Curve Sampling and Geometric Conditional Simulation

by

Ayres C. Fan

B.S., Electrical Engineering
University of Minnesota, 2000

S.M., Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 2003

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

February 2008

Signature of Author: _____

Department of Electrical Engineering and Computer Science
November 13, 2007

Certified by: _____

Alan S. Willsky
Edwin S. Webster Professor of Electrical Engineering
Thesis Supervisor

Accepted by: _____

Terry P. Orlando
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Curve Sampling and Geometric Conditional Simulation

by Ayres C. Fan

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering
and Computer Science at the Massachusetts Institute of Technology

Abstract

The main objective of this thesis is the development and exploitation of techniques to generate geometric samples for the purpose of image segmentation. A sampling-based approach provides a number of benefits over existing optimization-based methods such as robustness to noise and model error, characterization of segmentation uncertainty, natural handling of multi-modal distributions, and incorporation of partial segmentation information. This is important for applications which suffer from, *e.g.*, low signal-to-noise ratio (SNR) or ill-posedness.

We create a curve sampling algorithm using the Metropolis-Hastings Markov chain Monte Carlo (MCMC) framework. With this method, samples from a target distribution π (which can be evaluated but not sampled from directly) are generated by creating a Markov chain whose stationary distribution is π and sampling many times from a proposal distribution q. We define a proposal distribution using random Gaussian curve perturbations, and show how to ensure detailed balance and ergodicity of the chain so that iterates of the Markov chain asymptotically converge to samples from π. We visualize the resulting samples using techniques such as confidence bounds and principal modes of variation and demonstrate the algorithm on examples such as prostate magnetic resonance (MR) images, brain MR images, and geological structure estimation using surface gravity observations.

We generalize our basic curve sampling framework to perform conditional simulation: a portion of the solution space is specified, and the remainder is sampled conditioned on that information. For difficult segmentation problems which are currently done manually by human experts, reliable semi-automatic segmentation approaches can significantly reduce the amount of time and effort expended on a problem. We also extend our framework to 3D by creating a hybrid 2D/3D Markov chain surface model. For this approach, the nodes on the chain represent entire curves on parallel planes, and the slices combine to form a complete surface. Interaction among the curves is described by an undirected Markov chain, and we describe methods to sample from this model using both local Metropolis-Hastings methods and the embedded hidden Markov model (HMM) algorithm.

Thesis Supervisor: Alan S. Willsky
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

First, I would like to thank my thesis supervisor Alan Willsky. I have learned so much from him about how to think about hard estimation problems in a principled and rigorous manner, and his encyclopedic knowledge of so many fields is astounding. I have also been very fortunate to have a thesis committee which has provided me with guidance and assistance throughout my doctoral program. John Fisher has been involved with this work from the beginning, introducing me to the field of sampling and MCMC methods and helping to formulate the curve sampling algorithm. I am also deeply impressed by his ability to come up with interesting acronyms for various methods, though, sadly, we were unable to use PERPS. Sandy Wells always has a unique and creative way of thinking about problems, and his general knowledge about medical imaging and coordinating interaction with clinical research groups was essential. And I am grateful to Polina Golland for providing an excellent outside perspective and always reminding me of the need to focus on the big-picture issues.

The work here would not have been possible without the contributions of a number of other people. For the medical imaging applications, Clare Tempany provided the prostate MR data and has always been very encouraging and supportive of my work; Martha Shenton supplied the brain MR data; and James Levitt performed the hand segmentations of the thalamus. I am indebted to Jonathan Kane for introducing me to the gravity inversion problem, and discussions with Ron Masters and Ed Biegert were very helpful.

The Stochastic System Group has been my home for over seven years. It would not have been able to function without the assistance of a number of very capable administrative assistants, especially Taylore Kelly, Laura Clarage, and Rachel Cohen. Many

people have passed through the group over the years. I want to especially acknowledge Andy Tsai for taking me under his wing when I first arrived; Walter Sun for being a very good friend and resource over the years (though his ignorance about topics such as the best college football and basketball conferences is shocking); and my officemates for the bulk of my time here: Dmitry Malioutov for always being willing to scavenge for food with me, Jason "−1" Johnson for always having interesting things to say, and Lei Chen for teaching me colorful phrases in Mandarin. I have also benefited from interactions with numerous group members over the years including Andrew Kim, Dewey Tucker, Martin Wainwright, Ron Dror, Alex Ihler, Junmo Kim, Erik Sudderth, Mujdat Cetin, Jason Williams, Pat Kreidl, Kush Varshney, Emily Fox, Venkat Chandrasekaran, Jin Choi, Sujay Sanghavi, Vincent Tan, and Michael Chen.

The support and friendship of numerous people I've met during my time in Minnesota (from Elm Creek to the U) and Massachusetts (including Edgerton, Sidney-Pacific, and LIDS) have helped me to make it through grad school (relatively) unscathed. Finally, I would like to thank my family which is scattered around the world, especially my parents Steve and Sue, my sister Eva, and my niece Elise and nephew Elliott to whom this thesis is dedicated. You two both have the potential to do anything you set your minds to. Don't disappoint us! (only kidding—sort of)

Contents

Abstract **3**

Acknowledgments **5**

List of Figures **11**

List of Tables **15**

Notational Conventions **17**

Common Acronyms **19**

1 Introduction **21**
1.1 Image Segmentation 22
1.2 Contributions 24
1.3 Thesis Summary 27

2 Background **31**
2.1 Markov-Chain Monte Carlo Sampling 31
2.1.1 Metropolis and Metropolis-Hastings 33
2.1.2 Gibbs Sampling 34
2.1.3 Embedded HMMs 36
2.2 Curve Evolution 40
2.2.1 Data-independent Curve Flows 42
2.2.2 Data-driven Curve Evolution 43
2.2.3 Global Optimizers and Statistical Methods 45
2.2.4 Numerical Implementations 46
2.3 Statistical Shape Segmentation 49
2.3.1 Point Distribution Models 49
2.3.2 Dense Shape Representations 50
2.3.3 Shape Distances 52

3 2D Curve Sampling Formulation **55**
3.1 Overall Algorithm . . . 56
3.2 Sampling from the Proposal Distribution . . . 58
3.2.1 Arc Length-parameterized Curve Perturbations . . . 59
3.2.2 Correlated Noise Process . . . 60
3.2.3 Mean Process . . . 63
3.2.4 Numerical Implementation . . . 64
3.2.5 Comparison to Existing Methods . . . 66
3.3 Evaluating the Hastings Ratio . . . 67
3.3.1 Forward Proposal Distribution Probability . . . 69
3.3.2 Reverse Perturbation . . . 69
3.3.3 Approximate ϕ Computation . . . 71
3.3.4 Existence Condition for the Reverse Perturbation . . . 76
3.3.5 Summary of Hastings Ratio Calculation . . . 77
3.4 Online Parameter Estimation . . . 77
3.4.1 Deterministic Transformation . . . 78
3.4.2 Augmentation of the Variable Space . . . 79

4 2D Curve Sampling Results **81**
4.1 Visualizing Curve Samples . . . 81
4.2 Shape Distributions . . . 83
4.2.1 Single Curve Target . . . 84
4.2.2 Shape Distributions Using Parzen Densities . . . 86
4.3 Image-based Examples . . . 89
4.3.1 Synthetic Gaussian Noise . . . 91
4.3.2 Prostate MR . . . 91
4.4 Online Parameter Estimation . . . 94
4.4.1 Experimental Setup . . . 95
4.4.2 Results . . . 98
4.5 Convergence . . . 101
4.5.1 Mixing Rate . . . 102
4.5.2 Convergence in the Number of Samples . . . 106

5 Conditional Simulation **111**
5.1 Incorporating Fixed Constraints . . . 112
5.2 Thalamus Segmentation . . . 115
5.3 Gravity Inversion . . . 120
5.3.1 Curve-based Gravity Model . . . 121
5.3.2 Circular Salt Example . . . 124
5.3.3 Synthetic Two-body Salt Example . . . 127
5.3.4 Real Geometry Example . . . 131
5.3.5 PCA Shape Variation Modes . . . 135

6 Hybrid 2D/3D Surface Sampling **139**
6.1 2D/3D Markov Chain Model 140
6.1.1 Symmetric Area Difference as a Coupling Term 142
6.1.2 Conditional Simulation Approaches 145
6.2 Single-slice Gap 145
6.3 Multiple Unknown Slices 148
6.3.1 Local Metropolis-Hastings Approach 149
6.3.2 Embedded HMM 150
6.3.3 Results for Multiple Contiguous Unknown Slices 151
6.3.4 Varying the Gap Size 153
6.4 Orthogonal Slice Information 161

7 Conclusions and Suggestions for Future Research **165**
7.1 Thesis Contributions 165
7.2 Suggestions for Future Research 169
7.2.1 Sampling Advances 169
7.2.2 Modeling Advances 172
7.2.3 Improved sample visualization 175

A Curve Extraction Algorithm **179**

B Approximate Reverse Perturbations **183**
B.1 Linear Approximation 183
B.2 Second-order Approximation 185

C Forward-Backward Algorithm for Undirected HMMs **191**

Bibliography **195**

List of Figures

1.1 Example prostate and brain MR images. 23
1.2 Gravity observations for salt body localization. 24
1.3 Most-probable and histogram images for prostate MR example. 25

2.1 Markov random field example with A and C conditionally independent given B. 35
2.2 Graph for Neal *et al.* Markov model. $\boldsymbol{x}_i$ are the random variables and $\boldsymbol{z}_i$ are the observations. 37
2.3 Example illustrating one iteration of the embedded HMM algorithm. . . 38
2.4 Negatively-oriented circle proceeds in a clockwise direction. Outward normal points away from the interior of the circle. 40
2.5 Level set surface used to represent an airplane shape. 47

3.1 Construction of curve evolution between two arbitrary curves using intermediate convex curves. 62
3.2 Illustration of the process of computing reverse perturbation value $\phi(q)$. 70
3.3 Illustration of the process of forming linear approximations to $\vec{C}_{\mathrm{a}}(p)$ to simplify the estimation of the reverse perturbation ϕ. 73
3.4 Illustration of the process of forming linear approximations to $\vec{C}_{\mathrm{a}}(p)$ to simplify the estimation of the reverse perturbation ϕ. 74
3.5 Illustration of a case when the reverse perturbation actually does not exist because no value of $\phi(q_0)$ will cause an intersection of the line along $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ with the original curve $\vec{C}$. 76

4.1 Target shape and initial curve for SAD target. 84
4.2 Results for SAD target with $\alpha = 1$ and $\beta = 0.01$. 85
4.3 Results for SAD target with $\alpha = 0.5$ and $\beta = 0.005$. 86
4.4 Results for Parzen density with SAD distance, $\alpha = 1$ and $\beta = 0.07$. . . . 87
4.5 Results for the Parzen SAD example after clustering the resulting samples. 88
4.6 Results for the synthetic Gaussian example. 90
4.7 Prostate segmentation using non-parametric intensity distributions. . . . 93

4.8 Clustered results for the prostate sampling problem. 94
4.9 Most likely samples and marginal confidence intervals from three different curve sampling algorithms with parameter estimation. 97
4.10 Estimated marginal distributions of m_0 and (m_1 for four different sampling algorithms. 100
4.11 Observed noisy image with true contour location in red and initial curve $\vec{C}_i^{(0)}$ for all $i \in \{1, 2, \ldots, 1000\}$. 103
4.12 Progression of histogram images with respect to time for the synthetic L2 example. 104
4.13 Log plot of the error in the histogram image as a function of time. . . . 105
4.14 Delta change in the histogram image as a function of time. 106
4.15 Progression of histogram images with respect to the number of samples for the synthetic L2 example. 107
4.16 Plot of the error and the delta change in the histogram image as a function of the number of samples. 108

5.1 Axial thalamus MR image with expert segmentation and estimated pdf. 115
5.2 Conditionally-simulated thalamus segmentation using non-parametric intensity distributions. 118
5.3 Depiction of idealized physical model for gravity inversion problem. . . . 122
5.4 Observed gravity profile for circular synthetic salt body. 124
5.5 Synthetic circular salt body: initialization, local minimum, and sampled results. 125
5.6 Noisy observed gravity profile for circular synthetic salt body. 126
5.7 Sampling results for synthetic circular salt body with noise. 127
5.8 Observed gravity profile for disjoint synthetic salt body with complex geometry. 128
5.9 Disjoint synthetic salt body with complex geometry: local minimum, initialization, unconstrained results. 129
5.10 Conditional simulation results for synthetic two-body salt example. . . . 130
5.11 Observed seismic for real salt body. 132
5.12 Gravity profile for real salt body example. 132
5.13 Conditional simulation results for two-body salt example derived from seismic image. 134
5.14 Principal modes of variation for the synthetic two-body salt example. . 136

6.1 Model of 3D volume using 2D slices connected as a Markov chain. . . . 141
6.2 Zero-order hold approximation to a 3D surface from curves located on parallel slices. 143
6.3 Hybrid 2D/3D Markov chain model with single-slice gap. 146
6.4 Single-slice hybrid 2D/3D Markov chain sampling results on a thalamus slice. 147

6.5 Model of 3D volume using 2D slices connected as a Markov chain with a multi-slice gap. 148
6.6 Results using the local Metropolis-Hastings and embedded HMM methods with a gap size of 4. 152
6.7 Most probable samples from the local Metropolis-Hastings approach with varying gap size. 154
6.8 Histogram images and marginal confidence bounds from the local Metropolis-Hastings approach with varying gap size. 155
6.9 Most probable samples from the embedded HMM approach with varying gap size. 156
6.10 Histogram images and marginal confidence bounds from the embedded HMM approach with varying gap size. 157
6.11 Variance image of a slice of the thalamus generated by the local Metropolis-Hastings approach. 159
6.12 Plot of the L2 distance between the histogram image and the the binary expert segmentation for each slice for a fixed gap size. 160
6.13 Plot of the L2 error as a function of the gap size G. 160
6.14 Markov chain model with perpendicular slice constraint. 161
6.15 Sagittal thalamus MR image with expert segmentation for two different slices. 163
6.16 Most probable samples, histogram images, and marginal confidence bounds using two sagittal slice segmentations to conditionally simulate axial slices. 164

7.1 Depiction of converting a known curve segment $\vec{C}_k$ into related region constraints $\mathcal{R}_{\text{in}}$ (light gray) and $\mathcal{R}_{\text{out}}$ (dark gray). 174
7.2 Depiction of idealized physical model for gravity inversion problem. . . . 174

B.1 Forming a linear approximation to $\vec{C}_{\text{a}}(p)$ to simplify the estimation of the reverse perturbation $\phi(q_0)$ when $\vec{C}_{\text{a}}$ is (a) convex or (b) concave at p_0. 184
B.2 Forming an approximation to $\vec{C}_{\text{a}}(p)$ using a circle to simplify the estimation of the reverse perturbation $\phi(q_0)$ when $\vec{C}_{\text{a}}$ is (a) convex or (b) concave at p_0. 186
B.3 Graphical depiction of both solutions to the quadratic equation for the circle approximation when $f(p_0) > 0$ and $\kappa_{\vec{C}_{\text{a}}}(p_0) > 0$ with the outward normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ pointing toward the top of the page. 188
B.4 Graphical depiction of both solutions to the quadratic equation for the circle approximation when $f(p_0) > 0$ and $\kappa_{\vec{C}_{\text{a}}}(p_0) < 0$ with the outward normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ pointing toward the top of the page. 189

List of Tables

4.1 Average m_0 and m_1 estimates for different curve sampling algorithms with parameter estimation. 99

Notational Conventions

Symbol	Definition
General Notation	
$\mathbb{R}$	set of real numbers
$\mathbb{R}^n$	space of n-dimensional vectors
$\boldsymbol{I}$	identity matrix
$\mathcal{A} \setminus \mathcal{B}$	set $\mathcal{A}$ minus the set $\mathcal{B}$
$\mathcal{A} \cap \mathcal{B}$	intersection of sets $\mathcal{A}$ and $\mathcal{B}$
$\mathcal{A} \cup \mathcal{B}$	union of sets $\mathcal{A}$ and $\mathcal{B}$
$\mathcal{A}^c$	complement of the set $\mathcal{A}$
$\{x_i\}_{i=1}^N$	the set of elements $\{x_1, x_2, \ldots, x_N\}$
$\boldsymbol{A}^{\mathrm{T}}$	transpose of matrix $\boldsymbol{A}$
$\lvert\cdot\rvert$	absolute value
$\langle \boldsymbol{x}, \boldsymbol{y} \rangle$	inner product of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$
$\lVert\cdot\rVert$	ℓ_2 norm
$\mathrm{d}(x_1, x_2)$	distance between x_1 and x_2 (under some metric)
$\circledast$	circular convolution operator
∇	gradient operator
div	divergence operator
f'	first derivative of f
$\mathcal{O}(\cdot)$	computational complexity is bounded by the argument
Curve Evolution	
I	image
Ω	image domain (typically $\Omega \in \mathbb{R}^2$)
$\vec{C}$	curve function, typically parameterized from 0 to 1
$\vec{C}_{\mathrm{a}}$	arc length-parameterized version of $\vec{C}$
E	energy functional for curve
$\Psi_{\vec{C}}$	level set function for $\vec{C}$
$\vec{\mathcal{T}}_{\vec{C}}$	tangent function to $\vec{C}$
$\vec{\mathcal{N}}_{\vec{C}}$	normal function to $\vec{C}$
$\kappa_{\vec{C}}$	curvature function of $\vec{C}$
$\mathcal{R}_{\vec{C}}$	region enclosed by curve $\vec{C}$
$\mathcal{H}$	Heaviside (or indicator) function
$L_{\vec{C}}$	curve length of $\vec{C}$
$\mathrm{d}s$	differential arc length

Symbol	Definition
Curve Evolution cont.	
$\vec{S}$	surface function
$\mathcal{S}$	set of 2D curves that approximate a surface
$\vec{c}$	individual 2D curves that make up a 2D/3D surface
MCMC Sampling	
π	target probability distribution
$\mathtt{q}$	proposal distribution
$\mathtt{a}$	acceptance function
$x^{(t)}$	value of variable x at time t
η	Hastings ratio
$\vec{\Gamma}$	candidate curve sample (perturbed version of $\vec{C}$)
$\vec{C}_{\mathrm{r}}$	curve after reverse perturbation from $\vec{\Gamma}$
f	forward curve perturbation (or force function for deterministic curve evolution)
ϕ	reverse curve perturbation
n	forward white noise
ν	reverse white noise
r	forward correlated noise
ξ	reverse correlated noise
h	kernel function for low-pass filter
$\boldsymbol{H}$	matrix implementing discrete circular convolution by h
$\mu_{\vec{C}}$	mean perturbation from $\vec{C}$
$\gamma_{\vec{C}}$	constant for balloon force in mean perturbation
Φ	histogram image from aggregated samples
ρ_i	constellation-generating probability distribution

Common Acronyms

Acronym	Definition
cdf	cumulative density function
cmf	cumulative mass function
HMM	hidden Markov model
iid	independent and identically distributed
MAP	maximum *a posteriori*
MCMC	Markov chain Monte Carlo
ML	maximum likelihood
MR	magnetic resonance
MRF	Markov random field
PCA	principal components analysis
pdf	probability density function
pmf	probability mass function
SAD	symmetric area difference
SDF	signed distance function
SNR	signal-to-noise ratio

Chapter 1

Introduction

THE main objective of this thesis is the development and exploitation of techniques to generate geometric samples for the purpose of *image segmentation*. Image segmentation is the process of identifying coherent regions within images [10, 11, 52, 68, 74, 126] and is an important tool in fields ranging from medicine to video coding to geophysics. Most existing segmentation methods are optimization-based. By creating a sampling-based approach, we are able to provide a number of benefits over existing methods such as robustness to noise and model error, characterization of segmentation uncertainty, and incorporation of partial segmentation information.

These forms of information can be important for people making decisions based on the output of a segmentation algorithm. Examples of this could include an oil company deciding if and where to drill a well; a clinician trying to diagnose and stage diseases; or a surgeon mapping out an optimal surgical plan. In all of these examples, risk and uncertainty are integral components of the decision-making process, and being able to understand and characterize the uncertainty in the image segmentation is crucial to making informed decisions.

We demonstrate our curve sampling algorithm on prostate magnetic resonance (MR) images, brain MR images, and underground geological structure estimation from surface gravity observations. In general, our sampling approach is most beneficial for problems where there is a large amount of uncertainty about the correct answer (due to, *e.g.*, model error, clutter, occlusion, low signal-to-noise ratio (SNR), or ill-posedness) as sampling-based methods can characterize the solution space better than optimization-based methods can. Sampling is especially useful in applications for which statistical shape models [73], a common modeling tool used to improve segmentation performance, are difficult to apply. Examples of this include geological formations [63] and brain tumors [33].

In the remainder of this chapter, we provide an overview of the problems we seek to address and the techniques which we have developed. We begin in Sec. 1.1 with a more detailed introduction to image segmentation and some of the current challenges in the field. Sec. 1.2 describes the major contributions of our work. We conclude in Sec. 1.3 with an outline of the structure of the thesis.

■ 1.1 Image Segmentation

Over the past few decades, there has been a geometric explosion in the quantity of information that is readily available, especially in digital form. These data are available in a wide variety of modalities from simple text to sound and images and video. In the medical field, three-dimensional (3D) methods such as computed tomography (CT) and magnetic resonance (MR) have supplanted two-dimensional (2D) X-rays in many clinical diagnosis procedures. In geology, overhead satellites and on-the-ground seismographs produce a great deal of information that mostly requires experts to analyze. And in photography, the advent of cheap digital imaging sensors has made it very easy to accumulate countless images (*e.g.*, sensing applications such as security cameras).

Computer vision techniques are designed to automatically extract meaning and knowledge from image data. Within that field is the important problem of image segmentation which is useful in a wide variety of fields. In medicine, segmenting the heart can provide ejection fraction calculations [106] and segmenting other organs (*e.g.*, the brain or the prostate) can assist in planning surgical procedures [40, 123] or diagnosing and staging cancer [109]. In oil exploration, segmenting different geological structures can help locate likely locations of oil and quantify the amount of oil within [98]. In higher-level vision algorithms, segmentation algorithms can be used as building blocks for recognition and decision algorithms [119].

A wide variety of techniques have been developed to generate automatic image segmentations. These range from methods involving Markov random field (MRF) models [32, 122] to atlas-based segmentation [55] to active contour methods. The latter are also often referred to as *curve evolution methods* and were pioneered by Kass *et al.* [52]. Active contour methods evolve a curve to minimize an energy functional (which can often be viewed as a negative log probability function) to find object boundaries. These methods can be broadly classified as edge-based [10, 52, 70, 93] or region-based [11, 58, 103, 117, 126, 130] depending on whether the energy functionals are defined

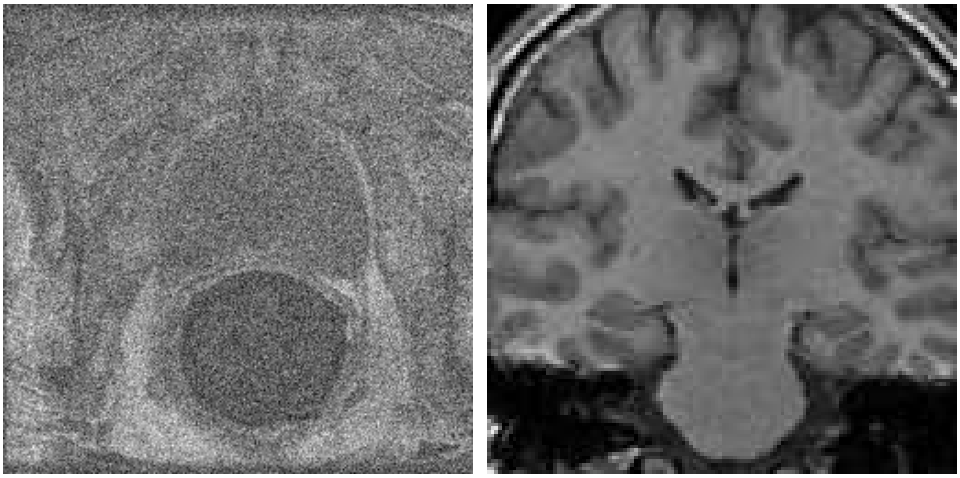


Figure 1.1. Example prostate MR image (left) and brain MR image (right).

over edge or region statistics of the image. Generally, curve evolution methods are implemented using gradient descent to minimize the energy resulting in algorithms which find local minima. More recently, some curve evolution techniques [22, 50, 105, 118] that can find global optima have been developed.

Segmentation is a difficult problem due to a variety of reasons such as poor data models, low signal-to-noise ratio (SNR), missing data, or indistinct edges. Given an image without any other context, there are probably many reasonable possible segmentations. We show in Fig. 1.1 MR images of the prostate and the brain. The prostate image has low SNR while the brain image suffers from limited contrast between different sub-cortical structures such as the thalamus. Prostate segmentation is an important part of developing a surgical plan for an experimental treatment for prostate cancer known as brachytherapy. This technique involves implanting radioactive seeds within the prostate [109] to target diseased tissue with radiation without damaging the healthy tissue. In order to do this, the diseased areas must be accurately located, and the seeds must be accurately placed. Currently this segmentation is manually performed by an expert radiologist. Segmentation of the thalamus in brain MR images is an important task in evaluating a number of diseases such as schizophrenia and Parkinson's [86].

In Fig. 1.2, we display synthetic gravity measurements of a 2D density field. For this problem, we wish to estimate the location of underground bodies of salt. Salt is less dense than rock, so the presence of salt appears as an anomaly in surface gravity

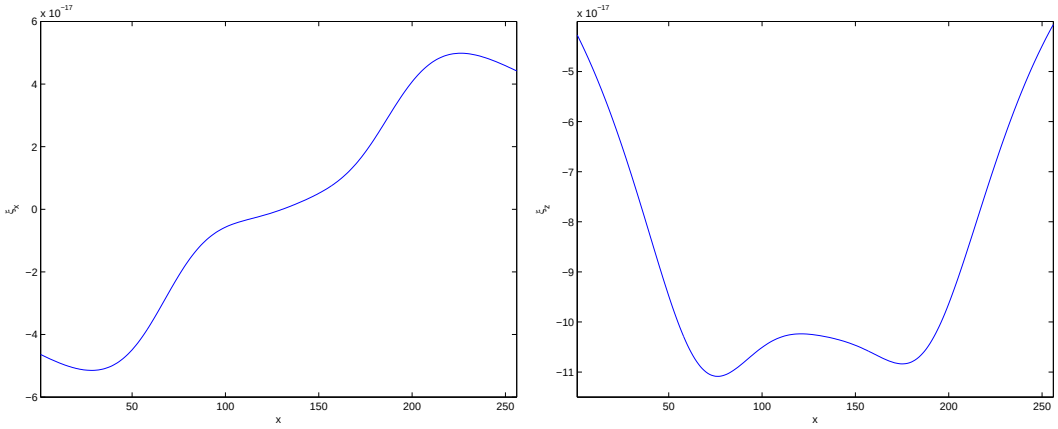


Figure 1.2. Gravity observations for salt body localization. The x-component is on the left, and the z-component is on the right.

observations [7]. Typically, seismographs are used to image beneath the surface of the earth, and a great deal of manual labor is expended to analyze and interpret seismic images for the purposes of oil exploration [2]. Seismic imaging below salt domes is challenging due to the large difference in seismic wave velocity between salt and the surrounding sediment which can cause bending and dispersion of the seismic waves. Knowing the exact boundaries of the salt can improve seismic image reconstruction as well as helping to localize potential locations where petroleum is likely to accumulate. For these reasons, additional imaging techniques such as gravity inversion are often employed to better segment salt regions. Note that this problem is quite ill-posed because, *e.g.*, for the example in Fig. 1.2, we only have 512 observations to reconstruct an image with 65,536 pixels. The current state-of-the-art algorithms involve extensive manual intervention by a geologist [75].

■ 1.2 Contributions

For challenging segmentation problems, traditional optimization-based methods suffer from a number of drawbacks. Gradient-based methods can only find a local optimum which may not be a very good segmentation, and they cannot provide an estimate of how close the segmentation is to the global optimum. Even global optimizers are less than ideal. If the probability distribution is naturally multi-modal, a global optimizer will typically only find one of the modes. For single-mode distributions, finding the

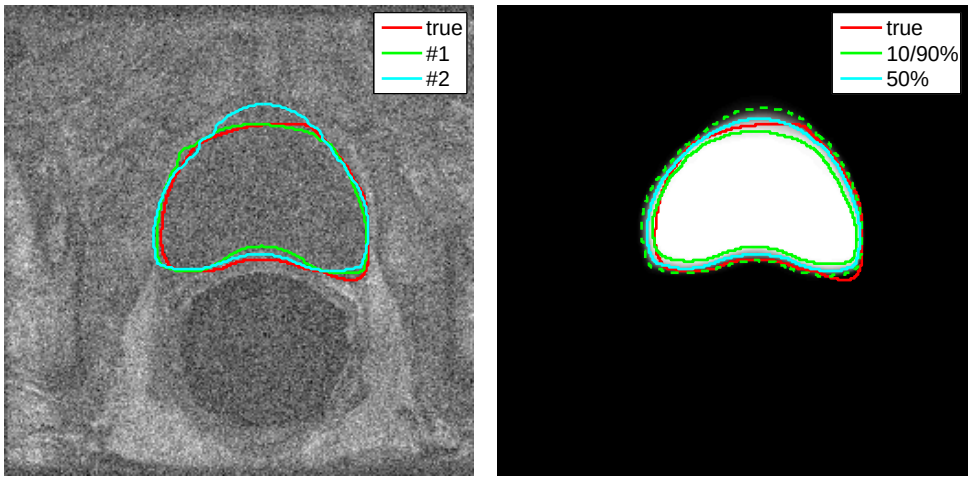


Figure 1.3. Most-probable (left) and histogram image with marginal confidence bounds (right) for prostate MR example.

global optimum does not provide an estimate of the confidence level in that result, and, due to model error and low SNR, it is often the case that the global optimum may not even be a good segmentation of the image. To address these issues, we create a framework that generates samples from a probability distribution over curves $\pi(\vec{C})$ and show how to visualize the samples to extract useful information. We show how this formulation allows us to incorporate user input to constrain the space of valid segmentations and extend the framework to generate samples from a novel hybrid slice-based 2D/3D surface model.

2D curve sampling

Our curve probability distributions are defined over high-dimensional spaces which cannot be sampled from directly. Instead, we can apply a class of techniques known as Markov chain Monte Carlo (MCMC) [44, 71] to generate samples from $\pi(\vec{C})$ in an iterative fashion. With these methods, a so-called *proposal distribution* `q` is defined from which it is easy to sample. Many samples from `q` are generated, and they are accepted or rejected in such a manner so that the results of this iterative procedure converge asymptotically to samples from $\pi(\vec{C})$. MCMC methods thus change the problem of sampling from $\pi(\vec{C})$ to that of generating many samples from `q`. We define samples from the proposal distribution `q` as being random Gaussian perturbations applied to the

normal of the curve. Note that this approach bears some resemblance to the method of Tu and Zhu [118] which also employs Gaussian-based curve perturbations. While there are a number of differences between that approach and ours (which we detail more fully in Sec. 3.2.5), the largest is that we show how to ensure asymptotic convergence of the iterate values so that our algorithm generates samples from the target distribution π. Tu and Zhu use a MCMC-based approach, but only to find optima of π and not to draw samples from it.

We visualize the pool of samples generated by our algorithm using a number of techniques. We examine the samples with the highest probability under the distribution $\pi(\vec{C})$ which should be close to what the output of a global optimizer would be. We also construct the marginal distribution of segmentation labels at each pixel to aggregate information from all of the samples. Level contours of this distribution can be drawn to create lines which represent segmentation confidence levels. The principal modes of variation can be generated in a method similar to the PCA shape models of Leventon *et al* [64]. We display some examples of the first three techniques for a prostate MR example in Fig. 1.3.

We demonstrate a number of advantages of this sampling approach over traditional optimization-based algorithms. Our approach only requires us to be able to evaluate $\pi(\vec{C})$ rather than having to formulate gradients of the distribution. This is a significant benefit in a number of situations such as when an analytical gradient cannot be formulated or for problems with non-local measurements (*e.g.*, the gravity inversion problem shown in Fig. 1.2) for which the gradient often only weakly moves the curve toward the optimal value. We are able to achieve global convergence due to the randomness in our sampling procedure which allows us to fully explore the configuration space. Multimodal distributions can be naturally represented as samples are drawn from all of the major modes. The modes can be automatically distinguished using simple clustering techniques. Sampling allows us to characterize the uncertainty associated with a given segmentation by creating curves which represent confidence bounds.

Conditional simulation

Many segmentation problems are currently done manually by human experts. For these difficult problems, reliable semi-automatic segmentation approaches can provide significant benefits. With these methods, an expert performs part of the segmentation and an algorithm automatically generates the rest of it. This can be implemented as an

interactive process where the expert can then refine the input given to the segmentation algorithm based on the previous output, and the algorithm provides an updated segmentation.

Our curve sampling approach can be easily modified to use knowledge about the location of part of the curve by making simple changes to the proposal distribution. This is in contrast to gradient-based approaches which would require complicated high-dimensional constrained optimization approaches. Incorporating extra information about the problem in this fashion can make problems more tractable and less ambiguous.

Note that the sampling process naturally lends itself to an interactive segmentation approach because the sampler provides estimates of where the greatest uncertainty is in the segmentation. This then indicates locations where an expert user could provide the most assistance. We demonstrate this approach on the challenging thalamus segmentation and gravity inversion problems.

Hybrid surface model

To extend our framework to 3D surfaces, we create a hybrid Markov chain surface model. In this approach, the nodes on the chain represent entire curves on parallel planes, and the slices are combined together to form a complete surface. Interaction among the curves is described by an undirected Markov chain which then specifies a probability distribution for the surface using local pairwise *potential functions*. We describe methods to sample from this model using both local Metropolis-Hastings methods and ones based on the embedded hidden Markov model (HMM) approach of Neal *et al.* [77].

As in the 2D curve sampling case, we can extend the framework to allow for conditional simulation where partial curves or entire curves are specified. We can also allow the user to specify segmentations on slices which are orthogonal to the primary slices of interest so the information provided can simultaneously affect all of the slices. We demonstrate these approaches on a 3D thalamus MR volume.

■ 1.3 Thesis Summary

Chapter 2: Background

In this chapter, we present background material necessary to understand the work in this thesis. We begin by introducing sampling techniques based on MCMC methods. These

include the well-known Metropolis-Hastings and Gibbs sampling algorithms along with the newer embedded HMM method. We then cover curve evolution including classical flows such as curvature; edge-based and region-based methods; and global optimizers and numerical implementation of these flows using the level set method of Sethian and Osher [82]. Finally, statistical shape segmentation methods are discussed including the active shape models of Cootes and Taylor [14], the level set-based principal components analysis (PCA) approach of Leventon *et al.* [65], and non-parametric Parzen densities using shape distances [57].

Chapter 3: 2D Curve Sampling Formulation

This chapter details our 2D curve sampling framework using a Metropolis-Hastings algorithm. A probability distribution over curves is defined by exponentiating standard curve energy functionals. We define a proposal distribution based on random Gaussian perturbations and show how to efficiently generate samples from this distribution. We discuss how to ensure asymptotic convergence of the MCMC algorithm by properly evaluating the proposal distribution and present approximations to make the computations tractable. We extend our approach to incorporate online estimation of parameters in the probability distribution.

Chapter 4: 2D Curve Sampling Results

Here we present results from our 2D curve sampling algorithm. We discuss ways to visualize the output of the sampling algorithm including most-probable samples, marginal pixel label distributions, confidence bounds, and principal modes of variation. We demonstrate results on shape probability distributions, synthetic images with Gaussian noise, and prostate MR images. We also implement online parameter estimation approaches and apply them to a low SNR synthetic image. We evaluate empirical convergence behavior and formulate an approximate stopping criteria for the algorithm.

Chapter 5: Conditional Simulation

We begin this chapter by showing how our basic curve sampling framework can be extended to perform conditional simulation mainly through modification of the proposal distribution. We segment thalamus images by incorporating user input that specifies the location of small portions of the curve. We also formulate a curve-based approach to the gravity inversion problem and construct both optimization- and sampling-based

implementations. We show how conditional simulation can aid in generating results for both of these difficult problems.

Chapter 6: Hybrid 2D/3D Surface Sampling

In this chapter, we construct a surface sampling approach based on a hybrid 2D/3D Markov chain model. We show that symmetric area difference (SAD) is a natural term to couple neighboring slices in the model, and how conditional simulation approaches can be naturally integrated. In contrast to the 2D conditional simulation case, here entire curves in some of the slices are specified by an expert. We first demonstrate a sampling approach based on our standard 2D curve sampling algorithm for the cases where there is a single unknown slice. For multiple contiguous unknown slices, we create methods based on local Metropolis-Hastings and embedded HMM techniques. We generalize this model to allow for expert segmentations in planes which are orthogonal to the slices in the Markov chain.

Chapter 7: Conclusion

This chapter concludes the thesis by summarizing the major contributions of our work. We also discuss a number of avenues for future research based on the curve sampling ideas generated by this thesis.

Chapter 2

Background

IN this chapter, we cover previous work that is important to understand the results in this thesis and to place our contributions in context. In Sec. 2.1, we begin with an introduction to MCMC methods for generating samples from arbitrary distributions. We then introduce curve evolution methods for image segmentation and specific algorithmic implementations such as level sets in Sec. 2.2. We conclude with Sec. 2.3 which discusses various statistical methods associated with segmentation such as statistical shape models and sampling-based segmentation techniques.

■ 2.1 Markov-Chain Monte Carlo Sampling

MCMC methods are a class of techniques for generating samples from distributions which are difficult or impossible to sample from directly. In general, low-dimensional distributions can be sampled using techniques such as matching cumulative distribution functions [84]. In higher dimensions, this becomes infeasible and iterative sampling techniques such as importance sampling, rejection sampling, or MCMC must be applied [89]. MCMC sampling is used in a variety of contexts including system simulation, model verification, Monte Carlo integration, optimization, and inference [9]. The two most widely used variants are Gibbs sampling [32] and Metropolis-Hastings methods [44, 71].

MCMC methods transform the problem of sampling from a target probability density function (pdf) $\pi(\boldsymbol{x})$ (with $\boldsymbol{x} \in \mathbb{R}^n$)[1] to that of iteratively generating a large number of samples from a *proposal distribution* $\mathsf{q}(\boldsymbol{x}^{(t)} \mid \boldsymbol{x}^{(t-1)})$ (from which it should be easy to sample). The samples from the proposal distribution are accepted or rejected in such a way so as to guarantee that this iterative process asymptotically produces samples from π.

[1] The MCMC formulation works identically for posterior distributions $\pi(\boldsymbol{x}) = \mathrm{p}(\boldsymbol{x}|\boldsymbol{z})$ where the distribution is conditioned on some observed data $\mathbf{z}$.

To accomplish this, a homogeneous Markov chain is constructed whose stationary distribution is π. If a chain has transition probability $\mathrm{T}(\boldsymbol{z}|\boldsymbol{x})$, a distribution $\mathrm{p}(\boldsymbol{x})$ is called the stationary distribution of the chain if:

$$\mathrm{p}(\boldsymbol{x}) = \int \mathrm{p}(\boldsymbol{z})\mathrm{T}(\boldsymbol{x}|\boldsymbol{z})\mathrm{d}\boldsymbol{z} \ . \tag{2.1}$$

This condition means that if $\mathbf{x} \sim \mathrm{p}(\,\cdot\,)$, transitioning a step in the chain leaves the distribution of $\mathbf{x}$ unchanged. If the chain is also *ergodic*[2], then simulating the chain will result in samples that asymptotically converge to samples from $\pi(\boldsymbol{x})$ from any initial distribution $\mathrm{p}_0(\boldsymbol{x})$ (see, *e.g.*, [9, 31, 76]). Note that this method only guarantees asymptotic convergence. For certain chains, a technique known as perfect sampling can be employed to know when the sampling procedure has converged [24], but in general, assessment of convergence or *mixing* of the chain is an open problem.

Typically the state transition of the chain is defined as the composition of two operations. At a given time t, a *candidate sample* $\boldsymbol{y}^{(t)}$ is first drawn from the proposal distribution $\mathtt{q}(\boldsymbol{y} \,|\, \boldsymbol{x}^{(t-1)})$. Then that candidate is accepted with probability defined by the *acceptance function* $\mathtt{a}(\boldsymbol{y}^{(t)} \,|\, \boldsymbol{x}^{(t-1)})$ where $\boldsymbol{x}^{(t)} = \boldsymbol{y}^{(t)}$ if the candidate is accepted and $\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)}$ if the candidate is rejected. We can then write the transition probability distribution as the product of the probability of generating the candidate and the probability of accepting the candidate:

$$\mathrm{T}(\boldsymbol{x}^{(t)} \,|\, \boldsymbol{x}^{(t-1)}) = \mathtt{q}(\boldsymbol{x}^{(t)} \,|\, \boldsymbol{x}^{(t-1)})\mathtt{a}(\boldsymbol{x}^{(t)} \,|\, \boldsymbol{x}^{(t-1)}) \ . \tag{2.2}$$

All MCMC methods share this framework and only differ in how $\mathtt{q}$ and $\mathtt{a}$ are specified.

Rather than directly showing that $\mathrm{p}(\boldsymbol{x})$ is the stationary distribution of the chain, it is often easier to show a stronger condition known as *detailed balance*:

$$\mathrm{p}(\boldsymbol{x})\mathrm{T}(\boldsymbol{z}|\boldsymbol{x}) = \mathrm{p}(\boldsymbol{z})\mathrm{T}(\boldsymbol{x}|\boldsymbol{z}) \ . \tag{2.3}$$

Detailed balance requires that the probability of being in a state $\boldsymbol{x}$ and transitioning to a state $\boldsymbol{z}$ is the same as the probability of being in $\boldsymbol{z}$ and transitioning to $\boldsymbol{x}$. A Markov chain satisfying this condition is called *reversible*, and detailed balance implies that p is the stationary distribution of the Markov chain. To see why this is so, we can integrate

[2] A chain is ergodic if it is aperiodic and *irreducible* (*i.e.*, starting from any state $\boldsymbol{x}_0$, there is non-zero probability to transition to any other $\boldsymbol{x} \in \mathbb{R}^n$ by simulating the chain).

(2.3) with respect to $\boldsymbol{z}$ to show that (2.1) is satisfied:

$$\begin{aligned}\int \mathrm{p}(\boldsymbol{z})\mathrm{T}(\boldsymbol{x}|\boldsymbol{z})\mathrm{d}\boldsymbol{z} &= \int \mathrm{p}(\boldsymbol{x})\mathrm{T}(\boldsymbol{z}|\boldsymbol{x})\mathrm{d}\boldsymbol{z} \\ &= \mathrm{p}(\boldsymbol{x}) \int \mathrm{T}(\boldsymbol{z}|\boldsymbol{x})\mathrm{d}\boldsymbol{z} \\ &= \mathrm{p}(\boldsymbol{x}) \ . \end{aligned} \tag{2.4}$$

■ 2.1.1 Metropolis and Metropolis-Hastings

Metropolis *et al.* [71] developed the first MCMC algorithm commonly referred to now as the Metropolis algorithm. This algorithm begins with an initial iterate $\boldsymbol{x}^{(0)}$ (which can be deterministic or drawn from some probability distribution). At time t, we draw a candidate sample from an arbitrary symmetric proposal distribution $\boldsymbol{y}^{(t)} \sim \mathtt{q}(\boldsymbol{y} \,|\, \boldsymbol{x}^{(t-1)})$. A distribution is symmetric if $\mathtt{q}(\boldsymbol{z}|\boldsymbol{x}) = \mathtt{q}(\boldsymbol{x}|\boldsymbol{z}) \,\forall\, \boldsymbol{x}, \boldsymbol{z}$. If $\pi(\boldsymbol{y}^{(t)})/\pi(\boldsymbol{x}^{(t-1)}) > 1$, we accept $\boldsymbol{y}^{(t)}$. Otherwise we accept it with probability $\pi(\boldsymbol{y}^{(t)})/\pi(\boldsymbol{x}^{(t-1)})$. Therefore the acceptance function is

$$\mathtt{a}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) = \min(\pi(\boldsymbol{y}^{(t)})/\pi(\boldsymbol{x}^{(t-1)}), 1) \ . \tag{2.5}$$

Another way to say this is that we accept any sample which increases the probability under the target distribution π, and we are more likely to reject the candidate sample the smaller its probability is relative to the previous iterate's probability. Note that if we were to change the acceptance rule to only accept values which increase $\pi(\boldsymbol{x}^{(t)})$, we would only be doing a stochastic maximization of the probability (which would find local maxima but not sample from π). If instead we were to accept all samples, we would get samples from some probability distribution (which would be the stationary distribution of the Markov chain with transition probability equal to $\mathtt{q}(\,\cdot \,|\, \cdot\,)$) which, in general, is not $\pi(\boldsymbol{x})$.

Hastings [44] later generalized the Metropolis algorithm for the case when $\mathtt{q}$ is not symmetric. This is now known as the Metropolis-Hastings algorithm. After drawing a candidate sample, we evaluate the Hastings ratio

$$\eta(\boldsymbol{y}^{(t)} \,|\, \boldsymbol{x}^{(t-1)}) = \frac{\pi(\boldsymbol{y}^{(t)})}{\pi(\boldsymbol{x}^{(t-1)})} \cdot \frac{\mathtt{q}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)})}{\mathtt{q}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)})} \ . \tag{2.6}$$

We accept $\boldsymbol{y}^{(t)}$ with probability $\min(\eta(\boldsymbol{y}^{(t)} \,|\, \boldsymbol{x}^{(t-1)}), 1)$. The only difference between the Metropolis and Metropolis-Hastings acceptance functions is the presence of the

additional terms involving q in (2.6). These can be viewed as a correction factor to remove the bias imparted by the proposal distribution (*i.e.*, if a sample is very likely to be generated by $\mathsf{q}(\cdot \,|\, \cdot)$, we need to compensate for that by reducing the probability that we will accept it).

The advantage of having an asymmetric proposal distribution is that we can create biased distributions that are more likely to move toward higher probability areas of the domain of π. This can increase the rate at which we accept candidate samples and thus speed up the mixing time of our chain. In general, the more similar q is to π, the faster the chain will mix. In the extreme case where $\mathsf{q}(\cdot \,|\, \boldsymbol{x}^{(t-1)}) = \pi(\,\cdot\,)$, the sampling procedure will converge after one iteration.

It is straightforward to show that Metropolis-Hastings satisfies detailed balance. Assume we have $\boldsymbol{x}^{(t-1)}$ and $\boldsymbol{y}^{(t)}$ such that $\eta(\boldsymbol{y}^{(t)} \,|\, \boldsymbol{x}^{(t-1)}) < 1$. Then we accept $\boldsymbol{y}^{(t)}$ with probability $\mathsf{a}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) = \eta(\boldsymbol{y}^{(t)} \,|\, \boldsymbol{x}^{(t-1)})$. For the reverse transition from $\boldsymbol{y}^{(t)}$ to $\boldsymbol{x}^{(t-1)}$, we accept $\boldsymbol{x}^{(t-1)}$ with probability $\mathsf{a}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)}) = 1$. Thus we see that:

$$\begin{aligned}
\pi(\boldsymbol{x}^{(t-1)})\mathrm{T}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) &= \pi(\boldsymbol{x}^{(t-1)})\mathsf{q}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)})\frac{\pi(\boldsymbol{y}^{(t)})\mathsf{q}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)})}{\pi(\boldsymbol{x}^{(t-1)})\mathsf{q}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)})} \\
&= \pi(\boldsymbol{y}^{(t)})\mathsf{q}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)}) \cdot 1 \\
&= \pi(\boldsymbol{y}^{(t)})\mathsf{q}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)})\mathsf{a}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)}) \\
&= \pi(\boldsymbol{y}^{(t)})\mathrm{T}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)}) \qquad (2.7)
\end{aligned}$$

Similar arguments hold when the situation is reversed and $\eta(\boldsymbol{y}^{(t)} \,|\, \boldsymbol{x}^{(t-1)}) \geq 1$. Therefore if the Markov chain is ergodic, Metropolis-Hastings-based algorithms will generate samples from π asymptotically. Note that the acceptance rule choice is not unique as $\tilde{\mathsf{a}}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) = \alpha\mathsf{a}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)})$ will also satisfy detailed balance for $0 < \alpha \leq 1$. However, these related acceptance rules will result in slower mixing times as more candidate samples will be rejected.

■ 2.1.2 Gibbs Sampling

The Gibbs sampler was introduced by Geman and Geman [32] for sampling from a Markov random field (MRF). The proposal distribution is defined so that $\boldsymbol{x}_S$, a subset of the complete random vector $\boldsymbol{x}$, is sampled from $\pi(\boldsymbol{x}_S \,|\, \boldsymbol{x}_{\setminus S})$ (where $\boldsymbol{x}_{\setminus S} = \boldsymbol{x} \setminus \boldsymbol{x}_S$). The set of variables $\boldsymbol{x}_{\setminus S}$ remains constant, and the candidate sample is always accepted. Note that the subset chosen can change deterministically (in a periodic fashion) or randomly with each iteration.

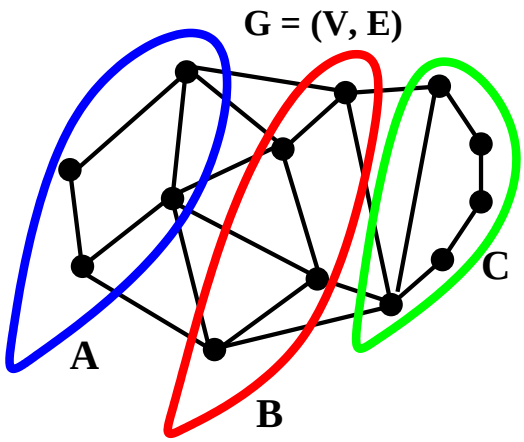


Figure 2.1. Markov random field example with A and C conditionally independent given B. Figure courtesy of D. Malioutov.

Gibbs sampling methods are most effective if it is possible to efficiently sample from $\pi(\boldsymbol{x}_S \,|\, \boldsymbol{x}_{\setminus S})$. This is generally true for models such as MRFs (also commonly referred to as *undirected graphical models*) which are a collection of random variables characterized by dependencies among subsets of those random variables [18, 48, 62]. With MRFs, a graph $G = (V, E)$ is defined, random variables are specified on the vertices V, and the edges E determine the local dependency structure.

For a given variable x_i, we define its neighbors $N(x_i)$ as the set of variables which have an edge between it and x_i (*i.e.*, $N(x_i) = \{x_j \in V \,|\, (x_i, x_j) \in E \text{ or } (x_j, x_i) \in E\}$). The graphical model then imposes conditional independence structure so that x_i is independent of all remaining variables $\boldsymbol{x} \setminus \{x_i, N(x_i)\}$ given $N(x_i)$. This means that $\mathrm{p}(x_i \,|\, \{\boldsymbol{x} \setminus x_i\}) = \mathrm{p}(x_i \,|\, N(x_i))$. More generally, consider the example in Fig. 2.1 in which two subsets of variables A and C are separated (with respect to the graph) by B. The sets of variables A and C are then conditionally independent given B.

For Gibbs sampling algorithms, this conditional independence property implies that $\pi(\boldsymbol{x}_S \,|\, \boldsymbol{x}_{\setminus S}) = \pi(\boldsymbol{x}_S \,|\, N(\boldsymbol{x}_S))$ for problems which have a MRF structure. When the neighborhood size is smaller (*i.e.*, there is a relatively sparse Markov structure for the random variables), we can expect $\pi(\boldsymbol{x}_S|\boldsymbol{x}_{\setminus S})$ to be simpler as $\boldsymbol{x}_S$ is directly dependent on fewer points. This tends to lead to more efficient Gibbs sampling algorithms.

One downside for approaches based on Metropolis-Hastings is that while they can capture global effects with appropriately constructed proposal distributions, for many

combinations of π and $\mathtt{q}$, very few candidate samples are actually accepted. This can result in slow and inefficient sampling algorithms because much of the computation is effectively wasted. In contrast, the Gibbs sampler accepts every candidate sample but uses very local proposal distributions. This can also be inefficient if there is long-range structure in the model. Which approach is superior is very much application dependent.

To show detailed balance, we note that $\boldsymbol{y}_{\setminus S}^{(t)} = \boldsymbol{x}_{\setminus S}^{(t-1)}$, $\mathtt{q}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) = \pi(\boldsymbol{y}_S^{(t)}|\boldsymbol{x}_{\setminus S}^{(t-1)})$ and $\mathtt{a}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) = 1$. Thus it is easy to show that:

$$\begin{aligned}
\pi(\boldsymbol{x}^{(t-1)})\mathrm{T}(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t-1)}) &= (\pi(\boldsymbol{x}_S^{(t-1)}|\boldsymbol{x}_{\setminus S}^{(t-1)})\pi(\boldsymbol{x}_{\setminus S}^{(t-1)}))(\pi(\boldsymbol{y}_S^{(t)}|\boldsymbol{x}_{\setminus S}^{(t-1)}) \cdot 1) \\
&= \pi(\boldsymbol{x}_S^{(t-1)}|\boldsymbol{y}_{\setminus S}^{(t)})\pi(\boldsymbol{y}_{\setminus S}^{(t)})\pi(\boldsymbol{y}_S^{(t)}|\boldsymbol{y}_{\setminus S}^{(t)}) \\
&= \pi(\boldsymbol{y}^{(t)})\mathrm{T}(\boldsymbol{x}^{(t-1)}|\boldsymbol{y}^{(t)}) \ . \qquad (2.8)
\end{aligned}$$

It is not actually necessary to be able to draw samples directly from $\pi(\boldsymbol{x}_S|\boldsymbol{x}_{\setminus S})$. A hybrid Gibbs/Metropolis-Hastings approach is possible in which we sample from a proposal distribution that only perturbs a subset of the variables [111]. We can then accept or reject that candidate according to the standard Metropolis-Hastings acceptance rule. The calculation of the target probability ratio can be simplified because

$$\begin{aligned}
\frac{\pi(\boldsymbol{y}^{(t)})}{\pi(\boldsymbol{x}^{(t-1)})} &= \frac{\pi(\boldsymbol{y}_S^{(t)}|\boldsymbol{y}_{\setminus S}^{(t)})\pi(\boldsymbol{y}_{\setminus S}^{(t)})}{\pi(\boldsymbol{x}_S^{(t-1)}|\boldsymbol{x}_{\setminus S}^{(t-1)})\pi(\boldsymbol{x}_{\setminus S}^{(t-1)})} \\
&= \frac{\pi(\boldsymbol{y}_S^{(t)}|\boldsymbol{y}_{\setminus S}^{(t)})\pi(\boldsymbol{x}_{\setminus S}^{(t-1)})}{\pi(\boldsymbol{x}_S^{(t-1)}|\boldsymbol{x}_{\setminus S}^{(t-1)})\pi(\boldsymbol{x}_{\setminus S}^{(t-1)})} \\
&= \frac{\pi(\boldsymbol{y}_S^{(t)}|\boldsymbol{y}_{\setminus S}^{(t)})}{\pi(\boldsymbol{x}_S^{(t-1)}|\boldsymbol{x}_{\setminus S}^{(t-1)})} \ . \qquad (2.9)
\end{aligned}$$

Detailed balance can be shown in a similar manner as in Sec. 2.1.1.

■ 2.1.3 Embedded HMMs

Neal *et al.* [77] developed an alternative iterative MCMC algorithm called the embedded hidden Markov model (HMM) algorithm. This method can efficiently capture global features for problems which have a chain structure to the state variables. For each iteration, the algorithm computes a number of independently-generated candidate samples for each state variable and simultaneously evaluates all possible combinations of the candidates using the *forward-backward* algorithm [94].

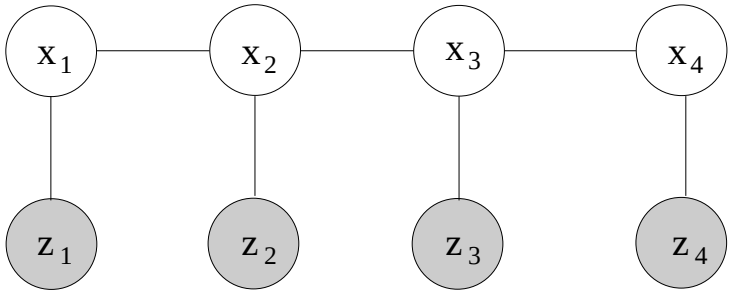


Figure 2.2. Graph for Neal *et al.* Markov model. $\boldsymbol{x}_i$ are the random variables and $\boldsymbol{z}_i$ are the observations.

Let there be a state vector $\mathbf{x} = [\mathsf{x}_1, \mathsf{x}_2, \ldots, \mathsf{x}_N]^{\mathrm{T}}$ and an observation vector $\mathbf{z} = [\mathsf{z}_1, \mathsf{z}_2, \ldots, \mathsf{z}_N]^{\mathrm{T}}$ which are continuously-valued random variables (*i.e.*, $\mathsf{x}_i, \mathsf{z}_i \in \mathbb{R}\ \forall\, i \in \{1, 2, \ldots, N\}$)[3]. This model has an overall posterior distribution $\pi(\boldsymbol{x} \,|\, \boldsymbol{z})$ with a conditional independence structure defined by the graphical model shown in Fig. 2.2. We view this model as a directed Markov chain (*i.e.*, the probability distribution is characterized in terms of state transitions) in this section but show how to adapt the embedded HMM method for undirected graphical models in Sec. 6.3.2. The posterior distribution can then be written as:

$$\pi(\boldsymbol{x} \,|\, \boldsymbol{z}) \propto \pi(x_1) \prod_{i=2}^{N} \pi(x_i \,|\, x_{i-1}) \prod_{i=1}^{N} \pi(z_i \,|\, x_i) \ . \tag{2.10}$$

To sample from the posterior, an HMM [88] is constructed for each iteration t. The HMM has the same graph structure as the original Markov model describing π (as in Fig. 2.2) with identical observations $\boldsymbol{z} = \{z_1, z_2, \ldots, z_N\}$ and hidden discrete variables $\{\chi_1^{(t)}, \chi_2^{(t)}, \ldots, \chi_N^{(t)}\}$ corresponding to the state variables $\{x_1, x_2, \ldots, x_N\}$.

To illustrate the general embedded HMM procedure, we show in Fig. 2.3 a 5-node Markov chain example. For each node i, a pool of candidate samples (referred to as a *constellation*) $\mathcal{C}_i^{(t)} = \{c_{i,1}^{(t)}, c_{i,2}^{(t)}, \ldots, c_{i,K}^{(t)}\}$ is created where each $c_{i,1}^{(t)} \in \mathbb{R}$ and K is a fixed parameter. The iterate value from the previous step must be included in the constellation, so let $c_{i,1}^{(t)} = x_i^{(t-1)}$. We indicate these points in Fig. 2.3 with square symbols (where the node index i is along the horizontal axis and the specific $c_{i,j}^{(t)}$ values are along the vertical axis). The remaining constellation values are independently sampled from

[3]Note that we will describe the method using scalar x_i and z_i, but it can be easily generalized to arbitrary dimensions.

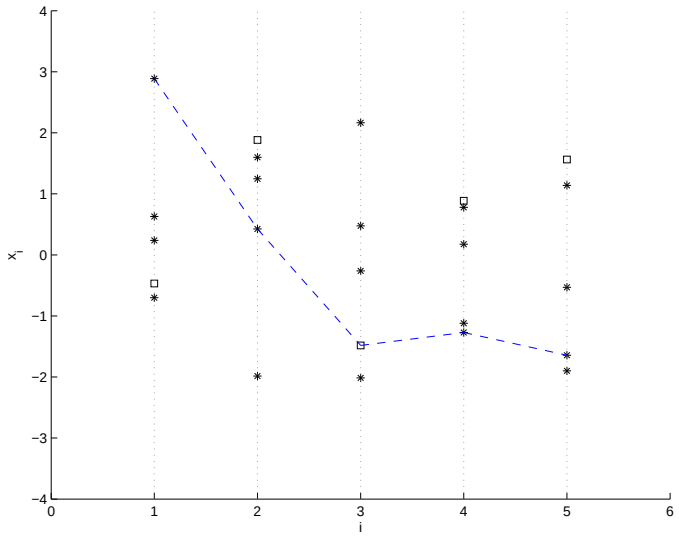


Figure 2.3. Example illustrating one iteration of the embedded HMM algorithm. The values of $x_i^{(t-1)}$ from the previous iteration are shown as squares, and the new constellation points $c_{i,j}^{(t)}$ sampled from the probability distribution ρ_i are displayed as asterisks. The new set of values $x_i^{(t)}$ sampled from the HMM are connected by the dashed blue line.

some probability distribution $\rho_i(c_i)$ (which can be different for each i)[4]. The support of $\prod_{i=1}^{N} \rho_i(c_i)$ must include the support of π. In Fig. 2.3, we generate constellations of size $K = 5$ and plot these new constellation points as asterisks.

The discrete HMM state variables χ_i take on values from $\{1, 2, \ldots, K\}$ and represent indices into the corresponding constellation of samples $\mathcal{C}_i^{(t)}$. At each time t, the embedded HMM algorithm draws a sample from $\mathrm{p}^{(t)}(\boldsymbol{\chi}^{(t)} \,|\, \boldsymbol{z})$, the posterior distribution of the HMM. The time superscript on $\mathrm{p}^{(t)}$ indicates that the probability distributions depend on the specific HMM and constellation at a given time t. The sampled $\chi_i^{(t)}$ values are then used to select particular constellation points for the updated state values:

$$x_i^{(t)} = c_{i,\chi_i^{(t)}}^{(t)} \quad . \tag{2.11}$$

We illustrate one particular sample path in Fig. 2.3 by connecting the selected constellation points with the dashed blue line.

We now describe how to construct and sample from $\mathrm{p}^{(t)}(\boldsymbol{\chi}^{(t)} \,|\, \boldsymbol{z})$. The posterior

[4]We discuss later in this section how this method can be generalized for the case when we are able to evaluate $\rho_i(c_i)$ but cannot sample from it directly.

probability of the HMM can be decomposed in a similar fashion to (2.10):

$$\mathrm{p}^{(t)}(\boldsymbol{\chi}^{(t)} \,|\, \boldsymbol{z}) \propto \mathrm{p}^{(t)}(\chi_1^{(t)}) \prod_{i=2}^{N} \mathrm{p}^{(t)}(\chi_i^{(t)} \,|\, \chi_{i-1}^{(t)}) \prod_{i=1}^{N} \mathrm{p}^{(t)}(z_i \,|\, \chi_i^{(t)}) \ . \tag{2.12}$$

We write the initial and transition probabilities of the HMM based on the target distribution π weighted by the constellation-generation probabilities ρ_i:

$$\mathrm{p}^{(t)}(\chi_1^{(t)}) \propto \frac{\pi(c^{(t)}_{1,\chi_1^{(t)}})}{\rho_1(c^{(t)}_{1,\chi_1^{(t)}})} \tag{2.13}$$

$$\mathrm{p}^{(t)}(\chi_i^{(t)} \,|\, \chi_{i-1}^{(t)}) \propto \frac{\pi(c^{(t)}_{i,\chi_i^{(t)}} \,|\, c^{(t)}_{i-1,\chi_{i-1}^{(t)}})}{\rho_i(c^{(t)}_{i,\chi_i^{(t)}})} \ . \tag{2.14}$$

Note that π is a probability density function whereas $\mathrm{p}^{(t)}(\chi_i^{(t)}|\chi_{i-1}^{(t)})$ is a probability mass function. The observation probabilities are the original model probabilities:

$$\mathrm{p}^{(t)}(z_i|\chi_i^{(t)}) = \pi(z_i|c^{(t)}_{i,\chi_i^{(t)}}) \ . \tag{2.15}$$

The overall posterior probability distribution of the HMM is then:

$$\mathrm{p}^{(t)}(\boldsymbol{\chi}^{(t)}|\boldsymbol{z}) \propto \frac{\pi(c^{(t)}_{1,\chi_1^{(t)}}) \prod_{i=2}^{N} \pi(c^{(t)}_{i,\chi_i^{(t)}} \,|\, c^{(t)}_{i-1,\chi_{i-1}^{(t)}}) \prod_{i=1}^{N} \pi(z_i \,|\, c^{(t)}_{i,\chi_i^{(t)}})}{\prod_{i=1}^{N} \rho_i(c^{(t)}_{\chi_i^{(t)}})} \ . \tag{2.16}$$

Note that $\mathrm{p}^{(t)}(\boldsymbol{\chi}^{(t)} \,|\, \boldsymbol{z})$ is then proportional to $\pi(\boldsymbol{x} \,|\, \boldsymbol{z})$ for $x_i = c^{(t)}_{i,\chi_i^{(t)}}$, and the probability of generating a specific constellation sequence is equal to $\prod_{i=1}^{N} \rho_i(c^{(t)}_{\chi_i^{(t)}})$. Neal *et al.* use these two facts to show that sampling from the sequence of $\mathrm{p}^{(t)}$ satisfies detailed balance with respect to π. Thus repeated iterations of the embedded HMM algorithm will result in asymptotic convergence to samples from π (assuming ergodicity).

The direct method to sample from (2.16) is to compute $\mathrm{p}^{(t)}(\boldsymbol{\chi}^{(t)}|\boldsymbol{z})$ for all combinations of $\{\chi_1^{(t)}, \chi_2^{(t)}, \ldots, \chi_N^{(t)}\}$. This is a costly $\mathcal{O}(K^N)$ operation. An alternative is to use the forward-backward algorithm which is able to perform this operation in $\mathcal{O}(NK^2)$. A well-known example of an optimization-based version of this algorithm is the Viterbi algorithm [4] which can be used to efficiently find the most likely state sequence of an HMM. We describe this algorithm in greater detail for an undirected graphical model in Appendix C.

For the case when we are unable to sample from $\rho_i(x_i)$ directly, the embedded HMM method can be adapted to instead construct a sub-chain of length K which satisfies detailed balance with respect to $\rho_i(c_i)$. This can be done, *e.g.*, by defining the transition probability in terms of generating candidate samples from a proposal distribution q_i and accepting or rejecting those samples according to the Metropolis-Hastings acceptance rule (with ρ_i as the target distribution).

For this sub-chain, $x_i^{(t-1)}$ is randomly placed as one of the constellation values by setting $c_{i,J}^{(t)} = x_i^{(t-1)}$ for J drawn uniformly from $\{1, 2, \ldots, K\}$. The remaining constellation values are generated by sampling the chain both forward from $k = J + 1$ to K (*i.e.*, from $\mathsf{q}_i(c_{i,k}^{(t)} \,|\, c_{i,k-1}^{(t)})\mathsf{a}_i(c_{i,k}^{(t)} \,|\, c_{i,k-1}^{(t)})$) and backward for $k = J - 1$ to 1 (*i.e.*, from $\mathsf{q}_i(c_{i,k}^{(t)} \,|\, c_{i,k+1}^{(t)})\mathsf{a}_i(c_{i,k}^{(t)} \,|\, c_{i,k+1}^{(t)})$). See [77] for a proof that using this constellation still results in a sampling algorithm which satisfies detailed balance.

Note that using this method results in duplicate state values in the constellation (*i.e.*, $c_{i,j}^{(t)} = c_{i,k}^{(t)}$ for some $j \neq k$) when a candidate sample is rejected. This is not an issue with the embedded HMM process because it samples over the indices χ_i into the constellation and does not directly involve the constellation values. From this perspective, even if $c_{i,j}^{(t)}$ and $c_{i,k}^{(t)}$ have equal values, they are distinct from the perspective of the embedded HMM algorithm. It is better, though, to avoid duplication of constellation values as much as possible to increase the diversity of the constellation values.

■ 2.2 Curve Evolution

Curve evolution methods are a class of algorithms that evolve curves to segment an image. The ideas developed here combine concepts from differential geometry [80] and

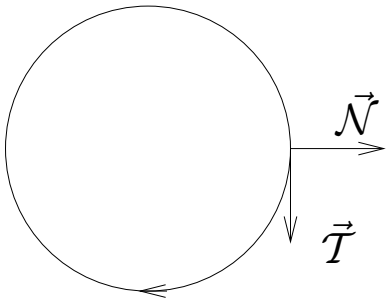


Figure 2.4. Negatively-oriented circle proceeds in a clockwise direction. Outward normal points away from the interior of the circle.

partial differential equations (PDEs) [92,96,106]. Let $\vec{C} : [0,1] \to \mathbb{R}^2$ be a simple closed curve parameterized from $p = 0$ to $p = 1$ ($\vec{C}$ is closed if $\vec{C}(0) = \vec{C}(1)$ and simple if it has no self-intersections). Here we are dealing with planar curves, but the theory extends well to embedded hyper-surfaces of codimension 1 in higher dimensions. Generally we will assume that the curve has negative orientation. This means that as we trace along the boundary of the curve with increasing p, the interior of the curve is to the right. A circle with a clockwise orientation is a negatively-oriented curve as shown in Fig. 2.4.

The goal then is to evolve the curve $\vec{C}$ so that it moves toward a desirable configuration through the use of a PDE. This PDE is often derived as the gradient descent of an energy functional using the calculus of variations [100]. We introduce an artificial time variable t that tracks the gradient descent evolution and write a time evolution equation for $\vec{C}$:

$$\frac{\partial \vec{C}}{\partial t}(p) = \vec{F}(p) \tag{2.17}$$

for $p \in [0,1]$. One can also write this time evolution equation in terms of a different orthogonal basis [101], the tangential component and the normal component of the curve. The tangent to the curve is

$$\vec{\mathcal{T}}(p) = \frac{\frac{\mathrm{d}\vec{C}}{\mathrm{d}p}}{\|\frac{\mathrm{d}\vec{C}}{\mathrm{d}p}\|} \tag{2.18}$$

and the normal component is the tangent rotated 90 degrees:

$$\vec{\mathcal{N}}(p) = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \vec{\mathcal{T}}(p) \ . \tag{2.19}$$

Note that for a negatively-oriented curve, $\vec{\mathcal{N}}$ is the outward normal to the curve. Then we can rewrite (2.17) as

$$\frac{\partial \vec{C}}{\partial t}(p) = f(p)\vec{\mathcal{N}}(p) + g(p)\vec{\mathcal{T}}(p) \tag{2.20}$$

with $f(p) = \left\langle \vec{F}, \vec{\mathcal{N}} \right\rangle$ and $g(p) = \left\langle \vec{F}, \vec{\mathcal{T}} \right\rangle$. The tangential component simply reparameterizes the curve, so we can express the same curve evolution solely in terms of the normal component:

$$\frac{\partial \vec{C}}{\partial t}(p) = f(p)\vec{\mathcal{N}}(p) \ . \tag{2.21}$$

The function $f(p)$ is often referred to as the force function. When the curve flow is composed only of the normal component, it is said to be a geometric PDE as the flow does not depend on the specific parameterization of the curve, only its geometry.

■ 2.2.1 Data-independent Curve Flows

Early work in curve evolution was concerned mostly with the theoretical behavior of certain geometric PDEs. The two primary flows of interest were the Blum prairie fire model [8]:

$$\frac{\partial \vec{C}}{\partial t} = -\vec{\mathcal{N}} \tag{2.22}$$

and the geometric heat flow equation [30]:

$$\frac{\partial \vec{C}}{\partial t} = -\kappa_{\vec{C}} \vec{\mathcal{N}} \ , \tag{2.23}$$

where $\kappa_{\vec{C}}$ is the curvature function of the curve [80]. If $\vec{C}(p) = (x(p), y(p))$, and $'$ indicates a derivative with respect to p, then

$$\kappa_{\vec{C}}(p) = \frac{x'(p)y''(p) - y'(p)x''(p)}{(x'^2(p) + y'^2(p))^{3/2}} \ . \tag{2.24}$$

The Blum prairie fire model plays a large role in shape analysis [106]. The idea is that as the curve flows inwards, it develops shocks (self-intersections). These shocks yield what is known as the medial axis representation of a shape which is closely related to the skeleton description of a shape [21].

Equation (2.23) is sometimes referred to as the Euclidean curve shortening flow [36] because it is the variation associated with minimizing the curve length using the Euclidean distance:

$$\mathrm{E}(\vec{C}) = \oint_{\vec{C}} \mathrm{d}s \ . \tag{2.25}$$

Here $\mathrm{d}s$ is a differential arc length and is equal to $\|\vec{C}'(p)\|\mathrm{d}p$. Early theoretical work showed that the Euclidean curve shortening flow shrinks an arbitrary embedded planar curve to a point without any shocks [30, 36].

More recently, Sundaramoorthi and Yezzi [104] created a method called more-than-topology-preserving curve evolution. In this method, they devise the following energy functional:

$$E(\vec{C}) = \iint_{\vec{C}\times\vec{C}} \left(\frac{1}{\|\vec{C}(\hat{s}) - \vec{C}(s)\|} - \frac{1}{\mathrm{d}_{\vec{C}}(\hat{s}, s)} \right) \mathrm{d}\hat{s}\mathrm{d}s \tag{2.26}$$

where $\mathrm{d}_{\vec{C}}(\hat{s}, s)$ is the distance between $\vec{C}(\hat{s})$ and $\vec{C}(s)$ along the curve.

A physical analogue of this model is a situation where negative electric charge has been evenly distributed along the curve. The $1/\|\vec{C}(\hat{s}) - \vec{C}(s)\|$ term then corresponds

to the electric field strength induced by that charge. The $1/\mathrm{d}_{\vec{C}}(\hat{s}, s)$ term is introduced so that the integrand in (2.26) is well-behaved as $s \to \hat{s}$. This energy functional can be seen to favor curve configurations where points are far away from each other in terms of Euclidean distance unless they are also close in terms of geodesic distance along the curve. The energy goes to infinity as the curve approaches a self intersection.

Sundaramoorthi and Yezzi show that the gradient flow of (2.26) is:

$$\frac{\mathrm{d}\vec{C}}{\mathrm{d}t}(s) = \lim_{\epsilon \to 0^+} \left[\mathcal{E}_\epsilon(s) + (\mathcal{P}_\epsilon(s) - \log(L_{\vec{C}}/2\epsilon))\kappa_{\vec{C}}(s)\right] \vec{\mathcal{N}}_{\vec{C}}(s) \tag{2.27}$$

where

$$\mathcal{E}_\epsilon(s) = \int_{\vec{C} \setminus B_\epsilon(s)} \frac{\left\langle \vec{C}(s) - \vec{C}(\hat{s}), \vec{\mathcal{N}}_{\vec{C}}(s) \right\rangle}{\|\vec{C}(s) - \vec{C}(\hat{s})\|^3} \mathrm{d}\hat{s} \tag{2.28}$$

$$\mathcal{P}_\epsilon(s) = \int_{\vec{C} \setminus B_\epsilon(s)} \frac{1}{\|\vec{C}(s) - \vec{C}(\hat{s})\|} \mathrm{d}\hat{s} \ . \tag{2.29}$$

In these equations, $B_\epsilon(s)$ is the portion of the curve $\vec{C}$ that is within ϵ distance (along the curve) of s. If some s_1 and s_2 exist so that $\vec{C}(s_1)$ and $\vec{C}(s_2)$ are close in terms of Euclidean distance but far in terms of distance along the curve, the $\mathcal{E}_\epsilon$ term in (2.28) will be large and in a direction which forces $\vec{C}(s_1)$ and $\vec{C}(s_2)$ apart.

■ 2.2.2 Data-driven Curve Evolution

Starting from the basic curve evolution framework, various approaches have evolved to use these techniques in image segmentation algorithms. Kass *et al.* [52] introduced a data-driven curve evolution framework known as *snakes*. This method attempts to segment an image by attracting the boundary to edges while also employing regularization terms that penalize the first and second derivatives of $\vec{C}$ to ensure smoothness. Caselles *et al.* [10] reasoned that the geometric heat equation was a reasonable starting point for a data-driven curve flow if an appropriate distance metric were chosen. The metric used in (2.25) is the Euclidean metric which penalizes all curve lengths equally. One common image model used for segmentation is that boundaries tend to occur on edges in the image (locations with large intensity gradients), so a reasonable distance metric is to impose shorter lengths on parts of the curve that are on edges in the image. Caselles *et al.* then proposed a modified energy functional with a weighted differential

arc length $\phi(s)\mathrm{d}s$:

$$\mathrm{E}(\vec{C}) = \oint_{\vec{C}} \phi(I(\vec{C}(s)))\mathrm{d}s \tag{2.30}$$

where $I : \Omega \to \mathbb{R}$ is the image. They called this method *geodesic active contours* because the minimum of this energy functional is the curve that minimizes the length of the curve in the new distance metric. The gradient flow for this energy functional is

$$\frac{\partial \vec{C}}{\partial t} = \phi(I)\kappa_{\vec{C}}\vec{\mathcal{N}} - (\nabla\phi \cdot \vec{\mathcal{N}})\vec{\mathcal{N}} \ . \tag{2.31}$$

For the ϕ function, they chose:

$$\phi(I(\vec{C}(s))) = \frac{1}{\|1 + \nabla(h \star I)(\vec{C}(s))\|} \tag{2.32}$$

where h is a smoothing kernel. Thus ϕ is small in regions in which there is a large image gradient and (2.30) is minimized when $\vec{C}$ is located along edges.

More recently, a new class of curve evolution methods known as region-based curve evolutions have become popular. With these techniques, region statistics rather than edge statistics are used to drive the evolution. In practice, these methods tend to be much more robust to noise as they use information from a larger number of pixels [126]. One example of a region-based curve evolution algorithm is the reformulation of the famous Mumford-Shah energy functional [74] by Tsai *et al.* [113]:

$$\mathrm{E}(\vec{C}, g) = \iint_{\Omega} (I - g)^2 \mathrm{d}\boldsymbol{x} + \beta \iint_{\Omega\setminus\vec{C}} \|\nabla g\|^2 \mathrm{d}\boldsymbol{x} + \alpha \oint_{\vec{C}} \mathrm{d}s \ . \tag{2.33}$$

Here we are trying to find a piecewise smooth image g which closely approximates I. Discontinuities in g are only allowed to occur on $\vec{C}$. The Chan-Vese energy functional [11] can be viewed as the piecewise-constant case of Mumford-Shah:

$$\mathrm{E}(\vec{C}) = \iint_{\mathcal{R}_{\vec{C}}} (I - m_1)^2 \mathrm{d}\boldsymbol{x} + \iint_{\mathcal{R}^c_{\vec{C}}} (I - m_0)^2 \mathrm{d}\boldsymbol{x} + \alpha \oint_{\vec{C}} \mathrm{d}s \tag{2.34}$$

$\mathcal{R}_{\vec{C}}$ denotes the region inside the curve, $\mathcal{R}^c_{\vec{C}}$ denotes the region outside the curve, and m_0 and m_1 are the respective mean intensities. One perspective is to view energy functionals as negative log probabilities. In the Chan-Vese case, $\mathrm{E}(\vec{C})$ then corresponds to a model with constant means within regions with additive white Gaussian noise.

There are numerous other energy functionals available including the region competition method of Zhu and Yuille [130], maximizing separation between the means and variances inside and outside the curve [113, 126], or maximizing mutual information between image intensities and pixel labels [58, 114].

■ 2.2.3 Global Optimizers and Statistical Methods

Many recent approaches in image segmentation have incorporated advanced statistical and numerical techniques developed in other signal processing and optimization contexts. These include stochastic optimization and Sobolev methods to find global minima, particle filtering methods for tracking and simulated annealing, and MCMC approaches for shape sampling.

Stochastic optimization methods are a class of algorithms used to help avoid local optima. These typically take on two general forms: simulated annealing methods and noisy gradient methods. Simulated annealing is based on Metropolis-like MCMC methods except the target distribution p has a temperature parameter T. Let π be the true target distribution and $\mathrm{p}(\boldsymbol{x};T) = \pi(\boldsymbol{x})^{1/T}$. The temperature T starts with a large value and is slowly decreased (much as annealing is used to reduce defects in metallurgy). For high temperatures the target distribution is flat, and it becomes sharper as T decreases. For the Metropolis algorithm, the candidate sample is accepted with probability

$$\mathrm{a}(\boldsymbol{y}|\boldsymbol{x}) = \min\left(1, \left[\frac{\pi(\boldsymbol{y})}{\pi(\boldsymbol{x})}\right]^{1/T}\right) \quad . \tag{2.35}$$

We can see that when the candidate sample increases the probability, we still always accept it. The difference is that the larger T is, the more likely we are to accept a candidate that decreases the probability. This allows the simulated annealing method to search the state space more thoroughly. As the temperature is slowly decreased, the target distribution $\mathrm{p}(\boldsymbol{x};T)$ becomes more peaked, and the algorithm tends only to accept candidate samples which increase the target probability. An example of this is the work by De Bruijne and Nielsen [22] who use simulated annealing along with a point distribution model to segment a variety of medical images.

Another global optimization approach was developed by Juan *et al.* [50]. The authors convert a standard geometric PDE into a stochastic PDE by adding a random walk component. They use the viscosity solutions of Lions and Souganidis [67] to prove that their method converges to the global minimum and demonstrate superior results on complicated images compared with standard gradient-descent methods.

A global optimization technique not based on stochastic methods is the Sobolev active contour method developed by Sundaramoorthi *et al.* [105]. Sobolev metrics in the space of curves are used to generate gradient flows with damped high frequency components. Numerically this results in smoothed versions of the gradient flows ob-

tained using a non-Sobolev metric such as L2. The Sobolev approach therefore generate gradient steps which first perform low-frequency global moves; as the solution moves close to the global minimum, higher-frequency refinements are automatically made.

Particle filtering is a statistical technique often used in tracking applications as an alternative to the Kalman filter for non-Gaussian and nonlinear models. Sun *et al.* [103] use particle filtering to track the movement of the left ventricle in cardiac magnetic resonance (MR). They use a shape representation based on SDFs and use PCA to characterize the probability distributions between slices at one time and the next. Florin *et al.* [29] use a particle filtering approach for blood vessel segmentation. Here they are transitioning between image slices in space rather than snapshots in time. This approach bears some similarity to the hybrid 2D/3D algorithm we present in Chap. 6, though the authors use particle filtering to find global maxima instead of generating samples, and they only track the center of the blood vessel in each slice rather than a complete contour.

Sampling was initially introduced for segmentation problems by Geman and Geman [32]. They used binary segmentation labels and an Ising model to encourage contiguous regions of pixels of the same type. They use a second MRF that lives on the edges of the first MRF to turn the edges on and off (similar to how the Mumford-Shah energy functional uses the curve to specify where the regularization term should be turned off). They then draw samples from the distribution using Gibbs sampling.

In [129], Zhu performed Gibbs sampling on what he termed Gestalt representations of shapes. Gestalt psychology studies how humans group parts into whole objects. Among these representations are line segments and a graph-like structure which represent higher-order connectivity between points due to symmetry. All of his shape representations are discrete which require some explicit notion of correspondence. Tu and Zhu [118] also introduce a MCMC method based on Zhu and Yuille's region competition framework [130] that is most similar to our work. We will explore these similarities and differences more thoroughly in Sec. 3.2.5.

■ 2.2.4 Numerical Implementations

Two main methods are used for numerically implementing active contour models. The discrete approach developed by Kass *et al.* [52] represents the curve with N marker points $\{s_1, s_2, \ldots, s_N\}$ and interpolates between those points to determine the remainder of the curve. To do the time evolution, one simply implements the evolution equation

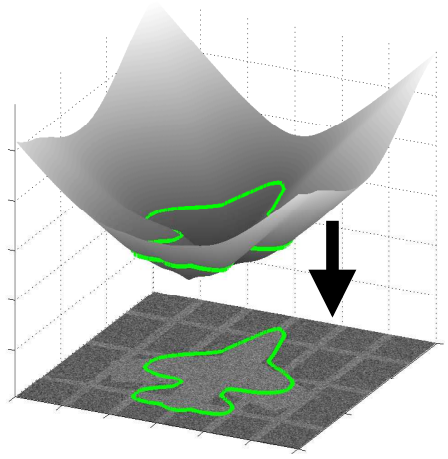


Figure 2.5. Level set surface used to represent an airplane shape. The zeroth level set is the contour location. Figure courtesy A. Tsai.

(2.21) with discrete time steps Δt and finite differences:

$$\vec{C}^{(t+\Delta t)}(s_n) = \vec{C}^{(t)}(s_n) + \Delta t \frac{\partial \vec{C}}{\partial t}^{(t)}(s_n) \; \forall \; n \in \{1, 2, \ldots, N\} \; . \tag{2.36}$$

This method is referred to as a Lagrangian method. Lagrangian representations are those in which specific points on the curve are tracked, so a fixed coordinate system for the variables does not exist. Eulerian representations are those in which the curve is defined using a function that is dense in the image domain.

On the surface, using snakes seems like a reasonable thing to do, but this approach actually causes many difficulties in practice [97]. In general, the time steps in the curve evolution must be kept small (due to the finite spatial sampling), and, even then, the curve needs to be regularly reparameterized to prevent points from clumping together in regions of high curvature. Another major drawback is difficulty in handling changes in topology.

The Eulerian level set framework introduced by Sethian and Osher [82,97] addresses many of these concerns. With level set methods, the curve is embedded as the zeroth level set of a surface $\Psi : \Omega \rightarrow \mathbb{R}$. An example of this process is shown in Fig. 2.5 where a surface is constructed whose zeroth level set is an airplane shape. Note that the choice of level set function is not unique for a given $\vec{C}$ as many Ψ can have the same zeroth

level set (*e.g.*, Ψ and 2Ψ).

To implement curve evolution approaches using level sets, the time evolution equation for $\vec{C}$ needs to be be transformed into a time evolution equation for Ψ. For the zeroth level set of Ψ to track $\vec{C}$, we need:

$$\Psi(\vec{C}(s)) = 0 \ . \tag{2.37}$$

By differentiating (2.37) with respect to t, we obtain

$$\frac{\partial \Psi}{\partial t} = -\frac{\partial \vec{C}}{\partial t} \cdot \nabla \Psi \ . \tag{2.38}$$

Various quantities that are important in curve evolution can be computed directly from Ψ [113]. For instance, the normal to the curve is

$$\vec{\mathcal{N}} = \frac{\nabla \Psi}{\|\nabla \Psi\|} \tag{2.39}$$

and the curvature is

$$\kappa_{\vec{C}} = \mathrm{div} \vec{\mathcal{N}} \ . \tag{2.40}$$

We can combine (2.21), (2.38), and (2.39) to obtain:

$$\frac{\partial \Psi}{\partial t} = -f \|\nabla \Psi\| \ . \tag{2.41}$$

Note that (2.41) is only defined on the zeroth level set as f only exists on the curve. To be valid, the values of f need to be extrapolated to the rest of the level set in a technique known as *velocity extension* [96]. Let φ be the velocity f extended to all of Ω. To be consistent with the original geometric PDE, values of $\varphi(\boldsymbol{x})$ are determined by starting at a point on the curve $\vec{C}(p)$ (which is also a point on the zeroth level set) and tracing along the normal to move between level sets (both inward and outward). All points on those traces have $\varphi(\boldsymbol{x})$ set to $f(p)$. In practice, this is a difficult procedure to implement, so an approximation is used where $\varphi(\boldsymbol{x}_0)$ takes on the value of $f(p_0)$ with $p_0 = \arg\min_p \|\vec{C}(p) - \boldsymbol{x}_0\|$.

One issue with level set methods is that a problem that was previously defined on the curve (1D) has now been transformed into a problem on all of Ω (2D) with an inherent increase in computational complexity. An approach to mitigate this effect is the narrowband method [96] which only updates the points of Ψ that are close to the zeroth level set. There is additional overhead in keeping track of which points are in the narrowband, but generally these approaches are able to make the computational complexity linear in the length of the curve.

■ 2.3 Statistical Shape Segmentation

The field of image segmentation has evolved to use a number of statistical techniques developed in other related signal-processing fields to provide robustness to noise and poor signal quality and to help the algorithms find global optima. These techniques include statistical shape priors, stochastic optimization, and various sampling-based approaches.

In all practical curve evolution algorithms, there has always been some regularization of the curve to provide robustness to noise [112]. For example, in many algorithms a curve length penalty is used as a regularization term. The rationale is that smooth curves are preferred over jagged curves, and smooth curves tend to be shorter. From an estimation point of view, this term can be seen as a prior on the space of curves. A curve length penalty has proven to be a useful prior, but there are many instances in which the quality of the data is so poor that more prior information needs to be incorporated into the model.

Another major issue with curve evolution methods is that the optimization technique used is usually gradient descent, but the energy functionals being optimized are not convex. This means that any local optima that is found using gradient descent has no guarantees as to how close it is to the global optimum. Stochastic optimization and sampling methods are two ways to attempt to avoid local minima.

■ 2.3.1 Point Distribution Models

The snake representation of Kass *et al.* [52] is finite dimensional with a fixed number of control points. If there are N marker points for a 2D curve, then all snakes are contained in $\mathbb{R}^{2N}$. Kendall originally explored the notion of shape spaces for point distribution models [53]. Each shape can be viewed as a point in a high-dimensional Riemannian space, and due to constraints on valid shapes (*e.g.*, closed curves), the shapes exist on a subset of that space known as a manifold.

Probably the most famous shape-based segmentation model is the active shape model developed by Cootes and Taylor (and many other collaborators) [14, 15]. This is a Lagrangian approach similar to snakes that defines a curve using N marker points along the boundary. Cootes and Taylor define a probability distribution for registered marker points that is assumed to be Gaussian and its behavior is characterized using principal components analysis (PCA). Given a set of K training vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K$

and a specified dimensionality L, PCA extracts the L-dimensional linear subspace of the data that contains the most variance [5]. For linear spaces, this is equivalent to performing a singular value decomposition (SVD) [45, 101] and keeping the L largest eigenvectors. The largest difficulty with active shape models is that correspondence between points must be specified explicitly. For some 2D examples, there may be obvious features to use as marker points. In general, though, it is very difficult to deal with the correspondence problem. Additional limitations relate to the discretized curve and are inherent in snakes-based methods: no topological change and difficulty with areas of high curvature.

■ 2.3.2 Dense Shape Representations

Grenander and Miller and collaborators [39] developed a segmentation framework they dubbed computational anatomy. This approach represents shapes as deformable templates (also referred to as atlases) and families of diffeomorphisms that are used to transform the templates. To segment an image, they find the diffeomorphism that best aligns the template with the observation image (according to some cost criterion). The family of diffeomorphisms needs to be chosen to be rich enough to allow for a wide-enough range of variation, but not too large otherwise the computational burden becomes too heavy. The atlas-based approach has been applied by a number of other researchers such as Pohl *et al.* [86] and the active appearance model of Cootes and Taylor [13].

The atlas-based approaches can be viewed as using a binary shape representation (where a 0 indicates outside the region, and a 1 indicates inside the region). An alternative approach is to build a shape model using another shape representation such as the signed distance function (SDF) [64, 116]. The SDF is a special level set function Ψ that is zero on the curve and has slope of 1 almost everywhere:

$$\|\nabla\Psi(\boldsymbol{x})\| = 1 \text{ a.e.} \tag{2.42}$$

This equation is known as the Eikonal equation and can be solved quickly using fast marching methods [96].

The SDF is often used as the canonical level set function for a given curve as there is a one-to-one mapping between curves and SDFs. We can then represent each shape by an SDF which is a point in an infinite-dimensional Hilbert space (*e.g.*, L2). The set of valid shapes is a manifold embedded in the Hilbert space due to the constraints

at every point imposed by (2.42). Like Cootes and Taylor, Leventon *et al.* [64, 65] use PCA to extract principal modes of shape variation. Instead of a point-based model though, they use SDFs as an Eulerian shape representation. Eulerian methods avoid the correspondence issues that plague Lagrangian methods because every point in the image domain now carries information about the curve.

Leventon *et al.* take a set of K curves $\{\vec{C}_i\}_{i=1}^{K}$, align them[5], and compute the SDF Ψ_i for each curve $\vec{C}_i$. The mean level set function is then computed:

$$\bar{\Psi}(\boldsymbol{x}) = \frac{1}{K}\sum_{i=1}^{K}\Psi_i(\boldsymbol{x}) \tag{2.43}$$

and subtracted from each level set function:

$$\tilde{\Psi}_i(\boldsymbol{x}) = \Psi_i(\boldsymbol{x}) - \bar{\Psi}(\boldsymbol{x}) \ . \tag{2.44}$$

Each zero-mean level set $\tilde{\Psi}_i(\boldsymbol{x})$ is then sampled in space to form an $M \times N$ matrix $\tilde{\boldsymbol{\Psi}}_i$ which is then converted to an $MN \times 1$ vector $\boldsymbol{a}_i$ by vertically stacking its columns in order. These vectors then form the columns of an $MN \times K$ matrix:

$$\boldsymbol{A} = \left(\ \boldsymbol{a}_1 \ \middle| \ \boldsymbol{a}_2 \ \middle| \ \ldots \ \middle| \ \boldsymbol{a}_K \ \right) \ . \tag{2.45}$$

The SVD of $\boldsymbol{A}$ is:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}} \tag{2.46}$$

where $\boldsymbol{U}$ is an orthogonal $MN \times K$ matrix, $\boldsymbol{\Sigma}$ is a diagonal $K \times K$ matrix, and $\boldsymbol{V}$ is an orthogonal $K \times K$ matrix. Typically for level set PCA applications, $MN >> K$, so the most efficient way to compute the SVD is to first compute $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ and note that it has an eigenvalue decomposition in terms of $\boldsymbol{V}$ and $\boldsymbol{\Sigma}^2$:

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}} = \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^{\mathrm{T}} \ . \tag{2.47}$$

Therefore $\boldsymbol{V}$ and $\boldsymbol{\Sigma}^2$ can be obtained from $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ using standard numerical techniques [45], and we can compute $\boldsymbol{U}$ as:

$$\boldsymbol{U} = \boldsymbol{A}\boldsymbol{V}\boldsymbol{\Sigma}^{-1} \ . \tag{2.48}$$

If $\boldsymbol{\Sigma}$ has its entries sorted from largest to smallest, the L largest principal components can be obtained from the first L columns of $\boldsymbol{U}$ (denoted as $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_L$) and the corresponding eigenvalues from the first L diagonal entries of $\boldsymbol{\Sigma}$ (referred to as

[5]The choice of alignment method is problem-dependent.

$\sigma_1, \sigma_2, \ldots, \sigma_L$). We can convert each $\boldsymbol{u}_i$ back into functions $\varphi_i(\boldsymbol{x})$ on Ω and construct shapes in the lower-dimensional subspace spanned by the principal components as the zeroth level sets of:

$$\Psi(\boldsymbol{x}) = \bar{\Psi}(\boldsymbol{x}) + \sum_{i=1}^{L} \alpha_i \sigma_i \varphi_i(\boldsymbol{x}) \tag{2.49}$$

where the α_i are arbitrary weights on each eigenvector.

One issue associated with this approach is that PCA finds linear subspaces, but SDFs are a nonlinear shape representation (*e.g.*, adding two SDFs together does not result in another SDF). Additionally, there is a nonlinear map between SDFs and curves, so it is not always clear exactly how the combinations of the eigenvectors move the curve. This can result in the linear combinations of eigenvectors having zeroth level sets that produce very jagged curves, especially in applications in which there is a large amount of variability in the training set. When training examples are not as close to each other, the effect of the nonlinearity becomes more pronounced.

Nonetheless, this shape model has been successfully applied in practice to shape classification problems and difficult segmentation problems such as the prostate and brain subcortical structures. Golland *et al.* [35] use this shape model along with support vector machines [120] to discriminate between normal and diseased subcortical structures. Tsai *et al.* [114, 115] use a parameterized level-set function that is strictly enforced to remain in the subspace of the training examples. They then choose the linear combination of the eigenvectors that minimizes an energy functional. This simplifies the optimization problem to a low-dimensional search over weights on the PCA basis vectors. In [115], Tsai *et al.* develop an algorithm to perform binary segmentation on prostate images with a Chan-Vese data term. In [114], they describe an M-ary version of the problem that involves segmenting the rectum and obdurator muscles in addition to the prostate. In this work, the authors use an information-theoretic energy term similar to that found in [58]. The rectum and obdurator muscles are relatively easy to segment, and they help constrain the location of the true object of interest. Yang and Duncan [124] use a similar approach for brain segmentation, except they allow the level set to deviate from the PCA subspace and penalize the deviation (in an L2 sense).

■ 2.3.3 Shape Distances

One approach to building a shape pdf is through the use of shape distances and kernel density estimators. The Parzen density is a way of non-parametrically estimating a pdf

from training examples [5]. Say we have N samples $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ drawn independently from a distribution $\mathrm{p}(\boldsymbol{x})$. Then one estimate of p is:

$$\hat{\mathrm{p}}(\boldsymbol{x}) = \frac{1}{N} \sum_{n=1}^{N} \mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_n) \tag{2.50}$$

where K is referred to as the kernel. As $N \to \infty$, $\hat{\mathrm{p}} \to \mathrm{p} \star \mathrm{K}(\boldsymbol{x}, \mathbf{0})$.

A common choice for the kernel function K is a Gaussian kernel:

$$\mathrm{K}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{1}{Z} \exp(-\mathrm{d}^2(\boldsymbol{x}_1, \boldsymbol{x}_2)/2\sigma^2) \ .$$

Note that d need not be a metric, though it should be non-negative and the minimum value should occur for $\boldsymbol{x}_1 = \boldsymbol{x}_2$. In general, the kernel function must integrate to 1 and will have a smoothing effect in the domain of p to compensate for the lack of training examples. Usually there is also a smoothing parameter which must be chosen appropriately to provide adequate results (*e.g.*, the variance σ^2 for the Gaussian kernel).

Thus we see that one way of defining a pdf on shapes is simply to define a distance between shapes. For shape models, the shape manifold is a nonlinear subset of the embedding space. When comparing two shapes, one measure of similarity is the geodesic distance on the manifold [60]. This can be difficult to compute, so it is often easier to use a metric that is defined on the embedding space. For instance, a measure of distance between two shapes could be the L2 distance between their respective SDFs. If the two shapes are similar (meaning they have small geodesic distance between them), this is a reasonable approximation to the geodesic distance as manifolds are locally Euclidean. If two shapes are very dissimilar or the manifold has high curvature, then this can prove to be a poor approximation (much as linear approximations to functions are most accurate on a local scale).

This Parzen-based approach is illustrated by Kim *et al.* [57]. In that work, the authors use SDFs and shape metrics defined on the embedding space to segment handwritten digits and images with missing data. Klassen *et al.* [60] developed an approach that uses geodesic distances (but not Parzen density estimators) for segmentation. They use either angle functions or curvature functions as their curve representation. To compute the geodesic distance between shapes, they employ an iterative path-finding algorithm that traces the geodesic from one point on the manifold to the other. At each iteration, they move closer to the target point in the embedding space and project the resulting point back onto the manifold in a manner similar to projected gradient descent [3]. This

projection operation is tractable because there are only a small number of constraints on their representation functions that determine whether curves are valid or not. With most implicit shape representations, this approach is not directly applicable as the representations have constraints at nearly every point in the image domain (*e.g.*, for SDFs, $\|\nabla\Psi\| = 1$ a.e.).

Chapter 3

2D Curve Sampling Formulation

TRADITIONAL curve evolution methods are optimization-based and typically return a single local optimum. This provides little insight as to how close the result is to the global optimum or how confident one should be in the answer. Finding the global maximum of a non-concave posterior distribution requires exhaustive search in general, and even having the global optimum may not be ideal. For low signal-to-noise ratio (SNR) or ill-posed problems, there may be multiple answers which plausibly explain the data, or the global optimum may simply not be a good segmentation due to model error.

In this chapter we describe an algorithm to draw samples from a posterior distribution on the space of curves using non-parametric representations and Markov chain Monte Carlo (MCMC) methods. MCMC techniques are useful for situations when one wishes to draw samples from a distribution π, but it is not possible to do so directly. Sampling methods for image segmentation using MRFs have certainly been used extensively, pioneered by the work of Geman and Geman [32]. While MRFs are localized statistical descriptions, curve sampling is an inherently geometric process that enables us to work explicitly in the space of shapes. This allows us, *e.g.*, to encode global object characteristics and statistical properties of shape directly into the model.

We begin with an overview of our overall curve sampling algorithm in Sec. 3.1 and the major assumptions and approximations which are made to make the problem computationally feasible. Sec. 3.2 describes a choice for the proposal distribution $\mathtt{q}$ by constructing a method to sample from it using a smoothed Gaussian perturbation. We then show how to evaluate the proposal distribution in Sec. 3.3 in order to compute the Hastings ratio and ensure detailed balance. An extension to incorporate online parameter estimation is described in Sec. 3.4.

■ 3.1 Overall Algorithm

To construct a sampling algorithm based on MCMC methods, there are three things which must be specified: the target distribution $\pi(\vec{C})$, the proposal distribution $\mathtt{q}(\vec{\Gamma}|\vec{C})$, and the acceptance probability $\mathtt{a}(\vec{\Gamma}|\vec{C})$. In this chapter, we describe a 2D curve sampling approach using the Metropolis-Hastings acceptance rule (see Sec. 2.1.1). Metropolis-Hastings is a natural choice for many problems with globally-coupled state spaces as they do not easily lend themselves to Gibbs sampling approaches.

The choice of the target distribution is problem-based. Standard curve evolution algorithms have an energy functional which they seek to minimize. A common alternative is to view $E(\vec{C})$ as the negative log of a probability density, so we can obtain a probability distribution (up to a scale factor) by exponentiating the energy:

$$\pi(\vec{C}) \propto \exp(-E(\vec{C})) \ . \tag{3.1}$$

Note that different choices for the energy functional may be more appropriate for a sampling framework than an optimization framework, and vice versa. For instance, a curve length penalty is a standard regularization term for optimization-based curve evolution because it results in a simple gradient flow with a nice geometric interpretation. While derivative information can be used in our sampling approach, in general, it is not as important because we only must be able to evaluate π (and not its derivative), so we have more freedom to use energies that have hard-to-evaluate derivatives if they provide better segmentation models.

There are a number of considerations when designing a proposal distribution. The most important is that the choice of the proposal must result in an ergodic Markov chain. This is necessary for asymptotic convergence of the chain to the target distribution. Once that is established, it is useful if it is easy to sample from the proposal, as MCMC methods change the problem of sampling from π to one of drawing many samples from $\mathtt{q}$. It is also advantageous for speed of convergence if samples from $\mathtt{q}$ tend to be in higher-probability regions of π. This can result in faster mixing of the chain and fewer iterations needed before convergence.

Here we implicitly define the proposal distribution $\mathtt{q}$ by explicitly defining a method of sampling from it. A candidate sample is generated by adding correlated Gaussian perturbations to the normal of the curve from the previous iteration. We begin here with a high-level description of the overall sampling algorithm:

1. Set $\vec{C}^{(0)}$ to some initial value (deterministic or random). Set $t = 1$.

2. Generate candidate sample $\vec{\Gamma}^{(t)} \sim \mathtt{q}(\vec{\Gamma} \,|\, \vec{C}^{(t-1)})$ by first creating a Gaussian perturbation $f^{(t)}$. Applying this to the normal of the previous curve results in

$$\vec{\Gamma}^{(t)}(p) = \vec{C}^{(t-1)}(p) + f^{(t)}(p)\vec{\mathcal{N}}_{\vec{C}^{(t-1)}}(p)\delta t \tag{3.2}$$

   for some positive constant δt. [Sec. 3.2]

3. Compute Hastings ratio $\eta(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})$. This requires evaluation of the forward and reverse perturbation probabilities $\mathtt{q}(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})$ and $\mathtt{q}(\vec{C}^{(t-1)} \,|\, \vec{\Gamma}^{(t)})$ as well as the target distribution probabilities $\pi(\vec{C}^{(t-1)})$ and $\pi(\vec{\Gamma}^{(t)})$. [Sec. 3.3]

4. Accept or reject $\vec{\Gamma}^{(t)}$ with probability $\min(\eta(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)}), 1)$ to obtain the current iterate value $\vec{C}^{(t)}$.

5. Increment t and return to Step 2.

The formulation we present in this chapter generally follows the form of curve evolution methods in which the curve is viewed as being continuous in space and with a differential time step (as in the perturbation defined in (3.2)). Naturally, any actual implementation necessarily must be discrete in space and with a small, but finite, time step δt.

This raises the question of whether we should simply use a discrete formulation to begin with. The main difficulty with a discrete formulation is how to handle the issue of curve reparameterization. With marker-point representations, the points tend to cluster together in sections of high curvature over time [96]. Without reparameterization then, it is possible that all of the points end up in very small portions of the curve which leaves insufficient points to adequately represent the remainder of the curve.

Therefore we take the viewpoint that we are simulating a continuous curve, and at each time t we construct a particular discrete representation of the curve. For the sampling computation in Step 2, we approximate $f^{(t)}(p)$ with a Gaussian random vector $\boldsymbol{f}^{(t)}$ and implement the perturbation using a narrowband level set approach. The evaluation of $\mathtt{q}(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})$ and $\mathtt{q}(\vec{C}^{(t-1)} \,|\, \vec{\Gamma}^{(t)})$ is then based on these discrete Gaussian perturbations[1]. A similar viewpoint is that we have a continuous, correlated Gaussian

[1] One concern with approximating a continuous process with discrete computation (and performing approximations in general) is whether detailed balance in the chain is being preserved. While it is difficult to provide empirical evidence that this is true due to the very high-dimensional state spaces in which we are working, the results we have obtained generally appear to be satisfactory. One possibility is to consider the stochastic acceptance functions of Kennedy and Kuti [54] or Lin *et al.* [66]. These authors devised acceptance functions which can accommodate error in the probability computations as long as those errors are unbiased.

process which we are discretizing to obtain $\boldsymbol{f}^{(t)}$ [51, 78].

In addition to discrete spatial sampling, we must also employ finite time steps δt to implement the curve perturbations. One major issue with finite time steps is maintaining the stability of the curve evolution process. A time step which is too large may cause the perturbation to result in an invalid curve (*e.g.*, one with self-intersections), though standard results from entropy-based level set implementations can be used to determine an appropriate value of δt [96]. A finite δt also affects our sampling algorithm when evaluating the reverse proposal distribution probability $\mathtt{q}(\vec{C}^{(t-1)} \,|\, \vec{\Gamma})$. The method we construct to perform this task in Sec. 3.3 is correct up to $\mathcal{O}(\delta t)$.

Finally, there is one main limitation related to our choice of level sets for the numerical implementation. One advantage of level set methods over the snake-based approach of Kass *et al.* [52] is natural handling of topological change. Unfortunately, the computations in our sampling formulation require an explicit correspondence between the parameterizations of an iterate $\vec{C}$ and a perturbed version $\vec{\Gamma}$ which does not allow topological change (or at least must handle it as a special case, *e.g.*, using a jump-diffusion process [38, 118]). This means that care must to taken to prevent the curve from splitting or merging with other curves.

Despite this issue, we believe level sets are still preferable to a pure marker-point representation. Level set methods are more numerically stable in general due to the use of entropy-preserving PDE solvers [92,96]. The issue of preventing topological change in level sets appears as an analogous problem for marker-point methods of preventing self-intersections of the curve[2]. Narrowband methods (described on page 48 in Sec. 2.2.4) have linear computation in terms of the curve length which is the same as snake-based approaches, and the conversion between implicit level set methods and explicit marker-point models can be done fairly efficiently, so using a level set approach does not impose a significant speed penalty.

■ 3.2 Sampling from the Proposal Distribution

In this section, we detail our approach to generating samples from $\mathtt{q}(\,\cdot \,|\, \vec{C}^{(t)})$. As noted in Sec. 3.1, candidate samples are generated by adding random perturbations to the normal of a curve. We begin by introducing a continuous formulation using a canonical curve

[2] A splitting of the curve using a level set occurs when two parts of the same curve intersect. A merging of two curves occurs when two separate curves intersect.

parameterization to ensure that the perturbations are geometric (*i.e.*, dependent only on the geometry on the curve and not the particular parameterization). Specific choices to generate the perturbations $f^{(t)}(p)$ using correlated Gaussian noise with smoothness-enforcing mean perturbations are discussed. We conclude with details of the numerical implementation using level sets and a comparison of our approach to the data-driven MCMC approach of Tu and Zhu [118].

■ 3.2.1 Arc Length-parameterized Curve Perturbations

When implementing the curve perturbations, it is useful to choose a standard curve parameterization so the perturbations are geometric (*i.e.*, dependent solely on the shape, not on the specific parameterization), and arc length is a natural choice as it ensures that the curve is evenly spaced on the parameter interval [0,1]. The arc length parameterization is defined so that a given parameter value p_0 is equal to the fraction of the total curve length $L_{\vec{C}}$ which is occupied by the curve segment from $\vec{C}(0)$ to $\vec{C}(p)$:

$$p = \frac{1}{L_{\vec{C}}} \int_0^p \mathrm{d}s = \frac{1}{L_{\vec{C}}} \int_0^p \|\vec{C}'(q)\| \mathrm{d}q \ . \tag{3.3}$$

Here we focus on curves defining simply-connected regions (see Sec. 5.2 for a discussion for the case of multiple disjoint regions). Note that $\|\vec{C}'(p)\|$ (the magnitude of the derivative of the curve with respect to its parameter p) is referred to as the speed of the curve and is related to its specific parameterization. The arc length parameterization is also known as the constant speed parameterization. The reason for this can be seen by applying the fundamental theorem of calculus to (3.3) and differentiating both sides with respect to p_0. This results in $\|\vec{C}'(p)\| = L_{\vec{C}}$.

With this in mind, we slightly modify the perturbation equation in (3.2) to use the arc length-parameterized curves. Let $\vec{C}^{(t-1)}$ be the iterate at time $t-1$ and $\vec{C}_{\mathrm{a}}^{(t-1)}$ be the geometrically-equivalent curve parameterized by arc length. We then generate a candidate sample $\vec{\Gamma}^{(t)}$ by adding a random perturbation $f^{(t)} : [0,1] \to \mathbb{R}$ to the normal of $\vec{C}_{\mathrm{a}}^{(t-1)}$:

$$\vec{\Gamma}^{(t)}(p) = \vec{C}_{\mathrm{a}}^{(t-1)}(p) + f^{(t)}(p)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}^{(t-1)}}(p)\delta t \tag{3.4}$$

where δt is a finite time step. We note the similarity between (3.4) and a discrete implementation of a gradient curve flow (as in (2.21)) and will often refer to the process of generating sequences of perturbations as a flow. This perturbation formulation then

means that the problem of generating $\vec{\Gamma}^{(t)}$ is now the problem of generating $f^{(t)}$. Note that while $\vec{C}_{\mathrm{a}}^{(t-1)}$ has an arc length parameterization (and $f^{(t)}$ is parameterized with respect to $\vec{C}_{\mathrm{a}}^{(t-1)}$), it is highly unlikely that $\vec{\Gamma}^{(t)}$ does as well. Subsequent steps of the algorithm then require a reparameterization of $\vec{\Gamma}^{(t)}$ into an arc length-parameterized $\vec{\Gamma}_{\mathrm{a}}^{(t)}$.

For practical sampling implementations, fast procedures are needed to generate sample paths of $f^{(t)}(p)$ and to compute the probability of those sample paths. While there are many possible choices, in this work we focus on generating $f^{(t)}(p)$ which are composed of a deterministic mean function $\mu_{\vec{C}_{\mathrm{a}}^{(t)}} : [0,1] \rightarrow \mathbb{R}$ (dependent on the previous iterate $\vec{C}_{\mathrm{a}}^{(t-1)}$) and a correlated zero-mean random process $r^{(t)} : [0,1] \rightarrow \mathbb{R}$. These are then added together to form $f^{(t)}$:

$$f^{(t)}(p) = \mu_{\vec{C}_{\mathrm{a}}^{(t-1)}}(p) + r^{(t)}(p) \ . \tag{3.5}$$

■ 3.2.2 Correlated Noise Process

We construct the correlated noise process $r^{(t)}(p)$ by circularly convolving a smoothing kernel $h(p)$ (*e.g.*, a Gaussian kernel) and white Gaussian noise $n^{(t)}(p)$ with variance σ^2:

$$r^{(t)}(p) = h \circledast n^{(t)}(p) \ . \tag{3.6}$$

The main advantage of using a Gaussian process is the special property that linear combinations of independent Gaussian variables remain Gaussian. Therefore the resulting correlated noise process can be expressed as another Gaussian process which would not be the case if we instead used, *e.g.*, a uniform process. This is useful because it simplifies the process of evaluating the probability of the perturbation (which we discuss in Sec. 3.3).

There are, of course, many alternative methods to construct the random part of the perturbation. Circular convolution is convenient because it is easy to compute and ensures $r^{(t)}(0) = r^{(t)}(1)$ (which is needed for the curve to remain closed after applying the perturbation). Other approaches could involve Fourier or wavelet bases to provide perturbations that are more localized in space and/or frequency. Hierarchical or multi-resolution models could increase global convergence speed by resolving lower-frequency components of the curve first.

The fact that a Gaussian does not have compact support makes it fairly easy to show that the resulting Markov chain is irreducible regardless of the mean perturbation.

Note that to show a chain is ergodic, we must show it is aperiodic in addition to being irreducible. Aperiodicity is generally a difficult property to prove [34]. Because our perturbations consist largely of correlated noise and the state space is so large, it is unlikely that the chain is periodic. Also, in the event that the chain is irreducible but has cyclic behavior, the *average* sample path behavior will still converge to π. Here we first show the construction of a finite-length transition along the chain with non-zero probability for two curves $\vec{C}_0$ and $\vec{C}_T$ which are similar to each other. We then sketch a more general construction for arbitrary pairs of curves.

We begin with the case where $\vec{C}_0$ and $\vec{C}_T$ are *registered*[3] convex curves embedded in the image domain Ω. To show that there exists a flow with non-zero probability that goes from $\vec{C}_0$ to $\vec{C}_T$, we define a family of curves parameterized by a time variable $\tau \in [0, T]$ by taking a convex combination of the two curves:

$$\vec{C}(p, \tau) = \vec{C}_0(p) + \frac{\tau}{T}(\vec{C}_T(p) - \vec{C}_0(p)) \ . \tag{3.7}$$

Differentiating (3.7) with respect to τ results in:

$$\frac{\mathrm{d}\vec{C}}{\mathrm{d}\tau}(p, \tau) = \frac{1}{T}(\vec{C}_T(p) - \vec{C}_0(p)) \ . \tag{3.8}$$

We can then write this equation as a geometric flow which only uses the normal component of the flow:

$$\frac{\mathrm{d}\vec{C}}{\mathrm{d}\tau}(p, \tau) = \frac{1}{T}\left\langle \vec{C}_T(p) - \vec{C}_0(p), \vec{\mathcal{N}}_{\vec{C}_\tau}(p) \right\rangle \vec{\mathcal{N}}_{\vec{C}_\tau}(p) \ . \tag{3.9}$$

By discretizing this flow using a sufficiently small time step δt, we can write:

$$\vec{C}(p, \tau + \delta t) = \vec{C}(p, \tau) + \frac{\delta t}{T}\left\langle \vec{C}_T(p) - \vec{C}_0(p), \vec{\mathcal{N}}_{\vec{C}_\tau}(p) \right\rangle \vec{\mathcal{N}}_{\vec{C}_\tau}(p) \ . \tag{3.10}$$

This means that the force corresponding to the forward perturbation equation (3.4) is $f^{(\tau)}(p) = \frac{1}{T}\left\langle \vec{C}_T(p) - \vec{C}_0(p), \vec{\mathcal{N}}_{\vec{C}_\tau}(p) \right\rangle$. Because the Gaussian has infinite support and the Metropolis-Hastings acceptance function always has a non-zero probability, we can immediately state that this perturbation has non-zero probability under the model described above.

[3] We use the word registered here to mean that the centers of the two curves are coincident in space, and the parameterizations of the two curves are aligned so that the line connecting $\vec{C}_0(p_0)$ and $\vec{C}_T(p_0)$ (for all $p_0 \in [0, 1]$) does not intersect any other point on $\vec{C}_0$ or $\vec{C}_T$. This is a weak definition of alignment, but it is sufficient to prevent self-intersections for the flow in (3.9).

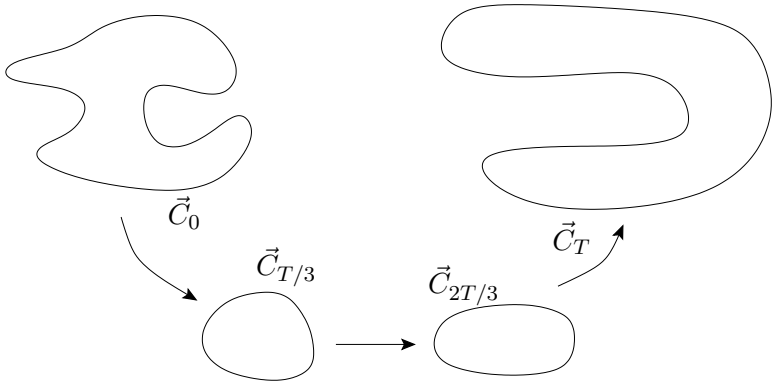


Figure 3.1. Construction of curve evolution with non-zero probability between two arbitrary curves $\vec{C}_0$ and $\vec{C}_T$. This construction transitions through intermediate convex curves $\vec{C}_{T/3}$ and $\vec{C}_{2T/3}$ which are found by running the geometric heat equation on $\vec{C}_0$ and $\vec{C}_T$ respectively.

For more complex geometries, the procedure just described can cause an intermediate curve $\vec{C}(\,\cdot\,,\tau_0)$ to develop a self-intersection for some value $\tau_0 \in (0,T)$. To avoid this problem, we can use the observation in Sec. 2.2 that a flow with a $-\kappa\vec{\mathcal{N}}$ force function shrinks a curve to a point without any self-intersections. Grayson [36], in fact, showed that the curve at an intermediate stage of the flow becomes convex. We therefore define intermediate convex curves $\vec{C}_{T/3}$ and $\vec{C}_{2T/3}$ which result from $-\kappa\vec{\mathcal{N}}$ evolutions of $\vec{C}_0$ and $\vec{C}_T$ respectively. The transition from $\vec{C}_{T/3}$ to $\vec{C}_{2T/3}$ can be constructed as above for the convex curve case. This results in an overall flow from $\vec{C}_0$ to $\vec{C}_T$ involving a $-\kappa\vec{\mathcal{N}}_{\vec{C}_\tau}$ force from $\tau \in [0, T/3]$; a $\left\langle \vec{C}_{2T/3}(p) - \vec{C}_{T/3}(p),\, \vec{\mathcal{N}}_{\vec{C}_\tau}(p) \right\rangle$ force for $\tau \in (T/3, 2T/3]$; and a $+\kappa\vec{\mathcal{N}}_{\vec{C}_\tau}$ force[4] for $\tau \in (2T/3, T]$. These flows each have non-zero probability under the Gaussian perturbation model. This process is illustrated in Fig. 3.1.

The fact that a Gaussian has infinite support also has a downside. As mentioned earlier, level set methods are only stable for small-enough changes in the curve location. For an implementation with finite step size, an appropriate choice for δt depends on the maximum magnitude of the force function f. While the infinite support of a Gaussian perturbation means that the maximum magnitude is unbounded, a finite value can be found for which the chance of a perturbation exceeding it is less than some number

[4] A numerical implementation of an evolution under $+\kappa\vec{\mathcal{N}}_{\vec{C}_\tau}$ must be done carefully as it is an unstable flow (akin to running the heat equation backwards in time). Here we are simply examining a theoretical construction for the existence of a flow with non-zero probability.

$\epsilon > 0$. This results in the need to be fairly conservative in choosing σ^2, the variance of the noise process n, and δt. If our perturbation f had finite support, we could design it and δt so that the support of the perturbation exactly matched the stability of our numerical PDE solver.

■ 3.2.3 Mean Process

While it is not necessary to have a mean component to the perturbation in order to satisfy detailed balance (and provide asymptotic convergence), empirical results have shown that it is needed to increase the mixing rate of the sampling algorithm so convergence occurs more quickly. Without a deterministic smoothing force, even if $r(p)$ is smooth and there is a smoothness prior in $\pi(\vec{C}\,|\,I)$, generating smooth curves from a large number of consecutive samples of the transition probability of the chain is extremely unlikely. While each $r^{(t)}(p)$ is smooth, the sum of N of these random fields is somewhat similar to circularly convolving $h(p)$ with a random white Gaussian process with variance proportional to N. This results in jagged curves being more likely after a large number of iterations regardless of the choice of h or σ^2.

To increase convergence speed, we need the mean to move the curve toward higher-probability regions of π. Certainly many reasonable choices are possible. One example would be setting $\mu_{\vec{C}_{\mathrm{a}}^{(t)}}(p)$ to be the gradient flow of $\log \pi(\vec{C}\,|\,I)$ at $\vec{C}_{\mathrm{a}}^{(t)}$, taking care to ensure that the mean component does not overly dominate the random component of the perturbation. In the work here, we typically define the mean as a negative curvature term times a regularization parameter α plus a positive balloon force:

$$\mu_{\vec{C}_{\mathrm{a}}^{(t)}}(p) = -\alpha \kappa_{\vec{C}_{\mathrm{a}}^{(t)}}(p) + \gamma_{\vec{C}_{\mathrm{a}}^{(t)}} \quad . \tag{3.11}$$

The curvature flow is a regularizing flow that encourages smooth curves. The positive constant $\gamma_{\vec{C}_{\mathrm{a}}^{(t)}}$ (which can be a function of $\vec{C}_{\mathrm{a}}^{(t)}$) counteracts the generally inward-flowing $-\kappa_{\vec{C}_{\mathrm{a}}^{(t)}}$ curve shortening term. We note that the method we used to show ergodicity of the chain in Sec. 3.2.2 is independent of the choice of $\mu_{\vec{C}_{\mathrm{a}}^{(t)}}(p)$ as are the computations necessary for detailed balance in Sec. 3.3.

Assuming an arc length parameterization for simplicity, the total curvature of a simple closed curve $\vec{C}$ can be written as

$$K = \oint_{\vec{C}} \kappa_{\vec{C}}(s) \mathrm{d}s = 2\pi \quad . \tag{3.12}$$

This result is obvious for a circle as $\kappa_{\vec{C}} = 1/R$ (where R is the radius of the circle), so

$K = 2\pi R/R = 2\pi$. For a derivation for arbitrary simple curves, see, *e.g.*, Klassen *et al.* [60]. Thus we can see that the average value of $-\kappa_{\vec{C}_{\mathrm{a}}^{(t)}}$ over the curve is $-2\pi/L_{\vec{C}_{\mathrm{a}}^{(t)}}$ which biases the perturbation toward smaller curves.

If the size of the curve does not vary greatly over the course of a simulation run, $\gamma_{\vec{C}_{\mathrm{a}}^{(t)}}$ can be fixed before running the sampler. If the curve size does change quite a bit, it may be necessary to vary $\gamma_{\vec{C}_{\mathrm{a}}^{(t)}}$ because the magnitude of $\kappa_{\vec{C}_{\mathrm{a}}^{(t)}}$ is inversely proportional to the size of the curve. If $\gamma_{\vec{C}_{\mathrm{a}}^{(t)}}$ is not adjusted, it will become too large for long curves (which would increase the bias toward even longer curves) and vice versa for short curves. A simple effective heuristic is to choose $\gamma_{\vec{C}_{\mathrm{a}}^{(t)}} = \gamma_0/L_{\vec{C}_{\mathrm{a}}^{(t)}}$. As noted earlier, the average value of $\kappa_{\vec{C}_{\mathrm{a}}^{(t)}}$ is also inversely proportional to $L_{\vec{C}_{\mathrm{a}}^{(t)}}$, so $\kappa_{\vec{C}_{\mathrm{a}}^{(t)}}$ and $\gamma_{\vec{C}_{\mathrm{a}}^{(t)}}$ will scale identically with respect to curve length. Chosen this way, the mean perturbation has a net effect of approximately zero when averaged over the curve (*i.e.*, it does not bias the proposal distribution to larger or smaller curves), but it has a smoothing effect on regions of high absolute curvature.

Several other choices are available as curve regularizing terms. The affine invariant curve evolution method of Sapiro and Tannenbaum [93] uses a $-\kappa_{\vec{C}}^{1/3}$ flow. This may be preferable to a $-\kappa_{\vec{C}}$ regularizing term for certain situations. Euclidean curvature flow has a bias toward circles while affine curvature flow prefers ellipses which could be advantageous for highly oblong objects. Our perturbation method using curvature flow also makes it highly unlikely to generate sections of the curve with large negative curvature because for those regions both $-\kappa_{\vec{C}}$ and $\gamma_{\vec{C}}$ are positive. This makes it very likely that the perturbation will force the curve outward. Using $\kappa_{\vec{C}}^{1/3}$ instead should ameliorate this effect somewhat. Another possibility is the more-than-topology-preserving flow of Sundaramoorthi and Yezzi [104] (see Sec. 2.2.1).

■ 3.2.4 Numerical Implementation

As discussed in Sec. 3.1, we implement the sampling process by approximating the continuous formulation as a discrete process with a finite time step δt. This is done using a hybrid approach of narrowband level sets and discrete marker point representations because some of the computations associated with the sampling process require access to an explicit representation of the curve. Therefore using level sets requires conversion back and forth between implicit and explicit representations which imposes additional

computational overhead[5].

The overall algorithm is implemented in a mix of Matlab [47] and C++ [102]. Matlab is useful for rapid prototyping and access to an extensive numerical codebase, while C++ is useful to speed up certain critical sections of the code. Note that this is somewhat inefficient as not all data structures are directly transferable between the two environments, so a conversion penalty must be paid at each iteration.

At the beginning of an iteration, the previous curve iterate $\vec{C}^{(t-1)}(p)$ is represented by a level set function $\Psi_{\vec{C}^{(t-1)}}$. We extract a length-N_c ordered sequence of curve points $\{\vec{C}_{\mathrm{a}}^{(t-1)}(p_i)\}_{i=0}^{N_c-1}$ with the set of parameterization points $\{p_i\}_{i=0}^{N_c-1}$ determined by our curve extraction algorithm based on the *marching squares* algorithm [69] (see Appendix A).

The random portion of the curve perturbation $r^{(t)}(p)$ is calculated as an N_r-length discrete vector $\boldsymbol{r}^{(t)}$ which is evenly sampled on the interval $[0, 1]$. We use the notation $\boldsymbol{r}^{(t)}[i]$ to indicate the ith element of $\boldsymbol{r}^{(t)}$ (for $i \in \{0, 1, \ldots, N_r - 1\}$). Then $\boldsymbol{r}^{(t)}[i] = r^{(t)}(i/N_r)$. This correlated noise is generated by first sampling an N_r-length white Gaussian vector $\boldsymbol{n}^{(t)} \sim \mathrm{N}(0, \sigma^2\boldsymbol{I})$ then multiplying it by a circulant matrix $\boldsymbol{H}$ which implements circular convolution with h, the low pass filter[6].

The resulting correlated noise vector $\boldsymbol{r}^{(t)} = \boldsymbol{H}\boldsymbol{n}^{(t)}$ is interpolated to the set of parameterization points $\{p_i\}_{i=1}^{N_c}$ corresponding to the discrete marker points of $\vec{C}_{\mathrm{a}}^{(t-1)}$. This interpolation can be done using, *e.g.*, linear interpolation or cubic spline methods. This results in $\tilde{\boldsymbol{r}}^{(t)}$, a new length-$N_c$ vector where the elements are defined as $\tilde{\boldsymbol{r}}^{(t)}[i] = r^{(t)}(p_i)$.

Once the interpolation is complete, we compute $\mu_{\vec{C}_{\mathrm{a}}^{(t)}}(p)$ on the curve points to obtain $\boldsymbol{\mu}_{\vec{C}^{(t-1)}}$. This combines with $\boldsymbol{r}^{(t)}$ to form the vector $\boldsymbol{f}^{(t)}$ on the curve points:

$$\boldsymbol{f}^{(t)} = \tilde{\boldsymbol{r}}^{(t)} + \boldsymbol{\mu}_{\vec{C}^{(t-1)}} \quad . \tag{3.13}$$

These perturbation values are then extended to all of Ω (using velocity-extension methods described in Sec. 2.2.4) to form $F^{(t)}(\boldsymbol{x})$ and used to update the level set function to

[5]It may be possible to formulate an analogous perturbation defined directly on the level set. Care would need to be taken in defining the probability of such a perturbation as the level set is a highly redundant curve representation (*i.e.*, many level set functions map to the same curve).

[6]This operation is efficiently implemented using a fast Fourier transform (FFT) [81], an $\mathcal{O}(N \log N)$ algorithm for computing the discrete Fourier transform. The discrete Fourier transform has the property of transforming circular convolution in the spatial domain to multiplication in the frequency domain, so we can perform the circular convolution (normally an $\mathcal{O}(N^2)$ operation) in $\mathcal{O}(N \log N)$.

obtain the new level set function of $\vec{\Gamma}^{(t)}$. If the computation of the mean involves curvature terms (such as in (3.11)), those are most easily and accurately computed directly from partial derivatives of the level set $\Psi_{\vec{C}^{(t-1)}}$. The partial derivatives are implemented as double-sided derivatives because the negative curvature flows are smoothing flows, so they use information from both directions on the curve. This is in contrast to the non-curvature terms (such as the $r^{(t)}(p)\vec{\mathcal{N}}_{\vec{C}^{(t-1)}}(p)$ component) which typically use single-sided derivatives, with the directionality chosen based on entropy-preserving conditions (which preserve sharp features). This is more fully described in the books by Sethian [96] or Sapiro [92].

The following summarizes the algorithmic steps for generating the candidate sample $\vec{\Gamma}^{(t)} \sim \mathfrak{q}(\vec{\Gamma} \,|\, \vec{C}^{(t-1)})$:

1. Extract discrete marker points $\{\vec{C}_{\mathrm{a}}^{(t-1)}(p_i)\}_{i=0}^{N_c-1}$ from level set function $\Psi_{\vec{C}^{(t-1)}}$.
2. Sample discrete white Gaussian noise vector $\boldsymbol{n}^{(t)} \sim \mathrm{N}(0, \sigma^2 \boldsymbol{I})$.
3. Filter noise to form correlated noise vector $\boldsymbol{r}^{(t)} = \boldsymbol{H}\boldsymbol{n}^{(t)}$ (where $\boldsymbol{H}$ implements the circular convolution operation with a low-pass filter) as in (3.6).
4. Interpolate $\boldsymbol{r}^{(t)}$ to discrete curve points and construct mean perturbation $\boldsymbol{\mu}_{\vec{C}^{(t-1)}}$ (from (3.11)). Combine together to form overall forward perturbation $\boldsymbol{f}^{(t)}$ as in (3.5) and (3.13).
5. Apply $\boldsymbol{f}^{(t)}$ to normal of $\vec{C}^{(t-1)}$ using a narrowband level set method to obtain

$$\Psi_{\vec{\Gamma}^{(t)}}(\boldsymbol{x}) = \Psi_{\vec{C}^{(t-1)}}(\boldsymbol{x}) - F^{(t)}(\boldsymbol{x})\|\nabla\Psi_{\vec{C}^{(t-1)}}(\boldsymbol{x})\|\delta t \tag{3.14}$$

   where $F^{(t)}(\boldsymbol{x})$ is a velocity-extended version of $\boldsymbol{f}^{(t)}$. This equation is the level set equivalent of (3.4) as defined in Sec. 2.2.4.

■ 3.2.5 Comparison to Existing Methods

The existing work most closely related to ours is that of Tu and Zhu [118] which is also an MCMC-based algorithm. Similar to our approach, the authors also consider perturbations with a deterministic component plus Gaussian noise. The major difference is that they are looking for global optima using a simulated annealing approach [59] and are not trying to draw samples from a probability distribution, so they do not attempt to maintain detailed balance in their Markov chain (which we address for our approach

in Sec. 3.3). Therefore they are unable to provide estimates of error variances and other higher-order statistics.

Other differences in the approaches include:

1. Their approach allows for merging and splitting of regions through a jump diffusion process. This would be a valuable extension for our method, though a major difficulty is the need to ensure that the chain satisfies detailed balance.

2. We use correlated Gaussian noise while they use white noise. Correlated noise creates smoother perturbations which allows us to have more freedom in setting a higher strength for the random component relative to the strength of the mean component of the perturbation.

3. Tu and Zhu must use the gradient flow of the log probability as a mean perturbation while we can use arbitrary mean perturbations. This is because our Markov chain satisfies detailed balance, so incorporation of the information from the probability distribution is properly included when accepting/rejecting samples.

■ 3.3 Evaluating the Hastings Ratio

Metropolis-Hastings sampling requires that we compute the Hastings ratio:

$$\eta(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)}) = \frac{\pi(\vec{\Gamma}^{(t)})}{\pi(\vec{C}^{(t-1)})} \cdot \frac{\mathsf{q}(\vec{C}^{(t-1)} \,|\, \vec{\Gamma}^{(t)})}{\mathsf{q}(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})} \ . \tag{3.15}$$

This requires us to evaluate both the forward and reverse proposal distribution probabilities in addition to the target distribution probabilities. This computation is needed in order to ensure detailed balance and, along with ergodicity, guarantee that our samples come asymptotically from the target distribution. As computing the target distribution probabilities $\pi(\vec{C}^{(t-1)})$ and $\pi(\vec{\Gamma}^{(t)})$ tends to be application specific, we do not discuss it in detail here except to note that any region-based terms in the computation can be efficiently implemented using update computations. This means that given the value of $\pi(\vec{C}^{(t-1)})$, we can evaluate $\pi(\vec{\Gamma}^{(t)})$ by only performing computations on the pixels in Ω where the segmentation label from $\vec{\Gamma}^{(t)}$ and $\vec{C}^{(t-1)}$ differs[7]. As the number of pixels

[7]Determining the list of pixels which have changed label is an $\mathcal{O}(N^2)$ computation for general level set methods (as every pixel in Ω must be examined) but is $\mathcal{O}(N)$ using narrowband methods. This is because only pixels in the band change label (for stability reasons, the maximum perturbation should not be more than 1 pixel), and the number of pixels in the band is linear in the length of the curve.

changing label will be approximately related to the length of the curve, this changes the computation from $\mathcal{O}(N^2)$ (as region-based energy functionals must generally be evaluated over the entire image domain) to $\mathcal{O}(N)$. The only full evaluation of π needs to be done for the initial curve $\vec{C}^{(0)}$.

The other terms in the Hastings ratio involve the proposal distribution $\mathtt{q}$. Because $\mathtt{q}$ is asymmetric, we must compute both $\mathtt{q}(\vec{\Gamma}^{(t+1)} \,|\, , \vec{C}^{(t)})$ and $\mathtt{q}(\vec{C}^{(t)} \,|\, \vec{\Gamma}^{(t+1)})$. This asymmetry is due to the mean component which biases the proposal distribution to generate perturbations in a certain direction. To see why this asymmetry occurs, consider the mean component defined in (3.11). Even though $\vec{C}^{(t)}$ and $\vec{\Gamma}^{(t+1)}$ are similar in terms of global geometric features, $\mu_{\vec{C}^{(t)}}(p)$ and $\mu_{\vec{\Gamma}^{(t+1)}}(q)$ are not identical because they are based on curvature, an inherently local feature. Thus the forward and reverse perturbations are biased in different directions. This explanation is somewhat incomplete because having the same mean perturbation will still result in an asymmetric proposal distribution. This is because the forward perturbation f and the reverse perturbation ϕ will be in opposite directions. So if f pushes the curve in the same direction as the mean perturbation, ϕ would necessarily push the curve in the opposite direction of the mean perturbation. The former is a likely event and the latter is unlikely, so the perturbations have different probabilities even with the same mean.

Evaluation of $\mathtt{q}(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t)})$ and $\mathtt{q}(\vec{C}^{(t)} \,|\, \vec{\Gamma}^{(t)})$ is non-trivial for the curve perturbations defined in the previous section, and the focus of this section is on methods to compute these values. We begin by showing how to calculate the probability of the discrete approximation for the forward proposal distribution sampling process. The accuracy of the computation depends both on the time step δt and the discretization density along the curve. We then define a continuous formulation for the reverse perturbation $\mathtt{q}(\vec{C}^{(t)} \,|\, \vec{\Gamma}^{(t)})$ and approximate its probability computation in an analogous manner to the forward proposal distribution. We conclude by showing methods to efficiently estimate the reverse perturbation function. One approach leads to an approximate existence condition for the reverse perturbation which is valid up to 2nd order.

Note that for the remainder of this section, we typically drop the time superscripts (with the understanding that $\vec{C}$ represents the previous iterate and $\vec{\Gamma}$ the candidate sample) when doing so does not cause ambiguity.

■ 3.3.1 Forward Proposal Distribution Probability

In this section, we detail the calculation of the value of the forward proposal distribution. In order to evaluate the forward proposal distribution probability $\mathsf{q}(\vec{\Gamma}\,|\,\vec{C})$, we note that the perturbation defined in (3.4) is a differential in the direction of the normal, so given $\vec{C}$, there is a one-to-one mapping between a perturbation f and a curve $\vec{\Gamma}$ for δt infinitesimal. For small finite δt, this remains approximately true, though there is a small chance that multiple f can perturb a given $\vec{C}$ into the same $\vec{\Gamma}$ (*i.e.*, two candidate curves with the same geometry but different parameterizations). Thus we can say that the probability of generating the candidate sample $\vec{\Gamma}$ given $\vec{C}$ (*i.e.*, $\mathsf{q}(\vec{\Gamma}|\vec{C})$) is approximately $\mathrm{p}(f)$, the probability of generating the perturbation f.

We approximate the actual computation of $\mathrm{p}(f)$ using the discrete approximation to f defined in Sec. 3.2.4. As noted previously, the random vector $\boldsymbol{f}$ is obtained as the mean vector plus a filtered white noise vector:

$$\boldsymbol{f} = \boldsymbol{H}\boldsymbol{n} + \boldsymbol{\mu}_{\vec{C}_\mathrm{a}} \ . \tag{3.16}$$

This results in $\boldsymbol{f}$ being drawn from a distribution $\mathrm{N}(\boldsymbol{\mu}_{\vec{C}}, \boldsymbol{H}\boldsymbol{H}^\mathrm{T})$. We can see that $\boldsymbol{f}$ is deterministically generated from $\boldsymbol{n}$. If $\boldsymbol{H}$ is invertible, the mapping between the two vectors is one-to-one. This results in the following probability computation:

$$\mathsf{q}(\vec{\Gamma}\,|\,\vec{C}) \approx \mathrm{p}_\mathbf{f}(\boldsymbol{f}) \propto \mathrm{p}_\mathbf{n}(\boldsymbol{n}) \propto \exp\left(-\frac{\boldsymbol{n}^\mathrm{T}\boldsymbol{n}}{2\sigma^2}\right) \ . \tag{3.17}$$

Note that the same computation is performed here regardless of whether $\mu_{\vec{C}_\mathrm{a}}$ is as detailed in (3.11) or defined in any other manner.

■ 3.3.2 Reverse Perturbation

In Sec. 3.2, we defined a method of generating samples from $\mathsf{q}(\vec{\Gamma}\,|\,\vec{C})$ by adding a Gaussian perturbation to the normal of $\vec{C}$. The reverse perturbation (*i.e.*, generating $\vec{C}$ from $\vec{\Gamma}$ through $\mathsf{q}(\vec{C}\,|\,\vec{\Gamma})$) is then obtained in a similar fashion by adding a different Gaussian perturbation to the normal of $\vec{\Gamma}$. We can set up analogous equations to (3.4) and (3.5) to define this perturbation:

$$\vec{C}_\mathrm{r}(q) = \vec{\Gamma}_\mathrm{a}(q) + \phi(q)\vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q)\delta t \tag{3.18}$$

$$\phi(q) = \mu_{\vec{\Gamma}_\mathrm{a}}(q) + h \circledast \nu(q) \ . \tag{3.19}$$

Here $\vec{C}_\mathrm{r}$ is a curve which is geometrically equivalent to $\vec{C}$ but with a different parameterization. It is obtained by perturbing $\vec{\Gamma}_\mathrm{a}$, the arc length-parameterized version of

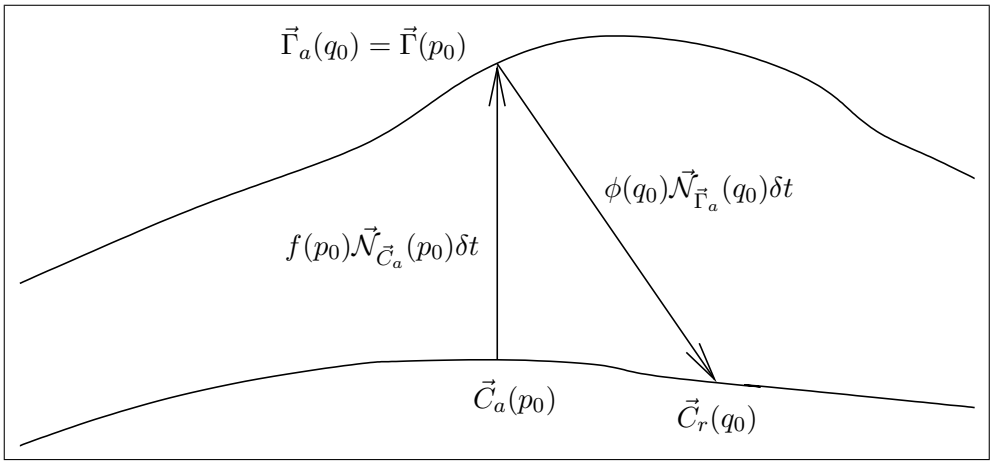


Figure 3.2. Illustration of the process of finding the reverse perturbation value $\phi(q_0)\delta t$ which takes $\vec{\Gamma}_{\mathrm{a}}(q_0)$ to a point $\vec{C}_{\mathrm{r}}(q_0)$ on the original curve $\vec{C}$ by tracing along the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$. We also show the forward perturbation $f(p_0)\delta t$ along the normal $\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$ which took $\vec{C}_{\mathrm{a}}(p_0)$ to $\vec{\Gamma}_{\mathrm{a}}(q_0)$.

$\vec{\Gamma}$, with a random perturbation ϕ. The perturbation ϕ is composed of a mean term (equivalent to that in (3.11) except derived from $\vec{\Gamma}$, not $\vec{C}$) plus filtered white Gaussian noise ν. We use a different curve parameter variable q here to make clear the fact that the forward and reverse perturbations have different parameterizations.

Note that the problem which we need to solve here differs from that of the forward perturbation. For that case, we know $\vec{C}$ and generate f to obtain $\vec{\Gamma}$. Now for the reverse perturbation, we know $\vec{C}$ and $\vec{\Gamma}$ and wish to obtain the perturbation ϕ which takes us from $\vec{\Gamma}$ to $\vec{C}$ in order to evaluate the reverse probability.

To establish some notation, we observe that—because $\vec{\Gamma}$ and $\vec{\Gamma}_{\mathrm{a}}$ are geometrically equivalent—for every $q_0 \in [0,1]$, there exists an index p_0 for which $\vec{\Gamma}(p_0) = \vec{\Gamma}_{\mathrm{a}}(q_0)$[8]. This means that there is a correspondence between $\vec{C}_{\mathrm{a}}(p_0)$ and $\vec{\Gamma}_{\mathrm{a}}(q_0)$, and we will use p_0 and q_0 below to index corresponding points in the two curves.

We can see from (3.18) that at a given point q_0, the value $\phi(q_0)\delta t$ is the distance we must trace back along the normal at $\vec{\Gamma}_{\mathrm{a}}(q_0)$ until we first intersect $\vec{C}$. This process is shown in Fig. 3.2. The point at which the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ intersects $\vec{C}$ is then labeled $\vec{C}_{\mathrm{r}}(q_0)$. For small δt and smooth curves $\vec{C}$ (say $\kappa_{\vec{C}}(p)$ small), this is generally possible to do for each $q_0 \in [0,1]$ using a brute-force numerical search. We also derive some

[8]The mapping between p_0 and q_0 is strictly monotonic if the parameterization is defined such that $\vec{\Gamma}(0) = \vec{\Gamma}_{\mathrm{a}}(0)$.

approximate methods to do this tracing procedure in Sec. 3.3.3. Note that it is possible that no valid choice for $\phi(q_0)$ exists to satisfy (3.18); we discuss this case further in Sec. 3.3.4.

In Sec. 3.3.1, we detailed a method to calculate $\mathsf{q}(\vec{\Gamma} \,|\, \vec{C})$ by computing the probability of the discrete forward perturbation $\boldsymbol{f}$. Similar arguments presented in that section hold for the reverse proposal distribution probability $\mathsf{q}(\vec{C} \,|\, \vec{\Gamma})$ where a discrete reverse perturbation vector $\boldsymbol{\phi}$ is defined in an analogous fashion to Sec. 3.2.4:

$$\mathsf{q}(\vec{C} \,|\, \vec{\Gamma}) \approx \mathrm{p}_\phi(\boldsymbol{\phi}) \propto \mathrm{p}_\nu(\boldsymbol{\nu}) \propto \exp\left(-\frac{\boldsymbol{\nu}^{\mathrm{T}}\boldsymbol{\nu}}{2\sigma^2}\right) \tag{3.20}$$

where $\boldsymbol{\nu} = \boldsymbol{H}^{-1}(\boldsymbol{\phi} - \boldsymbol{\mu}_{\vec{\Gamma}})$. Note that because the smoothing kernel h is a low-pass filter, applying $\boldsymbol{H}^{-1}$ can be numerically unstable (*i.e.*, $\boldsymbol{H}$ has a large condition number). This is not a major concern for us as the $\boldsymbol{\nu}$ value is only used to evaluate the Hastings ratio for this iteration. Any numerical imprecision that is imparted will likely not affect the accept/reject stage of the current iteration[9] and is not propagated to future iterations.

■ 3.3.3 Approximate ϕ Computation

In Sec. 3.3.2, we described how to obtain the reverse perturbation value $\phi(q_0)$ at $\vec{\Gamma}_{\mathrm{a}}(q_0)$ by tracing back along the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ until it intersects the curve at a point $\vec{C}_{\mathrm{r}}(q_0)$. This tracing procedure is cumbersome, though certainly possible to perform. A number of approximations or simplifications can be used to make the calculation of ϕ easier. Here we present three such approaches. We again use the notion that parameter values p_0 and q_0 exist which are related through $\vec{\Gamma}(p_0) = \vec{\Gamma}_{\mathrm{a}}(q_0)$ with $\vec{\Gamma}(p_0)$ being obtained from $\vec{C}_{\mathrm{a}}(p_0)$ through the forward perturbation equation (3.4). The first method assumes that the normal to the original curve is identical to the normal of the candidate curve for all p_0 (*i.e.*, $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0) = \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$). The second and third approaches approximate the curve near $\vec{C}_{\mathrm{a}}(p_0)$ with a line and a circle respectively. Note that the circle approximation is more complex (from a numerical point of view) than the first two as it requires the use of higher-order derivatives of the curve, but it does lead to an approximate condition for the existence of the reverse perturbation which we detail in Sec. 3.3.4.

These approaches to estimate ϕ are reasonable because $\vec{\Gamma}_{\mathrm{a}}(q_0)$ is generated from $\vec{C}_{\mathrm{a}}(p_0)$ using a small perturbation $f(p_0)\delta t$. This means that $\vec{C}_{\mathrm{a}}(p_0)$ is likely close (in a

[9] Due to the fact that a candidate sample is accepted with probability $\eta(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})$, there is no difference in behavior for a Hastings ratio of 100 or 1000, and very little difference in behavior for a Hastings ratio of 0.01 or 0.001

Euclidean distance sense) to $\vec{C}_{\mathrm{r}}(q_0)$ which then implies that local approximations to $\vec{C}_{\mathrm{a}}$ around p_0 should have small error near $\vec{C}_{\mathrm{r}}(q_0)$.

Constant normal approximation

The first simplifying assumption we make is that $\vec{\mathcal{N}}_{\vec{\Gamma}}(p_0) = \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$ (with this assumption, $\vec{\mathcal{N}}_{\vec{\Gamma}}(p_0)$ will no longer be orthogonal to $\vec{\Gamma}$ at p_0 in general). In this case, solving for ϕ is trivial. If we trace back along $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$, we will, of course, intersect $\vec{C}_{\mathrm{a}}(p_0)$. This can be seen in the example in Fig. 3.2. Thus we can write

$$\hat{\phi}_{\mathrm{cn}}(q_0) = -f(p_0) \ . \tag{3.21}$$

It is virtually impossible that $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ and $\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$ are actually identical everywhere on the curve. The curve normal is a unit-normalized derivative of the curve, so identical normal functions would imply that one curve is a translated and scaled version of the other. This is a probability 0 event with curves generated by a random Gaussian perturbation.

What is likely though is that $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ and $\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$ are very similar. As noted in Sec. 2.2, the normal vector is a rotation of the tangent vector. Thus we can write (using the definition of $\vec{\Gamma}(q_0)$ from (3.4))

$$\begin{aligned}
\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0) = \vec{\mathcal{N}}_{\vec{\Gamma}}(p_0) = \boldsymbol{R}\vec{\mathcal{T}}_{\vec{\Gamma}}(p_0) &= \boldsymbol{R}\frac{\vec{\Gamma}'(p_0)}{\|\vec{\Gamma}'(p_0)\|} \\
&= \frac{\boldsymbol{R}(\vec{C}'_{\mathrm{a}}(p_0) + f'(p_0)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)\delta t + f(p_0)\vec{\mathcal{N}}'_{\vec{C}_{\mathrm{a}}}(p_0)\delta t)}{\|\vec{C}'_{\mathrm{a}}(p_0) + f'(p_0)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)\delta t + f(p_0)\vec{\mathcal{N}}'_{\vec{C}_{\mathrm{a}}}(p_0)\delta t\|} \\
&= \frac{\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0) + \boldsymbol{R}(f'(p_0)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0) + f(p_0)\vec{\mathcal{N}}'_{\vec{C}_{\mathrm{a}}}(p_0))\delta t}{\|\vec{C}'_{\mathrm{a}}(p_0) + f'(p_0)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)\delta t + f(p_0)\vec{\mathcal{N}}'_{\vec{C}_{\mathrm{a}}}(p_0)\delta t\|}
\end{aligned} \tag{3.22}$$

where $\boldsymbol{R}$ is the rotation matrix which converts the tangent vector into a normal vector. When δt is small, $\|\vec{C}'_{\mathrm{a}}(p_0) + f'(p_0)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)\delta t + f(p_0)\vec{\mathcal{N}}'_{\vec{C}_{\mathrm{a}}}(p_0)\delta t\|$ is close to $\|\vec{C}'_{\mathrm{a}}(p_0)\| = 1$, so we can see that for $\delta t \to 0$, $\vec{\mathcal{N}}_{\vec{\Gamma}} \to \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}$. Therefore we can conclude that the constant normal approximation is reasonable for small δt.

Linear approximation

We can take a more sophisticated approach by constructing locally-linear approximations to $\vec{C}(p)$. As noted earlier, we expect $\vec{C}_{\mathrm{a}}(p_0)$ to be close to $\vec{C}_{\mathrm{r}}(q_0)$, so we can

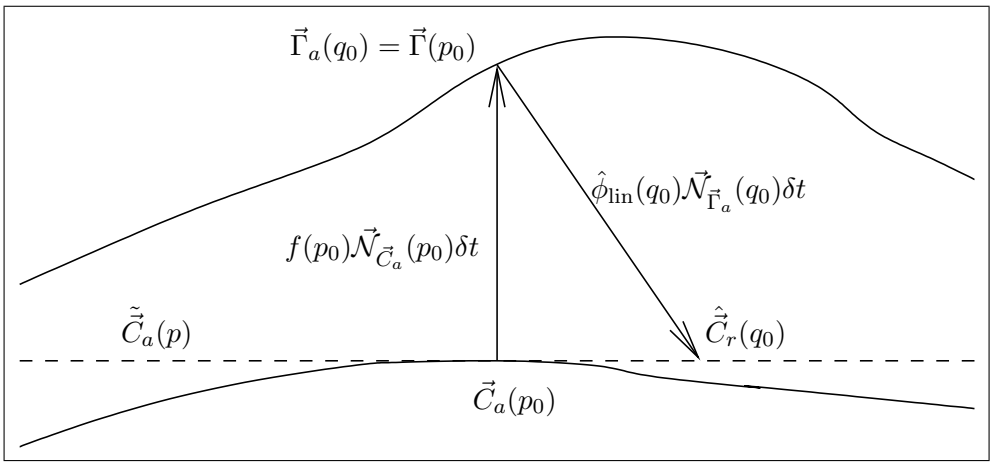


Figure 3.3. Illustration of the process of forming linear approximations to $\vec{C}_{\text{a}}(p)$ to simplify the estimation of the reverse perturbation ϕ. Here we show the perturbation $f(p_0)\delta t$ along the normal $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0)$ which took $\vec{C}_{\text{a}}(p_0)$ to $\vec{\Gamma}(p_0)$, the linear approximation to $\vec{C}_{\text{a}}(p)$ around p_0 (as the dashed line), and the estimated reverse perturbation $\hat{\phi}_{\text{lin}}(q_0)\delta t$ along $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ which intersects the linear approximation.

attempt to find $\phi(q_0)$ by forming a linear approximation to $\vec{C}_{\text{a}}(p)$ around p_0. We can write the line which is tangent to the curve at $\vec{C}_{\text{a}}(p_0)$ as $\vec{C}_{\text{a}}(p_0) + b\,\vec{\mathcal{T}}_{\vec{C}_{\text{a}}}(p_0)$ for $b \in \mathbb{R}$. Then to estimate $\phi(q_0)$, we simply need to trace along the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ until we intersect the line approximating $\vec{C}_{\text{a}}$. This process is illustrated in Fig. 3.3.

This linear approximation is a reasonable thing to do if $\vec{C}_{\text{a}}(p_0)$ is in fact close to $\vec{C}_{\text{r}}(q_0)$. Naturally, the higher the curvature (a second-derivative term of the curve), the higher the approximation error, and the need for smaller time steps or higher-order approximations. As shown in Fig. 3.3, $\hat{\phi}_{\text{lin}}(q_0)\delta t$ is less than the true value of $\phi(q_0)\delta t$ (which needs to extend the normal all the way to the solid line representing $\vec{C}$) as the curvature of $\vec{C}_{\text{a}}$ creates a divergence between the true curve and its linear approximation. If the curve instead were concave, $\hat{\phi}_{\text{lin}}(q_0)\delta t$ would be greater than the true value of $\phi(q_0)\delta t$.

The locally-linear approximation is computationally convenient because we can estimate $\phi(q_0)$ in closed form:

$$\hat{\phi}_{\text{lin}}(q_0) = -\frac{f(p_0)}{\left\langle \vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0),\, \vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0) \right\rangle} \quad . \tag{3.23}$$

We derive this formula in Appendix B.1. For this approximation, $\hat{\phi}_{\text{lin}}(q_0)$ always exists unless $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0) \perp \vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$. When $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0) = \vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ (as in the previous constant

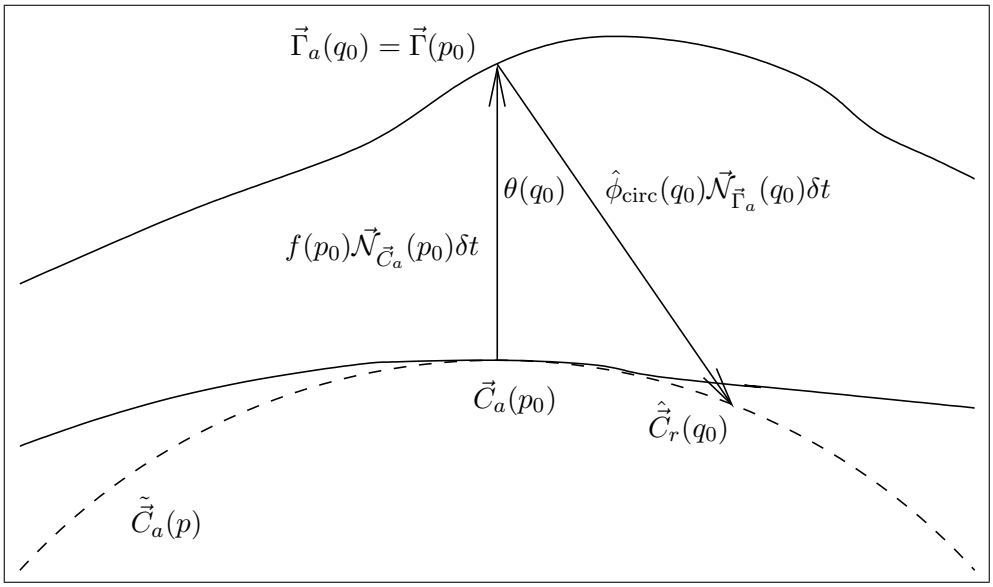


Figure 3.4. Illustration of the process of forming circle approximations to $\vec{C}_{\text{a}}(p)$ to simplify the estimation of the reverse perturbation ϕ. Here we show the perturbation $f(p_0)\delta t$ along the normal $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0)$ which took $\vec{C}_{\text{a}}(p_0)$ to $\vec{\Gamma}(p_0)$, the circle approximation to $\vec{C}_{\text{a}}(p)$ around p_0 (as the dashed line), and the estimated reverse perturbation $\hat{\phi}_{\text{circ}}(q_0)\delta t$ along $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ which intersects the circle approximation.

normal approximation), we get the expected $\hat{\phi}_{\text{lin}}(q_0) = -f(p_0)$. Otherwise the more $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0)$ and $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$ differ, the larger $\hat{\phi}_{\text{lin}}(q_0)$ becomes.

Second-order approximation

A better approximation can be made by using second-order derivatives in constructing the local approximation around $\vec{C}_{\text{a}}(p_0)$. The normal $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0)$ and the curvature $\kappa_{\vec{C}_{\text{a}}}(p_0)$ define a circle that passes through $\vec{C}_{\text{a}}(p_0)$ with radius $1/\kappa_{\vec{C}_{\text{a}}}(p_0)$ and center $\vec{C}_{\text{a}}(p_0) - \vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0)/\kappa_{\vec{C}_{\text{a}}}(p_0)$. We then wish to find the intersection of the line along the normal with this circle approximation to $\vec{C}$. This process is shown in Fig. 3.4. This can again be solved in closed form as:

$$\hat{\phi}_{\text{circ}}(q_0) = -f(p_0)\cos\theta(q_0) + \frac{\sqrt{1 - (1 + f(p_0)\kappa_{\vec{C}_{\text{a}}}(p_0)\delta t)^2 \sin^2\theta(q_0)} - \cos\theta(q_0)}{\kappa_{\vec{C}_{\text{a}}}(p_0)\delta t} \quad (3.24)$$

where $\theta(q_0)$ is the angle between $\vec{\mathcal{N}}_{\vec{C}_{\text{a}}}(p_0)$ and $\vec{\mathcal{N}}_{\vec{\Gamma}_{\text{a}}}(q_0)$). See Appendix B.2 for the derivation.

We now examine two limiting conditions to see if the second-order approximation produces the behavior we would expect. The first is when $\theta(q_0) = 0$ (*i.e.*, $\vec{\mathcal{N}}_{\vec{C}_a}(p_0) = \vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$). In this case, $\cos(\theta(q_0)) = 1$ and $\sin(\theta(q_0)) = 0$, and substituting into (3.24), we end up with:

$$\hat{\phi}_{\text{circ}}(q_0) = -f(p_0) \ . \tag{3.25}$$

This is the same result we obtained earlier for the constant normal assumption.

The other is for $\kappa_{\vec{C}_a}(p_0)$ small. The square root term in (3.24) can be rearranged by factoring out a $\cos\theta(q_0)$ and using the equality $\sec^2\theta(q_0) - \tan^2\theta(q_0) = 1$ to obtain

$$\cos\theta(q_0)\sqrt{1 - f(p_0)\kappa_{\vec{C}_a}(p_0)\delta t(2 + f(p_0)\kappa_{\vec{C}_a}(p_0)\delta t)\tan^2\theta(q_0)} \tag{3.26}$$

This can be approximated using the first two terms of the Taylor series expansion for the square root ($\sqrt{1+x} \approx 1 + x/2$ for $x << 1$) and substituted back into (3.24) to obtain:

$$\begin{aligned}
\hat{\phi}_{\text{circ}}(q_0) &= -f(p_0)\cos\theta(q_0) - \cos\theta(q_0)f(p_0)(1 + \tfrac{1}{2}f(p_0)\kappa_{\vec{C}_a}(p_0)\delta t)\tan^2\theta(q_0) \\
&= -f(p_0)\cos\theta(q_0)(1 + \tan^2\theta(q_0)) - \tfrac{1}{2}f^2(p_0)\kappa_{\vec{C}_a}(p_0)\delta t\tan^2\theta(q_0)\cos\theta(q_0) \\
&= -\frac{f(p_0)}{\cos\theta(q_0)} - \tfrac{1}{2}f^2(p_0)\kappa_{\vec{C}_a}(p_0)\delta t\frac{\sin^2\theta(q_0)}{\cos\theta(q_0)}
\end{aligned} \tag{3.27}$$

We note that $\cos\theta(q_0) = \left\langle \vec{\mathcal{N}}_{\vec{C}_a}(p_0), \vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0) \right\rangle$, so as $\kappa_{\vec{C}_a}(p_0)$ goes to zero, this approaches the answer given in (3.23) for the linear approximation. Note also that the correction factor in (3.27) based on $\kappa_{\vec{C}_a}(p_0)$ produces the general effect we would expect on the example in Fig. 3.4. Compared with the estimate for the linear case, for $f(p_0) > 0$ and $\kappa_{\vec{C}_a}(p_0) > 0$, the correction factor increases the magnitude of $\hat{\phi}(q_0)$ to account for the extra distance needed to intersect the curve as it curves away from the linear approximation to $\vec{C}_a(p)$ around p_0. The opposite holds for $f(p_0) > 0$ and $\kappa_{\vec{C}_a}(p_0) < 0$ (which we illustrate Appendix B in Fig. B.2).

Note that computing this estimate is more complex than the linear approximation and can be unstable due to the division by $\kappa_{\vec{C}_a}(p_0)$ as many curves have sections with zero or small curvature. Therefore we typically employ the linear approximation when implementing the curve sampling algorithm. We can derive an existence condition for the reverse perturbation from the second-order approximation, a topic which we explore in the next section.

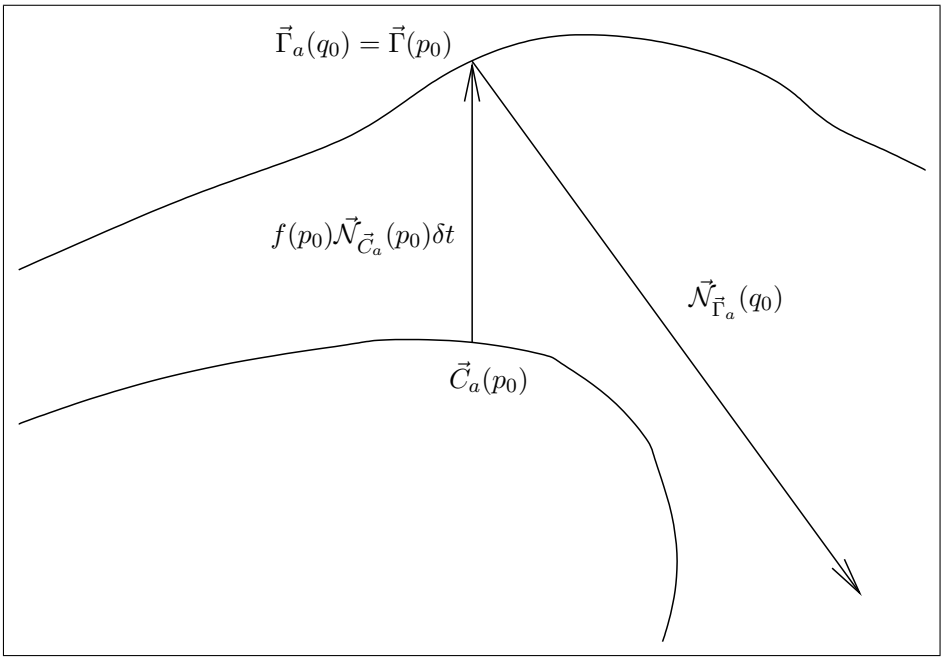


Figure 3.5. Illustration of a case when the reverse perturbation actually does not exist because no value of $\phi(q_0)$ will cause an intersection of the line along $\vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$ with the original curve $\vec{C}$.

■ 3.3.4 Existence Condition for the Reverse Perturbation

There is a possibility that for a given candidate sample $\vec{\Gamma}$, there does not exist a reverse perturbation ϕ such that a curve $\vec{C}_r$ (which is geometrically-identical to $\vec{C}$) can be formed. Such an instance is illustrated in Fig. 3.5. For this example, there is a large amount of curvature in $\vec{C}$ and the normal to the two curves are different enough so that tracing along the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$ will never intersect $\vec{C}$. Note that this is not a major issue for our sampling algorithm as this simply indicates that $\mathtt{q}(\vec{C}\,|\,\vec{\Gamma})$, the probability of generating $\vec{C}$ from $\vec{\Gamma}$, is 0. This then means that the candidate sample is rejected as the Hastings ratio $\eta(\vec{\Gamma}\,|\,\vec{C}) = 0$.

Nevertheless, it is useful to understand when the reverse perturbation does not exist because if this situation occurs frequently, time is wasted generating samples which will never be accepted. Using the result for the circle approximation in Sec. 3.3.3, we can derive a condition for when the reverse tracing along the normal will actually intersect $\vec{C}_a(p)$ and produce a valid $\phi(q_0)$. From (3.24), we can see that a real answer is given if and only if the part inside the square root is non-negative. This imposes the following

condition:

$$|\sin\theta(q_0)| < \frac{1}{|1 + f(p_0)\kappa_{\vec{C}_\mathrm{a}}(p_0)\delta t|} \ . \tag{3.28}$$

This implies then that the maximum allowable $\theta(q_0)$, the angle between the normal vectors at $\vec{C}_\mathrm{a}(p_0)$ and $\vec{\Gamma}_\mathrm{a}(q_0)$, is larger the smaller $f(p_0)\kappa_{\vec{C}_\mathrm{a}}(p_0)\delta t$ is. Not surprisingly, this means that smaller perturbations ($f(p_0)\delta t$) and straighter curves ($\kappa_{\vec{C}_\mathrm{a}}(p_0)$) lead to wider capture angles. This rule can help to choose an appropriate step size δt depending on how large we expect $\kappa_{\vec{C}_\mathrm{a}}(p)$ to be. Of course, this existence condition is only correct for curves with up to 2nd order derivatives. General curves with higher-order derivatives may cause deviations from this rule. This is especially true for larger values of $\theta(q_0)$ as this means that $\vec{C}_\mathrm{r}(q_0)$ will be further away from $\vec{C}_\mathrm{a}(p_0)$, and the effect of higher-order derivatives will be larger.

■ 3.3.5 Summary of Hastings Ratio Calculation

We can now combine all of the operations detailed above to compute the Hastings ratio and complete Step #3 of the overall curve sampling algorithm detailed on page 57:

1. Evaluate the target distribution values $\pi(\vec{\Gamma})$ and $\pi(\vec{C})$ (using update approaches when possible—see page 67).

2. Compute the discrete reverse perturbation $\boldsymbol{\phi}$ using either a direct backtracing operation or one of the approximate methods detailed in (3.21), (3.23), or (3.24).

3. Obtain the Hastings ratio (applying (3.17) and (3.20)):

$$\eta(\vec{\Gamma}\,|\,\vec{C}) = \frac{\pi(\vec{\Gamma})}{\pi(\vec{C})}\frac{\mathfrak{q}(\vec{C}\,|\,\vec{\Gamma})}{\mathfrak{q}(\vec{\Gamma}\,|\,\vec{C})} \approx \frac{\pi(\vec{\Gamma})}{\pi(\vec{C})}\exp\left(-\frac{1}{2\sigma^2}\left(\boldsymbol{\nu}^\mathrm{T}\boldsymbol{\nu} - \boldsymbol{n}^\mathrm{T}\boldsymbol{n}\right)\right) \tag{3.29}$$

   where $\boldsymbol{n}$ was the Gaussian random vector used to generate the forward perturbation and $\boldsymbol{\nu}$ is obtained from the reverse perturbation as $\boldsymbol{\nu} = \boldsymbol{H}^{-1}(\boldsymbol{\phi} - \boldsymbol{\mu}_{\vec{\Gamma}})$.

■ 3.4 Online Parameter Estimation

Most optimization-based curve evolution algorithms have data-related parameters in the energy functional which can either be specified by the user or dynamically learned while segmenting the image. Previously in our formulation, we assumed any parameters for π were known and fixed, and we made π implicitly dependent on the image and some model parameters. Online estimation of target distribution parameters is a useful

feature to make an algorithm less dependent on training data and user input. In this section we define parameters θ for a curve $\vec{C}$ and write the target distribution as $\pi(\vec{C} \,|\, I;\, \theta)$ to make the dependence on both I and θ explicit. Some examples of parameters in curve evolution energy functionals include mean parameters in the work of Yezzi *et al.* [126] and Chan and Vese [11]; smooth functions for the Mumford-Shah implementation by Tsai *et al.* [117]; and non-parametric probability densities by Kim *et al.* [58].

We propose two main approaches to incorporate an unknown θ into our curve sampling framework. One option is to change the problem of sampling from $\pi(\vec{C} \,|\, I;\, \theta)$ to that of sampling from a new target distribution $\tilde{\pi}(\vec{C} \,|\, I)$. This new distribution is related to π but is not dependent on θ. To use this method, we need to be able to efficiently compute $\tilde{\pi}$. A second option is to view θ as a random variable and augment the variable space to include both $\vec{C}$ and θ. This changes the target distribution to a joint distribution $\tilde{\pi}(\vec{C}, \theta \,|\, I)$. We compare implementations for these different methods in Sec. 4.4.

■ 3.4.1 Deterministic Transformation

Here we present a few simple ways to convert a distribution parameterized by θ into one which is not. This list is by no means comprehensive, and other options could be more appropriate depending on the model. One choice is to define $\tilde{\pi}(\vec{C} \,|\, I)$ as being proportional to the maximum value of π with respect to θ:

$$\tilde{\pi}_{\mathrm{ML}\theta}(\vec{C} \,|\, I) \propto \max_{\theta} \pi(\vec{C} \,|\, I;\, \theta) \ . \tag{3.30}$$

An equivalent view of this approach is that one first calculates the maximum likelihood (ML) estimate of θ (viewing the current value of $\vec{C}$ as an observed quantity):

$$\hat{\theta}_{\mathrm{ML}} = \arg\max_{\theta} \pi(\vec{C} \,|\, I;\, \theta) \ . \tag{3.31}$$

That estimate is then applied to compute $\tilde{\pi}_{\mathrm{ML}\theta}(\vec{C} \,|\, I) = \pi(\vec{C} \,|\, I;\, \hat{\theta}_{\mathrm{ML}})$. This approach is similar to what is normally done for gradient-based curve evolution which typically use a coordinate descent approach (where the energy E is alternately minimized with respect to $\vec{C}$ and θ). As an example, for the Chan-Vese energy functional, computing the ML parameter estimates is equivalent to calculating the mean intensities inside and outside the curve.

Another approach is to convert θ to a random variable and define a probability distribution $\mathrm{p}(\theta \,|\, I)$ (which can be dependent on the current image). This also makes the target distribution $\pi(\vec{C} \,|\, I, \theta)$ conditioned on θ instead of parameterized by it. We then define $\tilde{\pi}(\vec{C} \,|\, I)$ as the expected value of the distribution with the expectation taken over θ:

$$\tilde{\pi}_{\text{mean}}(\vec{C} \,|\, I) \propto \mathrm{E}_{\theta|I}\left[\pi(\vec{C} \,|\, I, \theta)\right] \quad . \tag{3.32}$$

This is equivalent to constructing the joint probability distribution $\pi(\vec{C} \,|\, I, \theta)\mathrm{p}(\theta \,|\, I) = \mathrm{p}(\vec{C}, \theta \,|\, I)$ then integrating out or summing over θ to find the marginal distribution with respect to $\vec{C}$. This option is most feasible when θ is discrete, so the expectation can be computed by evaluating π for each possible value of θ and summing the values (weighted by $\mathrm{p}(\theta)$). This is also feasible if the distribution for $\mathrm{p}(\theta \,|\, I)$ is a conjugate prior for $\pi(\vec{C} \,|\, I, \theta)$.

Rather than taking expectations of the probability density, we can take expectations of the log probability instead:

$$\log \tilde{\pi}_{\text{mean}}(\vec{C} \,|\, I) \propto \mathrm{E}_{\theta|I}\left[\log \pi(\vec{C} \,|\, I, \theta)\right] \quad . \tag{3.33}$$

For some π, this may be easier to compute. This typically results in a different $\tilde{\pi}_{\text{mean}}$ than in (3.32).

Choosing which version to use depends heavily on the structure of a particular problem. It is preferable that $\tilde{\pi}$ be computable in closed form (*i.e.*, an iterative algorithm such as gradient descent is not needed to compute (3.30) or a numerical or Monte Carlo integration technique is not needed to evaluate (3.32) or (3.33)). If this is not possible, the augmented variable approach which we present next may be more straightforward to implement.

■ 3.4.2 Augmentation of the Variable Space

Another approach to incorporating parameter estimation is not only to view the parameter variable as random but also to augment the state space to include both it and $\vec{C}$. This results in a new target distribution $\tilde{\pi}_{\text{aug}}(\vec{C}, \theta \,|\, I)$ which is a joint probability distribution over both $\vec{C}$ and θ. This can be decomposed into likelihood and prior terms as $\tilde{\pi}_{\text{aug}}(\vec{C}, \theta \,|\, I) \propto \mathrm{p}(I \,|\, \vec{C}, \theta)\mathrm{p}(\vec{C}, \theta)$, and if $\vec{C}$ and θ are independent, this can be written as:

$$\tilde{\pi}_{\text{aug}}(\vec{C}, \theta \,|\, I) \propto \mathrm{p}(I \,|\, \vec{C}, \theta)\mathrm{p}(\vec{C})\mathrm{p}(\theta) \quad . \tag{3.34}$$

The augmented state $\{\vec{C}, \theta\}$ also leads to an augmented candidate sample $\{\vec{\Gamma}, \chi\}$ for which we must create a revised proposal distribution $\mathtt{q}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)})$ and acceptance function $\mathtt{a}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)})$. We continue to use the Metropolis-Hastings acceptance rule, so the only change to $\mathtt{a}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)})$ is to use the revised target and proposal distributions in the computation of the Hastings ratio.

Instead of constructing proposal distributions which are based on the full augmented $\mathtt{q}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)})$, it may be more convenient to decompose it into simpler substructures. One choice is to chain together two separate proposal distributions $\mathtt{q}_1$ and $\mathtt{q}_2$ to form the overall proposal distribution:

$$\mathtt{q}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)}) = \mathtt{q}_2(\chi^{(t+1)} \,|\, \vec{\Gamma}^{(t+1)}, \vec{C}^{(t)}, \theta^{(t)}) \mathtt{q}_1(\vec{\Gamma}^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)}) \ . \quad (3.35)$$

In this scenario, we first sample $\vec{\Gamma}^{(t+1)}$ from $\mathtt{q}_1$, then we use that result in sampling $\chi^{(t+1)}$ from $\mathtt{q}_2$. Various conditional independence assumptions can be made to simplify this further. For instance, if we assume that $\vec{\Gamma}$ is independent of θ given $\vec{C}$ and χ is independent of both $\vec{C}$ and θ given $\vec{\Gamma}$, we obtain:

$$\mathtt{q}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)}) = \mathtt{q}_2(\chi^{(t+1)} \,|\, \vec{\Gamma}^{(t)}) \mathtt{q}_1(\vec{\Gamma}^{(t+1)} \,|\, \vec{C}^{(t)}) \ . \quad (3.36)$$

This formulation allows us, *e.g.*, to use our standard curve perturbation defined in Sec. 3.2 for $\mathtt{q}_1$ and define samples from $\mathtt{q}_2$ as $\hat{\chi}_{\mathrm{ML}}^{(t+1)}$+noise (where $\hat{\chi}_{\mathrm{ML}}^{(t+1)}$ is the maximum likelihood estimate of χ given $\vec{\Gamma}^{(t+1)}$ as in (3.31)). This is similar to the formulation in (3.30) with additional randomness added to the parameter estimation.

A different simplification is to assume that $\vec{\Gamma}$ is independent of θ given $\vec{C}$ and χ is independent of $\vec{C}$ and $\vec{\Gamma}$ given θ. This results in

$$\mathtt{q}(\vec{\Gamma}^{(t+1)}, \chi^{(t+1)} \,|\, \vec{C}^{(t)}, \theta^{(t)}) = \mathtt{q}_1(\vec{\Gamma}^{(t+1)} \,|\, \vec{C}^{(t)}) \mathtt{q}_2(\chi^{(t+1)} \,|\, \theta^{(t)}) \ . \quad (3.37)$$

This is a convenient choice because we can use our standard curve perturbation and independently define a perturbation for χ (*e.g.*, $\chi^{(t+1)} = \theta^{(t)} + \text{noise}$).

Chapter 4

2D Curve Sampling Results

In this chapter, we present results for our curve sampling algorithm on a variety of synthetic and real examples. Typically for each application, we generate 1000 samples $\{\vec{C}_i\}_{i=1}^{1000}$ from a target distribution[1]. With this many samples, a user cannot readily look at all of them and immediately discern useful information. In Sec. 4.1 we begin by describing some techniques to summarize the information contained in the output of our sampling algorithm. This includes the use of most-probable samples and a type of confidence bound on the location of the segmentation. We then demonstrate the algorithm on a number of examples. We begin with probability distributions which are just shape models (without an observed image) in Sec. 4.2, and then continue with synthetic noisy images and prostate magnetic resonance (MR) images in Sec. 4.3. Examples using the online parameter estimation extensions developed in Sec. 3.4 are shown in Sec. 4.4. We conclude in Sec. 4.5 by exploring the convergence behavior of our method and develop an empirical heuristic to judge convergence and establish an approximate stopping criterion for the iterative sampling procedure.

■ 4.1 Visualizing Curve Samples

In this section, we assume we already have N samples drawn from a probability distribution $\pi(\vec{C})$. We discuss specific choices for this target distribution for different applications in later sections of this chapter. To display the output of the sampling method, we will use four main visualization approaches:

1. Plotting the most probable samples (*e.g.*, Fig. 4.2(a)). While we do not always

[1]Each sample is generated independently of the others, so parallel computing systems can easily be used to increase sample throughput. All images used are 256×256, and each example was run for 2000-5000 iterations per sample depending on the complexity of the sample and how close the initialization is to the true boundary location. Computation time per sample is approximately 15-40 seconds.

present direct comparisons to gradient-based curve evolution methods, the most probable samples (*i.e.*, the samples which have the largest target distribution probability $\pi(\vec{C}_i)$) can be viewed as a proxy for what an optimization approach that can find global optima would capture.

2. Histogram images (*e.g.*, Fig. 4.2(c)). For each $\boldsymbol{x} \in \Omega$, $\Phi(\boldsymbol{x})$ is the percentage of curves $\vec{C}_i$ for which $\boldsymbol{x}$ is inside the curve (*i.e.*, $\Psi_{\vec{C}_i}(\boldsymbol{x}) < 0$). This is thus the marginal distribution over segmentation labels at each $\boldsymbol{x}$.

3. Marginal confidence bounds (*e.g.*, Fig. 4.2(c)). Given a histogram image $\Phi(\boldsymbol{x})$, we plot the level contours (the kth level contour is the set of points $\boldsymbol{x}$ such that $\Phi(\boldsymbol{x}) = k$). For instance, all points such that $\Phi(\boldsymbol{x}) = 50\%$ can be viewed as analogous to a median contour (half of the time these points are inside the curve, half of the time they are outside). Other level contours can be viewed as confidence bounds (*e.g.*, the 10% confidence bound is the contour outside of which all pixels were inside fewer than 10% of the samples).

4. Principal components analysis (PCA)-based shape distributions. In this approach, we use PCA-based techniques (discussed in Sec. 2.3.2) to extract principal modes of variation of the shape samples. This allows us to better account for the correlation structure of our probability distributions over curves. We describe this method more fully in Sec. 5.3.5 and demonstrate the approach on a geophysical inversion problem.

These visualizations are only possible because we are able to draw a large number of statistically-valid samples from the posterior distribution. Confidence bounds have been used in some previous image segmentation or reconstruction approaches [125,128], but those dealt with parametric shape representations (so the uncertainty was over a finite set of parameters). It is important to note that our confidence representations are based on marginal statistics from non-parametric shape distributions.

In some of the examples discussed in this chapter, the global optimum does not provide a satisfactory segmentation (even for the synthetic image which is generated exactly according to the data model used in the segmentation process) due to the high levels of noise and clutter in the images. As an analogy, consider the example of having N samples drawn from a Gaussian distribution $\mathrm{N}(\mu, \sigma^2)$. The maximum-likelihood (ML) estimate of the mean is simply the average of the samples, and the variance of

this estimate is σ^2/N. Therefore we would not be surprised for the ML estimate to be incorrect by $\pm\sigma/\sqrt{N}$ (which can be significantly different from the true answer if $\sigma/\sqrt{N} >> \mu$). Similarly, for image segmentation, we would expect some deviation between the most likely answer according to the model and the true correct answer. Having access to higher-order statistics such as error variances can help constrain the range of likely answers and provide guidance on how confident we should be in our results.

■ 4.2 Shape Distributions

In this section we demonstrate our curve sampling algorithm for sampling from shape probability distributions based on a shape distance function known as symmetric area difference (SAD). This distance is defined to be the area of the image domain Ω for which two curves $\vec{C}_1$ and $\vec{C}_2$ do not agree as to the proper segmentation label [57, 73]. We can write this mathematically as:

$$\mathrm{d}_{\mathrm{SAD}}(\vec{C}_1, \vec{C}_2) = \iint_{\mathcal{R}_{\vec{C}_1} \setminus \mathcal{R}_{\vec{C}_2}} \mathrm{d}\boldsymbol{x} \quad + \iint_{\mathcal{R}_{\vec{C}_2} \setminus \mathcal{R}_{\vec{C}_1}} \mathrm{d}\boldsymbol{x} \ . \tag{4.1}$$

SAD is not an ideal shape pseudometric because it is not really geometric and the computation is local in nature. For instance, a circle and an annulus with a very small hole will be very close under the SAD distance, but they have entirely different topologies. Also a circle and a circular-object with very jagged boundaries will also be quite similar under the SAD distance though local features such as curvature will be very different. Depending on the application, this may or may not be desirable. Coupling SAD with a smoothness prior may ameliorate this to some extent, though this is only possible if we actually want smooth curves.

For the examples we show below, the probability distributions are parameterized by one or more target shapes. Typically in real applications one would combine a shape model along with a data model to segment an observed image. Here we sample just from the shape models. This allows us to demonstrate the general behavior of our sampling algorithm on a relatively simple single-mode case as well as on a multi-modal example which an optimization-based algorithm would have difficulty characterizing. Note that while we use SAD as the distance function here, arbitrary distance functions can be substituted without substantially altering the algorithm.

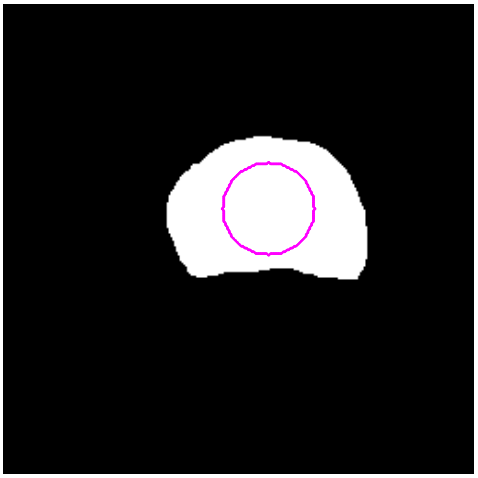


Figure 4.1. Target shape (in white) and initial curve (in magenta) for SAD target.

■ 4.2.1 Single Curve Target

In this section, we define a probability distribution with respect to a single target curve $\vec{C}_T$. An energy functional is formed as the combination of a shape distance and a regularizing curve length prior:

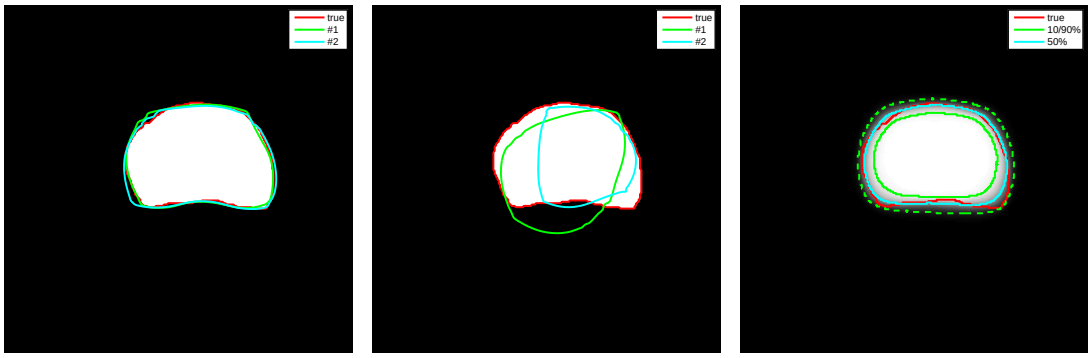
$$\mathrm{E}(\vec{C};\vec{C}_T) = \beta\,\mathrm{d}_{\mathrm{SAD}}(\vec{C},\vec{C}_T)) + \alpha \oint_{\vec{C}} \mathrm{d}s \ . \tag{4.2}$$

We then exponentiate the negative of the energy functional to construct our target probability distribution:

$$\pi(\vec{C};\vec{C}_T) \propto \exp(-\mathrm{E}(\vec{C};\vec{C}_T)) \propto \exp(-\beta\,\mathrm{d}_{\mathrm{SAD}}(\vec{C},\vec{C}_T))\exp(-\alpha \oint_{\vec{C}} \mathrm{d}s) \ . \tag{4.3}$$

The shape distance-based term in (4.3) is maximized when $\vec{C} = \vec{C}_T$. The more different $\vec{C}$ and $\vec{C}_T$ are (according to the SAD distance), the smaller this term will be. We note that when using gradient descent to minimize an energy functional $\mathrm{E}(\vec{C})$, it is equivalent to minimize $k\,\mathrm{E}(\vec{C};\vec{C}_T)$ for some $k > 0$. When sampling though, scaling the energy (which is the negative log of π) does affect how sharp or flat the target probability distribution is, so the absolute values of α and β matter, not just the relative values.

In Fig. 4.1, we show a target shape in white (which is taken from a radiologist-segmented prostate MR). Note that this is not an actual image which is given to our sampling algorithm but simply a visual representation of the target curve $\vec{C}_T$ which



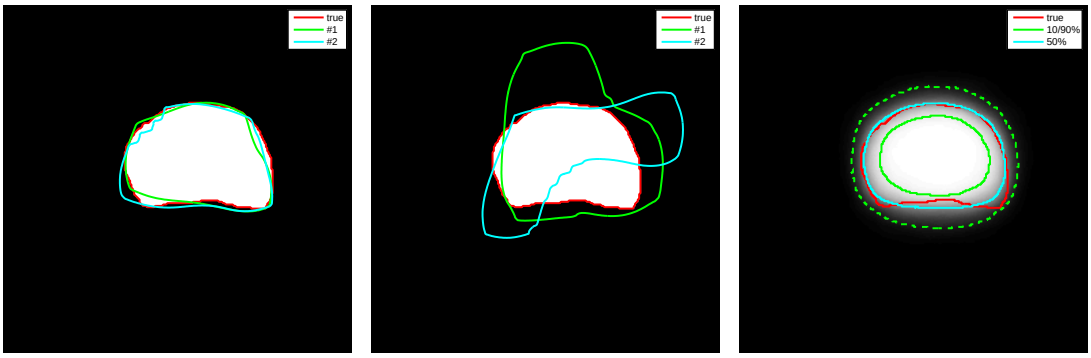
(a) (b) (c)

Figure 4.2. Results for SAD target with $\alpha = 1$ and $\beta = 0.01$. Target curve plotted with (a) two most likely samples; (b) two least likely samples; and (c) marginal confidence bounds and histogram image.

is used to parameterize π. In the figure, we also show the curve used to initialize the MCMC curve sampler in magenta. For this application, our experimental results are insensitive to the choice of initial conditions.

We show the results for parameter settings of $\alpha = 1$ and $\beta = 0.01$ in Fig. 4.2. The most probable samples are displayed in Fig. 4.2(a) and the least likely samples in Fig. 4.2(b). The latter are shown as an illustration of the range of samples which are actually possible under the curve sampling framework, but they are probably of little practical use. The most probable curves are quite similar to the target curve. In Fig. 4.2(c) we show the histogram image, median curve, and 10/90% confidence bounds. The 10% bound is the dashed green line, and the 90% bound is the solid green line. The true target $\vec{C}_T$ is quite close to the median curve, though there are problems in areas of high curvature (the bottom corners) and large negative curvature (the bottom section). Still, the 90% and 10% bounds largely bracket the target curve $\vec{C}_T$.

Now we run the same experiment except with the scaling parameters divided by 2: $\alpha = 0.5$ and $\beta = 0.005$. Decreasing the scaling parameters flattens π. This means that we are more likely to accept a candidate sample which decreased π simply because the relative difference in probability has been lessened (*e.g.*, consider a pair of curves $\vec{C}^{(t)}$ and $\vec{\Gamma}^{(t+1)}$ where $\pi(\vec{\Gamma}^{(t+1)})/\pi(\vec{C}^{(t)}) = 0.5$ for a given choice of α and β; if we instead divide α and β by 2, $\pi(\vec{\Gamma}^{(t+1)})/\pi(\vec{C}^{(t)}) = \sqrt{0.5} \approx 0.707$, and the Hastings ratio similarly increases). This is evident in the results in Fig. 4.3. Both the most likely samples and the least likely samples show more variation, the histogram image in Fig. 4.3(c) is more



(a) (b) (c)

Figure 4.3. Results for SAD target with $\alpha = 0.5$ and $\beta = 0.005$. Target curve plotted with (a) two most likely samples; (b) two least likely samples; and (c) marginal confidence bounds and histogram image.

diffuse, and the distance between the confidence bounds has increased as well. All of these behaviors show that the uncertainty level of the samples has increased.

■ 4.2.2 Shape Distributions Using Parzen Densities

The experiment in the previous section had a single curve target resulting in a unimodal target distribution without local maxima. If we have multiple target curves, we can use the idea from Kim *et al.* [57] and form a Parzen density distribution (see Sec. 2.3.3) with exponentiated $\mathrm{d_{SAD}}$ as the kernel function:

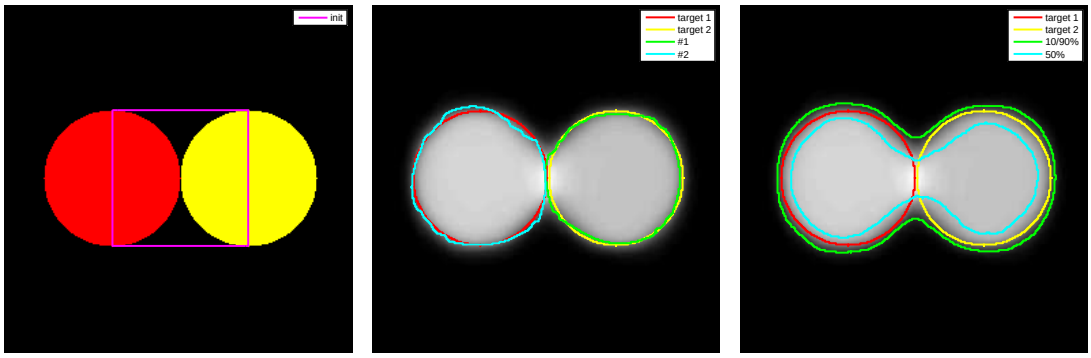
$$\mathrm{K}(\vec{C}, \vec{C}_i) = \exp(-\beta\, \mathrm{d_{SAD}}(\vec{C}, \vec{C}_i)) \ . \tag{4.4}$$

Coupled with a curve length prior, this results in an overall probability distribution of

$$\pi(\vec{C};\, \{\vec{C}_i\}_{i=1}^{M}) \propto \left(\sum_{i=1}^{M} \exp(-\beta\, \mathrm{d_{SAD}}(\vec{C}, \vec{C}_i)) \right) \exp(-\alpha \oint_{\vec{C}} \mathrm{d}s) \ . \tag{4.5}$$

Note that if $M = 1$, the target distribution here is identical to the one in (4.3) with $\vec{C}_T = \vec{C}_1$. The target distribution defined in (4.5) is clearly multi-modal as there is a peak for each of the target curves $\vec{C}_i$. We now demonstrate how our sampling algorithm can naturally represent the different modes.

In Fig. 4.4(a), we show a 256×256 example with two target curves which are circles (one in red, one in yellow) that are horizontally translated from each other. Again,



(a) (b) (c)

Figure 4.4. Results for Parzen density with SAD distance, $\alpha = 1$ and $\beta = 0.07$. (a) Initialization and two target curves. (b) Two most likely samples. (c) Marginal confidence bounds and histogram image.

this is simply to illustrate what the target curves look like, and this is not an actual image which is given to the sampling algorithm. We initialize $\vec{C}^{(0)}$ as a square centered between the two circles. The most probable curves in Fig. 4.4(b) are quite close to the target curves, and we actually do obtain one curve for each target curve for this realization.

The histogram image and marginal confidence bounds in Fig. 4.4(c) are not as informative though. We can see that the highest probability area (*i.e.*, the area with the largest $\Phi(\boldsymbol{x})$ values) is actually where the two circles meet, not in the middle of either circle. This is because curves attracted to either the right or the left curve can contribute counts to the histogram image in that area. Also while the 10% confidence bound does enclose both circles, the median contour is a shape that does not resemble either target shape and the 90% confidence bound does not even exist (because no points exist which are inside at least 90% of the samples). This is because the curves are attracted to either target with about 50% probability, so the highest histogram image value is only around 65%. Not surprisingly, we cannot (and should not) say with strong confidence that any specific pixel belongs inside of the curve.

The fundamental problem here is that our curves are global geometric objects, and we are attempting to describe the aggregate information using marginal statistics. Naturally a lot of structure is being lost. For instance, if a pixel in the middle of the left circle is inside a given curve sample, we can be pretty sure that a pixel in the middle of the right circle is not inside the curve. This is somewhat akin to looking at the mean

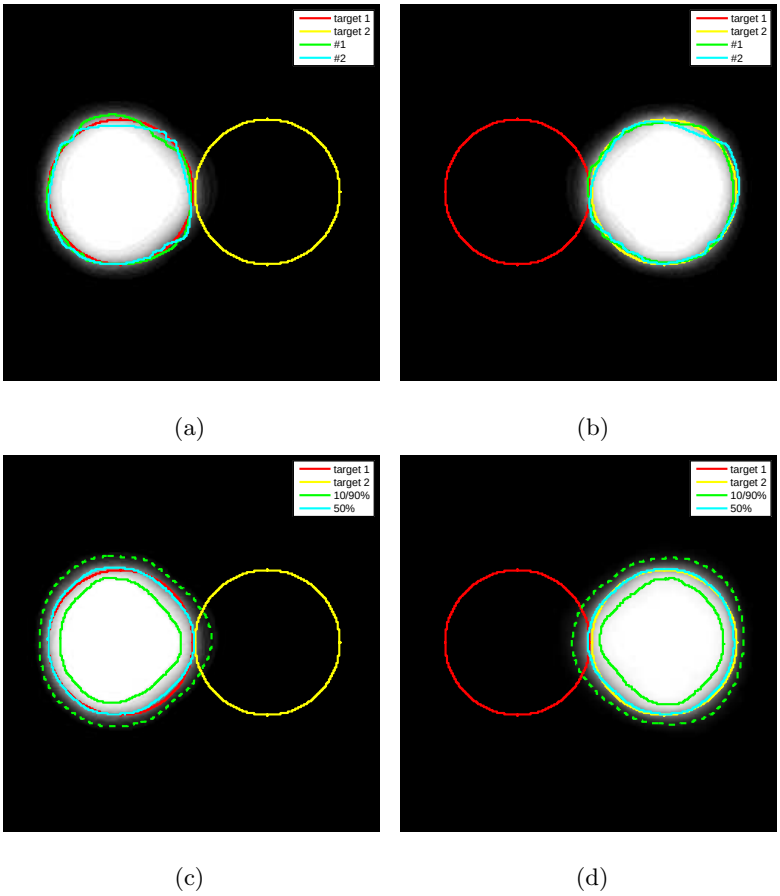


Figure 4.5. Results for the Parzen SAD example after clustering the resulting samples. (a)-(b) Two most likely samples for each cluster. (c)-(d) Marginal confidence bounds and histogram images.

of a Gaussian mixture composed of two Gaussian kernels with means at ± 1. The mean is 0, but that is not very informative.

What we can do then is cluster the samples so the ones from the left circle and from the right circle are separated. There are a number of popular clustering algorithms [5] (*e.g.*, k-means, spectral clustering), but their use is complicated by the high dimensionality of our problem and the nonlinear nature of the manifold on which closed curves exist. Therefore we use a simple supervised clustering approach here. We first manually choose a curve $\vec{C}_{\text{left}}$ corresponding to the left circle and another curve $\vec{C}_{\text{right}}$ corresponding to the right circle. All of the samples $\vec{C}_i$ for $i \in \{1, 2, \ldots, N\}$ are then compared to these two representative curves using symmetric area difference. A curve $\vec{C}_i$ is assigned to the left cluster if $\mathrm{d_{SAD}}(\vec{C}_i, \vec{C}_{\text{left}}) < \mathrm{d_{SAD}}(\vec{C}_i, \vec{C}_{\text{right}})$, otherwise it is assigned to the right cluster. This method is effective (and would be so using virtually any distance metric) because there is such a clear difference between the two clusters.

The results of doing this clustering are shown in Fig. 4.5. The most likely samples are close to the target curves, and the histogram images now look much more similar to the histogram image in Sec. 4.2.1. An alternative viewpoint is to look at these as conditional histogram images (*i.e.*, $\Phi(\boldsymbol{x} \,|\, \boldsymbol{x}_0 \in \mathcal{R}(\vec{C})) = \mathrm{P}(\boldsymbol{x} \in \mathcal{R}(\vec{C}) \,|\, \boldsymbol{x}_0 \in \mathcal{R}(\vec{C}))$ for some point $\boldsymbol{x}_0 \in \Omega$).

Note that there is a slight skew for curves in the left cluster to the right and similarly for the curves in the right cluster (as seen in Fig. 4.5(c)-(d)). This is because segmentation errors that include the other target curve region receive higher likelihood. For instance, say we have a curve $\vec{C}$ which is exactly equal to the left target curve $\vec{C}_1$. We consider two perturbations of $\vec{C}$: a curve $\vec{\Gamma}_R$ which has a slight bulge on the right side of the object or $\vec{\Gamma}_L$ which has a symmetric bulge on the left side. Then $\mathrm{K}(\vec{\Gamma}_R, \vec{C}_1) = \mathrm{K}(\vec{\Gamma}_L, \vec{C}_1)$, but $\mathrm{K}(\vec{\Gamma}_R, \vec{C}_2) > \mathrm{K}(\vec{\Gamma}_L, \vec{C}_2)$ because there will be a slight overlap with $\vec{C}_2$. This means that $\pi(\vec{\Gamma}_R; \vec{C}_1, \vec{C}_2) > \pi(\vec{\Gamma}_L; \vec{C}_1, \vec{C}_2)$, and we are more likely to accept $\vec{\Gamma}_R$.

■ 4.3 Image-based Examples

In this section, we present examples which incorporate image-based data terms into the target probability distribution. Note that while the exact nature of the problems here are quite different from Sec. 4.2, the overall curve sampling framework does not change significantly except for the need to compute a different target probability distribution.

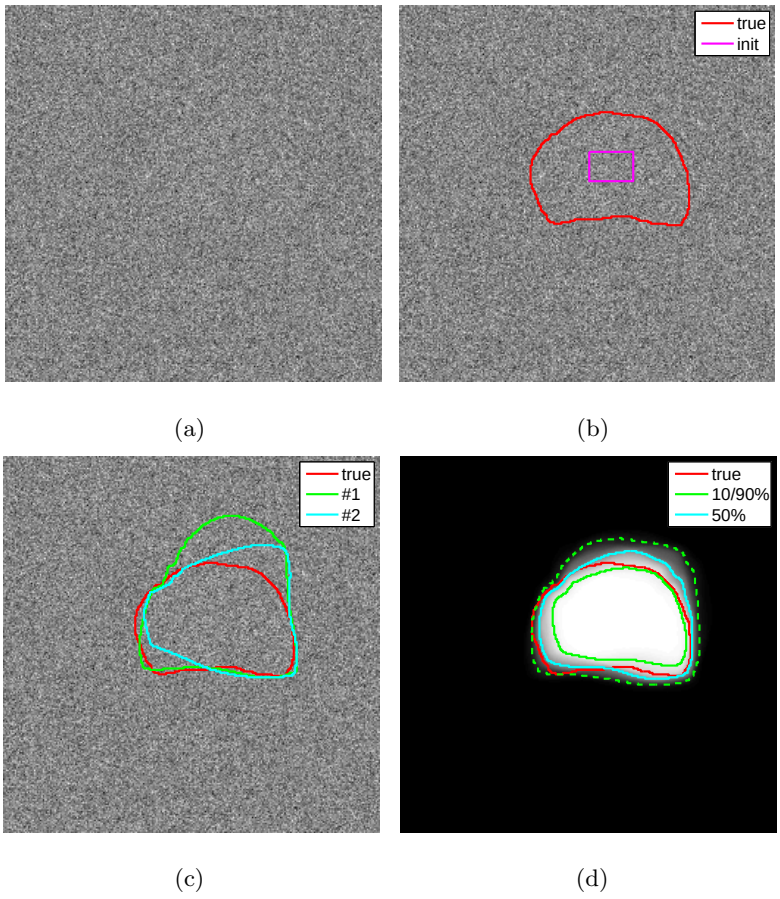


Figure 4.6. Results for the synthetic Gaussian example. (a) Observed image. (b) True boundary and initial curve. (c) Two most likely samples. (d) Marginal confidence bounds and histogram image.

In contrast, when using a gradient-based method, we would need to derive a separate gradient flow equation for each example.

We begin this section with a synthetic example consisting of a piecewise-constant image corrupted by very high levels of Gaussian noise. This is followed by an example where we segment a noisy prostate image with a data model using non-parametric image statistics.

■ 4.3.1 Synthetic Gaussian Noise

For this example, we generate a synthetic image I exactly according to the implicit Chan-Vese model:

$$I(\boldsymbol{x}) = m(\boldsymbol{x}) + w(\boldsymbol{x}) \tag{4.6}$$

where $m(\boldsymbol{x})$ is piecewise constant (0 outside the curve, 1 inside the curve) and $w(\boldsymbol{x})$ is white Gaussian noise with $\sigma_w = 10$ (for a SNR of -20 dB). Our energy functional then is (2.34) (which we slightly modify here to incorporate the noise variance):

$$\mathrm{E}(\vec{C}\,|\,I) = \iint\limits_{\mathcal{R}_{\vec{C}}} \left(\frac{I-m_1}{\sigma_w}\right)^2 \mathrm{d}\boldsymbol{x} + \iint\limits_{\mathcal{R}^c_{\vec{C}}} \left(\frac{I-m_0}{\sigma_w}\right)^2 \mathrm{d}\boldsymbol{x} + \alpha \oint\limits_{\vec{C}} \mathrm{d}s \tag{4.7}$$

with the mean values inside and outside the curve known *a priori*. This results in a target distribution:

$$\pi(\vec{C}\,|\,I) \propto \exp(-\mathrm{E}(\vec{C};I)) \ . \tag{4.8}$$

The observed image is shown in Fig. 4.6(a) with no readily-apparent structure due to the low SNR. The true location of the contour (which is a 2D prostate slice as in the example in Sec. 4.2.1) is shown in Fig. 4.6(b) in red with the initial curve shown in magenta. In this example, the most-probable samples shown in Fig. 4.6(c) are actually not close to the true boundary in the upper-right corner of the object. While the model is correct, the noise levels in this image are high enough to create regions where the incorrect segmentation is simply more likely under the model. Note, though, that the marginal confidence bounds in Fig. 4.6(d) show that the true contour location is mostly bracketed between the 90% and 10% confidence contours. The median contour is also quite close to the true boundary location indicating that the typical sample is perhaps providing more useful information than the most-likely sample. There is also a more diffuse histogram image (and a larger gap between the confidence bounds) in the upper right corner of the prostate which indicates greater uncertainty in that portion of the curve. This could be useful in providing guidance to an expert as to which part of the segmentation is least trustworthy and perhaps needs expert intervention.

■ 4.3.2 Prostate MR

Here we present results on a noisy T1-weighted prostate MR image. The images were obtained from a prostate MR image captured with endorectal-pelvic phased-array surface coils [46], and the bias field was removed using the technique in [26, 27]. Gaussian

noise was added to then simulate a T1-weighted body coil image (which has a low SNR). We assume that pixels are independent and identically distributed (iid) given the curve and learn (from segmented training data) non-parametric histogram distributions $\mathrm{p}(I(\boldsymbol{x}) \,|\, 0)$ and $\mathrm{p}(I(\boldsymbol{x}) \,|\, 1)$ (shown in Fig. 4.7(a)) to specify the pixel intensity distribution outside and inside the curve respectively. Note that the densities inside and outside the curve are very similar to each other. This is an advantage for sampling methods which are designed to cope with this type of uncertainty whereas optimization-based methods are likely to become caught in local minima. The overall data likelihood term is

$$\mathrm{p}(I \,|\, \vec{C}) = \prod_{\boldsymbol{x}} \mathrm{p}\left(I(\boldsymbol{x}) \,|\, \mathcal{H}(\Psi_{\vec{C}}(\boldsymbol{x}))\right) \tag{4.9}$$

with $\mathcal{H}$ the Heaviside function. Adding in a standard curve length penalty as the prior results in an overall target distribution of

$$\pi(\vec{C} \,|\, I) \propto \mathrm{p}(I \,|\, \vec{C}) \exp(-\beta \, \mathrm{d}_{\mathrm{SAD}}(\vec{C}, \vec{C}_i)) \ . \tag{4.10}$$

The histogram image and the marginal confidence bounds in Fig. 4.7(d) show that this distribution has three primary modes: one is around the correct prostate segmentation (the red contour); another segments just the rectum (the dark region located below the prostate); and the third encompasses both the prostate and the rectum. As can be seen in Fig. 4.7(c), the curves that segment the two regions together are actually from the most likely mode. The reason for this can be seen in the image intensity distributions. Due to the noise and our simple iid model, even though the rectum appears to be darker than the prostate, all pixel intensities below some threshold (approximately a value of 350) result in a higher likelihood for that pixel to be inside the curve. Hence the most likely curves are actually the ones that segment the prostate and rectum together.

Without having any additional *a priori* information, it would be difficult to say which of these three scenarios is the correct one. In fact, it is possible in some applications where multiple modes all provide reasonable explanations of the data. One approach we can take here is to utilize the information our sampling procedure provides to us. While the aggregate marginal statistics do not appear to be providing very useful information (though the 90% confidence boundary is located within the true prostate boundary), it is straightforward to again cluster the samples into prostate-only, rectum-only, and prostate and rectum segmentations. An expert user or a shape-driven classifier could then pick the best cluster to use. We show the most-likely samples and the marginal confidence boundaries for the prostate-only cluster in Fig. 4.8.

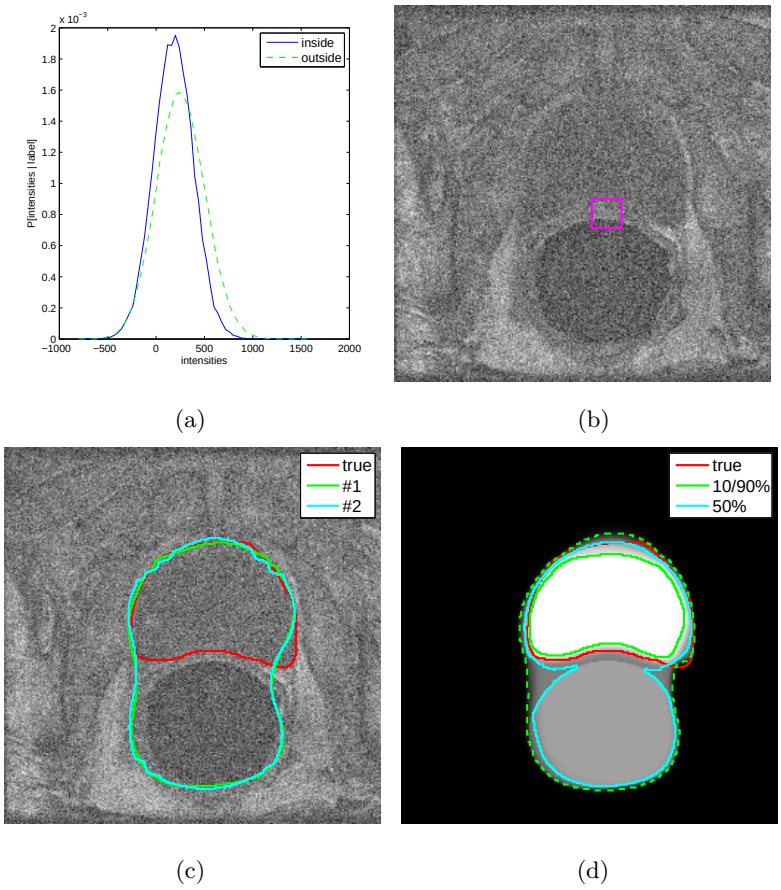
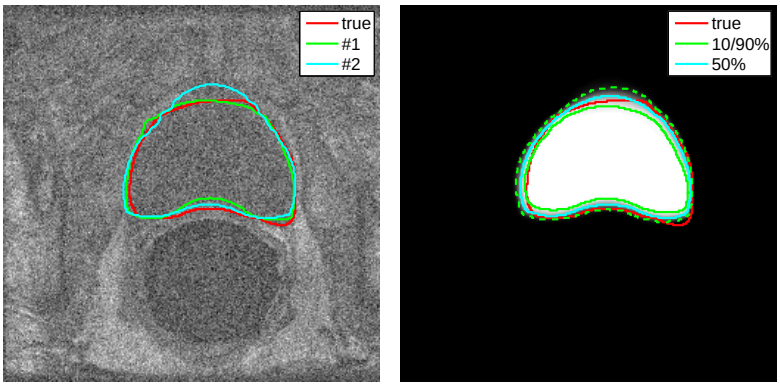


Figure 4.7. Prostate segmentation using non-parametric intensity distributions. (a) Pixel intensities for each class. (b) Initial curve. Expert segmentation with (c) two most likely samples and (d) marginal confidence bounds and histogram image.



(a) (b)

Figure 4.8. Clustered results for the prostate sampling problem. (a) Most likely samples and (b) histogram image and confidence bounds for prostate-only cluster.

Note that the fundamental reason for the multiple modes in this example is due to model deficiency as clearly there is structure outside of the curve which an iid assumption cannot adequately model. Another approach to correct this issue would be to construct a model that evolves two curves simultaneously, one for the prostate and one for the rectum. This would allow more separation based on pixel intensity. It would also be possible to construct joint shape models as in Tsai *et al.* [114, 116] or Yang and Duncan [124] to further take advantage of the structure in this image.

■ 4.4 Online Parameter Estimation

As discussed in Sec. 3.4, we can extend our basic curve sampling algorithm to incorporate online parameter estimation using both deterministic and stochastic approaches. In Sec. 4.3.1, we presented results on a synthetic noisy image using the Chan-Vese energy functional where we assumed the parameters (the mean values inside and outside the curve) were known. We will refer to this case as the "Known θ" version. Here we demonstrate the online parameter estimation methodology on the same Gaussian noise example using three techniques described previously: the maximization version (which we refer to as "ML θ") in (3.30); the noisy maximization version ("Noisy ML θ") in (3.36); and the independent perturbations ("Independent Perturbations") in (3.37).

■ 4.4.1 Experimental Setup

For the Gaussian noise example, the model parameters $\theta = [m_0, m_1]^{\mathrm{T}}$ are the mean intensities inside and outside the curve. As before, the true values of the parameters are $\theta^* = [0, 1]^{\mathrm{T}}$ and the standard deviation of the image noise is $\sigma_w = 10$. This again results in the signal-to-noise ratio (SNR) of the image being -20 dB. The Chan-Vese energy functional (defined in (4.7)), again, is:

$$\mathrm{E}(\vec{C} \,|\, I; \theta) = \iint_{\mathcal{R}_{\vec{C}}} \left(\frac{I - m_1}{\sigma_w} \right)^2 \mathrm{d}\boldsymbol{x} + \iint_{\mathcal{R}^c_{\vec{C}}} \left(\frac{I - m_0}{\sigma_w} \right)^2 \mathrm{d}\boldsymbol{x} + \alpha \oint_{\vec{C}} \mathrm{d}s \ . \tag{4.11}$$

The base target distribution $\pi(\vec{C} \,|\, I; \theta)$ is then proportional to $\exp(-\mathrm{E}(\vec{C} \,|\, I; \theta))$, and this distribution is used to create the modified versions for the online parameter estimation algorithms.

For the ML θ version, it is easy to show that the ML estimate of θ given a curve $\vec{C}$ is simply the sample averages inside and outside the curve (similar formulations can be found in Chan and Vese [11] and Yezzi *et al.* [126]):

$$\hat{\theta}_{\mathrm{ML}} = \begin{pmatrix} \frac{1}{A^c_{\vec{C}}} \iint_{\mathcal{R}^c_{\vec{C}}} I(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ \frac{1}{A_{\vec{C}}} \iint_{\mathcal{R}_{\vec{C}}} I(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \end{pmatrix} \tag{4.12}$$

where $A_{\vec{C}}$ is the area inside the curve and $A^c_{\vec{C}}$ is the area outside the curve. Following the formulation in (3.30), this results in a modified target distribution of:

$$\tilde{\pi}_{\mathrm{ML}\theta}(\vec{C} \,|\, I) \propto \max_{\theta} \pi(\vec{C} \,|\, I; \theta) = \pi(\vec{C} \,|\, I; \hat{\theta}_{\mathrm{ML}}) \ . \tag{4.13}$$

Computing the Hastings ratio is then largely unchanged compared to our original method: the proposal distribution is identical and the target distribution simply uses the ML estimate of the model parameters instead of specified fixed model parameters.

For the augmented variable versions (Noisy ML θ and Independent Perturbations), we need to specify a prior distribution for θ. The choice of this can obviously affect the final results given that it alters the target distribution. We choose to make m_0 and m_1 independent and uniformly distributed between $-M$ and M (where M is a large constant) so the prior on θ does not impose a strong bias on the results. This results in the following target distribution (using the definition in (3.34)):

$$\tilde{\pi}_{\mathrm{aug}}(\vec{C}, \theta \,|\, I) \propto \begin{cases} \frac{1}{(2M)^2} \pi(\vec{C} \,|\, I; \theta) & : \quad |m_i| \leq M \ \forall \ i, \\ 0 & : \quad \text{otherwise.} \end{cases} \tag{4.14}$$

Note that this implies that a perturbation which moves m_i out of $[-M, M]$ (for any i) will always be rejected because $\tilde{\pi}_{\text{aug}}$ will be zero.

To implement the Noisy ML θ approach (with the proposal distribution defined in (3.36)), we first perturb $\vec{C}^{(t-1)}$ to obtain $\vec{\Gamma}^{(t)}$. Then we obtain the ML estimate of χ (given $\vec{\Gamma}^{(t)}$) using the result in (4.12) and add zero-mean Gaussian noise $\boldsymbol{u}_{\chi^{(t)}}$ with covariance $\sigma_u^2 \boldsymbol{I}$. Compared with the deterministic ML θ version, the same candidate curves are generated, but the probability of their being accepted or rejected is changed due to the randomly-perturbed values of $\chi^{(t)}$. The Hastings ratio for this approach is:

$$\eta(\vec{\Gamma}^{(t)}, \chi^{(t)} \,|\, \vec{C}^{(t-1)}, \theta^{(t-1)}) = \frac{\tilde{\pi}_{\text{aug}}(\vec{\Gamma}^{(t)}, \chi^{(t)} \,|\, I)}{\tilde{\pi}_{\text{aug}}(\vec{C}^{(t-1)}, \theta^{(t-1)} \,|\, I)} \frac{\mathtt{q}_1(\vec{C}^{(t-1)} \,|\, \vec{\Gamma}^{(t)})}{\mathtt{q}_1(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})} \frac{\mathtt{q}_2(\theta^{(t)} \,|\, \vec{C}^{(t-1)})}{\mathtt{q}_2(\chi^{(t)} \,|\, \vec{\Gamma}^{(t)})} \quad . \tag{4.15}$$

Computation of $\mathtt{q}_1$ is the same as for our standard curve perturbation. Because $\mathtt{q}_2$ generates m_0 and m_1 by adding Gaussian noise to the ML estimates of those parameters, the probability of generating $\theta^{(t-1)}$ given $\vec{C}^{(t-1)}$ and $\chi^{(t)}$ given $\vec{\Gamma}^{(t)}$ is simply the probability of generating the respective Gaussian noise vectors $\boldsymbol{u}_{\theta^{(t-1)}}$ and $\boldsymbol{u}_{\chi^{(t)}}$. Therefore we can write the ratio of the $\mathtt{q}_2$ terms as

$$\frac{\mathtt{q}_2(\theta^{(t-1)} \,|\, \vec{C}^{(t-1)})}{\mathtt{q}_2(\chi^{(t)} \,|\, \vec{\Gamma}^{(t)})} = \exp\left(-\frac{1}{2\sigma_u^2} (\boldsymbol{u}_{\theta^{(t-1)}}^{\mathrm{T}} \boldsymbol{u}_{\theta^{(t-1)}} - \boldsymbol{u}_{\chi^{(t)}}^{\mathrm{T}} \boldsymbol{u}_{\chi^{(t)}}) \right) \quad . \tag{4.16}$$

For the Independent Perturbations version (using the proposal distribution in (3.37)), we again use our standard curve perturbation to generate $\vec{\Gamma}^{(t)}$. To create $\chi^{(t)}$, we take $\theta^{(t-1)}$ and perturb it with zero-mean Gaussian noise $\boldsymbol{v}_{\chi^{(t)}}$ with covariance $\sigma_v^2 \boldsymbol{I}$. The perturbation from $\theta^{(t)}$ to $\chi^{(t)}$ is unbiased, and the Gaussian distribution is symmetric around zero, so $\mathtt{q}_2(\chi^{(t)} \,|\, \theta^{(t-1)})$ is symmetric. This means we can simply write the Hastings ratio as

$$\eta(\vec{\Gamma}^{(t)}, \chi^{(t)} \,|\, \vec{C}^{(t-1)}, \theta^{(t-1)}) = \frac{\tilde{\pi}_{\text{aug}}(\vec{\Gamma}^{(t)}, \chi^{(t)} \,|\, I)}{\tilde{\pi}_{\text{aug}}(\vec{C}^{(t-1)}, \theta^{(t-1)} \,|\, I)} \frac{\mathtt{q}_1(\vec{C}^{(t-1)} \,|\, \vec{\Gamma}^{(t)})}{\mathtt{q}_1(\vec{\Gamma}^{(t)} \,|\, \vec{C}^{(t-1)})} \quad . \tag{4.17}$$

Because this approach is not tied to θ_{ML} at all, it may be more likely than the two maximization-based methods to fully explore the configuration space and avoid being stuck in local maxima. It could also be less computationally efficient because this method generates θ in an unbiased fashion, and the maximization-based methods automatically generate θ estimates in high-probability regions of π.

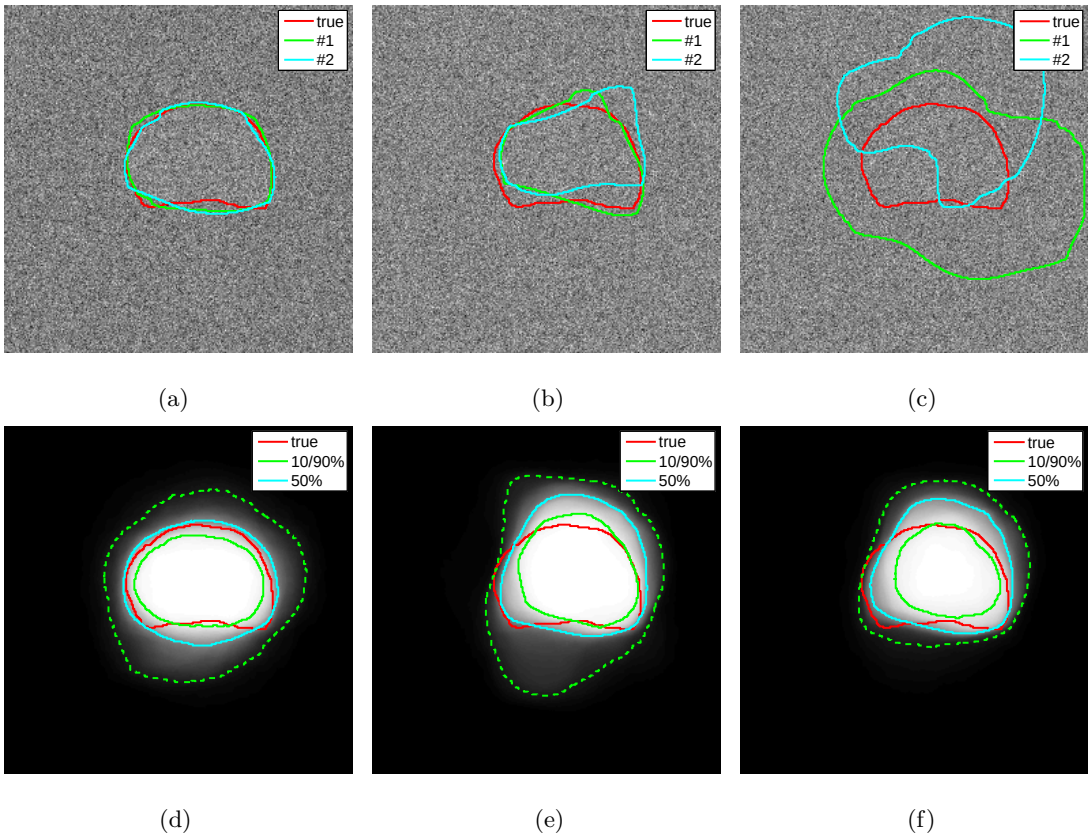


Figure 4.9. Most likely samples and marginal confidence intervals from three different curve sampling algorithms with parameter estimation. ML θ is shown in (a) & (d), Noisy ML θ with $\sigma_u = 0.05$ in (b) & (e), and Independent Perturbations with $\sigma_v = 0.01$ in (c) & (f).

■ 4.4.2 Results

In this section, we show the results from three different simulations involving ML θ, Noisy ML θ, and Independent Perturbations. For the Noisy ML θ version, we set the noise parameter $\sigma_u = 0.05$, and the Independent Perturbation version has noise parameter $\sigma_v = 0.01$. We use a higher noise parameter for the Noisy ML θ version because there is inherently less variance for this version (which takes the ML estimate of θ and adds noise) than for the Independent Perturbations version (where the parameter value θ is perturbed independently from $\vec{C}$). Even with $\sigma_v < \sigma_u$, the Independent Perturbations implementation displays much more variability than the Noisy ML θ one in the examples that follow.

The ML θ, Noisy ML θ, and Independent Perturbation methods generate sets of samples $\{\vec{C}_{\mathrm{ML},i}, \theta_{\mathrm{ML},i}\}_{i=1}^{N}$, $\{\vec{C}_{\mathrm{NML},i}, \theta_{\mathrm{NML},i}\}_{i=1}^{N}$, and $\{\vec{C}_{\mathrm{IP},i}, \theta_{\mathrm{IP},i}\}_{i=1}^{N}$, respectively (with $N = 1000$ for all three cases). We display the results in Fig. 4.9. The most probable $\vec{C}_{\mathrm{ML},i}$ samples in Fig. 4.9(a) are quite close to the true boundary, especially compared with the most likely samples from the case where the true mean values are known *a priori* (Fig. 4.6). With fixed m_0 and m_1 values, the most likely samples contained errors at the top of the shape simply due to the noise realization. In contrast, the ML θ approach can adapt the mean values to better match the observed data. The downside is that not specifying the mean values increases the variability in the overall sampling process relative to the fixed m_0/m_1 case. This can be seen in the confidence bounds and histogram image in Fig. 4.9(d) compared with previous results in Fig. 4.6(d).

The Noisy ML θ and Independent Perturbation approaches attempt to sample from the same target distribution using different proposal distributions. We can see in Fig. 4.9(e) and (f) that the 50% confidence boundaries in both are quite similar, but the 10% confidence boundaries diverge, and the most probable samples in 4.9(b) and (c) are extremely different. The reason the Independent Perturbation approach allows such outliers is two-fold: θ is allowed much freedom of movement[2], and the SNR of the observed image is very low. The latter is significant because it results in a high likelihood for poor values of θ to be accepted[3]. The erroneous θ values then lead to a

[2]While technically any value of θ has non-zero probability for the Noisy ML θ approach, in reality any deviation from $\hat{\theta}_{\mathrm{ML}}$ of more than $\sim 3\sigma_v$ is very unlikely.

[3]Consider that the expected squared error for a given pixel inside the curve (assuming a correct segmentation and correct m_1 estimate) is the noise variance of the image σ_w^2, and the expected squared error for an incorrect mean estimate $\tilde{m}_1$ is $\sigma_w^2 + (m_1 - \tilde{m}_1)^2$. Because $\sigma_w^2 = 100$, even relatively large

	(a)	(b)	(c)	(d)	(e)	(f)
Algorithm	$\bar{m}_0$	10/90%	$\bar{m}_1$	10/90%	$\bar{m}_{0,\mathrm{ML}}$	$\bar{m}_{1,\mathrm{ML}}$
Ground truth	N/A	N/A	N/A	N/A	-0.015	1.021
Known θ	N/A	N/A	N/A	N/A	-0.015	0.954
ML θ	0.030	0.020/0.041	0.787	0.594/0.975	0.030	0.787
Noisy ML θ	-0.022	-0.090/0.046	0.755	0.505/0.949	-0.022	0.754
Indep. Perturbations	0.048	-0.479/0.580	0.938	0.386/1.471	-0.014	0.846

Table 4.1. Average model parameter estimates with 10/90% confidence bounds for different curve sampling algorithms with parameter estimation. $\bar{m}_0$ and $\bar{m}_1$ are the sample average of the means as estimated by the sampling algorithms. $\bar{m}_{0,\mathrm{ML}}$ and $\bar{m}_{1,\mathrm{ML}}$ are the ML estimates of the means given the curve samples.

wider range of outcomes for the curve samples as well.

In Table 4.1 we display summary statistics for the sets of samples $\{\theta_{\mathrm{ML},i}\}$, $\{\theta_{\mathrm{NML},i}\}$, and $\{\theta_{\mathrm{IP},i}\}$. Columns (a) and (c) show the sample averages of m_0 and m_1 respectively for the different methods. We also display 10/90% confidence bounds of these estimates in columns (b) and (d). These bounds indicate the range over which the middle 80% of the samples occurred. The most accurate estimates overall come from the Independent Perturbations approach, though the confidence intervals are also much larger for this method than either ML θ or Noisy ML θ.

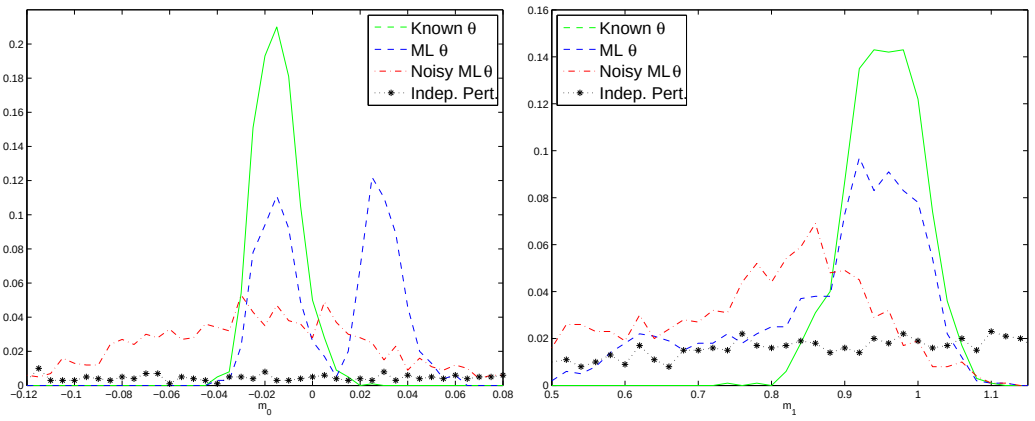
An alternative computation that we can perform is to look at each sampled curve $\vec{C}_i$ for a given sampling algorithm, and rather than using the corresponding sampled parameter value θ_i, we instead generate the most-likely estimate of the parameters given $\vec{C}_i$:

$$\hat{\theta}_i = \max_{\theta} \pi(\vec{C}_i \,|\, I;\, \theta) \ . \tag{4.18}$$

Note that these ML parameter estimates are no longer samples from the augmented target distribution for the Noisy ML θ and Independent Perturbation algorithms.

The main advantage of examining these ML estimates is that we can perform this computation even for sampling algorithms which do not attempt to estimate θ. For instance, for the Known θ version, we use a fixed $\theta = [0, 1]^{\mathrm{T}}$ in the actual sampling procedure to generate $\vec{C}_{\mathrm{known},i}$, but as a post-processing step, we can compute the ML

errors in $\tilde{m}_1$ (say $|m_1 - \tilde{m}_1| \approx 1$) are dwarfed by the noise, and this allows estimates of θ with large errors to be accepted with fairly high probability.



(a) (b)

Figure 4.10. Estimated marginal distributions of (a) m_0 and (b) m_1 for four different sampling algorithms: Known θ, ML θ, Noisy ML θ, and augmented state with Independent Perturbations.

estimate $\hat{\theta}_{\text{known},i}$ based on the sampled curve.

We then show in columns (e) and (f) the ML parameter estimates for the ground truth curve[4] as well as sample averages of the ML parameter estimates for the sets of sampled curves $\{\vec{C}_{\text{known},i}\}$, $\{\vec{C}_{\text{ML},i}\}$, $\{\vec{C}_{\text{NML},i}\}$, and $\{\vec{C}_{\text{IP},i}\}$. Note that for the ML θ case, columns (a) and (c) are identical to columns (e) and (f) because both sets of columns are computing the ML estimate of θ given the curve. Not surprisingly, the most accurate estimates of θ come from the Known θ case when the sampling procedure uses the correct model values θ^*. Again, the most accurate estimates out of the three online parameter estimation implementations come from the Independent Perturbation method.

In Fig. 4.10 we show histogram plots of the sampled values of θ for the ML θ, Noisy ML θ, and Independent Perturbations approaches[5]. We also display the histogram of the estimated ML $\{\hat{\theta}_{\text{known},i}\}$ values for the Known θ algorithm. While the sample

[4]For the ground truth segmentation $\vec{C}^*$ (*i.e.*, the curve used to generate I), we can compute the ML values of θ given that curve and the observed image I. These values differ slightly (as can be seen in Table 4.1) from the model values of $m_0 = 0$ and $m_1 = 1$ due to the specific realization of the image noise.

[5]Note that the range of values for the x-axis in Fig. 4.10(a) is much smaller for the x-axis in Fig. 4.10(b). This was chosen due to the generally smaller sample variability for most of the methods, but for the Independent Perturbation method, most of the sample values actually fall outside this range which is why the frequency values appear to be so low.

average for $\theta_{\text{IP},i}$ generated by the Independent Perturbation method is superior to the other parameter estimation methods, the variance is also much larger.

We note that the marginal distributions of θ differ quite significantly between the Noisy ML θ method and the Independent Perturbations method even though they should be drawn from the same target distribution. The reason for this is the mismatch between the augmented target distribution $\tilde{\pi}_{\text{aug}}$ in (4.14) (where we define θ to have a uniform distribution) and the proposal distribution for the Noisy ML θ approach in (3.36). The Markov chain which we are simulating for the Noisy ML θ case is irreducible (any value of θ has non-zero probability due to the infinite support of the Gaussian perturbation). That said, extremely large and small values of θ are unlikely to be generated by this method (as the perturbation standard deviation is $\sigma_u = 0.05$), so convergence of the sampling algorithm has likely not occurred yet for the results presented in this section and likely would not happen in a reasonable amount of time.

Of the three estimation methods tested on this example, the ML θ version produced the best overall results. The most-likely samples are good segmentations, and the 50% confidence bound is also quite accurate except at the bottom of the shape. In contrast, both methods which used the augmented variable space $\{\vec{C}, \theta\}$ have much more sample variability and median contours which grossly over segment the top of the shape. The prior distribution imposed on θ is very weak, and a more informative prior would significantly decrease the sample variability.

Note that the example in this section involved a relatively simple case with an image which was generated directly according to the data likelihood model. In more cluttered examples or ones in which there is model mismatch, the augmented variable versions may allow for faster convergence and more comprehensive exploration of the configuration space due to the additional uncertainty they allow in the model parameters. Additionally, for this example we were able to compute the ML θ values in closed form. This is not always feasible, and an augmented variable space algorithm is likely preferable in those situations.

■ 4.5 Convergence

MCMC methods provide an asymptotic guarantee of convergence if detailed balance and ergodicity are satisfied, but often there is not a simple method to determine whether convergence has occurred. A number of heuristic methods have been developed to help

assess the convergence status of MCMC sampling methods [12,19,31], but these methods tend to be model-specific or infeasible to apply for our high-dimensional curve sampling problem. For instance, consider a Markov chain with transition probability $\mathrm{T}(x \,|\, y)$ and stationary distribution $\pi(x)$. When simulating this chain, the distribution of the state at time t is $\pi^{(t)}(x) = \int \pi^{(t-1)}(y)\mathrm{T}(x \,|\, y)\mathrm{d}y$. There is a general result which states that the Kullback-Leibler (KL) divergence between $\pi^{(t)}$ and π decreases monotonically with t [31, 76] for an irreducible Markov chain. KL divergence is an information-theoretic definition of disparity between two probability distributions [17]:

$$\mathrm{D}(\pi \,||\, \pi^{(t)}) = \int \pi(x) \log \frac{\pi(x)}{\pi^{(t)}(x)} \mathrm{d}x \ . \tag{4.19}$$

Unfortunately this fact is difficult to use to create a stopping criteria. In order to compute $\mathrm{D}(\pi \,||\, \pi^{(t)})$, we typically must construct an approximate distribution $\hat{\pi}^{(t)}$ from a collection of samples from $\pi^{(t)}$. Computing the KL divergence between this sample-based approximation and π can be quite difficult, especially for high-dimensional problems. An additional consideration is that even if the KL divergence can be computed, it is a noisy measure because we can only approximate $\hat{\pi}^{(t)}$ with a finite number of samples. This means that the KL divergence between $\hat{\pi}^{(t)}$ and π is not necessarily monotonically decreasing.

In this section we provide some experimental results to assess two forms of convergence. The first is convergence of $\pi^{(t)}$ to π (known as mixing of the chain). This helps to determine how many iterations of the MCMC sampler should be run before one can be reasonably confident that the pool of samples adequately represents π. The second is convergence of $\hat{\pi}^{(t)}$ to $\pi^{(t)}$ (or $\hat{\pi}$ to π). This relates to deciding how many samples from the target distribution are needed to adequately represent the distribution.

■ 4.5.1 Mixing Rate

Rather than attempting to use the full probability distribution over curves to evaluate convergence, we can instead use marginal statistics, much as we did when visualizing samples and computing confidence bounds. We run our standard MCMC sampling algorithm for 1000 samples on the synthetic Gaussian noise example which was used previously in Sec. 4.3.1 (Fig. 4.6) and Sec. 4.4 (Fig. 4.9). We display the observed image along with the true boundary location and initial curve in Fig. 4.11. At a given time t in the iterative process, we can generate a time-dependent histogram image $\Phi^{(t)}$ from the set of current iterate values $\{\vec{C}_i^{(t)}\}_{i=1}^{1000}$.

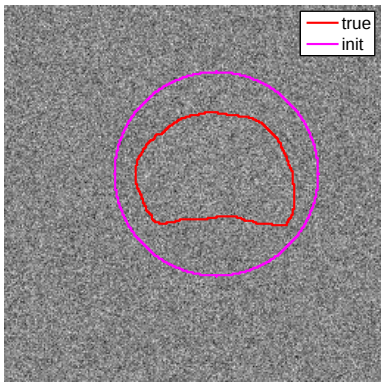


Figure 4.11. Observed noisy image with true contour location in red and initial curve $\vec{C}_i^{(0)}$ for all $i \in \{1, 2, \ldots, 1000\}$.

In Fig. 4.12 we show a time progression of $\Phi^{(t)}$ for times ranging from $t = 100$ to $t = 40,000$. As shown in Fig. 4.11, each initial iterate $\vec{C}_i^{(0)}$ is a circle. For $t = 100$ in the upper-left corner of Fig. 4.12, most iterates have not changed significantly from the initialization, and the 10/90% confidence bounds form a tight band around the median contour. As we go from $t = 100$ to $t = 5000$, we can see the median contour steadily moves inward (toward the true curve location in red) and the spread between the 10% and 90% confidence bounds also increases. We expect the latter behavior as more randomness is added to the process the more iterations we perform. Finally, we can see that the marginal confidence bounds and the histogram image do not change significantly for $t > 5000$ indicating that the samples appear to have converged to a final result.

We can also use the histogram images to assess convergence in a more quantitative fashion. Let Φ be the final histogram image (which we approximate with $\Phi^{(40,000)}$). We then define an error value $\mathcal{E}(\Phi^{(t)}, \Phi)$ between the converged histogram image and $\Phi^{(t)}$. We choose to define the error as the L1 distance between the two histogram images:

$$\mathcal{E}(\Phi^{(t)}, \Phi) = \int |\Phi^{(t)}(\boldsymbol{x}) - \Phi(\boldsymbol{x})| \mathrm{d}\boldsymbol{x} \ . \tag{4.20}$$

The L1 metric is often a reasonable choice because it is moderate in its behavior: it does not penalize large errors as heavily as the L2 metric and does not penalize small errors as heavily as the L0 metric.

Fig. 4.13 shows a log-scale plot of $\mathcal{E}(\Phi^{(t)}, \Phi)$ as a function of time. We can see that

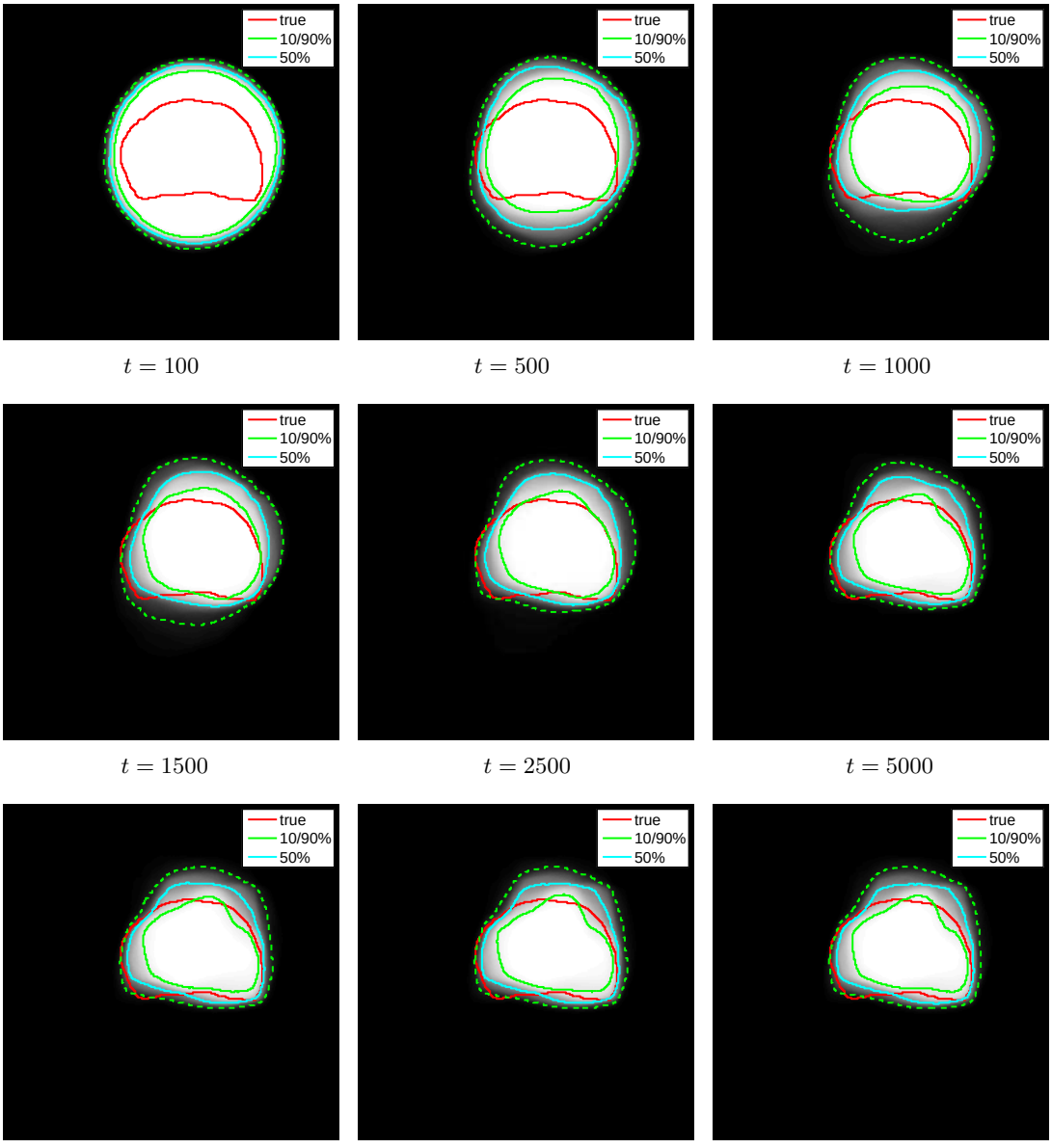


Figure 4.12. Progression of histogram images with respect to time for the synthetic L2 example. We begin with $t = 100$ in the upper-left corner and continue on to $t = 40,000$ in the lower-right corner.

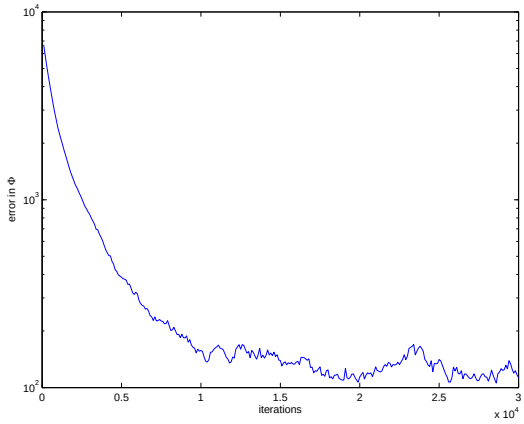
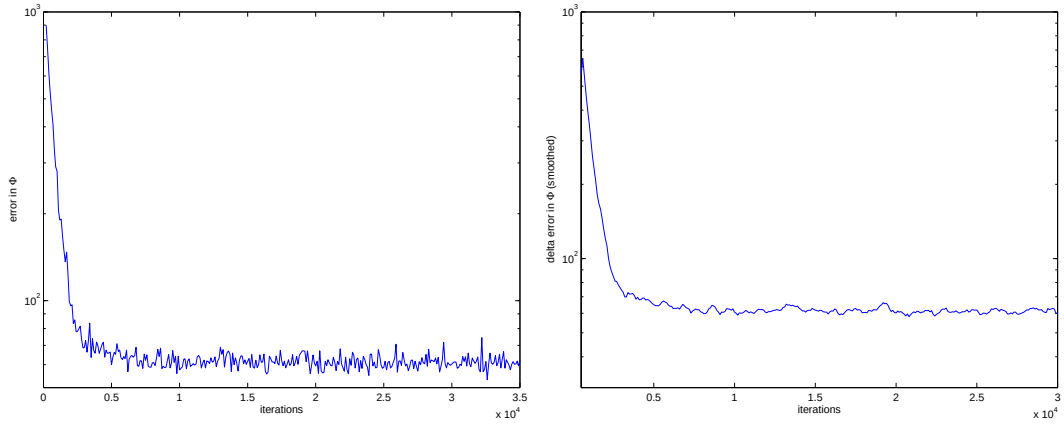


Figure 4.13. Log plot of the error in the histogram image as a function of time. The error is defined as the L1 error between the histogram image at a given time and the histogram image for $t = 40,000$.

the error steadily decreases until $t = 10,000$. From that point, $\mathcal{E}$ fluctuates in what appears to be a random fashion. We attribute this behavior to the fact that we only use a finite number of samples to construct $\Phi^{(t)}$, so the random fluctuations associated with the samples are not completely averaged out. From the results in this section, we conclude then that our curve sampling algorithm does converge for this synthetic example, and anecdotal experience with the other experiments in this chapter indicates that it also converges for a wide variety of segmentation problems.

While the previous analysis can be used to evaluate convergence retrospectively, it does not help to determine when to halt the sampling procedure (because $\Phi^{(40,000)}$ is unknown until we actually run the sampler for 40,000 iterations). Instead of comparing $\Phi^{(t)}$ with the converged histogram image, we can instead compute a delta error between $\Phi^{(t)}$ and a recent previous histogram image (*e.g.*, $\Phi^{(t-100)}$) and see how large the change is. If we had an infinite number of samples, the delta error $\mathcal{E}(\Phi^{(t-100)}, \Phi^{(t)})$ would go to zero as $t \to \infty$ because once the samples have converged to the stationary distribution of the chain, simulating the chain leaves the distribution of the samples unchanged. With a finite number of samples, we would expect $\mathcal{E}(\Phi^{(t-100)}, \Phi^{(t)})$ to decrease until it reached a plateau around which it would fluctuate due to the innate variability of the samples.

We show a plot of $\mathcal{E}(\Phi^{(t-100)}, \Phi^{(t)})$ as a function of t in Fig. 4.14(a). We can see that this quantity rapidly decreases until approximately $t = 5000$. For $t > 5000$,



(a) (b)

Figure 4.14. (a) Log plot of the delta change in the histogram image as a function of time. The error is defined as the L1 distance between the histogram image at time t and at time $t + 100$. (b) Time-averaged version of (a) where the error values are averaged over 5 intervals.

$\mathcal{E}(\Phi^{(t-100)}, \Phi^{(t)})$ fluctuates around a small value. In Fig. 4.14(b), we show the average of the previous 5 delta errors in order to smooth the error plot and make inflection points in the curve more readily apparent. This means that at a time t, we plot the average of the errors $\mathcal{E}(\Phi^{(t-500)}, \Phi^{(t-400)})$ through $\mathcal{E}(\Phi^{(t-100)}, \Phi^{(t)})$. From this plot we can see that the rapid decrease in the error ends around $t = 3000$, and the downward trend in the error largely ends around $t = 9000$. Thus we can conclude that for this example, a reasonable point to end the iterative sampler when using the delta error as a stopping criterion is for values ranging from $t = 3000$ to $t = 9000$. This generally agrees with the convergence behavior we observed previously in Fig. 4.13 when we had the converged histogram image.

■ 4.5.2 Convergence in the Number of Samples

In this section, we address the problem of determining the number of samples N_s needed to represent the configuration space and produce adequate histogram images and confidence bounds. We use a similar analysis to the previous section, except here we vary N_s instead of t. We begin by generating a set of 1250 samples $\{\vec{C}_i\}_{i=1}^{1250}$ for the synthetic noisy Gaussian image used previously (using $t = 10,000$) and then examine the behavior of the histogram image as we use different subsets of $\{\vec{C}_i\}_{i=1}^{1250}$ to construct

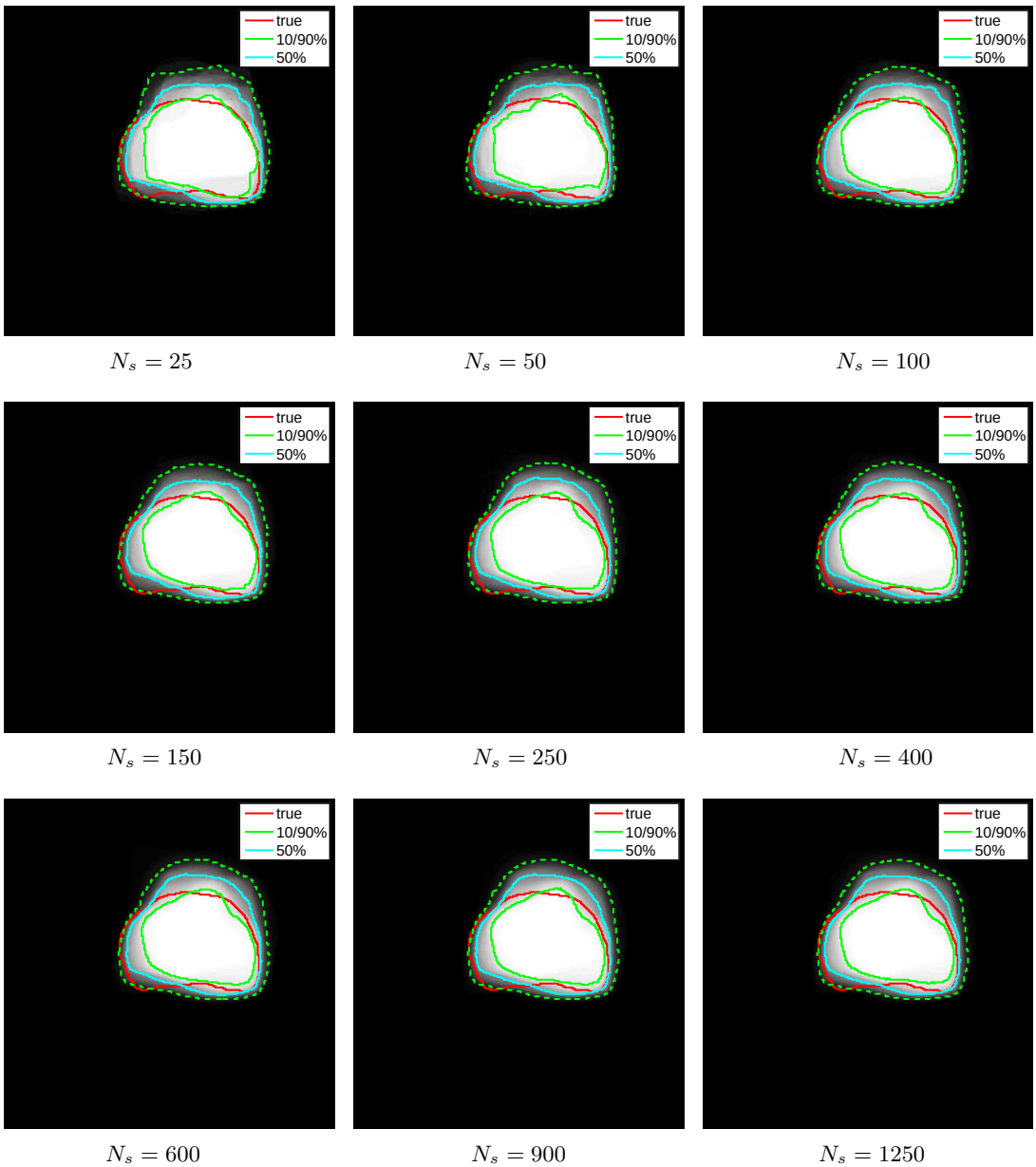
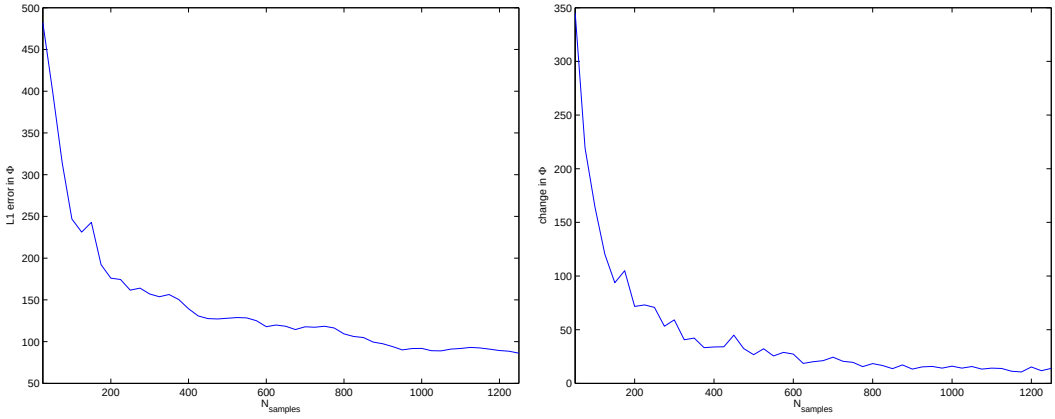


Figure 4.15. Progression of histogram images with respect to the number of samples N_s for the synthetic L2 example. We begin with $N_s = 25$ in the upper-left corner and continue on to $N_s = 1250$ in the lower-right corner.



(a) (b)

Figure 4.16. (a) Plot of the error in the histogram image and (b) delta change in the histogram image, both as a function of the number of samples. The error is defined as the L1 error between the histogram image for a given number of samples and the histogram image for $N_s = 5000$.

it.

Let $\Phi_{1:N_s}$ be the histogram image derived from all samples from $\vec{C}_1$ to $\vec{C}_{N_s}$. In Fig. 4.15 we display a progression of these histogram images from $N_s = 25$ to $N_s = 1250$. We can see that even with just 25 samples, the confidence bounds for $\Phi_{1:25}$ are similar to those from $\Phi_{1:1250}$ (*i.e.*, as we add samples, the confidence bounds become smoother but otherwise do not change significantly). From a qualitative perspective, $N_s = 250$ seems indistinguishable from $N_s = 1250$ and even $N_s = 100$ or 150 is probably adequate for most purposes[6].

We can also quantitatively compare the convergence errors using the L1 metric as before. To do so, we generate 5000 samples (which are independent of the 1250 samples previously generated) and form a histogram image Φ from them. Empirical observations indicate that the histogram image does not substantially change when adding more samples after this point, so we believe this histogram image has converged with respect

[6]One application of this result is for the interactive semi-automatic segmentation process we describe in Chap. 5. With this approach, an expert user provides a partial segmentation, and the algorithm generates samples conditioned on this information. Based on the samples, the expert can then update the partial segmentations to generate a new set of samples. In this situation, the quality of the intermediate results can be lower than for the final segmentation results. The analysis in this section gives a sense of how many samples we need to generate to provide an adequate intermediate result.

to N_s (*i.e.*, for N_s large, adding another sample $\vec{C}_{N_s+1}$ does not significantly change the histogram image by more than some $\epsilon > 0$). In Fig. 4.16(a), we plot $\mathcal{E}(\Phi_{1:N_s}, \Phi)$, the L1 error between $\Phi_{1:N_s}$ and Φ, as a function of N_s. The error decreases rapidly until $N_s = 300$ and slowly declines from there until $N_s = 950$. At that point the error appears to plateau.

A delta error calculation can also be performed. For a given N_s, we compare the current histogram image $\Phi_{1:N_s}$ with $\Phi_{1:(N_s-25)}$ (which omits the last 25 samples). We plot the resulting $\mathcal{E}(\Phi_{1:(N_s-25)}, \Phi_{1:N_s})$ in Fig. 4.16(b). There is a rapid decline in the delta error until $N_s = 400$ and a slower decline until approximately $N_s = 800$. For $N_s > 800$, the delta error does not significantly change. The results for this delta error calculation largely agree with the previous case in Fig. 4.16(a), and we can conclude that generating 400 to 800 samples for this example is sufficient to produce adequate histogram images and confidence bounds.

Chapter 5

Conditional Simulation

THE problem of semi-automatic segmentation is one which has drawn a fair amount of attention in recent years. With these methods, a user manually segments part of the curve, and an algorithm is used to complete the unknown portions. Falcao *et al.* [25] and Mortensen and Barrett [72] independently introduced similar methods now commonly referred to as Live Wire. These methods use Dijkstra's algorithm [23] (a form of dynamic programming [4]) to find the shortest paths between manually-segmented segments. Distance is typically defined to be inversely related to image gradients so shortest paths follow edges in the image. Perez *et al.* [85] developed a curve tracing algorithm based on particle filtering. Their algorithm samples curve segments in an image, and their framework allows the user to specify forbidden regions which the curve should not enter. When the particle filter produces multiple branches, this allows them to specify which branch should be taken. Protiere and Sapiro [87] introduced an interactive segmentation framework based on "scribbles." In this approach, a user draws contours which are rough, cartoon-like approximations to the true boundary. The algorithm automatically learns a set of statistics (based on Gabor filters) from these scribbles and uses this information to segment the remainder of the image. The user can then provide more scribbles to refine the estimate.

In this chapter, we show how to naturally incorporate knowledge of the exact location of a portion of the solution[1] into our sampling framework based on an implementation of something that is often referred to in the geophysics literature as *conditional simulation* [121]. With conditional simulation, one draws samples from a multi-dimensional probability distribution where a subset of the variables are conditioned on specified fixed values for the other variables. For curve sampling, this corresponds to

[1] We discuss in Chap. 7 possible approaches to incorporate uncertainty about the provided curve locations.

knowing the location of some portions of the curve and simulating the remainder of the curve.

With optimization-based approaches, using this kind of constraint would typically require the use of constrained-optimization techniques [3] which are quite difficult to implement for high-dimensional problems. One downside of the Live Wire approach for semi-automatic segmentation is that the information used to decide where to extrapolate the curve locations is local in nature (*e.g.*, edge data). In contrast, by sampling curves from conditional probability distributions, we generate curves based on global, region-based energy terms. These allow us to construct curve segments for the unknown portions of the curve that are geometrically consistent with the curve as a whole, provide greater robustness to noise, and enable the segmentation of objects whose boundaries are not characterized by edges in an image (*e.g.*, the gravity inversion example which appears in Sec. 5.3).

We begin with a description of how to extend our base curve sampling algorithm to perform geometric conditional simulation in Sec. 5.1. We demonstrate the usefulness of this approach for segmenting subcortical brain structures in Sec. 5.2 and reconstructing underground salt bodies from surface gravity data in Sec. 5.3.

■ 5.1 Incorporating Fixed Constraints

For simplicity, consider the situation where there is a single known part of the curve $\vec{C}_{\mathrm{k}} : [0, b] \to \Omega$ ($b \in [0, 1]$) and an unknown part of the curve $\vec{C}_{\mathrm{u}} : [b, 1] \to \Omega$. Assuming an arc length parameterization with respect to the full curve, the two pieces combine to form the overall curve as:

$$\vec{C}_{\mathrm{a}}(p) = \begin{cases} \vec{C}_{\mathrm{k}}(p) & : \quad p \in [0, b] \\ \vec{C}_{\mathrm{u}}(p) & : \quad p \in [b, 1] \end{cases} \tag{5.1}$$

For $\vec{C}_{\mathrm{a}}$ to be closed, we need $\vec{C}_{\mathrm{k}}(b) = \vec{C}_{\mathrm{u}}(b)$, $\vec{C}_{\mathrm{k}}(0) = \vec{C}_{\mathrm{u}}(1)$, and $\vec{C}_{\mathrm{k}}$ and $\vec{C}_{\mathrm{u}}$ to be continuous. While we only discuss the case where $\vec{C}_{\mathrm{k}}$ is a single contiguous curve segment on $[0, b]$, it is straightforward to generalize this approach for multiple known intervals $\vec{C}_{\mathrm{k},i} : [a_i, b_i] \to \Omega$.

Now we wish to sample from a modified target distribution over $\vec{C}_{\mathrm{u}}$ conditioned on the image and the known part of the curve:

$$\tilde{\pi}(\vec{C}_{\mathrm{u}} \,|\, I, \vec{C}_{\mathrm{k}}) \propto \mathrm{p}(I \,|\, \vec{C}_{\mathrm{u}}, \vec{C}_{\mathrm{k}}) \mathrm{p}(\vec{C}_{\mathrm{u}} \,|\, \vec{C}_{\mathrm{k}}) = \mathrm{p}(I \,|\, \vec{C}) \mathrm{p}(\vec{C}_{\mathrm{u}} \,|\, \vec{C}_{\mathrm{k}}) \ . \tag{5.2}$$

We can see that the data likelihood term is unchanged, but the prior term on $\vec{C}$ is now a conditional prior on $\vec{C}_{\mathrm{u}}$ given $\vec{C}_{\mathrm{k}}$. Note that we can alternatively write the prior as

$$\mathrm{p}(\vec{C}_{\mathrm{u}} \,|\, \vec{C}_{\mathrm{k}}) = \frac{\mathrm{p}(\vec{C}_{\mathrm{u}}, \vec{C}_{\mathrm{k}})}{\mathrm{p}(\vec{C}_{\mathrm{k}})} = \frac{\mathrm{p}(\vec{C})}{\mathrm{p}(\vec{C}_{\mathrm{k}})} \ . \tag{5.3}$$

Because $\vec{C}_{\mathrm{k}}$ is constant, $\mathrm{p}(\vec{C}_{\mathrm{k}})$ is also constant, so we can write the target distribution as:

$$\tilde{\pi}(\vec{C}_{\mathrm{u}} \,|\, I, \vec{C}_{\mathrm{k}}) \propto \mathrm{p}(I \,|\, \vec{C})\mathrm{p}(\vec{C}) \tag{5.4}$$

with $\vec{C}$ and $\vec{C}_{\mathrm{u}}$ related through (5.1). We can then see that computing the conditional target distribution is the same computation as in our regular sampling algorithm; the only major changes must occur with the proposal distribution.

The space of closed curves forms a manifold in an embedding Hilbert space [53, 60]. The space of closed curves which contain $\vec{C}_{\mathrm{k}}$ exist as a submanifold in that manifold. Previously we ensured that our perturbations would not deviate from the manifold of closed curves by having perturbations which satisfied $f(0) = f(1)$. Now we must further constrain our proposal distribution to ensure that we do not allow the known part of $\vec{C}$ to change. To do so, we need to modify our proposal distribution so that the perturbation $f(p) = 0$ for $p \in [0, b]$. One way to implement this is to multiply the random perturbations we defined in Sec. 3.2 by a scalar field $d : [0, 1] \to \mathbb{R}$ (with $d(p) = 0$ for $p \in [0, b]$):

$$f(p) = d(p) \left(\mu_{\vec{C}_{\mathrm{a}}}(p) + h(p) \circledast n(p) \right) \ . \tag{5.5}$$

To make $\vec{\Gamma}$ smooth around the end points of $\vec{C}_{\mathrm{k}}$ (*i.e.*, around $\vec{\Gamma}(0)$ and $\vec{\Gamma}(b)$), d should smoothly transition from 0 to 1 over a finite interval ϵ. This can be done, *e.g.*, in a piecewise-linear fashion as:

$$d(p) = \begin{cases} 0 & : \quad p \in [0, b] \\ \frac{p-b}{\epsilon} & : \quad p \in [b, b+\epsilon] \\ 1 & : \quad p \in [b+\epsilon, 1-\epsilon] \\ \frac{1-p}{\epsilon} & : \quad p \in [1-\epsilon, 1] \end{cases} \tag{5.6}$$

Computationally, this is equivalent to multiplying our random vector $\boldsymbol{f}$ by a diagonal matrix $\boldsymbol{D}$ (with the entries of $\boldsymbol{D}$ corresponding to discretized values of $d(p)$): $\boldsymbol{f} = \boldsymbol{D}(\boldsymbol{\mu} + \boldsymbol{H}\boldsymbol{n})$. This results in $\boldsymbol{f}$ being drawn from a Gaussian distribution $\mathrm{N}(\boldsymbol{D}\boldsymbol{\mu}, \boldsymbol{D}\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D})$. This is a degenerate probability distribution as some entries of $\boldsymbol{f}$ have zero variance, so

we should only evaluate the proposal distribution using the the perturbation $\boldsymbol{f}_{\mathrm{u}}$ on the unknown part of the curve. Let $\boldsymbol{D}_{\mathrm{u}}$ be a rectangular $M \times N$ block-diagonal matrix which is $\boldsymbol{D}$ with all rows containing only zeros removed. Then $\boldsymbol{f}_{\mathrm{u}} = \boldsymbol{D}_{\mathrm{u}}(\boldsymbol{\mu}+\boldsymbol{H}\boldsymbol{n}) = \boldsymbol{D}_{\mathrm{u}}\boldsymbol{\mu}+\boldsymbol{r}_{\mathrm{u}}$ where $\boldsymbol{r}_{\mathrm{u}} = \boldsymbol{D}_{\mathrm{u}}\boldsymbol{H}\boldsymbol{n}$. Putting it all together, we can evaluate the proposal distribution as

$$\mathfrak{q}(\vec{\Gamma} \,|\, \vec{C}) = \mathfrak{q}(\vec{\Gamma}_{\mathrm{u}} \,|\, \vec{C}_{\mathrm{u}}) \propto \exp\left(-\frac{1}{2\sigma^2}\boldsymbol{r}_{\mathrm{u}}^{\mathrm{T}}(\boldsymbol{D}_{\mathrm{u}}\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}_{\mathrm{u}}^{\mathrm{T}})^{-1}\boldsymbol{r}_{\mathrm{u}}\right) \tag{5.7}$$

Computing the reverse perturbation probability follows a similar procedure.

Note that this conditional simulation approach is more subject to numerical problems than our basic curve sampling framework. Two main issues arise, both of which involve self-intersections of the curve. The first is that the portion of $\vec{C}_{\mathrm{u}}$ immediately adjacent to $\vec{C}_{\mathrm{k}}$ can move (due to the random perturbations) into a configuration which can cause a sharp change in the curve at the juncture between the known and unknown portions of the curve[2]. It is possible once the curve has a sharp bend that a perturbation will cause the unknown part of the curve to intersect $\vec{C}_{\mathrm{k}}$. The second issue is that a portion of $\vec{C}_{\mathrm{u}}$ which is far from $\vec{C}_{\mathrm{k}}$ (in terms of distance measured along the curve) intersects $\vec{C}_{\mathrm{k}}$. In standard curve evolution, this simply causes a topology change, but this is problematic for our sampling method, both because topology changes are not allowed, and because it causes a change in $\vec{C}_{\mathrm{k}}$ (which should be fixed).

When such a numerical error occurs, we restart the sampling process from the initial curve $\vec{C}^{(0)}$. This does introduce a bias in the sampling process, but these types of errors generally occur infrequently in our numerical experiments because the mean perturbation in our proposal distribution makes it difficult to generate regions of high negative curvature (which are needed to create these pinched-off self-intersection points). If an experiment results in many of these numerical errors, it may be necessary to modify the proposal distribution to explicitly attempt to prevent these occurrences. One approach that may help with the first issue is to use a mean perturbation that enforces smoothness at the junctions between the known and unknown segments of the curve. In Sec. 5.3.4 (page 133), we formulate a different approach which aids with both types of numerical issues. This method uses Sundaramoorthi and Yezzi's more-than-topology-preserving curve flow (see Sec. 2.2.1) to cause the mean perturbation to explicitly disfavor curve movements that lead to topological changes.

[2]While we ensured in (5.5) and (5.6) that f was small in the regions adjacent to $\vec{C}_{\mathrm{k}}$, this does not mean that larger movements cannot occur over a number of iterations.

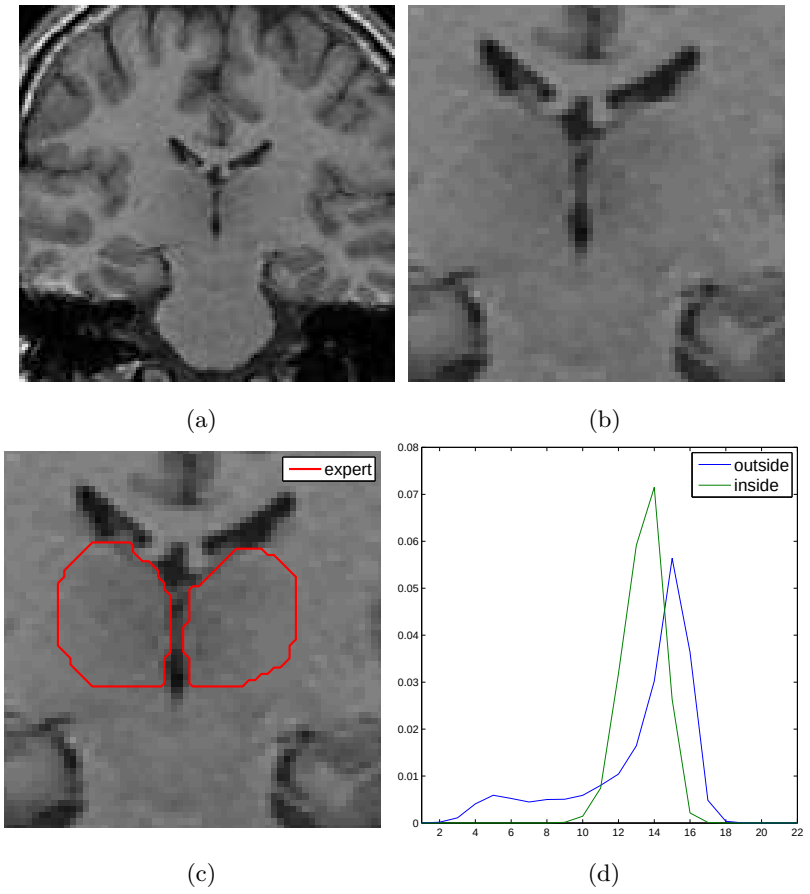


Figure 5.1. Axial thalamus MR image. (a) Original 128×128 image. (b) Magnified portion of the same image centered on the thalamus. (c) Overlay of radiologist segmentation. (d) Histograms (using only points within 10 pixels of the thalamus boundaries) inside and outside the curve estimated from training data.

■ 5.2 Thalamus Segmentation

Segmenting sub-cortical structures in brain MR images is a challenging task due to the low contrast between tissue types and the small size of the structures involved. An example of an axial slice is shown in Fig. 5.1(a). One approach by Pohl *et al.* [86] to make the segmentation problem better posed is to use strong prior shape models to constrain the range of likely curves. The approach we take here is to use our conditional simulation approach to incorporate a small amount of user input to dramatically reduce the sample variance.

We use a likelihood model similar to the one used to segment the prostate in Sec. 4.3.2 where we used training data to learn non-parametric intensity distributions inside and outside the curve and assumed the intensity values are iid given the curve. The main difference here is that, due to the low contrast, the large amount of clutter in brain MR images basically eliminates the discriminatory power of global non-parametric densities to do thalamus segmentation. To counteract these effects, we apply the idea of Yezzi *et al.* [126] to use a local banded intensity model: only those pixels located within the band of pixels less than a distance d_0 to the curve are used for both learning of the intensity models and evaluation of the data likelihood. The distance between a point $\boldsymbol{x}$ and a curve $\vec{C}$ is defined using the Hausdorff distance [73]:

$$\mathrm{d_H}(\boldsymbol{x}, \vec{C}) = \min_p \|\boldsymbol{x} - \vec{C}(p)\| \ . \tag{5.8}$$

This is a convenient choice when using level sets. If $\tilde{\Psi}_{\vec{C}}$ is the signed-distance function whose zeroth level set is $\vec{C}$, the Hausdorff distance at a point $\boldsymbol{x}$ is simply $|\tilde{\Psi}_{\vec{C}}(\boldsymbol{x})|$. The signed distance function can be efficiently computed using fast marching methods [96].

The thalamus consists of two disjoint regions, so we must modify our curve representation. As mentioned in Sec. 3.3, we cannot allow topological change when perturbing $\vec{C}$ into $\vec{\Gamma}$. Han *et al.* [43] and Segonne *et al.* [95] have developed topology-preserving active contour methods which use a single level set function to represent a known number of disjoint regions. Unfortunately, these approaches would be difficult to adapt for our method because they enforce topology preservation by modifying the force as applied to the level set, not the force directly on the curve itself. Computing the probability of this modified perturbation would then be quite difficult as the change to the force on the level set would need to be related to a change to the force on the curve, and this computation is needed to evaluate the Hastings ratio and ensure detailed balance.

The approach we take instead is simply to represent each separate curve with its own level set function. This is similar to the approach Yezzi *et al.* [126] use to segment multiple regions. Let $\boldsymbol{C} = \{\vec{C}_1, \vec{C}_2\}$ be composed of two curves $\vec{C}_1$ and $\vec{C}_2$, each of which has a corresponding level set function $\Psi_{\vec{C}_1}$ and $\Psi_{\vec{C}_2}$. These two curves combine together to form a label map $\lambda_{\boldsymbol{C}} : \Omega \rightarrow \{0, 1\}$ which specifies whether a given pixel is inside or outside of the thalamus (we do not differentiate between the left and right halves of the thalamus). We define $\lambda_{\boldsymbol{C}}(\boldsymbol{x})$ to be 1 when $\boldsymbol{x}$ is inside either $\vec{C}_1$ or $\vec{C}_2$ or 0 otherwise:

$$\lambda_{\boldsymbol{C}}(\boldsymbol{x}) = \mathcal{H}(-\Psi_{\vec{C}_1}(\boldsymbol{x})) + \mathcal{H}(-\Psi_{\vec{C}_2}(\boldsymbol{x})) - \mathcal{H}(-\Psi_{\vec{C}_1}(\boldsymbol{x}))\mathcal{H}(-\Psi_{\vec{C}_2}(\boldsymbol{x})) \ . \tag{5.9}$$

We do not directly penalize $\vec{C}_1$ and $\vec{C}_2$ for overlapping because they represent the same object, though a curve length penalty does naturally discourage overlap. Any overlap between the curves increases the curve length without affecting the actual segmentation.

We also need to modify our proposal distribution to account for the fact that we now have two curves. We do this by defining our overall proposal distribution to consist of independent perturbations to the individual curves:

$$\mathtt{q}(\boldsymbol{\Gamma} \,|\, \boldsymbol{C}) = \prod_i \mathtt{q}_i(\vec{\Gamma}_i \,|\, \vec{C}_i) \tag{5.10}$$

where $\boldsymbol{\Gamma} = \{\vec{\Gamma}_1, \vec{\Gamma}_2\}$. While the curves are perturbed independently, one overall Hastings ratio $\eta(\boldsymbol{\Gamma} \,|\, \boldsymbol{C})$ is computed, and the curves are jointly accepted or rejected.

The banded-likelihood approach leads to the following data likelihood term:

$$\mathrm{p}(I \,|\, \boldsymbol{C}) \;=\; \prod_{\substack{\{\boldsymbol{x} \,|\, \exists\, i \text{ s.t.} \\ |\tilde{\Psi}_{\vec{C}_i}(\boldsymbol{x})| \,\leq d_0\}}} \mathrm{p}\left(I(\boldsymbol{x}) \,|\, \lambda_{\boldsymbol{C}}(\boldsymbol{x})\right) \quad . \tag{5.11}$$

The overall target distribution incorporates a standard curve length prior on each of the curves in addition to the likelihood:

$$\pi(\boldsymbol{C}) \propto \mathrm{p}(I \,|\, \boldsymbol{C}) \exp\left(-\alpha \sum_i \oint_{\vec{C}_i} \mathrm{d}s\right) \quad . \tag{5.12}$$

In Fig. 5.1(b)-(c), we show a magnified section of the brain MR image along with an expert segmentation of the thalamus. The dark areas between and above the thalamus are known as the ventricles and provide a fairly strong intensity contrast to the thalamus; the intensity differences between the thalamus and the surrounding cerebral tissue is more subtle. The banded histogram distribution learned from 5 expert-segmented volumes (with $d_0 = 10$ pixels) is shown in Fig. 5.1(d). We can see that the thalamus is, on average, slightly darker than the surrounding tissue.

While we did attempt to segment this example using the regular unconstrained curve sampling algorithm presented in Chap. 3, there is simply too much similarity between the intensity distributions inside and outside the curve to segment the thalamus using intensity alone. The likelihood term in (5.11) appears in the Hastings ratio as the likelihood of the candidate sample $\boldsymbol{\Gamma}$ divided by the likelihood of the previous iterate $\boldsymbol{C}$, and this largely becomes a ratio involving the pixels where $\lambda_{\boldsymbol{C}}$ and $\lambda_{\boldsymbol{\Gamma}}$ disagree on the proper label. Because $\mathrm{p}(I(\boldsymbol{x}) \,|\, 0)$ and $\mathrm{p}(I(\boldsymbol{x}) \,|\, 1)$ are so similar, the Hastings ratio

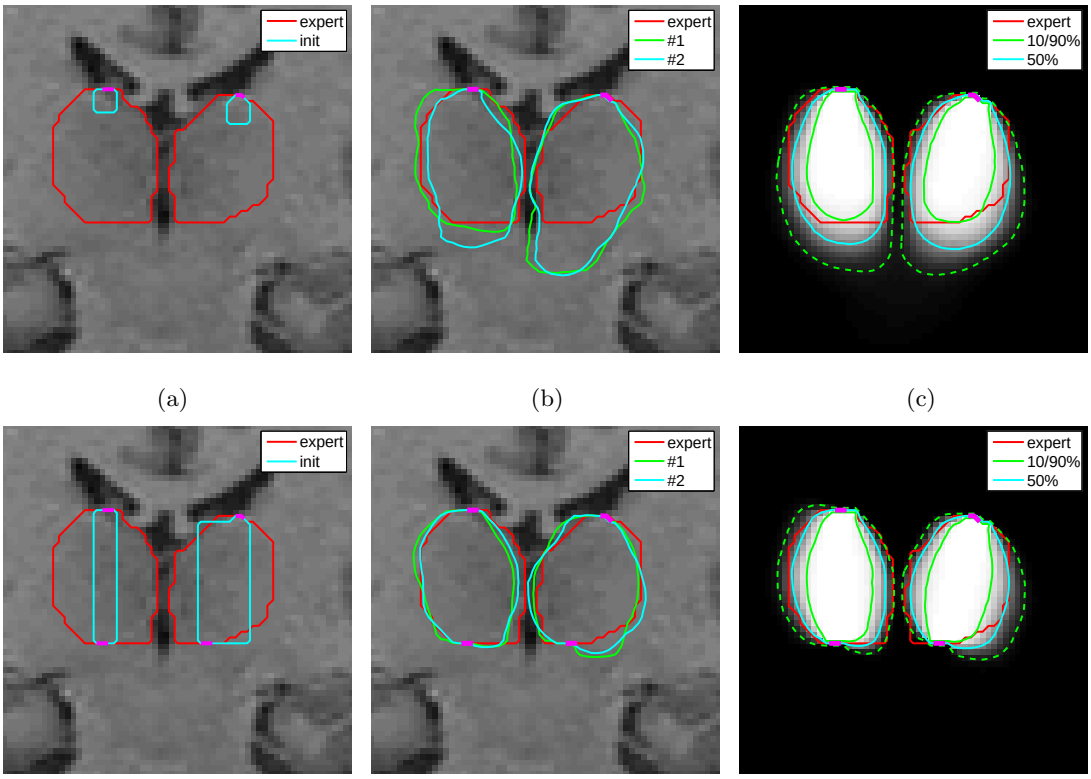


Figure 5.2. Conditionally-simulated thalamus segmentation using non-parametric intensity distributions. The top row ((a)-(c)) has a point fixed at the top for each half of the thalamus, and the bottom row ((d)-(f)) has those same points fixed along with an additional point at the bottom of each half. The initializations for each version are shown in (a) & (d), the two most probable samples in (b) & (e), and the histogram images and confidence intervals in (c) & (f). The fixed portions of the curve $\vec{C}_k$ are shown in magenta.

is likely to be close to 1, so many candidate samples will be accepted even when they introduce a number of segmentation errors.

To introduce extra constraints on the sampling problem, we apply our conditional simulation approach by specifying small portions of the curve *a priori*. Computation per sample was approximately 20 seconds for 5000 iterations. While we have two curve interfaces to track, the curves are also quite small which reduces the computational burden. In Fig. 5.2(a)-(c) we show an example where two small sections of the curve are given to the algorithm—one at the top of each half of the thalamus—and the remainder of the curve is generated conditioned on that information. The known portion of the curve is drawn in magenta in all of the figures. Overall the segmentation is fairly accurate, especially on the top and the sides. The 10/90% marginal confidence bounds also completely bracket the expert-generated boundaries of the thalamus.

Note that the sampling method provides information about where the greatest uncertainty is and thus where additional expert assistance can improve the results the most. Not surprisingly, we can see in Fig. 5.2(c) that the bottom (the part which is farthest from the information that was already supplied by the expert) possesses the most sample variability as evidenced by the more diffuse histogram image and larger gap between the confidence bounds than the other parts of the thalamus.

We can then add new fixed locations at the bottom of the thalamus and generate a new set of samples conditioned on the constraints on both the top and bottom. The results can be seen in Fig. 5.2(d)-(f). The two most likely samples now adhere much more closely to the expert-segmented boundaries as does the median contour, and the overall sample variability has also been reduced. The expert contour is completely enclosed within the 10/90% confidence bounds except for a small portion of the right half of the thalamus where our model is perhaps forcing the samples to be too smooth.

Overall we can see that our conditional simulation framework allows us to produce very reasonable results from what otherwise would be an underconstrained problem simply by introducing a small amount of user-generated information. Furthermore, the samples provide information to help us decide where to target additional expert guidance, and we can use this to iteratively refine the segmentation to correct errors and reduce the sample variance. The iterative process can continue until the user is satisfied with the result.

■ 5.3 Gravity Inversion

In petroleum geophysical applications, it is often important to be able to accurately locate underground salt bodies. Fluid does not flow through salt bodies, so water, oil, and gas tend to become trapped next to them. The top of salt bodies are usually easy to locate using standard seismic imaging techniques [2, 98], but locating the bottom is more difficult due to the strong dispersion and bending of seismic waves passing through the salt/sediment interface; without accurate knowledge of the lower salt boundaries, high-quality seismic imaging reconstruction below the salt body is impossible.

Gravity inversion is one technique that can be used to localize salt bodies [7]. Salt is less dense than the surrounding rock, so the presence of salt will cause a deviation from the expected gravitational field at the earth's surface which can be measured by very precise devices called *gravimeters*. Using these surface gravity measurements, gravity inversion techniques then try to reconstruct the underlying geology.

Gravity measurements have been used since the 1930s for petroleum exploration applications, and, in fact, were the primary exploration tool used until supplanted by seismic imaging [75]. Traditional gravity processing involved manual interpretation by geophysicists. Later, the advent of computers allowed a more interactive analysis in which an expert user could provide an estimate of the underground structure, evaluate the resulting gravity profile using a forward model, and use the resulting error to update the estimate [49, 91]. Another approach is to view the structure estimation problem as an inverse problem. Various methods have been devised such as classical formulations using Fourier transforms [79] and more recent approaches such as using simulated annealing and genetic optimization to solve a combinatorial optimization problem [61]. See Nabighian *et al.* [75] for a more comprehensive survey.

In an inverse problem, the output of some function is known, and the parameters of the function are estimated based on the output. One approach then is to formulate an energy functional which penalizes the error between the observations and the output of the forward model for a given parameter configuration. With an optimization-based approach, one would typically calculate derivatives of that energy functional and attempt to minimize it. For our sampling-based method, we simply need to be able to *evaluate* the forward model. This is advantageous because complex forward models can be easily integrated into this framework[3].

[3]Non-gradient-based optimization methods can also be used in a similar fashion, but these tend to

In this section we begin by describing a formulation of the gravity inversion problem using geometric curves. We show how to find local minima of it using a gradient-based curve evolution approach and how to draw samples with our curve sampling framework. We demonstrate results first on a single-region example with simple geometry and then on more complex two-body salt examples using conditional simulation. Note that conditional simulation is a particularly powerful tool for this problem because the salt bodies do not have consistent shape properties. Thus shape models cannot be easily used to help constrain the solution as in many medical imaging applications.

The results we present in this section are based on synthetic models with observations generated exactly according to our forward model in (5.13)[4]. This enables us to isolate the behavior of our segmentation model from more complex behavior in the forward model and to derive gradient flows as in (5.23).

■ 5.3.1 Curve-based Gravity Model

We discuss a 2D version of the gravity inversion problem formulation here, but the approach naturally generalizes to 3D as well. We assume there is an array of N_g evenly-spaced gravimeters on the surface located at $\boldsymbol{x}_i = (i\Delta x, 0)$ (for $i \in \{1, 2, \ldots, N_g\}$), and the measurements are preprocessed by subtracting various effects (such as the geoid and centrifugal force from the rotation of the earth) so the resulting observed values only depend on the mass distribution $\rho(\boldsymbol{x})$ inside the image domain Ω. See Fig. 5.3 for a depiction of the modeled physical configuration. Each location $\boldsymbol{x}_i$ then has an associated vector gravity field $\vec{g}_i = (g_{x,i}, g_{z,i})^{\mathrm{T}}$ which can be modeled as:

$$\vec{g}_i = G \int_{\Omega} \frac{\rho(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{x}_i)}{\|\boldsymbol{x} - \boldsymbol{x}_i\|^3} \mathrm{d}\boldsymbol{x} \tag{5.13}$$

with G the universal gravitational constant. This is simply an expression of Newton's law of universal gravitation. Note that trying to estimate ρ from $\vec{g}_i$ is a very ill-posed problem. In the first example we show in this section, there are 512 surface measurements from which we reconstruct an image containing 65,536 pixels.

We make the simplifying assumption that the mass density is constant within both salt and non-salt regions. For the salt bodies, this is a fairly accurate assumption save

be less computationally efficient than gradient-based methods.

[4]Measurement noise can also be added to the experiments (*e.g.*, additive white Gaussian noise). The results are largely similar to the results we present, though the uncertainty in the segmentation naturally increases. We show an example with noise in Sec. 5.3.2.

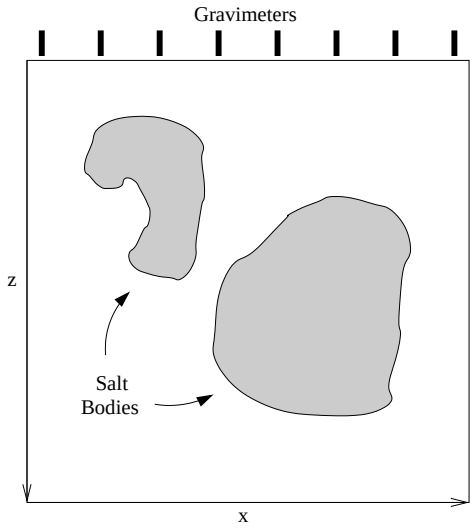


Figure 5.3. Depiction of idealized physical model for gravity inversion problem. Salt regions are shown in gray, background (rock) regions are in white, and an array of evenly-spaced gravimeters are shown at the top.

for the increase of density with respect to depth. For non-salt regions, more complex models incorporating so-called *stratigraphic* information (*i.e.*, rock layer structure) can be utilized when working with real gravity measurements. This will complicate the segmentation process by introducing a set of parameters to estimate.

This piecewise constant assumption results in a simple form for ρ:

$$\rho(\boldsymbol{x}; \vec{C}) = \rho_0 + \Delta\rho\mathcal{H}(-\Psi_{\vec{C}}(\boldsymbol{x})) \ . \tag{5.14}$$

We can actually simplify this further if ρ_0 is known *a priori* by subtracting ρ_0 everywhere (adjusting the gravity measurements as well):

$$\rho(\boldsymbol{x}; \vec{C}) = \Delta\rho\mathcal{H}(-\Psi_{\vec{C}}(\boldsymbol{x})) \ . \tag{5.15}$$

This means that ρ is zero in non-salt regions and $\Delta\rho$ inside the salt and simply captures the density anomaly caused by the presence of salt. We can then combine (5.15) with (5.13) to form the following forward gravity model (where the gravity values are now a function of $\vec{C}$):

$$\vec{g}_i(\vec{C}) = \Delta\rho G \int_{\mathcal{R}_{\vec{C}}} \frac{(\boldsymbol{x} - \boldsymbol{x}_i)}{\|\boldsymbol{x} - \boldsymbol{x}_i\|^3} \mathrm{d}\boldsymbol{x} \ . \tag{5.16}$$

Let $\vec{\xi}_i$ be the observed gravity measurement at each $\boldsymbol{x}_i$. We measure the error between our forward model and the observations using the computationally-simple L2 distance. This leads to a method for which derivatives are easy to formulate analytically and enables us to compare our sampling approach with gradient-based curve evolution. We form the overall energy functional by computing the error between each observed and curve-generated gravity measurement:

$$\mathcal{E}_i^2(\vec{C}) = \|\vec{g}_i(\vec{C}) - \vec{\xi}_i\|^2 \ , \tag{5.17}$$

summing these errors, and adding a regularizing curve prior:

$$E(\vec{C}) = \sum_{i=1}^{N_g} \mathcal{E}_i^2(\vec{C}) + \alpha \oint_{\vec{C}} \mathrm{d}s \ . \tag{5.18}$$

This then results in a target distribution of

$$\pi(\vec{C}) \propto \exp(-E(\vec{C})) \ . \tag{5.19}$$

Note that (5.18) is quite computationally complex, requiring $\mathcal{O}(N_g M^2)$ computation for an $M \times M$ image. Due to the linearity of the computation in (5.13), we can use an update version (as described on page 67 in Sec. 3.3) to compute the energy of $\vec{\Gamma}$ (a curve derived from $\vec{C}$) by updating $E(\vec{\Gamma})$ from $E(\vec{C})$ using only the pixels that have changed label. This version has complexity $\mathcal{O}(N_g M)$ (where the number of changing pixels is assumed to be proportional to curve length, and the curve length is proportional to M). As the computation of the other aspects of the narrowband-based curve sampling method is $\mathcal{O}(M)$, computing the forward model forms a significant fraction of the overall computation, especially for large images[5].

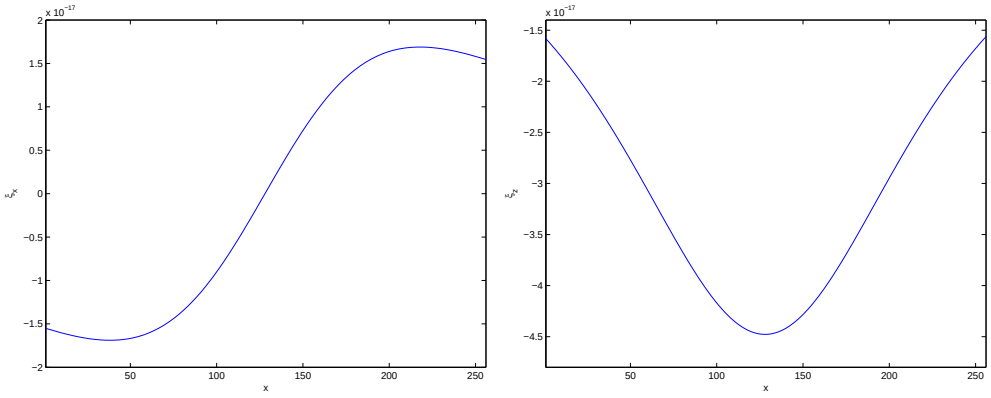
This L2-based energy functional is straightforward to differentiate with respect to the curve. There is a standard curve evolution result (see Tsai [117] for a derivation) that the gradient flow for

$$E(\vec{C}) = \iint_{\mathcal{R}_{\vec{C}}} F(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \tag{5.20}$$

is simply

$$\frac{\mathrm{d}\vec{C}}{\mathrm{d}t}(p) = -F(\vec{C}(p))\vec{\mathcal{N}}_{\vec{C}}(p) \ . \tag{5.21}$$

[5]For a 3D implementation, direct computation is $\mathcal{O}(N_g^2 M^3)$ and an update version is $\mathcal{O}(N_g^2 M^2)$ while the numerical aspects of the narrowband surface evolution requires $\mathcal{O}(M^2)$ computation per iteration. Approximations based on fast multipole methods [37] or multiresolution methods can be employed to increase computational efficiency if necessary.



(a) (b)

Figure 5.4. Observed gravity profile for circular synthetic salt body. (a) ξ_x. (b) ξ_z.

Thus we see that, after applying the chain rule and the definition of $\vec{g}_i(\vec{C})$ in (5.16), the gradient flow for each measurement error term in (5.17) is

$$\frac{\mathrm{d}\mathcal{E}_i^2}{\mathrm{d}t}(p) = -2\Delta\rho G \frac{\left\langle \vec{g}_i(\vec{C}) - \vec{\xi}_i,\, \boldsymbol{x}_i - \vec{C}(p) \right\rangle}{\|\boldsymbol{x}_i - \vec{C}(p)\|^3} \vec{\mathcal{N}}_{\vec{C}}(p) \ , \tag{5.22}$$

and the overall gradient flow is

$$\frac{\mathrm{d}\vec{C}}{\mathrm{d}t}(p) = \sum_{i=1}^{N_g} \frac{\mathrm{d}\mathcal{E}_i^2}{\mathrm{d}t}(p) - \alpha\kappa_{\vec{C}}(p)\vec{\mathcal{N}}_{\vec{C}}(p) \ . \tag{5.23}$$

The derivative of each L2 error term can be seen to be the error between the model and observed gravity $\vec{g}_i(\vec{C}) - \vec{\xi}_i$ projected onto the line connecting $\vec{C}(p)$ and $\boldsymbol{x}_i$ and scaled by the inverse square of the distance between $\vec{C}(p)$ and $\boldsymbol{x}_i$.

■ 5.3.2 Circular Salt Example

We begin with an example with a single salt body with simple geometry. Computationally, evaluation of the forward model is quite slow, and convergence appears to be slower, perhaps because there is weaker coupling between the curve and the observations due to the inverse nature of the problem. Overall computation time is 60 seconds per sample for 6000 iterations.

In Fig. 5.4, we show the observed vector gravity profile. For consistency with the geophysics literature, we use x to indicate the horizontal axis and z the vertical axis. The

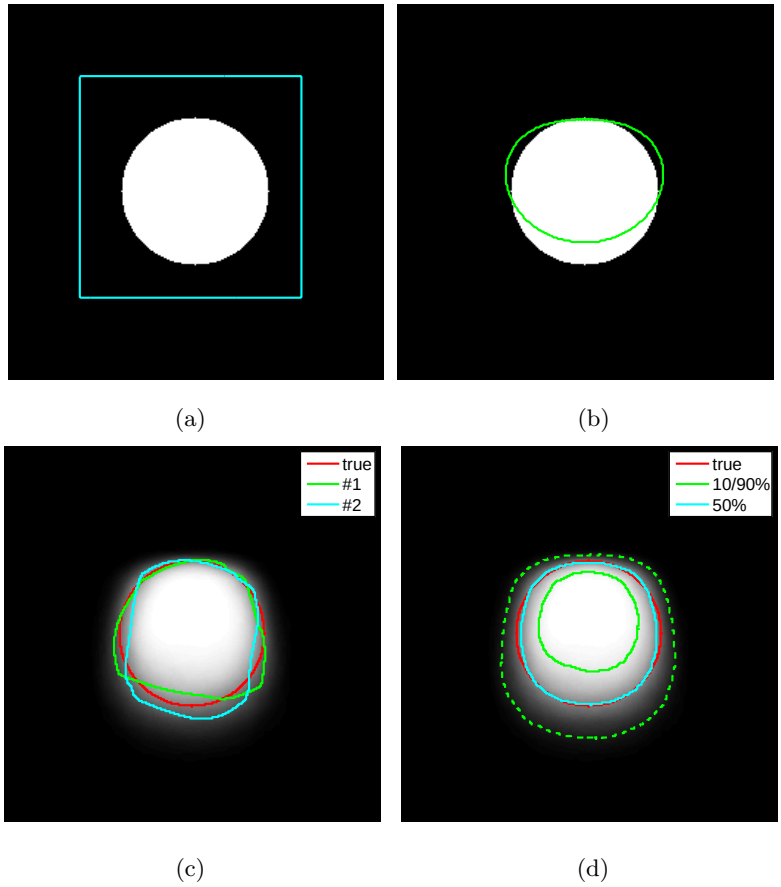
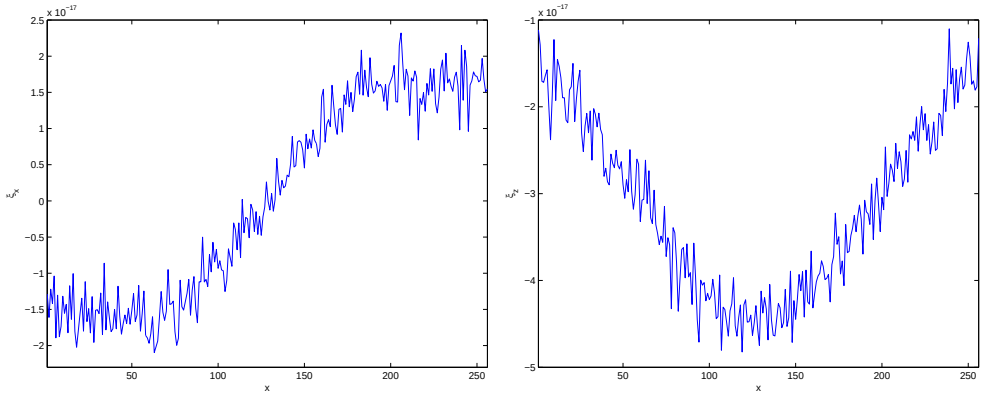


Figure 5.5. Synthetic circular salt body example. (a) Initial curve. (b) Local minimum found using gradient descent. (c) Most likely samples and (d) marginal confidence bounds for unconstrained curve sampling.



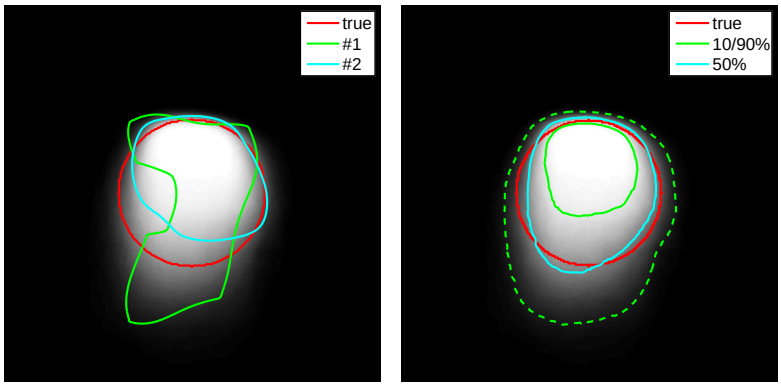
(a) (b)

Figure 5.6. Observed gravity profile for circular synthetic salt body with white Gaussian noise added. Overall SNR is 20 dB. (a) ξ_x. (b) ξ_z.

initialization used for both the optimization-based approach and the sampling approach is shown in Fig. 5.5(a). The white object represents the location of the salt body. In Fig. 5.5(b) we can see the result for the gradient flow with α experimentally tuned to produce the best results. The resulting curve is somewhat close to the true boundary location but has under-segmented the bottom salt due to the regularization parameter. As we noted earlier, the gradient flow in (5.22) for the error term for each individual gravimeter is inversely proportional to the square of the distance from the curve point to the gravimeter, but the $-\kappa_{\vec{C}}$ term does not have a similar behavior. Thus we can see that this results in a disproportionately strong regularizing force at the bottom of the salt. This is not surprising since any prior distribution does introduce a bias to the results of an estimation algorithm. The curve length penalty can be adjusted for depth to counteract this effect, but it is not clear that is desirable because there is more uncertainty at the bottom of the salt than at the top due to the weaker gravity coupling.

In Fig. 5.5(c)-(d) we show the output of our sampling algorithm. The two most likely samples are quite close to the true salt boundary, and the median contour is as well. The 10/90% confidence bounds enclose the true salt boundary, and, as we would expect, there is more uncertainty at the bottom of the salt body than at the top due to the inverse relationship with distance in the measurement model.

Adding noise to the observations can have fairly large effects on the inversion process



(a) (b)

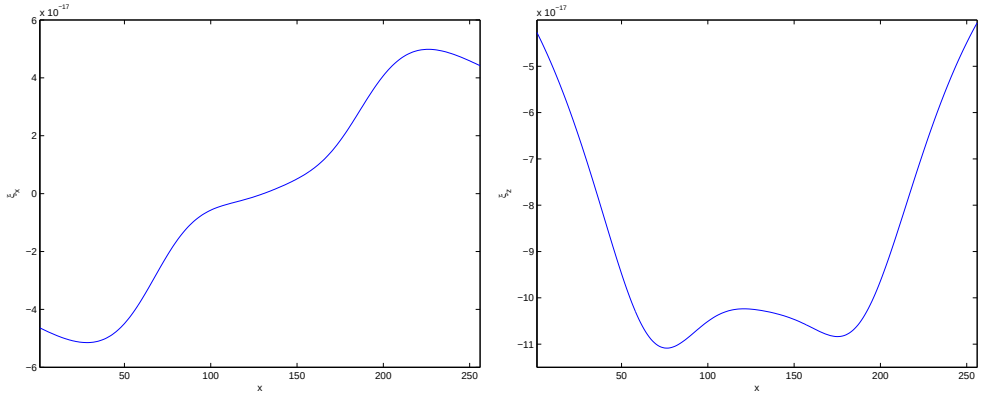
Figure 5.7. Synthetic circular salt body example with noise. (a) Most probable samples and (b) marginal confidence bounds for unconstrained curve sampling.

due to the ill-posed nature of the problem, even for relatively high SNR levels. In Fig. 5.6, we show observed gravity profiles with white Gaussian noise added. The variance of the noise is chosen so that the SNR is 20 dB. The overall shape of the gravity profiles remains similar to those in Fig. 5.4. We show the results of running our sampling algorithm on these observations in Fig. 5.7. Comparing the results here with those in Fig. 5.5, we can see that the top salt location is still quite accurate with only a small gap between the 10% and 90% confidence bounds. The bottom salt exhibits much larger variance, and the most probable samples do as well. Overall, though, the marginal confidence bounds still enclose the true salt boundary, and the median contour is quite close to it also.

■ 5.3.3 Synthetic Two-body Salt Example

Fig. 5.8 shows the gravity profile for the salt body configuration in Fig. 5.9(a). This example is much more challenging than the circular salt body due to the more complex geometry and two disjoint regions. We use the same approach described in Sec. 5.2 to use separate narrowband level set functions to represent the two salt bodies. Note that in real-world gravity inversion problems, it is unlikely that the number of disjoint regions is known. In these cases, an expert user can provide guidance, or multiple experiments can be run with varying numbers of regions.

In Fig. 5.9(a) we show the initialization used to start the sampling process with the



(a) (b)

Figure 5.8. Gravity profile for a synthetic salt body example with two disjoint regions and complex geometry. (a) ξ_x. (b) ξ_z.

top of the salt set to the correct location. This is a reasonable thing to do because in many gravity inversion problems, the gravity data is acquired simultaneously with a seismic image from which it is much easier to locate the top salt due to the strong reflection at the rock/salt interface.

The result of starting from that initialization using the optimization-based curve evolution approach is displayed in Fig. 5.9(b). Again, the regularization parameter α was optimized to produce the best result. As in the circular salt example, the upper part of the object is fairly reasonable, but, due to the ill-posedness and amount of uncertainty of the problem, even starting from an initialization that is fairly close to the true solution still results in the depth of the bottom salt being severely underestimated. Fig. 5.9(c)-(d) shows the output of our unconstrained curve sampling approach. While the 50% confidence contour is close to the true salt location in a coarse sense, finer-scale details are largely non-existent. The 10% confidence bound does provide a reasonable envelope for the location of the bottom salt. Due to the increased complexity in the geometry and the two separate regions, computation time per sample is around 3 minutes with 10,000 iterations per sample.

To further refine these estimates, we can fix the location of the top salt instead of simply using the information in a weak way in the initialization. We show the results of this approach in Fig. 5.10(a)-(b). The most likely samples for this version are close to the true boundary, and the geometry near the top of the salt body is much better

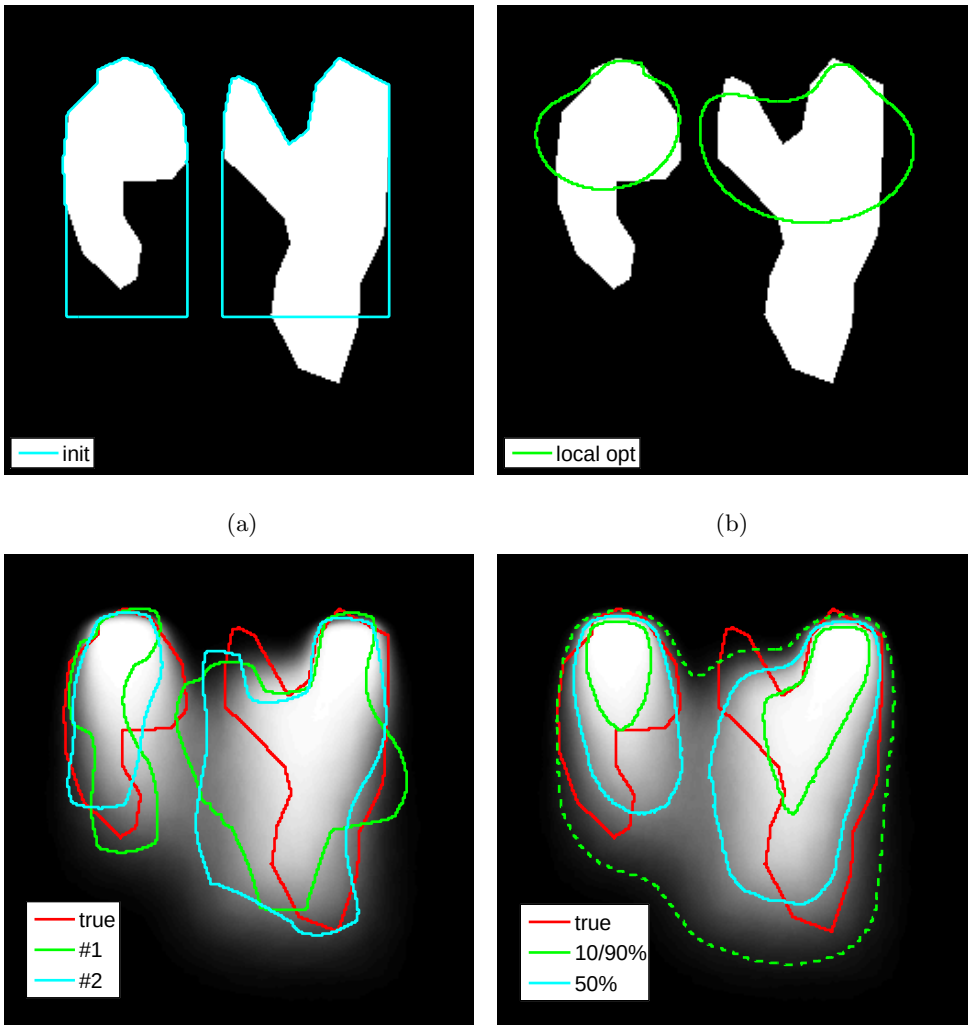


Figure 5.9. Synthetic salt body example with two disjoint regions and complex geometry. (a) Initial curve. (b) Local minimum found using gradient descent. (c) Most likely samples and (d) marginal confidence bounds for unconstrained curve sampling.

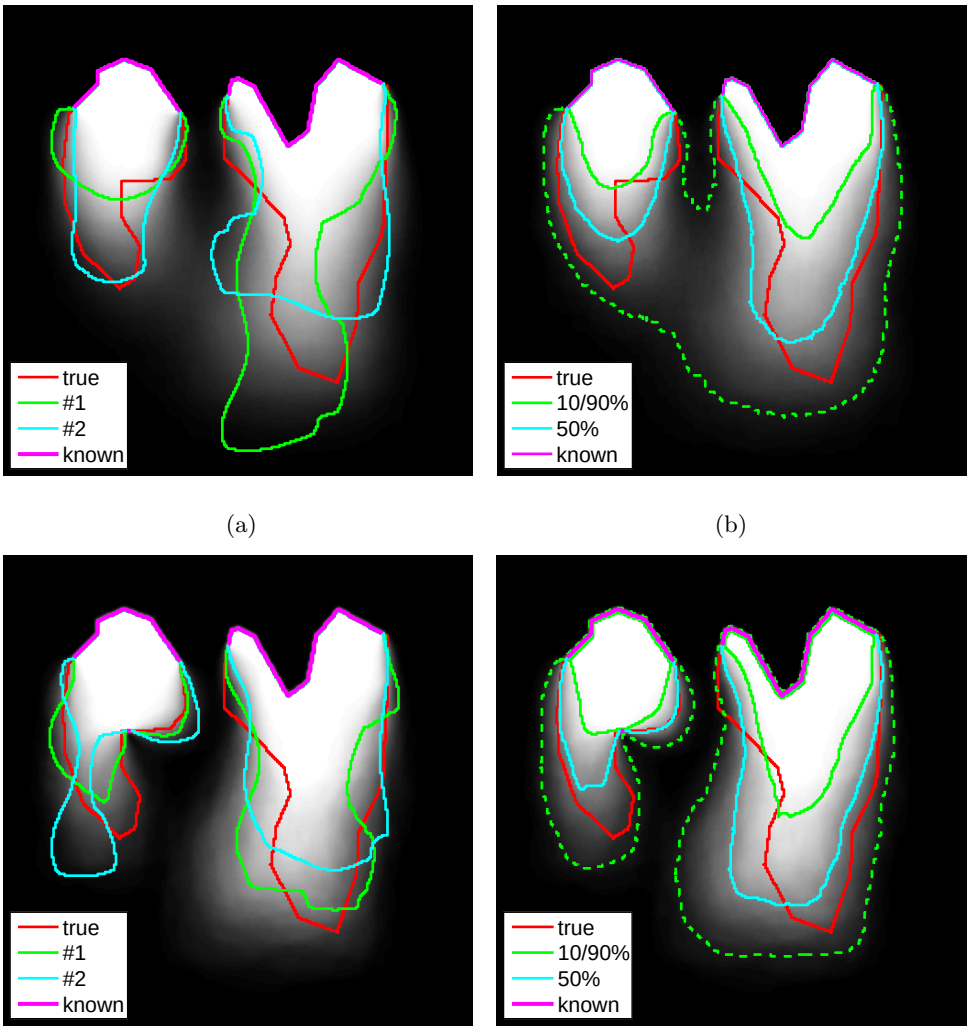


Figure 5.10. Synthetic salt body example with two disjoint regions and complex geometry. Conditional simulation with top salt fixed in examples in the top row, top salt and a bottom salt point fixed in the bottom row. (a) and (c) show two most likely samples; (b) and (d) show the marginal confidence bounds and histogram images. The known portion of the curve is shown in magenta.

preserved.

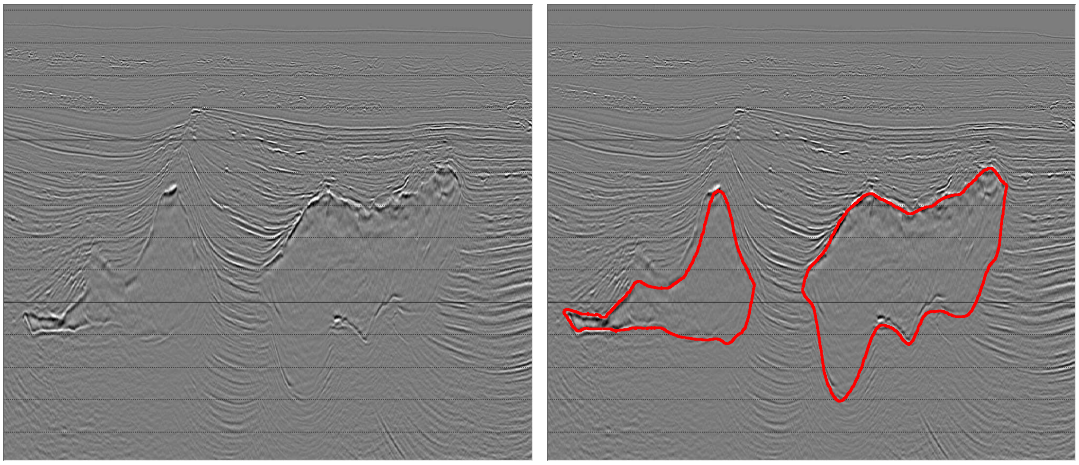
One consistent problem is with areas known as *recumbencies* where the interface bows inward and the boundary has strong negative curvature. Geometric features such as this are unlikely to be generated by the proposal distribution defined in Sec. 5.1, and due to the ill-posed nature of the gravity inversion problem, there is not a strong indication from the observed data that these recumbencies exist. Different proposal distributions could improve the performance by using a smoothing term which is less biased against shapes with negative curvature (*e.g.*, $\kappa_{\vec{C}}^{1/3}$ regularization) or by explicitly generating specific types of geometric features. Rather than modifying the proposal distribution, we can instead change the target distribution by incorporating additional curve constraint information. These constraints could be provided by a geologist who has some hypotheses as to some bottom salt locations due to the stratigraphic structure surrounding the salt body; from observed anomalies in seismic images of the image domain or other data acquisition modalities; or from physical sources of information such as an exploratory well which has been drilled to assess the situation.

In Fig. 5.10(c)-(d) we show such an example where both the top salt and an additional point on the recumbency on the left salt body are fixed. After incorporating this additional constraint, we observe that the overall geometric structure of the left salt body is now much more similar to the true boundary. The 10 and 90% confidence bounds fully bracket the true contour, and the 90% bound clearly splits the right and left salt bodies into separate pieces. Overall, simply specifying small regions on sharp geometric features is a powerful tool to increase the overall segmentation accuracy.

■ 5.3.4 Real Geometry Example

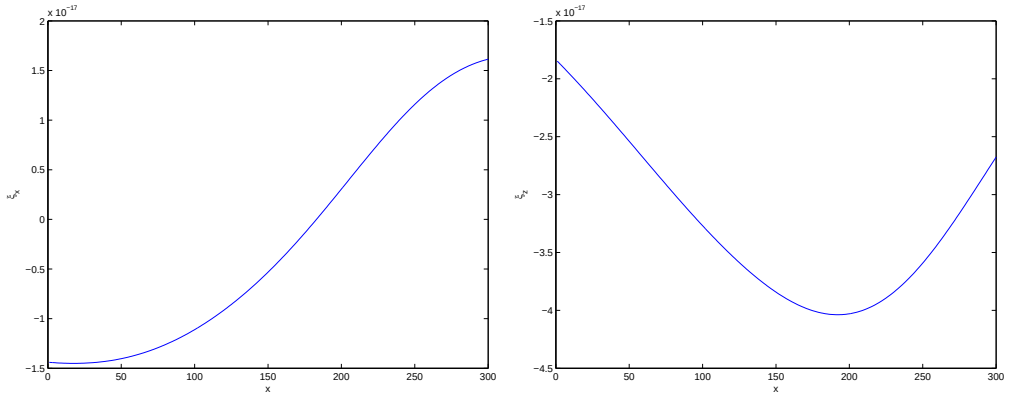
The salt bodies used in the previous examples were purely synthetic in nature, though the level of geometric complexity in the two-body problem is similar to real-world examples. In this section we take a real seismic image (shown in Fig. 5.11(a)) which has already been manually interpreted by an expert geologist. We show in Fig. 5.11(b) the contour the geologist has drawn based on information in the seismic image. We then use that salt interpretation to generate a synthetic gravity measurement according to the model in (5.13). This results in an example which has geometry that is indicative of a real-world example but still uses our simplified forward gravity model. We show the resulting gravity profile in Fig. 5.12.

For this particular gravity inversion example, the left salt body (in Fig. 5.11(b)) has



(a) (b)

Figure 5.11. Seismic image containing two salt bodies. (a) Observed seismic image. (b) Same image with expert-determined boundaries overlaid.



(a) (b)

Figure 5.12. Gravity profile for real salt body example. (a) ξ_x. (b) ξ_z.

a long, thin region in its left half. This means that even relatively small changes of the curve can cause the bottom of the curve to intersect the top. To address this issue, we modify the proposal distribution to make topological changes unlikely by adding a term to the mean perturbation inspired by the more-than-topology-preserving curve evolution method developed by Sundaramoorthi and Yezzi (see Sec. 2.2.1)[6]. In that method, an energy functional is formulated such that curves approaching a self-intersection cause the energy to go to infinity. The resulting gradient flow contains a force that pushes curve points away from each other with magnitude inversely proportional to the square of the distance between the points.

Rather than implementing the full topology-preserving gradient flow (described in (2.26), (2.28), and (2.29)), we add a simpler term to the mean perturbation of our proposal distribution $\mathtt{q}$ which has a similar effect:

$$\mu_{\vec{C}_{\mathrm{a}}}(p) = -\alpha\kappa_{\vec{C}_{\mathrm{a}}}(p) + \gamma_{\vec{C}_{\mathrm{a}}} + \beta \max_{q\in[0,1]\setminus B_\epsilon(p)} \frac{1}{\|\vec{C}_{\mathrm{a}}(p) - \vec{C}_{\mathrm{a}}(q)\|^2} \tag{5.24}$$

where $B_\epsilon(p)$ is an interval with radius ϵ around p (modulo 1 so the interval wraps around 0 when $p < \epsilon$ or around 1 when $p > 1 - \epsilon$). Note that this new force term is always positive, so it is a balloon force which can only move curves outward. Therefore it can only prevent topology changes which occur as the curve tries to pinch off and split into two and not when different curves or different sections of a curve try to merge. The latter scenario is not a primary concern for this example, but the full more-than-topology-preserving flow may be needed for situations when that occurs.

The computation of this term is of order $\mathcal{O}(M^2)$, so it is relatively slow compared with the $\mathcal{O}(M)$ terms in the narrowband level set implementation. Unless an example has many self-intersections occurring during the sampling process, using the topology-preserving term may or may not be worth the computational trade-off[7].

[6] We do not alter the target distribution as splitting a single curve into multiple disjoint curves should already be a lower-probability event for both the prior term (as disjoint curves have more curve length) and the likelihood term (because the segmentation will be incorrect). If desired, one can explicitly build a topological constraint into π, though this will increase the amount of required computation. One approach could be to use the energy term of Sundaramoorthi and Yezzi. Another could be based on a model where any topological number of the curve $\vec{C}$ (*i.e.*, the number of disjoint regions) not equal to 1 has zero probability.

[7] A proof-of-concept implementation using Matlab actually decreases the sampling rate for our algorithm by close to 70% when adding this topology-preserving term, so each sample requires approximately 12 minutes to generate. A more production-ready implementation would have a more moderate impact on speed, probably on the order of 10 to 20%.

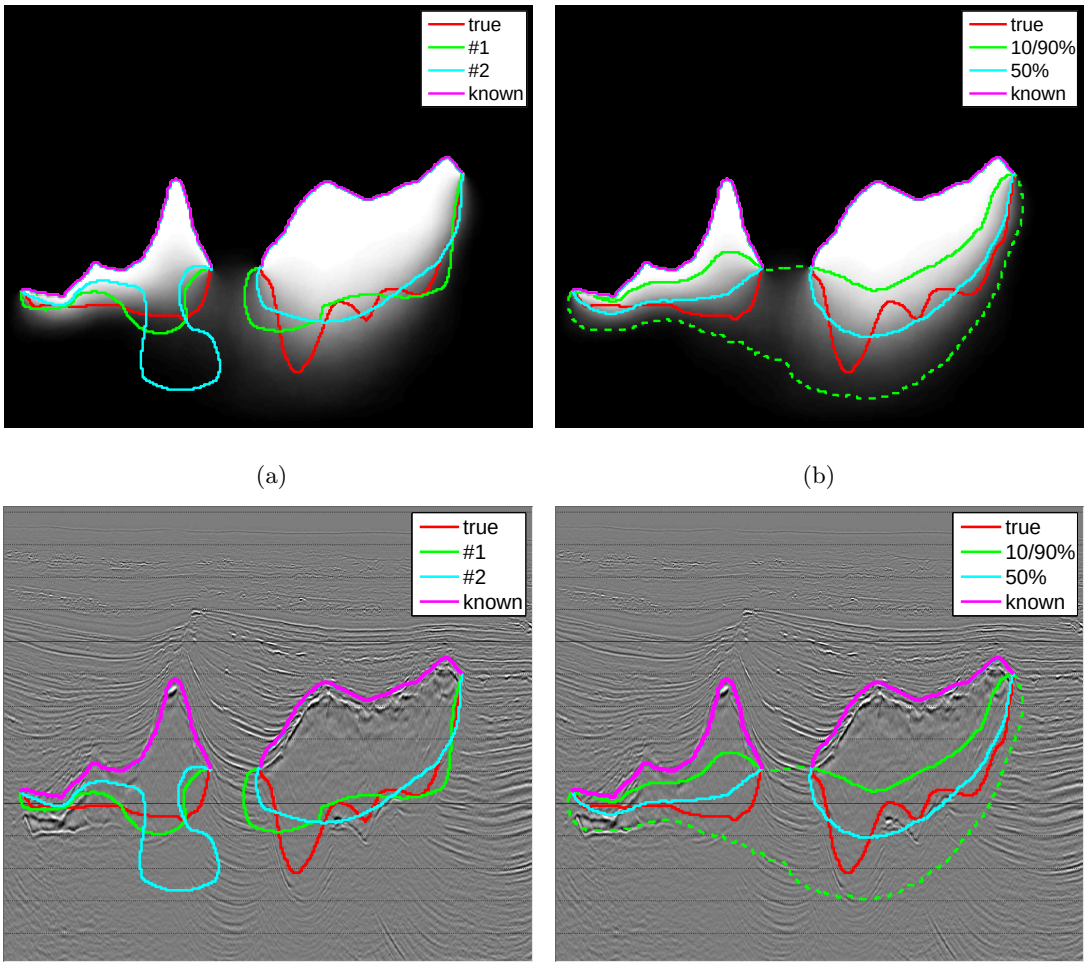


Figure 5.13. Conditional simulation results (with top salt fixed) for two-body salt example derived from seismic image. The two most probable samples are shown in (a) with the marginal histogram image and (c) with the seismic image. Similarly, the marginal confidence bounds are shown in (b) and (d). The top salt locations are shown in magenta.

We show the results using this topology-preserving balloon force for our conditional simulation algorithm in Fig. 5.13. Similar to the previous example, the top salt is given to the algorithm as a fixed constraint. The two most likely samples and the marginal confidence bounds are shown overlaid on both the histogram image and the original seismic image. The complex geometry of the bottom salt is not captured by the sampling process, but the 10/90% confidence bounds do provide reasonable limits on the range of locations for the bottom salt. Note that a joint inversion process involving both the seismic data and the gravity measurements could further improve the quality of the segmentation results. We discuss this possibility in more detail in Chap. 7.

■ 5.3.5 PCA Shape Variation Modes

Marginal statistics are less useful for problems with strong global coupling among different regions of the image as they often cannot adequately capture that kind of behavior. In the gravity inversion problem, the model is fully connected: each measurement $\vec{\xi}_i$ is dependent on every single image pixel through (5.13). As a result, the locations of the top of the salt and the bottom should be correlated. If the estimate of the top is too low, the observed gravity at the surface will be too small. This can be partially offset if mass is added to the bottom of the salt body which results in the estimate of the bottom also being too low.

Principal components analysis (PCA) is one method that is often employed to examine the correlation structure of high dimensional random variables. Here we follow the approach of Leventon *et al.* [65] (see Sec. 2.3.2) and apply PCA to a set of signed-distance functions (SDFs). These SDFs correspond to curves which are samples from π. The PCA algorithm results in a Euclidean mean level set $\bar{\Psi}$, eigenvalues σ_i, and eigenvectors Ψ_i. We can visualize these eigenmodes by displaying the zeroth level set of $\Psi = \bar{\Psi} \pm \sigma_i \Psi_i$. Ψ here is then one standard deviation away from $\bar{\Psi}$ in the direction of Ψ_i.

We show the results of this procedure in Fig. 5.14 for three different *versions* of the synthetic two-body experiment: Version A which uses the unconstrained samples; Version B which has the top salt fixed; and Version C which has the top salt and a point on the recumbency fixed. We display the mean contour in yellow and the true salt boundary in red. We show the $+\sigma_i$ deviation using the solid green line and the $-\sigma_i$ mode with the dashed green line.

Viewing these eigenmode contours allows us to see how the physics of the inversion

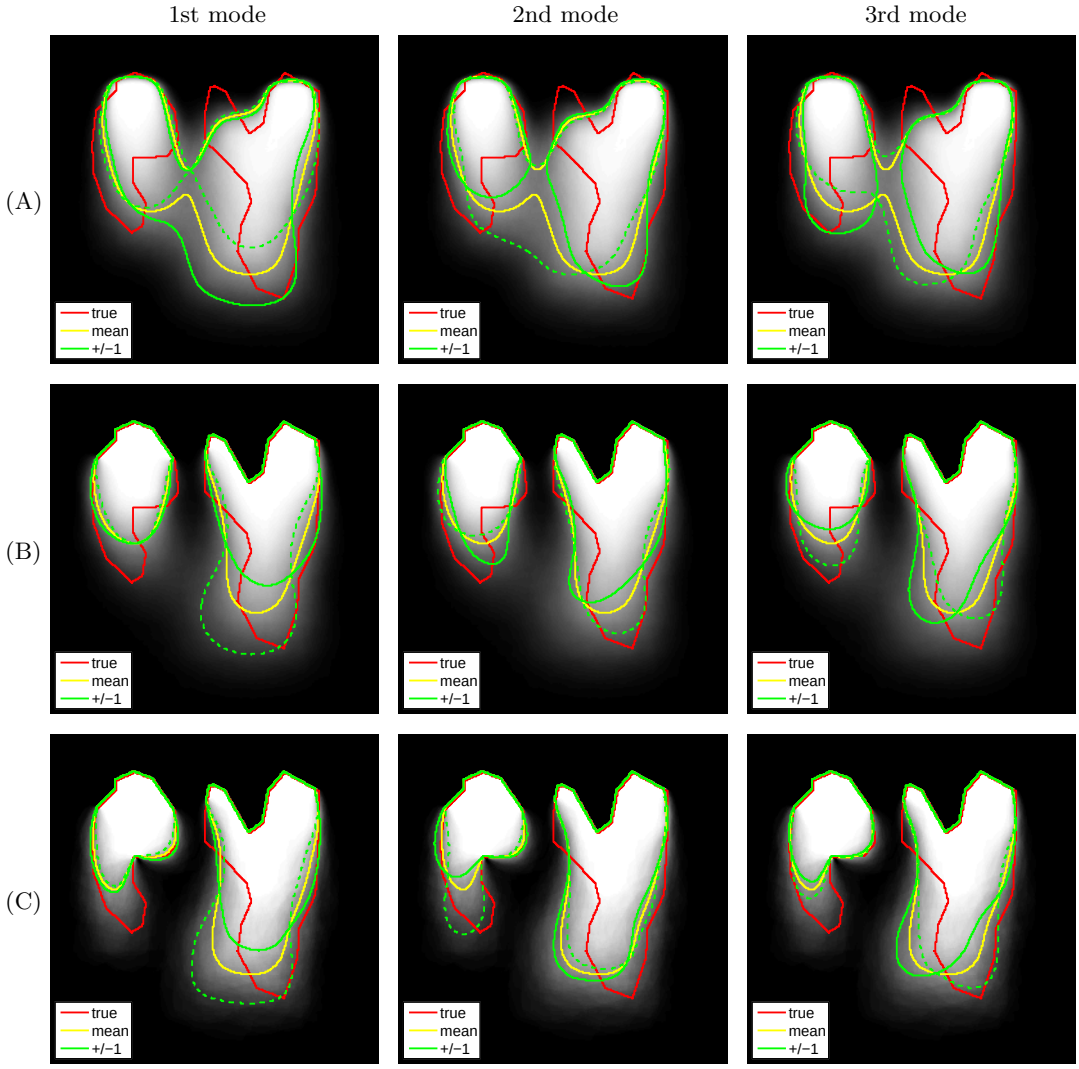


Figure 5.14. Principal modes of variation for the synthetic two-body salt example. The columns contain the three largest eigenmodes (for $\pm\sigma_i$) sorted from biggest to smallest. The first row contains Version A using the unconstrained samples; the second row has Version B with the top salt fixed; and the third row has Version C with both the top salt and a point on the recumbency fixed.

problem is captured in the structure of the probability distribution. For instance, for all three versions, the first eigenmode (in the first column of Fig. 5.14) corresponds to up-and-down movement of the location of the right bottom salt. This can be seen from the fact that the solid ($+\sigma_1$) and dashed ($-\sigma_1$) contours differ from each other primarily in the vertical direction at the bottom of the right salt body. This behavior corresponds with our physical intuition of the problem as this region is farthest from the gravimeters and thus the most weakly coupled to the measurements. We also note that in this eigenmode, the depth of the right salt bottom appears to be largely uncorrelated with the location of the left salt body as evidenced by the limited variability of the green contour for the latter.

The behavior of the second mode (shown in the middle column of Fig. 5.14) varies across the three versions. For Version A, the solid green line for the $+\sigma_2$ mode joins the two salt bodies and the dashed green line for the $-\sigma_2$ mode separates them. Version B has anti-correlated up-and-down movements in the left and right salt bottoms. The $+\sigma_2$ solid green line in the right salt body is higher than the yellow mean curve while the corresponding solid green line in the left salt body is lower than the mean contour. The opposite holds for the dashed green line of the $-\sigma_2$ mode. Version C has a mode that follows the movement of the bottom of the left salt body while largely leaving the right salt body segmentation unaffected. In fact, the dashed green contour for the $-\sigma_2$ mode almost perfectly segments the true left salt body (shown in red).

The primary effect of the third mode for all three versions is a left-and-right movement of the lower part of the right salt body with the solid and dashed green curves bracketing the mean curve in the horizontal direction. Note that there is a positive correlation between this movement and the movement of the left salt body. For Version A, when the $+\sigma_3$ solid green contour for the right salt body is to the right of the yellow mean curve, this forces the dashed green contour to be lower than the mean curve for the left salt body. In this situation, the right salt body has less area than the mean contour case which decreases the overall measured gravity at the surface. The decreased gravitational force is partially counteracted by adding mass to the left salt body. For Versions B and C, if the lower part of the right salt body moves to the right, the bottom of the left salt body is lower (and vice versa). This is because when the right salt body moves to the right, it moves away from most of the gravimeters which decreases the amount of gravitational force it contributes. This is again partially offset by an increase in mass in the left salt body.

PCA shape models can be a powerful tool to analyze and visualize the structure of the probability distribution embedded in the curve samples. The modes of variation show the areas in which there is the greatest uncertainty and direction in which that uncertainty exists. This provides a much richer description than simply using marginal statistics as correlations among different objects can be captured. We discuss possible extensions to further characterize the structure of the target distribution in Chap. 7.

Chapter 6

Hybrid 2D/3D Surface Sampling

TRADITIONAL curve evolution methods generalize in a natural fashion to three dimensions (3D). The image domain Ω becomes a subset of $\mathbb{R}^3$, and a surface $\vec{S} : [0,1] \times [0,1] \rightarrow \Omega$ is embedded in Ω to segment the image. Gradient descent formulations can be constructed in an analogous fashion to the 2D case, and the surface evolution can be performed with a level set formulation (with Ψ now a surface in $\mathbb{R}^4$).

Unfortunately, a similarly straightforward extension to 3D for our curve sampling approach is not possible. With 2D curves, we can take an arbitrary curve $\vec{C}$ and write a geometrically-equivalent curve parameterized by arc length $\vec{C}_{\mathrm{a}}$ (described in Sec. 3.2.1). For 3D surfaces, such a standard parameterization for $\vec{S}(p,q)$ does not exist in general[1]. This is problematic because application of the forward perturbation explicitly depends on the parameterization chosen (so applying a perturbation f to two geometrically-equivalent surfaces with different parameterizations $\vec{S}_1$ and $\vec{S}_2$ would result in geometrically-different candidate samples) as does the computation of the reverse perturbation ϕ.

Due to these limitations, we construct a hybrid 2D/3D surface model where we view the surface as a collection of curves (typically on parallel planes) which are combined together to form a complete surface. A slice-based approach can also be a natural representation because in many applications, experts process 3D volumes on a slice-by-slice basis (*e.g.*, radiologists segment objects by drawing curves in each slice, not as a complete surface in 3D). Interaction between the curves is described by a first-order undirected Markov chain whose nodes are entire curves, and curves on neighboring slices are joined by edges of the graph. Then instead of directly specifying a surface sampling algorithm, we describe how to generate samples from this hybrid model. As

[1] If only convex surfaces are considered, a number of unique parameterizations can be constructed (*e.g.*, one based on spherical coordinates).

in the 2D curve sampling case, we can extend the framework to allow for conditional simulation where partial curves or entire curves are specified.

In Sec. 6.1, we describe the construction of the hybrid 2D/3D Markov chain surface model. Sec. 6.2 shows an application of our 2D/3D Markov chain model for the case when we are given the values of the slices directly above and below a slice of interest and how this simplifies the problem. A generalization of this approach is described in Sec. 6.3 for the case when there are multiple contiguous unknown slices. Approaches based on Gibbs sampling and the embedded HMM approach of Neal *et al.* (see Sec. 2.1.3) are presented, both of which use our 2D curve sampling techniques as the main computational engine. An alternative conditional simulation formulation which uses expert segmentations in planes orthogonal to the slices in the Markov chain model is considered in Sec. 6.4.

■ 6.1 2D/3D Markov Chain Model

In this section, we describe the general construction of our hybrid 2D/3D surface approximation using locally-specified *potential functions* describing the interactions between neighboring slices. We discuss the advantages and disadvantages of such a model and describe one particular probability model constructed based on symmetric area difference (SAD).

Let $\Omega \subset \mathbb{R}^3$ be the image domain and $Y : \Omega \to \mathbb{R}$ be an observed image volume (as before, the formulation is easily extended to incorporate vector observations). We approximate a surface $\vec{S} : [0,1] \times [0,1] \to \Omega$ with a collection of 2D curves on parallel planes $\mathcal{S} = \{\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_N\}$. We generally view the curves as being on slices parallel to the x-y plane and each $\vec{c}_i$ having a z-coordinate of $i\Delta z$ (where Δz is a fixed slice thickness).

The overall probability distribution we describe for our surface model is based on a first-order Markov chain graph structure, though we note that a generalization to higher-order Markov models is straightforward. An example of such a model is shown in Fig. 6.1 Here nodes in the graph represent either entire 2D curves $\vec{c}_i$ or associated local observations y_i. Requiring the observations to be local is usually a reasonable restriction because image observations in many applications are local in nature (*e.g.*, assuming ideal image formation, a pixel value in an MR image corresponds to a specific measured value at that location in space). For situations in which observations are

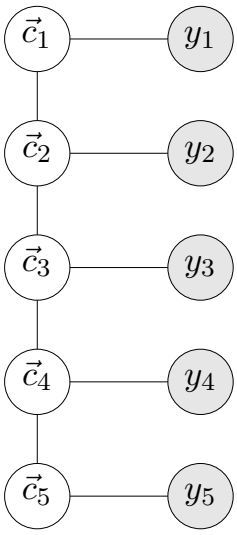


Figure 6.1. Model of 3D volume using 2D slices connected as a Markov chain. The y_i are observations and $\vec{c}_i$ are curves in each slice.

more global (*e.g.*, the gravity measurements in Sec. 5.3), more complicated graphical models may need to be constructed.

We view the Markov chain as an undirected graphical model (see pg. 35 in Sec. 2.1.2). The Hammersley-Clifford theorem [42] tells us that the joint probability distribution over $\mathcal{S}$ and Y factorizes into a product of non-negative potential functions:

$$\mathrm{p}(\mathcal{S}, Y) \propto \prod_{i=1}^{N} \Phi_i(\vec{c}_i) \prod_{i=1}^{N-1} \Phi_{i,i+1}(\vec{c}_i, \vec{c}_{i+1}) \prod_{i=1}^{N} \Lambda_i(\vec{c}_i, y_i) \ . \tag{6.1}$$

Our target distribution is the posterior probability $\pi(\mathcal{S}) = \mathrm{p}(\mathcal{S} \,|\, Y) \propto \mathrm{p}(\mathcal{S}, Y)$ (leaving the observed data implicit in π).

The edge potential functions Λ_i are defined between nodes containing curves and nodes containing observations. These are akin to data likelihood terms, and as in the 2D case, specification of Λ_i is related to the measurement and noise models of a given problem. The self potential functions Φ_i are defined on the nodes themselves. They represent local prior information and can be, *e.g.*, a shape model or a smoothness-enforcing curve length prior.

The previous two types of terms are analogous to ones which appear in 2D curve sampling formulations. The main addition for our hybrid 2D/3D approach is the presence of the edge potential functions $\Phi_{i,i+1}$ which are defined between nodes on the graph

corresponding to adjacent curves. These edge potentials can be seen as coupling terms that give 3D structure to the slices and should incorporate knowledge about how we expect neighboring slices to behave with respect to each other. This can be in the form of dynamical shape models [103] or shape distance-based terms. The latter approach is the one we take in this chapter. Shape distances are a logical choice for an underlying coupling term as generally we expect the curve in one slice to be similar to the curve in a neighboring slice. We can then exponentiate the distance to obtain an edge potential function (with a positive scalar ξ to control how strong the coupling is):

$$\Phi_{i,i+1}(\vec{c}_i, \vec{c}_{i+1}) = \exp(-\xi \mathrm{d}(\vec{c}_i, \vec{c}_{i+1})) \ . \tag{6.2}$$

There are a number of existing shape distances available such as Hausdorff distance or L2 between signed distance functions [20, 56, 73]. Here we discuss the use of symmetric area difference (SAD).

■ 6.1.1 Symmetric Area Difference as a Coupling Term

As discussed in Sec. 4.2.1, the SAD between two curves $\vec{C}_1$ and $\vec{C}_2$ is the area of the image domain Ω where the two curves disagree on the image label. We repeat the definition of the distance from (4.1) for convenience:

$$\mathrm{d}_{\mathrm{SAD}}(\vec{C}_1, \vec{C}_2) = \iint_{\mathcal{R}_{\vec{C}_1} \setminus \mathcal{R}_{\vec{C}_2}} \mathrm{d}\boldsymbol{x} \ + \iint_{\mathcal{R}_{\vec{C}_2} \setminus \mathcal{R}_{\vec{C}_1}} \mathrm{d}\boldsymbol{x} \ . \tag{6.3}$$

This then results in edge potential functions

$$\Phi_{i,i+1}(\vec{c}_i, \vec{c}_{i+1}) = \exp(-\xi \mathrm{d}_{\mathrm{SAD}}(\vec{c}_i, \vec{c}_{i+1})) \ . \tag{6.4}$$

To see why SAD is an appropriate choice, we illustrate the connection between it and Euclidean surface area, a classical surface regularizing energy term:

$$\mathrm{E}(\vec{S}) = \oiint_{\vec{S}} \mathrm{d}A \ . \tag{6.5}$$

To form an approximation to the surface area using our slice-based model, we construct surfaces $\vec{c}_i \oplus \vec{c}_{i+1}$ (which are only a function of $\vec{c}_i$ and $\vec{c}_{i+1}$) connecting every neighboring pair of curves. For $\vec{c}_1$ and $\vec{c}_N$, the curves on the top and bottom slices, surfaces $\mathbf{0} \oplus \vec{c}_1$ and $\vec{c}_N \oplus \mathbf{0}$ are created that also close off the top and bottom of the object. This results

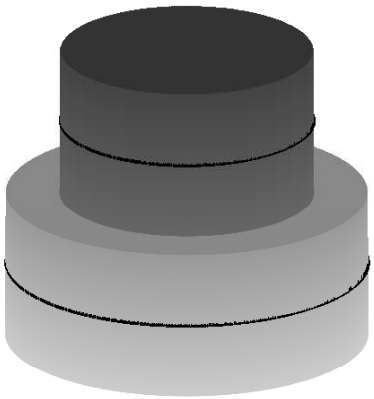


Figure 6.2. Zero-order hold approximation to a 3D surface from curves located on parallel slices. The curves are drawn in black. The surface is constructed by extending each curve up by $\Delta z/2$ and down by the same distance and covering any gaps in the surface.

in an overall approximation to the surface area

$$\oiint_{\vec{S}} \mathrm{d}A \approx \iint_{\mathbf{0}\oplus\vec{c}_1} \mathrm{d}A + \sum_{i=1}^{N-1} \iint_{\vec{c}_i\oplus\vec{c}_{i+1}} \mathrm{d}A + \iint_{\vec{c}_N\oplus\mathbf{0}} \mathrm{d}A \ . \tag{6.6}$$

While one typically regularizes a surface by minimizing the surface area in (6.5), we do not actually want the operator $\oplus$ to construct the minimum-area surface between $\vec{c}_i$ and $\vec{c}_{i+1}$. The minimum surface problem (also known as the soap bubble problem) is one that has been studied rather extensively [83], and the minimum surface connecting two curves in parallel planes actually bows inward (like an hourglass) and has zero mean curvature everywhere. In contrast, a sphere (which is the shape with minimum surface area for a given volume) has constant positive curvature everywhere and bends outward.

Instead of finding minimum surfaces, we construct a surface approximation $\tilde{\vec{S}}$ using a method that creates surfaces which are piecewise-constant in the z-direction. We refer to this approach as a zero-order hold or cylindrical approximation. As shown in Fig. 6.2, each slice has its curve extended upward for $\Delta z/2$ and downward for $\Delta z/2$.

Where the extensions from neighboring slices meet, the holes are covered by a surface which is parallel to the x-y plane. The remaining holes from the top and bottom slices are closed off as well.

The area of the covering surface for curves $\vec{c}_i$ and $\vec{c}_{i+1}$ is simply the symmetric area difference between the two curves. For neighboring curves which have non-zero overlap (*i.e.*, $\mathcal{R}_{\vec{c}_i} \cap \mathcal{R}_{\vec{c}_{i+1}} \neq 0$), this is clear as the gaps between the curve extensions occur in locations where the curves disagree. If the curves are disjoint, we simply cover the ends of each of the cylindrical extensions resulting in the surface splitting into two disjoint surfaces there. The area of the extra surface added in this case is simply the area of $\mathcal{R}_{\vec{c}_i}$ plus the area of $\mathcal{R}_{\vec{c}_{i+1}}$ which again is the SAD between the two curves. For the special case of the top and bottom curves $\vec{c}_1$ and $\vec{c}_N$, we simply close the holes so the covering surface area is equal to the area enclosed by the respective curve.

We can see then that

$$\iint\limits_{\vec{c}_i \oplus \vec{c}_{i+1}} \mathrm{d}A = \frac{\Delta z}{2} \oint_{\vec{c}_i} \mathrm{d}s + \mathrm{d}_{\mathrm{SAD}}(\vec{c}_i, \vec{c}_{i+1}) + \frac{\Delta z}{2} \oint_{\vec{c}_{i+1}} \mathrm{d}s \tag{6.7}$$

which results in an overall surface area approximation of

$$\oiint_{\vec{S}} \mathrm{d}A \approx \oiint_{\tilde{\vec{S}}} \mathrm{d}A = \Delta z \sum_{i=1}^{N} \oint_{\vec{c}_i} \mathrm{d}s + \sum_{i=1}^{N-1} \mathrm{d}_{\mathrm{SAD}}(\vec{c}_i, \vec{c}_{i+1}) + \iint\limits_{\mathcal{R}_{\vec{c}_1}} \mathrm{d}\boldsymbol{x} + \iint\limits_{\mathcal{R}_{\vec{c}_N}} \mathrm{d}\boldsymbol{x} \ . \tag{6.8}$$

Overall, this approximation has the same general effect as the surface area penalty in (6.5). Smaller shapes are preferred over larger shapes and smooth boundaries over jagged ones. With true Euclidean surface area, the minimizing gradient flow is negative mean curvature which smooths out regions of high curvature. Similarly for the approximation using SAD, the gradients of the curve length terms regularize within each slice, and the $\mathrm{d}_{\mathrm{SAD}}(\vec{c}_i, \vec{c}_{i+1})$ terms smooth across slices.

One nice aspect of using SAD as the distance measure is that topological change between slices is handled naturally (*e.g.*, SAD is computable when $\vec{c}_i$ is one simply-connected curve and $\vec{c}_{i+1}$ consists of two disjoint curves). Note, though, that while different slices can have different numbers of curves, as in the 2D case, our sampling-based method continues to require the number of curves within each slice to be fixed *a priori*.

■ 6.1.2 Conditional Simulation Approaches

Our hybrid 2D/3D Markov models can be readily adapted for conditional simulation, especially for the case when a given slice is assumed to be known exactly. With this type of input, the known curve can be treated as an observed node. The idea then is to have an expert segment some of the slices in the image, and the sampling algorithm would fill in the remainder.

Due to the chain structure of the model, an observed slice always breaks the model into two separate pieces (unless the node corresponding to the observed slice is located at either end of the chain). If we have N nodes in the chain and node k is observed, slices $\{\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_{k-1}\}$ and $\{\vec{c}_{k+1}, \vec{c}_{k+2}, \ldots, \vec{c}_N\}$ are independent conditioned on $\vec{c}_k$. This case is equivalent to having two segmentation problems: one with a chain containing $\{\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_k\}$ and another containing $\{\vec{c}_k, \vec{c}_{k+1}, \ldots, \vec{c}_N\}$ with $\vec{c}_k$ observed in each one.

Due to this decomposition, we can assume without loss of generality that any problem we consider only has a single contiguous sequence of unknown slices. Any other situation can be decomposed into smaller problems with this structure. This means that for a chain of length N, all nodes but the first and last are unknown. The first and last may or may not be known.

Note that it is also possible to specify partial curves within a slice in addition to or instead of specifying entire curves. For the ith slice, the curve variable at that node decomposes into an unknown part $\vec{c}_{i,u}$ and a known part $\vec{c}_{i,k}$ as in Chap. 5. The Markov chain sampling techniques we discuss later in this chapter apply similarly for this case except that the curve perturbations must be based on the 2D conditional simulation techniques in Chap. 5 rather than the regular perturbations in Chap. 3.

■ 6.2 Single-slice Gap

As noted previously in Sec. 6.1.2, when some of the slices are known and assumed to be correct, the sampling process is split into a number of independent smaller sampling problems. We begin by examining a case where a given curve is unknown but its neighbors above and below are both known. This is illustrated in Fig. 6.3 where $\vec{c}_2$ is the unknown curve and $\vec{c}_1$ and $\vec{c}_3$ are both known.

More generally, let $\vec{c}_k$ be the unknown slice. Examining the factorization in (6.1) of the target distribution into edge and self potential functions, we see that the target distribution for $\pi_k(\vec{c}_k) = \mathrm{p}(\vec{c}_k \,|\, \vec{c}_{k-1}, \vec{c}_{k+1}, y_k)$ (which is only a distribution over $\vec{c}_k$) can

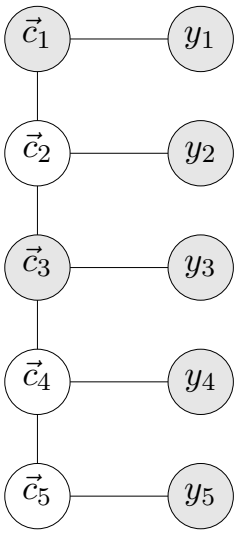


Figure 6.3. Hybrid 2D/3D Markov chain model with a single-slice gap. The y_i are observations and $\vec{c}_i$ are curves in each slice. The curves $\vec{c}_1$ and $\vec{c}_3$ are shaded to indicate they are observed.

be written as:

$$\pi_k(\vec{c}_k) \propto \Phi_k(\vec{c}_k)\Phi_{k-1,i}(\vec{c}_{k-1},\vec{c}_k)\Phi_{k,k+1}(\vec{c}_k,\vec{c}_{k+1})\Lambda_k(\vec{c}_k,y_k) \ . \tag{6.9}$$

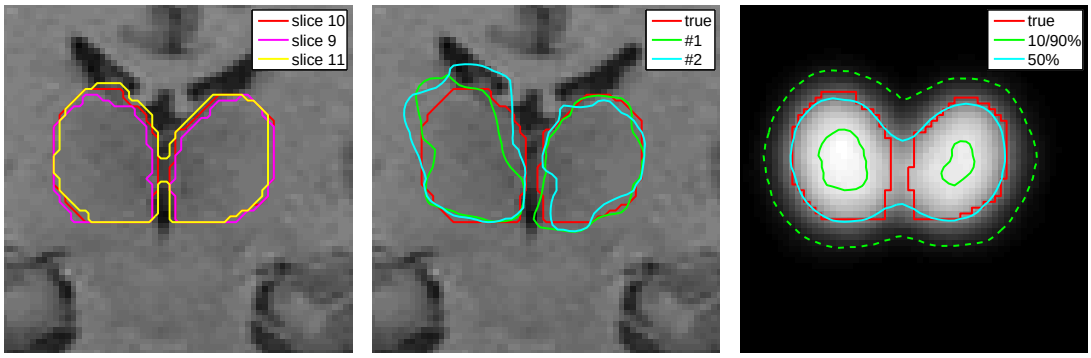
We see that the target distribution takes on a form similar to our original 2D curve sampling formulation. The self potential Φ_i is analogous to a regularization term, and Λ_i is similar to a data likelihood term. The major difference is the addition of the edge potentials $\Phi_{i-1,i}$ and $\Phi_{i,i+1}$. This then looks like a model with a shape prior except the prior information comes from neighboring slices rather than segmentations of other exemplars. For this problem, we can then use our standard 2D curve perturbation approach using the target distribution specified in (6.9).

We apply this approach to the same thalamus slice used previously in Sec. 5.2. We define the data potential function Λ_i the same as the model in (5.11). The self and edge potentials are:

$$\Phi_i = \exp(-\alpha \oint_{\vec{c}_i} \mathrm{d}s) \tag{6.10}$$

$$\Phi_{i,i+1} = \exp(-\xi \mathrm{d}_{\mathrm{SAD}}(\vec{c}_i,\vec{c}_{i+1})) \ . \tag{6.11}$$

In Fig. 6.4, we show the results of applying this single-slice sampling approach. As can be seen in Fig. 6.4(a), the curves in slices 9 through 11 are very similar. This



(a) (b) (c)

Figure 6.4. (a) Image on slice 10 of thalamus volume with true $\vec{c}_9$, $\vec{c}_{10}$, and $\vec{c}_{11}$ superimposed. (b) Most probable samples and (c) histogram image with confidence bounds for single-slice 2D/3D Markov chain sampler.

means that the neighboring slices strongly constrain the range of possible samples. Not surprisingly, this leads to most-probable samples in Fig. 6.4(b) which are quite accurate. The confidence bounds in Fig. 6.4(c) fully enclose the true curve location, and the median contour is quite accurate on the portions of the curves away from the ventricles (the dark objects directly above the thalamus and between the two halves of the thalamus). Conversely, the median contour does not separate the two halves of the thalamus. As in some examples in Chap. 5, this is simply due to the fact that we are using marginal statistics of a complex, high-dimensional random variable. If we examine all of the samples generated by the algorithm, we see that a very small proportion of them (less than 1%) result in curves which overlap in the middle.

This example is relatively simple because the neighboring slices provide so much additional information for the sampling problem. In problems where there is more slice-to-slice variability (*e.g.*, a volume which is not as finely spaced in the z-direction), the single-slice problem could be more challenging. For this thalamus segmentation problem, the expert providing the segmentations may be doing too much work, and less-dense slice information may be sufficient to adequately segment the thalamus. We examine this problem in the next section.

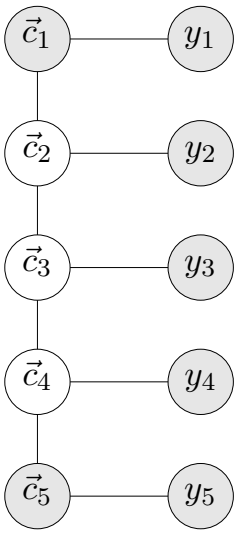


Figure 6.5. Model of 3D volume using 2D slices connected as a Markov chain. The y_i are observations and $\vec{c}_i$ are curves in each slice. $\vec{c}_1$ and $\vec{c}_5$ are shaded to indicate they are observed.

■ 6.3 Multiple Unknown Slices

When multiple contiguous slices are unknown, we must attempt to sample from the joint probability distribution over all of the unknown slices. An example of a model for this type of problem is shown in Fig. 6.5 where the top and bottom slices are fixed, but the middle three slices are unknown. Let $\mathcal{S}_u$ be the unknown slices and $\mathcal{S}_k$ be the known slices. The structure of the probability distribution for this model suggests three possible approaches:

1. A global Metropolis-Hastings approach which perturbs the entire $\mathcal{S}_u$ and accepts or rejects the resulting Γ_u.

2. A local Metropolis-Hastings approach where we only perturb a subset of $\mathcal{S}_u$ (typically just one slice) and accept or reject the resulting Γ_u based on the local perturbation.

3. An implementation of the embedded HMM algorithm of Neal *et al.* (see Sec. 2.1.3).

The global Metropolis-Hastings approach may be advantageous when one is able to create a good global perturbation function (*i.e.*, one that incorporates a strong inter-slice model) that can generate candidate samples in high probability areas of π. If

such a proposal distribution is not available, these global approaches can often result in slow convergence. Consider a scenario where the overall proposal distribution $\mathtt{q}(\Gamma_u \,|\, \mathcal{S}_u)$ consists of independent perturbations for each $\vec{c}_i$. In this case, unless the acceptance rate of the algorithm is extremely high, these methods will throw away many samples which improve some of the slices but also worsen others. In contrast, a local Metropolis-Hastings approach can accept or reject these slice candidates individually.

For this reason, we focus on local Metropolis-Hastings and embedded HMM approaches in this section. Each of these methods also has its advantages and disadvantages. Both of them generate candidate curves in a slice-based fashion with our basic 2D curve perturbation method at the core. The local Metropolis-Hastings method is simpler to implement because it essentially implements a sequence of 2D Metropolis-Hastings steps. For this same reason, it may be more difficult for a local Metropolis-Hastings approach to make large global changes in the state as the perturbations and accept/reject decision are confined to a given slice and its neighbors.

■ 6.3.1 Local Metropolis-Hastings Approach

A true Gibbs sampling algorithm generates samples from the local conditional probability $\pi_k(\vec{c}_k \,|\, \vec{c}_{k-1}, \vec{c}_{k+1}, y_k)$. Note that this distribution is the same as the one examined for the single-slice case in (6.9), and it has a similar decomposition into potential functions as before. We do not know how to sample from this distribution in closed form, but, as noted in Sec. 2.1.2, it is sufficient to sample from a chain whose stationary distribution is $\pi_k(\vec{c}_k \,|\, \vec{c}_{k-1}, \vec{c}_{k+1}, y_k)$ to maintain detailed balance with respect to the full target distribution $\pi(\mathcal{S} \,|\, Y)$. This can be thought of as a Metropolis-Hastings sampler with local proposal distributions.

Therefore we can construct an overall local Metropolis-Hastings sampling algorithm as follows. Let $k^{(t)}$ be the index of the active slice at a given time t. We do L steps of a 2D Metropolis-Hastings sampling algorithm where the target distribution $\pi(\vec{c}_{k^{(t)}} \,|\, \vec{c}^{\,(t-1)}_{k^{(t)}-1}, \vec{c}^{\,(t-1)}_{k^{(t)}+1}, y_{k^{(t)}})$ is defined similarly to (6.9), and the proposal distribution is the same as defined in Chap. 3. At the end, we generate a new index $k^{(t+1)}$ and move on to the next iteration.

The next index can be chosen in a random or deterministic fashion. One example of a random selection would be to simply choose the next $k^{(t+1)}$ from $\{1, \ldots, N\}$ uniformly. This is in fact the choice we make in the experiments in this section. A deterministic schedule is to simply increment $k^{(t)}$. In some cases, it may be possible to design a

deterministic schedule that can improve on the convergence rate of a random schedule. For instance, in the conditional simulation problems, we would expect there to be less variance in the slices which are closer to the known slices. Thus it may be advantageous to concentrate early samples on samples from those slices, or it may be possible to simply use fewer iterations on those slices.

Note that incorporating conditional simulation (where one or more slices may be known) does not significantly alter the sampling procedure. The computations involved in the local Metropolis-Hastings sampler are restricted to a single slice and use probability distributions conditioned on the current values of its neighbors. Thus the only change that needs to occur is in the sample scheduling (*i.e.*, we need not ever visit the known slices).

■ 6.3.2 Embedded HMM

To implement an embedded HMM algorithm, we must specify the constellation sampling probability functions ρ_i (see Sec. 2.1.3). Additionally, because we are unable to sample from ρ_i directly and instead wish to sample from a Markov chain $\mathcal{M}_i$ whose stationary distribution is ρ_i (see page 39 in Sec. 2.1.3), we must specify the transition probability T_i of each sub-chain $\mathcal{M}_i$. Note that in order for the embedded HMM method to satisfy detailed balance with respect to π, ρ_i cannot be dependent on the current values of the slices (*e.g.*, $\rho_i(\vec{\gamma}_i \,|\, \vec{c}_i) = \rho_i(\vec{\gamma}_i)$). Once we specify how to construct the constellation, we can form the embedded HMM and sample from it according to the algorithm of Neal *et al.* (see Sec. 2.1.3 and Appendix C for a discussion of the forward-backward algorithm for undirected chain models).

For simplicity, we choose to make each ρ_i a uniform distribution[2], and the transition probability is chosen to be a Metropolis-Hastings accept/reject step where the candidate sample is generated from our standard curve proposal distribution (from which we describe how to sample from in Sec. 3.2 and how to evaluate in Sec. 3.3). Let $\mathtt{q}_i(\vec{\gamma}_i \,|\, \vec{c}_i)$ be the proposal distribution for a slice i, and τ index the Markov chain with stationary

[2]More complicated constellation probabilities can be used if desired. For instance, one choice would be to use the self and data potential functions:

$$\rho_i(\vec{c}_i) = \Phi_i(\vec{c}_i)\Lambda_i(\vec{c}_i, y_i) \ . \tag{6.12}$$

Using a uniform distribution is simpler computationally and, from our experiments, appears to be effective.

distribution ρ_i used to generate the constellation points. Then a candidate sample $\vec{\gamma}_i^{(t,\tau)}$ is sampled from $\mathsf{q}(\cdot \mid \vec{\gamma}_i^{(t,\tau-1)})$, and the constellation point $\vec{c}_i^{(t,\tau)}$ is filled by accepting or rejecting $\vec{\gamma}_i^{(t,\tau)}$ with probability:

$$\eta(\vec{\gamma}_i^{(t,\tau)} \mid \vec{\gamma}_i^{(t,\tau-1)}) = \frac{\mathsf{q}(\vec{\gamma}_i^{(t,\tau-1)} \mid \vec{\gamma}_i^{(t,\tau)})}{\mathsf{q}(\vec{\gamma}_i^{(t,\tau)} \mid \vec{\gamma}_i^{(t,\tau-1)})} \ . \tag{6.13}$$

Note that ρ_i does not appear here because it is identical for all $\vec{\gamma}_i$.

■ 6.3.3 Results for Multiple Contiguous Unknown Slices

In this section, we investigate the application of the local Metropolis-Hastings and embedded HMM sampling algorithms for the case when two known slices are separated by multiple unknown slices. The potential functions for the target distribution are defined exactly as in Sec. 6.2, and the overall distribution is written in terms of the potential functions as in (6.1).

In the example in Fig. 6.6, slices 12 and 17 are provided by an expert, and we sample from the intermediate slices conditioned on the known slices. The initialization for each algorithm is a pair of small circles located within the thalamus for each unknown slice[3]. The local Metropolis-Hastings approach is run for 3000 iterations per unknown slice (or 12,000 total iterations), and the embedded HMM algorithm is run for 600 iterations with a constellation size of 4.

We display the output from our local Metropolis-Hastings sampling algorithm in rows (a) and (b) of Fig. 6.6 and from our embedded HMM sampling algorithm in rows (c) and (d). Within each row, we show the most probable samples or histogram images for each slice with the slice number indicated at the top of each column. We can see that even when allowing a gap of 4 contiguous unknown slices, both the local Metropolis-Hastings and embedded HMM approaches result in most probable samples and median contours which are generally quite close to the expert segmentation and 10% and 90% confidence bounds that enclose most of the expert segmentation as well. Note that slices 13 and 16 generally appear to be more accurate and have less variance (in terms of the diffuseness of the histogram image and the distance between the 10% and 90% confidence bounds) than slices 14 and 15 (the two middle columns in Fig. 6.6).

[3]Given the expert-segmented slices, we could use those to generate better initializations for the unknown slices by interpolating between the curves from the user-specified slices. We use a non-expert-driven version here to show robustness to poor initializations.

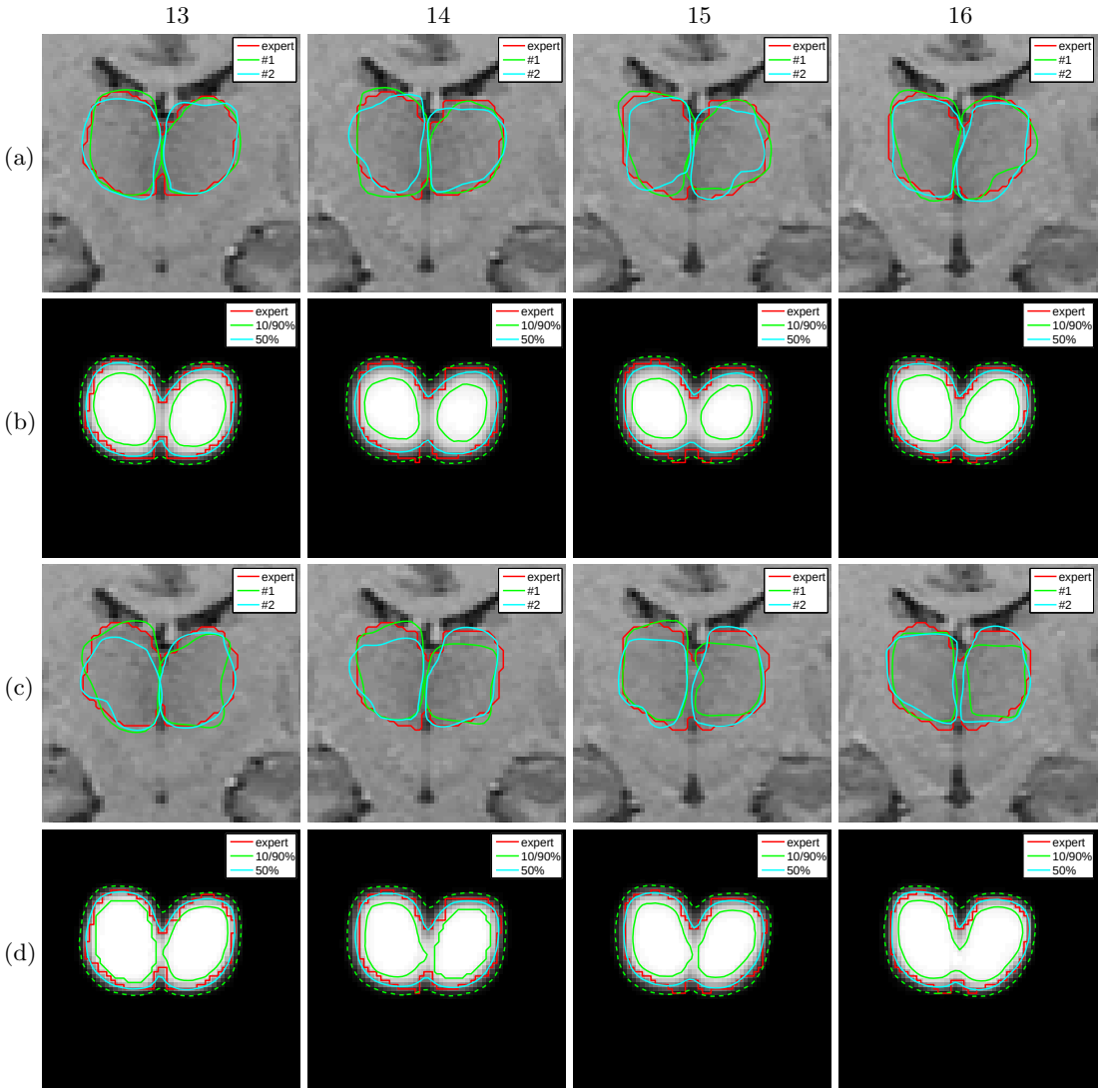


Figure 6.6. Results using the local Metropolis-Hastings (rows (a) and (b)) and embedded HMM (rows (c) and (d)) methods. Slices 12 and 17 are fixed, and slices 13-16 are conditionally simulated given those constraints. We show the two most probable samples for each slice in rows (a) and (c), and histogram images and marginal confidence bounds in rows (b) and (d). The top of each column is labeled with the slice index from which the images come.

This conforms with our expectations as slices 13 and 16 are directly adjacent to expert-segmented slices whereas slices 14 and 15 must rely solely on edge potentials connected to slices from which we are also sampling.

Overall the results from the local Metropolis-Hastings sampler appear similar to those from the embedded HMM algorithm. This is to be expected as both methods are generating samples from the same target distribution π. Minor discrepancies do appear such as the 10% confidence bounds (the solid green lines) in slice 16. For the local Metropolis-Hastings method, the 10% confidence bounds form two disjoint regions while they form one region with a narrow connection for the embedded HMM method. The differences are relatively minor for this example and subsequent ones we show in Sec. 6.3.4, and they could arise from the fact that we are implementing the continuous curve sampling formulation with a discrete approximation.

In terms of computation, anecdotally we observe that the embedded HMM approach converges more rapidly than the local Metropolis-Hastings approach, but the difference is slight for this example. The target probability distribution for this problem appears to be unimodal, so the ability of the embedded HMM algorithm to do more global perturbations to move between modes does not benefit this example as much as it could other examples. We would expect the embedded HMM approach to provide the largest gains when there is strong slice coupling[4] and multiple modes between which we need to be able to transition.

A constellation size of 4 was used in these experiments because choosing larger values increased the computation time per sample without appreciably increasing the mixing rate. Because our constellation curves are generated from a chain with a uniform stationary distribution, we are largely accepting or rejecting the candidate curves randomly and not according to whether a given curve is good in terms of π. Thus after generating a few perturbations along the chain, there is a reasonable likelihood that the constellation curves being generated have low probability under π and thus have little chance of being selected by the embedded HMM algorithm.

■ 6.3.4 Varying the Gap Size

In Sec. 6.2 and Sec. 6.3.3, we demonstrated our sampling framework on examples with 1 or 4 contiguous unknown slices, respectively. Naturally there is more uncertainty and

[4]If the slices are strongly coupled, when we have a curve $\vec{c}_i$ which largely agrees with its neighbors $\vec{c}_{i-1}$ and $\vec{c}_{i+1}$, we are unlikely to accept a perturbation which moves $\vec{c}_i$ away from its neighbors.

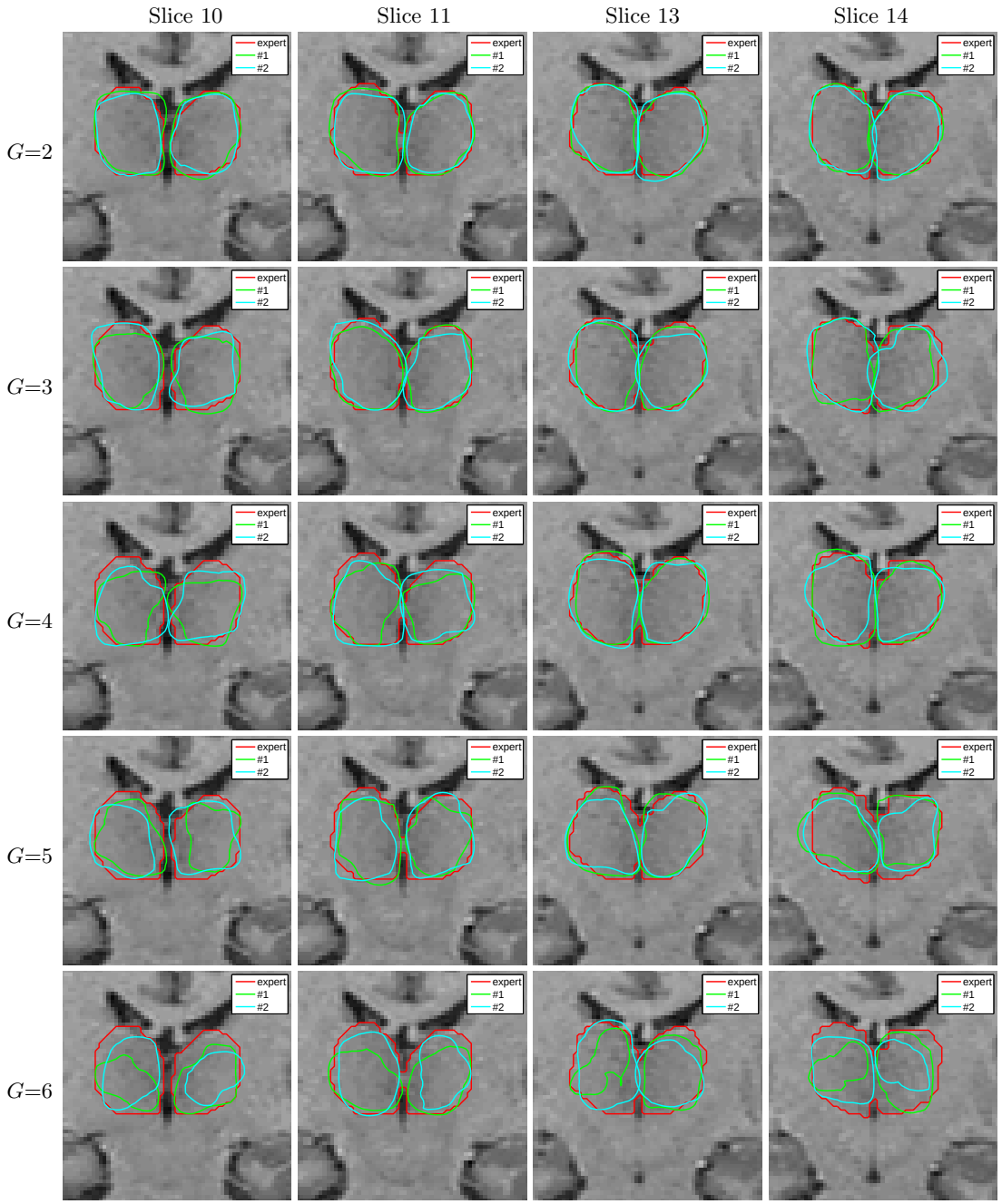


Figure 6.7. Most probable samples from the local Metropolis-Hastings approach applied to slices 5-19 of a thalamus volume. We vary the size of the gap G between known slices in each row and indicate in the left margin the gap size. We show slices 10, 11, 13, and 14 for each example. Slices 11 and 13 are always next to a known slice (slice 12). Except in the first row when $G = 2$, slices 10 and 14 are not next to a known slice.

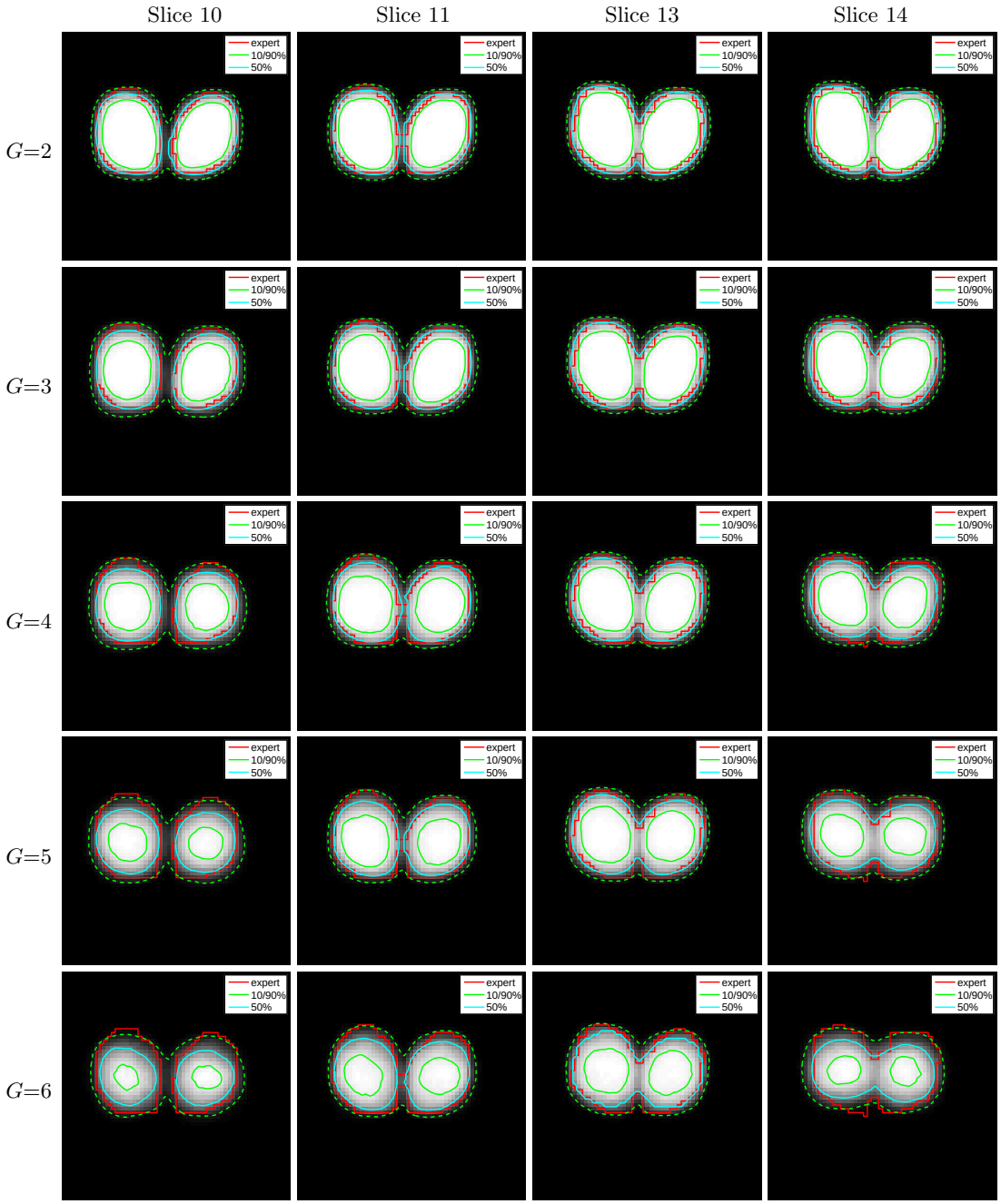


Figure 6.8. Histogram images and marginal confidence bounds from the local Metropolis-Hastings approach applied to slices 5-19 of a thalamus volume. We vary the size of the gap G between known slices in each row and indicate in the left margin the gap size. We show slices 10, 11, 13, and 14 for each example. Slices 11 and 13 are always next to a known slice (slice 12). Except in the first row when $G = 2$, slices 10 and 14 are not next to a known slice.

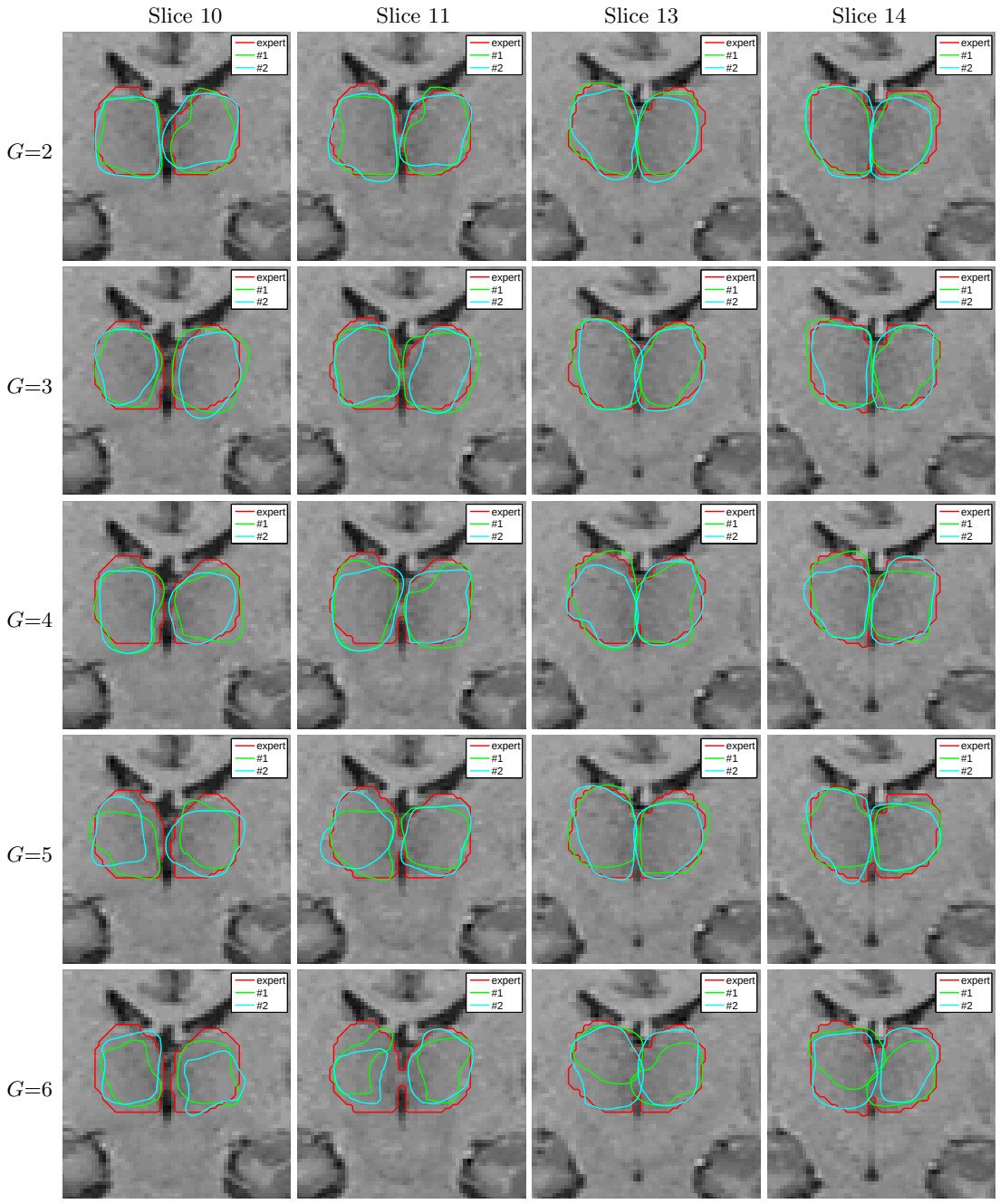


Figure 6.9. Most probable samples for the embedded HMM approach applied to slices 5-19 of a thalamus volume. We vary the size of the gap G between known slices in each row and indicate in the left margin the gap size. We show slices 10, 11, 13, and 14 for each example. Slices 11 and 13 are always next to a known slice (slice 12). Except in the first row when $G = 2$, slices 10 and 14 are not next to a known slice.

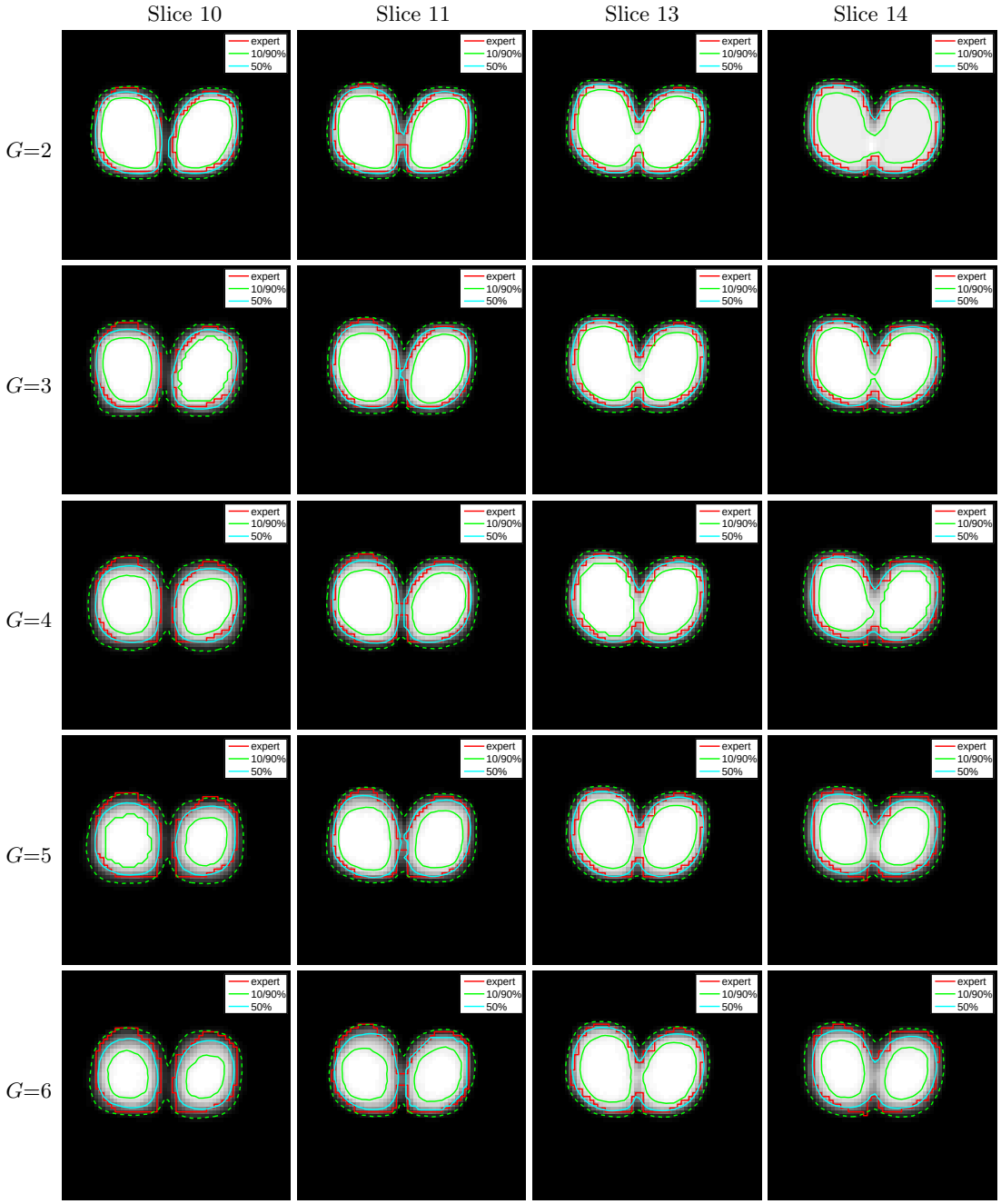


Figure 6.10. Histogram images and marginal confidence bounds for the embedded HMM approach applied to slices 5-19 of a thalamus volume. We vary the size of the gap G between known slices in each row and indicate in the left margin the gap size. We show slices 10, 11, 13, and 14 for each example. Slices 11 and 13 are always next to a known slice (slice 12). Except in the first row when $G = 2$, slices 10 and 14 are not next to a known slice.

larger error when there are more unknown slices. In this section, we investigate in a more detailed manner the relationship between the size of the gap between known slices and the behavior of the resulting samples.

We begin by presenting visual results for slices 5 to 19 of the thalamus volume where we vary the gap size G using values ranging from 2 to 6. For these examples, the G slices directly above and below slice 12 are unknown (*i.e.*, slices $12-G$ to 11 and slices 13 to $12+G$), and the remaining slice segmentations are provided by an expert (*i.e.*, slices $11-G$, 12, and $13+G$). In Figs. 6.7 and 6.8, we show the most probable samples and histogram images for the local Metropolis-Hastings approach, and in Figs. 6.9 and 6.10, we display the same images for the embedded HMM method. We only show a subset of the slices due to space considerations. As indicated on the figures, slices 11 and 13 are in the middle two columns, and slices 10 and 14 are in the left- and right-most columns. Slices 11 and 13 are neighbors of a known slice (slice 12) for all gap sizes. For a gap size of 2, all unknown slices must be neighbors of a known slice. For gap sizes greater than 2, slices 10 and 14 only have other unknown slices as neighbors.

We can see that as the gap size increases, the quality of the segmentations provided by the samples gradually decreases: the most probable samples and median contours become less like the true segmentations for $G = 6$ compared with $G = 2$ or 3. Additionally, the amount of uncertainty increases as well. This is evident from the increase in spacing between the confidence intervals.

As we would expect, qualitative inspection of the sampling visualizations confirms the intuition that larger gaps between known slices results in more uncertainty and less accurate segmentations. We can also develop a number of quantitative measures to evaluate our sampling algorithm:

1. Symmetric area difference (SAD) between the median contour and the expert contour. This measure tabulates the number of pixels which have differing segmentation labels and can be seen to be related to a probability of error on a pixel-by-pixel basis.

2. Dice measure [86] between the median contour and the expert contour. This measure is similar to SAD, except it weights correct labels twice as much as incorrect labels.

3. L2 distance between the histogram image and the binary 0/1 expert label map. Again, this is similar to SAD, except that the histogram image provides a soft

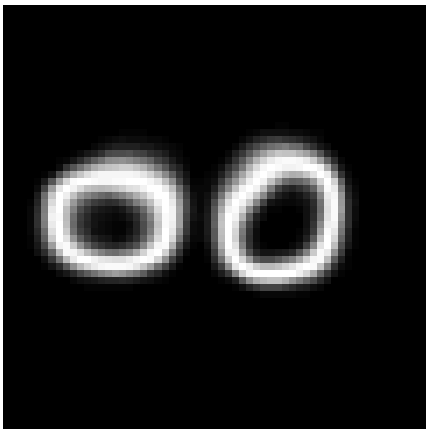


Figure 6.11. Variance image of a slice of the thalamus generated by the local Metropolis-Hastings approach.

decision label at each pixel.

4. Total image variance. The histogram image $\Phi(\boldsymbol{x})$ provides an estimate of the marginal label probability for each pixel $\boldsymbol{x}$. We can convert this estimated probability to an estimated variance (using the standard result for a Bernoulli random variable) as $\sigma^2(\boldsymbol{x}) = \Phi(\boldsymbol{x})(1-\Phi(\boldsymbol{x}))$. Note that unlike the previous three methods, this approach does not require either an expert or true segmentation boundary.

   By computing this variance for each pixel, we can obtain an image such as the one shown in Fig. 6.11. The variance image is brightest in areas where we are most uncertain about the correct segmentation label. If we sum this variance image over all pixels for a given slice, we obtain a measure of the total amount of variability. In the extreme case where every sample results in the same segmentation, the variance image would simply be 0 everywhere so the sum of the variance would be 0 as well.

In general, these different measures provide similar information about segmentation error and uncertainty for this thalamus example, so we focus here solely on the results using the L2 error measure. In Fig. 6.12, we show the L2 error on a slice-by-slice basis for the case when the gap size is 6 (slices 5, 12, and 19 are known). We can see that the error increases the farther a given slice is from the known slices. This confirms the qualitative observations made previously.

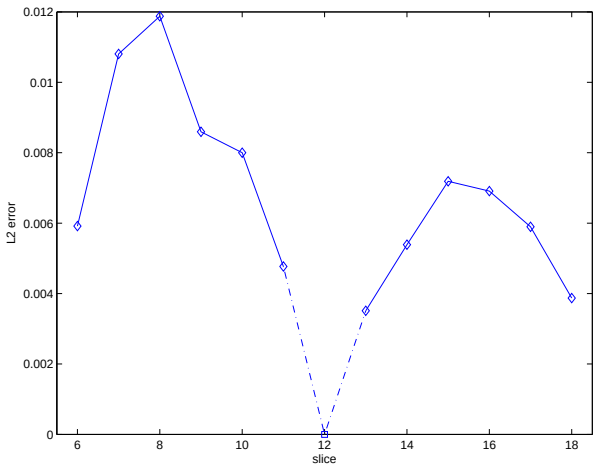


Figure 6.12. Plot of the L2 error for each slice using the L2 distance between the histogram image and the the binary expert segmentation. The gap size in this example is 6 on both sides of slice 12.

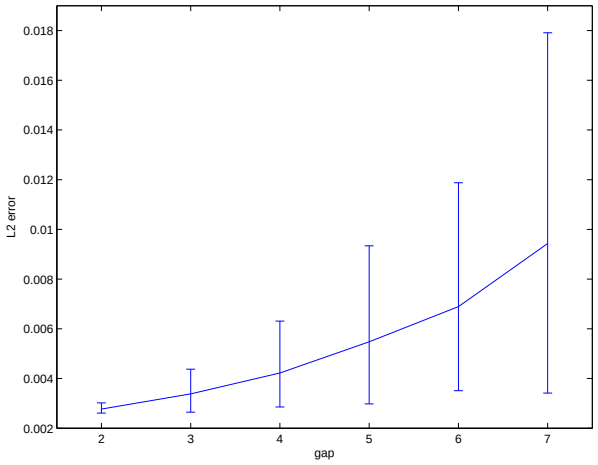


Figure 6.13. Plot of the L2 error as a function of the gap size G. For each value of G, we compute the L2 error $\mathcal{E}_{G,i}$ for each slice i (as in Fig. 6.12) and then calculate the minimum, mean, and maximum values over all of the slices. We plot the mean values with error bars that represent the range of L2 errors (minimum to maximum) for each gap size G.

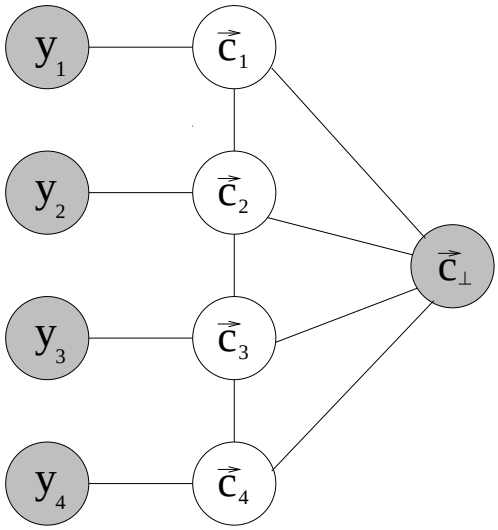


Figure 6.14. Markov chain model with perpendicular slice constraint.

We can also use the L2 error to measure the performance as a function of the gap size G. In Fig. 6.13, we calculate aggregate statistics of the slice-by-slice L2 errors for different values of G. We can see that the average error per slice increases in a superlinear fashion as does the maximum error. The minimum error generally increases as well, but to a much smaller extent. The reason for this is that, as we increase the gap size, some slices become quite far from known slices (*e.g.*, for $G = 2$, each unknown slice is no more than 1 slice away from a known one while for $G = 7$, the middle slice is 4 slices away from a known one), but there are always slices which are directly adjacent to known slices. While those slices do become more difficult to segment with increasing G (because the known information on the other side is farther away), the influence is much weaker due to the neighboring known slice which acts as an anchor.

■ 6.4 Orthogonal Slice Information

In the preceding section, we considered examples in which all of the slices in the Markov model were in parallel planes. An extension is to introduce the ability to have slices oriented in orthogonal directions as well. This type of model is illustrated in Fig. 6.14 where most of the curve variables are parallel to each other (which we will refer to as being in the primary orientation), but an additional perpendicular slice $\vec{c}_\perp$ has been

included (which we term the orthogonal or secondary plane).

In general, having these orthogonal curves means that the model for the surface is over-specified: multiple curves contain information about the same location on the surface. This can cause conflicting information which needs to be resolved. Here we avoid this situation by assuming that information provided by the expert user is self consistent (*i.e.*, a segmentation in the primary plane and another in the orthogonal plane must not disagree on any pixel label), and the only orthogonal slice variables introduced into the model have exactly known segmentations.

By treating the orthogonal curves as being exact, this fixes certain locations on each of the curves with the primary orientation. If we imagine a primary orientation in the x-y plane and an orthogonal curve in the y-z plane, the secondary plane intersects the primary plane along a line in the y-direction. The locations where an orthogonal curve intersects the plane in slice k are then locations through which the primary curve $\vec{c}_k$ should pass as well[5]. This type of information is exactly of the same form as that used in the 2D conditional simulation approaches explored in Chap. 5 in which certain parts of the curve were specified by an expert user.

For the model in Fig. 6.14, having an observed perpendicular slice does not change the overall chain structure. Thus we can apply the same multi-slice surface perturbation methods described in Sec. 6.3, except we perform 2D conditional simulation for each slice using the orthogonal slice information. As in Chap. 5, this simply requires a modification of the proposal distribution that respects the constraints.

In medical imaging, three standard slice orientations are used. Relative to a standing person, *axial* slices are horizontal, *sagittal* slices are vertical and divide the body into left and right sections, and *coronal* slices are also vertical and divide the body into front and back sections. The primary slices used previously in, *e.g.*, Fig. 6.6, are axial slices. In Fig. 6.15 we show two sagittal slices of the same thalamus volume used previously[6] One slice is located in the left thalamus and the other in the right thalamus. An expert

[5]In fact, there is additional information provided by the orthogonal curve: locations along the intersecting line in the y-direction where the curve is *not* located are places where the primary curve should also not be. We do not incorporate this kind of information into our model.

[6]In this example, as is typical for many cases, slices from a single 3D volume are used for the different orientations. Some medical imaging applications have separate scans taken in the axial, coronal, and sagittal directions due to an inability to obtain sufficient inter-slice resolution. For this type of problem, a registration step may need to be performed as a preprocessing step to align the different slice orientations.

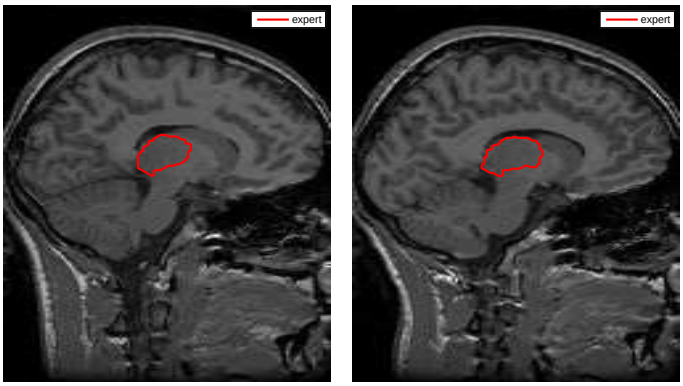


Figure 6.15. Sagittal thalamus MR image with expert segmentation for two different slices.

segmentation of the thalamus boundary has been overlaid for each slice. We show the results using our orthogonal 2D/3D algorithm to generate samples using gap sizes of 7 and 8 in Fig. 6.16. Slices 4, 12, and 21 and the two orthogonal curves are provided to the algorithm, and the intermediate slices are conditionally simulated. We again only show a representative subset of the unknown slices. We can see that even though the gap sizes used here are larger than any considered in Sec. 6.3.4, the results are quite accurate overall (as can be seen in comparison to, *e.g.*, Figs. 6.7 and 6.8 for gap sizes of 5 or 6). The expert curve boundaries are close to the most probable curves, and they are generally bracketed by the 10% and 90% confidence bounds.

We can compute the average L2 error between the histogram image and the binary expert segmentations as we did before in Sec. 6.3.4. The value of that error for this particular example is 0.0048. We can see from Fig. 6.13 that this value falls between the average error for gap sizes of 4 or 5. Thus we can see that to segment slices 4 to 21 of this volume, we can obtain similar segmentation error by using 3 primary segmentations and 2 orthogonal segmentations (as in the example here) or 4 to 5 primary segmentations. Evaluating the amount of work an expert must expend to provide the segmented slices is non-trivial, and it is complicated in this example by the fact that the thalamus appears as two disjoint objects in the primary (axial) slices and one single object in the secondary (sagittal) slices. In general, though, we can see that for a given amount of work for the expert user, there is a trade-off (in terms of segmentation accuracy) between information provided on the primary slices versus information given on the secondary slices.

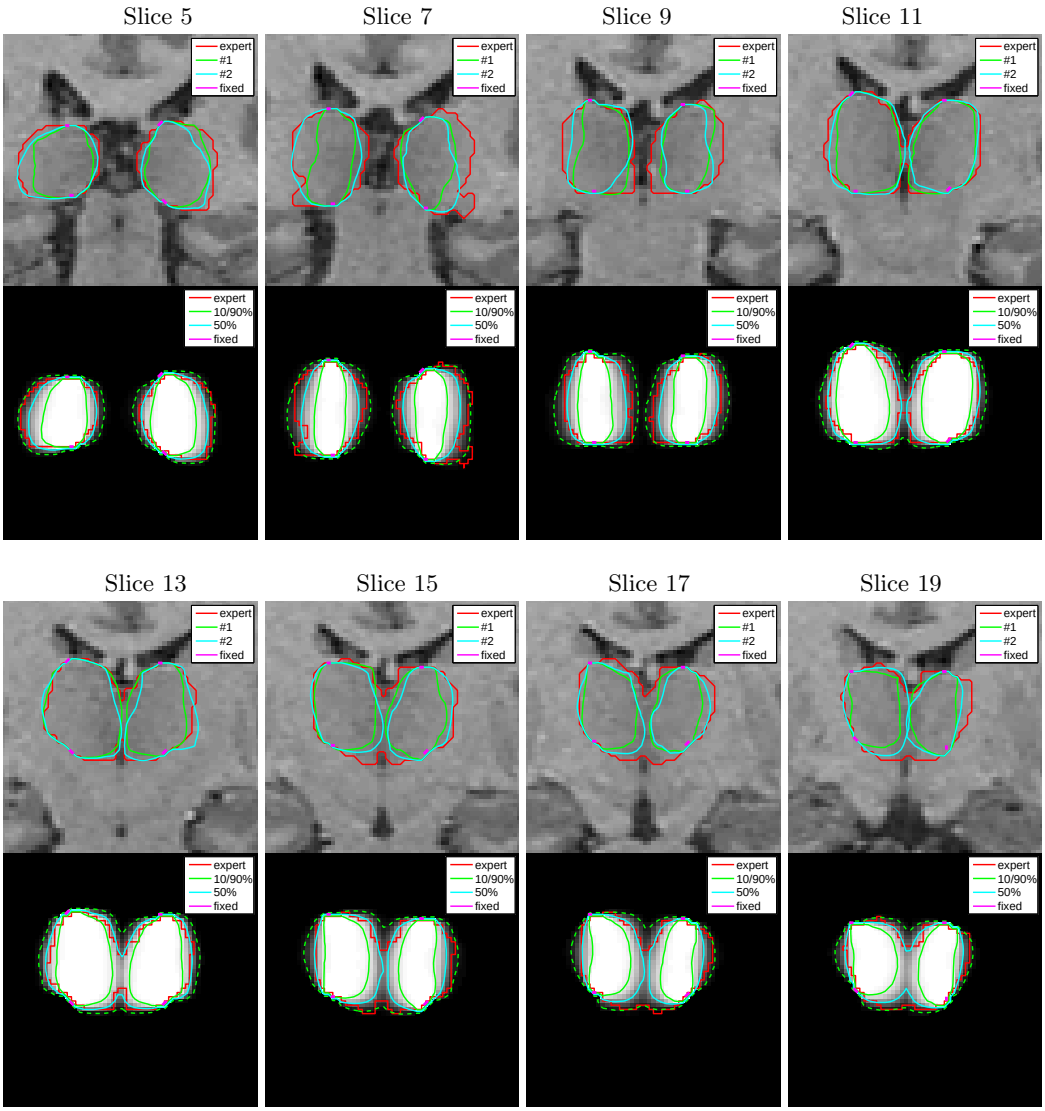


Figure 6.16. Most probable samples, histogram images, and marginal confidence bounds for the case when two sagittal slice segmentations are used to conditionally simulate axial slices. Slices 4, 12, and 21 in the primary plane are also specified.

Chapter 7

Conclusions and Suggestions for Future Research

IN this thesis, we have presented a framework for drawing samples from probability distributions over the space of curves and have developed methods to address a number of application areas including medical imaging and geophysical inverse problems. We summarize the major contributions and results in Sec. 7.1 and conclude in Sec. 7.2 with suggestions for future research avenues based on this work.

7.1 Thesis Contributions

The major contribution of this thesis involved developing a general method of sampling curves using iterative MCMC methods for image segmentation applications. This approach is quite different from traditional curve-based approaches which use optimization techniques to segment images. We demonstrated how to display information about a large set of samples in a visually intuitive fashion and showed the advantages this provided for applications which suffer from low SNR, poor contrast, or ill-posedness. We showed how to extend the algorithm to perform conditional simulation (which allows semi-automatic segmentation) and segment 3D surfaces using a slice-based Markov chain representation.

Curve sampling with detailed balance

In Chap. 3, we constructed a complete curve sampling framework based on the generic Metropolis-Hastings MCMC algorithm. To generate samples from a target distribution $\pi(\vec{C})$, MCMC methods construct a Markov chain whose stationary distribution is π. Simulating this chain then results in asymptotic convergence to π if the chain is ergodic

(*i.e.*, irreducible and aperiodic). Metropolis-Hastings algorithms construct this chain by defining a transition probability (conditioned on the previous iterate $\vec{C}$) that first samples a candidate curve $\vec{\Gamma}$ from a proposal distribution $\mathtt{q}(\vec{\Gamma} \,|\, \vec{C})$, and then accepts or rejects $\vec{\Gamma}$ with probability defined by the Hastings ratio (which only requires evaluation of the target and proposal distributions). Metropolis-Hastings methods thus transform the problem of sampling from π to that of generating many samples from $\mathtt{q}$.

We define a proposal distribution consisting of Gaussian perturbations f along the normal of the curve. In order to generate smooth curves, f has spatial correlation (implemented using circular convolution with a low-pass filter) and a mean force using negative curvature. The perturbations are made geometric by using an arc length parameterization and implemented using a hybrid of implicit narrowband level sets and explicit marker-point models.

We show that evaluating $\mathtt{q}(\vec{\Gamma} \,|\, \vec{C})$ is well-approximated by evaluating the probability of a discretization of our continuous perturbation f. Computing the reverse proposal distribution $\mathtt{q}(\vec{C} \,|\, \vec{\Gamma})$ first requires construction of ϕ, the reverse perturbation which generates $\vec{C}$ from $\vec{\Gamma}$. Given ϕ, evaluation of $\mathtt{q}(\vec{C} \,|\, \vec{\Gamma})$ is then similar to the forward proposal distribution case. We describe three approaches to approximate the computation of ϕ in closed form. These methods form local approximate models of the curve $\vec{C}$ using lower-order derivatives. This then leads to an existence condition for ϕ which is exact for curves without higher-order derivatives (*e.g.*, torsion), and approximate for general curves.

Visualizing curve samples

Our sampling procedure can be used to generate hundreds or thousands of samples. In Chap. 4, we discussed a number of visualization techniques to concisely present the information contained in these samples. This included showing the samples with the highest probability under π; a histogram image of the marginal probability of each pixel being inside the curve; marginal confidence bounds that provide a notion of segmentation uncertainty; and PCA-based shape eigenvectors.

We demonstrated the advantages of our sampling method on a number of examples including shape models with single or multiple modes, a synthetic image with extremely low SNR, and a noisy prostate MR image. We also showed convergence properties of our algorithm. The confidence bounds allow us to see the areas of largest uncertainty as well as a range of reasonable locations for the true contour location. The randomness

in the sampling process allow our method to avoid being trapped in local optima. We showed examples in which the global mode of the probability distribution is actually not a good segmentation due to the specific realization of the noise in the image. This means that even a global optimizer would have produced sub-par segmentations, but our method is able to correctly bound the true location of the curve within confidence intervals. For multi-modal examples, we visualize the multiple modes using a simple clustering technique, whereas most optimization-based approaches would only be able to find the mode which was closest to the initialization.

Online parameter estimation

In Sec. 3.4, we developed methods to do online estimation of parameters of the target distribution $\pi(\vec{C};\, \theta)$. This is useful in many practical sampling problems in which the parameters θ cannot be known *a priori*. We created two general approaches to this problem, both of which involve modifying the target distribution $\pi(\vec{C};\, \theta)$. The first method results in a new distribution $\tilde{\pi}_{\mathrm{ML}\theta}$ which is simply $\pi(\vec{C};\, \theta)$ maximized with respect to θ. This approach can use the same proposal distribution defined previously. The second makes θ a random variable and creates an augmented distribution $\tilde{\pi}_{\mathrm{aug}}(\vec{C}, \theta)$ from which we draw samples. This method requires an augmented proposal distribution $\mathsf{q}(\vec{\Gamma}, \chi \,|\, \vec{C}, \theta)$ as well. We demonstrated these approaches on a synthetic noisy image in Sec. 4.4.

Conditional simulation

Conditional simulation is a technique that involves fixing part of the state space and sampling from the remaining variables. We show in Chap. 5 how our sampling approach can be naturally applied to this problem by using an unchanged target distributions π with a modified proposal distribution which preserves the fixed locations. This method leads to an interactive semi-automatic approach to solving challenging segmentation problems.

We demonstrated this approach first on a thalamus MR image. This is a difficult segmentation problem due to the limited contrast between the thalamus and the surrounding subcortical tissue. We make the problem tractable by using a portion of a radiologist's segmentation to specify small sections of of the curve and conditionally simulate the remainder. We also showed how the uncertainty estimates can be used by the user to determine which region of the curve needs the greatest additional assistance.

Another problem we address is that of gravity inversion. Here we wish to infer the location of underground salt bodies using gravity measurements taken at the surface of the earth. This is an ill-posed problem due to both the non-local nature of the measurements and having orders of magnitude fewer observations than points to reconstruct. We formulated a curve-based energy functional for this problem and derive its gradient flow. We demonstrated the results of both our sampling and optimization approaches on purely synthetic examples with both simple and complex geometrical structure, and an example using estimated real salt body boundaries extracted from a seismic image. The optimization-based approach has great difficulty finding the salt boundary in these examples, but our conditional simulation approach produces much better results and dramatically reduces the estimation variance compared with the unconstrained curve samples. PCA-based shape models are used to explore the principal modes of variation of the probability distribution.

Hybrid 2D/3D Surface Model

In Chap. 6, we constructed a hybrid 2D/3D surface model to sample 3D surfaces. Here the overall probability distribution is specified as an undirected Markov model in which the nodes represent entire curves on parallel planes. The curves combine together to form an overall surface. Undirected Markov models have probability distributions which can be written in terms of local potential functions, and we show how symmetric area difference is a natural coupling term to use. The Markov model also leads to conditional simulation approaches in which entire slices are specified instead of just curve segments.

We demonstrated sampling implementations using this model on a thalamus MR volume. For the case when the curves directly above and below a given curve are fixed, we can use our standard 2D curve sampler with a modified target distribution to incorporate terms which couple the slice with its neighbors. When multiple contiguous slices are unknown, we sample from the hybrid 2D/3D probability distributions using two approaches: a local Metropolis-Hastings method and a method based on the embedded HMM approach of Neal *et al.* [77]. The local Metropolis-Hastings approach uses proposal distributions which perturb only one slice at a time (conditioned on the current values of the other slices). The embedded HMM method independently generates a constellation of candidate samples for each slice, but selects the next iterate value in a global fashion by evaluating complete sets of constellation points over the full joint target probability distribution over all slices. Conceptually, the local Metropolis-Hastings

approach is simpler than the embedded HMM algorithm, but it may have more difficulty generating large coordinated moves involving multiple slices. We also developed an extension which uses curve information on planes orthogonal to the primary slices of interest. This approach allows an expert to provide constraints on all of the slices by only segmenting a small number of slices.

■ 7.2 Suggestions for Future Research

The framework we have constructed for curve sampling can be used to target a variety of challenging segmentation and shape reconstruction problems. A number of other problems with large uncertainty about the correct solution could substantially benefit from using our curve sampling approach. This includes ultrasound imaging of the heart [103, 107] and prostate [1], brain tumor detection and segmentation in MR images [33], natural images [118], and joint inversion using seismic and gravity data for geophysical applications [2, 98, 108].

Many choices were made at various stages of the implementation of this algorithm. In the remainder of this section, we describe future research topics which branch out from our basic framework. The approaches we describe in Sec. 7.2.1 involve improving the basic curve sampling engine to increase the speed of convergence. In Sec. 7.2.2, we consider curve models which can further exploit the power of our sampling methods. We conclude in Sec. 7.2.3 by discussing some possible techniques for visualizing the output of our sampling algorithm and the structure of the target distribution π.

■ 7.2.1 Sampling Advances

Sampling is naturally more computationally intensive than optimization-based methods. Each sample takes a random walk through the configuration space, and many samples must be generated to adequately represent that space. Current implementations require 15-180 seconds per sample for 2D examples and a multiple of that (depending on the number of slices) for 3D examples. Reducing the computation time needed would be especially useful for semi-automatic approaches in which the user is actively interacting with the segmentation algorithm. Better numerical implementations or proposal distributions which more efficiently generate desired curves could reduce the time needed to generate each sample.

Numerical improvements

Our numerical implementation uses a mix of Matlab and C++ code. This causes computational inefficiency due to the need to convert data structures between the two. Additionally, Matlab is inherently slower than C++ for a number of tasks. Doing a full implementation in C++ could increase the rate at which we generate samples by a factor of 5 or 10. Another method to increase overall throughput is to develop an implementation using parallel computing. Many of the operations in the sampling procedure are independent of each other and lead to significant inherent parallelism which can lead to methods with sampling rates that scale linearly with the number of processors.

Multiresolution proposal distributions

Our sampling method currently uses the same proposal distribution throughout the sampling process. In many applications, coarse-scale features (*i.e.*, scale, location, orientation, and rough overall shape) are more important than finer-scale detail, and they also typically require more time to resolve. One method to help address this issue would be to first run an optimization algorithm to set the initial curve $\vec{C}^{(0)}$, but this may not be effective for multi-modal distributions or situations when optimization can only find poor local optima. A simple extension of this approach would be to design a version of the sampler which can only generate low-frequency perturbations (which tend to correspond to coarse-scale features). This method could rapidly find the individual modes, then the regular sampler could be used to generate the finer-scale detail. One worry with this approach is whether appropriate mixing of the chain would still occur, as the low-frequency version may converge to a stationary distribution which differs significantly from the stationary distribution for the our standard curve sampling implementation. This would likely result in incorrect weightings given to different modes.

Another approach would be to design perturbations which are explicitly multiresolution. These could be based on, *e.g.*, wavelets or Fourier representations. The main challenges for this approach are the need to maintain detailed balance (requiring the computation of both the forward perturbation $\mathtt{q}(\vec{\Gamma} \,|\, \vec{C})$ and the reverse perturbation $\mathtt{q}(\vec{C} \,|\, \vec{\Gamma})$) and the establishment of ergodicity. The latter could be especially difficult if the proposal distribution is modified with time.

Feature-generating proposal distributions

One area in which our current implementation has difficulty is generating regions of large positive or negative curvature. While the probability of our algorithm generating a complex object (*e.g.*, a hand) is non-zero (due to the ergodicity property we showed in Sec. 3.2.2), generating such a sample is such a low-probability event that the amount of time we may have to wait for it to occur can be impracticably large. This behavior is evident in some of our examples such as the difficulty generating the recumbency in the gravity inversion example in Sec. 5.3.3.

The reason this behavior occurs is that both our correlated Gaussian noise and the curvature term in the mean perturbation are designed to encourage fine-scale smoothness in the curve. This, unfortunately, leads to penalizing sharp features with large positive or negative curvature. One possible resolution of this problem could involve revising the proposal distribution to allow sharper features at a medium scale while maintaining smoothness at fine scales. Another could be to have a proposal distribution which explicitly generates likely object characteristics (*e.g.*, if we were doing segmentation of a hand, it would be useful to have a proposal distribution which generates fingers). Specifying such a distribution and maintaining detailed balance would both be challenging tasks for this method.

Topological change

Currently our sampling formulation does not allow for topological change due to the need to maintain correspondence between $\vec{C}$ and the candidate sample $\vec{\Gamma}$. In many situations this is acceptable because the number of regions is known *a priori*. For other cases such as gravity inversion or natural images, this may not be the case. Manual approaches to this could involve running the sampler using varying numbers of curves and examining the output retrospectively. A more automated approach could involve jump-diffusion methods [39] in which curves can be created or destroyed as part of the proposal distribution. Ensuring detailed balance for this approach could be difficult.

Full 3D formulation

Our hybrid 2D/3D model may not be ideal for surfaces with a great deal of curvature (which may cause topological changes in a 2D slice but not in the 3D surface) or when there is a large amount of uncertainty as to the location of the top and bottom of the

object (because the Markov chain structure for the hybrid model must be fixed *a priori*). As we discussed in Chap. 6, the major impediment to formulating a full 3D surface sampling algorithm is the need to develop a canonical surface parameterization. This is needed to ensure that the perturbations are geometric as ultimately the forward perturbation function $f : [0,1] \times [0,1] \to \mathbb{R}$ and reverse perturbation $\phi : [0,1] \times [0,1] \to \mathbb{R}$ are defined with respect to a specific parameterization. Note that for convex surfaces, many standard parameterizations exist (*e.g.*, a parameterization using spherical-coordinate angles). Work by Haker *et al.* [41] on conformal surface maps could be used to create parameterizations for an arbitrary surface by creating a mapping between that surface and a sphere (which is convex).

■ 7.2.2 Modeling Advances

Our framework of curve sampling opens up a number of avenues for new image segmentation models, either by allowing techniques that are infeasible with optimization methods or those which gain extra usability from having samples. Here we discuss several interesting possible approaches. Note, however, that we do not discuss a number of conventional methods which could also enhance our methods. These include neighbor-coupled shape models (*e.g.*, Tsai *et al.* [116] or Yang and Duncan [124]) for the thalamus and ventricles, and nested or hierarchical curve models for the central zone and the peripheral zone of the prostate [109] (*e.g.*, Yezzi *et al.* [127]).

Region-based user inputs

For conditional simulation, rather than specifying part of a curve, it may be more natural for a user or another algorithm to provide different types of segmentation guidance to our conditional simulation algorithm. One example of this would be to allow the user to specify the segmentation label on regions of the image. Thus if a region $\mathcal{R}_{\text{in}}$ is specified as being inside the curve, any candidate curve $\vec{\Gamma}$ which does not have $\mathcal{R}_{\text{in}}$ in its interior is rejected. Conversely, if the user forces $\mathcal{R}_{\text{out}}$ to be outside the curve, any candidate which includes a portion of $\mathcal{R}_{\text{out}}$ is rejected. These rejection conditions can be expressed mathematically as:

$$\mathcal{R}_{\text{in}} \cap \mathcal{R}_{\vec{\Gamma}}^{c} \neq \emptyset \tag{7.1}$$

$$\mathcal{R}_{\text{out}} \cap \mathcal{R}_{\vec{\Gamma}} \neq \emptyset \ . \tag{7.2}$$

For computational efficiency reasons, it is preferable if these constraints are also incorporated into the proposal distribution in some fashion. Otherwise if a constraint is close to being violated, many candidate samples will be automatically rejected for not satisfying the region constraints.

Uncertainty in user inputs

The approaches we presented for conditional simulation assume that the user input is correct. It may be advantageous to relax the constraint and allow some uncertainty to be associated with the user input. This could reduce the amount of time a user spends constructing the input by decreasing the level of required precision.

There are a number of ways we could imagine using uncertain user input. Given a user-specified portion of the curve $\vec{C}_k : [0, b] \to \mathbb{R}^2$, one approach is to construct a spatially-varying energy image $\varphi(\boldsymbol{x}; \vec{C}_k)$ which has low-energy "wells" around the user-specified curve segments. One way to specify this energy image is using the Hausdorff distance:

$$\varphi(\boldsymbol{x}; \vec{C}_k) = 1 - \exp(-\beta \min_{p \in [0,b]} \|\boldsymbol{x} - \vec{C}_k(p)\|) \ . \tag{7.3}$$

For small Hausdorff distance (the minimum distance between a point $\boldsymbol{x}$ and all the points on $\vec{C}_k$), φ is close to 0. For large Hausdorff distance, φ is close to 1. We can then add a term to the target distribution which prefers the curve to have a low-energy configuration:

$$\tilde{\pi}(\vec{C}; \vec{C}_k) \propto \pi(\vec{C}) \exp(- \oint_{\vec{C}} \varphi(\boldsymbol{x}; \vec{C}_k) \mathrm{d}s) \ . \tag{7.4}$$

This approach is similar in nature to the geodesic active contours method of Caselles *et al.* [10].

For the approach using region-based user input (described in the previous subsection), we can replace the hard constraints in (7.1) and (7.2) with related soft constraints:

$$\tilde{\pi}(\vec{C}; \mathcal{R}_{\text{in}}) \propto \pi(\vec{C}) \exp\left(- \iint_{\mathcal{R}_{\text{in}}} (1 - \mathcal{H}(\Psi_{\vec{C}}(\boldsymbol{x}))) \mathrm{d}\boldsymbol{x} \right) \tag{7.5}$$

$$\tilde{\pi}(\vec{C}; \mathcal{R}_{\text{in}}) \propto \pi(\vec{C}) \exp\left(- \iint_{\mathcal{R}_{\text{out}}} \mathcal{H}(\Psi_{\vec{C}}(\boldsymbol{x})) \mathrm{d}\boldsymbol{x} \right) \ . \tag{7.6}$$

An alternative approach is to transform the curve constraint into a region constraint. We can do this by enclosing the user-specified curve segment $\vec{C}_k$ between regions $\mathcal{R}_{\text{in}}$ and $\mathcal{R}_{\text{out}}$ which are inside and outside the curve respectively. See Fig. 7.1 for an illustration

Figure 7.1. Depiction of converting a known curve segment $\vec{C}_k$ into related region constraints $\mathcal{R}_{\text{in}}$ (light gray) and $\mathcal{R}_{\text{out}}$ (dark gray).

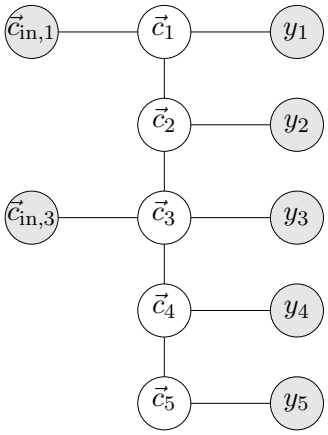


Figure 7.2. Length-5 chain with user-segmented curves specified as soft constraints. This adds extra "observed" nodes $\vec{c}_{\text{in},1}$ and $\vec{c}_{\text{in},3}$.

of this process. If we set the initial curve $\vec{C}^{(0)}$ to be on $\vec{C}_k$, then the actual curve is allowed to fluctuate between the regions but cannot substantially differ from $\vec{C}_k$.

For our hybrid 2D/3D models, uncertainty in the user-specified slice segmentations can be represented using the Markov model in Fig. 7.2. Here the user has supplied $\vec{c}_{\text{in},1}$ and $\vec{c}_{\text{in},3}$ which directly influence $\vec{c}_1$ and $\vec{c}_3$. The interactions are specified with a pairwise edge potential, and a natural choice for this potential is a shape distance function as in Chap. 6. Note that this approach can also allow curves specified in orthogonal planes to be inconsistent. This again can make the process of specifying partial segmentation information much easier by removing the onus (currently placed on the user in our implementation) to ensure that all segmentations supplied to the algorithm are consistent (*i.e.*, the segmentation label at a pixel $\boldsymbol{x}$ for a given constraint does not disagree with the label at that pixel set by any other constraints).

Markov single-loop models for curve segments

Felzenszwalb and Schwartz [28] describe a multiresolution tree-based shape model where the node elements are curve segments. This is reminiscent of the behavior of our algorithm for 2D conditional simulation in which some curve segments are specified, and the remaining curve segments are sampled given the known segments. A Markov model which contains a single loop (connecting the curve segments into a closed curve) may be an interesting way to model 2D curves and could lead to modeling flexibility similar to that of our hybrid 2D/3D Markov chain models. The shape-tree model of Felzenszwalb and Schwartz could also be a useful representation for the multiresolution proposal distributions described in Sec. 7.2.1.

Time-based curve models

The problem of doing inference on time-based Markov chain models for temporally-evolving curves has been studied by Sun *et al.* [103] using a particle filtering method and level set PCA shape representations. Instead of doing optimization, we could consider sampling from these models by adapting our hybrid 2D/3D models so that the variable changing between slices is time, not position. The techniques studied previously can largely be applied to this problem, though slight modifications would need to be made to handle directed Markov models instead of the undirected models used in Chap. 6.

■ 7.2.3 Improved sample visualization

The main techniques we currently use for visualizing the samples generated by our algorithm either display a subset of them (*e.g.*, most-probable samples) or only use marginal statistics (*e.g.*, marginal confidence bounds). The exception to this is the method based on PCA of the signed-distance function representations of the curve samples. Greater insight into the behavior of the target distribution π could be provided by improved methods to visualize the correlation structure.

PCA linear combinations

PCA is one method to build low-dimensional approximations to probability distributions. In Sec. 5.3.5, we visualized the principal modes of variation of an empirical shape distribution by computing the PCA decomposition and displaying individual eigenvectors at one standard deviation from the mean. Instead of restricting ourselves to one

mode at a time, we could build a tool that would let a user view linear combinations of the eigenvectors Ψ_i with arbitrary weights $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_N]^{\mathrm{T}}$:

$$\Psi_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \bar{\Psi} + \sum_{i=1}^{N} \lambda_i \sigma_i \Psi_i \ . \tag{7.7}$$

This could enable a relatively simple way for a user to manually examine likely curve configurations.

Interactive conditional simulation using PCA representations

Another application of the PCA representations is to increase the response speed of our algorithm when a user provides updated guidance. Currently we receive revised partial segmentation information from the user and use that information to generate an entirely new set of samples. A faster approximate approach is to search through the subspace defined by the eigenvectors to find the curve which most closely satisfies the new constraint[1].

Let $\vec{C}_{k,\tau} : [0, b] \rightarrow \mathbb{R}^2$ be the new constraint on the curve with τ indexing the sequence of constraints provided by the user. A vector of weights $\boldsymbol{\lambda}$ on the PCA eigenvectors defines a level set function $\Psi_{\boldsymbol{\lambda}}(\boldsymbol{x})$ which exists on the PCA subspace constructed from the samples using the previous set of constraints $\{\vec{C}_{k,1}, \vec{C}_{k,2}, \ldots, \vec{C}_{k,\tau-1}\}$. (as defined in (7.7)). We then wish to find the weights $\boldsymbol{\lambda}$ so that the curve $\vec{C}_{\boldsymbol{\lambda}}$ defined as the zeroth level set of $\Psi_{\boldsymbol{\lambda}}(\boldsymbol{x})$ is close to $\vec{C}_{k,\tau}$. We can formulate this as an energy functional to be minimized:

$$\mathcal{E}(\boldsymbol{\lambda}) = \min_{s} \int_0^b \|\vec{C}_k(p) - \vec{C}_{\boldsymbol{\lambda}}(s(p))\|^2 \mathrm{d}p \ . \tag{7.8}$$

The minimization over s maps $\vec{C}_{\boldsymbol{\lambda}}$ to the best match on $\vec{C}_k$. This is similar to the shape-based segmentation approach of Tsai *et al.* [116] in that we only need to search over a low-dimensional space of values to minimize $\mathcal{E}(\boldsymbol{\lambda})$. Finding such a minimum then results in a curve $\vec{C}_{\boldsymbol{\lambda}}$ that satisfies all constraints $\{\vec{C}_{k,1}, \vec{C}_{k,2}, \ldots, \vec{C}_{k,\tau-1}\}$, is close to the new constraint $\vec{C}_{k,\tau}$, and provides an estimate for the mode of the target distribution π.

Manifold-based representations

The space on which our curve probability distributions are defined is a high-dimensional nonlinear manifold. Reducing the dimensionality of the representation can often reveal

[1] All previous constraints are automatically satisfied because there is no variability at those locations.

a great deal about the underlying structure of the probability distribution. When most of the probability is concentrated on a small region of the manifold, the linear approximation constructed by PCA can represent the true distribution well. This approach fails, though, for multi-modal cases or when the probability distribution is spread over a region which is large relative to the curvature of the manifold at that location.

One natural extension to PCA is to combine a clustering technique (*e.g.*, the *ad hoc* clustering method we constructed for the Parzen-based shape distribution in Sec. 4.2.2 and the prostate example in Sec. 4.3.2) with local PCA approximations for each cluster (see Sloan *et al.* [99] for an implementation for a computer graphics application). Another approach to this problem involves applying the recent work in nonlinear dimensionality reduction such as the Isomap algorithm of Tenenbaum *et al.* [110], the locally linear embeddings of Roweis and Saul [90], and the dimensionality estimation tools of Costa and Hero [16]. These methods can be used to create methods that actually learn the structure of the probability distribution on the local area of the shape manifold and construct confidence bounds which are intrinsic to the manifold.

Appendix A

Curve Extraction Algorithm

For the curve extraction, we use a modified 2D version of the triangle-mesh extraction algorithm known as marching cubes [69]. The 2D variant is often referred to as marching squares and is a heuristic method to extract boundary points and line segments from a binary image. The algorithm looks at all 2×2 blocks of pixels in the image (for our narrowband representation, we need only look at blocks within the band as all curve points must be contained in the band). Curve points are placed between pixels where there is a change in label, and line segments connect the points. The 2×2 blocks can have 2^4 different permutations of labels (either 0 or 1). The key insight is that there are really only 4 different cases. Everything else is a rotated or complemented version of the base cases.

We will refer to the pixels in a 2×2 block with the indices $\begin{array}{|c|c|}\hline A & B \\ \hline C & D \\ \hline \end{array}$. For a coordinate system, the upper-left corner of pixel A is at $(0, 0)$, and the lower-right corner of pixel D is at $(2, 2)$. Any discretized measurements we have (*e.g.*, $\Psi_{\vec{C}}$) are defined to have been sampled at the center of each pixel. Now consider the 2×2 block $\begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array}$. All four pixels are outside of the curve, so clearly no curve points or segments exist in this block. Similar reasoning applies for the complementary block with all 1s.

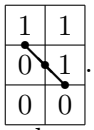
The block $\begin{array}{|c|c|}\hline 1 & 1 \\ \hline 0 & 1 \\ \hline \end{array}$ belongs to the base case with one 0 and three 1s. For this block, curve points exist between pixels A and C at $(\frac{1}{2}, 1)$ and pixels C and D at $(1, \frac{3}{2})$ with a line segment connecting the two points: $\begin{array}{|c|c|}\hline 1 & 1 \\ \hline 0 & 1 \\ \hline \end{array}$. The complementary block $\begin{array}{|c|c|}\hline 0 & 0 \\ \hline 1 & 0 \\ \hline \end{array}$ produces curve points and a line segment in the exact same location. Rotating the block simply causes the curve points to also rotate.

The base case with two 0s and two 1s can be illustrated with $\begin{array}{|c|c|}\hline 0 & 1 \\ \hline 0 & 1 \\ \hline \end{array}$. There are

two curve points, this time located between pixels A and B at $(1, \frac{1}{2})$ and pixels C and D at $(1, \frac{3}{2})$:  . Again, rotations or complements of this block produce similar configurations.

Finally, the last case has an ambiguity. Consider  . There are four curve points located at $(\frac{1}{2}, 1)$, $(1, \frac{1}{2})$, $(1, \frac{3}{2})$, and $(\frac{3}{2}, 1)$. The question is whether $(\frac{1}{2}, 1)$ is connected to $(1, \frac{1}{2})$ or $(1, \frac{3}{2})$. For the latter connection, that leaves $(1, \frac{1}{2})$ to get connected to $(\frac{3}{2}, 1)$ and line segments are formed separating pixels B and C:  . This is often referred to as 4-connectivity because only the 4 principals neighbors (up, down, left, or right) are used when considering connectivity. If instead we connect $(\frac{1}{2}, 1)$ to $(1, \frac{1}{2})$, a small isthmus is formed joining pixels B and C:  . This is often referred to as 8-connectivity as all 8 neighbors are used to determine connectivity. Either choice is acceptable. One just needs to be consistent about which choice is made.

The major problem with the standard marching squares algorithm for our purposes is that it produces an unsorted list of curve points and line segments, but when implementing our perturbations, we need an ordered list of points (starting from $\vec{C}(0)$ and going up to $\vec{C}(1)$). Sorting through this list to properly order the segments is an $\mathcal{O}(N^2)$ operation (because for each segment, we need to check all of the other segments to see which ones it connects to). We can avoid this step if we extract the line segments in an ordered fashion. If we examine the  case again, we can see that the bottom curve point at $(1, \frac{3}{2})$ will be connected to a curve point determined by the two pixels directly below pixels C and D. Thus based on the location of the curve point, we know which 2×2 block to examine next. This 2×2 block will share 2 pixels with the previous block and one curve point.

For instance, let  =  . In terms of coordinates, $(0, 1)$ is now the upper-left corner. There are curve points between pixels C and D at $(1, \frac{3}{2})$ and pixels D and F at $(\frac{3}{2}, 2)$. Coupled with the evaluation of the earlier 2×2 block with pixels A and B, we can see that $(\frac{1}{2}, 1)$ and $(\frac{3}{2}, 2)$ are connected through the point at $(1, \frac{3}{2})$:  .

Finally, we can modify the algorithm to allow for sub-pixel resolution in the curve locations since we actually have a level set representation, not just a binary image, and

our curve locations are defined as a zero-crossing of the level set. Let Ψ_{C} be the level set value at pixel C and Ψ_{D} be the level set value at pixel D, and consider the example with one 0 and three 1s again. While the y-coordinate of the curve point will remain at $\frac{3}{2}$, we can more accurately estimate the location of the x-coordinate by linearly interpolating the level set function values. We can write the value of the linear function $\tilde{\Psi}$ (as a function of x) between $\frac{1}{2}$ and $\frac{3}{2}$ as

$$\tilde{\Psi}(x) = \Psi_{\mathrm{C}} - (x - \tfrac{1}{2})(\Psi_{\mathrm{C}} - \Psi_{\mathrm{D}}) \ . \tag{A.1}$$

If we set this to zero and solve for x, we obtain:

$$\tilde{x} = \frac{1}{2} + \frac{\Psi_{\mathrm{C}}}{\Psi_{\mathrm{C}} - \Psi_{\mathrm{D}}} = \frac{1}{2} + \frac{1}{1 - \Psi_{\mathrm{D}}/\Psi_{\mathrm{C}}} \ . \tag{A.2}$$

Note that the quantity $\Psi_{\mathrm{D}}/\Psi_{\mathrm{C}}$ is always positive because either Ψ_{C} is negative and Ψ_{D} is positive or vice versa. Thus we can see that the value of $\tilde{x}$ is somewhere between $\frac{1}{2}$ and $\frac{3}{2}$.

Appendix B

Approximate Reverse Perturbations

We detailed in Sec. 3.3.3 three different approaches to estimate the reverse perturbation ϕ which maps the candidate sample $\vec{\Gamma}$ back to the previous iterate $\vec{C}$. We again define the parameter values p_0 and q_0 so they are related through $\vec{\Gamma}(p_0) = \vec{\Gamma}_{\mathrm{a}}(q_0)$. The forward perturbation equation (3.4) defines how the candidate sample point $\vec{\Gamma}(p_0)$ is obtained from $\vec{C}_{\mathrm{a}}(p_0)$.

The first approach assumes that $\vec{\mathcal{N}}_{\vec{\Gamma}}(p_0)$, the normal to the candidate sample $\vec{\Gamma}$ at p_0, is the same as $\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$, the normal to the curve at $\vec{C}_{\mathrm{a}}(p_0)$. The other two methods approximate $\vec{C}$ near p_0 using first- or second-order models and find the resulting estimate of ϕ in closed form. Here we present derivations of the closed-form formulas for the latter two methods.

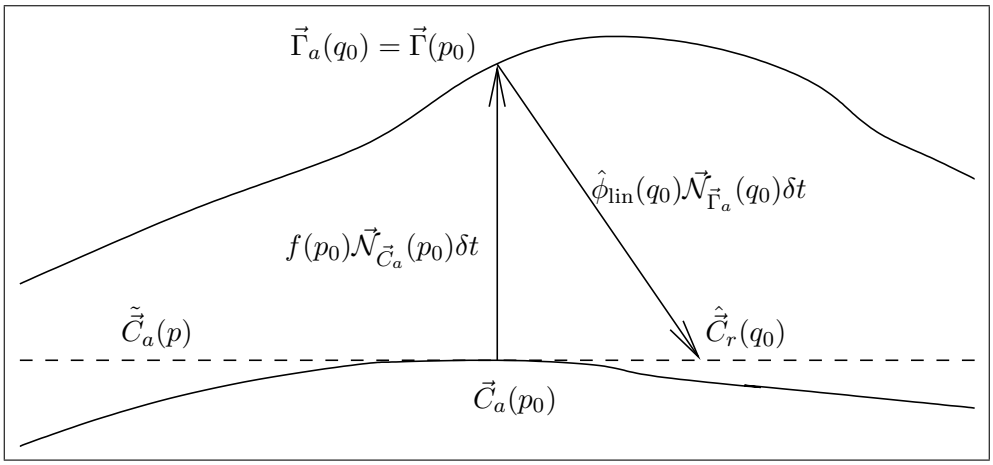
■ B.1 Linear Approximation

As noted in Sec. 3.3.3, we can approximate $\vec{C}_{\mathrm{a}}(p)$ around p_0 with the line $\vec{C}_{\mathrm{a}}(p_0) + b\,\vec{\mathcal{T}}_{\vec{C}_{\mathrm{a}}}(p_0)$ for $b \in \mathbb{R}$ and estimate $\phi(q_0)$ by finding the intersection of that line and the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$. We illustrate this process in Fig. B.1 for the cases when the curve is convex or concave. We can see that the linear approximation underestimates the magnitude of $\phi(q_0)$ for the convex case and overestimates it for the concave case.

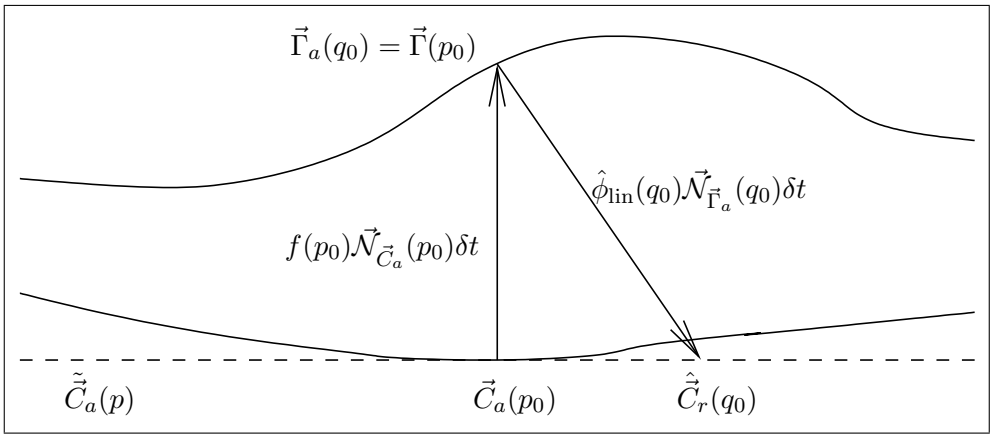
An equivalent description for the line defined by the tangent vector is the set of all points $\vec{v}$ such that

$$\left\langle \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0),\, \vec{v} - \vec{C}_{\mathrm{a}}(p_0) \right\rangle = 0 \ . \tag{B.1}$$

To find $\phi(q_0)$, we need to find the intersection between the line defined in (B.1) and the one formed by following the normal $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ from $\vec{\Gamma}_{\mathrm{a}}(q_0)$. This is done by defining $\vec{v}$ to be on the latter (*i.e.*, $\vec{v} = \vec{\Gamma}_{\mathrm{a}}(q_0) + \phi(q_0)\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)\delta t$ for some $\phi(q_0)$ value as in (3.18))



(a)



(b)

Figure B.1. Forming a linear approximation to $\vec{C}_{\text{a}}(p)$ to simplify the estimation of the reverse perturbation $\phi(q_0)$ when $\vec{C}_{\text{a}}$ is (a) convex or (b) concave at p_0.

and substituting that into (B.1):

$$\left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0),\, \vec{\Gamma}_\mathrm{a}(q_0) + \phi(q_0)\vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q_0)\delta t - \vec{C}_\mathrm{a}(p_0) \right\rangle \quad . \tag{B.2}$$

We can then simplify this by substituting the definition of the forward perturbation (from (3.4)) for $\vec{\Gamma}_\mathrm{a}(q_0)$:

$$\begin{aligned} 0 &= \left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0),\, (\vec{C}_\mathrm{a}(p_0) + f(p_0)\vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0)\delta t) + \phi(q_0)\vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q_0)\delta t - \vec{C}_\mathrm{a}(p_0) \right\rangle \\ &= \left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0),\, f(p_0)\vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0)\delta t + \phi(q_0)\vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q_0)\delta t) \right\rangle \\ &= f(p_0)\delta t + \phi(q_0)\left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0),\, \vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q_0) \right\rangle \delta t \end{aligned} \tag{B.3}$$

We can then see that

$$\hat{\phi}_\mathrm{lin}(q_0) = -\frac{f(p_0)}{\left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0),\, \vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q_0) \right\rangle} \quad . \tag{B.4}$$

This method is reminiscent in many ways of the Newton-Raphson method for finding zeros of a function. In that method successive linear approximations to the function are constructed, and the next iterate value is computed as the location where the linear approximation intersects the x-axis. We can also use an iterative method here to refine our estimate of ϕ. For each iteration, we need to compute both $\hat{\phi}^{(\tau)}(q_0)$ and the corresponding intersection point $\vec{C}_\mathrm{r}^{(\tau)}(q_0)$. Then for the next iteration, we construct a linear approximation around $\vec{C}_\mathrm{r}^{(\tau)}(q_0)$ instead of $\vec{C}_\mathrm{a}(p_0)$. The equation derived in (B.4) no longer holds, and we must use the more complex:

$$\hat{\phi}_\mathrm{lin}^{(\tau)}(q_0) = -\frac{\left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{r}^{(\tau-1)}}(q_0),\, \vec{C}_\mathrm{r}^{(\tau-1)}(q_0) - \vec{\Gamma}_\mathrm{a}(q_0) \right\rangle}{\left\langle \vec{\mathcal{N}}_{\vec{C}_\mathrm{r}^{(\tau-1)}}(q_0),\, \vec{\mathcal{N}}_{\vec{\Gamma}_\mathrm{a}}(q_0) \right\rangle \delta t} \quad . \tag{B.5}$$

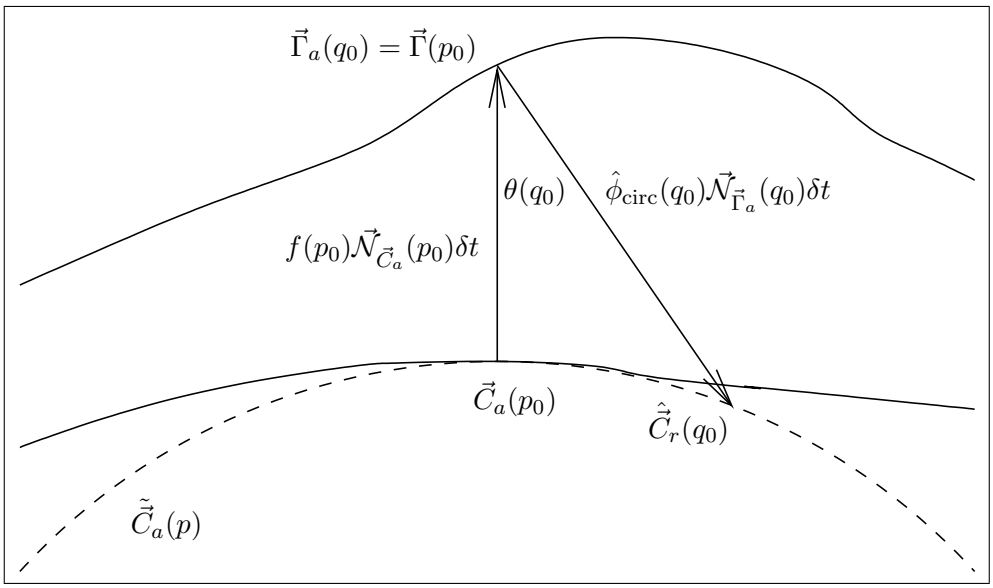
Like Newton-Raphson, there is no guarantee that this process will actually converge.

■ B.2 Second-order Approximation

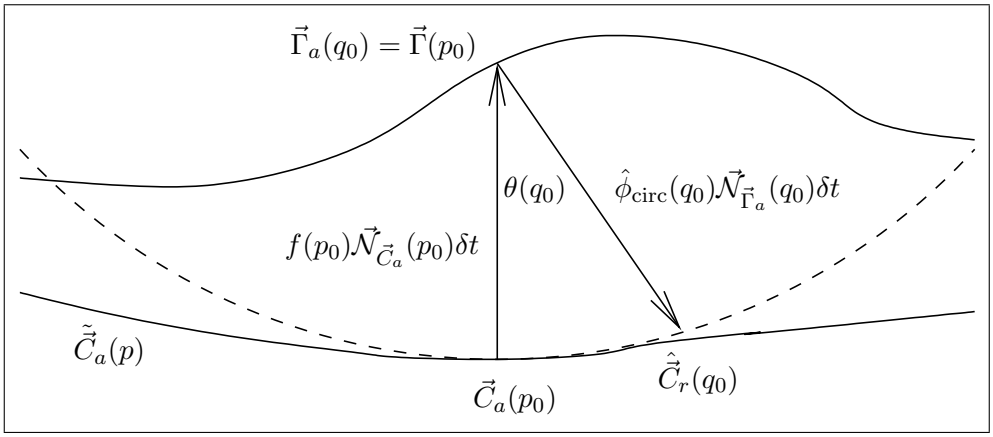
By incorporating curvature, we can define a local approximation using a circle rather than a line. As detailed in Sec. 3.3.3, the normal $\vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0)$ and the curvature $\kappa_{\vec{C}_\mathrm{a}}(p_0)$ define a circle that passes through $\vec{C}_\mathrm{a}(p_0)$ with radius $1/\kappa_{\vec{C}_\mathrm{a}}(p_0)$ and center $\vec{C}_\mathrm{a}(p_0) - \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0)/\kappa_{\vec{C}_\mathrm{a}}(p_0)$. This is again illustrated for convex and concave cases in Fig. B.2.

All points $\vec{v}$ on the circle must then satisfy:

$$\left\| \vec{v} - \left(\vec{C}_\mathrm{a}(p_0) - \vec{\mathcal{N}}_{\vec{C}_\mathrm{a}}(p_0)/\kappa_{\vec{C}_\mathrm{a}}(p_0) \right) \right\|^2 = 1/\kappa_{\vec{C}_\mathrm{a}}^2(p_0) \quad . \tag{B.6}$$



(a)



(b)

Figure B.2. Forming an approximation to $\vec{C}_{\text{a}}(p)$ using a circle to simplify the estimation of the reverse perturbation $\phi(q_0)$ when $\vec{C}_{\text{a}}$ is (a) convex or (b) concave at p_0.

Performing the same calculations as in the linear case by adding in (3.18) and using (3.4) results in:

$$\begin{aligned}
&\|\vec{\Gamma}_{\mathrm{a}}(q_0) + \phi(q_0)\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)\delta t - \vec{C}_{\mathrm{a}}(p_0) + \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)/\kappa_{\vec{C}_{\mathrm{a}}}(p_0)\|^2 = 1/\kappa^2_{\vec{C}_{\mathrm{a}}}(p_0) \\
&\Rightarrow \|f(p_0)\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)\delta t + \phi(q_0)\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)\delta t + \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)/\kappa_{\vec{C}_{\mathrm{a}}}(p_0)\|^2 = 1/\kappa^2_{\vec{C}_{\mathrm{a}}}(p_0) \\
&\Rightarrow \delta t^2\phi^2(q_0) + 2\xi_{\vec{C}_{\mathrm{a}}}(p_0)\cos\theta(q_0)\delta t\phi(q_0) + \xi^2_{\vec{C}_{\mathrm{a}}}(p_0) - 1/\kappa^2_{\vec{C}_{\mathrm{a}}}(p_0) = 0
\end{aligned} \tag{B.7}$$

where $\xi_{\vec{C}_{\mathrm{a}}}(p_0) = f(p_0)\delta t + 1/\kappa_{\vec{C}_{\mathrm{a}}}(p_0)$ and $\cos\theta(q_0) = \left\langle \vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0), \vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0) \right\rangle$ (so $\theta(q_0)$ is the angle between $\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$ and $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ as indicated on Fig. B.2).

This is a quadratic equation in terms of $\phi(p_0)$ which can solved in closed form as:

$$\begin{aligned}
\hat{\phi}_{\mathrm{circ}}(q_0) &= -\frac{\xi_{\vec{C}_{\mathrm{a}}}(p_0)\cos\theta(q_0)}{\delta t} \pm \frac{1}{\delta t}\sqrt{(\cos^2\theta(q_0) - 1)\xi^2_{\vec{C}_{\mathrm{a}}}(p_0) + 1/\kappa^2_{\vec{C}_{\mathrm{a}}}(p_0)} \\
&= -\left(f(p_0) + \frac{1}{\kappa_{\vec{C}_{\mathrm{a}}}(p_0)\delta t}\right)\cos\theta(q_0) \pm \frac{\sqrt{1 - (1 + f(p_0)\kappa_{\vec{C}_{\mathrm{a}}}(p_0)\delta t)^2\sin^2\theta(q_0)}}{|\kappa_{\vec{C}_{\mathrm{a}}}(p_0)|\delta t}
\end{aligned} \tag{B.8}$$

There are four main cases we need to consider here when deciding whether to use the positive or negative branch (which we will refer to as $\hat{\phi}^+(q_0)$ and $\hat{\phi}^-(q_0)$ respectively). The sign of $f(p_0)$ determines whether $\vec{\Gamma}_{\mathrm{a}}(q_0)$ is located inside (for $f(p_0) < 0$) or outside (for $f(p_0) > 0$) of $\vec{C}_{\mathrm{a}}(p)$. Similarly, the sign of $\kappa_{\vec{C}_{\mathrm{a}}}(p_0)$ determines whether the center of the approximating circle is inside (for $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) > 0$) or outside (for $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) < 0$) the curve. For $f(p_0) > 0$, we illustrate the cases where $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) > 0$ in Fig. B.3 and $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) > 0$ in Fig. B.4. We have drawn the circles in an exaggeratedly small manner (*i.e.*, with much higher curvature than the curve actually exhibits at $\vec{C}_{\mathrm{a}}(p_0)$) to illustrate the nature of the intersections. Note that geometrically, these cases are identical to situations where $f(p_0) < 0$ except with the sign of $\kappa_{\vec{C}_{\mathrm{a}}}(p_0)$ changed and the normal vectors to the curves pointing in the opposite direction. For instance, Fig. B.3 can represent the case when $f(p_0) > 0$ and $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) > 0$ (so the outward normal of $\vec{C}$ points toward the top of the page) or $f(p_0) < 0$ and $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) < 0$ (so the outward normal of $\vec{C}$ points toward the bottom of the page).

For $f(p_0) > 0$, intuitively we would expect that $\hat{\phi}_{\mathrm{circ}}(q_0)$ would be less than zero as it would be unlikely using small perturbations for $\vec{\mathcal{N}}_{\vec{C}_{\mathrm{a}}}(p_0)$ and $\vec{\mathcal{N}}_{\vec{\Gamma}_{\mathrm{a}}}(q_0)$ to be pointing in completely opposite directions. When $\kappa_{\vec{C}_{\mathrm{a}}}(p_0) > 0$, both solutions for $\hat{\phi}(q_0)$ are negative and correspond to first intersecting the near side of the circle then passing

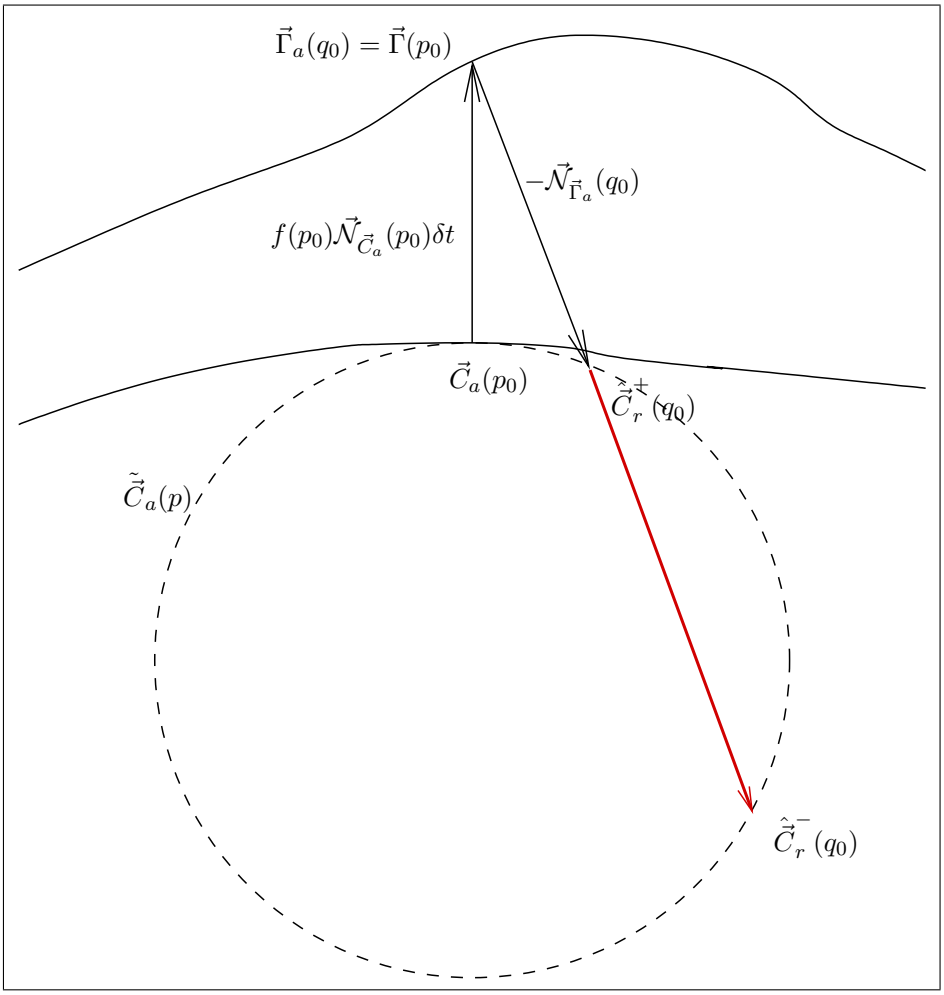


Figure B.3. Graphical depiction of both solutions to the quadratic equation for the circle approximation when $f(p_0) > 0$ and $\kappa_{\vec{C}_a}(p_0) > 0$ with the outward normal $\vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$ pointing toward the top of the page. The first intersection (at $\vec{C}_r^{+}(q_0)$) corresponds to $\hat{\phi}^{+}(q_0)$ and the second (at $\vec{C}_r^{-}(q_0)$) to $\hat{\phi}^{-}(q_0)$ (following the extension of the line in red). Geometrically, this case is identical to the dual situation when $f(p_0) < 0$, $\kappa_{\vec{C}_a}(p_0) < 0$, and the outward normal $\vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$ points toward the bottom of the page. The first intersection would then be $\hat{\phi}^{-}(q_0)$ and the second $\hat{\phi}^{+}(q_0)$.

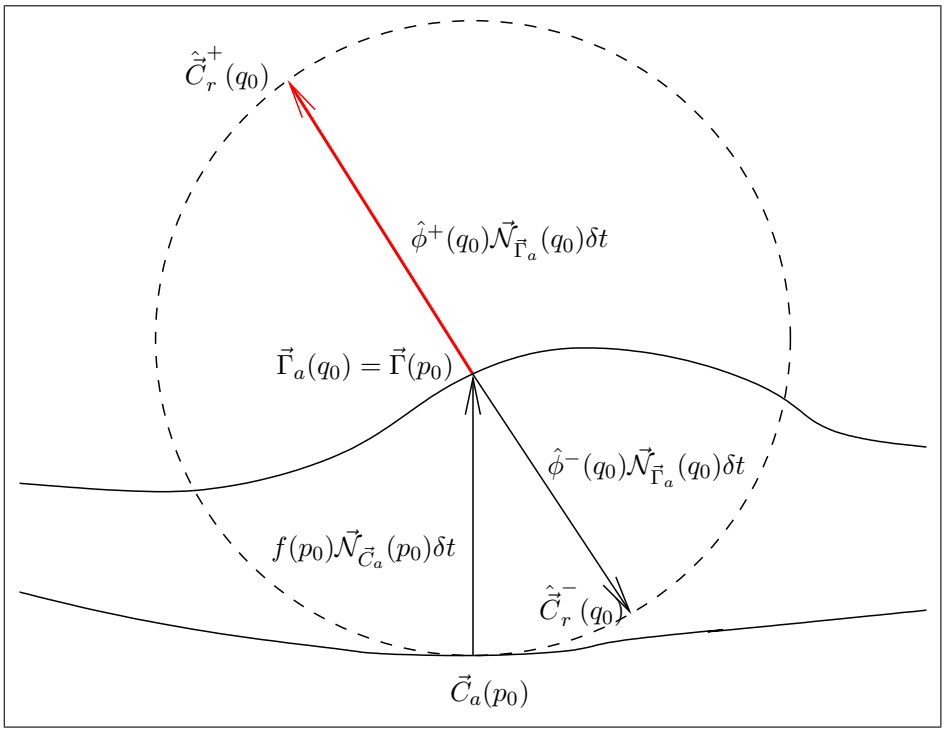


Figure B.4. Graphical depiction of both solutions to the quadratic equation for the circle approximation when $f(p_0) > 0$ and $\kappa_{\vec{C}_a}(p_0) < 0$ with the outward normal $\vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$ pointing toward the top of the page. The black line intersecting the lower-right corner of the circle corresponds to $\hat{\phi}^{-}(q_0)$ and the red line to $\hat{\phi}^{+}(q_0)$. Geometrically, this case is identical to the dual situation when $f(p_0) < 0$, $\kappa_{\vec{C}_a}(p_0) > 0$, and the outward normal $\vec{\mathcal{N}}_{\vec{\Gamma}_a}(q_0)$ points toward the bottom of the page. The bottom intersection would then be $\hat{\phi}^{+}(q_0)$ and the top intersection $\hat{\phi}^{-}(q_0)$.

all the way through the circle before intersecting the far side. This is illustrated in Fig. B.3. Clearly the first answer is the one we want, and, because $|\hat{\phi}^+(q_0)| \leq |\hat{\phi}^-(q_0)|$, this corresponds to $\hat{\phi}^+(q_0)$.

When $\kappa_{\vec{C}_\mathrm{a}}(p_0) < 0$, we can see in Fig. B.4 that there is one negative solution and one positive solution. The correct solution must be $\hat{\phi}^-(q_0)$ as that solution corresponds to tracing back toward $\vec{C}$. Similar analysis can be applied for $f(p_0) < 0$ to show that $\hat{\phi}^+(q_0)$ should be used for $\kappa_{\vec{C}_\mathrm{a}}(p_0) > 0$ and $\hat{\phi}^-(q_0)$ for $\kappa_{\vec{C}_\mathrm{a}}(p_0) < 0$. Therefore we can see that we should choose the branch based on the sign of $\kappa_{\vec{C}_\mathrm{a}}(p_0)$. This simplifies the overall expression to

$$\hat{\phi}_{\mathrm{circ}}(q_0) = -f(p_0)\cos\theta(q_0) + \frac{\sqrt{1-(1+f(p_0)\kappa_{\vec{C}_\mathrm{a}}(p_0)\delta t)^2\sin^2\theta(q_0)} - \cos\theta(q_0)}{\kappa_{\vec{C}_\mathrm{a}}(p_0)\delta t} \ . \tag{B.9}$$

Appendix C

Forward-Backward Algorithm for Undirected HMMs

The forward-backward algorithm is an efficient method to sample from the posterior distribution of a hidden Markov model (HMM). Here we assume that we have a static HMM with a state space $\boldsymbol{\chi} = \{\chi_1, \ldots, \chi_N\}$, observations $\boldsymbol{y} = \{y_1, \ldots, y_N\}$, and a posterior distribution which, due to its chain structure, can be factored into potential functions (as in (6.1) in Sec. 6.1) as:

$$\mathrm{p}(\boldsymbol{\chi} \,|\, \boldsymbol{y}) \propto \prod_{i=1}^{N} \Phi_i(\chi_i) \prod_{i=1}^{N-1} \Phi_{i,i+1}(\chi_i, \chi_{i+1}) \prod_{i=1}^{N} \Lambda_i(\chi_i, y_i) \ . \tag{C.1}$$

We group together terms as:

$$G_1(\chi_1) = \Phi_1(\chi_1)\Lambda_1(\chi_1, y_1) \tag{C.2}$$

$$G_i(\chi_{i-1}, \chi_i) = \Phi_i(\chi_i)\Phi_{i-1,i}(\chi_{i-1}, \chi_i)\Lambda_i(\chi_i, y_i), \ i \in \{2, \ldots, N\} \tag{C.3}$$

to obtain the following form of the posterior:

$$\mathrm{p}(\boldsymbol{\chi} \,|\, \boldsymbol{y}) \propto G_1(\chi_1) \prod_{i=2}^{N} G_i(\chi_{i-1}, \chi_i) \ . \tag{C.4}$$

An alternative factorization of the posterior distribution is in terms of transition probabilities along the chain:

$$\mathrm{p}(\boldsymbol{\chi} \,|\, \boldsymbol{y}) = \mathrm{p}(\chi_1 \,|\, \boldsymbol{y}) \prod_{i=2}^{N} \mathrm{p}(\chi_i \,|\, \chi_{i-1}, \boldsymbol{y}) \ . \tag{C.5}$$

From this decomposition, we can see that one method to generate a sample from the complete posterior distribution is to first sample from $\mathrm{p}(\chi_1 \,|\, \boldsymbol{y})$, then to sample from

$\mathrm{p}(\chi_2 \,|\, \chi_1, \boldsymbol{y})$ (given the previously sampled value of χ_1), and so on until we have sampled all χ_i from 1 to N. Generating these samples is simple because cumulative mass functions can be easily computed from probability mass functions, and a uniform random variable on $[0, 1]$ can be used to generate a sample of a discrete random variable characterized by its cumulative mass function.

We can compute the marginal distribution of χ_1 conditioned on all of the observations $\boldsymbol{y}$ (the first term in (C.5)) by summing the posterior distribution over χ_2 to χ_N:

$$\mathrm{p}(\chi_1 \,|\, \boldsymbol{y}) = \sum_{\chi_2} \cdots \sum_{\chi_N} \mathrm{p}(\boldsymbol{\chi} \,|\, \boldsymbol{y}) \ . \tag{C.6}$$

In general, computing this has exponential complexity as there are K^N elements in $\mathrm{p}(\boldsymbol{\chi} \,|\, \boldsymbol{y})$.

Using the form of the posterior in (C.4), we can rewrite (C.6):

$$\mathrm{p}(\chi_1 \,|\, Y) \propto G_1(\chi_1) \sum_{\chi_2} G_2(\chi_1, \chi_2) \cdots \sum_{\chi_N} G_N(\chi_{N-1}, \chi_N) \ . \tag{C.7}$$

The summation of G_N over χ_N results in a function which only depends on χ_{N-1}:

$$m_{N \to N-1}(\chi_{N-1}) = \sum_{\chi_N} G_N(\chi_{N-1}, \chi_N) \ . \tag{C.8}$$

This process can be continued to obtain:

$$m_{i \to i-1}(\chi_{i-1}) = \sum_{\chi_i} G_i(\chi_{i-1}, \chi_i) m_{i+1 \to i}(\chi_i) \ . \tag{C.9}$$

This culminates in:

$$\mathrm{p}(\chi_1 \,|\, \boldsymbol{y}) \propto G_1(\chi_1) m_{2 \to 1}(\chi_1) \ . \tag{C.10}$$

To obtain the actual probability mass values, we simply need to normalize the $G_1(\chi_1) m_{2 \to 1}(\chi_1)$ values so that $\mathrm{p}(\chi_1 \,|\, \boldsymbol{y})$ sums to 1.

The forward sweep of the model is identical to the well-known *belief propagation* algorithm [6]. In this method, the intermediate functions $m_{i \to i-1}$ are referred to as messages from node i to node $i - 1$. This message summarizes all of the information node i has about node $i - 1$ from its own observations and those from nodes $i + 1$ to N.

The reverse computation then needs to compute $\mathrm{p}(\chi_i \,|\, \chi_{i-1}, \boldsymbol{y})$ for i from 2 to N. When χ_{i-1} has a fixed value, the Markov nature of the chain means that the nodes from 1 to $i - 2$ do not contribute to $\mathrm{p}(\chi_i \,|\, \chi_{i-1}, \boldsymbol{y})$, and the conditional probability is

proportional to the joint probability $\mathrm{p}(\chi_{i-1}, \chi_i \,|\, \boldsymbol{y})$ (as $\mathrm{p}(\chi_{i-1} \,|\, \boldsymbol{y})$ is a constant). The joint probability is:

$$\begin{aligned}\mathrm{p}(\chi_{i-1}, \chi_i \,|\, \boldsymbol{y}) &\propto \sum_{\chi_{i+1}} \cdots \sum_{\chi_N} \prod_{j=i}^{N} G_j(\chi_{j-1}, \chi_j) \\ &\propto G_i(\chi_{i-1}, \chi_i) \sum_{\chi_{i+1}} G_N(\chi_i, \chi_{i+1}) \cdots \sum_{\chi_N} G_N(\chi_{N-1}, \chi_N) \\ &\propto G_i(\chi_{i-1}, \chi_i) m_{i+1 \to i}(\chi_i) \ . \end{aligned} \tag{C.11}$$

This equation fuses the information from all nodes to the right with the information that the specific selection of χ_{i-1} provides us about χ_i.

Using the filtering and smoothing computations in (C.10) and (C.11), we can construct the factorization of the posterior density in (C.5) and generate samples from the HMM. The forward sweep of the algorithm needs to compute $N-1$ messages with $\mathcal{O}(K^2)$ complexity for each message, so the overall forward computation has complexity $\mathcal{O}(K^2 N)$. The reverse sweep needs to compute $N-1$ conditional probabilities, each of which requires linear computation in K. Therefore overall complexity for the reverse sweep is $\mathcal{O}(KN)$.

Bibliography

[1] P. Abolmaesumi and M. Sirouspour. An interacting multiple model probabilistic data association filter for cavity boundary extraction from ultrasound images. *IEEE Trans. Med. Imag.*, 23(6):772–784, 2004.

[2] K. Aki and P. G. Richards. *Quantitative Seismology.* Freeman, 1980.

[3] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, 2nd edition, 1999.

[4] D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena, 3rd edition, 2005.

[5] C. M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, 1995.

[6] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[7] R. Blakely. *Potential Theory in Gravity and Magnetic Applications.* Cambridge University Press, 1995.

[8] H. Blum. A transformation for extracting new descriptors of shape. In W. Wathen-Dunn, editor, *Symp. Models Percep. Speech Visual Form*, pages 362–380. MIT Press, 1967.

[9] P. Bremaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation and Queues.* Springer-Verlag, 1999.

[10] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *Intl. J. Comp. Vis.*, 22(1):61–79, 1997.

[11] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Trans. Imag. Proc.*, 10:266–277, 2001.

[12] D. Chauveau and J. Diebolt. An automated stopping rule for mcmc convergence assessment. Technical Report 3566, INRIA, 1998.

[13] T. F. Cootes, C. Beeston, G. J. Edwards, and C. J. Taylor. A unified framework for atlas matching using active appearance models. In *Proc. of IPMI*, number 1613 in LNCS, 1999.

[14] T. F. Cootes, D. Cooper, C. J. Taylor, and J. Graham. Active shape models - their training and application. *Comp. Vis. and Image Understanding*, 61(1):38–59, Jan 1995.

[15] T. F. Cootes and C. J. Taylor. Statistical models of appearance for computer vision. Technical report, Manchester University, 2001. http://www.isbe.man.ac.uk/~bim/Models/app_model.ps.gz.

[16] J. A. Costa and A. O. Hero. Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Trans. Sig. Proc.*, 52(8):2210–2221, 2004.

[17] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

[18] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag, 1999.

[19] M. K. Cowles and B. P. Carlin. Markov chain Monte Carlo diagnostics: A comparative review. *J. Am. Stat. Soc.*, 91(434):883–904, 1996.

[20] D. Cremers and S. Soatto. A pseudo-distance for shape priors in level set segmentation. In *Proc. of 2nd IEEE Workshop on Var., Geom., and Level Set Methods*, 2003.

[21] E. Davies. *Machine Vision: Theory, Algorithms and Practicalities*. Academic Press, 1990.

[22] M. de Bruijne and M. Nielsen. Shape particle filtering for image segmentation. In *MICCAI 2004, LNCS 3216*, volume I, pages 168–175, 2004.

[23] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[24] P. M. Djuric, Y. Huang, and T. Ghirmai. Perfect sampling: a review and applications to signal processing. *IEEE Trans. Sig. Proc.*, 50(2):345–356, 2002.

[25] A. Falcao, J. Udupa, S. Samarasekera, S. Sharma, B. Hirsch, and R. Lotufo. User-steered image segmentation paradigms - Live Wire and Live Lane. *Graph. Mod. and Image Proc.*, 60(4):233–260, 1998.

[26] A. Fan, W. M. Wells III, J. W. Fisher III, M. Cetin, S. Haker, R. Mulkern, C. Tempany, and A. Willsky. A unified variational approach to denoising and bias correction in mr. In *Proceedings of IPMI 2003*, pages 148–159, 2003.

[27] A. C. Fan. A variational approach to MR bias correction. Master's thesis, MIT, 2003.

[28] P. Felzenszwalb and J. Schwartz. Hierarchical matching of deformable shapes. In *CVPR*, 2007.

[29] C. Florin, N. Paragios, and J. Williams. Globally optimal active contours, sequential Monte Carlo and on-line learning for vessel segmentation. In *ECCV*, pages 476–489, 2006.

[30] M. Gage and R. S. Hamilton. The heat equation shrinking convex plane curves. *J. Diff. Geom.*, 23:69–96, 1986.

[31] D. Gamerman. *Markov Chain Monte Carlo: Stochastic simulation for Bayesian inference*. Chapman & Hall, 1997.

[32] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE PAMI*, 6:721–741, 1984.

[33] D. Gering, W. E. L. Grimson, and R. Kikinis. Recognizing deviations from normalcy for brain tumor segmentation. In *MICCAI*, 2002.

[34] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. CRC Press, 1996.

[35] P. Golland, W. E. L. Grimson, M. E. Shenton, and R. Kikinis. Deformation analysis for shaped based classification. In *Proceedings of IPMI*, LNCS 2082, pages 517–530, 2001.

[36] M. Grayson. The heat equation shrinks embedded plane curves to round points. *J. Diff. Geom.*, 26:285–314, 1987.

[37] L. Greengard and V. Rokhlin. A new version of the fast multipole method for the Laplace equation in three dimensions. *Acta Numerica*, 6:229–269, 1997.

[38] U. Grenander and M. I. Miller. Representations of knowledge in complex systems. *J. of the Royal Stat. Soc. Series B*, 56(4):549–603, 1994.

[39] U. Grenander and M. I. Miller. Computational anatomy: An emerging discipline. *Quarterly of Applied Mathematics*, LVI(4):617–694, 1998.

[40] W. E. L. Grimson, R. Kikinis, F. Jolesz, and P. Black. Image-guided surgery. *Scientific American*, June 1999.

[41] S. Haker, L. Zhu, A. Tannenbaum, and S. Angenant. Optimal mass transport for registration and warping. *Intl. J. Comp. Vis.*, 60(3):225–240, 2004.

[42] J. M. Hammersley and P. Clifford. Markov field on finite graphs and lattices. unpublished manuscript, 1971.

[43] X. Han, C. Xu, and J. Prince. A topology preserving level set method for geometric deformable models. *IEEE Trans. Patt. Anal. Mach. Intell.*, 25(6):755–768, 2003.

[44] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[45] R. A. Horn and C. R. Johson. *Matrix Analysis*. Cambridge University Press, 1990.

[46] H. Hricak, S. White, D. Vigneron, J. Kurhanewicz, A. Kosco, et al. Carcinoma of the prostate gland: MR imaging with pelvic phased-array coils versus integrated endorectal-pelvic phased-array coils. *Radiology*, 193:703–709, 1994.

[47] B. R. Hunt, R. L. Lipsman, J. M. Rosenberg, et al. *A Guide to MATLAB, 2e: for Beginners and Experienced Users*. Cambridge University Press, 2006.

[48] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer, New York, 2001.

[49] M. W. Jessell. Three-dimensional modeling of potential-field data. *Comp. & Geosci.*, 27:455–465, 2001.

[50] O. Juan, R. Keriven, and G. Postelnicu. Stochastic motion and the level set method in computer vision: Stochastic active contours. *Intl. J. Comp. Vis.*, 69(1), 2006.

[51] I. Karatzas and S. Shreve. *Brownian Motion and Stochastic Calculus*. Springer-Verlag, 2nd edition, 1991.

[52] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Intl. J. Comp. Vis.*, 1(4):321–331, 1988.

[53] D. Kendall, D. Barden, T. Carne, and H. Le. *Shape and shape theory*. Wiley, 1999.

[54] A. D. Kennedy and J. Kuti. Noise without noise: a new Monte Carlo method. *Phys Rev Letters*, 54(23):2473–2476, 1985.

[55] R. Kikinis, M. E. Shenton, D. V. Iosifescu, et al. A digital brain atlas for surgical planning, model driven segmentation and teaching. *IEEE Trans. Vis. and Comp. Graphics*, 2(3):232–241, 1996.

[56] J. Kim. *Nonparametric Statistical Methods for Image Segmentation and Shape Analysis*. PhD thesis, MIT, 2005.

[57] J. Kim, M. Cetin, and A. S. Willsky. Nonparametric shape priors for active contour-based image segmentation. *Signal Proc.*, to appear.

[58] J. Kim, J. W. I. Fisher, A. J. Yezzi, M. Cetin, and A. S. Willsky. A nonparametric statistical method for image segmentation using information theory and curve evolution. *IEEE Trans. Imag. Proc.*, 14(10):1486–1502, 2005.

[59] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[60] E. Klassen, A. Srivastava, W. Mio, and S. Joshi. Analysis of planar shapes using geodesic paths on shape spaces. *IEEE Trans. Patt. Anal. Mach. Intell.*, 26(3):372–383, 2004.

[61] R. Krahenbuhl and Y. Li. Hybrid optimization for a binary inverse problem. In *Intl. Mtg. Soc. Expl. Geophys.*, 2004.

[62] S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[63] L. B. Leopold and W. B. Langbein. River meanders. *Sci. Am.*, 214:60–70, 1966.

[64] M. Leventon. *Statistical Shape Models in Medical Image Analysis*. PhD thesis, MIT, 2000.

[65] M. Leventon, W. E. L. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *IEEE Conf. on Comp. Vis. and Patt. Recog.*, pages 316–323, 2000.

[66] L. Lin, K. F. Liu, and J. Sloan. A noisy Monte Carlo algorithm. *Phys. Rev. D*, 61(7), 2000.

[67] P. Lions and P. Souganidis. Fully nonlinear stochastic partial differential equations. *C.R. Acad. Sci. Paris Series I Math*, 326:1085–1092, 1998.

[68] S. P. Liou, Chiu, A. H., and R. C. Jain. A parallel technique for signal-level perceptual organization. *IEEE Trans. Patt. Anal. Mach. Intell.*, 13:317–325, 1991.

[69] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21(3):163–169, 1987.

[70] R. Malladi, J. Sethian, and B. Vemuri. Shape modeling with front propagation: a level set approach. *IEEE Trans. Patt. Anal. Mach. Intell.*, 17:158–175, 1995.

[71] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *J. Chem. Phys.*, 21(6), 1953.

[72] E. N. Mortensen and W. A. Barrett. Interactive segmentation with intelligent scissors. *Graph. Mod. and Image Proc.*, 60(5):349–384, 1998.

[73] D. Mumford. Mathematical models of shape: do they model perception? In *Proc. of SPIE*, volume 1570, pages 2–10, 1991.

[74] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math*, 42:577–685, 1989.

[75] M. N. Nabighian, M. E. Ander, V. Grauch, R. O. Hansen, et al. Historical development of the gravity method in exploration. *Geophysics*, 70(6):63–89, 2005.

[76] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Univ. of Toronto, 1993.

[77] R. M. Neal, M. J. Beal, and S. T. Roweis. Inferring state sequences for non-linear systems with embedded hidden markov models. In e. a. S. Thrun, editor, *Advances in Neural Information Processing Systems 16*, 2003.

[78] B. Oksendal. *Stochastic Differential Equations. An Introduction with Applications.* Springer-Verlag, 5th edition, 2000.

[79] D. W. Oldenburg. The inversion and interpretation of gravity anomalies. *Geophysics*, 39:526–536, 1974.

[80] B. O'Neill. *Elementary Differential Geometry.* Academic Press, 2nd edition, 1997.

[81] A. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-Time Signal Processing.* Prentice-Hall, 1999.

[82] S. Osher and J. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulation. *J. Comp. Phys.*, 79:12–49, 1988.

[83] R. Osserman. *A Survey of Minimal Surfaces.* Dover, 1986.

[84] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 4th edition, 2001.

[85] P. Perez, A. Blake, and M. Gangnet. Jetstream: probabilistic contour extraction with particles. In *Proc. ICCV*, volume 2, pages 524–531, 2001.

[86] K. M. Pohl, J. Fisher, W. E. L. Grimson, R. Kikinis, and W. Wells. A Bayesian model for joint segmentation and registration. *Neuroimage*, 31:228–239, 2006.

[87] A. Protiere and G. Sapiro. Interactive image segmentation via adaptive weighted distances. *IEEE Trans. Imag. Proc.*, 16(4):1046–1057, 2007.

[88] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[89] B. D. Ripley. *Stochastic Simulation.* Wiley, 2006.

[90] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[91] R. W. Saltus and R. J. Blakely. HYPERMAG, an interactive, two-dimensional gravity and magnetic modeling program. Technical Report 83-241, U.S. Geol. Survey, 1983.

[92] G. Sapiro. *Geometric Partial Differential Equations and Image Analysis*. Cambridge University Press, 2001.

[93] G. Sapiro and A. Tannenbaum. On affine plane curve evolution. *J. of Func. Anal.*, 119:79–120, 1994.

[94] S. L. Scott. Bayesian methods for hidden markov models: Recursive computing in the 21st century. *J. Am. Stat. Assoc.*, 97:337–351, 2000.

[95] F. Segonne, J. P. Pons, W. E. L. Grimson, and B. Fischl. Active contours under topology control - genus preserving level sets. In *ICCV MMBIA*, pages 135–145, 2005.

[96] J. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[97] J. A. Sethian. Curvature and the evolution of fronts. *Comm. in Math. and Physics*, 101:487–499, 1985.

[98] R. E. Sheriff and L. P. Geldart. *Exploration Seismology*. Cambridge University Press, 1995.

[99] P. Sloan, J. Hall, J. Hart, and J. Snyder. Clustered principal components for precomputed radiance transfer. In *Proc. of ACM SIGGRAPH*, pages 382–391, 2003.

[100] G. Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley, MA, 1986.

[101] G. Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, Wellesley, MA, 1993.

[102] B. Stroustrup. *The C++ Programming Language*. Addison-Wesley, 3rd edition, 2000.

[103] W. Sun, M. Cetin, R. Chan, et al. Segmenting and tracking the left ventricle by learning the dynamics in cardiac images. In *IPMI 2005*, pages 553–565, 2005.

[104] G. Sundaramoorthi and A. Yezzi. More-than-topology-preserving flows for active contours and polygons. In *ICCV 2005*, volume 2, pages 1276–1283, 2005.

[105] G. Sundaramoorthi, A. Yezzi, and A. Mennucci. Sobolev active contours. *Intl. J. Comp. Vis.*, 73(3):345–366, 2007.

[106] A. Tannenbaum. Three snippets of curve evolution theory in computer vision. *Math. and Comp. Modelling J.*, 24:103–119, 1996.

[107] Z. Tao and H. Tagare. Stopping rules for active contour segmentation of ultrasound cardiac images. *Medical Imaging*, 5747:475–484, 2005.

[108] W. M. Telford, L. P. Geldart, and R. E. Sheriff. *Applied Geophysics*. Cambridge University Press, 1990.

[109] C. M. C. Tempany, editor. *The Male Pelvis*. Magnetic Resonance Imaging Clinics of North America. W.B. Saunders Co., 1996.

[110] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[111] L. Tierney. Markov chains for exploring posterior distributions. Technical Report 560, School of Stat., Univ. of Minn., 1991.

[112] A. Tikhonov and V. Arsenin. *Solution of Ill-Posed Problems*. W.H. Winston, Washington, D.C., 1977.

[113] A. Tsai. *Curve Evolution and Estimation-Theoretic Techniques for Image Processing*. PhD thesis, MIT, 2000.

[114] A. Tsai, W. Wells, C. Tempany, E. Grimson, and A. Willsky. Mutual information in coupled multi-shape model for medical image segmentation. *Medical Image Analysis*, 8(4):429–445, 2004.

[115] A. Tsai, A. Yezzi, W. M. Wells III, C. Tempany, D. Tucker, A. Fan, W. E. L. Grimson, and A. S. Willsky. Model-based curve evolution technique for image segmentation. In *IEEE Conf. on Comp. Vision and Patt. Recog.*, volume I, pages 463–468. IEEE, 2001.

[116] A. Tsai, A. Yezzi, W. M. Wells III, C. Tempany, D. Tucker, A. Fan, W. E. L. Grimson, and A. S. Willsky. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE Trans. Med. Imag.*, 22(2):137–154, Feb. 2003.

[117] A. Tsai, A. Yezzi, and A. Willsky. Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification. *IEEE Trans. Imag. Proc.*, 10(8):1169–1186, 2001.

[118] Z. Tu and S. C. Zhu. Image segmentation by data-driven Markov chain Monte Carlo. *IEEE Trans. Patt. Anal. Mach. Intell.*, 24(5):657–673, 2002.

[119] S. Ullman. *High-Level Vision: Object Recognition and Visual Cognition*. MIT Press, 2000.

[120] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, 1999.

[121] L. Wang. Modeling complex reservoir geometries with multiple-point statistics. *Math. Geol.*, 28(7):895, 1996.

[122] W. M. Wells III, W. E. L. Grimson, R. Kikinis, and F. A. Jolesz. Adaptive segmentation of MR data. *IEEE Trans. Med. Imag.*, 15:429–442, 1996.

[123] T. Z. Wong, S. G. Silverman, J. R. Fielding, et al. Open-configuration MR imaging, intervention, and surgery of the urinary tract. *Urologic Clinics of N. Amer.*, 25:113–122, 1998.

[124] J. Yang and J. Duncan. Neighbor-constrained segmentation with 3d deformable models. In *Proceedings of IPMI*, pages 198–209, 2003.

[125] J. C. Ye, Y. Bresler, and P. Moulin. Asymptotic global confidence regions in parametric shape estimation problems. *IEEE Trans. Inf. Theory*, 46(5):1881–1895, 2000.

[126] A. Yezzi, A. Tsai, and A. Willsky. A statistical approach to snakes for bimodal and trimodal imagery. In *Seventh IEEE International Conference on Computer Vision*, 1999.

[127] A. Yezzi, A. Tsai, and A. Willsky. A fully global approach to image segmentation via coupled curve evolution equations. *J. Vis. Comm. and Image Repr.*, 13(1/2):195–216, 2002.

[128] Y. Zhang, A. O. Hero, and W. L. Rogers. Simultaneous confidence intervals for image reconstruction problems. In *Proceedings of ICASSP*, pages 317–320, 1994.

[129] S. C. Zhu. Embedding Gestalt laws in Markov random fields a theory for shape modeling and perceptual organization. *IEEE Trans. Patt. Anal. Mach. Intell.*, 21(11):1170–1187, 1999.

[130] S. C. Zhu and A. L. Yuille. Region competition: unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Trans. Patt. Anal. Mach. Intell.*, 18(9):884–900, 1996.