Numerical Construction of Diffeomorphism
and the Applications to Grid Generation and Image Registration

by

XI CHEN

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON
Aug 2016
To my mother Lihua Wang,

and my father Huaizhen Chen
ACKNOWLEDGEMENTS

I would like to thank my supervising professor Dr. Guojun Liao for constantly motivating and encouraging me, and also for his invaluable advice during the 5 years of my doctoral studies. I wish to thank my academic advisors Dr. Benito Chen, Dr. Rencang Li, Dr. Chaoqun Liu and Dr. Jianzhong Su for their interest in my research and for taking time to serve in my graduate committee.

I would also like to extend my appreciation to our department of mathematics lead previously by Dr. Jianping Zhu and now by Dr. Jianzhong Su, for supporting my doctoral studies. I wish to thank all the professors of our department who taught me in these years, especially Dr. Benito Chen, Dr. Rencang Li and Dr. Gaik Ambartsoumian, from whose classes I benefitted most.

I am grateful to all the teachers who taught me during the exactly 20 years I spent in schools, in China and the Unites States. Especially I would like to thank Mr. Benyan Dou, my math teacher in middle school, who taught and encouraged me to self-study when I was a 10-year-old boy. Without him, I wouldn’t be so interested in mathematics and choose it as my major. He inspired me that studying is a lifetime journey, and real studying is to study by myself.

I would like to express my deep gratitude to all the schools and universities I attended, which are Hefei ShiYan school, Hefei No.1 high school, University of Science and Technology of China, and UT Arlington. I am proud to be one part of these
great schools and universities. I wish they will be proud of me in the future.

Finally, I would like to say thank you to all my friends who have helped and
companied me in my time of living in the United States. I am extremely fortunate
to meet you.

May 27, 2016
ABSTRACT

Numerical Construction of Diffeomorphism and the Applications to Grid Generation and Image Registration

Xi Chen, Ph.D.
The University of Texas at Arlington, 2016

Supervising Professor: Guojun Liao

Diffeomorphism is an active research topic in differential geometry. In this area, the existence and construction of diffeomorphism under certain constraints is an interesting and meaningful task. J. Moser first proved the existence of diffeomorphism under a Jacobian determinant constraint. Later, Dr. Liao along with his co-authors, proposed the deformation method to construct diffeomorphisms. A div-curl system is created in the construction of diffeomorphisms. Since the Jacobian determinant has a direct physical meaning in grid generation, i.e. the grid cell size, the deformation method was applied successfully to grid generation and adaptation problems.

In this dissertation, we review the deformation method, focus again on the construction of diffeomorphisms, address clearly a new formation of the deformation problem especially for moving domains. In theory, the deformation method provides one diffeomorphic solution to a nonlinear differential equation.
Inspired by the div-curl system in the deformation method, we developed a new method to construct diffeomorphisms, through a completely different approach. The idea is to control directly the Jacobian determinant and the curl vector of a transformation. Based on calculus of variation and optimization, we proposed a new variational method with prescribed Jacobian determinant and curl vector.

In the study of the two methods of diffeomorphisms construction, we observed the important role of the Jacobian determinant and the curl vector in determining a diffeomorphism. Hence, the corresponding uniqueness problem deserves an investigation. In this dissertation, we discuss this problem by both numerical experiments and theoretical analysis.

Last, we turn to non-rigid image registration, which shares the basic idea of finding transformations. The same equations for divergence and curl vectors are used as constraints to minimize a similarity measure. We designed graphical user interfaces for grid generation and image registration to demonstrate all the methods discussed in this dissertation.
# TABLE OF CONTENTS

ACKNOWLEDGEMENTS ............................... iv  
ABSTRACT ........................................... vi  
LIST OF ILLUSTRATIONS ............................ xi  
LIST OF TABLES .................................... xiii

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 From diffeomorphism to the deformation method of grid generation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 From the deformation method to a new variational method</td>
<td>2</td>
</tr>
<tr>
<td>1.3 From diffeomorphism to non-rigid image registration</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Organization of the dissertation</td>
<td>3</td>
</tr>
<tr>
<td>2. A NOTE ON THE DEFORMATION METHOD FOR MOVING DOMAINS</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>4</td>
</tr>
<tr>
<td>2.2 One-dimensional case</td>
<td>5</td>
</tr>
<tr>
<td>2.2.1 Problem formulation</td>
<td>5</td>
</tr>
<tr>
<td>2.2.2 Theoretical derivation</td>
<td>6</td>
</tr>
<tr>
<td>2.2.3 Numerical implementation and examples</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Two- and Three-dimensional case</td>
<td>18</td>
</tr>
<tr>
<td>2.3.1 Problem formulation</td>
<td>18</td>
</tr>
<tr>
<td>2.3.2 Theoretical derivation</td>
<td>19</td>
</tr>
<tr>
<td>2.3.3 Numerical implementation and examples</td>
<td>23</td>
</tr>
<tr>
<td>3. NEW VARIATIONAL METHOD WITH PRESCRIBED JACOBIAN DETERMINANT AND CURL VECTOR</td>
<td>29</td>
</tr>
</tbody>
</table>
3.1 Introduction ............................................. 29
3.2 New Variational Method: Version 1 ......................... 30
  3.2.1 Problem formulation ................................ 30
  3.2.2 Theoretical derivation ............................... 30
  3.2.3 Algorithm implementation ......................... 32
3.3 Development of New Variational Method: Version 2 ....... 33
  3.3.1 Discussion and improvements ....................... 33
  3.3.2 Derivation of the gradient: 2D ...................... 34
  3.3.3 Derivation of the gradient: 3D ...................... 37
  3.3.4 Algorithm implementation ......................... 39
3.4 Development of New Variational Method: Version 3 ....... 39
3.5 Numerical examples ..................................... 40
3.6 Conclusion .............................................. 44
3.7 Graphical User Interface: GridPanel ....................... 44

4. STUDY ON THE UNIQUENESS OF TRANSFORMATION WITH JACOBIAN DETERMINANT AND CURL VECTOR ......................... 46
  4.1 Introduction ........................................... 46
  4.2 Experiments of Recovering Transformations ............... 47
  4.3 Theoretical analysis .................................. 52
    4.3.1 One diffeomorphism is the identity map .......... 53
    4.3.2 Two diffeomorphisms close to the identity map .. 55
    4.3.3 Conclusion ....................................... 55

5. APPLICATION TO NON-RIGID IMAGE REGISTRATION ............ 57
  5.1 Image registration and transformations .................. 57
  5.2 Optimal control approach .............................. 57
  5.3 New algorithm of nonrigid image registration ............ 58
5.4 Development of non-rigid image registration algorithm  
5.4.1 Simplification  
5.4.2 Symmetric scheme  
5.5 Conclusion  
5.6 Graphical User Interface:  
REFERENCES  
BIOGRAPHICAL STATEMENT
LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The problem we study</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>Illustration of examples 1 and 2: $\phi(\xi, 1)$. Red dots represent numerical solution, blue stars represent analytic solution</td>
<td>14</td>
</tr>
<tr>
<td>2.3</td>
<td>Illustration of example 3: blue stars for $\phi(\xi, 0)$, green circles for $\phi(\xi, 0.5)$, red dots for $\phi(\xi, 1)$</td>
<td>16</td>
</tr>
<tr>
<td>2.4</td>
<td>Illustration of example 4</td>
<td>17</td>
</tr>
<tr>
<td>2.5</td>
<td>Illustration of example 1, red dots represent numerical solution, blue stars represent analytic solution</td>
<td>24</td>
</tr>
<tr>
<td>2.6</td>
<td>Example 2, (a) shows a function $f(d)$ used to define $f(x, t)$. (b) shows the constructed diffeomorphism $\phi(\xi, 1)$</td>
<td>25</td>
</tr>
<tr>
<td>2.7</td>
<td>Example 3, from a known map back to identity map</td>
<td>27</td>
</tr>
<tr>
<td>2.8</td>
<td>Illustration of example 4, here $C_1 = 0.8, C_2 = 5, p = 0.03$</td>
<td>28</td>
</tr>
<tr>
<td>3.1</td>
<td>Illustration of some 2D numerical transformations constructed by the new variational method with prescribed Jacobian determinant and prescribed curl vector</td>
<td>41</td>
</tr>
<tr>
<td>3.2</td>
<td>3D transformation concentrates toward a annulus constructed by the new variational method with prescribed Jacobian determinant and prescribed curl vector</td>
<td>42</td>
</tr>
<tr>
<td>3.3</td>
<td>3D transformation moves toward a sphere’s surface constructed by the new variational method with prescribed Jacobian determinant and prescribed curl vector</td>
<td>43</td>
</tr>
</tbody>
</table>
3.4 GridPanel example .......................................................... 45
3.5 GridPanel example .......................................................... 45
4.1 Experiment 1, 65 × 65 grid nodes. The black star dots * represent \( \phi_0 \), and red dots · represent constructed \( \phi \) ......................................................... 48
4.2 Experiment 1-continued, 65 × 65 grid nodes. The black star dots * represent \( \phi_0 \), and red dots · represent constructed \( \phi \) ......................................................... 49
4.3 Experiment 2, 65 × 65 grid nodes. The black star dots * represent \( \phi_0 \), and red dots · represent constructed \( \phi \) ......................................................... 50
4.4 Experiment 2-continued, 65 × 65 grid nodes. The black star dots * represent \( \phi_0 \), and red dots · represent constructed \( \phi \) ......................................................... 51
5.1 Image Registration Example 1 .............................................. 61
5.2 Image Registration Example 2 .............................................. 62
5.3 illustration of symmetric scheme ........................................... 63
5.4 Image Registration Tools example 1 ..................................... 66
5.5 Image Registration Tools example 2 ..................................... 66
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Comparison of example 1: $\phi(\xi, 0.5)$</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>Comparison of example 1: $\phi(\xi, 1)$</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>Comparison of example 2: $\phi(\xi, 1)$</td>
<td>15</td>
</tr>
<tr>
<td>2.4</td>
<td>Comparison of example 3: $\phi(\xi, 1)$</td>
<td>16</td>
</tr>
<tr>
<td>2.5</td>
<td>Comparison of example 4: $\phi(\xi, 1)$</td>
<td>17</td>
</tr>
<tr>
<td>2.6</td>
<td>$L^2$-norm of the difference $J(\phi)$ and $f(\phi)$ at each time step $t$</td>
<td>26</td>
</tr>
<tr>
<td>4.1</td>
<td>Comparison of Experiment 1</td>
<td>49</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of Experiment 1</td>
<td>51</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

1.1 From diffeomorphism to the deformation method of grid generation

In differential geometry, a diffeomorphism is map between manifolds which is
differentiable and has a differentiable inverse, and it is an active research topic. In 1990,
B.Dacorogna and J.Moser [1] proved the existence of diffeomorphisms \( \phi : \Omega \rightarrow \Omega \) such
that
\[
\begin{cases}
\det \nabla \phi(\xi) = f(\xi) & \xi \in \Omega, \\
\phi(\xi) = \xi & \xi \in \partial \Omega.
\end{cases}
\] (1.1)

Liu[6, 7], S.Osher [7], D.Fleitas [8], G. Pena [7, 9], Z.Lei[10], improved B.Dacorogna
and J. Moser’s technique, and proposed the deformation method of grid generation.
In this method, a diffeomorphism \( \phi : \Omega \rightarrow \Omega \) is numerically constructed, such that \( \phi \)
maps a grid of \( \Omega \) to a new grid of \( \Omega \), whose grid sizes (approximated by the Jacobian
determinant of \( \phi \), namely \( \det \nabla \phi \)) equal to a prescribed scalar monitor function \( f > 0 \)
at \( \phi(\xi) \). Correspondingly, the PDE 1.1 is extended to the following one, which is
more meaningful in the physical domain:
\[
\begin{cases}
\det \nabla \phi(\xi) = f(\phi(\xi)) & \xi \in \Omega, \\
\phi(\xi) = \xi & \xi \in \partial \Omega.
\end{cases}
\] (1.2)

This is the steady version of the deformation method. A series of applications were
made, including adaptive moving grid [11, 12, 10], and steady euler flow calculations
[6]. In the meantime, the dynamic version of the deformation method on fixed domains
was developed in [7],[9],[13], based on solving Poisson’s equations.
A milestone occurred in 2004. The Least-Squares Finite Element Method was first applied to solve the div-curl system in [14](see (2.10)), which extends the deformation method of grid generation to moving domains. This version constructs numerically a diffeomorphism $\phi : \Omega_0 \to \Omega_t$, such that for $\forall t \in [0, 1]$

$$
\begin{align*}
\det \nabla \phi(\xi, t) &= f(\phi(\xi, t), t) \quad \xi \in \Omega_0, \\
\phi(\xi, 0) &= \xi, \\
\phi(\xi, t) &\in \partial \Omega_t \quad \xi \in \partial \Omega_0.
\end{align*}
$$

(1.3)

In the first part of this dissertation (chapter 2), we review the mathematical foundation of the deformation method of grid generation, and propose a new formation of the moving domain problem.

1.2 From the deformation method to a new variational method

The success of the deformation method of grid generation actually relies on the div-curl system, it makes us notice the significant importance of the curl vector ($\nabla \times$) in the construction of a diffeomorphism. Therefore, we tried to explore another completely different approach by directly controlling the Jacobian determinant ($\det \nabla$) and the curl vector ($\nabla \times$). We used calculus of variation to formulate a new variational method with prescribed Jacobian determinant and curl vector of constructing transformations[15]. A recovering experiment was designed by the new variational method, which leads to a potential uniqueness problem about transformations with the same Jacobian determinant and curl vector. In the second part of this dissertation (chapter 3 and 4), we discuss the new variational method and study the uniqueness problem.
1.3 From diffeomorphism to non-rigid image registration

Image registration is the process of establishing a one-to-one correspondence between pixels of two images such that a similarity measure (energy function) is optimized. It’s about to construct transformations. In order to determine the one-to-one correspondence accurately and efficiently, we can optimize a similarity measure under some similar constraints to the *div-curl system of the deformation method*. In 2008, G. Liao proposed optimal control approach in [16] and further developed it in [17]. Our Image registration algorithms, along with a symmetric scheme are described in the third part of this dissertation (chapter 5).

1.4 Organization of the dissertation

There are 5 chapters in this dissertation. Chapter 1 is the Introduction. Chapter 2 presents a note on the *deformation method* for moving domains. Chapter 3 proposes a *new variational method*. Chapter 4 studies a uniqueness problem. Chapter 5 focuses on image registration problems.
CHAPTER 2

A NOTE ON THE DEFORMATION METHOD FOR MOVING DOMAINS

2.1 Introduction

The deformation method of grid generation has been extended to moving domain problem since 2004, when LSFEM was introduced to solve the div-curl system[14]. As a competitive method in solving practical grid generation and adaptation problems, the deformation method of grid generation usually focuses on constructing one diffeomorphism. Namely, for a given monitor function $f_0(x)$, let $f(x,t) = 1 - t + tf_0(x)$ on $t \in [0,1]$, apply the deformation method to construct $\phi(\xi,1)$ such that $J(\phi(\xi,1)) = f_0(\phi(\xi,1))$. (Note: $J = \det \nabla$, means the Jacobian determinant, same hereinafter). The intermediate steps are not cared, and the difference between $J(\phi(\xi,1))$ and $f_0(\phi(\xi,1))$ is not carefully examined. Also there are some ambiguities in boundary conditions.

Here we want to review the the deformation method back again as a mathematical method to construct diffeomorphisms. We carefully address the conditions and equations, clearly set up the boundary conditions, examine the domain issues, moreover, prove the existence of the solution. Indeed, the deformation method is a method to construct a family of diffeomorphisms $\phi(x,t)$ with property $J(\phi) = f(\phi,t)$ for a given monitor function $f(x,t)$, and the property is guaranteed to be true for any $t$.

Before starting our review of the deformation method, let’s refresh the problem we study again by the following illustration. Given domains $\Omega_t$ and a monitor function $f(x,t)$, we want to construct diffeomorphisms $\phi(\xi,t)$ from $\Omega_0$ to $\Omega_t$, such that $J(\phi(\xi,t)) = f(\phi(\xi,t))$. 

4
We now start our review in one-dimensional case (1D) first, then move to general two- and three-dimensional cases.

2.2 One-dimensional case

2.2.1 Problem formulation

Let $\Omega_t := [a(t), b(t)] \subset \mathbb{R}$ be smoothly moving domains, i.e. $a(t), b(t)$ is differentiable, with $\Omega_0 = [a, b]$. Given a function $f(x, t) \in C^1(x,t) > 0$ on domain of $(x,t): \Omega_t \times [0, 1]$, such that

$$f(x,0) = 1, \quad (2.1)$$

$$\int_{\Omega_t} \frac{1}{f(x,t)} dx = |\Omega_0| = b - a. \quad (2.2)$$

A diffeomorphism

$$\phi(\xi, t) : \Omega_0 \rightarrow \Omega_t$$

such that $\forall t \in [0, 1]$

$$\phi_\xi(\xi, t) = f(\phi(\xi, t), t) \quad (2.3)$$

can be constructed by solving the following 2 differential equations (2.4) and (2.5).
First, determine $u(x, t)$ on $\mathbb{R}$ by solving:

$$
\begin{cases}
    u_x = -\frac{\partial}{\partial t} \left( \frac{1}{f(x, t)} \right), \\
    u(a(t), t) = \frac{a'(t)}{f(a(t), t)}, \quad \text{or} \\
    u(b(t), t) = \frac{b'(t)}{f(b(t), t)}.
\end{cases}
$$

(2.4)

Second, determine $\phi(\xi, t)$ on $\Omega_0$ by solving:

$$
\begin{cases}
    \frac{\partial \phi}{\partial t} (\xi, t) = f(\phi(\xi, t), t)u(\phi(\xi, t), t), \\
    \phi(\xi, 0) = \xi.
\end{cases}
$$

(2.5)

We name this method the deformation method (1D).

2.2.2 Theoretical derivation

It is clear that the above 2 differential equations (2.4) and (2.5) are both solvable (see Remark 1 below about the domain issue of (2.4), Remark 2 about the boundary condition of (2.4)). So here we need to prove:

S1. $\phi_x(\xi, t)$ exists;

S2. (2.3) is satisfied, namely $\phi_x(\xi, t) = f(\phi(\xi, t), t)$;

S3. $\phi(a, t) = a(t)$ and $\phi(b, t) = b(t)$ for $\forall t$, namely $\phi(\xi, t)$ does map $\Omega_0$ onto $\Omega_t$;

in order to claim $\phi(\xi, t)$ is the desired diffeomorphism from $\Omega_0$ to $\Omega_t$.

S1. comes naturally true from the following theorem about smooth dependence on initial conditions of ordinary differential equations [18].

**Theorem 1.** Let $\Omega \subset \mathbb{R}^{n+1}$ be an open set, and suppose that $f : \Omega \rightarrow \mathbb{R}^n$ is $C^1$. For $(s, p) \in \Omega$, the unique local solution $x(t, s, p)$ of the initial value problem

$$
\frac{d}{dt} x(t, s, p) = f(t, x(t, s, p)), \quad x(s, s, p) = p
$$

is $C^1$ in its open domain of definition

$$
D = \{ (t, s, p) \in \mathbb{R}^{n+2} : \alpha(s, p) < t < \beta(s, p), (s, p) \in \Omega \}.
$$
The differential matrix $D_p x(t, s, p)$ satisfies the so-called linear variational equation

$$\begin{aligned}
\frac{d}{dt} D_p x(t, s, p) &= D_x f(t, x(t, s, p)) D_p x(t, s, p), \\
D_p x(s, s, p) &= I.
\end{aligned}$$

**Proof of S1.** Simply let $n = 1$, $(s, p) = (0, \xi)$, $x(t, s, p) = \phi(t, \xi)$, $f = f(x, t)u(x, t)$ in Theorem 1, we can get $D_p x(t, s, p) = \phi(t, \xi) = \phi_\xi(t, \xi)$ exists. □

After (2.4), (2.5) and S1., we now have a differentiable map $\phi(\xi, t) : \Omega_0 \to \mathbb{R}$. Proofs of S2. and S3. continue as follows:

**Proof of S2.** Define $H(\xi, t) = \frac{\phi_\xi(\xi, t)}{f(\phi, \xi, t)}$ on $\Omega_0 \times [0, 1]$ (also see Remark 1 about the domain issue of $f(\phi, \xi, t)$ here). Consequently $H(\xi, 0) = 1$.

Our goal here is to prove $\frac{dH(\xi, t)}{dt} \equiv 0$, then $H(\xi, t) \equiv H(\xi, 0) = 1$.

We have

$$\frac{dH(\xi, t)}{dt} = \left[ \frac{d}{dt} \phi_\xi(\xi, t) \right] \frac{1}{f(\phi(\xi, t), t)} + \phi_\xi(\xi, t) \left[ \frac{d}{dt} \left( \frac{1}{f(\phi(\xi, t), t)} \right) \right]$$

$$= \frac{d}{dt} \phi_\xi(\xi, t) f(\phi(\xi, t), t) - \phi_\xi(\xi, t) \frac{d}{dt} f(\phi(\xi, t), t)\frac{1}{f(\phi(\xi, t), t)^2}.$$ 

The numerator $\frac{d}{dt} \phi_\xi(\xi, t) f(\phi(\xi, t), t) - \phi_\xi(\xi, t) \frac{d}{dt} f(\phi(\xi, t), t)$

$$= \left[ \phi_r(t, \xi) f(\phi(\xi, t), t) - \phi_\xi(\xi, t) \right] f_x(\phi(\xi, t), t) \phi_t(\xi, t) + f_t(\phi(\xi, t), t)$$

$$= \left[ f(\phi(\xi, t), t) u(\phi(\xi, t), t) \right] \phi(\phi(\xi, t), t) - \phi_\xi(\xi, t) \left[ f_x(\phi(\xi, t), t) \phi_t(\xi, t) \\
+ f_t(\phi(\xi, t), t) \right]$$

$$= \left[ f_x(\phi(\xi, t), t) \phi_\xi(\xi, t) u(\phi(\xi, t), t) + f(\phi(\xi, t), t) u_x(\phi(\xi, t), t) \phi_\xi(\xi, t) \right]$$

$$f(\phi(\xi, t), t) - \phi_\xi(\xi, t) \left[ f_x(\phi(\xi, t), t) f(\phi(\xi, t), t) u(\phi(\xi, t), t) + f_t(\phi(\xi, t), t) \right]$$

$$= f^2(\phi(\xi, t), t) u_x(\phi(\xi, t), t) \phi_\xi(\xi, t) - \phi_\xi(\xi, t) f_t(\phi(\xi, t), t)$$

$$= f^2(\phi(\xi, t), t) u_x(\phi(\xi, t), t) - f_t(\phi(\xi, t), t) \phi_\xi(\xi, t)$$

$$= \left[ f^2(x, t) u_x(x, t) - f_t(x, t) \right] \phi_\xi(\xi, t)$$

$$= 0.
Hence, \( H(\xi, t) = \frac{\phi_\xi(\xi, t)}{f(\phi, \xi, t)} \equiv 1 \implies \phi(\xi, t) \equiv f(\phi(\xi, t), t) \). \( \square \)

**Proof of S3.** The boundary condition of (2.4) gives us

\[ a'(t) = f(a(t), t)u(a(t), t). \]

And let \( \xi = a \) in (2.5), we have

\[
\begin{align*}
\begin{cases}
\frac{\partial \phi}{\partial t}(a, t) = f(\phi(a, t), t)u(\phi(a, t), t), \\
\phi(a, 0) = a = a(0).
\end{cases}
\end{align*}
\]

Since \( f(y, t) \in C^1(y, t) \), and \( u(y, t) \) is \( C^1 \) in \( y \), continuous in \( t \), by Picard’s existence theorem, the ordinary differential equation (ODE) initial value problem

\[
\begin{align*}
\begin{cases}
\frac{\partial y}{\partial t} = f(y, t)u(y, t), \\
y(0) = a,
\end{cases}
\end{align*}
\]

has a unique solution. Comparing the above two, we must have \( \phi(a, t) = a(t) \).

Then, let’s recall S2. and use integration by substitution to get

\[
\int_{\phi(a, t)}^{\phi(b, t)} \frac{1}{f(x, t)} dx = \int_{\phi(a, t)}^{\phi(b, t)} \frac{1}{f(\phi(\xi, t), t)} d\phi(\xi, t)
\]

\[
= \int_{a}^{b} \frac{\phi_\xi(\xi, t)}{f(\phi(\xi, t), t)} d\xi
\]

\[
= \int_{a}^{b} 1 d\xi = |\Omega_0|
\]

\[
= \int_{a(t)}^{b(t)} \frac{1}{f(x, t)} dx. \quad \text{by (2.2)}
\]

Noticing that \( \phi(a, t) = a(t) \) and \( f(x, t) > 0 \), we must have \( \phi(b, t) = b(t) \). \( \square \)

Now, we finish the theoretical derivation of the deformation method(1D). Given domains \( \Omega_t \) and a scalar function \( f(x, t) \) on \( \Omega_t \) with certain properties (2.1,2.2), a diffeomorphism \( \phi(\xi, t) : \Omega_0 \to \Omega_t \) with \( \phi_\xi(\xi, t) = f(\phi(\xi, t), t) \) can be constructed.
Several remarks are listed below, Remark 1, 2 refine the derivation, Remark 3, 4, 5 discuss some special cases.

We studied the 1D case in order to get some simple but fundamental ideas about the general deformation approach, and in fact, most of the results here can be extended naturally to general case. We will discuss the general case later. In the next sections, a numerical algorithm and some numerical examples are presented to show the correctness and accuracy of the deformation method (1D).

**Remark 1.** Solving differential equation (2.4) on $\mathbb{R}$ doesn’t conflict with $f(x,t)$ is only defined on $\Omega_t$. Because first, theoretically we can smoothly extend $f(x,t)$ from $\Omega_t$ to $\mathbb{R}$ such that (2.4) is solvable on $\mathbb{R}$. Second, later we know $\phi(\xi,t)$ maps $\Omega_0$ onto $\Omega_t$ and only $u(x,t) \in \Omega_t$ is used in (2.5). So we only need to solve (2.4) on $\Omega_t$.

**Remark 2.** The boundary condition in (2.4) guarantees the range of $\phi(\xi,t)$, and it can be substituted with $u(b(t),t) = \frac{b'(t)}{f(b(t),t)}$, namely the 2 boundary conditions are equivalent to each other. A brief proof is given here.

**Proof of Remark 2.** Suppose $x = x(\xi,t)$ is any $C^1$ map from $\Omega_0$ to $\Omega_t$. By change of variables, we have

$$0 = \frac{d}{dt} \int_{\Omega_t} \frac{1}{f(x,t)} dx = \frac{d}{dt} \int_{\Omega_0} \frac{1}{f(x,t)} x_\xi d\xi = \int_{\Omega_0} \frac{1}{f(x,t)} \frac{d}{dt} (x_\xi) d\xi$$

$$\quad = \int_{\Omega_0} x_\xi \frac{d}{dt} (\frac{1}{f(x,t)}) d\xi + \int_{\Omega_0} \frac{1}{f(x,t)} \frac{d}{dt} (x_\xi) d\xi$$

$$\quad = - \int_{\Omega_0} \frac{x_\xi (f_x x_t + f_t)}{f^2(x,t)} d\xi + \int_{\Omega_0} \frac{(x_t)_\xi}{f(x,t)} d\xi$$

$$\quad = - \int_{\Omega_0} \frac{x_\xi f_t}{f^2(x,t)} d\xi - \int_{\Omega_0} \frac{x_\xi f_x x_t}{f^2(x,t)} dx + \int_{\Omega_0} \frac{(x_t)_\xi}{f(x,t)} d\xi$$

$$\quad = \int_{\Omega_t} \frac{\partial}{\partial t} (\frac{1}{f(x,t)}) dx + \int_{\Omega_0} \frac{(x_t)}{f(x,t)} d\xi.$$
Which leads to \( \int_{\Omega_0} \frac{x_t}{f(x,t)} \xi d\xi = -\int_{\Omega_t} \frac{\partial}{\partial t} \frac{1}{f(x,t)} \xi d\xi = \int_{\Omega_t} u_x dx, \) and
\[
\Rightarrow \frac{x_t}{f(x,t)} \bigg|_a^b = \int_{a(t)}^{b(t)} u_x dx = u \bigg|_{a(t)}^{b(t)},
\]
\[
\Rightarrow \frac{b'(t)}{f(b(t),t)} - \frac{a'(t)}{f(a(t),t)} = u(b(t)) - u(a(t)),
\]
\[
\Rightarrow u(b(t)) = \frac{b'(t)}{f(b(t),t)} \iff u(a(t)) = \frac{a'(t)}{f(a(t),t)}.
\]

\( \square \)

**Remark 3.** One special case in practice is letting \( \Omega_t \equiv \Omega_0 \), namely the fixed domain problem. In this special case, \( f(x,t) > 0 \) should be given on \( \Omega_0 \times [0,1] \), such that
\[
f(x,0) = 1, \quad (2.1')
\]
\[
\int_{\Omega_0} \frac{1}{f(x,t)} dx = |\Omega_0| = b - a. \quad (2.2')
\]

After solving differential equations (2.4) and (2.5), we can get \( \phi(\xi,t) : \Omega_0 \to \Omega_0 \) which satisfies (2.3).

**Remark 4.** Another special case is \( f(x,t) = \frac{b(t) - a(t)}{b-a} \), where now we can get \( \phi(\xi,t) = \frac{b(t) - a(t)}{b-a}(\xi - a) + a(t) \) after solving differential equations (2.4) and (2.5). As expected, \( \phi(\xi,t) \) is a linear transformation from \( \Omega_0 \) to \( \Omega_t \). A brief proof is also given here.

**Proof of Remark 4.** For short, let \( g(t) = b(t) - a(t), c = b-a \), so \( \frac{1}{f(x,t)} = \frac{c}{g(t)} \). (2.4) is now
\[
\begin{cases}
  u_x = \frac{cg'(t)}{g^2(t)}, \\
  u(a(t),t) = \frac{ca'(t)}{g(t)}.
\end{cases}
\]

Simply integrate it to get
\[
u(x,t) = \frac{cg'(t)}{g^2(t)} x - \frac{cg'(t)a(t)}{g^2(t)} + \frac{ca'(t)}{g(t)}.
\]

Plugging it into (2.5), we have
\[
\phi_t = \frac{g'(t)}{g(t)} \phi - \frac{g'(t)a(t)}{g(t)} + a'(t).
\]
which is
\[(\frac{\phi}{g(t)})' = (\frac{a(t)}{g(t)})'.\]

Therefore
\[\frac{\phi}{g(t)} = \frac{a(t)}{g(t)} + h(\xi) \implies \phi(\xi, t) = a(t) + g(t)h(\xi).\]

Considering the boundary condition \(\phi(\xi, 0) = \xi\), we can get \(h(\xi) = \frac{\xi - a}{b - a}\). Finally,
\[\phi(\xi, t) = a(t) + \frac{b(t) - a(t)}{b - a}(\xi - a).\]

\[\square\]

**Remark 5.** A simple way of constructing \(\phi(\xi, t)\) in 1D case is to directly solve the ordinary differential equation on \(\Omega_0\) for any fixed \(t\):

\[
\begin{aligned}
\begin{cases}
\phi_\xi(\xi, t) = f(\phi(\xi, t), t), \\
\phi(a, t) = a(t).
\end{cases}
\end{aligned}
\tag{2.6}
\]

It looks like in 1D case, the deformation method (1D) may not have advantages over directly solving ODE (2.6). However, in general 2D and 3D cases, where we cannot directly solve the ODE, the deformation method still works in the construction of diffeomorphisms. Please check the following sections for details.

### 2.2.3 Numerical implementation and examples

#### 2.2.3.1 Algorithm implementation

We implemented the deformation method (1D) by the following 4-step algorithm, which bases on a scheme of multiple time steps:

- **Step 1: Initialize.** Starts from \(t = 0\), \(\phi(\xi, 0) = \xi, f(x, 0) = 1\).
- **Step 2: ODE (2.4).** Compute \(-\frac{\partial}{\partial t}(\frac{1}{f(x, t)})\), then solve (2.4) to get \(u(x, t)\) on \(\Omega_t\).
• **Step 3: ODE (2.5).** Update $\phi(\xi,t)$ from $\Omega_t$ to $\Omega_{t+dt}$ by (2.5).

• **Step 4: Next time step.** Move to next time step $t = t + dt$, back to Step 2 till $t = 1$.

### 2.2.3.2 Precision analysis

When using numerical methods to implement the above algorithm of constructing $\phi(\xi,t)$, the precision relies on the following parts:

1. Evaluating $-\frac{\partial}{\partial t} \left( \frac{1}{f(x,t)} \right)$, with error $O(h_t)$ or higher order.

2. Solving ODE (2.4), with error $O(h_x)$ or higher order.

3. Solving ODE (2.5), with error $O(h_t)$ or higher order.

4. Potentially evaluating $f(\phi(\xi,t),t)$ by interpolation, with error $O(h_x)$ or higher order.

### 2.2.3.3 Numerical Example 1: moving domains with $f(x,t)$ normalized already

Let $t \in [0,1], \Omega_t = [0, 1 - 0.9t], \text{ and } f(x,t) = 1 - 0.9t$. The analytic solution to this problem is

$$\phi(\xi,t) = (1 - 0.9t)\xi.$$

We use 11 equally spaced nodes on $[0,1]$ and 11 equally distributed time steps from 0 to 1, i.e. $h_x = h_t = 0.1$. Apply second order numerical methods for each part, we get the numerical solutions as follows:
<table>
<thead>
<tr>
<th>Nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytic solution</td>
<td>0</td>
<td>0.055</td>
<td>0.110</td>
<td>0.165</td>
<td>0.2200</td>
<td>0.2750</td>
</tr>
<tr>
<td>numerical solution</td>
<td>0</td>
<td>0.05489</td>
<td>0.10979</td>
<td>0.16468</td>
<td>0.21958</td>
<td>0.27447</td>
</tr>
<tr>
<td>error</td>
<td>0</td>
<td>1.05e-4</td>
<td>2.1e-4</td>
<td>3.15e-4</td>
<td>4.2e-4</td>
<td>5.25e-4</td>
</tr>
<tr>
<td>Nodes</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>analytic solution</td>
<td>0.3300</td>
<td>0.3850</td>
<td>0.4400</td>
<td>0.4950</td>
<td>0.5500</td>
<td></td>
</tr>
<tr>
<td>numerical solution</td>
<td>0.32937</td>
<td>0.38426</td>
<td>0.43916</td>
<td>0.49405</td>
<td>0.54895</td>
<td></td>
</tr>
<tr>
<td>error</td>
<td>6.3e-4</td>
<td>7.35e-4</td>
<td>8.39e-4</td>
<td>9.44e-4</td>
<td>10.49e-4</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1. Comparison of example 1: $\phi(\xi, 0.5)$.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytic solution</td>
<td>0</td>
<td>0.01</td>
<td>0.02</td>
<td>0.03</td>
<td>0.04</td>
<td>0.05</td>
</tr>
<tr>
<td>numerical solution</td>
<td>0</td>
<td>0.00978</td>
<td>0.01956</td>
<td>0.02934</td>
<td>0.03913</td>
<td>0.04891</td>
</tr>
<tr>
<td>error</td>
<td>0</td>
<td>2.18e-4</td>
<td>4.35e-4</td>
<td>6.53e-4</td>
<td>8.7e-4</td>
<td>10.88e-4</td>
</tr>
<tr>
<td>Nodes</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>analytic solution</td>
<td>0.06</td>
<td>0.07</td>
<td>0.08</td>
<td>0.09</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>numerical solution</td>
<td>0.05869</td>
<td>0.06847</td>
<td>0.07826</td>
<td>0.08804</td>
<td>0.09782</td>
<td></td>
</tr>
<tr>
<td>error</td>
<td>13.06e-4</td>
<td>15.23e-4</td>
<td>17.41e-4</td>
<td>19.58e-4</td>
<td>21.76e-4</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.2. Comparison of example 1: $\phi(\xi, 1)$. 
2.2.3.4 Example 2: fixed domain

Let $t \in [0, 1], \Omega_0 = \Omega_t = [0, 1], f(x, t) = 1 - t + t\sqrt{2x + \frac{1}{4}}$ and be normalized on $\Omega_t$ by $f(x, t) = f(x, t) \int_{\Omega_t} \frac{1}{f(x, t)} dx$. The analytic solution at $t = 1$ to the problem is

$$\phi(\xi, 1) = \frac{1}{2}(\xi^2 + \xi).$$
Similar to example 1, use 11 equally spaced nodes on \([0, 1]\) and 11 equally distributed time steps from 0 to 1, i.e. \(h_x = h_t = 0.1\). Apply second order numerical methods for each part, we get the numerical solutions as follows:

<table>
<thead>
<tr>
<th>Nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytic solution</td>
<td>0</td>
<td>0.0550</td>
<td>0.1200</td>
<td>0.1950</td>
<td>0.2800</td>
<td>0.3750</td>
</tr>
<tr>
<td>numerical solution</td>
<td>0</td>
<td>0.0567</td>
<td>0.1198</td>
<td>0.1935</td>
<td>0.2782</td>
<td>0.3731</td>
</tr>
<tr>
<td>error</td>
<td>0</td>
<td>0.0017</td>
<td>-0.0002</td>
<td>-0.0015</td>
<td>-0.0018</td>
<td>-0.0019</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nodes</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytic solution</td>
<td>0.4800</td>
<td>0.5950</td>
<td>0.7200</td>
<td>0.8550</td>
<td>1.0000</td>
</tr>
<tr>
<td>numerical solution</td>
<td>0.4782</td>
<td>0.5935</td>
<td>0.7190</td>
<td>0.8545</td>
<td>1.0000</td>
</tr>
<tr>
<td>error</td>
<td>-0.0018</td>
<td>-0.0015</td>
<td>-0.0010</td>
<td>-0.0005</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2.3. Comparison of example 2: \(\phi(\xi, 1)\).

2.2.3.5 Example 3: moving domains with \(f(x,t) = \frac{b(t)-a(t)}{b-a}\)

Let \(t \in [0, 1], \Omega_t = [1 + t \sin t, 2 + \frac{1}{2} t^2]\), \(f(x,t) = \frac{b(t)-a(t)}{b-a} = 1 + \frac{1}{2} t^2 - t \sin t\). As mentioned in Remark 4, the analytic solution is

\[
\phi(\xi, t) = (1 + \frac{1}{2} t^2 - t \sin t)(\xi - 1) + 1 + t \sin t.
\]

By using 11 equally spaced nodes on \([1, 2]\) and 11 equally distributed time steps from 0 to 1, we get the numerical solutions as follows:
<table>
<thead>
<tr>
<th>Nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytic solution</td>
<td>1.8415</td>
<td>1.9073</td>
<td>1.9732</td>
<td>2.0390</td>
<td>2.1049</td>
<td>2.1707</td>
</tr>
<tr>
<td>numerical solution</td>
<td>1.8415</td>
<td>1.9073</td>
<td>1.9731</td>
<td>2.0389</td>
<td>2.1047</td>
<td>2.1705</td>
</tr>
<tr>
<td>error</td>
<td>0</td>
<td>3.83e-5</td>
<td>7.66e-5</td>
<td>11.49e-5</td>
<td>15.32e-5</td>
<td>19.15e-5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nodes</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytic solution</td>
<td>2.2366</td>
<td>2.3024</td>
<td>2.3683</td>
<td>2.4341</td>
<td>2.5000</td>
</tr>
<tr>
<td>numerical solution</td>
<td>2.2364</td>
<td>2.3022</td>
<td>2.3680</td>
<td>2.4338</td>
<td>2.5000</td>
</tr>
<tr>
<td>error</td>
<td>22.98e-5</td>
<td>26.81e-5</td>
<td>30.63e-5</td>
<td>34.46e-5</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2.4. Comparison of example 3: $\phi(\xi, 1)$.

Figure 2.3. Illustration of example 3: blue stars for $\phi(\xi, 0)$, green circles for $\phi(\xi, 0.5)$, red dots for $\phi(\xi, 1)$. 

16
2.2.3.6 Example 4: moving domains

Let \( t \in [0, 1], \Omega_t = [0, 1 + t^2], f(x, t) = 1 - t + 2t\sqrt{x + \frac{1}{4}} \) and be normalized on \( \Omega_t \) by \( f(x, t) = f(x, t) \int_{\Omega_t} \frac{1}{f(x, t)} \, dx \). The analytic solution at \( t = 1 \) to the problem is

\[
\phi(\xi, 1) = \xi^2 + \xi.
\]

Similarly, the numerical solutions:

\[
\begin{array}{cccccc}
\text{Nodes} & 1 & 2 & 3 & 4 & 5 \\
\text{analytic solution} & 0 & 0.1100 & 0.2400 & 0.3900 & 0.5600 & 0.7500 \\
\text{numerical solution} & 0 & 0.1097 & 0.2395 & 0.3893 & 0.5591 & 0.7489 \\
\text{error} & 0 & 3.14e-4 & 5.30e-4 & 7.11e-4 & 8.95e-4 & 11.12e-4 \\
\end{array}
\]

\[
\begin{array}{cccccc}
\text{Nodes} & 7 & 8 & 9 & 10 & 11 \\
\text{analytic solution} & 0.9600 & 1.1900 & 1.4400 & 1.7100 & 2.0000 \\
\text{numerical solution} & 0.9586 & 1.1883 & 1.4379 & 1.7073 & 2.0000 \\
\text{error} & 13.82e-4 & 17.21e-4 & 21.41e-4 & 26.53e-4 & 0 \\
\end{array}
\]

Table 2.5. Comparison of example 4: \( \phi(\xi, 1) \).

(a) Green dots for \( \phi(\xi, 0.5) \), blue stars for \( \phi(\xi, 0) \), red dots for numerical solution, blue stars for analytic solution.

(b) \( \phi(\xi, 1) \), red dots for numerical solution, blue stars for analytic solution.

Figure 2.4. Illustration of example 4.
2.2.3.7 Conclusion

We can see clearly that all examples get satisfied results. And further studies by theoretical analysis and numerical examples indicate that the precision of evaluating $-\frac{\partial}{\partial t}(\frac{1}{f(x,t)})$ effects the numerical outcomes most. As mentioned in Remark 5, the deformation method (1D) is not our final object. We shall end the discussion in 1D case here, and move to general cases with the same basic ideas.

2.3 Two- and Three-dimensional case

2.3.1 Problem formulation

Now let $\Omega_t \subset \mathbb{R}^n (n = 2, 3)$ be moving domains, $v(x,t)$ be the velocity field on the boundary $\partial \Omega_t$. Given a scalar function $f(x,t) \in C^1(x,t) > 0$ on domain of $(x,t) : \Omega_t \times [0,1]$ such that

$$f(x,0) = 1,$$  \hspace{1cm} (2.7)

$$\int_{\Omega_t} \frac{1}{f(x,t)} dx = |\Omega_0|.$$  \hspace{1cm} (2.8)

A diffeomorphism

$$\phi(\xi, t) : \Omega_0 \rightarrow \Omega_t$$

such that $\forall t \geq 0$

$$J(\phi(\xi, t)) = f(\phi(\xi, t), t)$$  \hspace{1cm} (2.9)

can be constructed by solving the following 2 differential equations (2.10) and (2.11). (Note: Here $J(\phi(\xi, t)) = \det \nabla \phi(\xi, t)$ is the Jacobian determinant of $\phi(\xi, t)$.)
• First, determine $u(x, t)$ on $\mathbb{R}^n$ by solving the div-curl-system:

\[
\begin{align*}
\text{div } u(x, t) &= -\frac{\partial}{\partial t} \left( \frac{1}{f(x, t)} \right), \\
curl u(x, t) &= 0, \\
u(x, t) &= \frac{v(x, t)}{f(x, t)} \quad \text{on } \partial\Omega_t.
\end{align*}
\]  

(2.10)

• Second, determine $\phi(\xi, t)$ on $\Omega_0$ by solving:

\[
\begin{align*}
\frac{\partial \phi}{\partial t}(\xi, t) &= f(\phi(\xi, t), t)u(\phi(\xi, t), t), \\
\phi(\xi, 0) &= \xi.
\end{align*}
\]

(2.11)

We name this method the deformation method, and (2.10) the div-curl-system.

**Remark 6.** Comparing with the deformation method (1D), here in (2.10), we add one more curl equation. The purpose of adding this curl equation is just to make the div-curl-system have one unique solution $u$ [19]. It will not effect the property (2.9).

### 2.3.2 Theoretical derivation

Similarly, in order to claim the constructed $\phi(\xi, t)$ is the desired diffeomorphism, we shall prove the following statements:

S4. $\nabla_\xi \phi(\xi, t)$ exists and (2.9) is satisfied;

S5. $\frac{\partial \phi(\xi, t)}{\partial t} = v(\xi, t)$ on $\partial\Omega_0$, namely $\phi(\xi, t)$ does map $\Omega_0$ onto $\Omega_t$.

Before proving S4., we need to recall a theorem here [20].

**Theorem 2** (Liouville’s Theorem or Abel-Jacobi-Liouville Identity). If $A \in C(\mathbb{R}, \mathbb{R}^{n \times n})$ is a $n \times n$ matrix, and $X(t)$ is a matrix solution of $X'(t) = A(t)X(t)$, then

$$\det X(t) = \det X(t_0)e^{\int_{t_0}^t \text{Tr}A(s)ds},$$

furthermore,

$$\frac{d}{dt} \det X(t) = \text{Tr}A(t)\det X(t).$$
Proof of S4. The existence of $\nabla_\xi \phi(\xi, t)$ follows the same smooth dependence on initial conditions of ODE Theorem 1.

To prove (2.9), we define $H(\xi, t) = \frac{J(\phi(\xi, t))}{f(\phi(\xi, t), t)}$ on $\Omega_0 \times [0, 1]$, consequently $H(\xi, 0) = 1$.

Our goal here is still to prove $\frac{dH(\xi, t)}{dt} \equiv 0$, then $H(\xi, t) \equiv H(\xi, 0) = 1$.

Start with

$$\frac{dH(\xi, t)}{dt} = \frac{d}{dt} \frac{J(\phi(\xi, t))}{f(\phi(\xi, t), t)} = \frac{d}{dt} \frac{\det \nabla_\xi \phi(\xi, t)}{f(\phi(\xi, t), t)}$$

$$(*)$$

To be clear here, let $n = 3$ and

$$\phi = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} \quad u = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \quad \nabla_\phi f = \begin{pmatrix} f_{\phi_1}, f_{\phi_2}, f_{\phi_3} \end{pmatrix}_{3 \times 1}.$$

Consider $\frac{d}{dt} \nabla_\xi \phi(\xi, t),$

$$\frac{d}{dt} \nabla_\xi \phi(\xi, t) = \nabla_\xi \phi_1(\xi, t) = \nabla_\xi [f(\phi(\xi, t), t)u(\phi(\xi, t), t)]$$

$$= u(\phi(\xi, t), t) \cdot \nabla_\phi f(\phi(\xi, t), t) \cdot \nabla_\xi \phi(\xi, t) + f(\phi(\xi, t), t)\nabla_\phi u(\phi(\xi, t), t) \cdot \nabla_\xi \phi(\xi, t)$$

$$= [u(\phi(\xi, t), t) \cdot \nabla_\phi f(\phi(\xi, t), t) + f(\phi(\xi, t), t)\nabla_\phi u(\phi(\xi, t), t)] \cdot \nabla_\xi \phi(\xi, t).$$

By Theorem 2, we have

$$\frac{d}{dt} \det \nabla_\xi \phi(\xi, t)$$

$$= \text{Tr}[u(\phi(\xi, t), t) \cdot \nabla_\phi f(\phi(\xi, t), t) + f(\phi(\xi, t), t)\nabla_\phi u(\phi(\xi, t), t)]\det \nabla_\xi \phi(\xi, t)$$

$$= [\nabla_\phi f(\phi(\xi, t), t) \cdot u(\phi(\xi, t), t) + f(\phi(\xi, t), t)\text{div}_\phi u(\phi(\xi, t), t)]\det \nabla_\xi \phi(\xi, t).$$
Plug it back into (*),

\[
\frac{dH(\xi, t)}{dt} = \left[ \nabla \phi(\phi(\xi, t), t) \cdot u(\phi(\xi, t), t) + f(\phi(\xi, t), t) \right] \frac{1}{f(\phi(\xi, t), t)} \cdot \frac{1}{f(\phi(\xi, t), t)} \\
+ \left[ \frac{d}{dt} \frac{1}{f(\phi(\xi, t), t)} \right] \det \nabla \phi(\phi(\xi, t), t)
\]

\[
= \left[ \nabla \phi(\phi(\xi, t), t) \cdot u(\phi(\xi, t), t) + f(\phi(\xi, t), t) \right] \frac{1}{f^2(\phi(\xi, t), t)} \\
- \frac{\nabla \phi(\phi(\xi, t), t) \cdot \phi_t(\phi(\xi, t), t) + f_t(\phi(\xi, t), t)}{f^2(\phi(\xi, t), t)} \left[ \det \nabla \phi(\phi(\xi, t), t) \right]
\]

\[
= \left[ \nabla \phi(\phi(\xi, t), t) \cdot u(\phi(\xi, t), t) + f(\phi(\xi, t), t) \right] \frac{1}{f^2(\phi(\xi, t), t)} \\
- \frac{\nabla \phi(\phi(\xi, t), t) \cdot \phi(\phi(\xi, t), t) + f(\phi(\xi, t), t)}{f^2(\phi(\xi, t), t)} \left[ \det \nabla \phi(\phi(\xi, t), t) \right]
\]

\[
= \left[ f^2(\phi(\xi, t), t) \nabla \phi(\phi(\xi, t), t) \right] \frac{1}{f^2(\phi(\xi, t), t)} \\
= \left[ f^2(x, t) u(\phi(\phi(\xi, t), t), t) - f_t(x, \phi(\xi, t), t) \right] \frac{1}{f^2(\phi(\xi, t), t)} \\
= 0.
\]

Hence, \( H(\xi, t) \equiv 1 \Rightarrow J(\phi(\xi, t)) = f(\phi(\xi, t), t) \) for \( \forall t \).

**Proof of S5.** The proof is similar to the Proof of S3. Let \( \forall \xi_0 \in \partial \Omega_0 \) in (2.11), we get

\[
\begin{cases}
\frac{\partial \phi}{\partial t}(\xi_0, t) = f(\phi(\xi_0, t), t) u(\phi(\xi_0, t), t), \\
\phi(\xi_0, 0) = \xi_0.
\end{cases}
\]

From the boundary condition of (2.10), we have

\[
\begin{cases}
\frac{d\chi}{dt} = \nu(x, t) = f(x, t) u(x, t), \\
\chi(\xi_0, 0) = \xi_0.
\end{cases}
\]

By uniqueness of ODE initial value problem, we claim that \( \phi(\xi_0, t) = \chi(\xi_0, t) \) for \( \forall \xi_0 \in \partial \Omega_0 \). Namely, \( \phi(\xi, t) = \chi(\xi, t) \) on \( \partial \Omega_0 \). Moreover, \( \frac{\partial \phi(\xi, t)}{\partial t} = \nu(\xi, t) \) on \( \partial \Omega_0 \). \( \square \)
Remark 7. For the same reason in Remark 1, solving differential equation (2.10) on $\mathbb{R}^n$ doesn’t conflict with the condition that $f(x,t)$ is only defined on $\Omega_t$.

Remark 8. One special case in practice is $\Omega_t \equiv \Omega_0$, namely the fixed domain problem. In this special case, $f(x,t) > 0$ should be given on $\Omega_0 \times [0,1]$, such that

\[ f(x,0) = 1, \quad (2.7^*) \]
\[ \int_{\Omega_0} \frac{1}{f(x,t)}dx = |\Omega_0|. \quad (2.8^*) \]

After solving differential equations (2.10) and (2.11), we can get $\phi(\xi,t) : \Omega_0 \to \Omega_0$ which satisfies (2.9).

Remark 9. Another special case in practice is $f(x,t) = \frac{|\Omega_t|}{|\Omega_0|}$, we can get a uniform map $\phi(\xi,t) : \Omega_0 \to \Omega_t$.

Before ending the theoretical discussion, we want to make one more Remark.

Remark 10. Recall the deformation method, a diffeomorphism is constructed from the identity map, i.e. $\phi(\xi,0) = \text{id}(\xi)$. It works well both in theory and in practice. However, we could ask the question naturally: can we start the construction from any given map $\phi_0(\xi,0)$? The answer is of course yes, and the explanation is simple. Just consider any $t_0 > 0$, then what happens in $[t_0,1]$ is exactly what we discuss here. So we can re-address the deformation method one more time as:

Keep the same assumptions about $\Omega_t$ and $v(x,t)$. Given a map $\phi_0(\xi) : \Omega_0 \to \Omega_0$ and a scalar function $f(x,t) \in C^1(x,t) > 0$ on domain of $(x,t) : \Omega_t \times [0,1]$ such that

\[ f(\phi_0,0) = J(\phi_0), \quad (2.12) \]
\[ \int_{\Omega_t} \frac{1}{f(x,t)}dx = |\Omega_0|. \quad (2.13) \]

We construct a diffeomorphism

$\phi(\xi,t) : \Omega_0 \to \Omega_t$
such that $J(\phi(\xi, t)) = f(\phi(\xi, t), t)$, by solving the following 2 differential equations (2.14) and (2.15).

- First, determine $u(x, t)$ on $\mathbb{R}^n$ by solving the div-curl-system:

$$\begin{cases}
\text{div } u(x, t) = -\frac{\partial}{\partial t} \left( \frac{1}{f(x, t)} \right), \\
\text{curl } u(x, t) = 0, \\
u(x, t) = \frac{v(x, t)}{f(x, t)} \quad \text{on } \partial \Omega_t.
\end{cases} \tag{2.14}$$

- Second, determine $\phi(\xi, t)$ on $\Omega_0$ by solving:

$$\begin{cases}
\frac{\partial \phi}{\partial t} (\xi, t) = f(\phi(\xi, t), t)u(\phi(\xi, t), t), \\
\phi(\xi, 0) = \phi_0(\xi).
\end{cases} \tag{2.15}$$

A numerical example in the next section will use this Remark to show an interesting problem.

Now, we finish the theoretical derivation of the general deformation method. It is a natural extension of the deformation method (1D), a new div-curl-system (2.10) is introduced. Let’s move on to the numerical implementation part, where the LSFEM method [19] is used to solve the div-curl-system (2.10).

2.3.3 Numerical implementation and examples

2.3.3.1 Algorithm implementation

We can still implement the deformation method by the same 4-step algorithm:

- **Step 1: Initialize.** Starts from $t = 0$, $\phi(\xi, 0) = \xi$, $f(x, 0) = 1$.
- **Step 2: Div-curl system (2.10).** Compute $-\frac{\partial}{\partial t} \left( \frac{1}{f(x, t)} \right)$, then use LSFEM to solve (2.10), get $u(x, t)$ on $\Omega_t$.
- **Step 3: ODE (2.11).** Update $\phi(\xi, t)$ from $\Omega_t$ to $\Omega_{t+dt}$ by (2.11).
- **Step 4:** Next time step. Move to next time step \( t = t + dt \), back to Step 2 till \( t = 1 \).

2.3.3.2 Numerical Example 1: 2D fixed domain

Our first example considers a fixed domain \( \Omega_t \equiv [0, 1] \times [0, 1] \). Let \( f(x, y, t) = 1 - t^2 + t^2 \sqrt{2x + \frac{1}{4}} \), and be normalized on \( \Omega_t \) by
\[
 f(x, y, t) = f(x, y, t) \int_{\Omega_t} \frac{1}{f(x, y, t)} \, dx \, dy.
\]
The analytic solution at \( t = 1 \) to the problem is
\[
 \phi(\xi, \eta, 1) = \left( \frac{\xi^2 + \xi}{2}, \eta \right).
\]
The numerical solutions of \( \phi(\xi, \eta, 0) \) and \( \phi(\xi, \eta, 1) \) are showed in Figure 2.5. At \( t = 1 \),
\[
 \max(\| \phi_{\text{analytic}}(\xi, \eta, 1) - \phi_{\text{numerical}}(\xi, \eta, 1) \|) = 0.0018.
\]

Figure 2.5. Illustration of example 1, red dots represent numerical solution, blue stars represent analytic solution.

2.3.3.3 Example 2: 2D fixed domain

For a general \( f(x, t) \), we don’t have a analytic solution \( \phi_{\text{analytic}} \). But we can examine the difference between \( J(\phi) \) and \( f(\phi) \), which is expected to be 0.
Let $d = \sqrt{(x - 0.4)^2 + (y - 0.6)^2} - 0.1$, i.e. the distance to the circle $(x - 0.4)^2 + (y - 0.6)^2 = 0.1^2$. Define $f(x)$ based on the function $f(d)$ showed in Figure 2.6(a). $f(x, t) = 1 - \sin t + \sin tf(x)$ and normalized on a fixed domain $\Omega_t \equiv [0, 1] \times [0, 1]$. Figure 2.5(b) shows the constructed diffeomorphism $\phi(\xi, 1)$. Table 2.6 shows the $L^2$-norm between $J(\phi)$ and $f(\phi)$ at each time step $t$ from 0 to 1, with $dt = 0.1$ and $dt = 0.02$. We can clearly check the accuracy.

![Figure 2.6](image-url)
\[ \| J(\phi) - f(\phi) \|_2, \ dt = 0.1 \]  \quad \| J(\phi) - f(\phi) \|_2 \ dt = 0.02 \\
0.1 \quad 1.6827e-005 \quad 1.0807e-005 \\
0.2 \quad 1.0734e-004 \quad 4.2828e-005 \\
0.3 \quad 3.2271e-004 \quad 9.5412e-005 \\
0.4 \quad 6.5313e-004 \quad 1.7480e-004 \\
0.5 \quad 0.0011 \quad 2.7650e-004 \\
0.6 \quad 0.0017 \quad 3.9221e-004 \\
0.7 \quad 0.0024 \quad 4.6594e-004 \\
0.8 \quad 0.0034 \quad 5.9361e-004 \\
0.9 \quad 0.0051 \quad 8.8787e-004 \\
1.0 \quad 0.0066 \quad 0.0011 \\

Table 2.6. \( L^2 \)-norm of the difference \( J(\phi) \) and \( f(\phi) \) at each time step \( t \)

Note: Professor S.Turek’s group studied the precision problem with the Poisson based deformation method\[21\], they also applied the deformation method to simulation of multiple falling balls in water\[22\].

2.3.3.4 Example 3: Back to identity map

The third example deals with an interesting problem mentioned in Remark 10. We want to construct diffeomorphisms from some \( \phi_0 \) back to the identity map \( \text{id}(\xi) \). Given a known diffeomorphism \( \phi_0 \), numerically we can find \( f(x) \) such that \( f(\phi_0) = J(\phi_0) \). Define \( f(x, t) = t + (1-t)f(x) \) and normalized on \( \Omega_t \), then in theory, apply the deformation method, we shall expect \( \phi(\xi, 1) = \xi \).

Take the constructed \( \phi(\xi, 1) \) in the previous example 2 as \( \phi_0 \) here. The results are shown in Figure 2.7, \( \max \| \phi(\xi, 1) - \xi \| = 0.0150 \).
2.3.3.5 Example 4: 2D moving domain

In this example, $\Omega_0$ is set to be a unit square, $\Omega_1$ be a unit circle, and $\Omega_t$ be a series of intermediate domains determined by the boundary nodes. Let $d_1 =$
\[ \sqrt{(x - \frac{1}{2})^2 + (y - \frac{1}{4})^2}, d_2 = \sqrt{(x - \frac{1}{2})^2 + (y - \frac{3}{4})^2}, d_3 = | x - \frac{1}{2} - \frac{1}{4}\sin 2\pi y |. \] Define \( f(x, y) \) as:

\[
f(x, y) = \begin{cases} 
1 - C_1e^{-C_2d_{\min(d_1, d_2)}} & d_1 \leq p \text{ or } d_2 \leq p, \\
1 - C_1e^{-C_2d_3} & \text{else},
\end{cases}
\]

\( C_1, C_2, p \) are parameters. \( f(x, y, t) = 1 - t + tf(x, y) \) and normalized on \( \Omega_t \).

Figure 2.8. Illustration of example 4, here \( C_1 = 0.8, C_2 = 5, p = 0.03 \).
CHAPTER 3
NEW VARIATIONAL METHOD WITH PRESCRIBED JACOBIAN DETERMINANT AND CURL VECTOR

3.1 Introduction

In the deformation method, we first determine a velocity field $u$, then use this velocity field $u$ to construct $\phi$. The ultimate criterion is actually the Jacobian determinant of $\phi$. So we could ask, since what we want is the Jacobian determinant of a diffeomorphism $\phi$, can we just control it directly, without the help of velocity field $u$? We explored this idea of controlling the Jacobian determinant directly, and developed a new approach based on an optimization scheme. The core procedure is to define a cost functional, derive the variational gradient, apply gradient descend method to minimize the cost functional, and construct $\phi$ iteratively. Because the highlight of this approach is the derivation of the variational gradient, we named it new variational method with prescribed Jacobian determinant.

Later, inspired from the div-curl system, we learnt it is for sure that with the same Jacobian determinant, different curl vectors can lead different diffeomorphisms [23]. This fact reminds us that the curl vector of $\phi$ is also needed in the cost functional. Therefore, we extended the new approach to new variational method with prescribed Jacobian determinant and curl vector.

In this chapter, different versions of the new variational method are proposed. Theoretical derivations of the variational gradient are discussed in detail. Also, numerical algorithms and examples are presented to show the satisfactory results in the numerical construction of diffeomorphisms.
More over, in the next chapter, we will use this method to study a potential uniqueness problem in differential geometry.

First, let’s look at the *new variational method with prescribed Jacobian determinant*.

### 3.2 New Variational Method: Version 1

#### 3.2.1 Problem formulation

Given a domain $\Omega \subset \mathbb{R}^n$, $n = 1, 2, 3$, and a scalar function $f_0(x) > 0$ defined on $\Omega$ with

$$
\int_{\Omega} f_0(x) dx = \mid \Omega \mid .
$$

(3.1)

Define a cost functional $ssd$ to be:

$$
ssd = \frac{1}{2} \int_{\Omega} (J(\phi(x)) - f_0(x))^2 dx.
$$

(3.2)

A transformation $\phi(x) : \Omega \to \Omega$ such that $J(\phi(x)) = f_0(x)$ can be numerically constructed by minimizing $ssd$ with respect to a control function $f(x)$ under the following 2 constraints (3.3) and (3.4).

$$
\phi(x) = \phi_{old}(x) + u(x) dt,
$$

(3.3)

where $dt$ is an artificial time step, and $u(x)$ is a velocity vector satisfying:

$$
\begin{aligned}
\text{div } u &= f \quad \text{in } \Omega, \\
\text{curl } u &= 0 \quad \text{in } \Omega, \\
u &= 0 \quad \text{on } \partial \Omega.
\end{aligned}
$$

(3.4)

We name this method *new variational method with prescribed Jacobian determinant*, and call $f_0(x)$ the prescribed Jacobian determinant monitor function.

#### 3.2.2 Theoretical derivation

In order to apply the gradient descend method to minimize $ssd$, we shall derive the variational derivative $\frac{\partial ssd}{\partial f}$ by variational calculus. Without loss of generality, we
show the theoretical derivation in two-dimensional case.

Now \( \phi = (\phi_1, \phi_2), \ u = (u_1, u_2) \). Notice that (3.4) leads to the poisson’s equation:

\[
\begin{align*}
\Delta u_1 &= f_x, \\
\Delta u_2 &= f_y.
\end{align*}
\] (3.5)

Let \( \delta f \) be any given variation of \( f \) which vanishes on the boundary, we have:

\[
\delta ssd = \int (J(\phi(x)) - f_0(x))\delta J(\phi(x))dx
= \int (J(\phi) - f_0)\delta J(\phi)dx
= \int (J(\phi) - f_0)\delta(\phi_{1x}\phi_{2y} - \phi_{2x}\phi_{1y})dx
= \int (J(\phi) - f_0)(\delta u_{1x}dt\phi_{2y} + \phi_{1x}\delta u_{2y}dt - \delta u_{2x}dt\phi_{1y} - \phi_{2x}\delta u_{1y}dt)dx.
\]

The last line above follows \( \delta \phi = \delta u dt \) from (3.3).

Let \( P = (J(\phi) - f_0)dt \), we can continue as

\[
\delta ssd = \int P(\delta u_{1x}\phi_{2y} + \phi_{1x}\delta u_{2y} - \delta u_{2x}\phi_{1y} - \phi_{2x}\delta u_{1y})dx
= \int P[(\phi_{2y}, -\phi_{2x}) \cdot \nabla \delta u_1 + (-\phi_{1y}, \phi_{1x}) \cdot \nabla \delta u_2]dx.
\]

Define \( a_1 = -P(\phi_{2y}, -\phi_{2x}), a_2 = -P(-\phi_{1y}, \phi_{1x}) \), and introduce \( g_1, g_2 \) satisfying

\[
\begin{align*}
\Delta g_1 &= \nabla \cdot a_1, \\
\Delta g_2 &= \nabla \cdot a_2.
\end{align*}
\] (3.6)

Consequently,

\[
\delta ssd = \int (-a_1 \cdot \nabla \delta u_1 - a_2 \cdot \nabla \delta u_2)dx.
\]

Before continuing, we need to recall two Corollaries of divergence theorem here.

**Corollary 1.** By Applying the divergence theorem to the product of a scalar function \( v \) and a vector field \( u \) on domain \( \Omega \), we have

\[
\int_{\Omega} (u \cdot \nabla v + v (\nabla \cdot u))dV = \int_{\partial \Omega} (v u \cdot n)ds,
\]

where \( n \) is the outward unit normal vector.
**Corollary 2.** By Applying the divergence theorem to the product of a scalar function \(v\) and a vector field \(\nabla w\) on domain \(\Omega\), we have
\[
\int_{\Omega} (\nabla w \cdot \nabla v + v(\Delta w))dV = \int_{\partial \Omega} (v \nabla w \cdot \mathbf{n})dS,
\]
where \(\mathbf{n}\) is the outward unit normal vector.

Apply Corollary 1 (\(\star\)) and Corollary 2 (\(\star\star\)):

\[
\delta ssd = \int (-\mathbf{a}_1 \cdot \nabla \delta u_1 - \mathbf{a}_2 \cdot \nabla \delta u_2)d\mathbf{x}
\]
\[
= \int (\nabla \cdot \mathbf{a}_1 \delta u_1 + \nabla \cdot \mathbf{a}_2 \delta u_2)d\mathbf{x} \quad (\star)
\]
\[
= \int (\Delta g_1 \delta u_1 + \Delta g_2 \delta u_2)d\mathbf{x}
\]
\[
= \int (g_1 \delta \Delta u_1 + g_2 \delta \Delta u_2)d\mathbf{x} \quad (\star\star)
\]
\[
= \int (g_1 \delta f_x + g_2 \delta f_y)d\mathbf{x}
\]
\[
= \int ((g_1, g_2) \cdot \nabla \delta f)d\mathbf{x}
\]
\[
= \int (-\nabla \cdot (g_1, g_2))\delta f d\mathbf{x}.
\]

Finally, we obtain the variational derivative
\[
\frac{\partial ssd}{\partial f} = -\nabla \cdot (g_1, g_2) = -(g_{1x} + g_{2y}).
\]

3.2.3 Algorithm implementation

The gradient descend optimization algorithm can be briefly described as follows:

1. **Initialize** \(\phi_{old} = \mathbf{id}, f = 0\).
2. **Compute** \(P, \mathbf{a}_1, \mathbf{a}_2, \nabla \cdot \mathbf{a}_1, \nabla \cdot \mathbf{a}_2\).
3. **Solve** Poisson’s equations (3.6) to get \(g_1, g_2\), and then \(\frac{\delta ssd}{\partial f}\).
4. **Update** $f$ by $f_{\text{new}} = f_{\text{old}} - \delta_{\text{ssd}} \frac{\delta_{\text{ssd}}}{\delta f} \times t\text{step}dt$, where $t\text{step}$ is an optimization parameter.

5. **Solve** Poisson’s equations (3.5) to get $u_1, u_2$.

6. **Update** $\phi$ by (3.3).

7. **Back** to 2, keep iterating until a preset tolerance or a preset number of iteration steps is reached.

3.3 Development of New Variational Method: Version 2

3.3.1 Discussion and improvements

Later, we made some improvements as follows:

1. The Jacobian determinant of a transformation alone usually cannot determine the transformation itself uniquely. Namely, the nonlinear equation $\det \nabla \phi = f_0$ is not guaranteed to have an unique solution. Inspired by the div-curl-system in the deformation method, we added the curl vector into the cost functional, therefore extended the similarity measure (3.2) to

\[
\text{ssd} = \frac{1}{2} \int_{\Omega} [(J(\phi(x)) - f_0(x))^2 + \alpha (\text{curl}(\phi(x)) - g_0(x))^2] dx. \tag{3.7}
\]

Here $\alpha \geq 0$ is a weight parameter, $f_0$ is the prescribed Jacobian determinant monitor function, and $g_0$ is the prescribed curl vector monitor function.

So the problem we study now is to construct numerically a transformation $\phi : \Omega \rightarrow \Omega$, such that, the Jacobian determinant of $\phi$, i.e. $J(\phi)$, equals a prescribed monitor function $f_0$, and the curl of $\phi$, i.e. $\text{curl}(\phi)$, equals a prescribed monitor function $g_0$.

2. Removed the artificial time step in (3.3), directly let

\[
\phi(x) = x + u(x). \tag{3.8}
\]
3. The constraint (3.4) were generalized to
\[
\begin{align*}
\text{div } u &= f \quad \text{in } \Omega, \\
\text{curl } u &= g \quad \text{in } \Omega, \\
\quad u &= 0 \quad \text{on } \partial \Omega.
\end{align*}
\] (3.9)

Which leads to (in 2D)
\[
\begin{align*}
\Delta u_1 &= f_x - g_y := f_1, \\
\Delta u_2 &= f_y + g_x := f_2.
\end{align*}
\] (3.10)

We then used \( f_1 \) and \( f_2 \) as the control functions and directly set:
\[
\Delta u = f = (f_1, f_2)
\] (3.11)

with fixed boundary conditions.

Combining (3.7), (3.8) and (3.11), we name the new version new variational method with prescribed Jacobian determinant and curl vector. Next, let’s derive the variational gradient for the new set-up.

3.3.2 Derivation of the gradient: 2D

Let \( \phi = (\phi_1, \phi_2), u = (u_1, u_2), f = (f_1, f_2) \), and take \( \alpha = 1 \) for the simplicity of presentation (See Remark 11 at the end of this section for the general cases). The new variational method: version 2 is to minimize
\[
\text{ssd} = \frac{1}{2} \int_{\Omega} [(J(\phi) - f_0)^2 + (\text{curl}(\phi) - g_0)^2]dA
\]
subject to the constraints that the displacement \( u \) satisfies:
\[
\begin{align*}
\Delta u_1 &= f_1, \\
\Delta u_2 &= f_2 \quad \text{in } \Omega, \\
\quad u_1, u_2 &= 0 \quad \text{on } \partial \Omega.
\end{align*}
\] (3.12)
For arbitrary \( \delta f_1 \) and \( \delta f_2 \) that vanish on \( \partial \Omega \),
\[
\delta_{ssd} = \oint_{\Omega} (J(\phi) - f_0) \delta_{ssd} + (\text{curl}(\phi) - g_0) \delta_{ssd} dA
\]
\[
= \oint_{\Omega} [(J(\phi) - f_0) \delta(\phi_{1x}\phi_{2y} - \phi_{1y}\phi_{2x}) + (\text{curl}(\phi) - g_0) \delta(\phi_{1x} - \phi_{1y})] dA
\]
\[
= \oint_{\Omega} [(J(\phi) - f_0) (\delta\phi_{1x}\phi_{2y} + \phi_{1x}\delta\phi_{2y} - \delta\phi_{1y}\phi_{2x} - \phi_{1y}\delta\phi_{2x})
\]
\[
+ (\text{curl}(\phi) - g_0)(\delta\phi_{2x} - \delta\phi_{1y})] dA.
\]

From (3.8), we know \( \delta \phi = \delta \mathbf{u} \), so \( \delta\phi_{ix} = \delta u_{ix}, \delta\phi_{iy} = \delta u_{iy}, i = 1, 2 \). Plug it back in the above equation, we get
\[
\delta_{ssd} = \oint_{\Omega} [(J(\phi) - f_0) (\delta u_{1x}\phi_{2y} + \phi_{1x}\delta u_{2y} - \delta u_{1y}\phi_{2x} - \phi_{1y}\delta u_{2x})
\]
\[
+ (\text{curl}(\phi) - g_0)(\delta u_{2x} - \delta u_{1y})] dA.
\]

Let \( P = (J(\phi) - f_0), Q = (\text{curl}(\phi) - g_0) \), we have
\[
\delta_{ssd} = \oint_{\Omega} [P(\delta u_{1x}\phi_{2y} + \phi_{1x}\delta u_{2y} - \delta u_{1y}\phi_{2x} - \phi_{1y}\delta u_{2x}) + Q(\delta u_{2x} - \delta u_{1y})] dA.
\]

Notice that \( \nabla \delta u_i = (\delta u_{ix}, \delta u_{iy}), i = 1, 2 \), so
\[
\delta_{ssd} = \oint_{\Omega} [P((\phi_{2y}, -\phi_{2x}) \cdot \nabla \delta u_1 + (-\phi_{1y}, \phi_{1x}) \cdot \nabla \delta u_2)
\]
\[
+ Q((0, -1) \cdot \nabla \delta u_1 + (1, 0) \cdot \nabla \delta u_2)] dA.
\]

Define
\[
\begin{cases}
-\mathbf{a}_1 = P(\phi_{2y}, -\phi_{2x}) + Q(0, -1), \\
-\mathbf{a}_2 = P(-\phi_{1y}, \phi_{1x}) + Q(1, 0).
\end{cases}
\] (3.13)

We now get
\[
\delta_{ssd} = \oint_{\Omega} (-\mathbf{a}_1 \cdot \nabla \delta u_1 - \mathbf{a}_2 \cdot \nabla \delta u_2) dA.
\]

Recall that \( u_i = 0 \) on \( \partial \Omega \), so \( \delta u_i = 0 \) on \( \partial \Omega \). By Corollary 1,
\[
\delta_{ssd} = \oint_{\Omega} (-\mathbf{a}_1 \cdot \nabla \delta u_1 - \mathbf{a}_2 \cdot \nabla \delta u_2) dA
\]
\[
= \oint_{\Omega} ((\nabla \cdot \mathbf{a}_1) \delta u_1 + (\nabla \cdot \mathbf{a}_2) \delta u_2) dA - \int_{\partial \Omega} (\delta u_1 (\mathbf{a}_1 \cdot \mathbf{n}) + \delta u_2 (\mathbf{a}_2 \cdot \mathbf{n})) dS
\]
\[
= \oint_{\Omega} ((\nabla \cdot \mathbf{a}_1) \delta u_1 + (\nabla \cdot \mathbf{a}_2) \delta u_2) dA.
\]
Now define \( g = (g_1, g_2) \) as
\[
\begin{align*}
\Delta g_1 &= \nabla \cdot a_1, \\
\Delta g_2 &= \nabla \cdot a_2, \\
g_1, g_2 &= 0 \quad \text{on } \partial \Omega.
\end{align*}
\] (3.14)

Then,
\[
\delta_{ssd} = \iint_\Omega (\Delta g_1 \delta u_1 + \Delta g_2 \delta u_2) dA.
\]

By the Corollary 2, for \( i = 1, 2 \), we get
\[
\begin{align*}
\iint_\Omega (\nabla g_i \cdot \nabla \delta u_i + \delta u_i \Delta g_i) dA &= \int_{\partial \Omega} (\delta u_i \nabla g_i \cdot n) dS = 0, \\
\iint_\Omega (\nabla g_i \cdot \nabla \delta u_i + g_i \Delta \delta u_i) dA &= \int_{\partial \Omega} (g_i \nabla \delta u_i \cdot n) dS = 0.
\end{align*}
\]

Finally,
\[
\delta_{ssd} = \iint_\Omega (\Delta g_1 \delta u_1 + \Delta g_2 \delta u_2) dA \\
= \iint_\Omega (g_1 \delta (\Delta u_1) + g_2 \delta (\Delta u_2)) dA \\
= \iint_\Omega (g_1 \delta f_1 + g_2 \delta f_2) dA,
\]
which is
\[
\frac{\partial ssd}{\partial f_i} = g_i, \quad i = 1, 2. \quad (3.15)
\]

**Remark 11.** At the end of derivation, we want to point out that if we change the identity map \( x \) to any given transformation \( \phi^*(x) \) in (3.8), the derivation above is still right. Generally speaking, (3.8) can be replaced by
\[
\phi(x) = \phi^*(x) + u(x), \quad (3.8^*)
\]
where \( \phi^*(x) \) is any given transformation. This means the construction can start with any given transformation, the identity transformation is just a special case.

**Remark 12.** For arbitrary weight parameter \( \alpha \) in (3.7), the derivation above is still right just by changing \( Q = \alpha (\text{curl}(\phi) - g_0) \).
Remark 13. The fixed boundary conditions in (3.12) and (3.14) ($u = 0$ and $g = 0$ on $\partial \Omega$) are actually needed by the 2 Corollaries to move the gradient operator ($\nabla \cdot$) or the laplace operator ($\Delta$). The fixed boundary conditions are trivial ones that can make the 2 Corollaries work. It is clear that the Neumann boundary conditions $\nabla u_i \cdot n = 0$ and $\nabla g_i \cdot n = 0$ can make Corollary 2 work, but may fail Corollary 1. Generally speaking, in theory, it seems the only correct boundary condition is the fixed boundary condition. However, in numerical practice, Neumann boundary conditions can also lead satisfied results. Related discussion continues at the end of this chapter.

3.3.3 Derivation of the gradient: 3D

The general derivation is similar for three-dimensional case. The only major difference is the component of $a_i$ in (3.13).

Let $\phi = (\phi_1, \phi_2, \phi_3)$, $u = (u_1, u_2, u_3)$, $f = (f_1, f_2, f_3)$, $g_0 = (g_{01}, g_{02}, g_{03})$, and take $\alpha = 1$.

The cost functional is

$$ ssd = \frac{1}{2} \iiint_{\Omega} [(J(\phi) - f_0)^2 + (\text{curl}(\phi) - g_0)^2] dV, $$

subject to

$$ \begin{cases} 
\Delta u_1 &= f_1, \\
\Delta u_2 &= f_2, \\
\Delta u_3 &= f_3 & \text{in } \Omega, \\
u_1, u_2, u_3 &= 0 & \text{on } \partial \Omega. 
\end{cases} $$

(3.16)

Now,

$$ J(\phi) = \phi_{1x}(\phi_{2y}\phi_{3z} - \phi_{3y}\phi_{2z}) - \phi_{2x}(\phi_{1y}\phi_{3z} - \phi_{3y}\phi_{1z}) + \phi_{3x}(\phi_{1y}\phi_{2z} - \phi_{2y}\phi_{1z}), $$

and

$$ \text{curl}(\phi) = (\phi_{3y} - \phi_{2z}, \phi_{1z} - \phi_{3x}, \phi_{2x} - \phi_{1y}). $$
Let \( P = (J(\phi) - f_0), Q_1 = (\phi_{3y} - \phi_{2z} - g_{01}), Q_2 = (\phi_{1z} - \phi_{3x} - g_{02}), Q_3 = (\phi_{2x} - \phi_{1y} - g_{03}). \)

For arbitrary \( \delta f_i (i = 1, 2, 3) \) that vanish on \( \partial \Omega \), we have

\[
\delta \text{ssd} = \int_{\Omega} (J(\phi) - f_0) \delta J(\phi) + (\text{curl}(\phi) - g_0) \cdot \delta \text{curl}(\phi) \, dV
\]

\[
= \int_{\Omega} [P \delta (\phi_{1x} \phi_{2y} \phi_{3z} - \phi_{2y} \phi_{3z}) - \phi_{2x} (\phi_{1y} \phi_{3z} - \phi_{3y} \phi_{1z}) + \phi_{3z} (\phi_{1y} \phi_{2z} - \phi_{2y} \phi_{1z})] + (Q_1, Q_2, Q_3) \cdot \delta (\phi_{3y} - \phi_{2z}, \phi_{1z} - \phi_{3x}, \phi_{2x} - \phi_{1y})] \, dV
\]

\[
= \int_{\Omega} [P \delta u_{1x} \phi_{2y} \phi_{3z} + \delta u_{2y} \phi_{1x} \phi_{3z} + \delta u_{3z} \phi_{1x} \phi_{2y} - \delta u_{1x} \phi_{3y} \phi_{2z} - \delta u_{2y} \phi_{1x} \phi_{3y} - \delta u_{3y} \phi_{1x} \phi_{2z} - \delta u_{1y} \phi_{2z} \phi_{3y} - \delta u_{1z} \phi_{2x} \phi_{3y} + \delta u_{1z} \phi_{2y} \phi_{3z} - \delta u_{2x} \phi_{2y} \phi_{3z} + \delta u_{2x} \phi_{3z} \phi_{1y} + \delta u_{2z} \phi_{1y} \phi_{3z} - \delta u_{1z} \phi_{2x} \phi_{3y} - \delta u_{2x} \phi_{2z} \phi_{3y} - \delta u_{2x} \phi_{3x} \phi_{1z} - \delta u_{3x} \phi_{2y} \phi_{1z} + Q_1 (\delta u_{3y} - \delta \phi_{2z}) + Q_2 (\delta u_{1z} - \delta \phi_{3x}) + Q_3 (\delta u_{2x} - \delta \phi_{1y})] \, dV
\]

\[
= \int_{\Omega} [P \delta u_{1x}, \delta u_{1y}, \delta u_{1z}] \cdot (\phi_{2y} \phi_{3z} - \phi_{3y} \phi_{2z}, \phi_{3x} \phi_{2z} - \phi_{2x} \phi_{3z}, \phi_{2x} \phi_{3y} - \phi_{2y} \phi_{3x}) + P(\delta u_{2x}, \delta u_{2y}, \delta u_{2z}) \cdot (\phi_{3y} \phi_{1z} - \phi_{1y} \phi_{3z}, \phi_{1x} \phi_{3z} - \phi_{3x} \phi_{1z}, \phi_{3x} \phi_{1y} - \phi_{1z} \phi_{3y}) + P(\delta u_{3x}, \delta u_{3y}, \delta u_{3z}) \cdot (\phi_{1y} \phi_{2z} - \phi_{2y} \phi_{1z}, \phi_{2x} \phi_{1z} - \phi_{1x} \phi_{2z}, \phi_{1x} \phi_{2y} - \phi_{2x} \phi_{1y}) + (\delta u_{1x}, \delta u_{1y}, \delta u_{1z}) \cdot (0, -Q_3, Q_2) + (\delta u_{2x}, \delta u_{2y}, \delta u_{2z}) \cdot (Q_3, 0, -Q_1) + (\delta u_{3x}, \delta u_{3y}, \delta u_{3z}) \cdot (-Q_2, Q_1, 0)] \, dV.
\]

Define

\[
\begin{aligned}
-a_1 &= P(\phi_{2y} \phi_{3z} - \phi_{3y} \phi_{2z}, \phi_{3x} \phi_{2z} - \phi_{2x} \phi_{3z}, \phi_{2x} \phi_{3y} - \phi_{2y} \phi_{3x}) \\
&\quad + (0, -Q_3, Q_2), \\
-a_2 &= P(\phi_{3y} \phi_{1z} - \phi_{1y} \phi_{3z}, \phi_{1x} \phi_{3z} - \phi_{3x} \phi_{1z}, \phi_{3x} \phi_{1y} - \phi_{1z} \phi_{3y}) \\
&\quad + (Q_3, 0, -Q_1), \\
-a_3 &= P(\phi_{1y} \phi_{2z} - \phi_{2y} \phi_{1z}, \phi_{2x} \phi_{1z} - \phi_{1x} \phi_{2z}, \phi_{1x} \phi_{2y} - \phi_{2x} \phi_{1y}) \\
&\quad + (-Q_2, Q_1, 0),
\end{aligned}
\]

and \( g = (g_1, g_2, g_3) \) satisfies

\[
\begin{aligned}
\Delta g_1 &= \nabla \cdot a_1, \\
\Delta g_2 &= \nabla \cdot a_2, \\
\Delta g_3 &= \nabla \cdot a_3, \\
g_1, g_2, g_3 &= 0 \quad \text{on} \ \partial \Omega.
\end{aligned}
\]
We can get

\[
\delta ssd = \int_\Omega (-a_1 \cdot \nabla \delta u_1 - a_2 \cdot \nabla \delta u_2 - a_3 \cdot \nabla \delta u_3)dV
= \ldots
= \int_\Omega (g_1 \delta f_1 + g_2 \delta f_2 + g_3 \delta f_3)dV.
\]

Exactly,

\[
\frac{\partial ssd}{\partial f_i} = g_i, i = 1, 2, 3. \tag{3.19}
\]

3.3.4 Algorithm implementation

We can implement this new version by similar gradient descent optimization scheme as follows:

1. **Initialize** \( \phi = id, u = 0, f = 0 \).
2. **Compute** \( a_i \) by (3.13 or 3.17), then **solve** Poisson’s equation (3.14 or 3.18) to get \( g_i \).
3. **Update** \( f \) by \( f_{i, new} = f_{i, old} - g_i \times tstep \), where \( tstep \) is an optimization parameter.
4. **Solve** Poisson’s equation (3.12 or 3.16) to get \( u_i \).
5. **Update** \( \phi \) by (3.8).
6. **Back** to 2, keep iterating until a preset tolerance or a preset number of iteration steps is reached.

3.4 Development of New Variational Method: Version 3

The last version considers \( f_0(\phi(x)) \) and \( g_0(\phi(x)) \) instead of \( f_0(x) \) and \( g_0(x) \) in the cost functional. Namely define \( ssd \) as:

\[
ssd = \frac{1}{2} \int_\Omega [(J(\phi) - f_0(\phi))^2 + \alpha(\text{curl}(\phi) - g_0(\phi))^2]d\mathbf{x}. \tag{3.20}
\]

39
In 2D case, keep the same constraints (3.12) and definition (3.13), and let \( P = (J(\phi) - f_0(\phi)), Q = (\text{curl}(\phi) - g_0(\phi)) \) correspondingly. We can make the similar derivation:

\[
\delta ssd = \int_{\Omega} [P(\delta J(\phi) - \delta f_0(\phi)) + Q(\delta \text{curl}(\phi) - \delta g_0(\phi))] \, dx
\]

\[
= \int_{\Omega} [P\delta J(\phi) + Q\delta \text{curl}(\phi) - P\delta f_0(\phi) - Q\delta g_0(\phi)] \, dx
\]

\[
= \int_{\Omega} [(\nabla \cdot a_1)\delta u_1 + (\nabla \cdot a_2)\delta u_2 - P\delta f_0(\phi) - Q\delta g_0(\phi)] \, dx
\]

\[
= \int_{\Omega} [(\nabla \cdot a_1)\delta u_1 + (\nabla \cdot a_2)\delta u_2

\]

\[
- P\nabla f_0(\phi) \cdot (\delta u_1, \delta u_2) - Q\nabla g_0(\phi) \cdot (\delta u_1, \delta u_2)] \, dx
\]

\[
= \int_{\Omega} [(\nabla \cdot a_1 - Pf_0^x(\phi) - Qg_0^x(\phi))\delta u_1 + (\nabla \cdot a_2 - Pf_0^y(\phi) - Qg_0^y(\phi))\delta u_2] \, dx.
\]

Now define \( g = (g_1, g_2) \) by

\[
\begin{cases}
\Delta g_1 = \nabla \cdot a_1 - Pf_0^x(\phi) - Qg_0^x(\phi), \\
\Delta g_2 = \nabla \cdot a_2 - Pf_0^y(\phi) - Qg_0^y(\phi), \\
g_1, g_2 = 0 \quad \text{on } \partial \Omega.
\end{cases}
\]

Again,

\[
\frac{\partial ssd}{\partial f_i} = g_i, \quad i = 1, 2.
\]

3.5 Numerical examples

Several numerical examples of constructing transformations are illustrated in this section. The first 4 examples in 2D take \( g_0 = 0, \alpha = 1 \) and corresponding prescribed Jacobian determinant monitor functions \( f_0 \).
Figure 3.1. Illustration of some 2D numerical transformations constructed by the new variational method with prescribed Jacobian determinant and prescribed curl vector.

The next examples illustrate two 3D transformations, iso-surfaces of the Jacobian determinant (i.e. the volume size) are used to show the interior structure of the 3D transformations. In both examples, set $\alpha = 0.01$, $\mathbf{g}_0 = \mathbf{0}$. Figure 3.2 shows a trans-
formation which concentrates toward a annulus, Figure 3.3 shows a transformation which moves toward a sphere’s surface.

Figure 3.2. 3D transformation concentrates toward a annulus constructed by the new variational method with prescribed Jacobian determinant and prescribed curl vector.
Figure 3.3. 3D transformation moves toward a sphere’s surface constructed by the new variational method with prescribed Jacobian determinant and prescribed curl vector.

All examples here are with fixed boundary conditions, as discussed in Remark 13. But it doesn’t mean this method can not deal with moving boundary problems, we have at least the following 3 approaches:

1. **Boundary match.** By Remark 11, we can use a transformation $\phi^*(x)$ to move the boundary first, then it will be a fixed boundary problem.

2. **Domain padding.** We can pad the domain $\Omega$ a little bit to $\Omega^*$, do a fixed boundary problem on $\Omega^*$, then map back to $\Omega$.

3. **The deformation method.** We can combine with the deformation method to deal with the boundary or even domain issues.
3.6 Conclusion

The above examples show the construction of transformations by the new variational method with prescribed Jacobian determinant and prescribed curl vector is successful in practice. However, we need to admit that, unlike the deformation method, the new variational method is a numerical method other than a theoretical solution. We can not claim that the constructed transformation $\phi$ is definitely a diffeomorphism. Although, $J(\phi) > 0$ and fixed boundary condition does imply that $\phi$ is a diffeomorphism in theory [24, 25]. We also leave out the convergence analysis of the gradient descend optimization scheme.

At this point, the two methods of constructing diffeomorphisms/transformations have been presented in detail. At last, we want to emphasize the importance of the function $f(x,t)$ in the deformation method and $f_0(x)$ in the variational method. The quality of the transformation constructed highly depends on the quality of these 2 functions. Besides their normalization conditions (2.7,2.8) and (3.1), more importantly, they have to be reasonable. Namely, there should exist a diffeomorphism $\phi$ such that $J(\phi) = f(\phi, t)$, or a transformation $\phi$ such that $J(\phi) = f$ before we try to construct it. In other words, if given a $f$, but there is no such a transformation such that $J(\phi) = f$, of course we can not construct a transformation successfully. Again, how to define a function $f(x,t)$ or $f_0(x)$ properly is the key issue in the applications of the 2 methods to grid generation problems as well as other potential areas.

3.7 Graphical User Interface: GridPanel

We also designed a graphical user interface named “GridPanel” to demonstrate our two methods. It can generate a grid based on the lines you draw automatically and instantaneously. The next figures show some works by GridPanel.
Figure 3.4. GridPanel example.

Figure 3.5. GridPanel example.
CHAPTER 4

STUDY ON THE UNIQUENESS OF TRANSFORMATION WITH JACOBIAN DETERMINANT AND CURL VECTOR

4.1 Introduction

In the study of the two transformation construction methods, we deeply learnt the importance of the curl vector $\text{curl} \phi$ in the form of a transformation. A natural question is how important it is? In the theory of differential geometry, we already know that the divergence $\nabla \phi$ and the curl vector $\nabla \times \phi$ together can determine a transformation $\phi$ uniquely under certain boundary conditions. Recall the variational method with prescribed jacobian determinant and curl vector, we use $J(\phi)$ and $\text{curl} \phi$ to construct a transformation $\phi$, so is there a uniqueness statement also? Namely, can the Jacobian determinant and the curl vector together determine a transformation uniquely under a general condition? If not, are there any stronger conditions such that the uniqueness is guaranteed? Intuitively, the Jacobian determinant characterizes the local volume size, and the curl vector characterizes the local rotation, it seems these two determine the local behavior of a transformation.

In order to study this uniqueness problem, we first designed a recovering experiment, and achieved positive observations from numerical tests. These numerical results inspired us to further explore the problem in Mathematics. Later, we proved a related statement under a strong condition in 2D. All these will be discussed in this chapter. And let’s start with the recovering experiment.
4.2 Experiments of Recovering Transformations

Recall the variational method, the constructed \( \phi \) should satisfy \( J(\phi) = f_0(x) \) and \( \text{curl}(\phi) = g_0(x) \). Now, if we take a known transformation \( \phi_0 : \Omega \to \Omega \), compute its Jacobian determinant as our \( f_0 \) and curl vector as \( g_0 \), namely set \( f_0(x) = J(\phi_0(x)), g_0(x) = \text{curl}(\phi_0(x)) \), then apply the variational method. What will we get? Will the constructed \( \phi \) undoubtedly be the known transformation \( \phi_0 \)? If it is true, does it mean the transformation with the Jacobian determinant \( f_0 \) and curl vector \( g_0 \) is unique as \( \phi_0 \)? Consequently, we carried out this procedure by numerical examples, compared the constructed \( \phi \) and \( \phi_0 \) in details. This is our experiment of recovering transformations.

The following figures and tables illustrate corresponding results of two experiments. In each experiment, we tried different values of \( \alpha : 0, 1, 0.1, 10 \). Notice that if \( \alpha = 0 \), only the Jacobian determinant is used, curl vector is not used.

Our first experiment uses a "brain-like" 2D transformation. Random gaussian noise is added to get \( \phi_0 \). Figure 4.1, 4.2 and table 4.1 show the results and comparison.
Figure 4.1. Experiment 1, 65 × 65 grid nodes. The black star dots * represent $\phi_0$, and red dots · represent constructed $\phi$. 
Figure 4.2. Experiment 1-continued, $65 \times 65$ grid nodes. The black star dots * represent $\phi_0$, and red dots · represent constructed $\phi$.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\alpha = 0$</th>
<th>$\alpha = 1$</th>
<th>$\alpha = 0.1$</th>
<th>$\alpha = 10$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ssd$</td>
<td>4.8134</td>
<td>0.6377</td>
<td>1.6869</td>
<td>1.8762</td>
</tr>
<tr>
<td>maximal distance$^1$</td>
<td>0.3444</td>
<td>0.0706</td>
<td>0.0757</td>
<td>0.0797</td>
</tr>
<tr>
<td>average distance</td>
<td>0.1109</td>
<td>0.0177</td>
<td>0.0198</td>
<td>0.0203</td>
</tr>
<tr>
<td>maximal angle difference$^2$</td>
<td>25.8343</td>
<td>15.7230</td>
<td>16.5480</td>
<td>16.6378</td>
</tr>
<tr>
<td>average angle difference</td>
<td>2.6132</td>
<td>1.7481</td>
<td>1.8057</td>
<td>1.9603</td>
</tr>
</tbody>
</table>

Computational domain is $[1, 65] \times [1, 65]$, $^1$distance means $\|\phi(x) - \phi_0(x)\|_2, x \in \Omega$, $^2$angle difference means the corresponding angle differences in every corresponding quadrilateral.

Table 4.1. Comparison of Experiment 1
The second experiment uses a 2D transformation with significant curl vectors, which provides more convincing results. Also a noise is added. Figure 4.3, 4.4 and table 4.4 show the results and comparison.

Figure 4.3. Experiment 2, 65 × 65 grid nodes. The black star dots * represent $\phi_0$, and red dots · represent constructed $\phi$. 
Figure 4.4. Experiment 2-continued, 65 × 65 grid nodes. The black star dots * represent $\phi_0$, and red dots · represent constructed $\phi$.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\alpha = 0$</th>
<th>$\alpha = 1$</th>
<th>$\alpha = 0.1$</th>
<th>$\alpha = 10$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssd</td>
<td>$2.9607e+003$</td>
<td>$0.4004$</td>
<td>$0.5845$</td>
<td>$0.4924$</td>
</tr>
<tr>
<td>maximal distance$^1$</td>
<td>13.9553</td>
<td>0.1311</td>
<td>0.1738</td>
<td>0.1434</td>
</tr>
<tr>
<td>average distance</td>
<td>4.6372</td>
<td>0.0129</td>
<td>0.0162</td>
<td>0.0141</td>
</tr>
<tr>
<td>average angle difference</td>
<td>28.4427</td>
<td>1.0460</td>
<td>1.1987</td>
<td>1.1026</td>
</tr>
</tbody>
</table>

Computational domain is $[1, 65] \times [1, 65]$, $^1$distance means $\|\phi(x) - \phi_0(x)\|_2$, $\alpha \in \Omega$, $^2$angle difference means the corresponding angle differences in every corresponding quadrilateral.

Table 4.2. Comparison of Experiment 1
Comparing the cases of $\alpha = 0$ and $\alpha \neq 0$, it is obvious that the curl vector plays an important role in transformations. With both the Jacobian determinant and curl vector, we can recover a transformation quite precisely. On one hand, the numerical results encourage us to think about the uniqueness problem in mathematical theory, which will be discussed next. On the other hand, these experiments show again the variational method is reliable and accurate in transformation construction. By the way, we can also observe that there is no significant difference among different values of $\alpha$.

4.3 Theoretical analysis

First we describe the problem as: suppose there are 2 diffeomorphisms $\Phi$ and $\Psi$ from $\Omega$ to $\Omega$, such that

\begin{align*}
J(\Psi) & = J(\Phi), \quad (4.1) \\
\text{curl} \Psi & = \text{curl} \Phi, \quad (4.2) \\
\Psi & = \Phi \text{ on } \partial \Omega. \quad (4.3)
\end{align*}

Is it true that $\Psi \equiv \Phi$ on $\Omega$?

We start our study by the simplest case in 2D, where one of the diffeomorphism is the identity map.
4.3.1 One diffeomorphism is the identity map

Let $\Phi$ be the identity map, $u$ is the difference between $\Phi$ and $\Psi$, bounded on $\Omega$ and zero on $\partial \Omega$. More specifically,

$$
\Phi = \text{identity}, \quad (4.4)
$$

$$
\Psi - \Phi = u, \quad (4.5)
$$

$$
\|u\|_{H^2_0(\Omega)} = O(\epsilon), \quad (4.6)
$$

$$
u = 0 \quad \text{on} \quad \partial \Omega. \quad (4.7)
$$

Then $\Phi(x, y) = (x, y)$, by (4.4), $J(\Phi) = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}$.

$\Psi(x, y) = (x + u_1(x, y), y + u_2(x, y))$, so

$$
J(\Psi) = \begin{vmatrix} 1 + u_{1x} & 0 + u_{2x} \\ 0 + u_{1y} & 1 + u_{2y} \end{vmatrix} = 1 + u_{1x} + u_{2y} + u_{1x}u_{2y} - u_{1y}u_{2x}
$$

$$
= 1 + u_{1x} + u_{2y} + J(u) = 1 + \text{div} \ u + J(u).
$$

By (4.1), we must have $\text{div} \ u + J(u) = 0$, and by (4.2), $\text{curl} \ u = 0$, i.e.

$$
\begin{cases}
\text{div} \ u = -J(u), \\
\text{curl} \ u = 0.
\end{cases} \quad (4.8)
$$

Which leads to the following equations:

$$
\begin{cases}
\Delta u_1 = (-J(u))_x = O(\epsilon^2), \\
\Delta u_2 = (-J(u))_y = O(\epsilon^2).
\end{cases} \quad (4.9)
$$

For the first and second derivative of $u$ is $O(\epsilon)$ (4.6), and $J(u) = u_{1x}u_{2y} - u_{1y}u_{2x}$.

Note that $u = (u_1, u_2)$, $\Delta u = (\Delta u_1, \Delta u_2)$, we have

$$
\|\Delta u\|_{L^2} = \left( \int\int_{\Omega} |\Delta u|^2 \right)^{\frac{1}{2}} = O(\epsilon^2). \quad (4.10)
$$
By Green’s first identity and (4.7), we have
\[
\iint_{\Omega} |\nabla u|^2 = \iint_{\Omega} u \cdot \Delta u.
\]
(4.11)

By Cauchy-Schwarz inequality, the RHS of (4.11):
\[
| \iint_{\Omega} u \cdot \Delta u | \leq \iint_{\Omega} | u \cdot \Delta u | \leq \| u \|_{L^2} \| \Delta u \|_{L^2}.
\]

By Poincaré’s inequality, the LHS of (4.11):
\[
\iint_{\Omega} |\nabla u|^2 \geq C \| u \|^2_{L^2},
\]
with C is an optimal constant.

Combine those two,
\[
C \| u \|^2_{L^2} \leq \iint_{\Omega} |\nabla u|^2 = \iint_{\Omega} u \cdot \Delta u \leq \| u \|_{L^2} \| \Delta u \|_{L^2}.
\]
(4.12)

First, we get
\[
\| u \|_{L^2} \leq \| \Delta u \|_{L^2} \leq O(\epsilon^2).
\]

Plug it back into (4.11), second we get \( \| \nabla u \|_{L^2}^2 = \iint_{\Omega} |\nabla u|^2 \leq \| u \|_{L^2} \| \Delta u \|_{L^2} O(\epsilon^2) \leq O(\epsilon^4) \).

Namely,
\[
\| u \|_{L^2} \leq O(\epsilon^2), \quad \| \nabla u \|_{L^2} \leq O(\epsilon^2).
\]
(4.13)

Last, \( \Delta u_1 = ( -J(u) )_x = -u_{1x}u_{2y} - u_{1y}u_{2x} \), \( \Delta u_1 = -u_{1x}u_{2y} - u_{1x}u_{2y} + u_{1y}u_{2x} + u_{1y}u_{2x} \) \( \Rightarrow | \Delta u_1 | \leq O(\epsilon^4) \). Same with \( \Delta u_2 \).

We can apply the same procedure (4.11-4.13) again under the new condition \( \| \Delta u \|_{L^2} \leq O(\epsilon^4) \), and achieve \( \| u \|_{L^2} \leq O(\epsilon^4) \) at the end.

It is clear to see this iterative work will achieve results as \( \| u \|_{L^2} \leq O(\epsilon^2), O(\epsilon^4), O(\epsilon^6) \)....

Hence, \( \| u \|_{L^2} = 0 \), namely \( u \equiv 0, \Psi = \Phi \).
4.3.2 Two diffeomorphisms close to the identity map

Next, we suppose $\Phi$ and $\Psi$ both differ a small diffeomorphism with identity map, i.e.:

$$\Phi = \text{id} + u \quad (4.14)$$

$$\Psi = \text{id} + v \quad (4.15)$$

$$\|u\|_{H^2_0(\Omega)} = O(\epsilon) \quad \|v\|_{H^2_0(\Omega)} = O(\epsilon) \quad (4.16)$$

$$u = v = 0 \quad \text{on} \quad \partial \Omega. \quad (4.17)$$

By (4.1,4.2), we have

$$\begin{cases}
J(\Phi) = 1 + \text{div } u + J(u) = J(\Psi) = 1 + \text{div } v + J(v), \\
curl u = curl v,
\end{cases}$$

which equals

$$\begin{cases}
\text{div} (u - v) = -(J(u) - J(v)), \\
curl (u - v) = 0.
\end{cases} \quad (4.18)$$

Similarly,

$$\begin{cases}
\Delta (u - v)_1 = -(J(u) - J(v))_x = O(\epsilon^2) \\
\Delta (u - v)_2 = -(J(u) - J(v))_y = O(\epsilon^2).
\end{cases} \quad (4.19)$$

We can carry out the similar iterative procedure, achieve $\|u - v\|_{L^2} \leq O(\epsilon^2), O(\epsilon^4), O(\epsilon^6), ...$

Hence, $\|u - v\|_{L^2} = 0$, namely $u \equiv v$, $\Psi = \Phi$.

4.3.3 Conclusion

Overall, we just get a very weak statement. If two diffeomorphisms are "very close" to the identity map, with fixed boundary condition, same Jacobian determinant and same curl vector imply they are the same map. It is quite far from a general conclusion. And we couldn't even figure out the uniqueness for the following simple
problem: If $J(\phi) = 1$, $\text{curl}\phi = 0$, $\phi|_{\partial\Omega} = \text{identity}$, is $\phi : \Omega \to \Omega$ the identity map?

Despite this, we are still optimistic about the potential uniqueness.
CHAPTER 5
APPLICATION TO NON-RIGID IMAGE REGISTRATION

5.1 Image registration and transformations

A image is actually a scalar function $I(x,y)$ in 2D, $I(x,y,z)$ in 3D. So for 2 different images $I_1(x,y,z)$ and $I_2(x,y,z)$, Image registration process is to find a transformation $\Phi$ such that $I_1(\Phi) = I_2$ or $I_2(\Phi) = I_1$. Hence, our methods of construction of diffeomorphism can be potentially applied in this area.

5.2 Optimal control approach

The first attempt[16] is based on one version of the deformation method. Let $f$ be a positive function such that $\int\int_{\Omega}(f - 1)dx = 0$. The following two steps will generate a transformation $\Phi$ with $J(\Phi) = f(\Phi)$.

- First, to generate a vector field $u$ from the div-curl system:

\[
\begin{align*}
\text{div } u &= f - 1, \\
\text{curl } u &= g \text{ on } \Omega, \\
u &= 0 \text{ on } \partial\Omega.
\end{align*}
\]

(5.1)

- Second, to form a grid node velocity field for $s$ in $[0,1]$:

\[
v = u/(s + (1 - s)f).
\]

Then find the transformation $\Phi(x) = \Phi(x,1)$, where $\Phi(x,s)$ is determined from the differential equation

\[
\partial\Phi(x,s)/\partial s = v(\Phi,s).
\]
The main idea of the optimal control approach in image registration is to find $\Phi$ by optimizing a similarity measure, subject to the above equations. Previously, we use $f$ and $g$ as the control functions. By adjusting $f$ and $g$, we use the above equations to form the largest possible search space which consists of all smooth, invertible transformations.

But, the normalization requirement of $f$ (i.e. $\int\int_{\Omega} (f - 1) dx = 0$) and the positivity requirement $f > 0$ are not easy to maintain during the optimization process. Moreover, the calculation of variational gradient is not based on solid mathematical derivation.

5.3 New algorithm of nonrigid image registration

Next, a new algorithm based on variational method is developed[17]. Given the template image $T$ and the reference image $R$ in 2D, we determine a registration transformation $\Phi$ by iteratively minimizing

$$SSD(\Phi) = \frac{1}{2} \int\int_{\Omega} (T(\Phi(x)) - R(x))^2 dx$$  \hspace{1cm} (5.2)

under constraints:

1. The registration transformation $\Phi$ is updated from the current transformation by solving one-step forward finite difference method. Namely:

$$\Phi(x) = \Phi_{old}(x) + \frac{u(x)}{1 + div u} \Delta t.$$  \hspace{1cm} (5.3)

2. $u$ satisfies,

$$\Delta u = F = (f_1, f_2), u = 0 \text{ on } \partial \Omega.$$  \hspace{1cm} (5.4)

3. Control functions $f_1$ and $f_2$ are the defined by

$$f_1 = f_{x_1} - g_{x_2} \text{ and } f_2 = f_{x_2} + g_{x_1},$$  \hspace{1cm} (5.5)
where \( f, g \) are the same ones in (5.1).

We can derive that
\[
\frac{\partial SSD}{\partial F} = (a + \nabla b) \Delta t.
\]

Vector \( a \) is determined by solving the Poisson equations
\[
\Delta a = w \text{ on } \Omega, \\
a = 0 \text{ on } \partial \Omega, \tag{5.6}
\]
with
\[
w = \frac{T(\Phi(x)) - R(x)}{1 + \text{div} u} \nabla T(\Phi(x)),
\]
and function \( b \) is determined by solving the Poisson equation
\[
\Delta b = h \text{ on } \Omega, \\
b = 0 \text{ on } \partial \Omega, \tag{5.7}
\]
with
\[
h = \frac{T(\Phi(x)) - R(x)}{(1 + \text{div} u)^2} \nabla \cdot u.
\]

The derivation of \( \frac{\partial SSD}{\partial F} \) is briefly showed as follows.

Let \( F = (f_1, f_2) \), and \( \delta F = (\delta f_1, \delta f_2) \) be variations that vanish at the boundary. Then \( \Delta \delta u = \delta F = (\delta f_1, \delta f_2) \) from (5.4). We have
\[
\delta SSD = \iint_{\Omega} (T - R) \delta T dx = \iint_{\Omega} (T - R) \nabla T \cdot \delta \Phi dx.
\]
Since \( \delta \Phi = \delta \left( \frac{u}{1 + \text{div} u} \right) \Delta t = \frac{\delta u}{1 + \text{div} u} \Delta t - \frac{u(\text{div}(\delta u))}{(1 + \text{div} u)^2} \Delta t \), we get
\[
\delta SSD = \iint_{\Omega} \frac{(T - R) \nabla T \cdot \delta u}{1 + \text{div} u} \Delta t dx - \iint_{\Omega} \frac{(T - R)(\nabla T \cdot u)}{(1 + \text{div} u)^2} \text{div}(\delta u) \Delta t dx.
\]

59
Let $a$ and $b$ be the solutions to the Poisson equations (5.6) and (5.7) respectively. Then,

$$
\delta SSD = \int_{\Omega} w \cdot \delta u \Delta t \, dx - \int_{\Omega} h \, div(\delta u) \Delta t \, dx \\
= \int_{\Omega} (\Delta a \cdot \delta u - \Delta b \, div(\delta u)) \Delta t \, dx \\
= \int_{\Omega} (\Delta a \cdot \delta u - b \, \Delta (div(\delta u))) \Delta t \, dx \\
= \int_{\Omega} (\Delta a \cdot \delta u - b \, div(\Delta \delta u)) \Delta t \, dx \\
= \int_{\Omega} (a \cdot \Delta \delta u + \nabla b \cdot \Delta \delta u) \Delta t \, dx \\
= \int_{\Omega} ((a + \nabla b) \delta F) \Delta t \, dx.
$$

(5.8)

Hence $\partial SSD / \partial F = (a + \nabla b) \Delta t$.

An optimization scheme by gradient descend method can be applied now.

5.4 Development of non-rigid image registration algorithm

5.4.1 Simplification

Similar to what we did in the variational method:version 2, we can change the constraints (5.3,5.4,5.5) to

$$
\Phi(x) = x + u(x),
$$

(5.9)

and

$$
\Delta u = F = (f_1, f_2), u = 0 \text{ on } \partial \Omega,
$$

(5.10)

where $f_1$ and $f_2$ are two independent control functions, no longer we need (5.5).

Now, we can derive that

$$
\partial SSD / \partial F = a,
$$

where vector $a$ is determined by solving the Poisson equations

$$
\begin{cases}
\Delta a = (T(\Phi(x)) - R(x)) \nabla T(\Phi(x)) \text{ on } \Omega, \\
a = 0 \text{ on } \partial \Omega.
\end{cases}
$$

(5.11)
We shall just present some image registration results here to demonstrate the capacity of our algorithms.

(a) Reference Image R
(b) Template Image T
(c) Register T to R
(d) The Transformation
(e) Register R to T
(f) The Transformation

Figure 5.1. Image Registration Example 1.
Figure 5.2. Image Registration Example 2.
5.4.2 Symmetric scheme

Invertibility and Transitivity are two important properties of transformation. These two properties are not certainly implied in the algorithms above. In [26] and [27], a Symmetric Scheme is mentioned, so we tried to extend our algorithm in the same way:

Suppose there exists an implicit middle image $M$, and we register both the reference image $R$ and the template image $T$ simultaneously to this $M$. Namely to construct two transformations $h_R$ and $h_T$, such that

$$SSD(h_T, h_R) = \frac{1}{2} \int \int_O (T(h_T(x)) - R(h_R(x)))^2 dx$$

is minimized.

![Figure 5.3. illustration of symmetric scheme.](image)

Similarly, we shall derive the gradient $\frac{\partial SSD}{\partial f}$ and $\frac{\partial SSD}{\partial g}$ based on the following assumptions:

$$h_R(x) = x + u(x),$$

$$h_T(x) = x + v(x).$$

(5.12)
With the displacement \( u \) and \( v \) satisfy the Poisson’s equations

\[
\begin{cases}
\Delta u = f, \\
\Delta v = g, \\
u, v = 0 \text{ on } \partial \Omega.
\end{cases}
\] 

(5.13)

Let \( f = (f_1, f_2), g = (g_1, g_2) \), and \( \delta f = (\delta f_1, \delta f_2), \delta g = (\delta g_1, \delta g_2) \) be variations that vanish at the boundary. And from (5.13), \( \Delta \delta u = \delta f, \Delta \delta v = \delta g \).

By variational calculus,

\[
\delta SSD = \iint_{\Omega} (T(h_T(x)) - R(h_R(x))) (\nabla T \cdot \delta h_T(x) - \nabla R \cdot \delta h_R(x)) \, dx.
\]

Since \( \delta h_T(x) = \delta v(x), \delta h_R(x) = \delta u(x) \), we get

\[
\delta SSD = \iint_{\Omega} (T(h_T(x)) - R(h_R(x))) \nabla T \cdot \delta v - \nabla R \cdot \delta u \, dx.
\]

Let \( a \) and \( b \) be the solutions to the Poisson equations:

\[
\begin{cases}
\Delta b = (T(h_T(x)) - R(h_R(x))) \nabla T, \\
\Delta a = -(T(h_T(x)) - R(h_R(x))) \nabla R, \\
a, b = 0 \text{ on } \partial \Omega.
\end{cases}
\] 

(5.14)

Now,

\[
\delta SSD = \iint_{\Omega} (\Delta a \cdot \delta u + \Delta b \cdot \delta v) \, dx
\]

\[
= \iint_{\Omega} (a_1 \Delta u_1 + a_2 \Delta u_2 + b_1 \Delta v_1 + b_2 \Delta v_2) dx
\]

\[
= \iint_{\Omega} (a_1 \delta u_1 + a_2 \Delta u_2 + b_1 \Delta v_1 + b_2 \Delta v_2) dx
\]

\[
= \iint_{\Omega} (a \cdot \delta \Delta u + b \cdot \delta v) \, dx
\]

\[
= \iint_{\Omega} (a \cdot \delta f + b \cdot \delta g) \, dx.
\]

Finally, we obtain

\[
\frac{\partial SSD}{\partial f} = a
\]

\[
\frac{\partial SSD}{\partial g} = b.
\] 

(5.15)
After finding $h_R, h_T, h_T \circ h_R^{-1}$ is the transformation from $R$ to $T$, $h_R \circ h_T^{-1}$ is the transformation from $T$ to $R$, and they are invertible and transitive.

5.5 Conclusion

Overall, Image registration is a complicated task, it requires many other works such as pre-processing, post-processing, segmentation etc. There are three components provide a common classification schema for registration methods: the transformation model, the similarity(or correspondence) measures, and the optimization strategy. Many methods with different characteristic features are invented in this area. We want to point out that the major features of our non-rigid image registration method are: it bases on a solid but simple mathematical foundation, and it can produce a smooth transformation between images.

5.6 Graphical User Interface:

We shall end this chapter with another graphical user interface we designed for image registration: ”Image Registration Tools”. It allows anyone to input two images as template image and reference image, then implement our registration algorithm, and provide the corresponding transformation. The next two figures illustrate works done by ”Image Registration Tools”.

65
Figure 5.4. Image Registration Tools example 1.

Figure 5.5. Image Registration Tools example 2.
REFERENCES


BIOGRAPHICAL STATEMENT

Xi Chen was born in Hefei, Anhui, China, in 1990. He received his B.S. degree from the special class for the gifted young, University of Science and Technology of China, in 2010, his Ph.D. degree from The University of Texas at Arlington in 2016, all in Mathematics and Applied Mathematics. During his 5 years doctoral studies, he also served as a Graduate Teaching Assistant in the department of mathematics, taught multiple undergraduate courses as instructor. His research interest is in the areas of numerical grid generation and image registration. He received Outstanding Graduate Research Award in 2016.