration schemes are verified for one-dimensional test cases accompanied by analytical solutions. The numerical efficiency of the different time integration schemes is also explored. Extensions of the implementation to two-dimensional test cases are given in Section 7. Section 8 summarizes the manuscript and provides a thorough results analysis and an outlook for future work.

2 Nonlinear Schrödinger equation

The focus of this work revolves around the soliton dynamics in the NLS-type equation as a test model for spatial discretization using the SBP-projection technique and various schemes for time
integration. This section provides a brief outline of the NLS-type equations and their attributes. The equation in one-dimension reads

\[ i \frac{\partial}{\partial t} u(x, t) = -\frac{\partial^2}{\partial x^2} u(x, t) + F(u(x, t)). \]  

(1)

With cubic nonlinearity \( F(u(x, t)) = \beta |u(x, t)|^2 u(x, t) \), the equation, in the broadest sense, models the slow evolution of an envelope of a fast oscillating wave packet in a nonlinear medium. Here, \( \beta \) is a dimensionless constant that controls the ratio of dispersive effects to nonlinear effects. Further, considering cubic nonlinearity, the system is completely integrable. Integrability of a dynamic system means here possessing an invariant under evolution and it is closely associated with having a closed-form solution and being "in principle" solvable analytically.

We note that the NLS equation (1) with cubic nonlinearity is referred to as focusing if \( \beta > 0 \), and as defocusing if \( \beta < 0 \). These nomenclatures are appropriated from electromagnetic propagation in optical media and it refers to the increase or decrease in the amplitude of the probe beam. The focusing NLS equation admits soliton solutions that decay at infinity, often designated as bright solitons. On the other hand, the defocusing NLS can permit soliton solutions, dubbed "dark solitons", that do not vanish at infinity due to high oscillatory background intensity. For a discussion of dark soliton consult [66] and references therein. For these reasons, we only consider the focusing form of the NLS equation and NLS-related systems.

For nonlinear dispersive problems, the SBP discretization may become inaccurate or even unstable due to spurious high-frequent oscillations in the solution. The SBP discretization is carried out in this work with standard SBP operators; introduced in the following section, which do not possess a sampling mechanism for spurious oscillations. These oscillations are caused by high-frequency components or due to discontinuity in the solution and might lead to a solution blow-up if the time integrator fails to eliminate them. An explicit approach to time integration often fails to capture the propagation of these waves without demanding time step restriction, thus leading to numerical instability. The nonlinearity of the NLS equation can lead to a strong gradient and the emergence of rapid oscillation zones, dubbed dispersive shock waves. This can arise in i.e. the semi-classical limit of the NLS-equation which is provided as a test example for numerical experimentation in this work. More details regarding nonlinear dispersive problems and dispersive oscillation can be found in [67].

3 Summation-by-parts and the spatial discretization

Summation-by-parts operators (SBP) are essentially a class of operators that relate to finite difference schemes. Diagonal norm SBP operators of order \( p \) are constructed mainly from central difference interior point operators, where \( p \) refers to the order of accuracy of the underlying scheme
SBP operators with non-central difference schemes were also presented in [68]). While a $p$th-order diagonal norm SBP-operator implies a $p$th-order of accuracy in the interior, the boundary accuracy is restricted to $\frac{p}{2}$th-order when $p$ is even and $\frac{(p-1)}{2}$th-order when $p$ is odd.

The appealing aspect of SBP operators is their SBP property which makes them a well-suited and operable high-order discretization method for PDEs. To prove the stability of the continuous problem, the energy method provides a passably straightforward approach whose procedure is mimicked by the SBP operators for the discrete problem. The projection method enforces the boundary such that the SBP property is maintained and hence the discrete energy estimate can be attained. The SBP-projection approximation method provides a verifiable stable method in the sense that the solution’s growth is bound. In this section, we introduce the relevant concepts that underpin the SBP projection method in the context of one-dimensional NLS equation

$$\frac{\partial}{\partial t} u(x, t) = i\frac{\partial^2}{\partial x^2} u(x, t) + i\beta |u(x, t)|^2 u(x, t)$$

(2)

**Definition 1.** Let the inner product for real-valued functions $u_1, u_2 \in L^2[x_l, x_r]$ be defined by $(u_1, u_2) = \int_{x_l}^{x_r} u_1 u_2 \, dx$ and the correspondent norm is given by $||u||^2 = (u, u)$.

**Definition 2.** Problem (2) is regarded as well posed problem if $||u(x, t)||^2 \leq e^{\eta t} ||u(x, t = 0)||$ or equivalently $\frac{\partial}{\partial t} ||u(x, t)||^2 \leq 0$.

The energy estimate for (2) is obtained directly from the energy method by taking the inner product $(u, \frac{\partial}{\partial t} u)$, and consequently $\frac{\partial}{\partial t} ||u||^2 = (u, \frac{\partial}{\partial t} u) + (\frac{\partial}{\partial t} u, u)$ as follows

$$\left(\frac{\partial}{\partial t} u, u\right) = \left(u, i\frac{\partial^2}{\partial x^2} u\right) + (u, i\beta |u|^2 u) = -(\frac{\partial}{\partial x} u, \frac{i}{2} \frac{\partial}{\partial x} u) + (u, i\beta |u|^2 u) + iu^* \frac{\partial}{\partial x} u |x_r^\tau_x|.$$

(3)

$$\left(\frac{\partial}{\partial t} u, u\right) = \left(i\frac{\partial^2}{\partial x^2} u, u\right) + (i\beta |u|^2 u, u) = -(i\frac{\partial}{\partial x} u, \frac{i}{2} \frac{\partial}{\partial x} u) + (i\beta |u|^2 u, u) - iu^* \frac{\partial}{\partial x} u |x_r^\tau_x|.$$

(4)

Summing Eq. (3) and Eq. (4) leads to

$$\frac{\partial}{\partial t} ||u||^2 = iu^* \frac{\partial}{\partial x} u |x_r^\tau_x| - iu^* \frac{\partial}{\partial x} u |x_l^\tau_x|.$$

(5)

To satisfy the well-posedness requirement, the imposed boundary conditions are chosen such that $\frac{\partial}{\partial t} ||u||^2 \leq 0$ in Eq. (5). One set of well-posed boundary conditions can be given as follows

$$\frac{\partial}{\partial t} u = i\frac{\partial^2}{\partial x^2} u + i\beta |u|^2 u, \quad t > 0,$$

$$\frac{\partial}{\partial t} u = g_l(t), \quad x = x_l,$$

$$\frac{\partial}{\partial t} u = g_r(t), \quad x = x_r,$$

$$u = f(x), \quad t = 0,$$

(6)

where $f(x)$ is the initial data, $g_l(t)$ and $g_r(t)$ are the boundary data. To arrive at the SBP formulation of the above equation, we present the following set of definitions,
Definition 3. Given a discrete real-valued functions $V^T = [V_1, V_2, ..., V_m]$ and $U^T = [U_1, U_2, ..., U_m]$ on a grid defined by $\{x_i\}_{i=1,2,...,m}$. The discrete inner product is defined as $(V, U)_\mathbb{H} = V^T \mathbb{H} U$. The discrete norm follows as $||V||_\mathbb{H} = \sqrt{(V, V)_\mathbb{H}}$, where $\mathbb{H}$ is a symmetric positive definite matrix that defines a quadrature rule.

Definition 4. The first-derivative SBP operator $D_1 = \mathbb{H}^{-1}(Q + \frac{1}{2}(e_m e_m^T - e_1 e_1^T))$ approximates $\partial_{x_1}$, where $\mathbb{H} = \mathbb{H}^T > 0$ is a diagonal norm matrix defining the discrete inner product, $Q + Q^T = 0$, and $e_1^T V \simeq u(x = x_1)$ and $e_m^T V \simeq u(x = x_r)$ are finite difference approximations of the left and right boundary. For details, we refer to [45].

Definition 5. The second-derivative SBP operator $D_2 = \mathbb{H}^{-1}(-M - e_1 d_1^T + e_m d_m^T)$ approximates $\partial_{xx}$, where $M = D_1^T \mathbb{H} D_1 = M^T \geq 0$ is positive definite matrix, and $d_1^T V \simeq \partial_{xx} u(x = x_1)$ and $d_m^T V \simeq \partial_{xx} u(x = x_r)$ are finite difference approximations of the first derivatives at the left and right boundary. More elaborations are given in [47].

Within the same frame of reference, we can also present the definition of the third-derivative SBP operator,

Definition 6. The third-derivative SBP operator $D_3 = \mathbb{H}^{-1}(R + \frac{1}{2}(d_{1,1} d_{1,1}^T - d_{1,m} d_{1,m}^T) - e_1 d_{2,1}^T + e_m d_{2,m}^T)$ approximates $\partial_{xxx}$, where $R + R^T = 0$, and $d_{1,1}^T V \simeq \partial_{xxx} u(x = x_1)$ and $d_{2,m}^T V \simeq \partial_{xxx} u(x = x_r)$ are finite difference approximation of the second derivative at the left and right boundary. For details, we refer to [69].

Discretizing the domain $(x_l < x < x_r)$ using $m$ equidistant grid points gives a grid defined by the set $\{x_i\}_{i=1,2,...,m}$. The SBP-approximation of (6) reads

$$\frac{\partial}{\partial t} V = i D_2 V + i \beta |V|^2 \odot V, \quad t > 0,$$

$$L_r V = g_r(t), \quad x = x_r,$$

$$L_r V = g_l(t), \quad x = x_l,$$

$$V = f(x), \quad t = 0,$$

where $L_r = d_r$ and $L_r = d_m$ are the right and left boundary operators, respectively. The operator $\odot$ represents Hadamard product or element-wise operation.

Definition 7. Let $L = \begin{bmatrix} L_l \\ L_r \end{bmatrix}$ be the discrete boundary operator and $g = \begin{bmatrix} g_l \\ g_r \end{bmatrix}$ represent the boundary data. The projection operator is then defined as $\mathbb{P} = I - \mathbb{H}^{-1} L (L^T \mathbb{H}^{-1} L)^{-1} L$ and $[BC] = \mathbb{H}^{-1} (L^T \mathbb{H}^{-1} L)^{-1} g$ is the projected data. Here $\mathbb{H}$ defines the norm and $I$ is an $(m \times m)$ identity matrix, where $m$ is the grid size. Thorough elaboration is given in [48].

Including the boundary treatment in (7) using the projection method. The semi-discrete prob-
lem follows as

\[
\frac{\partial}{\partial t} V = \mathbb{P}(i\mathbb{D}_2(\mathbb{P}V + [BC])) + i\beta|\mathbb{P}V + [BC]|^2 \odot (\mathbb{P}V + [BC]) + [BC]
\]

(8)

**Definition 8.** Canceling the boundary data and lower order terms in the discrete setting (Here, Eq. (8)), the approximate solution \( V \) is energy stable if \( \frac{\partial}{\partial t}||V||_2^2 \leq 0 \) which is the discrete analogous of the well-posedness requirement \( \frac{\partial}{\partial t}||u||^2 \leq 0 \) in the continuous setting.

Dropping the boundary terms in Eq. (8) yields

\[
\frac{\partial}{\partial t} V = \mathbb{P}(i\mathbb{D}_2(\mathbb{P}V) + i\beta|\mathbb{P}V|^2 \odot (\mathbb{P}V)).
\]

(9)

To obtain the discrete energy estimate, Eq. (9) is multiplied by \( V^*H \), and it leads to

\[
V^*H \frac{\partial}{\partial t} V = V^*H[P(i\mathbb{D}_2(\mathbb{P}V) + i\beta|\mathbb{P}V|^2 \odot (\mathbb{P}V))]
\]

\[
= \omega^*(-M - e_1d_1^T + e_mD_m^T)(i\omega) + \omega^*(i\beta|\omega|^2 \odot (\omega)), \quad \omega = \mathbb{P}V.
\]

(10)

Adding the conjugate transpose,

\[
\Rightarrow \frac{\partial}{\partial t} V^*H^T V = (-i\omega^*)(-M^T - (e_1d_1^T)^T + (e_mD_m^T)^T)(\omega) + \omega(-i\beta|\omega|^2 \odot (\omega^*)).
\]

(11)

Thus, the energy estimate gives

\[
\frac{\partial}{\partial t} ||V||_2^2 = -i\omega^*M\omega - i\omega^*(e_1d_1^T)\omega - i\omega^*(e_mD_m^T)\omega + i\omega^*|\omega|^2 \odot \omega
\]

\[
+ i\omega^*M\omega + i\omega^*(d_1e_1^T)\omega + i\omega^*(d_me_m^T)\omega - i\omega^*|\omega|^2 \odot \omega,
\]

\[
\Rightarrow \frac{\partial}{\partial t} ||V||_2^2 = -i\omega^*(e_1d_1^T)\omega - i\omega^*(e_mD_m^T)\omega + i\omega^*(d_1e_1^T)\omega + i\omega^*(d_me_m^T)\omega
\]

\[
= -i(e_1^T\omega)^*(d_1^T\omega) - i(e_mD_m^T\omega)^*(d_m^T\omega) - i(d_1^T\omega)^*(e_1^T\omega) - i(d_m^T\omega)^*(e_m^T\omega)
\]

(12)

which is the discrete analog of the continuous energy estimate given in (5). The resemblance in the procedure of obtaining the discrete energy estimate highlights the advantage of the SBP-projection method in proving stability.

**Definition 9.** Throughout this work the convergence rate of the discretization error is verified by testing the numerical approximate solution against an analytical solution and is computed as follows

\[
q = \log_{10} \left( \frac{||u - V^{(m_1)}||_2}{||u - V^{(m_2)}||_2} \right) \left/ \log_{10} \left( \frac{m_1}{m_2} \right) \right. \right)^{1/d}
\]

(13)

where \( ||u - V^{(m)}||_2 \) is the equivalent L2 discrete norm of the error, \( v^{(m)} \) is the numerical solution for a grid size \( m \) and \( d \) refers to the number of space dimensions.
4 Time integration

The classical SBP-time marching methods are based on SBP operators to approximate the temporal derivatives, and the SAT technique is used to impose the initial conditions weekly. Traditionally, diagonal norm SBP operators are favorably considered for time discretization due to stability reasons [70, 71]. These are essentially based on centered finite differences in the interior of order $2p$ with a biased stencil for boundary closure of order $p$. The global truncation error of diagonal norm SBP-time marching methods is $O(\Delta t^{p+1})$ [72], where $\Delta t$ is the mesh size of the temporal grid. These methods are A- and L-stable which are desirable properties for stiff IBVP, and lead to superconvergence of the end-time solution [54]. The superconvergence property of SBP discretizations is proved if the formulation is dual consistent to be of the order of the associated norm [73, 74]. For a diagonal norm SBP-time marching methods, superconvergence means here that the last step solution’s error converges at a rate $2p$, with $2p$ being the order of accuracy in interior stencil [75]. This implies that the end-time solution has a convergence order that is associated with the underlying quadrature [56].

In this manuscript, a more general framework for SBP is considered for time integration, referred to as generalized SBP (GSBP). First proposed in [76], GSBP is featured by operators with non-repeating interior nodes, absence of either or both boundary nodes, and typically non-equidistant nodal allocation (abscissa). Every quadrature rule principally can be associated with a corresponding SBP operator, and conversely, every positive-definite norm matrix of an SBP operator possesses an underlying quadrature rule. It was proven that the presence of quadrature is necessary and sufficient to derive an associated SBP operator [77]. In contrast to the classical SBP, the requirement of repeated internal stencil is removed for the GSBP-time marching methods. Therefore, it is possible to acquire a prescribed order of accuracy involving a smaller number of temporal grid points, hence, a gain in efficiency. General accuracy and stability attributes of the classical SBP-time marching methods were expanded to include the generalized framework in [56]. We Remark that for GSBP-marching methods the interpolated solution converges with the order of the norm rather than the underlying quadrature.

While the order of the classical SBP-time marching methods is of the order of the underlying quadrature $\tau$, it was proven in [56] that the guaranteed order of the diagonal norm GSBP-time marching method and its associated diagonal norm obeys the relation $p = \min(2q+1, \tau)$, where $q$ is the order of the GSBP operators. Thus, we can infer that the superconvergence of the interpolated solution is of the same order as of the method, which does not hold necessarily for the dense norm GSBP-time marching method. For the purpose of this work, we consider the following $s = 4$ stages/grid points GSBP[$\rho, q, \tau$]-time marching methods given in the same paper [56], a time-marching method based on Legendre-Gauss-Lobatto LGL[2$s-2$, $s-1, 2s-2$] [78, 79], time-marching method based on Legendre-Gauss LG[2$s-1$, $s-1, 2s$] and a diagonally implicit (DI) GSBP method of order 4. These are diagonal norm, A- and L-stable time-marching methods.
on and for the sake of simplicity, the two terms SBP and GSBP will be used exchangeable as there is no marked difference in the follow-up introduction.

4.1 SBP-SAT time integrator

Consider the following semi-discrete IBVP

\[
\frac{\partial}{\partial t} u(x, t) = \lambda u(x, t), \quad u(x, 0) = u_0, \quad \forall t \in [0, T].
\]

(14)

where \(\lambda\) constitute a spacial descretization, and \(u(x, 0) = u_0\) is the initial data.

A discretization of the first temporal derivative with finite difference SBP operator can be defined according to [45] as follows,

Definition 10. The first-derivative SBP operator \(D_t = H^{-1}Q\) of order \(P\) approximates

\[D_t u \approx \frac{\partial}{\partial t} u\]

on a grid defined by \(\{t_i\}_{i=1,2,...,m}\), and \(t_i\) is unique but not requisite uniformly distributed. The operator \(D_t\) satisfies the relation \(Q + Q^T = \tau_R \tau_L^T - \tau_L \tau_R^T\), where \(H_t\) is a symmetric positive definite discrete norm matrix that defines a quadrature rule and approximates the \(L_2\) scalar product \([U, V]_{H_t} \approx \int_0^T u(t)v(t)dt\). The projection operator \(\tau_L, \tau_R\) approximates the projection into the boundary as \(\tau_T^L u \approx u(0), \tau_T^R u \approx u(T)\).

Applying a time marching method with the SBP operator to the IBVP (14) requires a strategy of imposing the initial boundary condition. The SAT technique as mentioned above provides a way to enforce the initial data weekly. Essentially, a SAT term is added to the semi-discrete problem such wise the solution’s values at the initial boundary are drawn/forced toward the initial data weekly. This coupling approximates the numerical solution and the boundary conditions simultaneously, hence the name. Providing SAT term, the full discrete formulation of (14) gives

\[
D_t u = \lambda u - \sigma H_t \tau_L(u_0 - \tau_T^L u),
\]

\[
= (D_t - \sigma H_t \tau_L \tau_T^L )u = -\sigma H_t \tau_L u_0 + \lambda u,
\]

(15)

where \(\sigma \in \mathcal{R}\) is a penalty term that dictates the coupling strength between the solution at \(t = 0\) and initial data and is yet to be determined. Presuming that (14) is a well-posed problem possessing an energy-stable semi-discrete estimate, a GSBP-time marching method with the SAT approach yields a fully discretized energy estimate, hence a proof of numerical stability given the right choice of \(\sigma\). Stability inspections in [54, 55, 80] demonstrate that the choice of \(\sigma < -\frac{1}{2}\) induce a stable approximation, and invertible left-hand matrix \((D_t - \sigma H_t \tau_L \tau_T^L)\) under the assumption that all eigenvalues have strictly real positive parts. The invertibility proof for SBP-time marching methods with repeating stencil operators of higher order than two was given rigorously only in the recent work [81]. Generally, the assumption does not hold for certain SBP schemes (Theorem 2.1. in [61]). For the particular value of \(\sigma = -1\), a clean energy estimate is obtained. Furthermore, the
choice of $\sigma = -1$ leads to a dual consistent problem and allows for a multi-block implementation. This has important implications on the stability characteristics and the method’s performance (More on that to be covered).

Next, we substitute $\lambda$ in (15) with the NLS semi-discrete formulation Eq. (8). To match the spatial and the temporal discretization, we define the following modified spacial and temporal operators by an appropriate Kronecker scaling,

$$
\begin{align*}
\mathbb{D}_t &= (D_t - \sigma H_t \tau_L \tau_L^T) \otimes I_x, \\
\mathbb{R} &= \sigma H_t \tau_L \otimes I_x, \\
\mathbb{D}_x &= D_2 \otimes \mathbb{I}_t, \\
\mathbb{D}_x &= D_2 \otimes \mathbb{I}_t,
\end{align*}
$$

where $I_x$ and $I_t$ are identity matrices of dimension $m_x$ and $m_t$, respectively. The subscripts $x$ and $t$ denote correspondingly a spacial and a temporal operator. The resulting fully discrete NLS formulation gives

$$
\mathbb{D}_t U = -\mathbb{R} U_0 + \frac{i}{2}\mathbb{D}_x U + \frac{i}{2}D_2 [BC] - i\beta F(U),
$$

with $U$; the global solution, $U_0$; the initial global solution of the time marching method, $[BC]$; the imposed boundary conditions and $F(\cdot)$ denotes a nonlinear functions. The configuration of these vectors are

$$
\begin{align*}
U &= [u_0, u_1, \ldots, u_{m_t}]^T, \\
U_0 &= [u_0, u_0, \ldots, u_0]^T, \\
u_i &= [u_{i,0}, u_{i,1}, \ldots, u_{i,m_x}]^T, \\
F(U) &= [F(u_0), F(u_1), \ldots, F(u_{m_t})]^T, \\
[BC] &= \text{vec}(H^{-1}_x L^T (L^T H_x L^T)^{-1} [g_0, g_1, \ldots, g_{m_t}]).
\end{align*}
$$

In the above, the first index marks time and the second marks space in $u_{i,j}$, and $g_i$ corresponds to the spacial boundary data of the temporal node $i$.

4.2 SBP-Projection time integrator

Full discretization of IBVP via the SBP-projection is a recent development regarding the SBP-based time marching methods. For the purpose of clarity and brevity in introducing this new method, we start by presenting the following set of definitions,

**Definition 11.** Let $\mathcal{N} = \{v \in \mathbb{R}^n | D_1 v = 0\}$ be a Null space of the SBP operator $D_1$, $\mathcal{V}_0 = \{u \in \mathbb{R}^n | u(t = 0) = \tau_L^T u = 0\}$ be a vector space for all grid functions that vanish at the interpolated first boundary point, and $\mathcal{V}_1 = \{u \in \mathbb{R}^n | \exists v \in \mathbb{R}^n : u = D_1 v\}$ be a vector space of all grid functions that can be expressed as derivative of other grid functions.. Remark that $\mathcal{V}_0, \mathcal{V}_1 \subset \mathcal{N}$. 

11
Definition 12. Considering \( \frac{\partial^n}{\partial x^n} (P^{n-1}(x)) = 0 \), where \( P^{n-1}(x) \) is complete polynomial, then all the terms in \( P^{n-1}(x) \) are said to be nullspace modes of \( \frac{\partial^n}{\partial x^n} \). Thus, \( n \)-order derivative operator \( \mathbb{D} \) can be defined to be nullspace consistent if it contains all nullspace modes or \( v_i = cx^{i<n} \in \mathcal{N} \), where \( c \) is some coefficient.

Definition 13. The SBP operator \( \mathbb{D}_1 \) is nullspace consistent if the following statement holds,

\[
\forall u \in \mathcal{R}^n \Rightarrow u \in \mathcal{N} \leftrightarrow u \in \text{Span}\{1\} \Rightarrow \dim(\ker\mathbb{D}_1) = 1, \quad 1 = [1, 1, ..., 1]^T.
\]

The current scheme is modeled around the intention of transforming (14) into an ordinary differential equation (ODE) of integral form

\[
u(x, t) = u(x, t = 0) + \int_0^T \lambda u(x, \tau) d\tau.
\]  

(19)

For that purpose, two operators are required; i) the discrete inverse of the first derivative operator \( J \) which is an integral operator that approximates \( J f^T \), and ii) a projection operator \( F \) which projects the discrete right-hand side \( \lambda U \) into the range of the derivative operator \( \mathbb{D} \) before applying the inverse \( J \), such that the full discrete formulation of (14) gives

\[
U = U_0 + JF(\lambda U)
\]  

(20)

A concise encapsulation for the reasoning that leads to deriving the SBP-projection time marching method in [65], aided by the theorems (Theorem 3.19, 9.14, 9.15, and 9.18 in [82]) can be as follows,

Set the assumption that \( \mathbb{D} : \mathcal{V}_0 \rightarrow \mathcal{V}_1 \) is bijective mapping. Thus, implying injectivity; \( \mathbb{D}^{-1} : \text{im}\mathbb{D} \rightarrow \mathcal{V}_0 \), and subjectivity; \( \text{im}\mathbb{D} = \mathcal{V}_1 \). put together, it follows that there exist \( J : \mathcal{V}_1 \rightarrow \mathcal{V}_0 \), and \( \mathbb{D}^{-1} \) is invertible. Remark that generally, the \( \mathbb{D} \) matrices used in the current work i.e. 4-stages LGL-quadrature-based second derivative GSBP operator, do not satisfy this criterion and hence they are not invertible. Therefore, we replace the inverse with the pseudo-inverse. We conjecture that this substitution doesn’t have repercussions on the uniqueness of the solution providing that there is a projection operator that projects the right-hand side into \( \text{col}(\mathbb{D}) \).

Next, providing that \( \mathbb{D}^* = \mathbb{H}^{-1}\mathbb{D}^{-1}\mathbb{H} \) is the adjoint operator of \( \mathbb{D} \) [75], the following relation holds \( \ker\mathbb{D}^* = [\text{im}\mathbb{D}]^\perp \) (theorem 3.19 [82]). By definition \( \text{im}\mathbb{D} = \{ \mathbb{D}u \mid u \in \mathcal{V}_0 \} \) represents all the resolved grid functions, thereafter, if we define the set \( \{\mathbf{o}\} \) to be a complete orthonormal basis set of \( \ker\mathbb{D}^* = \text{span}\{\mathbf{o}\} \), then \( \{\mathbf{o}\} \) can represent all the unresolved grid function (grid oscillations as labeled in [83]), and \( [\text{span}\{\mathbf{o}\}]^\perp = \text{im}\mathbb{D} = \mathcal{V}_1 \).
With all that said, we can bring to the fore that the generic hand right side $\lambda U$ might not be defined on $V_1$. This necessitates projecting $\text{span}\{o\}$ onto its orthogonal complement $[\text{span}\{o\}]^\perp = \text{im}D$. Considering $\{o\}$ is a complete orthonormal basis set in $V_1$, The projection theorem states that $V_1 = \text{span}\{o\} \ominus [\text{span}\{o\}]^\perp$ (theorem 9.15 [82]). Therefore, the best approximation of $v \in V_1$ from within $\text{span}\{o\}$ is $v = (v, o)_H o$ (theorem 9.14 [82]) and normalizing with $\|o\|_H^2$, we can deduce the relation $\hat{v} = \frac{o^*}{\|o\|_H^2} v$, where $\frac{o^*}{\|o\|_H^2}$ is the orthogonal projection onto $\text{span}\{o\}$ and retrospectively $F = I - \frac{o^*}{\|o\|_H^2}$ is the orthogonal projection onto $[\text{span}\{o\}]^\perp = \text{im}D = V_1$.

Conclusively, we can define the projection operator that projects the semi-discrete approximation $\lambda U$ onto the space of $D$ by

$$F = I_x - \frac{o^*}{\|o\|_H^2}.$$

(21)

In the context of the NLS equation, the fully discrete approximation is obtained by substituting $\lambda$ in (20) by the semi-discrete SBP projection approximation in (8), which is

$$U = U_0 + [\mathbb{JF}](\frac{i}{2} \mathbb{D}_x U + \frac{i}{2} \mathbb{D}_x BC - i\beta F(U)).$$

(22)

Excluding $[\mathbb{JF}]$ in above, all vector configurations following the definitions in (18).

Remark that the columns of $\mathbb{J}$ maps vectors onto ker$\tau^T_L = V_0$, and the columns of $\mathbb{F}$ maps vectors onto im$D = V_1$ only. In view of that, the product $\mathbb{JF}$ might not necessarily be mapped onto $V_1$. This can be safeguarded by applying $c_j = c_j - \tau^T_L c_j$ which ensures that $\tau^T_L v = 0$ for any $v \in V_0$. Thereafter, the product $\mathbb{JF}$ is Kronecker-scaled with $I_x$.

4.3 Implicit-explicit additive Runge-Kutta

An alternative strategy is to implement a combination of implicit and explicit methods, termed (IMEX) [84, 85], which is often granted to treat non-linear stiff problems. (For more about stiffness [86]). Stiffness is a burdensome property that impedes traditional explicit numerical integrators from efficiently solving a PDE. This necessitates the implementation of time integration schemes that demand less effort in finding a solution and hence reduce the computational burden. While the NLS equation is not stiff in the traditional sense, we consider implementing the IMEX scheme as another benchmark for numerical efficiency besides the RK4. For nonlinear dispersive models such as the NLS, a very accurate time integration is required which leads to very small time steps using traditional explicit methods. Therefore, the IMEX scheme circumvents this problem by offering the possibility of using a much larger time step size and hence a better numerical efficiency.
In the implicit-explicit combination, the diffusion linear part of the equation is treated by the implicit scheme and the non-linear term by the explicit scheme. In this work, we consider a popular IMEX scheme in the literature, dubbed Additive Runge-Kutta [87]. The considered scheme is of fourth order and comprises a combination of explicit singly diagonally implicit Runge–Kutta (ESDIRK) and explicit Runge–Kutta (ERK). The ESDIRK scheme is an implicit Runge-Kutta scheme, although the word explicit appears in the name.

Considering the following NLS equation with cubic nonlinearity,

$$\frac{\partial}{\partial t} u(x, t) = i \frac{\partial^2}{\partial x^2} u(x, t) + i\beta |u|^2 u = F_l(u(x, t)) + F_{nl}(u(x, t)).$$

(23)

The partitioned Runge-Kutta method (ESDIRK-ERK) is devised to allow for the partitioning of equations by term. Here, Eq. (23) involves terms of different types where $F_l(u)$ denotes the diffusion part and is integrated with ESDIRK, whereas the $F_{nl}(u)$ denotes the nonlinear part and is integrated with ERK. Here, the subscripts, $l$, and $nl$ indicate the term to be linear or non-linear. This treatment evades using an iterative solver which can be computationally demanding and complicated by the characteristics of the matrix to be inverted.

Butcher coefficients of the considered fifth-order IMEX(ESDIRK-ERK) scheme can be found in [87]. In the usual Butcher notation, the coefficients are $a_{ij} \in \mathbb{R}^{s \times s}$, $c_i$, $b_j \in \mathbb{R}^s$, where $s$ corresponds to the number of stages. Denoting the coefficients of ERK with a superscript $E$ and the coefficients of ESDIRK with $I$, the scheme possesses the property of identical abscissa i.e. $c^I = c^E$ which facilitates the coupling between the explicit and the implicit scheme, and $b^I = b^E = b$ with $b_1 = 0$. Thus, the superscript on $b$ coefficients renders superfluous, and the first stage is treated explicitly for both schemes. In addition, the values of the diagonal entries $a_{ii}$ are identical for the ESDIRK scheme. This is critical for overall efficiency as it helps in evading the use of an iterative method. The scheme can be given as follows

$$u^{n+1} = u^n + \Delta t \sum_{i=1}^s b^I_i F_l(u^{(i)}) + \Delta t \sum_{i=1}^s b^E_i F_{nl}(u^{(i)}) = u^n + \Delta t \sum_{i=1}^s b_i (F_l(u^{(i)}) + F_{nl}(u^{(i)})),$$

(24)

and stage values by

$$u^{(i)} = u^n + \Delta t \sum_{j=1}^{i-1} a^I_{ij} F_l(u^{(j)}) + \Delta t \sum_{j=1}^{i-1} a^E_{ij} F_{nl}(u^{(j)}) + \Delta t a^I_{ii} F_l(u^{(i)}),$$

(25)

where the superscripts $(i)$ and $n$ denotes stage solution and the solution at $n\Delta t$, respectively.

The SBP formulation with the projection method of (23) with the above scheme can be elicited in

$$\left(I - i\Delta t a^I_{ii} \mathcal{P} \mathcal{D}_2 \right) V^{n+1} = \mathcal{P} \left(V^n + i\Delta t \sum_{j=1}^{i-1} a^I_{ij} (\mathcal{D}_2 V^{(j)}) + i\Delta t \sum_{j=1}^{i-1} a^E_{ij} |V^{(j)}|^2 \odot V^{(j)}) + [\mathcal{BC}] \right).$$

(26)
As was mentioned above, the diagonal coefficients \( a_{ii}^I \) in the ESDIRK scheme hold the same value. Therefore, the construction of right-hand side matrix \((I - i\Delta t a_I^P D_2)\) need to be done once. We consider one-time matrix decomposition prior to the iterative computation of (26) followed by repetitively solving the linear system with the same decomposition matrices. Observe that if the time step is influenced by an error control function i.e. PID-controller, matrix decomposition needs to be done for each time step.

5 Solving nonlinear iterative system

In this section, we are concerned with solving a large non-Hermitian linear system given by \( n \times n \) complex matrix \( A \); coefficient matrix, and a complex \( n \)-vector \( b \); the right-hand side vector. Thus, the problem considered is:

Find \( u \in \mathbb{C}^n \) such that

\[
Au = b, \tag{27}
\]

where \( u \) is a vector of unknown representing the nonlinear system’s solution. The only assumption made is that \( A \) is a non-singular matrix.

Eq. (17), and (22) can be formulated as an iteration scheme using Newton’s method, respectively, to be

\[
\left( \nabla_t - \frac{i}{2} \nabla_x + i\beta J(U^k) \right) \Delta U = -\Re U_0 - \nabla_t U^k + \frac{i}{2} \nabla_x U^k + \frac{i}{2} D_x [BC] - i\beta F(U^k), \tag{28}
\]

and

\[
\left( I - J(U) \right) \left( \left( \frac{i}{2} \nabla_x - i\beta J(U^k) \right) \right) \Delta U = U_0 - U^k + J(U) \left( \frac{i}{2} \nabla_x U^k + \frac{i}{2} \nabla_x U^k + \frac{i}{2} D_x [BC] - i\beta F(U^k) \right), \tag{29}
\]

where \( J(U) = \text{diag}(J(U)) \), and \( J(\cdot) \) is the Jacobian of \( F(\cdot) \).

Eq. (28) and (29) are followed by

\[
U^{k+1} = U^k + \Delta U, \tag{30}
\]

and take the form of the nonlinear system (27) with a complex-valued non-Hermitian left-hand side coefficient matrix. The most popular method of choice for such a system is the Generalized Minimal Residual Method (GMRES) iterative solver, first derived in [88]. The general idea is to project the solution on a carefully selected orthogonal basis of the Krylov subspace and minimize the norm of the residual vector at every step over the Krylov subspace. The method in its original version is based on the Arnoldi process for constructing the orthogonal basis of the Krylov subspace. However, it is how the basis is acquired and stored that marks the chief distinction between different varieties of GMRES. Elaborate review can be found in [89].
Storage and computation burden in the GMRES algorithm increases as the Krylov subspace grows at each iteration. Therefore, we consider left-preconditioned restarted GMRES with the Arnoldi process as the variant of choice. Restarted GMRES, denoted by GMRES\((m)\) is addressed by recycling the Krylov subspace; performing \(m\) iterations, and then plugging the resulting approximate solution as an initial guess to start another \(m\) iteration. Thus, the basis vectors are discarded and a new set is constructed every cycle. With regard to the preconditioning step, the weak nonlinearity of the NLS equation will be exploited and the left-hand side matrix is taken Jacobian-free as a preconditioner. Accordingly, the matrices
\[
\begin{pmatrix}
\mathbf{D}_t - \frac{i}{2}\mathbf{D}_x
\end{pmatrix}, \quad \begin{pmatrix}
\mathbf{I} - [J\mathbf{F}](\frac{i}{2}\mathbf{D}_x)
\end{pmatrix}
\]
are taken as preconditioners for the schemes (28) and (29), respectively.

Furthermore, we consider the flexible variant of GMRES (FGMRES) which allows changes in the preconditioning at every step [90]. This variant possesses appealing properties some of which enhanced robustness, and flexibility in the preconditioning with the same convergence property as the standard GMRES. Any iterative solver can be used as a preconditioner for FGMRES. For example, the standard GMRES itself can be set as a preconditioner with relaxed parameters i.e. tolerance and the restart parameter.

5.1 Linear iterative approximation

In this subsection, we exploit the weak nonlinearity even further in order to reach a more numerically efficient formulation and step up from Newton’s method to a more general deflection correction method. Experimental observations on the standard GMRES showed a consistently well-conditioned system throughout the numerical simulation when using the preconditioners in (31). From a physical standpoint, Recall that the nonlinear term depends only on \(u\) which represents a generic slow varying amplitude, and hence the change in the nonlinear term depends solely on the amplitude’s change which is small. Therefore, we redraft the formulations in (28) and (29) as
\[
\mathbf{D}_t U - \frac{i}{2}\mathbf{D}_x U - i\beta \mathbf{F}(U) = -\Re U_0 + \frac{i}{2} \mathbf{D}_x [\mathbf{BC}],
\]
and
\[
U - [J\mathbf{F}](\frac{i}{2}\mathbf{D}_x U - i\beta \mathbf{F}(U)) = U_0 + \frac{i}{2} [J\mathbf{F}][\mathbf{D}_x [\mathbf{BC}]],
\]
respectively. Both formulations can be represented in the form \(\mathbf{A}u = b\), and \(\mathbf{A}\) is a nonlinear operator due to the presence of nonlinear term \(\mathbf{F}(\cdot)\).

We aim to approximate the nonlinear system \(\mathbf{A}u = b\) with a linear one \(\tilde{\mathbf{A}}u = b\), where \(\tilde{\mathbf{A}}\) is a linear operator that approximates the nonlinear \(\mathbf{A}\). Assuming that \(\tilde{x}\) is the solution to \(\tilde{\mathbf{A}}u = b\), and
thus the incorrect solution to $\tilde{A}\tilde{u} = \tilde{b}$, the error term is then given by $e(\tilde{u}) = \tilde{A}\tilde{u} - \tilde{b}$. Accordingly, the target problem can be corrected with $\Delta u = \tilde{A}^{-1}e(\tilde{u})$ and the corrected approximate solution is therefore given by

$$
\tilde{\tilde{u}} = \tilde{u} - \tilde{A}^{-1}\tilde{A}\tilde{u} + \tilde{A}^{-1}\tilde{b} = (I - \tilde{A}^{-1}\tilde{A})\tilde{u} + \tilde{A}^{-1}\tilde{b}.
$$

(34)

Plugging in the exact solution $u^*$ in the relation (34) gives $u^* = (I - \tilde{A}^{-1}\tilde{A})u^* + \tilde{A}^{-1}\tilde{b}$. Following the subtraction of both relations we arrive to $\tilde{\tilde{u}} - u^* = (I - \tilde{A}^{-1}\tilde{A})(\tilde{u} - u^*)$, and thus the success of this approach depends on the contractivity of $(I - \tilde{A}^{-1}\tilde{A})$. From previous observations, the contractivity condition is satisfied by the preconditioners in (31).

Substituting $\tilde{A}$ in (34) with the metrics in (31), we derive the linear iterative approximation of (28) and (29), and it reads

$$
\left(\overline{D}_x - \frac{i}{2}\overline{D}_x\right)\Delta U = -RU_0 - \overline{D}_x U^k + \frac{i}{2}\overline{D}_x U^k + \frac{i}{2}\overline{D}_x[BC] - i\beta F(U^k),
$$

(35)

and

$$
\left(I - [JF]\overline{D}_x\right)\Delta U = U_0 - U^k + [JF]\left(\frac{i}{2}\overline{D}_x U^k + \frac{i}{2}\overline{D}_x U^k + \frac{i}{2}\overline{D}_x[BC] - i\beta F(U^k)\right),
$$

(36)

followed by $U^{k+1} = U^k + \Delta U$. Remark that any approximate inverse can be used as a linear operator $\tilde{A}$.

5.2 Multi-block implementation

In the context of the numerical implementation, it is important to note that we make use of multi-block implementation for the fully discrete problems in (17) and (22). The fully discrete system of the IBVP obtained via the SBP-SAT or SBP-projection discretization can be very large and hence forbiddingly difficult to solve. Alternately to solving a memory-demanding large system all at once, the time interval of interest is sliced into smaller segments (blocks), and each segment is solved separately in a consecutive manner. With this approach, a minimal number of grid points can be used in each block, while the number of blocks can be expanded to an arbitrary number. To solve the system in each block individually, the final solution of each block should be independent of other blocks such that it can plugged as an initial condition into the successive block. For the SBP-SAT time marching method, the condition for each block’s solution to be independent is the choice of the SAT parameter $\sigma = -1$. For the SBP projection, no condition is stated for the block’s solution to be independent. The advantage of multi-block implementation is remarked considering that we are using a four-stage quadrature-based marching method in this work. This low temporal resolution makes integration on the whole time interval of interest impractical and the implementation is unstable. More on that follows in Section 8.
Figure 1: Depiction of multi-block implementation. The time interval of interest is divided into two blocks where the interface solution is the end-time solution of block one and it is plugged as the initial condition in the integration of block two. The end-time solution $t_f$ of block 2 depends on $t_i$ but not on the solutions in the other grid points in block 1.

6 Numerical results in one dimension

The numerical results given below are obtained from numerical implementation using the R2022b Matlab version and performed on Dell XPS 13 9380 4 cores i7-8565u. Source code for all the implementations can be found in the GitHub repository [91].

6.1 Test case KdV equation

For a first test case study, we consider the nonlinear KdV equation

$$\frac{\partial u}{\partial t} = -\frac{\partial^3}{\partial x^3} u - 6u \frac{\partial}{\partial x} u,$$

which admits soliton solution and can be used to model the propagation of solitary waves on the water surface. An analytical solution is given in [92] to be

$$u = \frac{1}{2} \text{sech}\left(\frac{1}{2}(x - t)^2\right).$$

Eq. (37) can be rewritten in a skew-symmetric form

$$\frac{\partial}{\partial t} u = -\frac{\partial^3}{\partial x^3} u - 2 \frac{\partial}{\partial x} (u^2) - 2u \frac{\partial}{\partial x} u.$$  

The SBP-projection approximation of (39) gives

$$\frac{\partial}{\partial t} V = -\mathbb{P}(\mathbb{D}_3 W - 2\mathbb{D}_1 (\mathbb{W} W) - 2\mathbb{W} D_1 W),$$

where $W = \mathbb{P} V + [BC]$, and $\mathbb{W} = \text{diag}(W)$.

Following the same procedures and notations in Section 3, the continuous energy estimate of (39) gives
and the mimicking discrete estimate follows as

\[ \frac{\partial}{\partial t} \| \mathbf{V}^e \|^2 = -\left( \mathbf{d}_1^T \mathbf{V} \right)^2 - 2(\mathbf{e}_1^T \mathbf{V})(\mathbf{d}_2^T \mathbf{V}) - 2(\mathbf{e}_1^T \mathbf{V})(\mathbf{d}_2^T \mathbf{V})^2 + \left( (\mathbf{d}_1^T \mathbf{V})^2 - 2(\mathbf{e}_1^T \mathbf{V})(\mathbf{d}_2^T \mathbf{V}) - 4(\mathbf{e}_1^T \mathbf{V})^3 \right). \]

(42)

The linear iterative approximation of the projection in time and SAT in time follow, respectively,

\[ \left( \mathbb{I} + [\mathbb{F}] \right) \mathbf{V}^{k+1} = \mathbf{V}^k + \Delta \mathbf{V}^k, \]

(43)

\[ \mathbf{V}^{k+1} = \mathbf{V}^k + \Delta \mathbf{V}^k \]

(44)

where \( \mathbf{V}^k = \text{diag}(\mathbf{V})^k \).

For this test case and for the sake of comparison, we consider also the SBP-SAT method for spatial discretization with the semi-discrete form given by

\[ \mathbf{V}_t = -\mathbb{D}_2 \mathbf{V} - 2\mathbb{D}_1 (\mathbb{D} \mathbf{V}) - 2\mathbb{D}_1 \mathbf{V} + \text{SAT}_t + \text{SAT}_{r_1} + \text{SAT}_{r_2}. \]

(45)

Given

\[ \begin{align*}
\text{SAT}_t &= \alpha_t \mathbb{H}^{-1} \mathbf{e}_1 (2(\mathbf{V}_1) + |\mathbf{V}_1|) \mathbf{V}_1 + \mathbf{d}_2 \mathbf{V} - g_1(t), \\
\text{SAT}_{r_1} &= \alpha_{r_1} \mathbb{H}^{-1} \mathbf{e}_m (2(\mathbf{V}_m) + |\mathbf{V}_m|) \mathbf{V}_m + \mathbf{d}_2 \mathbf{V} - g_{r_1}(t), \\
\text{SAT}_{r_2} &= \alpha_{r_2} \mathbb{H}^{-1} \mathbf{d}_{m} \mathbf{d}_1 \mathbf{V} - g_{r_2}(t),
\end{align*} \]

(46)

with \( \alpha_t = -1, \alpha_{r_1} = 1 \) and \( \alpha_{r_2} < -\frac{1}{2} \), Eq. (45) corresponds to a set of strongly well-posed boundary conditions given by

\[ \begin{align*}
2(u + |u|)u + \frac{\partial^2 u}{\partial x^2} &= g_1(t), \quad x = x_l, \\
2(u - |u|)u + \frac{\partial^2 u}{\partial x^2} &= g_{r_1}(t), \quad x = x_r, \\
\frac{\partial u}{\partial x} &= g_{r_2}(t), \quad x = x_r,
\end{align*} \]

(47)

where \( g_1(t), g_{r_1}(t) \) and \( g_{r_2}(t) \) are the boundary data. To derive the linear approximation of the SBP in time and SAT in time schemes for (45), we define the following operators

\[ \begin{align*}
\mathbb{E}_1 &= \mathbb{I}_l \otimes \mathbf{e}_1^T, \quad \mathbb{E}_m = \mathbb{I}_l \otimes \mathbf{e}_m^T, \\
\mathbb{D}_{1,1} &= \mathbb{I}_m \otimes \mathbf{d}_1^T, \quad \mathbb{D}_{1,m} = \mathbb{I}_m \otimes \mathbf{d}_1^T, \\
\mathbb{D}_{2,1} &= \mathbb{I}_m \otimes \mathbf{d}_2^T, \quad \mathbb{D}_{2,m} = \mathbb{I}_m \otimes \mathbf{d}_2^T,
\end{align*} \]

(48)

which gives

\[ \begin{align*}
[SAT] &= \mathbb{I}_l \otimes \left( \alpha_t \mathbb{H}^{-1} \mathbf{e}_1 \mathbf{d}_1^T + \alpha_{r_1} \mathbb{H}^{-1} \mathbf{e}_m \mathbf{d}_2^T + \alpha_{r_2} \mathbb{H}^{-1} \mathbf{d}_1 \mathbf{d}_1^T \right).
\end{align*} \]

(49)
Moreover, the following sets of operators are defined,

\[
\mathbb{H}_t = \alpha_1 \text{diag}(\mathbb{H}^{-1}(1_{m_x \times 1} \otimes e_1)),
\]

\[
\mathbb{H}_{r1} = \alpha_1 \text{diag}(\mathbb{H}^{-1}(1_{m_y \times 1} \otimes e_m)),
\]

\[
\mathbb{H}_{r2} = \alpha_2 \text{diag}(\mathbb{H}^{-1}(1_{m_y \times 1} \otimes d_{1:m})),
\]

and

\[
[S\text{AT}]_1 = \mathbb{D}_{2:1} V^k + 2(\mathbb{E}_1 V^k + |\mathbb{E}_1 V^k|) \odot \mathbb{E}_1 V^k - g_t,
\]

\[
[S\text{AT}]_{r1} = \mathbb{D}_{2:m} V^k + 2(\mathbb{E}_m V^k - |\mathbb{E}_m V^k|) \odot \mathbb{E}_m V^k - g_{r1},
\]

\[
[S\text{AT}]_{r2} = \mathbb{D}_{1:m} V^k - g_{r2},
\]

which gives

\[
[S\text{AT}] = \mathbb{H}_t \left([S\text{AT}]_1 \otimes 1_{m_x \times 1}\right) + \mathbb{H}_{r1} \left([S\text{AT}]_{r1} \otimes 1_{m_y \times 1}\right) + \mathbb{H}_{r2} \left([S\text{AT}]_{r2} \otimes 1_{m_y \times 1}\right).
\]

Thus, The linear iterative approximation of the projection in time and SAT in time follow, respectively,

\[
\left(\mathbb{D}_t + \mathbb{D}_3 - [S\text{AT}]\right) \Delta V^k = -\mathbb{R} V_0 - (\mathbb{D}_t + \mathbb{D}_3) V^k - 6V^kD_1 V^k + [S\text{AT}],
\]

and

\[
\left(I + \mathbb{J} F(\mathbb{D}_3 - [S\text{AT}])\right) \Delta V^k = V_0 - V^k - [\mathbb{J} F(\mathbb{D}_3 V^k + 6V^kD_1 V^k - [S\text{AT}])].
\]

We run numerical experimentation over the domain \(x \in [-10, +10]\), and end time \(t = 1\), with the exact solution and the boundary conditions taken from Eq. (38). Table 1 presents the results of both projection in time and SAT in time when the spacial discretization is given with the SBP6-projection. Table 2 presents the results of both projection in time and SAT in time when the spacial discretization is given with the SBP6-SAT. Data comparison of the results shows an apparent better numerical efficiency for the projection technique for both spatial and temporal discretization.

| Grid | \(q\) | \(|e|_0^2\) | CPU | \(|e|_0^2\) | CPU | \(|e|_0^2\) | CPU | \(|e|_0^2\) | CPU |
|------|------|------|-----|------|-----|------|-----|------|-----|
| 51   | 5.8275 | 9.4367e-08 | 3.4062 | 9.4366e-08 | 0.0781 | 9.4438e-08 | 0.0781 | 9.4372e-08 | 0.1094 | 9.4400e-08 | 0.0938 |
| 101  | 5.9717 | 9.4367e-08 | 3.4062 | 9.4366e-08 | 0.0781 | 9.4438e-08 | 0.0781 | 9.4372e-08 | 0.1094 | 9.4400e-08 | 0.0938 |
| 201  | 6.0022 | 9.494e-09 | 34.5625 | 1.4914e-09 | 0.1094 | 1.5993e-09 | 0.1250 | 1.4824e-09 | 0.1250 | 1.4971e-09 | 0.1406 |

Table 1: \(L_2\) error norm and CPU time taken at end time \(t = 1\) over the domain \(x \in [-10, +10]\) using the SBP6-projection approximation method in space. The data set is obtained using the schemes (43) and (44) for the LGL- and LG-quadrature-based time marching methods against RK4 as a benchmark.
SAT in space

Projection in time

SAT in time

RK4
LGL
LG

LGL
LG

Grid | q | $|\varepsilon||\|_{L^2}$ | CPU | $|\varepsilon||\|_{L^2}$ | CPU | $|\varepsilon||\|_{L^2}$ | CPU | $|\varepsilon||\|_{L^2}$ | CPU | $|\varepsilon||\|_{L^2}$ | CPU
--- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | ---
51  | 3.2499e-04 | 0.0625 | 3.2544e-04 | 0.0312 | 3.2536e-04 | 0.0469 | 3.2538e-04 | 0.0469
101 | 5.8289  | 6.0556e-06 | 0.6094 | 6.0540e-06 | 0.0625 | 6.0535e-06 | 0.0781 | 6.0540e-06 | 0.1094 | 6.0540e-06 | 0.0781
201 | 9.9287e-08 | 5.2500 | 9.9265e-08 | 0.1094 | 9.9370e-08 | 0.1250 | 9.9292e-08 | 0.1562 | 9.9265e-08 | 0.1250
401 | 6.0015  | 1.5731e-09 | 48.6719 | 1.5669e-09 | 0.1406 | 1.5835e-09 | 0.2031 | 1.5596e-09 | 0.1719 | 1.5835e-09 | 0.2031

Table 2: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-10, +10]$ using the SBP6-SAT approximation method in space. The data set is obtained using the schemes (53) and (54) for the LGL- and LG-quadrature-based time marching methods against RK4 as a benchmark.

![Figure 2: Plots of CPU time for different time marching schemes as a function of $L_2$-error. The first P or SAT notion refers to the projection or the SAT imposition of the boundary data, while the second refers to the imposition technique of the initial data.](image)

6.2 NLS with bright soliton solution

In this section, we proceed with the focusing NLS equation in one dimension with cubic nonlinearity given by

$$
i \frac{\partial u(x,t)}{\partial t} = -\frac{1}{2} \frac{\partial^2}{\partial x^2} u(x,t) + \beta |u(x,t)|^2 u(x,t), \quad \beta = -1.$$  \hfill (55)
The above equation admits a bright soliton solution in the form [66]

\[ u(x, t) = \frac{\alpha}{\sqrt{\beta}} \text{sech}(\alpha(x - \nu t - x_0)) \exp\left[i(\nu x - \frac{1}{2}(\nu^2 - a^2)t + \theta_0)\right], \]

(56)

when \( \beta < 0 \). Here, \( \frac{\alpha}{\sqrt{\beta}} \) represents the wave amplitude, \( \nu \) represents velocity, and \( x_0 \) and \( \theta_0 \) are the soliton’s shifts in space and phase at \( t = 0 \). Eq. (55) has been profitably employed in the study of solitons in dispersive nonlinear optical fibers. The semi-discrete formulation with SBP-projection reads

\[ \frac{\partial}{\partial t} V = P\left( i\frac{\beta}{2} W - i\beta |W|^2 \odot W \right) + [BC], \quad W = P V + [BC]. \]

(57)

The solution \( u(x, t) \) represents a moving oscillatory bright soliton as depicted in Figure 4. Tables 3, 4, and 5 present \( L_2 \) error and CPU time for IMEX, projection in time, and SAT in time, respectively. Here, the SBP4-projection approximation method is used with an end time \( t = 1 \) over the domain \( x \in [-3, +3] \). Parameters’ values are given as follows \( \alpha = 2 \), \( \nu = 1 \), \( x_0 = 0 \) and \( \theta_0 = 0 \). The initial data and boundary conditions are taken from the exact solution (56). The results show remarkably better numerical efficiency for LGL- and LG-quadrature-based time marching methods. Comparing Tables 4 and 5, it is worth noting that the projection in time fares better than the SAT in time across all employed time integrator schemes.

<table>
<thead>
<tr>
<th>Grid</th>
<th>RK4</th>
<th>IMEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>q</td>
<td>(</td>
<td>e</td>
</tr>
<tr>
<td>51</td>
<td>0.0043</td>
<td>0.0469</td>
</tr>
<tr>
<td>101</td>
<td>1.0246</td>
<td>2.7906e-04</td>
</tr>
<tr>
<td>151</td>
<td>4.1818</td>
<td>5.4561e-05</td>
</tr>
<tr>
<td>201</td>
<td>4.0167</td>
<td>1.7311e-05</td>
</tr>
<tr>
<td>401</td>
<td>4.0099</td>
<td>1.9807e-06</td>
</tr>
<tr>
<td>801</td>
<td>4.0090</td>
<td>6.7752e-08</td>
</tr>
<tr>
<td>1601</td>
<td>4.0108</td>
<td>4.2193e-09</td>
</tr>
</tbody>
</table>

Table 3: \( L_2 \) error norm and CPU time taken at end time \( t = 1 \) over the domain \( x \in [-2, +2] \) using the SBP4-projection approximation method in space. Parameters’ values are \( \alpha = 2 \), \( \nu = 1 \), \( x_0 = 0 \) and \( \theta_0 = 0 \). The data set depicts a comparison between the IMEX(ESDIRK4-ERK4) integrator and RK4 as a benchmark.
### Table 4: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. The data set compares the LGL- and LG-quadrature-based, DI- and SBP4- time marching methods with the projection scheme in time against RK4 as a benchmark.

| Grid | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU |
|------|----------|-----|----------|-----|----------|-----|----------|-----|----------|-----|
| 51   | 0.0043   | 0.0469 | 0.0041   | 0.1406 | 0.0040   | 0.0156 | 0.0036   | 0.0156 | 0.0043   | 0.0156 |
| 101  | 2.7506e-04 | 0.1250 | 2.7514e-04 | 0.2969 | 2.5088e-04 | 0.0312 | 2.7137e-04 | 0.0312 | 2.7516e-04 | 0.1250 |
| 151  | 5.4561e-05 | 0.2344 | 5.4694e-05 | 0.3281 | 5.3822e-05 | 0.0925 | 5.0535e-05 | 0.0469 | 5.4661e-05 | 0.2656 |
| 201  | 1.7311e-05 | 0.5469 | 1.7325e-05 | 0.4062 | 1.4968e-05 | 0.0925 | 1.7020e-05 | 0.0781 | 1.7333e-05 | 0.3594 |
| 401  | 1.6837e-06 | 2.5781 | 1.0915e-06 | 0.1562 | 1.0387e-06 | 0.0781 | 1.0506e-06 | 0.1406 | 1.0852e-06 | 1.656  |
| 801  | 6.7752e-08 | 19.9375 | 6.7849e-08 | 3.2656 | 6.3584e-08 | 0.0469 | 6.5685e-08 | 0.0625 | 6.8229e-08 | 3.5000 |
| 1601 | 4.2193e-09 | 158.7656 | 4.2414e-09 | 9.2500 | 2.7780e-09 | 0.6875 | 3.9844e-09 | 0.1875 | 4.2469e-09 | 13.7344 |

### Table 5: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. The data set compares the LGL- and LG-quadrature-based, DI- and SBP4- time marching methods with the SAT scheme in time against RK4 as a benchmark.

| Grid | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU | $|e|_2^2$ | CPU |
|------|----------|-----|----------|-----|----------|-----|----------|-----|----------|-----|
| 51   | 0.0043   | 0.0469 | 0.0043   | 0.1250 | 0.0043   | 0.0156 | 0.0043   | 0.0156 | 0.0043   | 0.0625 |
| 101  | 2.7506e-04 | 0.1250 | 2.7514e-04 | 0.2969 | 2.7490e-04 | 0.0469 | 2.7492e-04 | 0.0312 | 2.7506e-04 | 0.2344 |
| 151  | 5.4561e-05 | 0.2344 | 5.4694e-05 | 0.3281 | 5.4626e-05 | 0.0781 | 5.3717e-05 | 0.0625 | 5.4791e-05 | 0.3438 |
| 201  | 1.7311e-05 | 0.5469 | 1.7325e-05 | 0.4062 | 1.7316e-05 | 0.0925 | 1.7302e-05 | 0.0781 | 1.7333e-05 | 0.6250 |
| 401  | 1.6837e-06 | 2.5781 | 1.0915e-06 | 0.1562 | 1.0856e-06 | 0.0781 | 1.0829e-06 | 0.1406 | 1.0852e-06 | 1.656  |
| 801  | 6.7752e-08 | 19.9375 | 6.7849e-08 | 3.2656 | 6.3584e-08 | 0.0469 | 6.5685e-08 | 0.0625 | 6.8229e-08 | 3.5000 |
| 1601 | 4.2193e-09 | 158.7656 | 4.2414e-09 | 9.2500 | 2.7780e-09 | 0.6875 | 3.9844e-09 | 0.1875 | 4.2469e-09 | 11.8125 |

Table 4: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. The data set compares the LGL- and LG-quadrature-based, DI- and SBP4- time marching methods with the projection scheme in time against RK4 as a benchmark.

Table 5: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. The data set compares the LGL- and LG-quadrature-based, DI- and SBP4- time marching methods with the SAT scheme in time against RK4 as a benchmark.
(a) Comparison of LG- and LGL-quadrature-based time marching methods against the RK4 and IMEX-scheme.

(b) Comparison of DI- and SBP4-time marching methods against the RK4 and IMEX-scheme

Figure 3: Efficiency plots of different time integrators given in CPU time as a function of $L_2$-error. The dashed and the straight lines represent the efficiency plots when the SAT in time and the projection in time are used, respectively.
Figure 4: Three-dimensional profile of the solution $u(x,t)$, the $t$-axis depicts a moving oscillatory bright soliton across time.

6.3 NLS with two interacting solitons

Proceeding with a more challenging test case, we consider the following NLS example with a focusing nonlinearity, constructed in [93]

$$i \frac{\partial}{\partial t} u = -\frac{\partial^2}{\partial x^2} u - 2|u|^2 u. \tag{58}$$

In this example, we are looking at a related case to signal propagation in optical fibers, which involves the interaction of two stationary solitons. As derived in [93], the exact solution can be given as

$$u(x,t) = \frac{8 \exp(4it) \left(9 \exp(-4x) + 16 \exp(4x)\right) - 32 \exp(16it) \left(4 \exp(-2x) + 9 \exp(2x)\right)}{-128 \cos(12t) + 4 \exp(-6x) + 16 \exp(6x) + 81 \exp(-2x) + 64 \exp(2x)}. \tag{59}$$

The SBP-projection of (58) reads

$$\frac{\partial}{\partial t} V = P(iD_2 W + 2iW \odot |W|^2) + [BC], \quad W = PV + [BC]. \tag{60}$$

Figure 7 represents a three-dimensional profile of the solution $u(x,t)$. The solution depicts two interacting solitons with exponentially decaying tails. Thus, homogeneous Dirichlet boundary conditions can be prescribed if the domain is taken to be wide enough. Given the solution is stationary and periodic in time, this is a good test example of checking stability via long-time simulation. However, the SBP formulation gives already a mathematically provable stability proof, Eq. (12). Due to the solitary interaction, the solution is observed to be less smooth in comparison with the previous example, and the $L_2$-norm of both the first and the second derivative are of a high order of magnitude which makes this example more challenging from the computational point of view.
In this test case, we restrict the comparison of time integrators to LGL- and LG-quadrature-based time marching methods. Tables 6 and 7 give the results of the time marching methods with RK4 as a benchmark. Here, the solution is obtained via the SBP4-projection approximation method in space, end time $t = 1$ over the domain $x \in [-2, +2]$. The results in Tables 6 are given for the linear approximation while the results in Table 7 are given using the iterative solvers GMRES and FGMRES. In both data sets, the same trend is observed as in the previous examples of better numerical efficiency for the projection in time in contrast to the SAT in time. Comparing both Tables 6 and 7, one can remark the large gain in numerical efficiency for the linear approximation of SBP formulation in Section 5.1.

| Grid | $g$ | $|e|_2$ | CPU | $|e|_2$ | CPU | $|e|_2$ | CPU | $|e|_2$ | CPU |
|------|-----|--------|-----|--------|-----|--------|-----|--------|-----|
| 101  | 0.0421 | 0.6875 | 0.0422 | 0.1562 | 0.0419 | 0.1094 | 0.0418 | 0.2031 |
| 201  | 3.9914 | 2.9688 | 0.0027 | 0.1875 | 0.0026 | 0.1562 | 0.0027 | 0.2031 |
| 301  | 4.0175 | 5.3310e-04 | 12.2656 | 5.3295e-04 | 0.7031 | 5.3343e-04 | 0.1875 | 5.3343e-04 | 0.2516 |
| 801  | 4.0052 | 1.0575e-05 | 301.5625 | 1.0560e-05 | 2.4688 | 1.0560e-05 | 2.8438 |

Table 6: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. The data set is obtained using the linear approximation scheme and it depicts a comparison between the projection in time against the SAT in time with the LGL- and LG-quadrature-based time marching methods referenced by RK4 as a benchmark.
Figure 5: CPU time comparison between RK4 and LGL- and LG-quadrature-based time marching methods as a function of $L_2$ error norm. Here, we use the linear approximation for both the projection and the SAT in time.

| Grid | $q$ | $|e|_2$ | CPU | $|e|_2$ | CPU | $|e|_2$ | CPU | $|e|_2$ | CPU |
|------|-----|---------|-----|---------|-----|---------|-----|---------|-----|
| 101  | 0.0421 | 0.6875  | 0.0422 | 1.3125  | 0.0418 | 1.3750  | 0.0422 | 1.3125  | 0.0418 | 1.3750  | 0.0422 | 1.3125  | 0.0418 | 1.3750  |
| 1201 | 3.9914 | 2.9688  | 0.0027 | 2.5000  | 0.0027 | 1.9219  | 0.0027 | 4.0312  | 0.0027 | 6.2500  | 0.0027 | 2.9062  |
| 301  | 4.0175 | 5.3310  | 12.2656 | 5.3310  | 7.0938  | 5.3343  | 4.8594  | 5.3320  | 7.0969  | 5.3343  | 10.1094 |
| 401  | 4.0057 | 1.6896  | 37.2031 | 1.6896  | 12.5049 | 1.6896  | 7.4375  | 1.6896  | 17.5038 | 1.6896  | 20.8438 |
| 801  | 4.0052 | 1.0575  | 301.5625 | 1.0575  | 19.0781 | 1.0604  | 15.5781 | 1.0562  | 29.7031 | 1.0604  | 35.8125 |

Table 7: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. The data set depicts a comparison between the projection in time against the SAT in time with the LGL-quadrature-based time marching methods and GMRES and FGMRES iterative solvers.
6.4 NLS in the semi-classical limit

In this section, we look at an interesting and numerically challenging example of NLS in the semi-classical limit. The NLS equation is given in [94] by

$$i\epsilon \frac{\partial}{\partial t} u(x, t) = -\frac{1}{2} \epsilon^2 \frac{\partial^2}{\partial x^2} u(x, t) - |u(x, t)|^2 u(x, t).$$

(61)
Eq. (61) is an example of focusing NLS with a nonlinear dispersion process governed by the dispersion parameter \( \epsilon \). In the weak limit (small dispersion), referred to as the semi-classical limit \( \epsilon \to 0 \), small-scale oscillations emerge that give rise to modulation instability; an exponential growth of weak perturbed waves during their propagation through the nonlinear dispersive media. Examples of this phenomenon have been observed in e.g. plasma shocks [95] and optical shocks in optical fibers [96]. In the focusing version of NLS, the combination of small dispersion and nonlinearity can have a destabilizing effect on the periodic wavetrain which corresponds to modulation instability. The result is wave breaking succeeded by the onset of \( O(\epsilon) \) wavelength oscillations in the amplitude of the solution. This is captured in the figure 8 that is taken from a simulation using the SBP4-approximation method in space and LG-based-quadrature time marching method with projection in time.

What makes this problem interesting is that this instability occurs in the focusing NLS which is well understood as a stable problem in the absence of background phase oscillation [94]. It is worth emphasizing that the small oscillations observed in the moving wavepacket, Figure 8, are produced by the nonlinear process dominating the group velocity dispersion. This enables the modulation instability to break the pulse into multi-pulses each of a smaller duration. Therefore, in the semi-classical limit, problem (61) can be thought of as strongly nonlinear dispersive phenomena.

The numerical computation of the solution of the problem (61) is markedly challenging for the near zero dispersion limit. The small value of the dispersion parameter necessitates a much smaller value of mesh size \( \Delta x \). The problem is aggravated by the time step restriction when deploying explicit time integrators such as the classical RK4. This renders the problem prohibitively expensive from the computational standpoint. Table 8 and 9 present the \( L_2 \) error norm and CPU time for different dispersive parameters’ values \( \epsilon \). Here, the numerical experimentation was executed such that the \( L_2 \) error norm retains the same order of magnitude. One can observe the near-exponential increase in mesh size corresponding to the monotonic decrease in the value of \( \epsilon \).

The authors in [94] suggest an exact solution to Eq. (61) of the form

\[
  u(x, t) = A(x, t) \exp\left(\frac{i}{\epsilon} \theta(x, t)\right),
\]

where \( A(x, t) \) and \( \theta(x, t) \) are, respectively, real-valued amplitude and phase functions. The solution (62) represents a monochromatic wavepacket propagating along the axis with rapidly decaying tails. For the purpose of numerical experimentation, we employ the method of manufactured solution (MMS) [97, 98] to verify and validate the SBP-projection technique as well as examine convergence and numerical efficiency of the time integrators. We select a priori solution that takes the form of (62) and it reads

\[
  \tilde{u} = 2\sech(x + t) \exp\left(\frac{2i\sech(x + t)}{\epsilon}\right),
\]

The authors in [94] suggest an exact solution to Eq. (61) of the form

\[
  u(x, t) = A(x, t) \exp\left(\frac{i}{\epsilon} \theta(x, t)\right),
\]

where \( A(x, t) \) and \( \theta(x, t) \) are, respectively, real-valued amplitude and phase functions. The solution (62) represents a monochromatic wavepacket propagating along the axis with rapidly decaying tails. For the purpose of numerical experimentation, we employ the method of manufactured solution (MMS) [97, 98] to verify and validate the SBP-projection technique as well as examine convergence and numerical efficiency of the time integrators. We select a priori solution that takes the form of (62) and it reads

\[
  \tilde{u} = 2\sech(x + t) \exp\left(\frac{2i\sech(x + t)}{\epsilon}\right),
\]

The authors in [94] suggest an exact solution to Eq. (61) of the form

\[
  u(x, t) = A(x, t) \exp\left(\frac{i}{\epsilon} \theta(x, t)\right),
\]

where \( A(x, t) \) and \( \theta(x, t) \) are, respectively, real-valued amplitude and phase functions. The solution (62) represents a monochromatic wavepacket propagating along the axis with rapidly decaying tails. For the purpose of numerical experimentation, we employ the method of manufactured solution (MMS) [97, 98] to verify and validate the SBP-projection technique as well as examine convergence and numerical efficiency of the time integrators. We select a priori solution that takes the form of (62) and it reads

\[
  \tilde{u} = 2\sech(x + t) \exp\left(\frac{2i\sech(x + t)}{\epsilon}\right),
\]

The authors in [94] suggest an exact solution to Eq. (61) of the form

\[
  u(x, t) = A(x, t) \exp\left(\frac{i}{\epsilon} \theta(x, t)\right),
\]

where \( A(x, t) \) and \( \theta(x, t) \) are, respectively, real-valued amplitude and phase functions. The solution (62) represents a monochromatic wavepacket propagating along the axis with rapidly decaying tails. For the purpose of numerical experimentation, we employ the method of manufactured solution (MMS) [97, 98] to verify and validate the SBP-projection technique as well as examine convergence and numerical efficiency of the time integrators. We select a priori solution that takes the form of (62) and it reads

\[
  \tilde{u} = 2\sech(x + t) \exp\left(\frac{2i\sech(x + t)}{\epsilon}\right),
\]

The authors in [94] suggest an exact solution to Eq. (61) of the form

\[
  u(x, t) = A(x, t) \exp\left(\frac{i}{\epsilon} \theta(x, t)\right),
\]

where \( A(x, t) \) and \( \theta(x, t) \) are, respectively, real-valued amplitude and phase functions. The solution (62) represents a monochromatic wavepacket propagating along the axis with rapidly decaying tails. For the purpose of numerical experimentation, we employ the method of manufactured solution (MMS) [97, 98] to verify and validate the SBP-projection technique as well as examine convergence and numerical efficiency of the time integrators. We select a priori solution that takes the form of (62) and it reads

\[
  \tilde{u} = 2\sech(x + t) \exp\left(\frac{2i\sech(x + t)}{\epsilon}\right),
\]

The authors in [94] suggest an exact solution to Eq. (61) of the form

\[
  u(x, t) = A(x, t) \exp\left(\frac{i}{\epsilon} \theta(x, t)\right),
\]

where \( A(x, t) \) and \( \theta(x, t) \) are, respectively, real-valued amplitude and phase functions. The solution (62) represents a monochromatic wavepacket propagating along the axis with rapidly decaying tails. For the purpose of numerical experimentation, we employ the method of manufactured solution (MMS) [97, 98] to verify and validate the SBP-projection technique as well as examine convergence and numerical efficiency of the time integrators. We select a priori solution that takes the form of (62) and it reads

\[
  \tilde{u} = 2\sech(x + t) \exp\left(\frac{2i\sech(x + t)}{\epsilon}\right),
\]
Operating the governing equation (61) onto the chosen solution in (63) by substitution, thereby generating the source term

\[ S(x, t) = i\epsilon \frac{\partial}{\partial t} \bar{u} + \frac{1}{2} \epsilon^2 \frac{\partial^2}{\partial x^2} \bar{u} + |\bar{u}|^2 \bar{u}. \]  

(64)

Thus, the chosen solution in (63) is the exact solution to the modified NLS equation given by

\[ i\epsilon \frac{\partial}{\partial t} u = -\frac{1}{2} \epsilon^2 \frac{\partial^2}{\partial x^2} u - |u|^2 u + S(x, t). \]  

(65)

We run numerical experimentation on Eq. (65) using the SBP-projection approximation method with Neumann boundary condition and initial data taken from Eq. (63). In this section, we restrict the time integrator to the LGL-quadrature-based time marching method with the projection method to project the initial data. The results in Tables 8 and 9 are given for end time \( t = 1 \) and domain \( x \in [-5, +5] \). Comparing the datasets in both tables expounds the advantage of using SBP-integrator as opposed to RK4. One can observe the performance gap between both time integrators widens in near-exponential order as \( \epsilon \to 0 \). In addition, we present the number of blocks that are used for the SBP-integration which shows the number of blocks to be a function of numerical error and the nonlinearity of the problem. More on this observation is to be discussed in the conclusion section.

<table>
<thead>
<tr>
<th>RK4</th>
<th>( \epsilon = 1 )</th>
<th>( \epsilon = 0.5 )</th>
<th>( \epsilon = 0.25 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid</td>
<td>( g )</td>
<td>( |e|_2 )</td>
<td>CPU</td>
</tr>
<tr>
<td>101</td>
<td>5.8920e-04</td>
<td>0.0938</td>
<td>0.0469</td>
</tr>
<tr>
<td>201</td>
<td>4.0131</td>
<td>3.7226e-05</td>
<td>1.3418</td>
</tr>
<tr>
<td>401</td>
<td>4.0093</td>
<td>2.3348e-06</td>
<td>12.1400</td>
</tr>
</tbody>
</table>

| Grid  | \( \|e\|_2 \)       | CPU                 |
| 201   | 3.2138e-04          | 1.9062              |
| 501   | 1.0031              | 2.9569e-05         |
| 1001  | 4.0038              | 5.8395e-05         |

Table 8: \( L_2 \) error norm and CPU time for different dispersive parameter values \( \epsilon \) in the case of RK4 time integrator. The data set is taken such that the value of \( L_2 \) error norm is preserved as \( \epsilon \) is pulled toward the zero limit.

<table>
<thead>
<tr>
<th>LGL-Projection</th>
<th>( \epsilon = 1 )</th>
<th>( \epsilon = 0.5 )</th>
<th>( \epsilon = 0.25 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid</td>
<td>( g )</td>
<td>( |e|_2 )</td>
<td>CPU</td>
</tr>
<tr>
<td>101</td>
<td>7</td>
<td>5.8985e-04</td>
<td>0.0469</td>
</tr>
<tr>
<td>201</td>
<td>10</td>
<td>3.7224e-05</td>
<td>0.0469</td>
</tr>
<tr>
<td>401</td>
<td>21</td>
<td>2.3348e-06</td>
<td>0.1250</td>
</tr>
</tbody>
</table>

| Grid  | \( \|e\|_2 \)       | CPU                 |
| 201   | 3.2155e-04          | 0.0938              |
| 501   | 2.0121e-05          | 1.3418              |
| 1001  | 1.2596e-06          | 12.1400             |
| 1501  | 4.0035              | 5.8395e-05         |

Table 9: \( L_2 \) error norm and CPU time for different dispersive parameter values \( \epsilon \) in the case of LGL-quadrature-based time marching method. The data set is taken such that the value of \( L_2 \) error norm is preserved as \( \epsilon \) is pulled toward the zero limit.
Figure 8: View of small-scale oscillations of $O(\epsilon)$ in the amplitude of the solution produced by modulation instability.

7 Numerical results in two dimensions

In this section, we extend the implementation of the SBP-projection technique and the various considered time integrators into a two-dimensional problem. The implementation procedures regarding the SBP-projection in space differ slightly in the two-dimensional case compared to the one-dimensional described in Section 3. However, the final semi-discrete SBP formulation is identical as well as the implementation of the before-mentioned time integrators.

7.1 Gross–Pitaevskii with bright soliton

We consider the Gross–Pitaevskii given in [99] as a two-dimensional test case. The equation reads

$$i \frac{\partial}{\partial t} u(x, y, t) = -\frac{1}{2} \nabla u(x, y, t) - V_d(x, y)u(x, y, t) + |u(x, y, t)|^2 u(x, y, t),$$

with the initial condition given by

$$u(x, y, 0) = \sqrt{\xi}\exp\left(-\frac{k}{2}(x^2 + y^2)\right).$$

The Gross–Pitaevskii equation (66) can be thought of as a type two-dimensional NLS with an external potential and cubic nonlinearity. This model finds its application most commonly in the study of Bose-Einstein condensates, propagation of light in optical fibers, planar waveguides and recently in small-amplitude gravity waves on the surface of deep zero-viscosity-water as noted in [100].

With the nonlinearity given by $|u|^2u + V_d u$, Eq. (66) expresses a soliton propagating along the temporal direction in an optical medium with spatially Kerr-type nonlinearity. Check Figure 10 to view an example of a soliton solution. We consider the potential term as given in [99].
\[
V_d(x, y) = \frac{-k}{2}(x^2 + y^2) - \xi \exp\left(-k(x^2 + y^2)\right).
\] (68)

The potential (68) represents non-periodic modulation of the linear refractive index in the transverse direction of the probe beam under a parabolic and Gaussian distribution. The exact solution represents the amplitude of the probe beam and is derived in [99] as

\[
u(x, y, t) = \xi \exp\left(-\frac{k}{2}(x^2 + y^2)\right) \exp(-ikt).
\] (69)

Figure 11 presents the solution computed for \(T \in [0, 4]\) over the domain \(\Omega \in [-2, +2] \times [-2, +2]\) with a grid of size 50 \(\times\) 50 nodal points. A cross-section of the simulation at \(y = 0\) is shown in Figure 10, and it depicts soliton evolution as a function of time.

In terms of SBP-projection formulation in two dimensions, the two-dimensional second derivative SBP-operator in both \(x\) and \(y\) is defined as follows

\[
\mathcal{D}_{xx} = \mathcal{D}_2 \otimes I_m, \quad \mathcal{D}_{yy} = I_m \otimes \mathcal{D}_2,
\]

\[
\Rightarrow \mathcal{D} = \mathcal{D}_{xx} + \mathcal{D}_{yy}.
\] (70)

Here, the operator \(\mathcal{D}\) is the SBP discrete equivalent of \(\nabla\).

The norm is constructed to match the two-dimensional domain as

\[
H_x = H \otimes I_m, \quad H_y = I_m \otimes H, \quad H = H_x H_y
\] (71)

For homogeneous Dirichlet boundary conditions, the discrete boundary operators are

\[
L_x = \left[(e_1 \otimes I_m)^T (e_m \otimes I_m)^T\right]^T, \quad L_y = \left[(I_m \otimes e_1)^T (I_m \otimes e_m)^T\right]^T.
\] (72)

Thus, the two-dimensional analog of the projection operator in Section 3 can be set as

\[
P_x = I_m - H_x^{-1} L_x^T (L_x H_x^{-1} L_x)^{-1} L_x, \quad P_y = I_m - H_y^{-1} L_y^T (L_y H_y^{-1} L_y)^{-1} L_y,
\]

\[
\Rightarrow P = P_x P_y.
\] (73)

Given the boundary data \(g_x\) for the left and right boundary, and \(g_y\) for the top and bottom boundary, the projected data are

\[
[BC]_x = H_x^{-1} L_x^T (L_x H_x^{-1} L_x)^{-1} \partial_x, \quad [BC]_y = H_y^{-1} L_y^T (L_y H_y^{-1} L_y)^{-1} \partial_y,
\]

\[
\Rightarrow [BC] = \mathcal{C}([BC]_x + [BC]_y).
\] (74)

where \(\mathcal{C}\) is a corrector operator for corner points as they are treated twice by \([BC]_x\), and \([BC]_y\). Here, it is an identity matrix except for the entries corresponding to the corner points where it holds the value of \(\frac{1}{2}\).

The SBP projection discretization of the problem is formulated as
\frac{\partial}{\partial t} \mathbf{V} = \mathbb{P} \left( \frac{i}{2} \mathbb{D} \mathbf{W} + i \mathbf{V}_d \otimes \mathbf{W} + i |\mathbf{W}|^2 \otimes \mathbf{W} \right) + \{\text{BC}\}, \quad \mathbf{W} = \mathbb{P} \mathbf{V} + \{\text{BC}\}. \quad (75)

Incorporating the time integrals into the formulation (75) gives the following, IMEX-formulation;

\left( \mathbb{I}_{m^2} - i \Delta t \alpha \mathbb{P}' \left( \frac{i}{2} \mathbb{D} \mathbf{V}_d \right) \right) \mathbf{V}^{n+1} = \mathbb{P} \left( \mathbf{V}^n + i \Delta t \sum_{j=1}^{n-1} a_{ij} (\frac{i}{2} \mathbb{D} \mathbf{V}(j) + \mathbf{V}_d \otimes \mathbf{V}(j)) + i \Delta t \sum_{j=1}^{n-1} a_{ij} |\mathbf{V}(j)|^2 \otimes \mathbf{V}(j) \right) + \{\text{BC}\}, \quad (76)

SBP-projection in time;

\mathbf{V} = \mathbf{V}_0 + \mathbb{J} \mathbb{F} \left( \frac{i}{2} \mathbb{D} \mathbf{P} \mathbf{V} + \frac{i}{2} \mathbb{D} \{\text{BC}\} + i \mathbf{V}_d \mathbf{V} + i |\mathbf{V}|^2 \otimes \mathbf{V} \right), \quad (77)

with the linear iterative approximation

\left( \mathbb{I}_{m^2} - [\mathbb{J} \mathbb{F}] \left( \frac{i}{2} \mathbb{D} \mathbf{P} + i \mathbf{V}_d \right) \right) \mathbf{V}^{n+1} = \mathbf{V}_0 - \mathbf{V}^n + [\mathbb{J} \mathbb{F}] \left( \frac{i}{2} \mathbb{D} \mathbf{P} + i \mathbf{V}_d \right) \mathbf{V}^n + \frac{i}{2} \mathbb{D} \{\text{BC}\} + i |\mathbf{V}|^2 \otimes \mathbf{V}^n, \quad (78)

SBP-SAT in time;

\mathbb{D}_t \mathbf{V} = -R \mathbf{V}_0 + \frac{i}{2} \mathbb{D} \mathbf{P} \mathbf{V} + \frac{i}{2} \mathbb{D} \{\text{BC}\} + i \mathbf{V}_d \mathbf{V} + i |\mathbf{V}|^2 \otimes \mathbf{V}, \quad (79)

with the linear iterative approximation

\left( \mathbb{D}_t - \frac{i}{2} \mathbb{D} \mathbf{P} - i \mathbf{V}_d \right) \mathbf{V}^{n+1} = -R \mathbf{V}_0 + \left( - \mathbb{D}_t + \frac{i}{2} \mathbb{D} \mathbf{P} + i \mathbf{V}_d \right) \mathbf{V}^n + \frac{i}{2} \mathbb{D} \{\text{BC}\} + i |\mathbf{V}|^2 \otimes \mathbf{V}^n. \quad (80)

Tables 10 and 11 present $L_2$ error and CPU time for IMEX, projection in time and SAT in time. Here, the SBP4-projection approximation method is used with an end time $t = 1$ over the domain $\Omega = [-2, +2] \times [-2, +2]$ with parameters’ values $\xi = 1$, $k = 1$. The boundary data are taken from the exact solution (69). The results show remarkably better numerical efficiency for LGL- and LG-quadrature-based time marching methods in comparison with RK4.

Comparing Tables 10 and 11, the IMEX(ESDIRK4-ERK4) scheme in contrast to the one-dimensional case shows considerably better efficiency against SBP schemes. This is explained in two folds. First and primal explanation; the IMEX scheme is employed here without a step controller, but instead, the largest possible time step that preserves optimal convergence is used. The time step is taken to be in the magnitude $\Delta t \approx 0.5h$, and $h$ is the mesh size. By experimentally finding the optimum step size and then using it to record measurements, the algorithm is spared from some arithmetic complexity that was otherwise occupied by the controller. The justification for removing the controller is that the two-dimensional error estimation using the embedded scheme was too large for the controller to choose a proportionally big step size. We postulate the need for a different or more adapted controller than PID-controller. However, this falls outside the scope of this study and hence was not investigated. Second, the time consumed by left-hand matrix decomposition in the SBP formulation does not scale linearly with the problem size. Recap that we are using matrix decomposition of the left-hand matrix in (26), (35) and (36). The problem size in SBP schemes is of order $k$ times the problem size in the IMEX scheme. $k$ refers to the
number of temporal grid points. Thus, the Efficiency gap observed between IMEX and the SBP time integrator in one dimension is expected to diminish if the gap in problem size is large enough.

<table>
<thead>
<tr>
<th>Grid</th>
<th>RK4</th>
<th>IMEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>1.737e-05</td>
<td>0.5938</td>
</tr>
<tr>
<td>101</td>
<td>1.1067e-06</td>
<td>9.0312</td>
</tr>
<tr>
<td>151</td>
<td>3.9819</td>
<td>2.2314e-07</td>
</tr>
<tr>
<td>201</td>
<td>4.0303</td>
<td>7.0999e-08</td>
</tr>
</tbody>
</table>

Table 10: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $\Omega = [-2, +2] \times [-2, +2]$ using the SBP4-projection approximation method in space. Parameters’ values are $\xi = 1$, and $k = 1$. The data set depicts a comparison between the IMEX(ESDIRK4-ERK4) integrator and RK4 as a benchmark.

<table>
<thead>
<tr>
<th>Grid</th>
<th>RK4</th>
<th>LGL</th>
<th>LG</th>
<th>SAT in time</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>1.737e-05</td>
<td>1.738e-05</td>
<td>0.2188</td>
<td>1.875e-05</td>
</tr>
<tr>
<td>101</td>
<td>1.1067e-06</td>
<td>1.1033e-06</td>
<td>1.9844</td>
<td>1.116e-06</td>
</tr>
<tr>
<td>151</td>
<td>3.9819</td>
<td>2.2314e-07</td>
<td>54.3438</td>
<td>7.3750</td>
</tr>
<tr>
<td>201</td>
<td>4.0303</td>
<td>7.0999e-08</td>
<td>185.4375</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $\Omega = [-2, +2] \times [-2, +2]$ using the SBP4-projection approximation method in space. Parameters’ values are $\xi = 1$, and $k = 1$. The data set is obtained using the linear approximation scheme and it compares the projection in time with the LGL- and LG-quadrature-based time marching methods against RK4 as a benchmark.
Figure 9: CPU time comparison between RK4 and IMEX-scheme on one side, and LGL- and LG-quadrature-based time marching methods on the other, as a function of $L_2$ error norm. Here, we use the linear approximation for both the projection and the SAT in time.

Figure 10: Three-dimensional profile of the cross-section of the solution $u(x, y, t)$ at $y = 0$. The plot is taken from simulation over the domain $\Omega = [-2, +2] \times [-2, +2]$ for end time $t = 4$ and $\xi = 1$, and $k = 1$. 
7.2 Gross–Pitaevskii with saturated breathing soliton

We proceed with another example of the Gross–Pitaevskii equation that is given with saturable nonlinearity [99]. The equation reads

\[
i \frac{\partial}{\partial t} u(x, y, t) = -\nabla u(x, y, t) + \frac{E_0}{1 + V_d(x, y) + |u(x, y, t)|^2},
\]

with initial data given by

\[
u(x, y, 0) = \sqrt{\xi} \exp\left(\frac{-k}{2}(x^2 + y^2)\right),
\]

and the function \(V_d(x, y)\) defined as

\[
V_d(x, y) = \frac{E_0 - k^2(x^2 + y^2)}{k^2(x^2 + y^2)} - \xi \exp(-k^2(x^2 + y^2)).
\]

Eq. (81) is used to model the propagation of a polarized probe beam where the solution represents the amplitude of the probe beam, and \(V_d\) is a lattice intensity function. \(E_0\) and \(k\) are constants. The exact solution is given in [99] to be

\[
u(x, y, t) = \sqrt{\xi} \exp\left(\frac{-k}{2}(x^2 + y^2 + 4it)\right).
\]

Numerical simulation of Eq. (81) is shown in Figure 14. The figure depicts the time evolution of cross sections at \(y = 0\) of a stationary breathing soliton. The simulation is run for end time \(t = 5\) over the domain \(\Omega = [-2, +2] \times [-2, +2]\) with inhomogeneous Dirichlet boundary conditions and parameter’s values \(E_0 = 1\), \(\xi = 1\) and \(k = 1\).
The construction of SBP formulation and time integration schemes follow the same steps concluded in the previous section. Table 12 presents the results for both LGL- and LG-quadrature-based time marching methods when using the linear iterative approximation of the SBP formulation. LGL-quadrature-based time marching method with the projection technique is found to yield the best results, while the other schemes appear to be comparable in terms of numerical efficiency. The findings are consistent with the results obtained in the previous section, Table 11.

Next, we consider solving the problem by employing an iterative method. Table 13 presents the results using the GMRES and FGMRES. we restrict the experimentation on the LGL-quadrature-based time marching method since this scheme in combination with the projection technique exhibited the best results throughout this work. The obtained measurements nevertheless show better numerical efficiency than RK4. However, the efficiency lags considerably behind when compared to the linear approximation in Table 12.

| Grid | q       | \(|e|_{L_2}^2\) | CPU   | \(|e|_{L_2}^2\) | CPU   | \(|e|_{L_2}^2\) | CPU   | \(|e|_{L_2}^2\) | CPU   |
|------|---------|----------------|-------|----------------|-------|----------------|-------|----------------|-------|
| 51   | 1.064e-05 | 2.2812         | 1.0260e-05 | 0.4844         | 1.0394e-05 | 0.7812         | 1.0503e-05 | 1.0312         | 9.7223e-06 | 1.0625   |
| 101  | 4.2590   | 5.8175e-07     | 36.2969 | 5.7757e-07     | 2.7656 | 5.7567e-07     | 6.5000 | 5.9048e-07     | 10.6313  | 7.1875   |
| 151  | 4.0423   | 1.1448e-07     | 309.0938 | 1.1280e-07     | 12.7031 | 1.1775e-07     | 40.1875 | 1.1279e-07     | 52.1406  | 33.5469  |
| 201  | 4.0887   | 3.5555e-08     | 1.0471e-04 | 3.5212e-08     | 30.8750 | 3.8066e-08     | 110.7812 | 3.5871e-08    | 101.6719 | 90.5625  |

Table 12: \(L_2\) error norm and CPU time taken at end time \(t = 1\) over the domain \(x \in [-2, +2]\) using the SBP4-projection approximation method in space. Parameters’ values are \(E_0 = 1\), \(\xi = 1\) and \(k = 1\). The data set is obtained using the linear approximation scheme and it compares the projection in time with the LGL- and LG-quadrature-based time marching methods against RK4 as a benchmark.
Figure 12: CPU time comparison between RK4 and LGL- and LG-quadrature-based time marching method as a function of $L_2$ error norm. Here, we use the linear iterative approximation for both the projection and the SAT in time.

Table 13: $L_2$ error norm and CPU time taken at end time $t = 1$ over the domain $x \in [-2, +2]$ using the SBP4-projection approximation method in space. Parameters' values are $E_0 = 1$, $\xi = 1$ and $k = 1$. The data set is obtained by employing the iterative solver on the non-linear SBP formulation. The data set depicts a comparison between the projection in time against the SAT in time with the LGL-quadrature-based time marching method.
Figure 13: CPU time comparison between RK4 and projection and SAT in time schemes with the LGL- and LG-quadrature-based time marching methods and GMRES and FGMRES iterative solvers, as a function of $L_2$ error norm.

Figure 14: Three-dimensional profile of the cross-section of the solution $u(x, y, t)$ at $y = 0$. The plot is taken from simulation over the domain $\Omega = [-2, +2] \times [-2, +2]$ for end time $t = 5$ and $E_0 = 1$, $\xi = 1$, and $k = 1$.

8 Conclusion and outlook

In this manuscript, we present numerical implementations and investigations of various SBP-based time integrators in the context of weakly dispersive nonlinear systems. More specifically, The NLS equation with various soliton and soliton-like solutions. We have shown that the SBP spatial dis-
cretization and the boundary treatment via the projection method represent a robust, accurate, and highly stable numerical method for treating the NLS equation. This was proven via numerical error convergence against analytical solutions and stability analysis via the energy method. The semi-discrete SBP-projection scheme can be numerically efficient in terms of CPU time and execution time if combined with the SBP-based time marching methods for time discretization. In addition, a demonstrative extension to two-dimensional NLS equations is also provided.

The focus of this work is the implementation of the SBP-based time marching methods and the investigation of their numerical efficiency with RK4 and ESDIRK-ERK-based IMEX as a standard of comparison. To the best of our knowledge, no investigation of the numerical efficiency of the SBP-time marching method has been conducted in the context of non-linear equations generally and weakly dispersive NLS-type equations specifically. The imposition of the initial data for the SBP-based time marching method can, equivalently to SBP for spatial discretization, be combined with either the SAT or projection technique. In contrast to the SAT imposition of the initial data, the projection technique represents a part of recent advances in the field and has not been studied extensively in the literature. Numerical measurements of CPU execution time for one-dimensional test cases show steadily comparable numerical efficiency for the projection and SAT techniques. However, the error-CPU plots are observed to diverge for the two-dimensional test cases. The projection technique is shown to be overwhelmingly more efficient for the LGL-quadrature-based time marching method in both one and two dimensions. Whereas, for the SAT technique, the GL-quadrature-based time marching method appears to be either slightly better or comparable in terms of efficiency. Thus far, we can not explain this disparity between the LGL- and GL-quadrature-based time marching methods. Overall, the LGL-quadrature-based time marching method with the projection technique proved to be the most ascendant in terms of numerical efficiency in all the test cases.

Numerical experimentation shows that in general it is easier to apply an iterative solver when using the projection in time due to the structure of the left-hand matrix. For instance, the preconditioning of GMRES with general preconditioners i.e. ilu, ilu-crout, and the incomplete Cholesky yields a much smaller condition number than SAT in time. Further, we experiment with taking the exact inverse of the left-hand matrix and applying numerical dropping on it to yield a sparse approximate inverse (SAI) preconditioner. Numerical dropping is achieved here rather in the simple way of turning nonzero matrix entries below a certain threshold into zeros. It was observed that the percentage of the numerically dropped entries is much larger with projection in time and the resulting preconditioner is much more sparse while still yielding a lower condition number compared to SAT in time. This can indicate more flexibility and efficiency in constructing preconditioners from the wide range of SAI algorithms in the literature. Preconditioning with SAI preconditioner is out of the scope of this work as the approach is only efficient for parallel implementation. A worth-noting observation is that the SAI matrices obtained from numerical dropping
appear to have the same structure which can be predicted i.e using a convolutional neural network. In the case of a parallel implementation, this enables the deployment of a class of ultra-fast SAI algorithms that require a priory pattern for the SAI matrix to be constructed i.e. in [101].

The implementation of SBP time marching methods as was mentioned previously is based on multi-block implementation. For all test cases in this work, the number of blocks required for i.e. LG- and LGL-time marching methods to get an optimal convergence was anywhere between 3 and 100 blocks depending on the problem size and type of problem. Generally, a more refined spatial mesh requires more blocks to obtain the smallest possible numerical error. This can be understood as the numerical error being a function of mesh size in both spatial and temporal domains. For a very small number of blocks, the method can be unstable. for instance, setting a very large spatial mesh while integrating over only one or two blocks leads to instability. The minimal permissible number of blocks before the method becomes unstable varies from problem to problem. Apart from the NLS in the semi-classical limit, the restriction was measured for all test cases to be around 3 to 5 blocks except for the test case in Subsection 6.3 that required up to 60 blocks. We recap that this test case simulates two interacting solitons and possesses a relatively strong nonlinear term. For a strongly nonlinear term, the contractivity condition of the linear iterative approximation might not hold which necessitates a smaller block size and hence a smaller step size to shrink the contribution of the nonlinear term. Interestingly, when solving the nonlinear system with GMRES, the minimal permissible number of blocks to preserve stability is only slightly smaller. On the other hand, the number of blocks required to reach the optimum convergence is still the same. This expounds the enormous numerical advantage gained with linear iterative approximation in contrast to using an iterative solver.

In the context of the IMEX method, the ESDIRK-ERK scheme possesses a relatively small linear stability region for its ERK part. This constitutes the likely limiting cause for the time step beside iterative convergence problems with Newton’s method. Throughout this work, the time step restriction in the case of RK4 was set around \( \Delta t \approx CLF(\Delta x)^2 \) which preserves the optimal convergence order. Whereas, the restriction for IMEX(ESDIRK-ERK) was observed to be around \( \Delta t \approx CLF\Delta x \). In comparison with the SBP-based time marching method, IMEX yielded better results in the two-dimensional test case. As was mentioned above, this is explained by the absence of the controller and instead using the experimentally realized optimum time step size. optimum time step is defined here as the largest possible time step that can be taken without sacrificing convergence order or accuracy.

It was observed during the numerical experimentation for the one-dimensional case that the controllers PI and PID produce a near-optimum step size from the estimated error via the embedded Runge–Kutta. However, the error estimation in the two-dimensional test case was as large as such that the resulting step size was less than ideal. Therefore, we conjecture the need for a modified
error controller that takes the dimensionality of the problem into account. However, this was not investigated and instead, we contented with selecting a fixed step size. Following the discussion in [87], it is mentioned that PI and PID controllers are effective for explicit methods. On the other hand, PC-type controllers which are good at capturing the dynamics of an implicit method are unadvised for IMEX schemes. Therefore, by employing a controller as a PID-controller, we are attempting to control the explicit part of the scheme.
References


