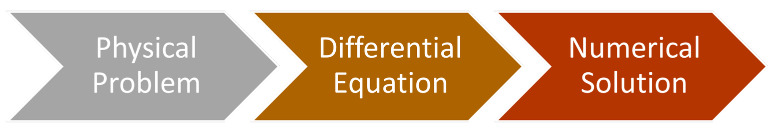
High-Order Operator Splitting Schemes for Stiff Differential Equations

Chloe Griffin

Converse University

Introduction

Scientists use differential equations to model many physical, biological, and chemical phenomena. A differential equation relates a quantity of interest's dynamics to a function of its derivatives. We approximate solutions to these equations.



Example: Exponential Population Growth

- Relates change in population to population times a growth rate: $\frac{dP}{dt} = aP$
- Replace derivatives with simple differences: $\frac{dP}{dt} \approx \frac{P(t^{n+1})-P(t^n)}{t^{n+1}-t^n}$

Create Time-Stepping Method:

- Explicit Forward Euler: $P(t^{n+1}) = P(t^n) + \Delta t a P(t^n)$
- Implicit Backward Euler: $P(t^{n+1}) = P(t^n) + \Delta t a P(t^{n+1})$

Some differential equations are stiff, which means they have a more restrictive stability requirement than accuracy requirement on their time step.

Project Description

We aim to use the largest possible time step and reduce computational effort. Therefore, we develop high-order operator splitting schemes to yield efficient, inexpensive approximations for stiff differential equations.

Methods

Stiff Test Problem: Diffusion

- Particle flux is proportional to the negative gradient [1]
- Relates the first derivative in time and sum of second derivatives in space

The 1D and 2D equations are given in Equation 1, where q is a source term.

$$u_t = u_{xx} + q(x,t) \mid u_t = u_{xx} + u_{yy} + q(x,y,t) \tag{1}$$

We use second order central finite difference stencils to approximate spacial derivatives, as shown in Equation 2.

$$u_{xx} \approx F(u^n_{i,j}) = \frac{u^n_{i+1,j} - 2u^n_{i,j} + u^n_{i-1,j}}{\Delta x^2} \;\Big|\; u_{yy} \approx G(u^n_{i,j}) = \frac{u^n_{i,j+1} - 2u^n_{i,j} + u^n_{i,j-1}}{\Delta y^2} \tag{2}$$

In general, accuracy and stability conditions dictate step size requirements.

Accuracy

We compare errors at our final time to determine the accuracy of our scheme. Error is the difference between the approximation and the actual solution:

$$E(t^n) = |u^n - u(t^n)|, \tag{3}$$

where u^n is the approximate value and $u(t^n)$ is the exact value. As we decrease the size of our time step, our approximation approaches the actual solution. The order of a scheme, p, measures the convergence rate at which the error decays with respect to the step size,

$$Error \approx C\Delta t^p \tag{4}$$

where Δt is the size of the largest step-size taken throughout the simulation.

Higher Order Methods

- Increase computational effort per time-step for more accurate simulations
- Justify added effort per step by reducing the number of steps

Stability Conditions

- Prevent the error from magnifying uncontrollably.
- For Explicit Euler: $O(\Delta x^2)$ in 1D and $O(\Delta x^2 + \Delta y^2)$ in 2D
- For Implicit Euler: Δt is unconditionally stable

We use implicit schemes for stiff equations, but these require expensive implicit solvers. Figures 1 shows the benefit of using an implicit scheme.

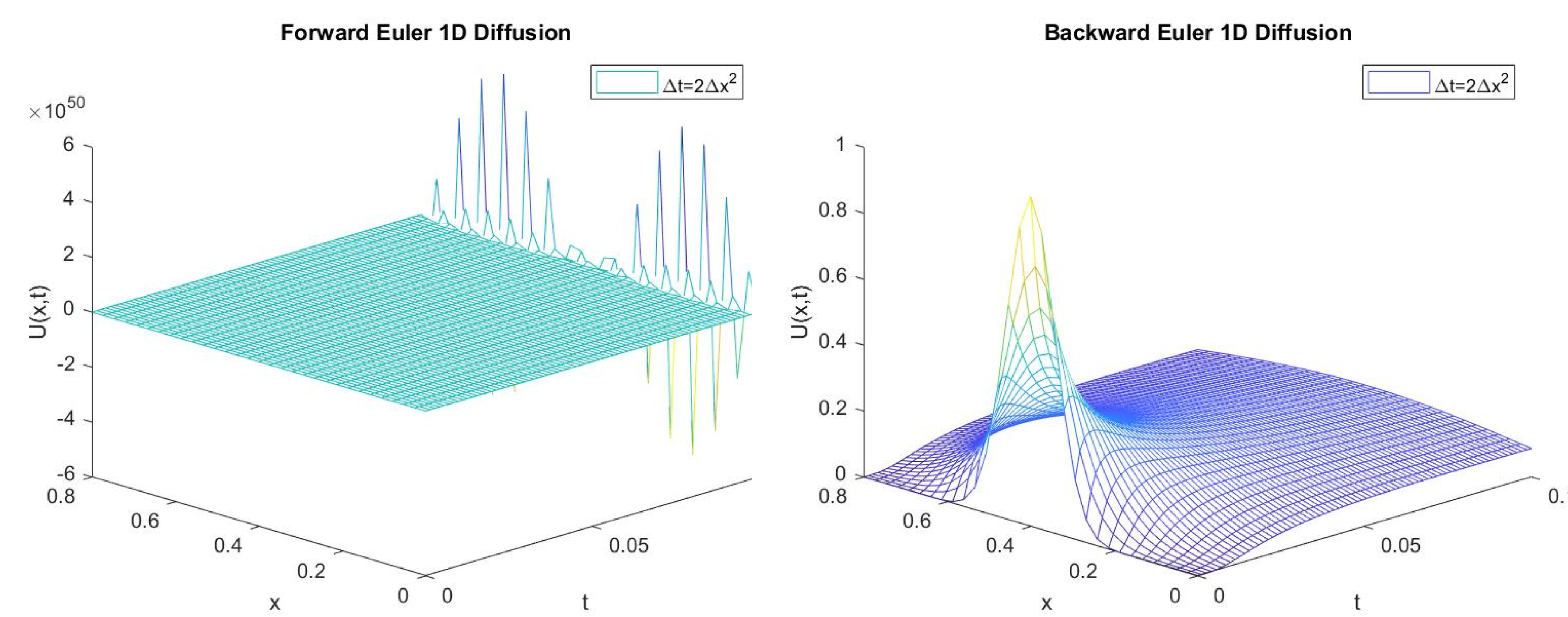


Figure 1. Plots of unstable explicit scheme and stable implicit scheme with $\Delta t = 2\Delta x^2$

For large problems, the computational effort needed for implicit, high-order schemes becomes prohibitive. To alleviate this cost, we apply splitting techniques to separate the problem into several components.

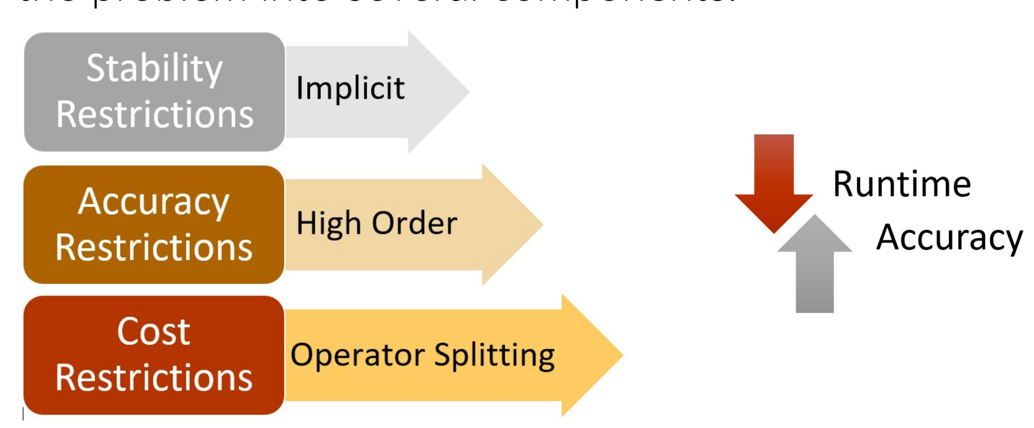


Figure 2. Method approach for stiff test problem

Results

BDF-2 Operator Splitting

We split the summation operator between the spatial derivatives and apply this technique to the second order backward differentiation formula (BDF-2)

In two dimensions, BDF-2 is given by,

$$u^{n+1} = \frac{-1}{3}u^{n-1} + \frac{4}{3}u^n + \frac{2\Delta t}{3}[F(u^{n+1}) + G(u^{n+1})] \tag{5}$$

We split the problem in an alternating style:

$$u_0^{n+1} = \frac{-1}{3}u^{n-1} + \frac{4}{3}u^n + \frac{2\Delta t}{3}F(u_0^{n+1}) \tag{6}$$

$$u_1^{n+1} = \frac{-1}{3}u^{n-1} + \frac{4}{3}u^n + \frac{2\Delta t}{3}[F(u_0^{n+1}) + G(u_1^{n+1})] \tag{7}$$

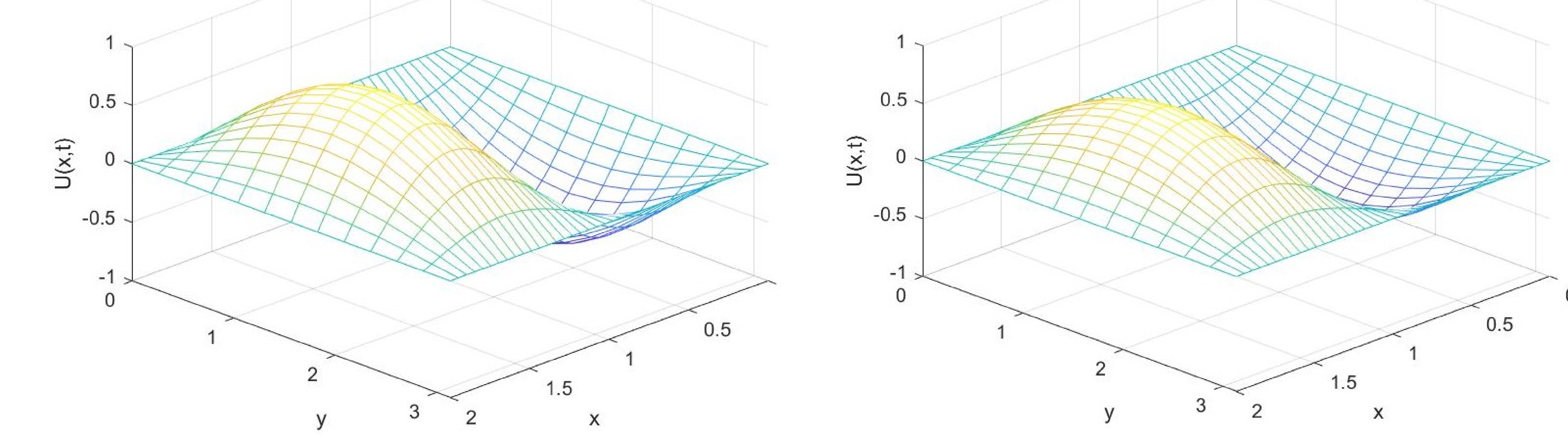
$$u_2^{n+1} = \frac{-1}{3}u^{n-1} + \frac{4}{3}u^n + \frac{2\Delta t}{3}[F(u_2^{n+1}) + G(u_1^{n+1})] \tag{8}$$

We perform 1D slices instead of a full 2D inversion, allowing for simple sequential implicit solves. Splitting the scheme introduces new error. Therefore, we make corrections to recover the high order. By rearranging the scheme, we see the iterative correction procedure,

$$u_{2k}^{n+1} = u_{2k-1}^{n+1} + \frac{2\Delta t}{3}[F(u_{2k}^{n+1}) - F(u_{2k-2}^{n+1})] \text{ for } k \geq 1 \tag{9}$$

$$u_{2k+1}^{n+1} = u_{2k}^{n+1} + \frac{2\Delta t}{3}[G(u_{2k+1}^{n+1}) - G(u_{2k-1}^{n+1})] \text{ for } k \geq 1 \tag{10}$$

We approach the original order of our scheme as we make additional corrections. An output of a two-correction scheme at time $t = 0$ and $t = 2$ can be seen below.



Corrections to the operator splitting method increase the order of our scheme and reduce the associated error, as shown in Table 1.

Order Recovery

Δt	Direct BDF-2	Operator Splitting	One Correction	Two Corrections
$1/20$	3.00E-04	4.20E-03	1.80E-03	2.60E-04
$1/40$	7.29E-05 (2.00)	2.20E-03 (0.92)	4.47E-04 (1.98)	6.75E-05 (1.91)
$1/80$	1.80E-05 (2.00)	1.10E-03 (0.96)	1.11E-04 (1.99)	1.73E-05 (1.95)
$1/160$	4.47E-06 (2.00)	5.59E-04 (0.98)	2.77E-05 (1.99)	4.40E-06 (1.97)

Table 1. Compares error and order with decreasing time steps for BDF-2 scheme without splitting, with operator splitting, with one correction, and with two corrections.

Although the number of implicit solves increases, the cost of each solve reduces, resulting in an efficient scheme. A runtime comparison is shown in Table 2.

Runtime Comparison

		Direct	Splitting	One Correction	Two Corrections
$N = (360)^2$	Time	69.847	4.989	7.698	10.181
	Error	3.40E-06	4.97E-04	2.18E-05	3.33E-06
$N = (720)^2$	Time	656.065	40.141	64.751	79.126
	Error	8.48E-07	2.50E-04	5.43E-06	8.39E-07

Table 2. Compares runtimes in seconds and error of schemes described for BDF-2 scheme without splitting, with operator splitting, with one correction, and with two corrections using 360 and 720 points in x and y.

Conclusions

Operator splitting with implicit, alternating error corrections is a successful strategy for the BDF-2 method. Splitting strategies are known to suffer order reduction when solving stiff problems. As shown in the results section, the correction strategy allows us to recover the proper accuracy of the underlying scheme while reducing computational and memory costs. In future research, we will apply this strategy to other high-order methods and make additional comparisons. This project was mentored by Dr. Zachary Grant at Oak Ridge National Lab.

References

[1] Romeo M. Flores. Coalification, gasification, and gas storage. In Romeo M. Flores, editor, *Coal and Coalbed Gas*, chapter Four, pages 167–233. Elsevier, 2014.

U.S. DEPARTMENT OF ENERGY | Office of Science

OAK RIDGE INSTITUTE FOR SCIENCE AND EDUCATION

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internship program

dcgriffin001@converse.edu