Analysis and Computation of Adaptive Moving Grids by Deformation

Pavel Bochev, Guojun Liao, and Gary dela Pena

Department of Mathematics
University of Texas at Arlington
Box 19408, Arlington, TX 76019-0408
Summary

1. Introduction

2. Equidistribution Principle

3. The Moving Grid Method

4. Numerical implementation
   - Least-squares principle for the div-curl system
   - Finite element method

5. Computational Experiments
Introduction

Why Moving Grids?

- Fixed reference domain
  $\Rightarrow$ simplified data structures and reduced overhead; in particular for three-dimensional domains

- Grid points are moved when and where the need arises
  $\Rightarrow$ overall efficiency of the adaptive process can be improved

When to use Moving Grids?

- Transient solutions with sharp gradients
- Complex geometries
- Two and three-dimensional regions

However...

Moving grid methods in two and three dimensions may introduce “mesh tangling”.

We develop efficient grid moving algorithms for one, two and three space dimensions which can be shown to be “mesh tangling” free in a rigorous mathematical way.
Previous Work on Moving Grid Methods


Equidistribution principle

Moving Grid Ingredients

Weight Function.

- \(f(x, t), x \in \mathbb{R}^n, t \in \mathbb{R}, f > 0\), differentiable and has a non-zero lower bound.
- If \(f\) is discrete (from an error estimator) it is interpolated
- Normalization property:
  \[
  \int_{\Omega} \left( \frac{1}{f} - 1 \right) dx = 0.
  \]

Mapping \(\varphi\).

- \(\varphi\) is a one-to-one mapping from the computational domain \(\Omega\) into the physical domain \(\Omega\)

Equidistribution condition.

- \(\det \nabla \varphi(x, t) = f(\varphi(x, t), t), \quad \text{all } x \in \Omega, \text{ and } t > t_0.\)
The Moving Grid Method

We describe a method to construct the diffeomorphism $\varphi$.

1. Define a vector field $\mathbf{v}$ by

\[
\begin{align*}
\text{div } \mathbf{v}(x, t) &= -\frac{\partial}{\partial t} \left( \frac{1}{f(x, t)} \right) \quad \text{in } \Omega \\
\text{curl } \mathbf{v} &= 0 \quad \text{in } \Omega \\
\mathbf{v} \cdot \mathbf{n} &= 0 \quad \text{on } \Gamma.
\end{align*}
\]

2. Define $\varphi$ to be the solution of the ODE system

\[
\begin{align*}
\frac{\partial}{\partial t} \varphi(x, t) &= \mathbf{v}(\varphi, t) \cdot f(\varphi, t) \quad \text{for } t > t_0 \\
\varphi(x, t_0) &= \varphi_0(x) \quad \text{for } t = t_0 \\
det \nabla \varphi_0(x) &= f(\varphi_0(x), t_0).
\end{align*}
\]

**Theorem.**

\[
\frac{\partial}{\partial t} \left( \frac{\det \nabla \varphi(x, t)}{f(\varphi(x, t), t)} \right) = 0 \quad \text{for all } t \geq t_0.
\]

**Corollary.** The mapping $\varphi$ satisfies

\[
\det \nabla \varphi(x, t) = f(\varphi(x, t), t) \quad \text{for all } t \geq t_0, \text{ all } x \in \Omega.
\]

This **deformation method** has its origins in Riemannian geometry:

Numerical Implementation

Successful numerical implementation of the grid moving algorithm combines two tasks:

1. **Generation of the vector field** \( \mathbf{v} \) **via solution of a div-curl system.**
   We seek solution method which
   
   - is suitable for complex geometries
   - does not require significant storage
   - is not computationally demanding
   - has high accuracy
   
   \[ \Rightarrow \text{finite element method of least-squares type} \]
   
   - no additional stability conditions
   - symmetric and positive definite discrete problems
   - assembly-free implementation
   - test and trial spaces need not satisfy the essential BC

2. **Computation of the diffeomorphism** \( \varphi \) **via solution of a system of nonlinear ODE’s.**
   Here we look for
   
   - efficiency
   - high-order approximation
   
   \[ \Rightarrow \text{explicit 4th order Runge-Kutta solver} \]
Least-squares principle

Least-squares quadratic functional

\[ J(u, f, g) = \| \text{div} u - f \|^2_0 + \| \text{curl} u - g \|^2_0 \]

Least-squares variational principle

Seek \( u \in H^1_n(\Omega) \) such that

\[ J(u, f, g) \leq J(v, f, g) \]

for all \( v \in H^1_n(\Omega) \).

Variational problem

Seek \( u \in H^1_n(\Omega) \) such that

\[ B(u, v) = \mathcal{F}(v) \]

for all \( v \in H^1_n(\Omega) \).

where

\[ B(u, v) = (\text{div} u, \text{div} v) + (\text{curl} u, \text{curl} v) \]

\[ \mathcal{F}(v) = (\text{div} v, f) + (\text{curl} v, g) . \]
Finite element solution

Finite element spaces

We assume that $S_h \subset H^1(\Omega)$ is a finite element space with the optimal approximation property

$$
\|u - u^h\|_r \leq Ch^{d+1-r}\|u\|_{d+1}, \quad r = 0, 1.
$$

Then we define the following discrete space

$$S_h = \{u^h \in S_h \times S_h \mid u^h \cdot n = 0 \text{ on } \Gamma\}.$$

Discrete problem

Seek $u^h \in S_h$ such that

$$B(u^h, v^h) = F(v^h) \quad (1)$$

for all $v^h \in S_h$.

Theorem.

The discrete problem (1) has a unique solution $u^h \in S_h$. For all sufficiently regular solutions of the div-curl system we have the error estimate

$$\|u - u^h\|_0 \leq Ch^d\|u\|_{d+1}.$$
Computational Experiments

One-dimensional examples

Example 1

\[ u(x, t) = e^{-\pi^2 t} \sin(\pi x), \quad 0 \leq x \leq 1. \]

Weight function: \( f(x, t) = \frac{1}{\sqrt{1 + \left( \frac{\partial u}{\partial x} \right)^2}}. \)

This function is used to study the stability of grids for time dependent PDE’s.

Example 2

One-dimensional Cubic Schrodinger Equation

\[ i \frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial x^2} + Q|u^2|u = 0. \]

Exact solution \( u(x, t) \) is a soliton traveling to the right with unit velocity and a maximum amplitude of \( \sqrt{2} \):

\[ u(x, t) = \sqrt{2} \exp\{i(0.5x + 0.75t)\} \sech(x - t) \]
Example 3

\[ u(x, t) = \begin{cases} \frac{tc(t)}{\cosh^2(c(t)(x-t-0.3))+1.0-8t} & 0 \leq t \leq 0.1 \\ 0.1c(t) & t > 1 \end{cases} \]

\[ c(t) = 1 + 3[1 + \tanh(20(t - 0.2))]; \quad 0 \leq x \leq 1. \]

Weight function: \[ f(x, t) = \frac{1}{u(x, t)}, \]

Two-dimensional examples

Example 4 (spike)

\[ u(x, y, t) = \sqrt{\frac{td}{\pi}} e^{-tdr} + 1.0 - 0.5t \quad 0 \leq x, y \leq 1, \]

where \( a = 0.5, b = 0.5, d = 40.0 \) and \( r = (x - a)^2 + (y - b)^2. \)

Weight function: \[ f(x, y, t) = \frac{1}{u(x, y, t)}. \]
Example 5 (double spike)

\[ u(x, y, t) = \sqrt{\frac{td}{\pi}} e^{-tdr_1} + ce^{-2tdr_2} + 1.0 - 0.5t \quad 0 \leq x, y \leq 1, \]

where

\[ r_1 = (x - a_1)^2 + (y - b_1)^2, \]
\[ r_2 = (x - a_2)^2 + (y - b_2)^2, \]

\[ a_1 = b_1 = 0.25, a_2 = b_2 = 0.75, d = 40 \text{ and } c = 1. \]

Weight function: \[ f(x, y, t) = \frac{1}{u(x, y, t)}. \]

Example 6

Adaptive grid clustered around a sinusoidal curve.

Weight function:

\[ f(x, y, t) = \begin{cases} 
1 & 0 \leq y < r - 0.2 \\
0.5 - 2.5(y - r)t + (1 - t) & r - 0.2 \leq y \leq r \\
0.5 + 2.5(y - r)t + (1 - t) & r < y \leq r + 0.2 \\
1 & r + 0.2 < y \leq 1 
\end{cases} \]

where \[ r = \frac{1}{2} + \frac{1}{4} \sin(2\pi x). \]