Data

Overview

- Euler's Method (1st order)
- Data-Driven Euler's Method
- Runge-Kutta 3rd order method (RK3)
- Runge-Kutta 4th order method (RK4)
- Data-Driven Runge-Kutta (DDRK3)
- Adams Methods: 1st, 2nd, 3rd, 4th, 5th order (AM1-AM5)
- Adams-Bashforth and Adams-Moulton methods
- Data-Driven Adams Methods
- Hierarchical data-driven architecture
- Transformer-inspired ODE solver
- Objective-C framework for Apple platforms

Euler's Method

Algorithm

- Simplest numerical method
- First-order accuracy
- Local truncation error: $O(h^2)$
- Fast computation
- Foundation for higher-order methods

Data-Driven Euler's Method

Enhanced Algorithm

- Hierarchical transformer layers
- Attention mechanisms
- Adaptive correction
- Enhanced accuracy over standard Euler

Runge-Kutta 3rd Order

Algorithm

- Good balance of accuracy and efficiency
- Suitable for nonlinear systems
- Local truncation error: $O(h^4)$

Adams Methods

Adams-Bashforth (Predictor)

$$y_{n+1} = y_n + \frac{h}{12}(23f_n - 16f_{n-1} + 5f_{n-2})$$

- Multi-step methods
- Predictor-corrector scheme
- Higher order accuracy

Parallel and Distributed Execution

Execution Modes

- OpenMP: Shared-memory multi-threading
- pthreads: POSIX threads for fine control
- MPI: Distributed computing
- Hybrid: MPI + OpenMP

- Parallel speedup up to N$\times$ with N workers
- Distributed scaling across nodes
- Concurrent execution of multiple methods

Real-Time, Online, and Dynamic Methods

Execution Modes

- Real-Time: Streaming data, minimal latency
- Online: Adaptive learning, incremental updates
- Dynamic: Adaptive step sizes, parameter tuning

- Suitable for live data feeds
- Adaptive to system changes
- Optimized for continuous operation

Nonlinear Programming Solvers

Methods

- Gradient Descent
- Newton's Method
- Quasi-Newton (BFGS)
- Karmarkar's Algorithm (polynomial-time LP)
- Interior Point
- Sequential QP
- Trust Region

- ODEs as optimization problems
- PDEs as optimization problems
- Polynomial-time guarantees (Karmarkar)
- Enhanced convergence

Combined Solvers

- Distributed + Data-Driven
- Online + Data-Driven
- Real-Time + Data-Driven
- Distributed + Online
- Distributed + Real-Time

Benefits

Maximum flexibility and performance through method combinations

Stacked and Hierarchical Architecture

- Transformer-inspired design
- Multiple processing layers
- Attention mechanisms
- Adaptive refinement
- Data-driven learning

Key Features

- Hierarchical state transformations
- Learnable weights and biases
- Self-attention for ODE solutions
- Adaptive step size control

Implementation

Core

- C/C++ implementation
- High performance
- Memory efficient

Test Cases: Exponential Decay

ODE

Exact: $y(t) = \exp(-t)$

C/C++ Implementation

```
void exponential_ode(double t, const double* y,
  double* dydt, void* params) {
  dydt[0] = -y[0];
}
```

Results

RK3: 0.000034s, 99.999992% accuracy
RK4: 0.000040s, 99.999992% accuracy
DDRK3: 0.001129s, 99.999977% accuracy

Test Cases: Harmonic Oscillator

ODE

Exact: $x(t) = \cos(t)$, $v(t) = -\sin(t)$

C/C++ Implementation

```
void oscillator_ode(double t, const double* y,
  double* dydt, void* params) {
  dydt[0] = y[1]; // dx/dt = v
  dydt[1] = -y[0]; // dv/dt = -x
}
```

Results

RK3: 0.000100s, 99.68% accuracy

RK4: 0.000110s, 99.68% accuracy

Map/Reduce Framework

Distributed ODE Solving

- Partitions state across mapper nodes
- Processes derivatives in parallel
- Aggregates through reducer nodes
- Fault tolerance via redundancy (R=3)

Time Complexity

Optimal with $m = r = \sqrt{n}$ mappers/reducers

Apache Spark Framework

RDD-Based Computation

- Resilient Distributed Datasets (RDDs)
- Lineage-based fault tolerance
- RDD caching for iterative algorithms
- Checkpointing for fast recovery

Performance

With caching: near-constant time per iteration

Karmarkar's Algorithm

Polynomial-Time Interior Point Method

- Solves ODEs as linear programs
- Maintains interior point $x > 0$
- Projective scaling transformations
- Polynomial complexity: $O(n^{3.5}L)$

Convergence

Guaranteed ϵ-optimal solution in polynomial time

Non-Orthodox Architectures

Alternative Computing Paradigms

- Micro-Gas Jet Circuits (fluid dynamics)
- Dataflow (Arvind) - tagged token model
- ACE (Turing) - stored-program computer
- Systolic Arrays - pipelined computation
- TPU (Patterson) - matrix acceleration
- GPUs: CUDA, Metal, Vulkan, AMD
- Standard Parallel: MPI, OpenMP, Pthreads
- GPGPU, Vector Processor, ASIC, FPGA, FPGA AWS F1, DSP
- Quantum: QPU Azure, QPU Intel Horse Ridge
- Specialized: TilePU Mellanox, TilePU Sunway, DPU, MFPU, NPU, LPU, AsAP, Xeon Phi
- Spiralizer Chord (Chandra, Shyamal) - Robert Morris hashing
- Lattice Waterfront (Chandra, Shyamal) - Turing variation

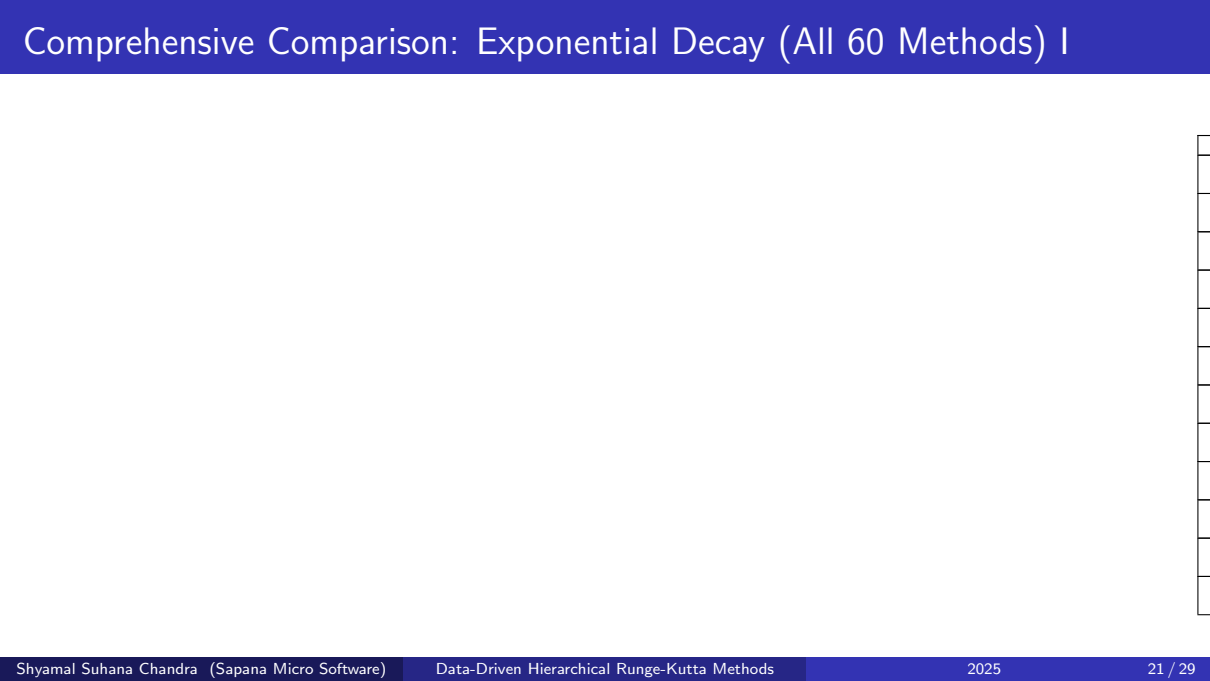
Advanced Features

- Karmarkar's Algorithm (polynomial-time LP)
- Map/Reduce framework for distributed solving
- Apache Spark with RDD caching
- Micro-Gas Jet circuits for low-power computation
- Dataflow (Arvind) for fine-grained parallelism
- ACE (Turing) for historical computation
- Systolic arrays for pipelined operations
- TPU (Patterson) for matrix acceleration
- GPU architectures (CUDA, Metal, Vulkan, AMD)
- Standard parallel computing (MPI, OpenMP, Pthreads)
- GPGPU, Vector Processor, ASIC, FPGA, FPGA AWS F1, DSP
- Quantum Processing Units (QPU Azure, QPU Intel Horse Ridge)
- Specialized Processing Units (TilePU Mellanox, TilePU Sunway, DPU, MFPU, NPU, LPU, A
- Interior Point Methods for non-convex optimization
- Multinomial Multi-Bit-Flipping MCMC
- Online and adaptive algorithms

Applications

- Nonlinear dynamical systems
- Chaotic systems (Lorenz, etc.)
- Engineering simulations
- Scientific computing
- Real-time visualization
- Discrete optimization problems

Comprehensive Comparison: Exponential Decay (All 60 Methods) I



Comprehensive Comparison: Exponential Decay (All 60 Methods) II

Comprehensive Comparison: Exponential Decay (All 60 Methods) III

Comprehensive Comparison: Exponential Decay (All 60 Methods) IV

Comprehensive Comparison: Exponential Decay (All 60 Methods) V

Comprehensive Comparison: Exponential Decay (All 60 Methods) VI

Comprehensive Comparison: Harmonic Oscillator (All 60 Methods) I

Comprehensive Comparison: Harmonic Oscillator (All 60 Methods) II

Comprehensive Comparison: Harmonic Oscillator (All 60 Methods) III

Comprehensive Comparison: Harmonic Oscillator (All 60 Methods) IV

Thank You