A novel second-order accurate Strang splitting type exponential integrator for stiff systems

Saburi Rasheed
Advisor: Dr. Bruce A. Wade
Department of Mathematics, University of Louisiana at Lafayette, LA

UNIVERSITY of LOUISIANA LAFAYETTE

Contact: saburi.rasheed1@louisiana.edu

Linkedin

Problem Statement

The stiff initial value problem obtained after spatial discretization of reaction-diffusion systems is

$$\begin{cases} u_t = -\mathbf{M}u(t) + N(t, u(t)), \\ u(0) = u_0, \end{cases}$$

where $\mathbf{M}$ is the diffusion matrix and $N(t, u(t))$ is a nonstiff/midly stiff reaction term.

Research Objectives

- Develop a fast and computationally efficient exponential integrator for stiff systems
- Linear stability and convergence analyses
- Experiment with Turing patterns

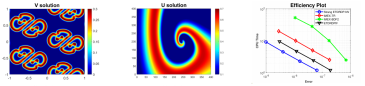
Conclusion

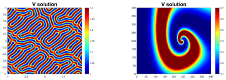
The new Strang-type exponential integrator for stiff problems is of second-order accuracy, L-stable, parallelizable and efficiently solve stiff reaction-diffusion systems under various initial and boundary conditions.

Contribution

- Solves stiff problems with second-order accuracy
- L-stable, and parallelizable numerical algorithm
- Works well for reaction-diffusion equations with mismatched boundary and initial data
- Handles problems with homogeneous Neumann, Dirichlet as well as periodic boundary conditions

Numerical Results





		Strang ETDRDP-NV		
k	h	Error	ROC	Time (s)
0.0200	1.0	9.4105×10^{-7}	0.00	0.00463
0.0100	1.0	2.3590×10^{-7}	2.00	0.01337
0.0050	1.0	5.9117×10^{-8}	2.00	0.01955
0.0025	1.0	1.4827×10^{-8}	2.00	0.03581

Strang ETDRDP-NV solutions with $k = 0.05, h = 1/256$, for $750s$ & $10,000s$ and $k = 1/24, h = 1$ (column 1 & 2 figures), efficiency plot and convergence table (Column 3 figures) of 2D Grey-Scott model and Prey-predator model respectively.

A Novel SAV-based Framework for Kinetic Equations

First Application of the Scalar Auxiliary Variable Method to Kinetic Equations

Motivation & Goal

Numerically solving spatially homogeneous kinetic equations, $\partial_t f = Q(f)$, presents a major challenge in preserving essential physical properties. Our goal is to develop robust, efficient, and structure-preserving schemes.

Contribution: The SAV Framework

- We introduce the **first application** of the **SAV** method to kinetic equations.
- Both **first and second-order** schemes are systematically constructed.
- We incorporate a **Lagrange Multiplier** technique to enforce the **positivity** of the solution.

Key Features & Impact

Our schemes for general homogeneous kinetic equations are:

- **Mass-Conserving**
- **Positivity-Preserving**
- **Modified Entropy-Dissipative**

The accuracy and properties are successfully validated on the challenging **Boltzmann** and **Landau** equations.

Poster: Efficient Neural Network Methods for Numerical Partial Differential Equations

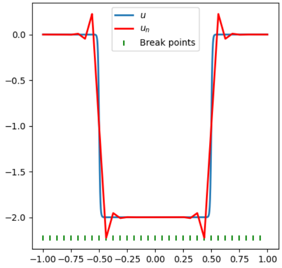
César Herrera

Department of Mathematics, Purdue University

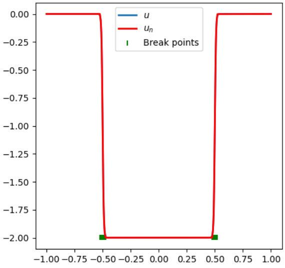
Poster Blitz, ICERM
July 2025

Poster and Research

- **Broader Goals:**
  - **Singular perturbations:** elliptic problems with boundary/interior layers
  - **Conservation laws:** hyperbolic PDEs with shocks/discontinuities
- **Neural Networks**
  - New class of approximating functions
  - Capable of **moving** the mesh



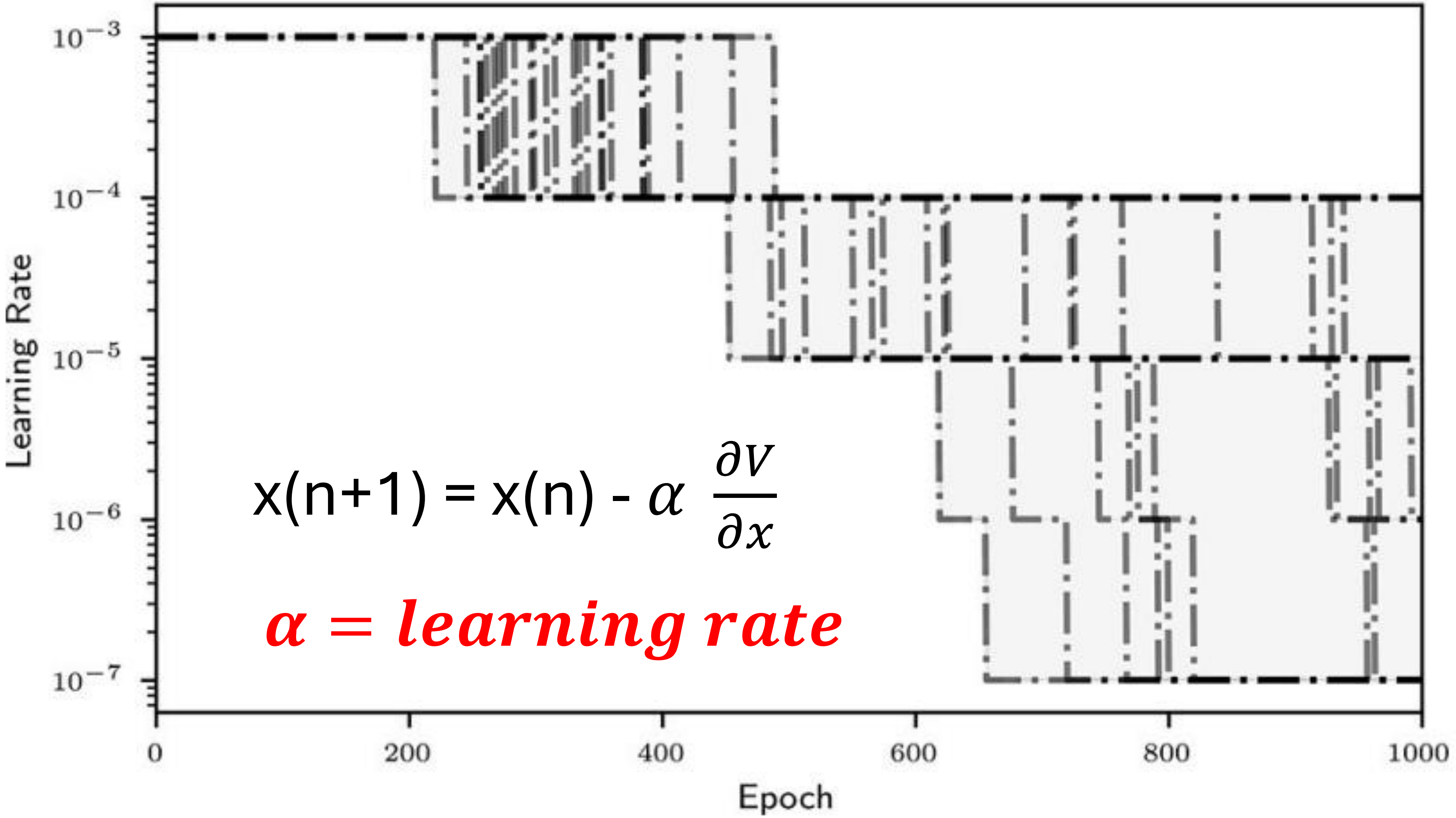
(a) **Fixed** uniform mesh



(b) **Moving** mesh

1. **Stiffness is a big problem in deep learning**
2. Stiffness is due to the ill-conditioning of the Hessian of the loss (this Hessian is the same as the Jacobian of the flow)
3. We fix this problem by avoiding gradient search completely, and we design our flow to minimize the condition number of the Jacobian of our new flow of particles
4. **Our learning rate is typically on the order of 0.1 to 0.01**.
5. In general, stiffness in deep learning is not caused by the stiffness of the physical process being learned (unlike Chris Rackaukas' work on stiffness)
6. Our flow is a SDE rather than a PDE or ODE

the learning rate (adaptively computed) is amazingly small for deep neural nets



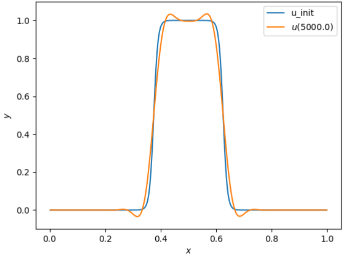
$$x(n+1) = x(n) - \alpha \frac{\partial V}{\partial x}$$

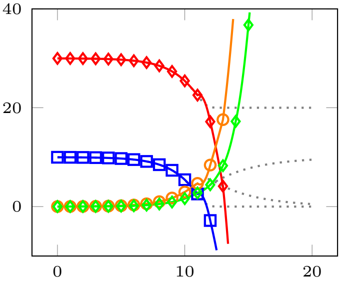
$$\alpha = learning\ rate$$

John Stephens

Joint with: Dr. R. Kirby

```
 0 SNES VI Function norm 0.00207269 Active lower constraints 0/0 upper constra
ints 0/0 Percent of total 0. Percent of bounded 0.
Traceback (most recent call last):
  File "/home/jdsteph2/research/ICERM_poster/CH2D_per.py", line 367, in <module
>
    advance_solver.solve(bounds=(slb, sub))  ## bounds
  File "petsc4py/PETSc/Log.pyx", line 188, in petsc4py.PETSc.Log.EventDecorator
.decorator.wrapped_func
  File "petsc4py/PETSc/Log.pyx", line 189, in petsc4py.PETSc.Log.EventDecorator
.decorator.wrapped_func
  File "/home/jdsteph2/fd/firedrake/firedrake/adjoint_utils/variational_solver.
py", line 108, in wrapper
    out = solve(self, **kwargs)
  File "/home/jdsteph2/fd/firedrake/firedrake/variational_solver.py", line 361,
 in solve
    solving_utils.check_snes_convergence(self.snes)
  File "/home/jdsteph2/fd/firedrake/firedrake/solving_utils.py", line 128, in c
heck_snes_convergence
    raise ConvergenceError(r"""Nonlinear solve failed to converge after %d nonl
inear iterations.
firedrake.exceptions.ConvergenceError: Nonlinear solve failed to converge after
 1 nonlinear iterations.
Reason:
   DIVERGED_FNORM_NAN
```





John Stephens

Joint with: Dr. R. Kirby