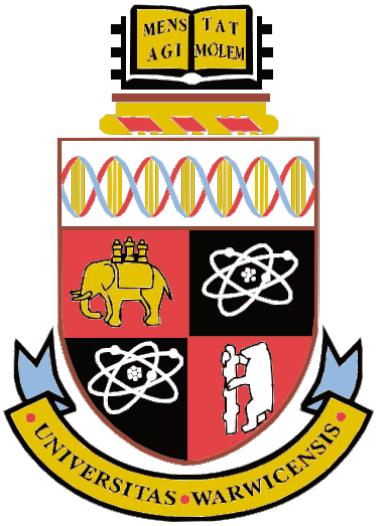

Krylov Subspace Methods for Exponential Integrators for Parabolic PDEs and Beyond

by

Harvey Sutton

MA4K8 Scholarly Report

Submitted to The University of Warwick

Mathematics Institute

April, 2000

Contents

1 Introduction

Parabolic PDEs can be used to model a wide array of physical phenomona in enginering and the natural sciences, such as heat flow, reaction diffusion and phase boundary problems. Therefore, the development of efficient and accurate methods by which numerical solutions to these PDEs can be computed, is of great importance. One possible approach is the use of exponential integrators. In order to illustrate these methods, consider briefly the following PDE:

$$u_t = \Delta u + f(u)$$

By discretizing in space by method of finite elements or finite difference, we can arrive at an ODE of the form.

$$u_{h,t} = -Au_h + f(u_h)$$

From here, we notice that we have a linear term Au_h where A is a matrix, u_h is a vector and a non-linear term $f(u_h)$. The exact solution to this equation can be written in the form:

$$u_h(t) = e^{-At}u_h(0) + \int_0^t e^{-(t-\tau)A} f(u_h(\tau))d\tau$$

as described by Hochbruck [10]. Exponential integrator methods attempt to approximate the above equation. The challenge for these systems is that for large A, computing its exponential can be too computationally demanding.

There exist many possible ways to compute approximations to the matrix exponential such as in Moler Et al's Nineteen Dubious Ways to Compute the Exponential of a Matrix[13]. Of most interest to us are the advancements in Krylov subspace methods such as the Arnoldi and Lanczos algorithms[14]. These methods work by projecting the matrix onto a smaller space, the Krylov subspace, where computing the matrix exponential is cheaper before projecting it back up to the full space. In this work, we will bring these two areas of exponential integrators and Krylov subspace approximations together, with the objective of finding more efficient and accurate numerical solutions to PDEs.

We will begin by introducing the exponential integrator schemes, followed by the Krylov methods, showing the connection between the two. Established analytical results for both Krylov and exponential integrator methods will be stated from the work of Saad, Y. [15] and Huang Et al. [11] respectivly. Following on from this, we provide our own analysis that ties these methods together, showing how the error from the Krylov methods and exponential integrators interact when used in conjunction with one another. Experimental results will also be provided, demonstrating performance and accuracy compared to other numerical methods, namely backwards Euler. We will then apply these combined methods

to phase boundary problems, in order to model dendritic crystal growth. Finally, we will show the application of these methods beyond parabolic PDEs.

2 Solution Schemes

Now, we outline the methods by which the ODEs arising from the discretization of PDEs in space can be solved. Consider the following PDE:

$$\dot{u}_t = N(u)$$

Discretizing in space gives the following:

$$\begin{aligned} M\dot{u}_h(t) &= N_h(u_h(t)) \\ \text{using } R(u_h(t)) &= N_h(u_h(t)) - DN(u_h(t))u_h(t) \text{ to denote the non-linear term gives} \\ M\dot{u}_h(t) &= DN(u_h(t))u_h(t) + R(u_h(t)) \\ \dot{u}_h(t) &= M^{-1}(DN(u_h(t))u_h(t) + R(u_h(t))) \end{aligned}$$

2.1 Methods

We present both conventional ODE solvers such as the forwards and backwards Euler method before moving on to the exponential integrator methods. For a time step τ we compute $u_h(t+\tau)$ in the following ways:

2.1.1 Forwards Euler

The forwards Euler method is given by:

$$u_h(t+\tau) = u_h(t) + \tau M^{-1} N_h(u_h(t))$$

2.1.2 Backward Euler

The backwards Euler method is given by:

$$u_h(t+\tau) = u_h(t) + \tau M^{-1} N_h(u_h(t+\tau))$$

As this method is implicit, we will need to employ a Newton solver.

2.1.3 Explicit Exponential Scheme

For the explicit exponential integrator schemes we use the following formula:

$$u_h(t+\tau) = e^{\tau M^{-1} DN(u_h(t))}(u_h(t) + M^{-1}R(u_h(t)))$$

2.1.4 First Order Exponential Integrator

Here, we also present another integrator from Huang Et al [11]

$$u_h(t+\tau) = e^{\tau M^{-1} DN(u_h(t))} u_h(t) + \tau \varphi_1(\tau M^{-1} DN(u_h(t))) R(u_h(t))$$

Where:

$$\varphi_k(z) = \int_0^1 e^{(1-\theta)z} \frac{\theta^{k-1}}{(k-1)!} d\theta, k \geq 1$$

2.1.5 Second Order Exponential Integrator

We also look at a second order exponential integrator [11].

$$\begin{aligned} &\text{writing } A = \tau M^{-1} DN(u_h(t)) \text{ for the sake of brevity} \\ &u_h(t+\tau) = e^A u_h(t) + \tau((\varphi_1(A)) - \frac{1}{c_2}\varphi_2(A)) R(u_h(t)) \\ &\quad + \frac{1}{c_2}\varphi_2(A) R(e^{c_2 A} u_h(t) + c_2 \tau \varphi_1(c_2 A) R(u_h(t))) \end{aligned}$$

Where $c \in (0, 1]$.

2.2 Mass Matrix

When using these schemes it is necessary to be able to compute the inverse mass matrix M^{-1}. Attempting to compute the exact inverse can be computationally intensive. While for a fixed spacial discretization this may be acceptable, as it will only need to be computed once, for an adaptive grid this may be impractical. As a result, we employ mass lumping where each row is summed up and placed on the diagonal of the matrix. From here, computing the inverse is straightforward.

3 Krylov Methods

When using the above exponential integrators, we need to be able to compute the matrix exponential e^A or, more precisely, the action of the exponential of a matrix to a vector $e^A v$. For a lower dimension, computing $e^A v$ is computationally cheap. However, when the dimension of the matrix A is large, then computing $e^A v$ is computationally intensive. One solution to this is to use Krylov subspace methods. These algorithms take a matrix $A \in \mathbb{R}^{n \times n}$, a vector $v \in \mathbb{R}^n$ and an integer m that determines the number of dimensions of the Krylov subspace used. From these algorithms, we will get a matrix $H \in \mathbb{R}^{m \times m}$ and another matrix $V \in \mathbb{R}^{n \times m}$ such that $A \approx VHV^T$ and $V^T V = I$. V contains basis vectors of the Krylov subspace $span(v, Av, ..., A^{m-1}v)$ and H is such that $V^T AV = H$. From here we get $Av \approx VHV^T v = VH||v||e_1$.

We can apply this to e^A as follows:

$$\begin{aligned} e^A &= \sum_{i=0}^{\infty} \frac{A^i}{i!} \\ &= \sum_{i=0}^{\infty} \frac{(VHV^T)^i}{i!} \\ &= \sum_{i=0}^{\infty} \frac{VH^iV^T}{i!} \end{aligned}$$

and then when computing $e^A v$ we get

$$\begin{aligned} e^A v &= (\sum_{i=0}^{\infty} \frac{VH^iV^T}{i!})v \\ &= \sum_{i=0}^{\infty} \frac{VH^iV^Tv}{i!} \\ &= \sum_{i=0}^{\infty} \frac{VH^i||v||e_1}{i!} \\ &= V(\sum_{i=0}^{\infty} \frac{H^i}{i!})||v||e_1 \\ &= Ve^H||v||e_1 \end{aligned}$$

We will still need to compute the exponential of the matrix H but for $m \ll n$ this will be computationally cheap.

When computing $\varphi_k(A)$, notice that the following holds:

$$\varphi_k(A)v = V\varphi_k(H)||v||e_1$$

where V, H are generated by one of the Krylov Methods above. We demonstrate this below:

$$\begin{aligned} \varphi_k(A)v &= \int_0^1 e^{(1-\theta)A} \frac{\theta^{k-1}}{(k-1)!} d\theta v \\ &= \int_0^1 e^{(1-\theta)A} d\theta v \\ &= \int_0^1 e^{(1-\theta)A} v d\theta \end{aligned}$$

applying a Krylov method giving $VH||v||e_1 \approx Av$ we get

$$= \int_0^1 Ve^{(1-\theta)H}||v||e_1 d\theta$$

$$= V \int_0^1 e^{(1-\theta)H} d||v||e_1$$
$$= V\varphi_k(H)||v||e_1$$

As a result of this, we will only need to genarate a single subspace for the numerical integrations.

It should also be noted, for the second order exponential integrator, that the Krylov space used to compute $(\varphi_1(A))R(u_h(t))$ is the same subspace used to $\tau\varphi_1(c_2A)R(u_h(t))$. This is also true for the computation of $e^A u_h(t)$ and $e^{c_2 A}u_h(t)$.

3.1 Algorithms

We now state the two Krylov subspace methods that we will be investigating, the Arnoldi and Lanzcos algorithms. Following this, we will investigate the benefits and drawbacks of each algorithm.

3.1.1 Arnoldi

Below we present the Arnoldi algorithm.

Algorithm 1 Arnoldi [8]

procedure ARNOLDI($A, \hat{v}_1, m$)
  $v_0 \leftarrow 0$
  for $j = 1, 2, ..., m$ **do**
    for $i = 1, 2, ..., j$ **do**
      $h_{ij} \leftarrow v_i^T A v_j$
    $\theta_j \leftarrow Av_j - \sum_{i=1}^{j} h_{ij} v_i$
    $h_{j+1,j} \leftarrow ||\theta_j||$
    $v_{j+1} \leftarrow \theta_j / h_{j+1,j}$

Here V is given by $v_1, ..., v_m$ and H is give by $h_1, ..., h_m$.

3.1.2 Lanczos

The algorithm below is the Lanzcos algorithm and it requires a symmetric matrix. [13]

Algorithm 2 Lanzcos[14]

procedure LANCZOS (A symmetric, $\hat{v}_1, m$)
  $v_0 = 0$
  for $i = 1, 2, ..., m$ **do**
    $\beta_i \leftarrow ||\hat{v}_i||$
    $v_i \leftarrow \hat{v}_i / ||\hat{v}_i||$
    $\alpha_i \leftarrow v_i^T A v_i$
    $\hat{v}_{i+1} \leftarrow A v_i - \alpha_i v_i - \beta_i v_{i-1}$

Here V is given by $v_1, ..., v_m$ and H is tridiagonal with the leading diagonal being $\alpha_1, ..., \alpha_m$ and the upper and lower diagonals being $\beta_2, ..., \beta_m$.

It should be noted that most of the computational cost arrises from applying A to a vector. For both algorithms this is only computed once per step, meaning that in total m matrix vector products are required.

3.1.3 Benefits and Drawbacks

The main difference between the above algorithms is that the Lanczos algorithm requires A to be a symmetric matrix, whereas the Arnoldi algorithm does not have this requirement. This does allow the Arnoldi algorithm to have more broad applications than the Lanczos method. However, the Lanczos algorithm is faster than the Arnoldi algorithm as a result of the internal loop in the Arnoldi algorithm. As a result, it is necessary to carefully select which method to use. This will depend upon the problem that we are working on.

3.2 Numerical Analysis

Here we present results from Saad[15]. This shows the convergence rates for the Arnoldi method.

Theorem 3.1. *Let A be any square matrix and let $\rho = ||A||_2$ then the error of the approximation of $e^{\tau A}v$ is such that*

$$||e^{\tau A}v - \beta V_m e^{H_m} e_1||_2 \leq 2\beta \frac{(\tau\rho)^m e^{\tau\rho}}{m!}$$

where

$$\beta = ||v||_2$$

From here, we see that the approximation gets more accurate for smaller values of τ, as well as for larger values of m given that $\tau\rho \leq 1$.

3.3 Numerical Accuracy and Performance

We now investigate how these Krylov methods compare to existing methods for computing the action of the matrix exponential to a vector such as those used in SciPy under

`scipy.sparse.linalg.expm_multiply`[1][9].

We will compare timings, as well as the required depth of the Krylov subspace necessary for accurate computations. The matrix $A \in \mathbb{R}^{n \times n}$ being used is sparse and tridiagonal with 2τ along the leading diagonal and $-\tau$ along the upper and lower diagonal for some $\tau > 0$. We use this matrix as matrices of this sort arrise for second order spacial derivatives, such as with the finite difference method. We use SciPy to compute a reference solution and then compare this to the approximation produced by the Krylov methods. The error is the Euclidian distance between the reference solution and the approximate solution. We will compare for a variety of matrix sizes and values of τ

Below, we compare the error to the dimension of the the Krylov subspace m.

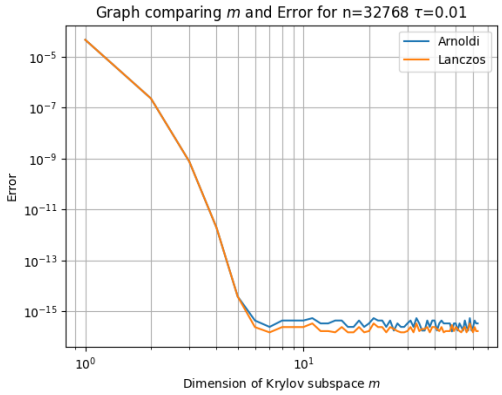

Figure 1: m vs error with matrix size $n = 32768$ and $\tau = 0.01$

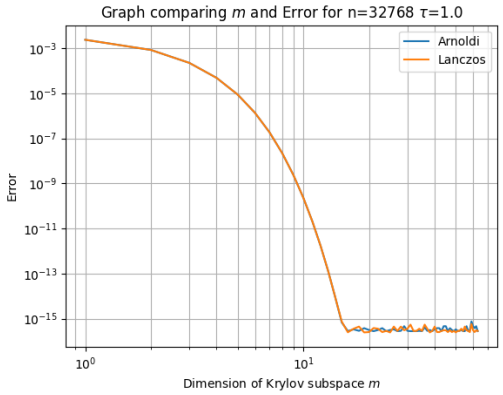

Figure 2: m vs error with matrix size $n = 32768$ and $\tau = 1$

We see the error decreases for larger Krylov subspaces before levelling out, demonstrating convergence with the SciPy reference solution. For $\tau = 0.01$, we see that a smaller Krylov subspace of around 6 is needed in order to achieve this convergence whereas for $\tau = 1$ a subspace of depth about 16 is sufficient. This is expected as shown in the theorem above.

Here, we observe the relationship between the computation time and the dimension of the Krylov subspace m.

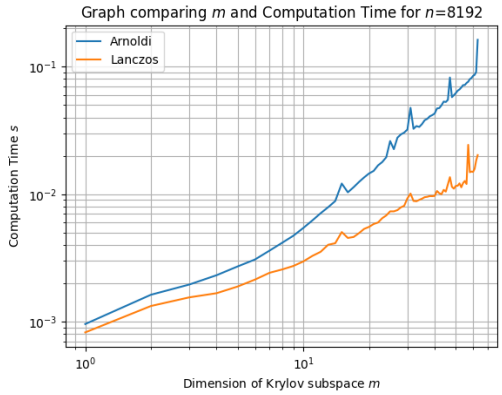

Figure 3: m vs computation time with matrix size 8192

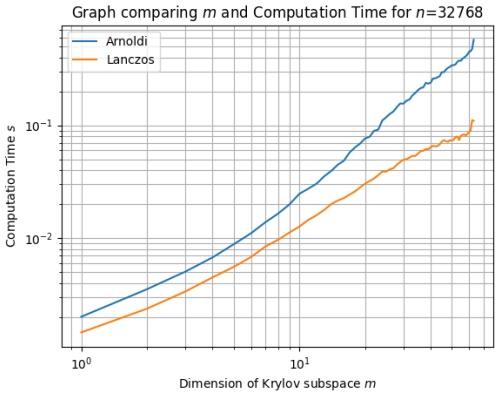

Figure 4: m vs computation time with matrix size 32768

With $n = 32768$ and a Krylov size of 10 the Lanczos and Arnoldi methods required just over 10^{-2} and 2×10^{-2} seconds respectively, whereas for a Krylov size of 50 the computation time was 8×10^{-2} and 3×10^{-1} respectively. As expected, the larger the Krylov subspace, the longer the time to compute. We also see that the Lanczos method performs faster the the Arnoldi method, again, as expected. The difference in performance appears to be more pronounced at larger Krylov sizes, with the Arnoldi method growing faster than the Lanczos method. This is due to the extra computations from the additional internal loop of the Arnoldi method. It should also be noted that the matrix being used is sparse, as it is tridiagonal, and as a result, extra compute time from the internal loop may not be as pronounced as for matrices that are dense.

3.4 Further Implementational Aspects

Throughout both the Arnoldi and Lanzcos algorithms the computation of $DN(u_h(t))v$ for some vector v is required. One possible approach is to comptute $DN(u_h(t))$ from $N(u_h)$ using automatic differentiation. This process is handled by the use of UFL[2] and DUNE[3]. From here, computing $DN(u_h(t))v$ is straightforward.

The computation, storage and hence use of an explicitly computed $DN(u_h(t))$ can be very demanding. As we only need $DN(u_h(t))v$ we can avoid needing to assemble $DN(u_h(t))$ using matrix free methods with the following approximations:

$$\begin{aligned} Au &= DN(u)u \\ &\approx \frac{N(u(1+\epsilon)) - N(u)}{\epsilon} \end{aligned}$$

for some small ϵ.

4 Numerical Analysis

In this section, we will look into the numerical analysis of the exponential integrators above, combined with the Krylov methods. We will begin by presenting some already established results for these methods. Next, we will attempt to extend these results in order to quantify the error produced when using the Krylov methods for approximating the action of the exponential of a matrix to a vector. We will use $x \lesssim y$ to denote $x \leq Cy$ for some $C > 0$ independent of x and y.

4.1 Formulation of the Problem

Here we state the problem as formulated by Huang Et al.[11]. We begin by considering a parabolic PDE with initial condition of $u_0 \in H^2(\Omega) \cap H_0^1(\Omega)$ of the form:

$$\begin{aligned} u_t &= D\Delta u + f(t,u) && x \in \Omega, 0 \leq t \leq T \\ u(0,x) &= u_0(x) && x \in \Omega \\ u(t,x) &= 0 && x \in \partial\Omega, 0 \leq t \leq T \end{aligned}$$

where T is the final time, Ω is a rectangular domain in $\mathbb{R}^d$, $D \geq 0$, $u(t,x)$ is the unknown function and $f(t,u)$ is a non-linear term.

The variational form of this problem is to find $u \in L^2(0,T;H_0^1(\Omega))$ and $u_t \in L^2(0,T : L^2(\Omega))$ such that

$$\begin{aligned} (u_t, v) + a(u,v) &= (f(t,u), v) \quad \forall v \in H_0^1(\Omega), 0 \leq t \leq T \\ u(0) &= u_0 \end{aligned}$$

where $(\cdot,\cdot)$ is the L^2 inner product on Ω. The bilinear form $a(\cdot,\cdot)$ is given by

$$a(w,v) = \int_\Omega D\nabla w \cdot \nabla v dx, \quad \forall w, v \in H_0^1(\Omega)$$

and is clearly symmetric.

We will now begin to formulate the finite element space V_h with which we will approximate H_0^1. As $\Omega \in \mathbb{R}^d$ is a rectangular domain, we can assume that $\bar{\Omega} := \prod_{i=1}^d [a_i, b_i]$. For every $i = 1, ..., d$, we can define a uniform partition of $[a_i, b_i]$ with the subinterval size $h_i = \frac{b_i - a_i}{N_i}$, to get the nodes $x_i^j = a_i + jh_i, j = 0, ..., N_i$ as $a_i = x_i^0 < x_i^1 < ... < x_i^{N_i} = b_i$. With this regular partition, we obtain a one-dimensional continuous piecewise linear finite element space for $[a_i, b_i]$. As

$$\begin{aligned} V_{h_i}^i(a_i,b_i) :&= \{v \in C[a_i,b_i] : v|_{[x_i^{j-1}, x_i^j]} \in \mathbb{P}_1([x_i^{j-1}, x_i^j]), 1 \leq j \leq N_i\} \cap H_0^1(a_i,b_i) \\ &= \text{span}\{\theta_i^1(x_i), ..., \theta_i^{N_i-1}(x_i)\} \end{aligned}$$

where $\theta_i^j(x_i)$ is the j-th nodal basis function of $V_h^i(a_i, b_i)$. Now we can write a finite element space for Ω as follows:

$$\begin{aligned} V_h &:= V_{h_1}^1(a_1, b_1) \otimes ... \otimes V_{h_d}^d(a_d, b_d) \\ &= \text{span}\{\theta_1^{i_1} \cdots \theta_d^{i_d}(x_d) : 1 \leq i_1 \leq N_1 - 1, ..., 1 \leq i_d \leq N_d - 1\} \end{aligned}$$

We now have that $V_h \subset H_0^1(\Omega)$. Let $h = \max_{1 \leq i \leq d} h_i$ be the mesh size of the corresponding uniformly rectangular partition $\mathcal{T}_h$ that generates V_h. All error analysis will assume $h \approx h_i$ for all i.

The finite element approximation for the above problem is then to find $u_h \in L^2(0, t : V_h)$ so that:

$$\begin{aligned} (u_{h,t}, v_h) + a(u_h, v_h) = (f(t, u_h), v_h) \forall v_h \in V_h, 0 \leq t \leq T \\ u_h(0) = P_h u_0 \end{aligned}$$

where the L^2 orthogonal projection operator is given by $P_h : L^2(\Omega) \to V_h$. This projection is stable with respect to the L_2, that is to say that $||P_h u||_0 \lesssim ||u||_0$ and $||P_h u||_1 \lesssim ||u||_1$.

The inverse inequality for finite elements (lemma 4.5.3 [6]) states that when $V_h \in H^1 \cap H^0$ and $0 < h \leq 1$, the following holds:

$$|v|_1 \lesssim h^{-1} ||v||_0$$

From this, we know:

$$a(w_h, v_h) \lesssim |w_h|_1 |v_h|_1 \lesssim h^{-2} ||w_h||_0 ||v_h||_0, \quad \forall w_h, v_h \in V_h \tag{1}$$

where the constants are independent of h. This shows that $a(\cdot, \cdot)$ is a bounded linear form over V_h with respect to the L_2 norm.

From here we can apply Riesz representation theorem to deduce that there exists a bounded linear operator $L_h : V_h \to V_h$ such that

$$a(w_h, v_h) = (L_h w_h, v_h), \quad \forall w_h, v_h \in V_h$$

Using the projection operator P_h, the problem can be rewritten to the equivalent system:

$$\begin{aligned} u_{h,t} + L_h u_h = P_h f(t, u_h) \\ u_h(0) = P_h u_0 \end{aligned}$$

4.2 Established Results

We now state the results given by Huang, et al[11]. Beginning with some assumptions:

Assumption 4.1. The function $f(t,\zeta)$ grows mildly with respect to ζ. That is there exists a number $p > 0$ for $d = 1, 2$ or $p \in (0, 2]$ for $d = 3$ such that:

$$|\frac{\partial f}{\partial \zeta}| \lesssim 1 + |\zeta|^p. \quad \forall t, \zeta \in \mathbb{R}$$

Assumption 4.2. The function $f(t,\zeta)$ is sufficiently smooth with respect to t. That is to say that for any given constant $K > 0$ the following holds:

$$\sum_{|\alpha| \leq 2} |D^\alpha f(t,\zeta)| \lesssim 1, \quad \forall t \in [0,T], \zeta \in [-K, K]$$

Assumption 4.3. The exact solution u(t) satisfies the following regularity conditions:

$$\sup_{0 \leq t \leq T} ||u(t)||_{2,\Omega} \lesssim 1,$$
$$\sup_{0 \leq t \leq T} ||u_t(t)||_{L^\infty(\Omega)} \lesssim 1$$

Where the hidden constant may depend on T.

Assumption 4.4. The exact solution u(t) satisfies the following regularity conditions:

$$\sup_{0 \leq t \leq T} ||u_{tt}(t)||_{L^\infty(\Omega)} \lesssim 1$$

where the hidden constant may depend on T.

We can now state the theorem that gives the error for the first order exponential integrator scheme.

Theorem 4.5. *Suppose the function f satisfies Assumptions 4.1 4.2 4.3. There exists a constant $h_0 > 0$ such that for $h < h_0$, we have the numerical solution given by:*

$$u_h^{n+1} = e^{-\tau L_h} u_h^n + \tau \varphi_1(-\tau L_h) P_h f(t_n, u_h^n)$$

satisfies the following:

$$||u(t_n) - u_h^n||_1 \lesssim \tau + h, \forall n = 1, ..., N_T$$

with a hidden constant that is independent of τ and h.

We also present the results for the second order exponential integrator scheme.

Theorem 4.6. *Suppose the function f satisfies Assumptions 4.1 4.2 4.3 and 4.4. There*

exists a constant $h_0 > 0$ such that for $h < h_0$, we have the numerical solution given by:

$$u_h^{n+1} = e^{-\tau L_h} u_h^n + \tau(\varphi_1(-\tau L_h) - \frac{1}{c}\varphi_2(-\tau L_h))P_h f(t_n, u_h^n) + \frac{1}{c}\varphi_2(-\tau L_h)P_h f(t_n + c\tau, e^{-c\tau L_h} u_h^n + c\tau\varphi_1(-c\tau L_h)P_h f(t_n, u_h^n))$$

satisfies the following:

$$||u(t_n) - u_h^n||_1 \lesssim \tau^2 + h, \forall n = 1, ..., N_T$$

with a hidden constant that is independent of τ and h.

4.3 Extension to Krylov Methods

In this section, we will prove the main theorem that ties the error of the first order method to the Krylov methods. We begin by stating the lemmas that we will need to prove this result. We omit the proof of lemmas already proven by Huang[11].

Lemma 4.7. *Suppose that the function f satisfies assumption 4.1, and the exact solution satisfies 4.3. Then we have that f is locally-Lipschitz continuous in a strip along the exact, i.e. for some $\epsilon > 0$ we have:*

$$||f(t, v) - f(t, w)||_0 \lesssim ||v - w||_1$$

for any $t \in [0, T]$ and $v, w \in V_h$ satisfying

$$max\{||v - u(t)||_1, ||w - u(t)||_1\} \leq \epsilon$$

Lemma 4.8. *Consider a Krylov approximation given by $V_m^T A V_m = H$ with respect to a vector v of depth m. If A is negative semi-definite then so is H.*

Proof.

$$\begin{aligned} x^T H x &= x^T V_m^T A V_m x \\ &= y^t A y \text{ where } y = V_m x \\ &\leq 0 \end{aligned}$$

□

Definition 4.9. We let e_m^A denote the Krylov approximation of e^A with respect to some vector v with an m dimensional Krylov subspace.

Lemma 4.10. *We have that $||e_m^A|| \leq m$ when A is negative semi-definite.*

Proof.

$$\begin{aligned}||e_m^A||_{op} &= ||e^{VHV^T}||_{op} \\ &= ||V_m e^H V_m^T||_{op} \\ &\leq ||V_m||_{fro}||e^H||_{op}||V_m^T||_{fro}\end{aligned}$$

We use the fact that A and hence H is negative semi-definite to imply that all of the eigen values λ of H are such that $\lambda \leq 0$ and hence we have that $||e^H||_{op} \leq e^0 = 1$. From here we see that

$$\leq ||V_m||_{fro}||V_m^T||_{fro}$$

we now use the fact that the m columns of V_m are orthonormal to get

$$\begin{aligned}&= \sqrt{m}\sqrt{m} \\ &= m\end{aligned}$$

□

Lemma 4.11. *$-\tau L_h$ is negative semidefinite. This was briefly shown as part of a lemma previously by Huang [11]. However, we formally show it here.*

Proof. As $a(\cdot,\cdot)$ a symmetric bilinear operator, it can be seen that L_h is a linear and symmetric operator that acts on V_h. We also have by the Poincare inequality that for some $\alpha > 0$:

$$\alpha||v_h||_0^2 \leq a(v_h, v_h), \quad \forall v_h \in V_h$$

Using this along with (1) gives

$$0 \leq \alpha \leq \lambda \quad \forall \lambda \in \lambda(L_h)$$

by Rayleigh representation theorem, where we have that $\lambda(L_h)$ denotes the set of eigen values of L_h. From here we see that the eigen values of $-\tau L_h$ are negative and hence $-\tau L_h$ is negative semidefinite. □

Definition 4.12. We write $u_{h,m}^n$ to denote the nth step of the exponential integrator method that uses a Krylov subspace of dimension m. We have that:

$$u_{h,m}^n = e_m^{-\tau L_h} u_{h,m}^{n-1} + \tau \int_0^1 e_m^{-\tau L_h(1-\theta)} d\theta P_h f(t, u_{h,m}^{n-1})$$

We can associate L_h with a matrix that operates on the basis of V_h

Lemma 4.13. *We claim that $u_{h,m}^n$ satisfies the following condition:*

$$||u_{h,m}^n||_1 \lesssim (m-\tau m)^{\frac{1}{\tau}} + m(1+(1-\tau)\tau m)$$

and hence

$$||P_h f(t, u_{h,m}^n)||_1 \lesssim (m-\tau m)^{\frac{1}{\tau}} + m(1+(1-\tau)\tau m) + C$$

Proof.

$$\begin{aligned}
||u_{h,m}^n||_1 &= ||e_m^{-\tau L_h} u_{h,m}^{n-1} + \tau \int_0^1 e_m^{-\tau L_h(1-\theta)} d\theta P_h f(t, u_{h,m}^{n-1})||_1 \\
&\leq ||e_m^{-\tau L_h} u_{h,m}^{n-1}||_1 + ||\tau \int_0^1 e_m^{-\tau L_h(1-\theta)} d\theta P_h f(t, u_{h,m}^{n-1})||_1
\end{aligned}$$

We now apply lemmas 4.11 and 4.10 to show that

$$\leq m||u_{h,m}^{n-1}||_1 + \tau m||P_h f(t_{n-1}, u_{h,m}^{n-1})||_1$$

we have that $||P_h f(t,u)||_1 \lesssim ||u||_1 + ||P_h f(t,0)||_1$ by Lipschitz continuity of f giving

$$\begin{aligned}
&\lesssim m||u_{h,m}^{n-1}||_1 + \tau m(||u_{h,m}^{n-1}||_1 + ||P_h f(t,0)||_1) \\
&\lesssim (m-\tau m)^n ||u_{h,m}^0||_1 + \tau m||P_h f(t,0)||_1 (n+(n-1)\tau m) \\
&\lesssim (m-\tau m)^n + \tau m(n+(n-1)\tau m) \\
&\lesssim (m-\tau m)^{\frac{1}{\tau}} + \tau m(\frac{1}{\tau} + (\frac{1}{\tau}-1)\tau m) \\
&\lesssim (m-\tau m)^{\frac{1}{\tau}} + m(1+(1-\tau)\tau m)
\end{aligned}$$

with the bound on $P_h f(t, u_{h,m}^n)$ being the result of Lipschitz continuity by taking $P_h f(t,0) \lesssim C$. This assumes that $P_h f(t,0)$ is bounded for all t which I think should follow from assumption 4.2. □

Definition 4.14. The error from the Krylov subspace approximation is given by $k(m, \tau, L_h)$. Associating L_h with a matrix that operates on the basis of V_h We define $k(m, \tau, L_h)$:

$$k(m, \tau, L_h) = max_{||v||_1=1} ||e^{\tau A} v - V_m e^{H_m} V_m^T v||_1$$

Where V_m and H_m are generated by our Krylov methods.

Lemma 4.15. *The error between $u_{h,m}^n$ and u_h^n is given by:*

$$||u_{h,m}^n - u_h^n||_1 \lesssim k(m, \tau, L_h)(m^{\frac{1}{\tau}} + m + 1 + \frac{1}{\tau})$$

where $k(m,\tau,L_h)$ denotes the error from the Krylov approximation that is dependant on m,τ,L_h.

Proof.

$$
\begin{aligned}
||u_{h,m}^n - u_h^n||_1 &= ||e_m^{-\tau L_h} u_{h,m}^{n-1} - e^{-\tau L_h} u_h^{n-1} \\
&+ \tau(\int_0^1 e_m^{-\tau L_h(1-\theta)} d\theta P_h f(t, u_{h,m}^{n-1}) - \int_0^1 e^{-\tau L_h(1-\theta)} d\theta P_h f(t, u_h^{n-1}))||_1 \\
&= ||(e_m^{-\tau L_h} - e^{-\tau L_h}) u_{h,m}^{n-1} \\
&+ e^{-\tau L_h}(u_{h,m}^{n-1} - u_h^{n-1}) \\
&+ \tau(\int_0^1 (e_m^{-\tau L_h(1-\theta)} - e^{-\tau L_h(1-\theta)}) d\theta P_h f(t, u_{h,m}^{n-1}) \\
&+ \int_0^1 e^{-\tau L_h(1-\theta)} d\theta P_h f(t, u_{h,m}^{n-1} - P_h f(t, u_h^{n-1})))||_1
\end{aligned}
$$

using Lipschitz continuity of f from lemma 4.7 and triangle inequality

$$
\begin{aligned}
&\lesssim ||(e_m^{-\tau L_h} - e^{-\tau L_h}) u_{h,m}^{n-1}||_1 \\
&+ ||e^{-\tau L_h}(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&+ \tau|| \int_0^1 (e_m^{-\tau L_h(1-\theta)} - e^{-\tau L_h(1-\theta)}) d\theta P_h f(t, u_{h,m}^{n-1})||_1 \\
&+ \tau|| \int_0^1 e^{-\tau L_h(1-\theta)} d\theta (u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&\lesssim ||(e_m^{-\tau L_h} - e^{-\tau L_h}) \frac{u_{h,m}^{n-1}}{||u_{h,m}^{n-1}||_1}||_1 ||u_{h,m}^{n-1}||_1 \\
&+ ||e^{-\tau L_h}|| ||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&+ \tau|| \int_0^1 (e_m^{-\tau L_h(1-\theta)} - e^{-\tau L_h(1-\theta)}) d\theta \frac{P_h f(t, u_{h,m}^{n-1})}{||P_h f(t, u_{h,m}^{n-1})||_1} ||||P_h f(t, u_{h,m}^{n-1})||_1 \\
&+ \tau|| \int_0^1 e^{-\tau L_h(1-\theta)} d\theta ||||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&\lesssim k(m,\tau,L_h)||u_{h,m}^{n-1}||_1 \\
&+ ||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&+ \tau k(m,\tau,L_h)||P_h f(t, u_{h,m}^{n-1})||_1 \\
&+ \tau||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&\lesssim k(m,\tau,L_h)||u_{h,m}^{n-1}||_1 \\
&+ (1+\tau)||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&+ \tau k(m,\tau,L_h)(||u_{h,m}^{n-1}||_1 + C) \\
&\lesssim (1+\tau) k(m,\tau,L_h)(||u_{h,m}^{n-1}||_1 + C)
\end{aligned}
$$

$$
\begin{aligned}
&+ (1+\tau)||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&\lesssim (1+\tau)k(m,\tau,L_h)((m-\tau m)^{\frac{1}{\tau}} + m(1+(1-\tau)\tau m) + C) \\
&+ (1+\tau)||(u_{h,m}^{n-1} - u_h^{n-1})||_1 \\
&\lesssim \sum_{i=1}^{n}[(1+\tau)^i]k(m,\tau,L_h)((m-\tau m)^{\frac{1}{\tau}} + m(1+(1-\tau)\tau m) + C) \\
&\lesssim k(m,\tau,L_h)(m^{\frac{1}{\tau}}(1-\tau 1)^{\frac{1}{\tau}} + m(1+(1-\tau)\tau m) + C)\frac{1-(1+\tau)^{\frac{1}{\tau}+1}}{\tau}
\end{aligned}
$$

we observe that $(1-\tau)^{\frac{1}{\tau}} \leq 1$, that $m(1+(1-\tau)\tau m) \lesssim m$ for $0 \leq \tau \leq 1$ and $\frac{1-(1+\tau)^{\frac{1}{\tau}+1}}{\tau} \lesssim 1 + \frac{1}{\tau}$

$$
\lesssim k(m,\tau,L_h)(m^{\frac{1}{\tau}} + m + 1 + \frac{1}{\tau})
$$

□

We can now prove the main theorem

Theorem 4.16. *Suppose the function f satisfies Assumptions 4.1 4.2 4.3. There exists a constant $h_0 > 0$ such that for $h < h_0$ we have*

$$
||u(t_n) - u_{h,m}^n||_1 \lesssim \tau + h + k(m,\tau,L_h)(m^{\frac{1}{\tau}} + m + 1 + \frac{1}{\tau}), \forall n = 1, .., N_T
$$

Proof.

$$
\begin{aligned}
||u(t_n) - u_{h,m}^n||_1 &= ||u(t_n) - u_h^n + u_h^n - u_{h,m}^n||_1 \\
&\leq ||u(t_n) - u_h^n||_1 + ||u_h^n - u_{h,m}^n||_1
\end{aligned}
$$

by the triangle inequality. We can now use theorem 4.5 and lemma 4.15 to see that

$$
\leq \tau + h + k(m,\tau,L_h)(m^{\frac{1}{\tau}} + m + 1 + \frac{1}{\tau}), \forall n = 1, ..., N_T
$$

□

Corollary 4.17. *Using the above theorem alongside what we already know about the convergence rates of Krylov subspaces we can observe the following:*

$$
||u(t_n) - u_{h,m}^n||_1 \lesssim \tau + h + 2\beta\frac{(\tau\rho)^m e^{\tau\rho}}{m!}(m^{\frac{1}{\tau}} + m + 1 + \frac{1}{\tau}), \forall n = 1, .., N_T
$$

where $||L_h|| \lesssim \rho$.

5 Numerical Experiments

5.1 Travelling Wave Allen Cahn

In this section, we run numerical experiments comparing the backward Euler method to Krylov methods for computing solutions to a PDE. Specifically, we will investigate the initial boundary value problem for the Allen Cahn equation on a strip with a travelling wave solution[16] given by:

$$\hat{u}(x,y,t) = \frac{e^{\frac{x-ct}{\sqrt{2}}}}{1+e^{\frac{x-ct}{\sqrt{2}}}}$$
$$\text{where } c = \frac{\sqrt{2}}{4}$$

and where the problem is defined as follows:

$$u_t = \Delta u + u(u-\frac{1}{4})(1-u) \ (x,y,t) \in \mathbb{R} \times (-1,1) \times [0,8]$$
$$u_0(x,y) = \hat{u}(x,y,0)$$

With homogeneous Neumann boundary conditions along the top and bottom of the domain.

However, for our numerical experiments we compute on the space $\Omega = (-16,16) \times (-1,1)$ and enforce the following Neumann boundary conditions along the left and right boundaries.

$$\nabla u \cdot \hat{n} = g \text{ for } g = \nabla \hat{u} \cdot \hat{n}$$

Where $\hat{n}$ is the unit normal on the boundary.

Where u is the solution, we write it in weak form:

$$\dot{u} = \Delta u + u(u-\frac{1}{4})(1-u)$$
$$\int_\Omega \dot{u}v = \int_\Omega \Delta u v + u(u-\frac{1}{4})(1-u)v \text{ for a smooth function } v \in C^\infty(\overline{\Omega})$$
$$\int_\Omega \dot{u}v = \int_\Omega u(u-\frac{1}{4})(1-u)v - \nabla u \cdot \nabla v + \int_{\partial\Omega} v\hat{n} \cdot \nabla \hat{u}$$

We now introduce our finite element space V_h for a spacial discretization size of $h \in \mathbb{R}$. We can now substitute in $u_h(t,x,y) = \sum_i u_i(t)v_i(x,y)$ where $u_i \in \mathbb{R}$ and $v_i \in V_h$ for

$i, j = 0, 1, ..., dim(V_h)$.

$$\text{we get } \sum_i \int_\Omega \dot{u}_i v_i v_j = \sum_i \Big(\int_\Omega u_i v_i (u_i v_i - \frac{1}{4})(1 - u_i v_i) v_j - \int_\Omega \nabla(u_i v_i) \cdot \nabla v_j + \int_{\partial\Omega} v_j \hat{n} \cdot \nabla \hat{u}_i v_i \Big)$$

$$\sum_i u_i \int_\Omega v_i v_j = -u_i \sum_i \Big(\int_\Omega \nabla v_i \cdot \nabla v_j + \int_\Omega R(u_h) v_j \Big)$$

$$\text{where we have } R(u) = u(u - \frac{1}{4})(1 - u) + \int_{\partial\Omega} \hat{n} \cdot \nabla \hat{u}$$

The above is true for all i, j. Writing in matrix vector form gives

$$M\underline{\dot{u}} = DN(\underline{u})\underline{u} + R(\underline{u})$$

$$\text{where } M_{i,j} = \int_\Omega v_i v_j dx$$

In the tests, we will investigate the relationship between step size, L_2 error given by $||\hat{u} - u_h||_{L_2}$ and calls to the operator N which will be a crude approximation to the computational cost. We compare the backward Euler, first and second order exponential methods over grid sizes: 128×8 and 256×8 and with time step $\tau = 8, 4, 2, 1, 0.5, 0.25, 0.125, 0.0625, 0.03125$.

5.1.1 Experimental Results

Below, we show the L_2 error with respect to the time step τ. Where in the legend EXP1LAN 16 denotes the first order method with equipped with the Lanzcos algorithm, Krylov size 16, likewise for EXP2LAN and the second order method. BE denotes the backwards Euler method.

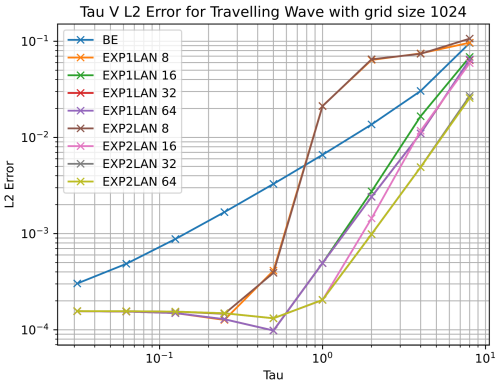

Figure 5: τ vs L_2 with grid size 1024

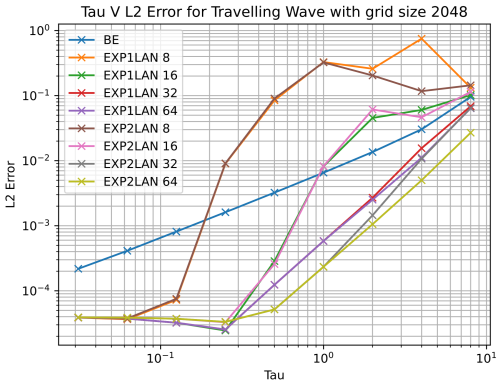

Figure 6: τ vs L_2 with grid size 2048

Firstly, note that all of the methods tested converge, tending towards an apparent minimum error, with the exponential methods levelling out. This minimum error is lower for a finer spacial discretization suggesting that the minimum error is the result of the spacial discretization. The Krylov methods converge faster than the backwards Euler method for

a Krylov size equal to or larger than 32 as shown by the steeper graph. The performance of the first and second order methods seem simillar both in terms of error and rate of convergence. We notice that when a smaller Krylov subspace, is used the exponential methods only start to converge for a sufficiently small τ. Likewise, when these methods start to converge, we observe that they perform similarly to the methods using a larger subspace. However, they will require fewer operator calls. This suggests that, for sufficiently small timesteps, a shallower subspace may suffice. In contrast, larger timesteps require a deeper Krylov subspace. As a result, it is important to look into whether it is more efficient to have a smaller Krylov subspace and τ or a larger space with corresponding larger τ.

Next we will look into operator calls in order to quantify this difference in performance. We compare the error with the number of calls to the operator. These operator calls are needed for computing N which is used with the matrix free methods as described above as well as for computing the non-linear term R.

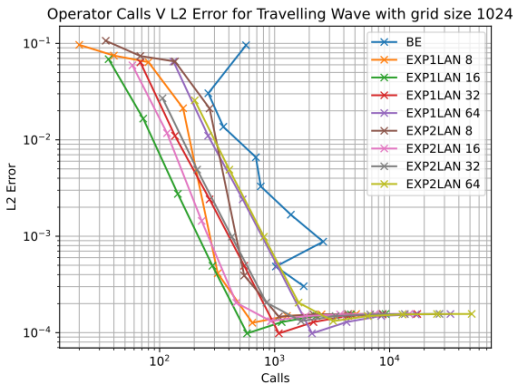

Figure 7: Operator calls vs L_2 with grid size 1024

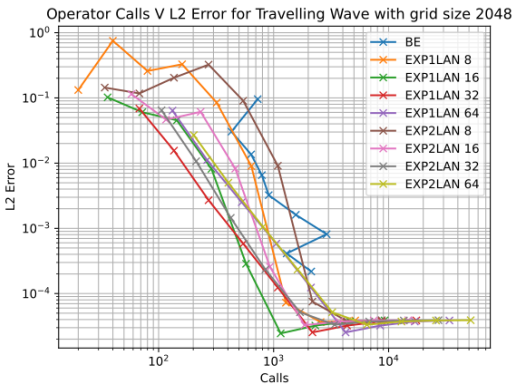

Figure 8: Operator calls vs L_2 with grid size 2048

As expected, when a larger Krylov subspace is used, more calls to the operator are required. This can be seen by comapring the first order exponential integrator with Krylov sizes 16 and 32, where both are identical but the latter has been shifted to the right, using double the number of operator calls. The backward Euler method appears somewhat erratic. This is due to the use of an internal Newton solver which does not have a fixed requirement for number of calls to the operator, instead, running until a given tolerance is reached. We also observe that the exponential methods outperform the BE by achieving a lower error with fewer calls to the operator. We also see that before the discretization error is reached, a Krylov size of 16 gives the lowest error for a given number of operator calls.

Here we present the experimental orders of convergence for the grid size 1024.

τ	BE	EXP1LAN 64	EXP2LAN 64
4	1.6562	2.5200	2.4275
2	1.1560	2.1472	2.2449
1	1.0544	2.1084	2.1753
0.5	1.0189	2.2355	2.1762
0.25	1.0017	2.2848	0.6355
0.125	0.9875	-0.3464	-0.1611
0.0625	0.9808	-0.2097	-0.0546
0.03125	0.9193	-0.0516	-0.0141

Table 1: Experimental orders of convergence

We observe that the rate of convergence for the second order method was in line with what has been shown analytically[11], converging at a rate of 2. The first order method converges with a rate of 2 despite an expectation of a rate of 1[11]. The exponential methods stop converging as they reach the discretization error. The backwards Euler converges at the rate expected of 1.

5.2 A 2D Allen Cahn Problem

The above problem, while done on a two dimensional domain, is really a one dimensional problem. We wish to see how these results carry over to a truly two dimensional problem. We will now begin looking into a two dimensional Allen Cahn Problem given by:

$$u_t = -\Delta u + u(u^2 - 1)\ \Omega = [0, 1] \times [0, 1], 0 \leq t \leq T$$

with Neumann boundary conditions. The initial condition is chosen by randomly setting points on the grid from a uniform distribution in the range $-0.9, 0.9$. The same randomly generated initial condition will be used for each test. As we do not have an exact solution, we will use the backwards Euler method with a timestep of $\tau = 10^{-4}$ to genarate a reference solution. We will compare the backwards Euler method to the first and second order exponential integrator methods described above. The tests are run on a 60×60 grid with the end time being $T = 24$. The error is computed by calculating the L_2 error between the prediction of the chosen stepper and the reference solution. Below we present our results.

5.2.1 Experimental Results

We begin by comparing the timestep size τ to the L_2 error for the different methods.

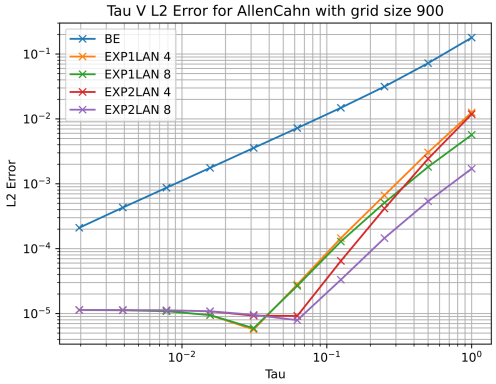

Figure 9: τ vs L_2 with grid size 900

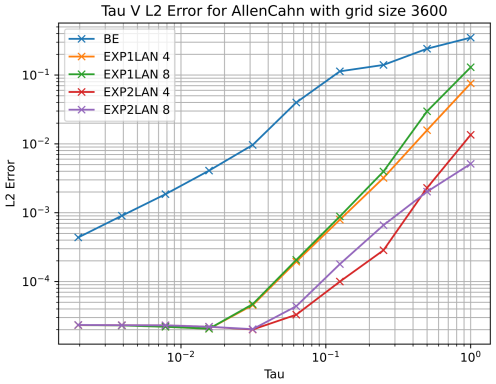

Figure 10: τ vs L_2 with grid size 3600

We observe convergence in error for both expoential methods for both Krylov sizes. These methods significantly outperform the backwards Euler method, similar to the travelling wave Allen Cahn problem given above. They reach the discretization error at around $\tau = 3 \times 10^{-2}$ significantly ahead of the backwards Euler method, which will need a timestep size more than ten times smaller to achieve the same error. For this problem, a relatively shallow subspace has been used compared to above and appears to be sufficient.

Again, we present the number of calls to the operator against the error:

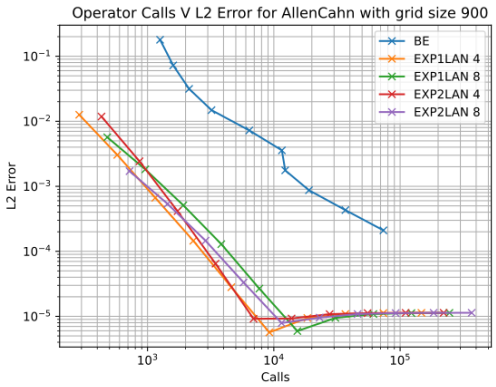

Figure 11: Operator calls vs L_2 with grid size 900

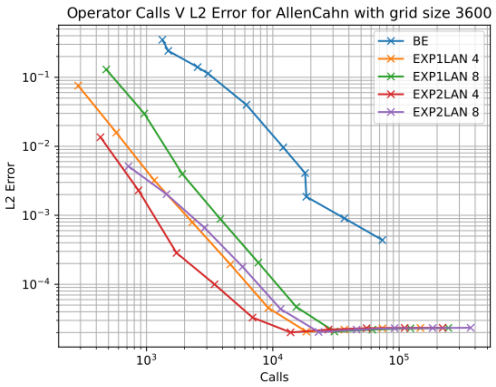

Figure 12: Operator calls vs L_2 with grid size 3600

For all methods, we see that to achieve a lower error requires more calls to the operator. We see that in terms of performance the backwards Euler method performs significantly worse in comparison to both of the exponential methods. The exponential methods reach convergence at around 10^4 operator calls.

Here we present the experimental orders of convergence on a grid size of 3600.

τ	BE	EXP1LAN 8	EXP2LAN 8
0.5	0.5337	2.1244	1.3417
0.25	0.7836	2.9057	1.6172
0.125	0.3088	2.1667	1.8766
0.0625	1.5046	2.0995	2.0343
0.03125	2.0486	2.1358	1.1062
0.015625	1.2371	1.1844	-0.1166
0.0078125	1.1377	-0.0937	-0.0679
0.00390625	1.0396	-0.0652	-0.0057
0.001953125	1.0482	-0.0180	-0.0144

Table 2: Experimental orders of convergence

Again, we observe first order convergence for the backwards Euler method with a convergence rate tending towards somewhere between 1 and 1.3. The exponential methods have a convergence rate in the range of 1.5 to 2.2 showing second order convergence.

5.3 Reaction Diffusion

For both of the above problems, the first order exponential method converged with rate 2. However, we expect the first order exponential method to converge at rate 1. We wish to measure the performance of this method in comparison to backwards Euler for a problem where it does not converge at rate 2 but at rate 1 as expected. In order to do this, we study the following reaction diffusion problem[11].

$$\begin{aligned} u_t &= -\frac{1}{2}\Delta u + \frac{1}{2}\pi^2 u + \frac{1}{2}\pi^2 e^{-\pi^2 t} sin(\pi x) sin(\pi y) \ (x,y) \in \Omega t \in \times [0, t_e] \\ u(0,x,y) &= (sin(\pi x) - 1) sin(\pi y) \ (x,y) \in \Omega \end{aligned}$$

With the following exact solution:

$$u(t,x,y) = e^{-\pi^2 t}(sin(\pi x) - 1) sin(\pi y)$$

with homogeneous Dirichlet boundary conditions and where $\Omega = (\frac{1}{2}, \frac{5}{2}) \times (0,1)$ and our end time is given by $t_e = 0.1$. We run our tests on grid sizes of 128×64 and 256×128 and use times steps sizes of $\tau = \frac{0.1}{2^i}$ for $i = 0, 1, ..., 4$.

We begin by comparing the timestep size τ to the L_2 error for the different methods.

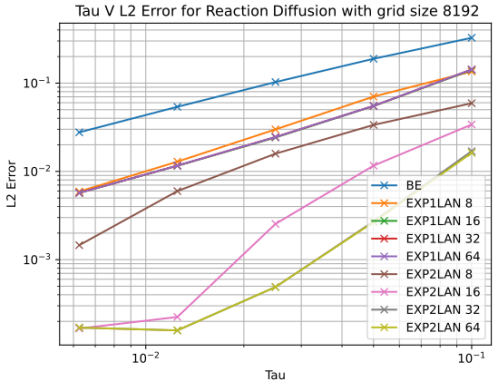

Figure 13: τ vs L_2 with grid size 8192

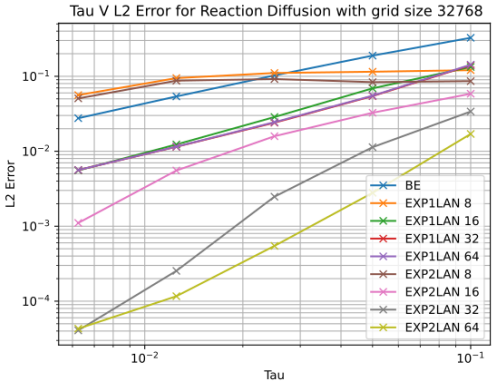

Figure 14: τ vs L_2 with grid size 32768

All methods appear to converge to the spacial discretization error. As above, we observe that the Krylov subspace must be sufficiently deep in order to be accurate. In this example, we can see that the second order method converges at a faster rate than the first order method as the slope for the second order method is steeper. We also see that the backwards Euler method has a higher error for a given value of τ the both exponential methods, normally around twice that of the first order expoential integrator method.

Again, we present the number of calls to the operator against the error.

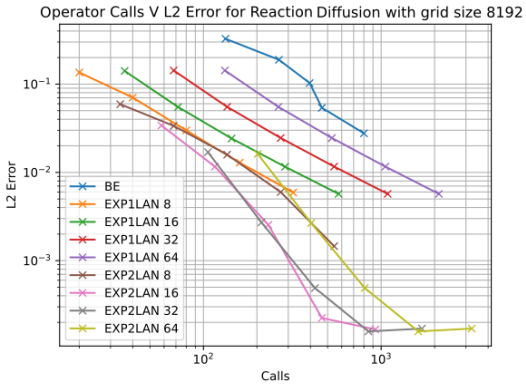

Figure 15: Operator calls vs L_2 with grid size 8192

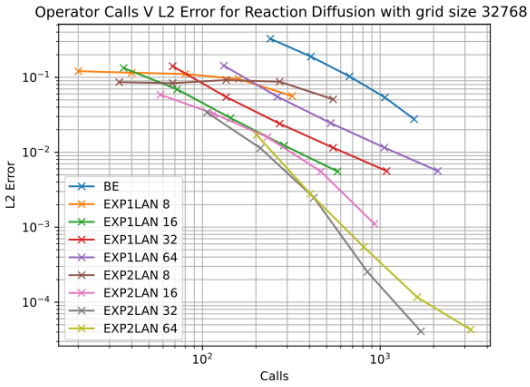

Figure 16: Operator calls vs L_2 with grid size 32768

Here we see that for all methods, more operator calls are required to produce results with a smaller error. The exponential methods achieved a given error with fewer operator calls than the backwards Euler method. The second order methods required fewer operator calls for a given error than the first order method, especially as the number of calls increased.

Here we present the experimental orders of convergence.

τ	BE	EXP1LAN 64	EXP2LAN 64
0.5	0.7902	1.3717	2.6086
0.25	0.8767	1.1735	2.3519
0.125	0.9320	1.0828	2.2299
0.0625	0.9635	1.0378	1.4358

Table 3: Experimental orders of convergence

Now we observe the expected results, with the first order exponential method converging at rate 1 and the second order method converging at rate 2.

6 Snowflakes

Here we explore the phase boundary of the formation of a snowflake. We will use an adaptive grid and compare the first and second order exponential integrator methods. We will also compare the limitations of these methods for differnt Krylov sizes.

6.1 Formulation of problem

We consider the problem on a domain $\Omega = \times[4,8] \times [4,8]$. The equations governing this phase boundary problem are as follows [12].

$$\begin{aligned}\tau\frac{\partial\phi}{\partial t} &= \nabla\cdot D\nabla\phi + \phi(1-\phi)m(\phi,T)\\ \frac{\partial T}{\partial t} &= D_T\nabla^2 T + \frac{\partial\phi}{\partial t}\end{aligned}$$

where

$$\begin{aligned}m(\phi,T) &= \phi - \frac{1}{2} - \frac{\kappa_1}{\pi}arctan(\kappa_2 T)\\ D &= \alpha^2(1+c\beta)\begin{pmatrix}1+c\beta & -c\frac{\partial\beta}{\partial\psi}\\ c\frac{\partial\beta}{\partial\psi} & 1+c\beta\end{pmatrix}\\ \text{With } \beta &= \frac{1-\Phi^2}{1+\Phi^2}\\ \Phi &= tan(\frac{N}{2}\psi)\\ \psi &= \theta + arctan(\frac{\frac{\partial\phi}{\partial y}}{\frac{\partial\phi}{\partial x}})\end{aligned}$$

and with the following constants

$$\begin{array}{cccc} D_T = 2.25 & \alpha = 0.015 & \tau = 3\cdot 10^{-4} & \kappa_1 = 0.9\\ \kappa_2 = 20 & c = 0.02 & N = 6 & \theta = \frac{\pi}{8}\end{array}$$

Substituting $\frac{\partial \phi}{\partial t}$ into the second equation and rewriting in weak form gives the following:

$$\int \frac{\partial \phi}{\partial t} v_\phi dx = \frac{1}{\tau} \int \phi(1-\phi) m(\phi, T) v_\phi - D\nabla\phi \cdot \nabla v_\phi dx$$
$$\int \frac{\partial T}{\partial t} v_T dx = \int \frac{1}{\tau} (\phi(1-\phi) m(\phi, T) v_T - \nabla v_T \cdot D\nabla\phi) - D_T \nabla v_T dx \cdot \nabla T$$

Where we use homogeneous Neumann boundary conditions.

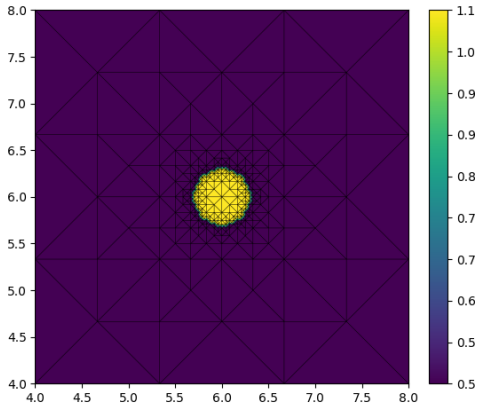

Figure 17: Initial condition

With an initial condition (figure 17) of:

$$\phi(0, x, y) = \begin{cases} 1 & (x-6)^2 + (y-6)^2 \leq 0.3 \\ 0 & \text{otherwise} \end{cases}$$
$$T(0, x, y) = -0.5 \text{ for } (0, x, y) \in \Omega$$

6.2 Numerical Solutions

Here we begin the comparison of the different methods. For all of them, we will use a time step size of 0.0005 and will run until an end time of 0.1.

We begin looking at the results for the first and second order methods with a Krylov subspace of dimension $16, 32$.

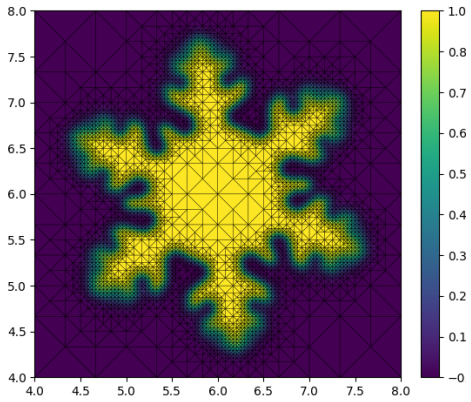

Figure 18: from first order exponential integrator with Krylov size 16

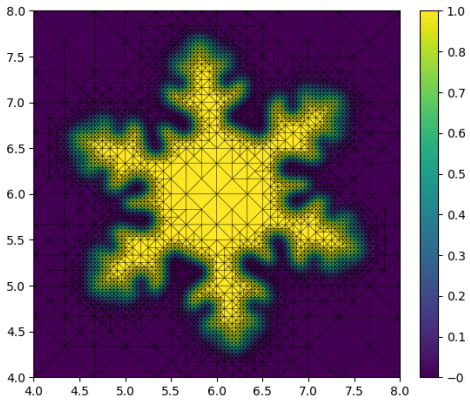

Figure 19: from first order exponential integrator with Krylov size 32

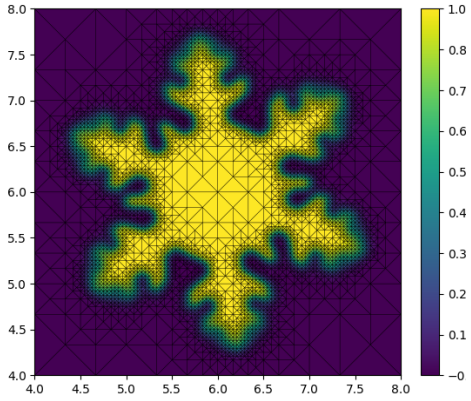

Figure 20: from second order exponential integrator with Krylov size 16

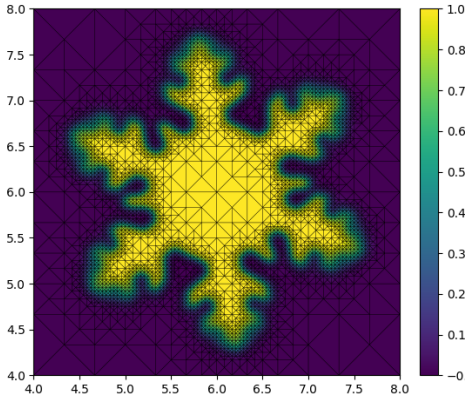

Figure 21: from second order exponential integrator with Krylov size 32

We observe that the snowflakes generated by both Krylov sizes are near identical, suggesting that setting the dimension of the Krylov space to 16 is sufficient for this problem.

6.3 Limitations

Now we compare the limitations of the first and second order exponential methods when the dimension of the Krylov space is smaller that for those used above. We compare it for Krylov sizes of 8 and 10.

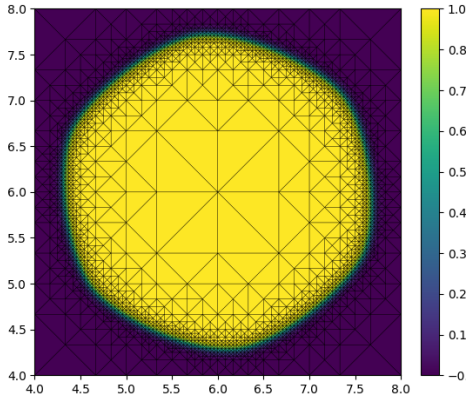

Figure 22: from first order exponential integrator with Krylov size 8 at time $t = 0.05$

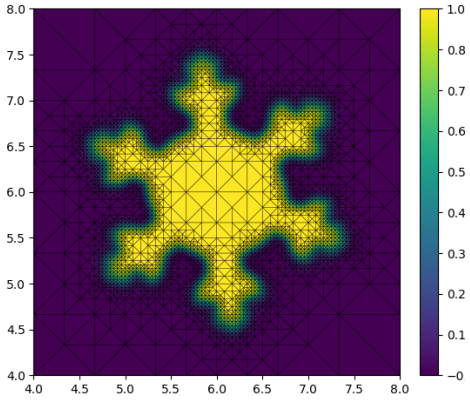

Figure 23: from first order exponential integrator with Krylov size 10

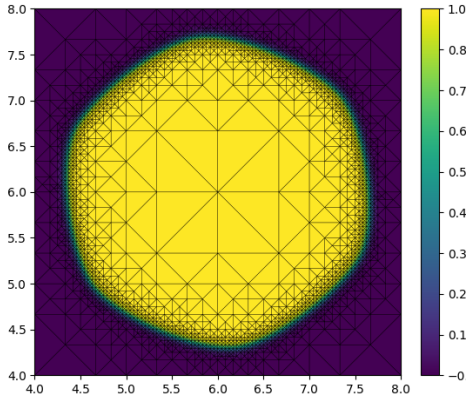

Figure 24: from second order exponential integrator with Krylov size 8 at time $t = 0.05$

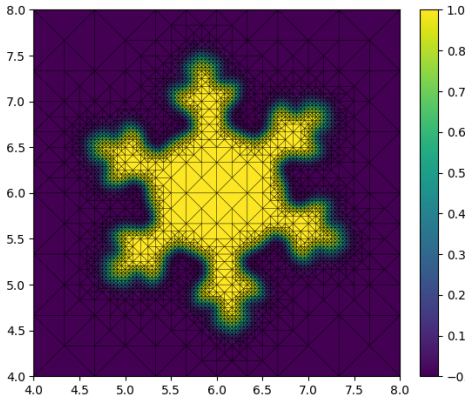

Figure 25: from second order exponential integrator with Krylov size 10

When a subspace of size 8 is used, as shown in figure 22, we observe that the snowflake continues to grow in the shape of a smooth hexagon without forming any dendrites. This hexagon continued to grow until it pressed up against the boundaries of the domain. When we use a subspace of size 10 we observe that, while dendrites now form, they are stubbier and produce less branching.

We now repeat the above but halving the step size and keeping the Krylov size the same, at 8 and 10.

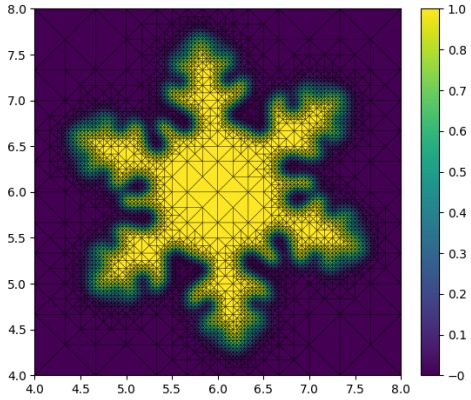

Figure 26: from first order exponential integrator with Krylov size 8

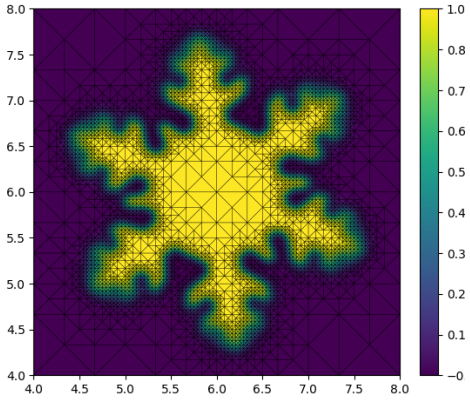

Figure 27: from first order exponential integrator with Krylov size 10

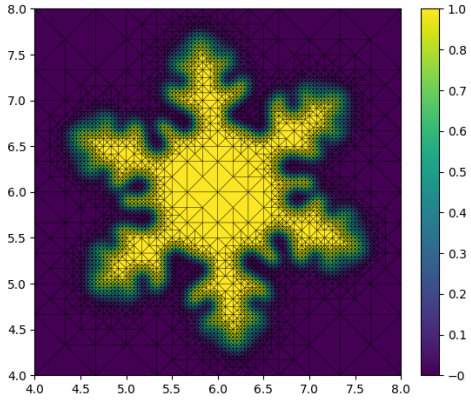

Figure 28: from second order exponential integrator with Krylov size 8

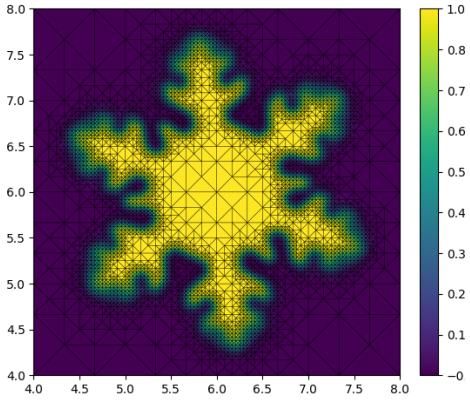

Figure 29: from second order exponential integrator with Krylov size 10

Even with a smaller Krylov subspace, we can generate plots similar to those that use a larger subspace so long as the timestep is small enough. This matches with what we previously observed with the above Allen Cahn eqautions where for sufficiently small τ different Krylov subspaces would produce the same error.

7 3D Crystal

We now begin looking into the modeling of a 3D Crystal.

7.1 Formulation of Problem

We employ the phase field model as described by [4].

$$\dot{\phi} = \nabla \cdot \frac{\partial}{\partial \nabla \phi}(\frac{1}{2}A^2) - \frac{Omega'(\phi)}{\delta^2} - \frac{g'(\phi)(\mu_0 - \mu)\Delta c}{\lambda \delta^2}$$

$$\dot{\mu} = a\nabla \cdot D\nabla\mu - a\Delta c g'(\phi)\dot{\phi}$$
$$\phi(0,x) = \phi_{t=0}$$
$$\mu(0,x) = \mu_{t=0}$$

For some initial conditions $\phi_{t=0}$ and $\mu_{t=0}$ and with an anisotropy given by A.

$$\phi_{t=0} = \frac{1}{1 + e^{\frac{-\sqrt{x^2+y^2+z^2}-R_0}{\delta}}}$$
$$\mu_{t=0} = \phi_{t=0} - \phi_{t=0} a(\bar{c} - c_s)$$

The constants are given by:

$$c_L = 0.9 \quad c_S = 0.5 \quad \mu_0 = 1 \quad \mu_{\text{inf}} = 0.04 \quad a = 4 \quad D_L = \tfrac{1}{12} \quad D_S = 10^{-4} D_L$$
$$\Delta c = c_L - c_S \quad \lambda = \tfrac{3R_c \Delta c^2}{\delta} \quad R_c = 10 \quad R_0 = 20 \quad \delta = 2 \quad \epsilon = 0.02$$

We will observe two different anisotropies. The first a "cubic" anisotropy given by:

$$A = \sum_i \sqrt{\frac{\partial\phi}{\partial x^i}^2 + \epsilon^2 (\frac{\partial\phi}{\partial x^1}^2 + \frac{\partial\phi}{\partial x^2}^2 + \frac{\partial\phi}{\partial x^3}^2)}$$

and an octohedral anisotropy given by [5]:

$$A = A_0(1 + \bar{\epsilon}(n_x^4 + n_y^4 + n_z^4))$$

where we have that $A_0 = 1 - 3\epsilon$ and $\bar{\epsilon} = \frac{4\epsilon}{1-3\epsilon}$.

7.2 Numerical Solutions

Here we present the numerical solution with the cubic anisotropy at time $t = 400$

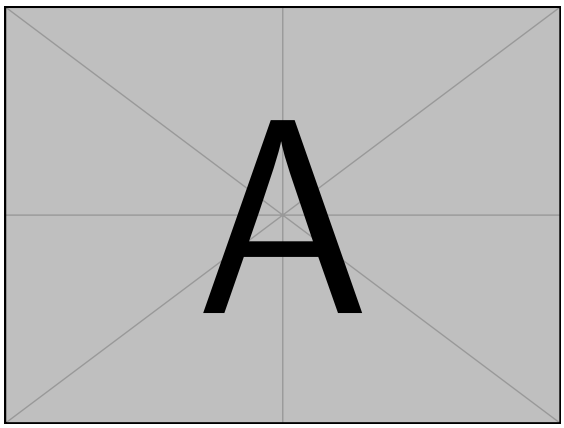

Figure 30: from first order exponential integrator with Krylov size 16

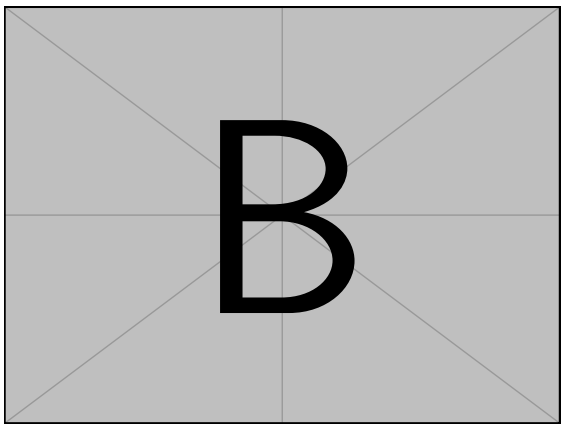

Figure 31: from first order exponential integrator with Krylov size 16

We observe that the cyrstal seed has now formed a cubic shape.

Here we present the numerical solution with the octohedral anisotropy at time $t =?$

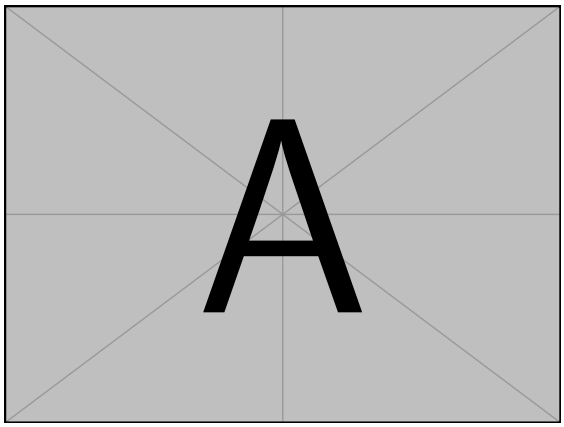

Figure 32: from first order exponential integrator with Krylov size 16

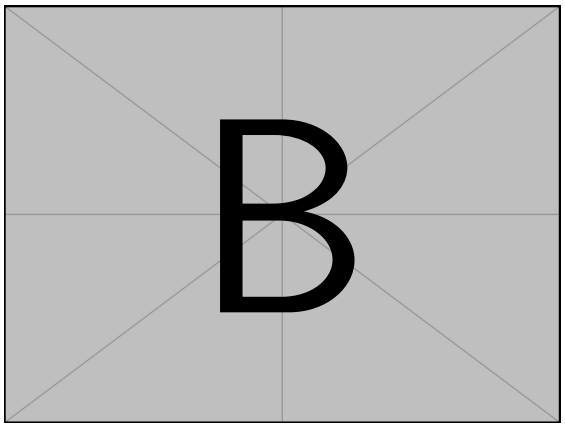

Figure 33: from first order exponential integrator with Krylov size 16

We observe that the cyrstal seed has now formed into the expected shape.

8 Double Slit Wave Model

In this section, we look into applying these exponential methods to the computation of solutions to the wave equation. Specifically, we will look into solutions to the double slit wave problem:

8.1 Formulation of Problem

Where our domain is given by Ω as seen in figure 34, we write the problem as:

$$\psi_{tt} = \nabla\psi \quad x \in \Omega, 0 < t \leq T\psi(0, x) = 0 \quad x \in \Omega$$

where $T = 3$ is our end time for the simulation. Let $S_L = \{0\} \times [0, 1] \subset \mathbb{R}^2$ denote the left side of the domain Ω. We implement the following Dirichlet boundary condition:

$$\psi = \frac{1}{10\pi} sin(10\pi t) x \in S_L$$

this will be the source of our waves. The remaining boundaries use homogeneous Neumann boundary conditions. Our methods, both exponential and backwards Euler, in section 2 require a first order equation with respect to the derivatives of t. This means that we will need to use a substitution. We can use $\psi_t = -p$ to get the following:

$$\psi_t = -p$$

$$p_t = \nabla\psi$$

with

$$p(0, x, y) = 0$$

This equation is now compatible with the methods that we are using.

8.2 Numerical Solutions

We can now compare backwards Euler and the two exponential integrator methods. First, we show that the domain as well as a solution computed with the backwards Euler method. For the backwards Euler, a timestep of $\tau = 0.001$ was found to be optimal, with larger timesteps causing numerical issues, leading to instability and smaller timesteps requiring excessive computation.

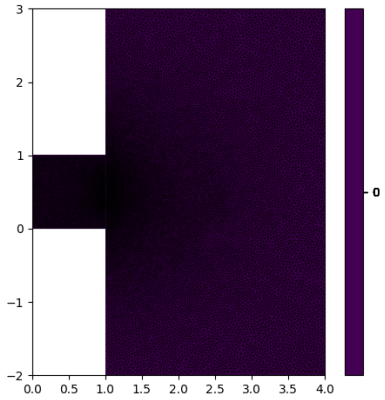

Figure 34: Domain Ω with grid displayed

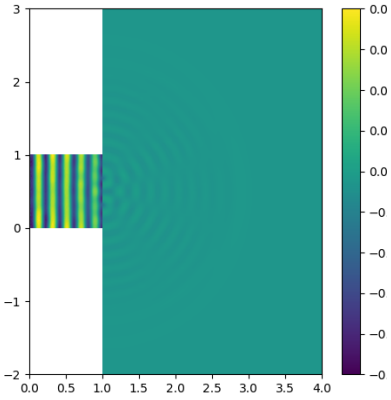

Figure 35: Solution at time $T = 3$ computed using the backwards Euler method

We now compare the backwards Euler to the first and second order exponential integrator methods. Note that due to the fact that the linear term given by $DN(u)$ is not symmetric we cannot use the Lanczos method for computing our matrix exponential. As a result, the Arnoldi algorithm for the matrix exponential has been used here exclusively.

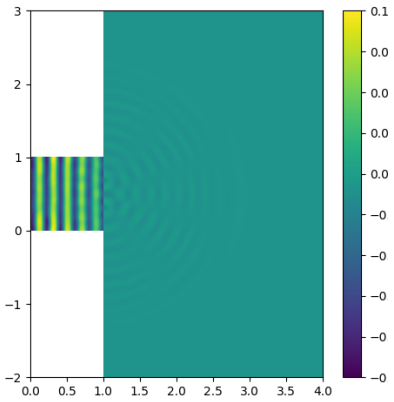

Figure 36: Solution at time $T = 3$ computed using the first order exponential integrator with timestep $\tau = 0.02$ and a Krylov size of 24

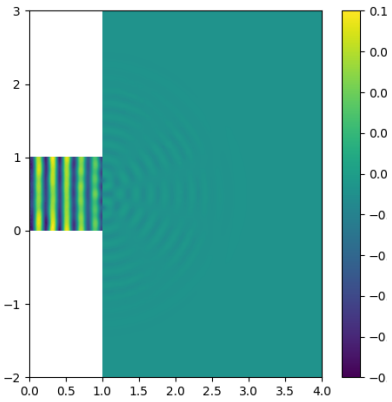

Figure 37: Solution at time $T = 3$ computed using the second order exponential integrator with timestep $\tau = 0.02$ and a Krylov size of 24

We see that the first and second order exponential methods using the Arnoldi algorithm are capable of producing results equivalent to that of the backwards Euler method.

8.3 Performance

We now observe the performance differences between backwards Euler and the exponential integrators. The number of calls to the operator is again used in order to measure performance. We present the methods used and the operator calls in the table below:

Method	Required τ	Krylov size	Operator Calls
Backwards Euler	0.001	-	119362
First Order Exponential	0.02	24	7800
Second Order Exponential	0.02	24	12300

Table 4: Performance of Various Methods

We observe that the backwards Euler method requires significantly more calls to the operator than both the first and second order exponential methods. The second order method requires more calls to the operator than the first order method. Both exponential methods required a similar timestep size τ. This suggests that the the exponential methods may be effective for computing solutions to the wave equation. This is significant as it shows the possible application of these methods to problems beyond parabolic PDEs.

9 Rising Bubble

Here we will look into computing a rising bubble, in order to investigate how well these solvers work for problems in fluid dynamics.

9.1 Formulation of Problem

The problem is given by the following equations[7]:

$$\begin{aligned}
\frac{\partial u_i}{\partial t} &= -\frac{\partial u_i u_j}{\partial x_j} + u_i \frac{\partial u_j}{\partial x_j} - c_p \theta \frac{\partial \pi}{\partial x_i} \\
\frac{\partial \pi}{\partial t} &= -\frac{\partial u_j \pi}{\partial x_j} + \pi \frac{\partial u_j}{\partial x_j} - \pi \frac{R}{c_v} \frac{\partial u_j}{\partial x_j} \\
\frac{\partial \theta}{\partial t} &= -\frac{\partial u_j \theta}{\partial x_j} + \theta \frac{\partial u_j}{\partial x_j}
\end{aligned}$$

Where Einstein summing notation is used. We use a domain $\Omega = (0, 1000) \times (0, 2000)$. The constatns are given by:

$$\begin{aligned}
c_p = 1005 \quad c_v = 717.95 \\
R = c_p - c_v
\end{aligned}$$

We enforce reflective boundary conditions along the sides of the domain. The initial condtions are as follows:

$$\begin{aligned}
u_i = 0 \\
\pi(0, x, y) = \pi_0 \\
\theta = 0
\end{aligned}$$

where π_0 is shown below

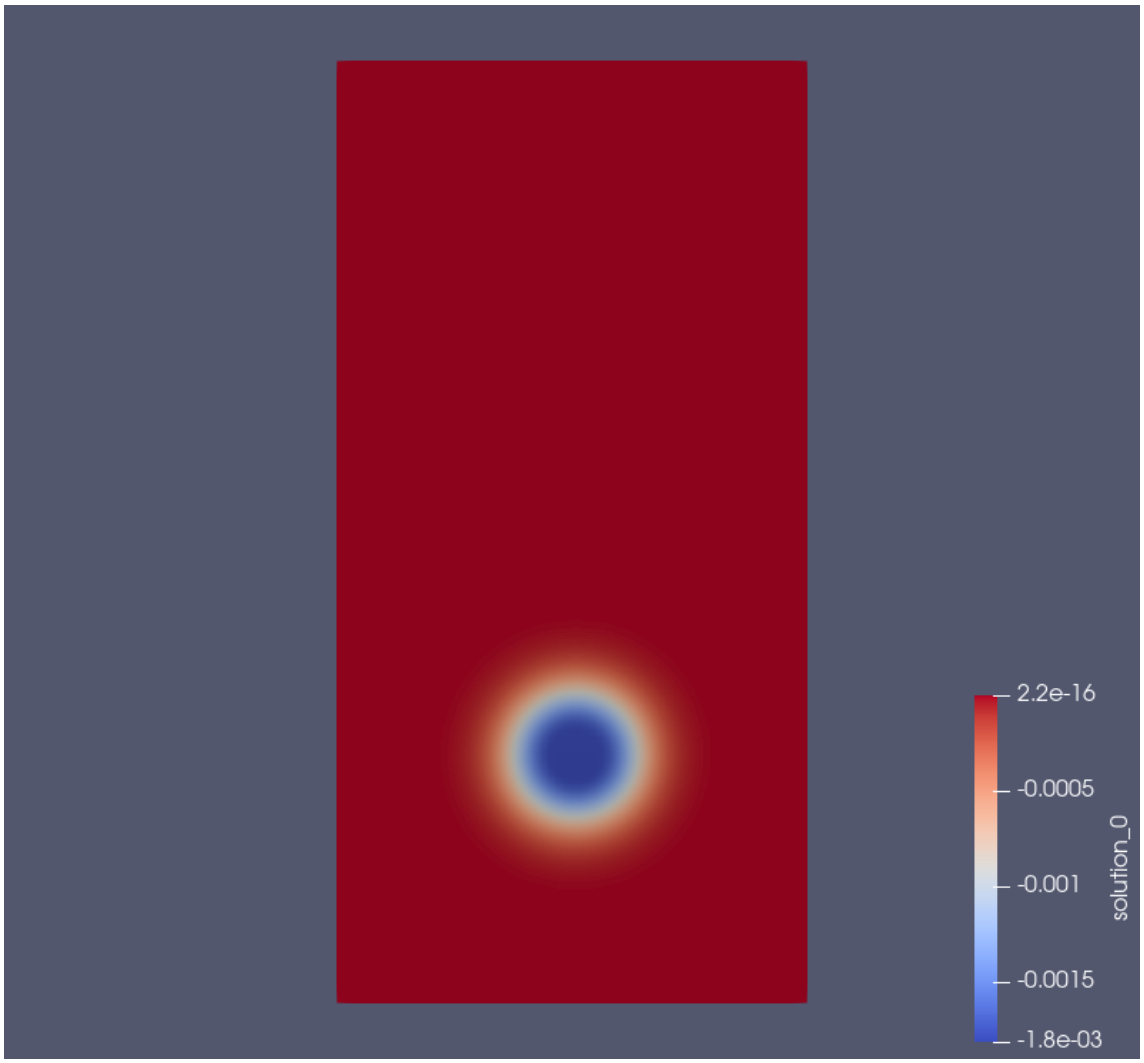

Figure 38: Initial Condition of π_0

9.2 Numerical Solutions

We run our simulations on a finite volume mesh. The grid is rectangular and of dimension 64×128. We use a timestep of $\tau = 1.8$ as this was optimal when used with the backwards Euler method. We plot the value of π at time $t = 4000$.

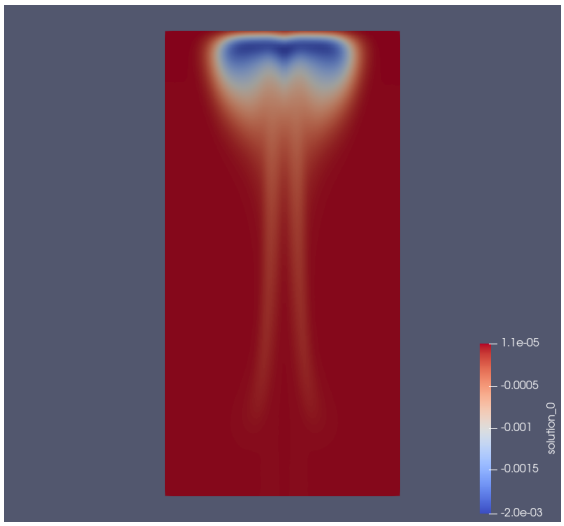

Figure 39: from first order exponential integrator with Krylov size 32

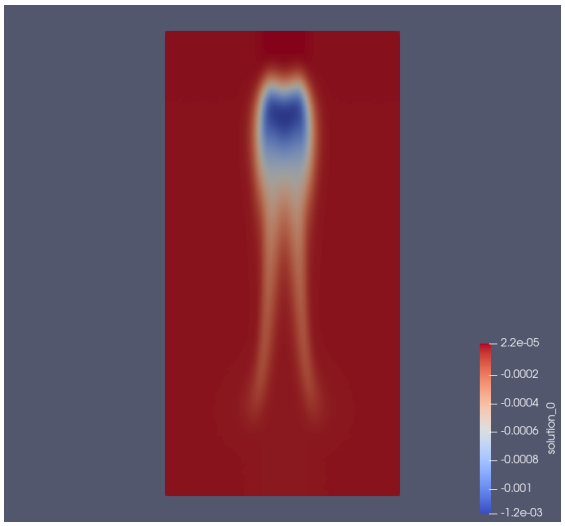

Figure 40: from first order exponential integrator with Krylov size 64

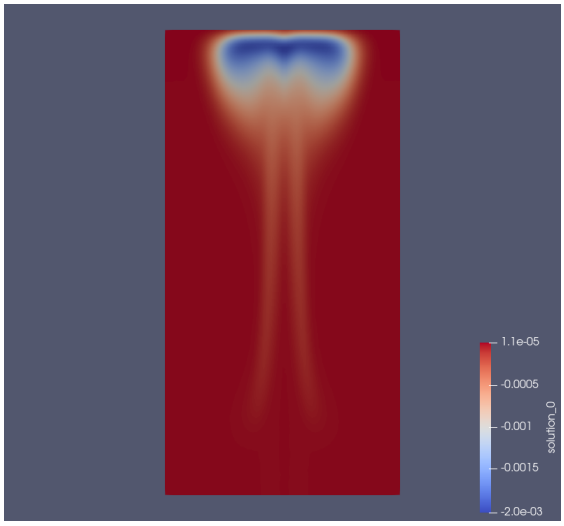

Figure 41: from second order exponential integrator with Krylov size 32

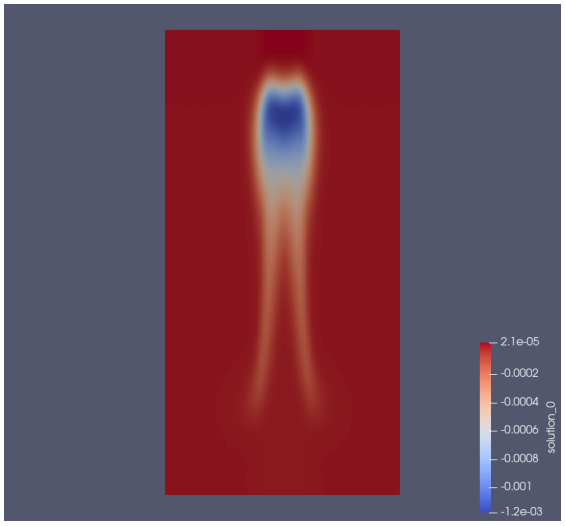

Figure 42: from second order exponential integrator with Krylov size 64

For Krylov sizes above 64 the computed solution didn't change. The solution computed with a Krylov size of 64 was the same as the solution by the backwards Euler method as well as for smaller τ. We therefore conclude that this is close to the true solution.

9.3 Performance

We now observe the performance differences between backwards Euler and the exponential integrators. The number of calls to the operator is again used in order to measure performance. We present the methods used and the operator calls in the table below:

Method	Krylov size	Operator Calls
Backwards Euler	-	156879
First Order Exponential	32	151640
First Order Exponential	64	303280
Second Order Exponential	32	227460
Second Order Exponential	64	454920

Table 5: Performance of Various Methods

We see that these exponential integrator methods required more operator calls that the backwards Euler method to achieve comparable results for this timestep size.

Further studies of these methods could include finding the optimal tau for these methods that minimises the number of operator calls. It would also make sense to explore the possible values of m in between 32 and 64 as we would expect to find that these methods converge to the expceted solution for a Krylov size between these two values. However, due to time constraints on the project it has not been possible to gather this information.

10 Conclusion

In summary, this has been a very successful project.

Bibliography

References

[1] Awad H. Al-Mohy and Nicholas J. Higham. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM Journal on Scientific Computing*, 33(2):488–511, January 2011.

[2] Martin S. Alnæs, Anders Logg, Kristian B. Ølgaard, Marie E. Rognes, and Garth N. Wells. Unified form language: A domain-specific language for weak formulations of partial differential equations. *ACM Transactions on Mathematical Software*, 40(2):1–37, February 2014.

[3] Peter Bastian, Markus Blatt, Andreas Dedner, Nils-Arne Dreier, Christian Engwer, René Fritze, Carsten Gräser, Christoph Grüninger, Dominic Kempf, Robert Klöfkorn, Mario Ohlberger, and Oliver Sander. The dune framework: Basic concepts and recent developments. *Computers & Mathematics with Applications*, 81:75–112, January 2021.

[4] P. C. Bollada, P. K. Jimack, and A. M. Mullis. Phase field modelling of hopper crystal growth in alloys. *Scientific Reports*, 13(1), August 2023.

[5] P.C. Bollada, C.E. Goodyer, P.K. Jimack, A.M. Mullis, and F.W. Yang. Three dimensional thermal-solute phase field simulation of binary alloy solidification. *Journal of Computational Physics*, 287:130–150, April 2015.

[6] Susanne C. Brenner and L. Ridgway Scott. *The Mathematical Theory of Finite Element Methods*. Springer New York, 2008.

[7] George H. Bryan and J. Michael Fritsch. A benchmark simulation for moist nonhydrostatic numerical models. *Monthly Weather Review*, 130(12):2917–2928, December 2002.

[8] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, January 2008.

[9] Nicholas J. Higham and Awad H. Al-Mohy. Computing matrix functions. *Acta Numerica*, 19:159–208, May 2010.

[10] Marlis Hochbruck. Exponential integrators. *Acta Numerica*, page 209–286, 2010.

[11] Jianguo Huang, Lili Ju, and Yuejin Xu. Efficient exponential integrator finite element method for semilinear parabolic equations. September 2022.

[12] Ryo Kobayashi. Modeling and numerical simulations of dendritic crystal growth. *Physica D: Nonlinear Phenomena*, 63(3–4):410–423, March 1993.

[13] Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, January 2003.

[14] I. U. OJALVO and M. NEWMAN. Vibration modes of large structures by an automatic matrix-reductionmethod. *AIAA Journal*, 8(7):1234–1239, July 1970.

[15] Y. Saad. Analysis of some krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 29(1):209–228, February 1992.

[16] Yoshihisa Morita Yukitaka Fukao and Hirokazu Ninomiya. Some entire solutions of the allen–cahn equation. *TAIWANESE JOURNAL OF MATHEMATICS*, 8(1):15–32, March 2004.