adaptive mesh refinement odes

The accurate and efficient numerical solution of ordinary differential equations (ODEs) is crucial across a vast spectrum of scientific and engineering disciplines. When dealing with ODEs that exhibit localized phenomena or require high precision in specific regions, traditional fixed-grid numerical methods can become computationally prohibitive. This is where the power of adaptive mesh refinement (AMR) techniques comes into play, particularly in the context of solving ODEs. Adaptive mesh refinement for ODEs dynamically adjusts the computational grid, concentrating grid points in areas where the solution is rapidly changing or exhibits significant features, while using fewer points in smoother regions. This intelligent resource allocation leads to substantial gains in computational efficiency without sacrificing accuracy. This article will delve into the fundamental principles of adaptive mesh refinement for ODEs, explore various algorithms and strategies, discuss their applications, and highlight the benefits and challenges associated with their implementation.

Unveiling the Power of Adaptive Mesh Refinement for ODEs

The realm of scientific computation frequently encounters problems governed by ordinary differential equations (ODEs). These equations are the bedrock of modeling dynamic systems, from celestial mechanics and chemical kinetics to population dynamics and circuit analysis. However, many real-world ODEs possess inherent characteristics that make their accurate and efficient numerical integration a significant challenge. Specifically, solutions to ODEs can exhibit sharp gradients, rapid oscillations, or localized discontinuities. Traditional fixed-grid numerical solvers, such as the standard Runge-Kutta or Adams-Bashforth methods, discretize the independent variable (often time) into uniformly spaced intervals. While effective for smoothly varying solutions, these methods can become computationally inefficient when applied to problems with localized complexity. To achieve high accuracy in such regions, a very fine grid spacing is required everywhere, leading to an excessive number of computations and memory

requirements. This is precisely where the transformative potential of **adaptive mesh refinement for ODEs** emerges as a sophisticated and powerful solution. By intelligently adapting the computational mesh to the solution's behavior, AMR techniques significantly enhance efficiency and accuracy, making it possible to tackle previously intractable problems.

The Fundamental Principles of Adaptive Mesh Refinement for ODEs

At its core, adaptive mesh refinement for ODEs is a computational strategy designed to optimize the accuracy and efficiency of numerical solvers. Instead of relying on a uniform discretization of the independent variable, AMR dynamically adjusts the grid resolution based on the characteristics of the solution being computed. The primary objective is to concentrate computational effort – more grid points – in regions where the solution exhibits significant changes or requires high precision, and to reduce the computational effort – fewer grid points – in regions where the solution is smooth or less critical. This dynamic adaptation allows for a significant reduction in the overall computational cost compared to fixed-grid methods, especially for problems with localized phenomena.

Why Traditional Methods Fall Short for Complex ODEs

Fixed-grid numerical methods for ODEs operate by discretizing the independent variable (commonly time) into a series of equally spaced steps. Methods like Euler's method, the improved Euler method, or various orders of Runge-Kutta methods rely on this uniform partitioning. While these methods are well-understood and widely used, they face limitations when confronted with ODEs whose solutions present localized, rapid variations. Consider an ODE describing a chemical reaction with a sudden burst of activity, or a physical system experiencing a transient shockwave. To capture these sharp features accurately with a fixed grid, the step size must be uniformly small across the entire domain. This often leads to:

- **Computational Inefficiency:** A large number of calculations are performed in regions where the solution is changing slowly, wasting valuable computational resources.
- **Memory Overheads:** Storing the solution at a very fine resolution across the entire domain can lead to excessive memory consumption, especially for long integration times or high-dimensional systems.
- **Potential for Instability:** In some cases, very small fixed step sizes might be necessary to maintain stability, further exacerbating the computational burden.

The inability of fixed-grid methods to selectively focus computational power on problematic regions necessitates a more dynamic approach, paving the way for the widespread adoption of adaptive mesh refinement for ODEs.

The Core Idea: Dynamically Adjusting Resolution

Adaptive mesh refinement for ODEs addresses the limitations of fixed-grid methods by employing a strategy of dynamic grid adjustment. The fundamental idea is to create a computational mesh that is not static but rather evolves alongside the solution. This evolution typically involves:

- **Refinement:** When the numerical solution indicates a region of rapid change or high error, the mesh in that specific region is subdivided into smaller intervals (cells or points). This increases the local resolution, allowing for a more accurate representation of the solution's behavior.
- **Coarsening:** Conversely, in regions where the solution is smooth and well-approximated by the current grid, the mesh can be coarsened, merging adjacent intervals. This reduces the number of grid points, thereby decreasing computational cost and memory usage.

The decision to refine or coarsen is typically guided by an error estimation procedure. This procedure quantifies the error introduced by the numerical approximation at each step or within specific regions. If the estimated error exceeds a predefined tolerance, refinement is triggered. If the error falls below a lower tolerance, coarsening might be considered.

Key Concepts in Adaptive Mesh Refinement for ODEs

Successfully implementing adaptive mesh refinement for ODEs requires a solid understanding of several core concepts. These concepts dictate how the mesh is adapted and how the numerical solution is managed across these dynamically changing grids.

Error Estimation and Error Control

The backbone of any AMR strategy is a robust error estimation mechanism. This component is responsible for quantifying the error introduced by the numerical discretization and approximation. For ODEs, this often involves estimating the local truncation error (LTE) or comparing solutions obtained with different step sizes or different approximation orders. Common approaches include:

- **Embedded Runge-Kutta Methods:** These methods compute two approximations of the solution at each step using different orders of accuracy. The difference between these approximations provides an estimate of the LTE.
- **Richardson Extrapolation:** This technique involves computing the solution with different step sizes and using the difference between these solutions to estimate the error.

- **Comparison of Time Stepping Methods:** For ODEs, one might compare results from a lower-order method with those from a higher-order method, with the discrepancy indicating error.

The goal of error control is to maintain the estimated error below a user-defined tolerance (e.g., telling the solver to keep the error below 1e-6). This tolerance dictates the level of adaptivity. A smaller tolerance will lead to more frequent refinement and a more accurate, albeit potentially more computationally expensive, solution.

Grid Management and Data Structures

Managing a dynamically changing grid requires specialized data structures. These structures must efficiently store and access the solution values on the refined and coarsened grid. Common data structures used in AMR for ODEs include:

- **Block-Structured Grids:** The computational domain is divided into a collection of blocks, and refinement can occur independently within these blocks. This allows for hierarchical refinement where coarser blocks contain finer sub-blocks.
- **Cell-Centric Structures:** Each grid cell (or interval for ODEs) stores its solution values and pointers to its neighbors, including parents and children in a hierarchical refinement structure.
- **Octrees/Quadtrees (for higher dimensions, but the principle applies):** While more commonly used in PDE contexts, the concept of hierarchical decomposition of the domain into smaller cells is analogous. For ODEs, this translates to a hierarchical subdivision of the independent variable's domain.

Efficient data structures are crucial for minimizing the overhead associated with grid operations, such as adding new grid points, removing them, and interpolating solutions between different grid levels.

Interpolation and Reconstruction

When the grid is refined, new grid points are introduced between existing ones. The solution values at these new points must be determined. Similarly, when coarsening occurs, information from finer grids needs to be appropriately transferred to coarser grids. This process involves interpolation and reconstruction techniques. For ODEs, these might include:

- **Linear Interpolation:** A simple method where the value at a new point is estimated as a linear combination of its nearest neighbors on the coarser grid.

- **Higher-Order Interpolation:** Using polynomial interpolation (e.g., cubic splines) to reconstruct the solution more accurately across grid interfaces.
- **Reconstruction from Differences:** For certain methods, the solution might be represented by differences between adjacent grid points, and these differences are interpolated.

The choice of interpolation method impacts the accuracy and smoothness of the solution across grid boundaries and is a critical component of AMR for ODEs.

Common Adaptive Mesh Refinement Algorithms for ODEs

Several distinct algorithms and approaches have been developed to implement adaptive mesh refinement for ODEs, each with its own strengths and weaknesses. These algorithms primarily differ in how they detect error, manage the grid, and perform the updates.

Time-Stepping Adaptivity (Variable Step Size Methods)

This is perhaps the most fundamental form of adaptivity applied to ODEs. Variable step size methods dynamically adjust the time step size (or the step along the independent variable) based on an error estimate. If the estimated error is too large, the step size is reduced. If the error is small, the step size can be increased.

- **How it works:** A predictor-corrector or embedded Runge-Kutta scheme is often used. After taking a step, an error estimate is calculated. Based on this estimate and a tolerance, the next step size is determined.
- **Pros:** Relatively simple to implement, widely available in standard ODE solvers.
- **Cons:** While adaptive in time step, it doesn't inherently create new grid points within existing intervals; it only adjusts the spacing between them. It can struggle with extremely localized, narrow features that persist for a significant duration.

Block-Based Adaptive Refinement

This approach is more akin to the AMR techniques used for partial differential equations (PDEs), but applied to the one-dimensional domain of an ODE. The computational domain is divided into blocks, and these blocks can be recursively refined.

- **How it works:** The independent variable's domain is initially divided into a coarse grid of blocks. If an error indicator suggests that a particular block needs higher resolution, it is subdivided into smaller blocks (e.g., two or four). This creates a hierarchy of grids with different resolutions.
- **Pros:** Can achieve very high local resolution, suitable for problems with spatially localized features.
- **Cons:** More complex to implement due to the need for sophisticated data structures to manage blocks and grid interfaces. Interpolation between different block resolutions can introduce complexities.

Event-Based Refinement

This strategy focuses on refining the mesh specifically when certain "events" occur in the solution, such as crossing a threshold, reaching a local extremum, or undergoing a significant change in derivative. This is particularly useful for ODEs where critical events drive the system's behavior.

- **How it works:** The solver monitors specific conditions on the solution. When an event is detected, the mesh is refined around the point of the event, and the integration might restart or continue with a finer resolution.
- **Pros:** Efficient for systems where key information is concentrated around specific events.
- **Cons:** Requires a clear definition of what constitutes an "event" and a robust mechanism for detecting these events accurately.

Hierarchical Multigrid Methods for ODEs

While often associated with PDEs, multigrid concepts can be adapted for ODEs, especially when dealing with stiff ODEs or boundary value problems. This involves solving the ODE on a hierarchy of grids.

- **How it works:** The ODE is solved on a coarse grid. Residuals are then projected onto finer grids, and corrections are applied. This process is repeated iteratively, with refinement occurring as needed on the hierarchy.
- **Pros:** Can offer very efficient convergence rates, particularly for problems with varying scales.
- **Cons:** Implementation can be complex, and the effectiveness depends on the specific ODE and the grid structure.

Implementing Adaptive Mesh Refinement for ODEs

Bringing adaptive mesh refinement for ODEs into practice involves several key implementation considerations. These range from selecting the right numerical methods to managing the computational resources effectively.

Choosing the Right ODE Solver

The underlying ODE solver plays a crucial role. For adaptive mesh refinement, solvers that inherently support variable step sizes are essential. Embedded Runge-Kutta methods (e.g., Dormand-Prince, Cash-Karp) are excellent choices as they provide built-in error estimation.

- **Considerations:**
- **Order of Accuracy:** Higher-order methods generally provide better accuracy and more reliable error estimates.
- **Stability:** For stiff ODEs, implicit solvers or specialized methods like Backward Differentiation Formulas (BDFs) might be necessary, and their adaptivity mechanisms need careful consideration.
- **Efficiency:** The computational cost of the solver itself should be balanced against the gains from adaptive refinement.

Error Indicator Design

The effectiveness of AMR hinges on the quality of the error indicators and estimators. These indicators should accurately reflect the true error and be computationally inexpensive to calculate.

- **Types of Errors to Monitor:**
- **Local Truncation Error (LTE):** The error introduced at a single step.
- **Global Error:** The accumulated error over the entire integration interval.
- **Solution Features:** Indicators can also be designed to detect specific solution features like steep gradients, oscillations, or zero crossings.

The threshold for refinement and coarsening (tolerance) needs to be carefully chosen to balance accuracy and efficiency. A single tolerance is often used, but adaptive strategies might employ different tolerances for different aspects of the solution.

Parallelization and Distributed Computing

For large-scale ODE problems, particularly those involving systems of ODEs or requiring extensive computation, parallelization is often necessary. Adaptive mesh refinement introduces challenges and opportunities for parallel implementation.

- **Load Balancing:** As the mesh adapts, the computational load on different processors can become uneven. Dynamic load balancing strategies are required to redistribute work effectively.
- **Communication:** When grids are refined or coarsened, data needs to be exchanged between processors responsible for different parts of the domain. Efficient communication protocols are vital.
- **Data Structures for Parallel AMR:** Specialized distributed data structures are needed to manage the adaptive grids across multiple processing units.

Software Libraries and Frameworks

Implementing AMR from scratch can be a significant undertaking. Fortunately, various scientific computing libraries and frameworks offer support for adaptive mesh refinement, simplifying the development process.

- **Examples:**
- **PETSc (Portable, Extensible Toolkit for Scientific Computation):** Provides tools for adaptive mesh refinement and parallel solution of differential equations.
- **AMR-based Frameworks (often for PDEs, but principles apply):** Libraries like CHOMP, SAMRAI, and OpenFOAM (though primarily for CFD) embody AMR principles that can inspire ODE solutions.
- **MATLAB's ODE Solvers:** Many of MATLAB's built-in ODE solvers (e.g., `ode15s`, `ode45`) inherently implement variable step size adaptivity.

Applications of Adaptive Mesh Refinement in ODE Problems

The ability of adaptive mesh refinement for ODEs to handle localized and transient phenomena makes it invaluable in a wide array of scientific and engineering applications.

Chemical Kinetics and Reaction-Diffusion Systems

Many chemical reactions involve species concentrations that change rapidly over short time scales or in specific spatial regions (when coupled with diffusion). Adaptive methods can accurately capture these transient behaviors without over-resolving the entire system.

- **Example:** Modeling combustion processes, where reaction rates can spike dramatically, or enzyme kinetics where product formation might be very fast initially.

Celestial Mechanics and Orbital Dynamics

Problems involving multiple celestial bodies under gravitational influence often exhibit periods of close approach or highly elliptical orbits, leading to rapid changes in acceleration. Adaptive mesh refinement can efficiently track these dynamic orbital changes.

- **Example:** Simulating the trajectory of asteroids or comets, or studying the dynamics of planetary systems.

Epidemiological Modeling

Tracking the spread of infectious diseases can involve simulating populations where outbreaks occur rapidly in localized areas. Adaptive methods can focus computational effort on these areas of high transmission.

- **Example:** Modeling the spatial spread of a virus, where early detection and containment efforts might require fine resolution in specific communities.

Circuit Simulation

Electronic circuits, especially those involving switching components or fast transient responses, can lead to ODEs with sharp changes in voltage and current. Adaptive solvers can accurately capture these transient behaviors.

- **Example:** Simulating the charging and discharging of capacitors or the behavior of transistors during switching operations.

Fluid Dynamics (when reduced to ODEs)

While fluid dynamics is primarily governed by PDEs, certain simplified models or analyses can reduce the problem to a system of ODEs. For instance, analyzing the dynamics of vortices or the evolution of fluid interfaces might involve ODE systems where localized phenomena are critical.

- **Example:** Analyzing the temporal evolution of specific modes in a fluid flow.

Robotics and Control Systems

The dynamics of robotic systems or the response of control systems to external stimuli can often be described by ODEs. Periods of rapid maneuvering or significant feedback adjustments may benefit from adaptive resolution.

- **Example:** Simulating the trajectory planning for a robot arm executing a complex motion.

Benefits and Challenges of Adaptive Mesh Refinement for ODEs

Adaptive mesh refinement for ODEs offers significant advantages but also presents certain implementation hurdles.

Benefits:

- **Enhanced Accuracy:** By concentrating grid points where needed, AMR can achieve higher accuracy for a given computational budget compared to fixed-grid methods, especially for solutions with localized features.
- **Computational Efficiency:** Significant reductions in computation time and memory usage are possible by avoiding unnecessary calculations in smooth regions.
- **Handling Complex Solutions:** AMR is well-suited for ODEs with solutions that exhibit sharp gradients, discontinuities, or transient behaviors that would be difficult or impossible to resolve with fixed grids.
- **Reduced Human Intervention:** The adaptive nature of the methods reduces the need for manual tuning of grid parameters or time steps by the user.

Challenges:

- **Implementation Complexity:** Developing and implementing robust AMR algorithms requires advanced knowledge of numerical methods, data structures, and potentially parallel computing.
- **Overhead Costs:** The processes of error estimation, grid management, and interpolation introduce computational overhead that can, in some cases, negate the benefits of adaptivity if not carefully managed.
- **Choice of Error Indicator:** Selecting an appropriate and efficient error indicator or estimator is crucial. A poorly chosen indicator can lead to inefficient refinement or inaccurate results.
- **Load Balancing in Parallel Implementations:** Ensuring efficient distribution of work across processors in parallel AMR remains a significant challenge, especially when the computational load changes dynamically.
- **Data Management:** The hierarchical and dynamic nature of adaptive grids can lead to complex data management requirements.

Future Directions in Adaptive Mesh Refinement for ODEs

The field of adaptive mesh refinement for ODEs continues to evolve, with ongoing research focused on improving efficiency, accuracy, and applicability.

- **Advanced Error Estimators:** Development of more sophisticated and reliable error estimators that can better predict and quantify error, leading to more intelligent refinement strategies.
- **Machine Learning Integration:** Exploring the use of machine learning techniques to predict regions requiring refinement or to adapt solver parameters dynamically based on learned solution patterns.
- **High-Dimensional Systems:** Extending AMR techniques to efficiently handle very large systems of ODEs, which are common in areas like molecular dynamics or complex network modeling.
- **Hardware Acceleration:** Leveraging modern hardware architectures, such as GPUs, to accelerate the adaptive refinement process and the underlying ODE solvers.
- **Unified AMR Frameworks:** Developing more general and user-friendly frameworks that can seamlessly apply AMR to a wide range of ODE problems with minimal user intervention.

Conclusion: Mastering ODEs with Adaptive Mesh Refinement

In summary, adaptive mesh refinement for ODEs represents a significant advancement in the numerical solution of differential equations. By intelligently allocating computational resources, AMR techniques enable the accurate and efficient resolution of problems characterized by localized phenomena, sharp gradients, and transient behaviors that often challenge traditional fixed-grid methods. The core principles of error estimation, dynamic grid management, and interpolation are fundamental to the success of adaptive strategies. While variable step-size methods offer a basic form of adaptivity, more advanced approaches like block-based refinement and event-driven strategies provide even greater control over accuracy and efficiency. Applications span a broad range of scientific and engineering disciplines, from chemical kinetics and orbital mechanics to epidemiology and circuit simulation. Despite the inherent complexity in implementation and the need for careful error indicator design, the benefits of enhanced accuracy and computational efficiency are substantial. As research continues to push the boundaries with advanced error estimators, machine learning integration, and hardware acceleration, adaptive mesh refinement for ODEs will undoubtedly remain a cornerstone of advanced scientific computation, empowering researchers to tackle increasingly complex and demanding problems.

Frequently Asked Questions

What are the primary advantages of using Adaptive Mesh Refinement (AMR) for solving Ordinary Differential Equations (ODEs)?

AMR offers significant advantages for ODEs, particularly for problems with localized regions of rapid change or high gradients. Key benefits include improved accuracy by concentrating computational resources where needed, reduced computational cost by avoiding unnecessary refinement in smooth regions, and the ability to handle complex or evolving solution behaviors efficiently. This leads to faster convergence and more accurate solutions with fewer computational resources compared to fixed-grid methods.

In the context of ODEs, what are common error indicators or 'estimators' used to trigger mesh refinement in AMR?

Common error indicators for ODE AMR include estimates of the local truncation error, often derived from comparing solutions computed with different order numerical integrators or by using higher-order approximations. Other indicators can be based on the magnitude of the derivative of the solution or its derivatives, or by detecting rapid changes in the solution's behavior. The goal is to identify regions where the current grid resolution is insufficient to accurately capture the solution's dynamics.

What are some specific applications or types of ODE problems where AMR is particularly beneficial?

AMR excels in ODE problems exhibiting stiff behavior, where solutions change on vastly different time scales. This includes simulations of chemical kinetics with fast and slow reactions, modeling biological systems with rapid cellular processes, and solving systems arising from the discretization of partial differential equations (PDEs) in time, like reaction-diffusion systems or wave propagation problems. Problems with moving fronts or singularities also greatly benefit.

How does the concept of 'grid hierarchy' or 'level-based' refinement work in AMR for ODEs?

In AMR for ODEs, the 'grid' is often a set of discrete time intervals or points. A grid hierarchy involves creating multiple levels of refinement. A coarser grid covers the entire time domain, while finer grids are introduced locally to capture regions where the error indicator suggests higher accuracy is needed. These finer grids are typically nested within coarser grids. The solution is advanced on all active grid levels, with data being exchanged (interpolated or projected) between adjacent levels to maintain consistency.

What are the challenges associated with implementing AMR for ODEs, and how are they typically addressed?

Implementing AMR for ODEs presents challenges such as managing the dynamically changing grid structure, ensuring consistent data transfer between grid levels, and developing robust error estimation strategies. Load balancing can also be an issue in parallel computing. These are addressed through careful data structures (e.g., linked lists, trees), interpolation/projection schemes to handle data movement, adaptive stopping criteria for error estimation, and dynamic task assignment in parallel implementations.

Additional Resources

Here are 9 book titles related to Adaptive Mesh Refinement (AMR) for Ordinary Differential Equations (ODEs), with short descriptions:

1. Adaptive Mesh Refinement for Differential Equations
This foundational text delves into the principles and algorithms behind AMR, specifically focusing on its application to solving differential equations. It covers the theoretical underpinnings, common refinement criteria, and data structures used to manage adaptive grids. Readers will gain a comprehensive understanding of how AMR dynamically adjusts computational resources to accurately capture solution features like discontinuities or high gradients, making it ideal for solving ODEs with localized complexities.

2. Numerical Solution of Ordinary Differential Equations with Adaptive Methods
This book specifically targets the challenges of solving ODEs and presents adaptive techniques as a powerful solution. It explores how AMR can be integrated into ODE solvers to efficiently handle problems where the solution behavior changes rapidly over time or state. The text likely discusses error estimation, time-stepping strategies coupled with spatial adaptation, and the

benefits of AMR for achieving accuracy and computational efficiency.

3. Introduction to Computational Fluid Dynamics: The Finite Volume Method
While primarily focused on CFD, this book often includes significant discussion on AMR techniques, as they are crucial for handling complex flow phenomena. The finite volume method, a common discretization approach in CFD, is well-suited for AMR. It explains how AMR, often implemented with structured or unstructured meshes that are refined in specific regions, significantly improves the accuracy and efficiency of simulating turbulent flows or flows with shock waves.

4. Computational Methods for Partial Differential Equations with Emphasis on Adaptive Techniques
Though the title mentions PDEs, many concepts and techniques for AMR are transferable and highly relevant to ODEs, especially in problems that can be cast as ODEs arising from spatial discretization of PDEs. This book likely covers the mathematical foundations of adaptive methods, including error estimators and hierarchical grids. It would provide insights into how to adapt mesh resolution based on solution properties, which is equally applicable to advanced ODE solvers.

5. The Art of Scientific Computing
This broad title often encompasses chapters or sections dedicated to robust numerical methods, including AMR. It would likely present AMR as a sophisticated tool for tackling computationally demanding problems. The book would probably showcase how AMR enables the accurate simulation of systems where localized phenomena dictate the computational effort, a principle directly applicable to complex ODE systems.

6. High-Order Numerical Methods for Ordinary Differential Equations
This book may explore advanced ODE solvers, and integrating AMR with high-order methods can lead to significant efficiency gains. It would likely discuss how to implement adaptive strategies that maintain high-order accuracy while intelligently refining the mesh where needed. The focus would be on achieving highly accurate solutions to ODEs more efficiently by concentrating computational power on challenging regions.

7. Parallel and Distributed Computation for Scientific Problems
AMR is often employed in parallel computing environments to distribute the computational workload effectively. This book would likely explain how adaptive mesh structures can be managed and refined in a parallel setting for ODE solutions. It would cover strategies for load balancing and communication overhead associated with dynamically changing meshes across multiple processors.

8. Finite Difference Methods for ODEs and PDEs
While finite difference methods are often associated with structured grids, this book might explore extensions to adaptive grids. It would discuss how to implement AMR with finite difference schemes, potentially by using block-structured grids or other adaptive mesh structures. The principles of error estimation and refinement criteria for finite difference approximations would be central to its AMR discussions.

9. Mathematical Modeling and Simulation of Dynamic Systems
This book would frame AMR as a vital tool for accurately simulating dynamic systems described by ODEs, particularly those exhibiting complex, time-evolving behavior. It would likely cover how AMR aids in capturing transient phenomena, bifurcations, or chaotic attractors by adapting the computational mesh in response to the evolving state of the system. The emphasis would be

on the modeling process and how AMR enhances the fidelity of simulations.

Adaptive Mesh Refinement Odes

Adaptive Mesh Refinement Odes

Related Articles

- adam smith economic theory for investment us
- actuarial science problem solving
- adhd child planning skills

Back to Home