Topics in large-scale statistical inference

by

Jeffrey Carroll Regier

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Statistics

and the Designated Emphasis

in

Communication, Computation, and Statistics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jon D. McAuliffe, Chair
Professor Philip B. Stark
Professor James W. Pitman
Professor Thomas L. Griffiths

Summer 2016

Topics in large-scale statistical inference

Abstract

Topics in large-scale statistical inference

by

Jeffrey Carroll Regier

Doctor of Philosophy in Statistics
and the Designated Emphasis in
Communication, Computation, and Statistics

University of California, Berkeley

Professor Jon D. McAuliffe, Chair

Statistical inference may be large-scale in terms of the size of the dataset, the dimension of the data, or the amount of data needed for provably accurate inference. This dissertation presents three applications of large-scale statistical inference. Part I considers finding and characterizing stars and galaxies in images from telescopes. Part II considers figuring out who wrote what in large collection of articles, where authors often do not have unique names. Part III considers approximating a high-dimensional function based on a small number of observations, a common problem when interpreting computer experiments.

Contents

List of Figures

List of Tables

Acknowledgments

I am grateful to Jon McAuliffe, Philip Stark, and Jim Pitman for mentoring me during my Ph.D.

Jon has been an outstanding Ph.D. advisor to me. By emulating Jon, I learned the practices and mindset necessary to succeed at research. Jon was supportive and nurturing throughout. He let my research interests guide our work, and fostered my interest in developing machine learning theory in the service of real applications. In addition to the quality of the mentoring, the sheer quantity of time Jon spent with me distinguished my Ph.D. experience from that of most students. Our work together appears in Part I and Part II.

Philip effectively co-advised me for multiple years. He impressed upon me the importance of challenging conventional wisdom that is based on unstated assumptions. I aspire to emulate the clarity of Philip's writing, and the clarity of his thinking about applied statistics. Our research together appears in Part III.

Jim involved me in his bibliographic research while I was still a masters student. He oversaw my transition from software engineer to statistics researcher. Without his guidance early on, it is unlikely that my academic career would have progressed as seamlessly as it did. Some of our work together is the basis for Chapter 4.

Additionally, I am grateful to my family and my friends, many with ties to the Hillegass-Parker House co-op, for their support and friendship during my doctoral work.

Chapter 1

Introduction

For large datasets, asymptotic computational complexity [1] determines whether a statistical inference procedure is feasible. These datasets may be large in terms of the number of observations, the dimension of the data, or any other statistic affecting the runtime. Procedures with nearly linear runtimes can be scaled to essentially arbitrary-sized datasets: the inference procedure requires the same magnitude of computation as collecting the data. Conversely, procedures with significantly super-linear runtimes cannot be applied to large datasets, regardless of the engineering effort devoted to the problem.

For many inference problems, direct solutions require either an exponential or a theoretically infinite number of steps. Procedures that iterate through all possible configurations of discrete variables, or all subsets of a set, for example, will suffer from a "combinatorial explosion," leading to at least exponential runtime. Procedures that consider each setting of a continuous variable will never terminate.

This dissertation presents three applications of statistical inference where direct solutions all require either exponential or infinite runtime.

Part I considers finding and characterizing astronomical objects in images from telescopes [2]. A subproblem is to determine whether each object $s = 1, \dots, S$ is a star or a galaxy. A tractable algorithm cannot evaluate a loss function at each of the 2^S possible configurations. If the loss function could be decomposed across astronomical objects, then just $2S$ evaluations would be needed. Yet neighboring astronomical objects often overlap in the images, so a good loss function will not typically decompose across them: knowledge about some astronomical objects helps to resolve others.

Part II applies statistical inference to author disambiguation—the task of figuring out which author is referenced by each mention of a person's name [3]. Author disambiguation matters because in a large collection of academic publications many authors' names are not unique. For instance, a collection may contain hundreds of authors named "J. Smith." Moreover, a single author may be referred to in multiple ways, due to misspellings, name changes, or spelling variations. Each mention of a person's name is accompanied by several attributes in addition to the name string itself, that can aid in disambiguation. These attributes include articles' keywords, articles' topics (explicit or inferred), and the name strings of co-authors appearing on the same article. A direct solution might consider every possible partition of the mentions. Another approach, common in practice, compares every pair of mentions, thus requiring quadratic runtime. But for sizable datasets, even quadratic runtime is intractable.

Part III considers approximating a real-valued, high-dimensional function based on a small number of observations, with minimal prior knowledge (real knowledge, not assumptions) about the function. This situation is commonplace when interpreting computer experiments [4]. A computationally expensive simulator maps a particular setting of unknown model parameters to real-valued output. A priori, little is known about the simulator, but some degree of smoothness is expected: small changes in model parameters yield small changes in output. The curse of dimensionality, however, suggests that, even with this regularity condition, an exponential number of observations are required to achieve adequate sample density [5]. Yet the curse of dimensionality is only a rule of thumb, not a theorem. Indeed, much analysis of computer experiments depends on it not applying. For a real dataset, from a climate model, we show that the number of additional observations that could be necessary to approximate the function with a useful degree of accuracy is intractable. This problem is large-scale not in terms of the data available, but rather in terms of the amount of data that would be required.

The problems of constructing an astronomical catalog (Part I) and disambiguating authors (Part II) have a fundamental similarity. In astronomical cataloging, the data are the number of photons landing in each pixel. Each photon originates from a single astronomical object. Partitioning the photons according to origin essentially solves the cataloging problem. Similarly, author disambiguation is partitioning the mentions according to author. In both cases, we partition large N observations (i.e., photons or mentions) into $O(N)$ parts (i.e., astronomical objects or authors).[1] There is a collective aspect to both problems too, where the information about one subset of the observations informs partitioning of a disjoint subset. For mentions, for example, a shared co-author can indicate that the mentions are coreferent. For photons, the assignment of neighboring photons to the same astronomical object suggests that these photons too are "coreferent."

That said, in astronomical cataloging, it is the catalog that is the final output, not a partition of photons. If multiple photons from different astronomical objects are recorded by the same pixel, there is no basis for distinguishing which photon came from which astronomical object, nor is there interest in knowing. In author disambiguation, it may be that an authoritative list of authors is important, rather than a partition of mentions. More likely, however, it is the partition that is desired.

Another unifying theme among the parts, with this one uniting all three, has to do with the relation between the solution proposed and the past standard practice. In the past, each of these problems has typically been approached with heuristics—algorithms that may be intuitively reasonable, but that do not follow from an explicit model of the data. Thus, their conclusions cannot be stated as following from interpretable assumptions. They

[1]Even clustering algorithms called "scalable" typically are only efficient at recovering $O(1)$ parts.

do not generally provide calibrated measures of the uncertainty of their predictions. They are difficult to extend to accommodate heterogeneous sources of data. They may make suboptimal predictions because they do not fully exploit prior knowledge.

In astronomy (Part I), the current standard practice for building a catalog of detected astronomical objects is to process the data through a "pipeline" of software routines [6]. One stage of the pipeline estimates the level of background noise, another finds bright spots in the images, and yet another classifies each bright spot as a star or a galaxy.

For author disambiguation (Part II), the current standard practice is agglomerative clustering, a greedy heuristic that does not provide uncertainty estimates. Worse still, every pair of articles needs to be compared, leading to quadratic runtime unless yet another heuristic, called "blocking," is used too. This multi-stage, algorithmic approach cannot provide interpretable uncertainty estimates. The conclusions do not follow from an interpretable model.

For approximating a simulator (Part III), standard practice is to interpolate the known output of the simulator with a computationally inexpensive "emulator," denoted $\hat{f}$. Kriging, MARS, and Polynomial Chaos are some common methods [7, 8, 9]. The belief that f is "well-behaved" or "smooth" is often said to justify letting the emulator stand in for f. However, this belief does not follow from any explicit and plausible assumptions about the data. Hence, procedures stemming from it are heuristics.

The alternative to heuristics is procedures motivated by explicit and plausible models of data. In Chapter 2, we propose a comprehensive statistical approach to astronomical object curation, represented as a graphical model. Fundamentally, the model assumes that the night sky has an underlying brightness that does not change on a human time scale, but that the number of photons detected at any particular pixel is a Poisson random variable. Each pixel has a unique rate, a complex but deterministic function of latent random variables and unknown model parameters, including each celestial object's position, brightness, color, and shape. The model also accounts for many details: the sky is organized into "fields"; images are taken through "filters" that selectively allow photons of certain wavelengths to pass; light is spread by Earth's atmosphere (the "point spread function"); and more.

Learning a catalog of astronomical objects amounts to conditioning on the observed pixel intensities and inferring the distribution of the latent random variables—the posterior distribution. Bayes's rule cannot be directly applied because the marginal likelihood of the data is not tractable to compute. Markov Chain Monte Carlo (MCMC), a common alternative, approximates the posterior by sampling [10]. The computational cost of MCMC, however, can be prohibitively high for large datasets. Variational inference, an alternative to MCMC, approximates the posterior through numerical optimization [11]. Typically, the approximation is constrained to a designated class of analytically tractable distributions, indexed by real-valued parameters. From this class, variational inference finds the distribution that most closely approximates the posterior in terms of Kullback-Leibler divergence.

We derive scalable variational inference procedures for modeling astronomical images. These procedures leverage conditional conjugacy and the delta method for moments to eliminate non-analytic integrals. To avoid modeling simplifications that would misfit the data, these procedures break with tradition in the variational inference community by using iterative second-order optimization methods rather than closed-form coordinate ascent updates.

In Chapter 3, we derive a flexible generative model for astronomical images of galaxies that melds neural networks and variational inference. For a particular galaxy image, let z be a low-dimensional latent random vector, distributed as a multivariate standard normal. Then the intensities of the pixels—that is, the observed random variables—conditionally follow a multivariate normal distribution too:

$$x|z \sim \mathcal{N}\left(f_\mu\left(z\right), f_\sigma\left(z\right)\right).$$

Here f_μ and f_σ are deep neural networks—deterministic non-linear functions that map the random vector z to the distributional parameters for x. Inference for this model involves 1) maximizing the likelihood of the data with respect to the model parameters—the weights of neural networks f_μ and f_σ; and 2) for the maximizers f_μ and f_σ, finding the approximating distribution closest to the posterior, that is, the conditional distribution of z given x. We restrict the candidate approximations to the class of distributions with the form $\mathcal{N}\left(g_\mu\left(x\right), g_\sigma\left(x\right)\right)$, where g_μ and g_σ are neural networks too, with a fixed architecture. Then variational inference amounts to finding the weights of g_μ and g_σ that minimize the Kullback-Leibler divergence between $\mathcal{N}\left(g_\mu\left(x\right), g_\sigma\left(x\right)\right)$ and the posterior.

In Chapter 5 we develop an approach to author disambiguation based on discriminative, undirected graphical models, known as conditional random fields. We model the probability of a partition Y of the mentions as

$$P(Y) \propto \prod_{y_i \in Y} \exp\left(\theta_0 + \sum_{j=1}^{J} \theta_j f_j\left(y_i\right)\right),$$

where the f_j are feature functions, the y_i are parts of a partition, and θ is the model's parameters. A Metropolis-Hastings sampler with split-merge proposals infers the posterior on Y. Contrastive divergence learns parameters θ from manually disambiguated training data. Developing feature functions such that the maximum-marginal-likelihood estimate $\hat{\theta}$ also induces good partitions, however, remains an open problem.

In Chapter 6, we develop an approach to author disambiguation based on spectral methods [12]. Learning similarity functions from training data for K-way spectral clustering is a well-studied problem [13]. For author disambiguation, however, K is the number of authors, which scales linearly in the number of articles, implying cubic runtime. Recursive spectral bipartitioning is a fast alternative, terminating after $O\left(\log n\right)$ bipartitions. Each bipartition uses the graph Laplacian to approximate the min-cut objective function. The similarity matrix, and hence the Laplacian, is neither sparse, since many articles have some similarity to each other, nor low rank, since we must recover $O(n)$ unique authors. Representing the similarity matrix as a product of sparse matrices enables an $O(n)$ right-multiplication of the Laplacian, and hence roughly $O(n)$ bipartitioning by iterative eigensolvers. Learning edge weights from training data for recursive bipartitioning, however, remains an open problem.

In Part III, we consider a function f that has been observed n times. We know only that f is Lipschitz continuous. This setting is common when f is a simulator, mapping settings to some quantity of interest, and each computer experiment constitutes one observation of f. In Chapter 7, we propose a procedure for sequentially selecting locations to observe f, in order to determine where in the domain f is above or below a particular threshold. While the procedure works well for low-dimensional functions, outperforming several baseline measures, its good performance does not carry over to high-dimensional functions. Intuitively, in high dimensions, Lipschitz continuity is not a strong enough condition to learn about f globally from only a modest number of local observations.

In Chapter 8, we make this intuition rigorous, through minimax analysis. We suppose f has the smallest Lipschitz constant $\hat{K}$ consistent with a fixed set of observations. (The function f must vary somewhat to interpolate the n observations.) How many additional observations might be required to "pin down" f everywhere to within some tolerance ϵ? Suppose $\hat{f}$ is the best possible emulator of f based on the observations—it minimizes worst-case error at every point of the domain. Given the n observations, how large could $\left|f(x) - \hat{f}(x)\right|$ be for some unobserved input x? We propose statistical tests, that, from a batch of observations, can determine how many additional observations could be needed, given the observed smoothness of the function, based on a minimax analysis. Applying the theoretical bounds derived in Chapter 8 to climate data demonstrates that, even with these optimistic assumptions, the number of observations required to emulate f well is intractable: tens of orders of magnitude more simulations than any supercomputer could run. Based on the data, the smoothness of f does not justify any emulator. Without stronger assumptions, any emulator is only a heuristic.

Part I

Astronomy

Chapter 2

Variational inference for a generative model of astronomical images

This chapter presents Celeste, a new, fully generative model of astronomical image sets—the first such model to be empirically investigated, to our knowledge. The work we report is an encouraging example of principled statistical inference applied successfully to a science domain underserved by the machine learning community. It is unfortunate that astronomy and cosmology receive comparatively little of our attention: the scientific questions are fundamental, there are petabytes of data available, and we as a data-analysis community have a lot to offer the domain scientists. One goal in reporting this work is to raise the profile of these problems for the machine-learning audience and show that much interesting research remains to be done.

Turn now to the science. Stars and galaxies radiate photons. An astronomical image records photons—each originating from a particular celestial body or from background atmospheric noise—that pass through a telescope's lens during an exposure. Multiple celestial bodies may contribute photons to a single image (e.g., Figure 2.1), and even to a single pixel of an image. Locating and characterizing the imaged celestial bodies is an inference problem central to astronomy. To date, the algorithms proposed for this inference problem have been primarily heuristic, based on finding bright regions in the images [6, 14].

Generative models are well-suited to this problem—for three reasons. First, to a good approximation, photon counts from celestial objects are independent Poisson processes: each star or galaxy has an intrinsic brightness that is effectively static during human time scales. In an imaging exposure, the expected count of photons entering the telescope's lens from a particular object is proportional to its brightness. When multiple objects contribute photons to the same pixel, their rates combine additively.

Second, many sources of prior information about celestial bodies are available, but none is definitive. Stars tend to be brighter than galaxies, but many stars are dim and many galaxies are bright. Stars tend to be smaller than galaxies, but many galaxies appear point-like as well. Stars and galaxies differ greatly in how their radiation is distributed over the visible spectrum: stars are well approximated by an "ideal blackbody law" depending only

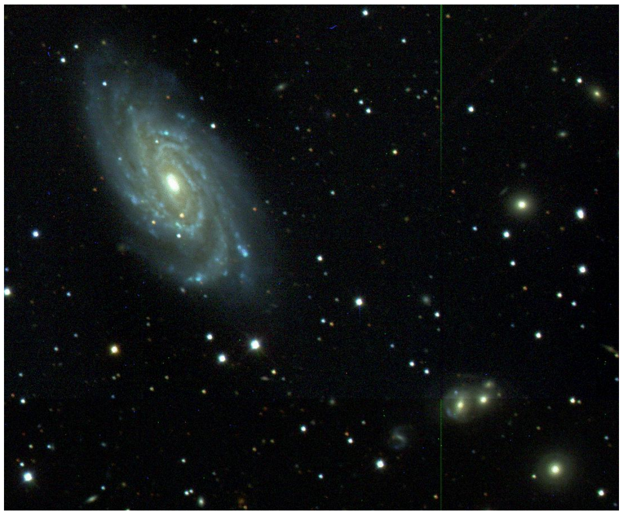


Figure 2.1: An image from the Sloan Digital Sky Survey [15] of a galaxy from the constellation Serpens, 100 million light years from Earth, along with several other galaxies and many stars from our own galaxy.

on their temperature, while galaxies are not. On the other hand, stars are not actually ideal blackbodies, and galaxies do emit energy in the same wavelengths as stars. Posterior inference in a generative model provides a principled way to integrate these various sources of prior information.

Third, even the most powerful telescopes receive just a handful of photons per exposure from many celestial objects. Hence, many objects cannot be precisely located, classified, or otherwise characterized from the data available. Quantifying the uncertainty of point estimates is essential—it is often as important as the accuracy of the point estimates themselves. Uncertainty quantification is a natural strength of the generative modeling framework.

Some astronomical software uses probabilities in a heuristic fashion [16], and a generative model has been developed for measuring galaxy shapes [17]—a subproblem of ours. But, to our knowledge, fully generative models for inferring celestial bodies' locations and characteristics have not yet been examined[1]. Difficulty scaling the inference for expressive generative models may have hampered their development, as astronomical sky surveys produce very large amounts of data. For example, the Dark Energy Survey's 570-megapixel digital camera, mounted on a four-meter telescope in the Andes, captures 300 gigabytes of sky images every night [19]. Once completed, the Large Synoptic Survey Telescope will house a 3200-megapixel camera producing eight terabytes of images nightly [20].

The remainder of this paper describes the Celeste model (Section 2.1) and its accompanying variational inference procedure (Section 2.2). Section 4 details our empirical studies on synthetic data as well as a sizable collection of astronomical images.

[1]However, see [18] for a workshop presentation proposing such a model.

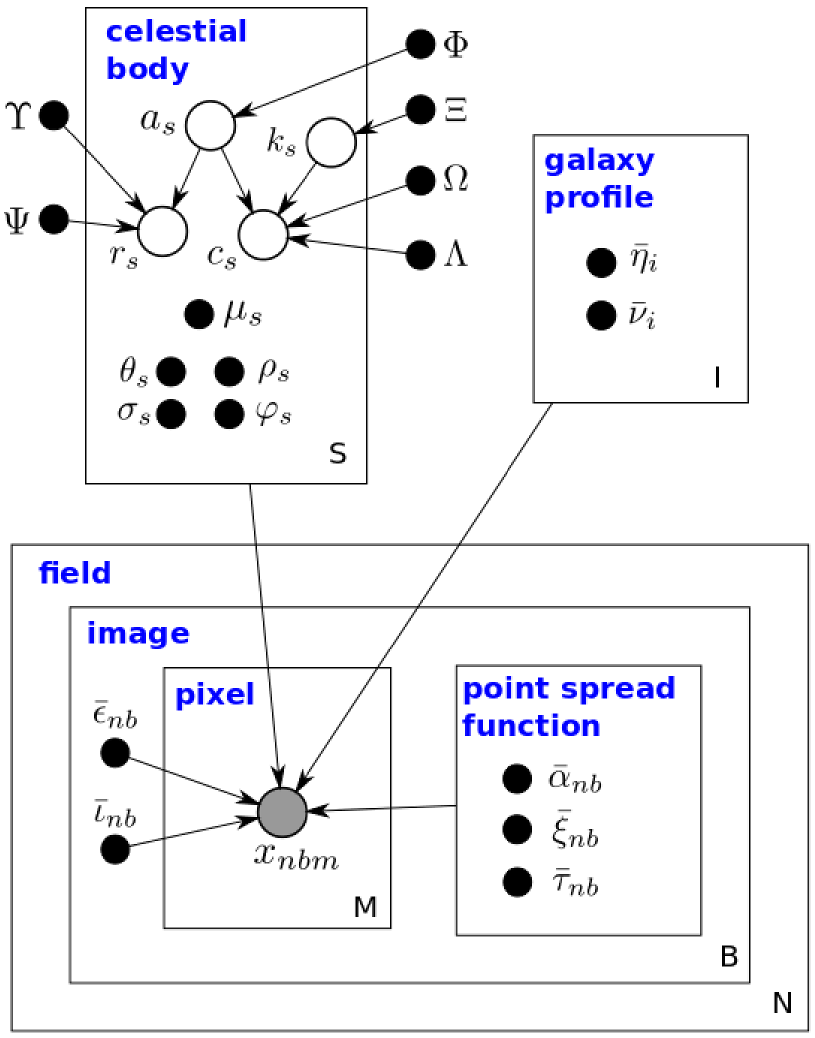


Figure 2.2: The Celeste graphical model. Shaded vertices represent observed random variables. Empty vertices represent latent random variables. Black dots represent constants. Constants with "bar" decorators, e.g. $\bar{\epsilon}_{nb}$, are set a priori. Constants denoted by uppercase Greek characters are also fixed; they denote parameters of prior distributions. The remaining constants and all latent random variables are inferred. Edges signify conditional dependency. Rectangles ("plates") represent independent replication.

2.1 The model

The Celeste model is represented graphically in Figure 2.2. In this section we describe how Celeste relates celestial bodies' latent characteristics to the observed pixel intensities in each image.

2.1.1 Celestial bodies

Celeste is a hierarchical model, with celestial objects atop pixels. For each object $s = 1, \ldots, S$, the unknown 2-vector μ_s encodes its position in the sky as seen from Earth. In Celeste, every celestial body is either a star or a galaxy. (In the present work, we ignore other types of objects, which are comparatively rare.) The latent Bernoulli random variable a_s encodes object type: $a_s = 1$ for a galaxy, $a_s = 0$ for a star. We set the prior distribution

$$a_s \sim \text{Bernoulli}(\Phi). \tag{2.1}$$

2.1.1.1 Brightness

The overall brightness of a celestial object s is quantified as the total radiation from s expected to reach a unit area of Earth's surface, directly facing s, per unit of time. However, we can also quantify brightness as the proportion of this radiation (per square meter, per second) that passes through each filter in a standardized filter set. Such a set is called a "photometric system." These standardized filters are approximately band-pass: each allows most of the energy in a certain band of wavelengths through, while blocking most of the energy outside the band. The physical filters attached to a telescope lens closely match the standardized filters of some photometric system.

In Celeste, we take the photometric-system approach—we directly model brightnesses with respect to the B filters of a fixed photometric system. We designate a particular filter as the "reference" filter, letting the random variable r_s denote the brightness of object s with respect to that filter. We make r_s dependent on a_s, since stars tend to be brighter than galaxies. For computational convenience, and because brightness is typically considered to be non-negative and real-valued, we set

$$r_s | (a_s = i) \sim \text{Gamma}\left(\Upsilon^{(i)}, \Psi^{(i)}\right). \tag{2.2}$$

Object s's brightnesses with respect to the remaining $B - 1$ filters are encoded using "colors." The color c_{sb} is defined as the log ratio of brightnesses with respect to filters b and $b + 1$. Here, the filters are ordered by the wavelength bands they let through. The $B - 1$ colors for object s are collectively denoted by c_s, a random $(B - 1)$-vector.

Celeste uses the color parameterization because stars and galaxies have very distinct prior distributions in color space. Indeed, for idealized stars—blackbodies—all $B - 1$ colors lie on a one-dimensional manifold indexed by surface temperature. On the other hand, though galaxies are composed of stars, theory does not suggest they lie near the same manifold: the stars in a galaxy can have many different surface temperatures, and some of the photons are re-processed to other energies through interactions with dust and gas. Figure 2.3 demonstrates that stars are much nearer a 1-dimensional manifold in color space than galaxies are.

We model the prior distribution on c_s as a mixture of D multivariate Gaussians. The number of mixture components D may be selected to minimize error on held out data, or

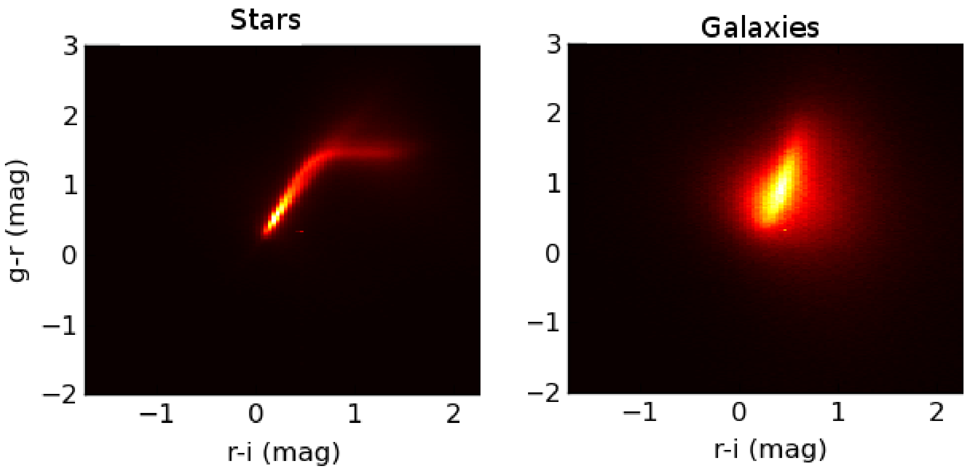


Figure 2.3: Density plots for two colors, based on an SDSS catalog containing hundreds of thousands of stars and galaxies.

kept small for computational efficiency. The random categorical variable k_s indicates which mixture component generated c_s. A priori,

$$k_s | (a_s = i) \sim \text{Categorical}\left(\Xi_1, \dots, \Xi_D\right) \tag{2.3}$$

and

$$c_s | (k_s = d, a_s = i) \sim \text{MvNormal}\left(\Omega^{(i,d)}, \Lambda^{(i,d)}\right). \tag{2.4}$$

A celestial body's brightnesses in the B filters, $(\ell_{sb})_{b=1}^{B}$, is uniquely specified by its reference-filter brightness r_s and its colors c_s.

2.1.1.2 Galaxies

The distance from Earth to any star (besides the Sun) exceeds the star's radius by many orders of magnitude. Therefore, stars are well modeled as points. Modeling the (two-dimensional) appearance of galaxies as seen from Earth is more involved. We divide the appearance of galaxy s into two parts: its per-filter brightnesses (ℓ_{sb}), as discussed in Section 2.1.1.1, and its "light kernel" $h_s(w)$, which describes how the apparent radiation from the galaxy is distributed over the sky. The argument w is in sky coordinates; the light kernel is a density function that integrates to one and is largest near the apparent galactic center. In Section 2.1.1.3, we show how these two parts of apparent galaxy brightness come together.

We take $h_s(w)$ to be a convex combination of two prototype functions, known in astronomy as the "exponential" and "de Vaucouleurs" prototypes:

$$h_s(w) = \theta_s h_{s1}(w) + (1 - \theta_s) h_{s2}(w), \quad \theta_s \in [0, 1]. \tag{2.5}$$

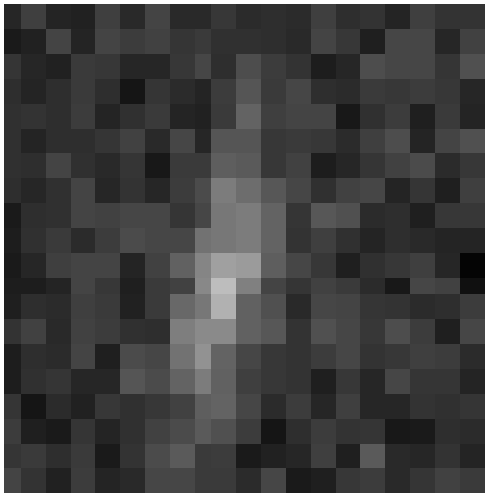


Figure 2.4: A distant galaxy, from the SDSS dataset, approximately 20 pixels in height, predicted to have effective radius $\sigma_s = 0.6$ arcseconds, rotation $\varphi_s = 80°$, and eccentricity $\rho_s = 0.17$. Credit: SDSS

The de Vaucouleurs prototype is characteristic of "elliptical" galaxies, which have smooth light kernels (Figure 2.5), whereas the exponential prototype matches "spiral" galaxies (Figure 2.1). The prototype functions $h_{s1}(w)$ and $h_{s2}(w)$ contain additional galaxy-specific parameters. In particular, each prototype function is a rotated, scaled mixture of bivariate normal distributions. The rotation and scaling are galaxy-specific, but the remaining parameters of each mixture are not:

$$h_{si}(w) = \sum_{j=1}^{J} \bar{\eta}_{ij} \phi(w; \mu_s, \bar{\nu}_{ij} W_s), \quad i = 1 \text{ or } 2. \tag{2.6}$$

In Equation (2.6), $(\bar{\eta}, \bar{\nu})_{ij}$ are pre-specified constants that characterize the exponential and de Vaucouleurs prototypes; μ_s is the center of the galaxy, in sky coordinates; W_s is a spatial covariance matrix; and ϕ is the bivariate normal density.

The light kernel $h_s(w)$ is a finite scale mixture of Gaussians: its mixture components have a common mean μ_s and covariance matrices that differ only in scale. The "isophotes" (level sets of the light kernel) are concentric ellipses. Although this model prevents us from fitting individual "arms," like those of the galaxy in Figure 2.1, most galaxies are not sufficiently resolved to see such sub-structure. (See Figure 2.4 for a typical galaxy image.) A more flexible galaxy model might overfit them.

The spatial covariance matrix W_s is parameterized by a rotation angle φ_s, an eccentricity (minor-major axis ratio) ρ_s, and an overall size scale σ_s:

$$W_s = R_s^\intercal \begin{bmatrix} \sigma_s^2 & 0 \\ 0 & \sigma_s^2 \rho_s^2 \end{bmatrix} R_s, \tag{2.7}$$

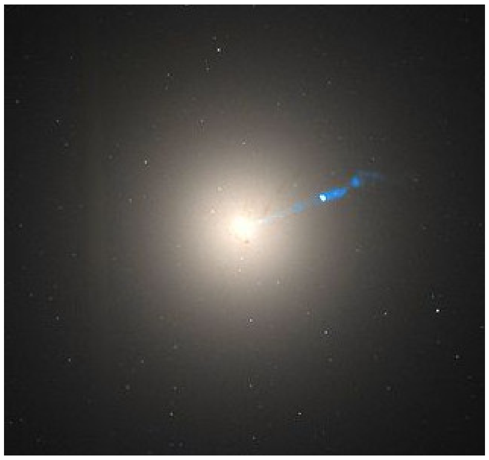


Figure 2.5: Messier 87, a galaxy that exhibits the de Vaucouleurs profile. Credit: NASA

where R_s is a rotation matrix,

$$R_s = \begin{bmatrix} \cos\varphi_s & -\sin\varphi_s \\ \sin\varphi_s & \cos\varphi_s \end{bmatrix}. \tag{2.8}$$

The scale σ_s is specified in terms of "effective radius"—the radius of the disc that contains half of the galaxy's light emissions before applying the eccentricity ρ_s.

2.1.1.3 The ideal sky view

In the upcoming Section 2.1.2, we account for distortions from pixelation, atmospheric blur, and background noise; and we deal in photons counted by a camera, rather than continuous-valued brightness (energy arriving at Earth). The developments of the previous sections represent the sky without these concerns; we call this part of the model the "ideal sky view." Let δ_{μ_s} denote the Dirac delta function—the light kernel of a star. Then the total apparent brightness (the ideal sky view) in filter b, at sky position w, is

$$G_b(w) = \sum_{s=1}^{S} \ell_{sb} g_{sa_s}(w), \tag{2.9}$$

where

$$g_{si}(w) = \begin{cases} \delta_{\mu_s}(w), & \text{if } i = 0 \text{ (“star")} \\ h_s(w), & \text{if } i = 1 \text{ (“galaxy”).} \end{cases} \tag{2.10}$$

2.1.2 Astronomical images

Returning to the Celeste graphical model, the data is represented in the bottom half of Figure 2.2. A "field" is a small, rectangular region of the sky. Fields may overlap. Each of the N fields in the data set is imaged B times, once per filter in the photometric system (Section 2.1.1.1).[2]

Each resulting image is a grid of M pixels. Each pixel in turn receives light primarily from celestial bodies near the pixel's corresponding region of the sky. The observed random variable x_{nbm} is the count of photons recorded at pixel m, in image b of field n.

The night sky is not completely dark owing to natural skyglow, a combination of reflected sunlight off dust particles in the solar system, night airglow from molecules in Earth's atmosphere, and scattered starlight and moonlight. We model the night sky's brightness as background noise, through a spatial Poisson process that is homogeneous for each image and independent of stars and galaxies. The noise rate depends on the image and the band because atmospheric conditions vary over time. Also, the atmospheric effects differ for imaging targets closer to the horizon. The image-specific constant $\bar{\epsilon}_{nb}$ denotes the rate of background noise.

Both $\bar{\epsilon}_{nb}$ and the brightnesses of celestial bodies are quantified in linear units of nanomaggies; nanomaggies are a physical unit of energy, not specific to any particular image [21]. The image-specific constant $\bar{\iota}_{nb}$ is the expected number of photons recorded in image b of field n, for a pixel receiving a brightness of one nanomaggy.

2.1.2.1 Point-spread functions

Ground-based astronomical images are blurred by a combination of small-angle scattering in Earth's atmosphere, the diffraction limit of the telescope, optical distortions in the camera, and charge diffusion within the silicon of the CCD detectors. Together these effects are represented by the "point spread function" (PSF) of a given image. Stars (other than the Sun) are points in the ideal sky view (Section 2.1.1.3), but the PSF typically spreads their photons over dozens of adjacent pixels.

When dealing with images, as in this section, we work in the image coordinate system. For any single image, there is a one-to-one mapping between image and sky coordinates, so nothing is lost; our notation suppresses the mapping for clarity.

We model the action of the PSF as a mixture of K Gaussians. Consider pixel m (in band b of image n), having coordinates w_m. The probability that a photon originating at coordinates w lands at w_m is given by the PSF

$$f_{nbm}(w) = \sum_{k=1}^{K} \bar{\alpha}_{nbk} \phi(w_m; w + \bar{\xi}_{nbk}, \bar{\tau}_{nbk}). \tag{2.11}$$

Here ϕ is the bivariate normal density. The parameters $(\bar{\alpha}_{nb}, \bar{\xi}_{nb}, \bar{\tau}_{nb})$ of the the PSF are specific to an image, in part to account for atmospheric conditions that vary between exposures, but are constant throughout the image.

[2] Cameras for optical astronomy typically use charge-coupled devices (CCDs), which convert light into electrons [19]. A CCD is a grid of millions of pixels, designed to accumulate the radiation arriving at each pixel during an exposure.

2.1.2.2 The Celeste likelihood

Convolving the ideal sky view (Equation 2.9) with the PSF and adding background noise yields the rate of photon arrivals for pixel m:

$$z_{nbm} := \epsilon_{nb} + \int G_b(w) f_{nbm}(w) dw \tag{2.12}$$

$$= \epsilon_{nb} + \sum_{s=1}^{S} \ell_{sb} \int g_{sa_s}(w) f_{nbm}(w) dw. \tag{2.13}$$

These normal-normal convolutions can be computed analytically. For stars ($a_s = 0$),

$$\breve{f}_{s0}(m) := \int g_{s0}(w) f_{nbm}(w) dw \tag{2.14}$$

$$= \sum_{k=1}^{K} \bar{\alpha}_{nbk} \phi(w_m; \mu_s + \bar{\xi}_{nbk}, \bar{\tau}_{nbk}). \tag{2.15}$$

Let $\theta_{s1} = \theta_s$ and $\theta_{s2} = 1 - \theta_s$. For galaxies ($a_s = 1$),

$$\breve{f}_{s1}(m) := \int g_{s1}(w) f_{nbm}(w) dw \tag{2.16}$$

$$= \sum_{k=1}^{K} \bar{\alpha}_{nbk} \sum_{i=1}^{2} \theta_{si} \sum_{j=1}^{J} \bar{\eta}_{ij} \cdot \phi(w_m; \mu_s + \bar{\xi}_{nbk}, \bar{\tau}_{nbk} + \bar{\nu}_{ij} W_s). \tag{2.17}$$

Let $a = (a_s)_{s=1}^S$, $r = (r_s)_{s=1}^S$, and $c = (c_s)_{s=1}^S$. Then the expected number of photons landing in pixel m is

$$F_{nbm}(a, r, c) = \bar{\iota}_{nb}[\epsilon_{nb} + z_{nbm}]. \tag{2.18}$$

For $n = 1, \ldots, N$, $b = 1, \ldots, B$, and $m = 1, \ldots, M$, we finally obtain the likelihood

$$x_{nbm} | a, r, c \overset{\text{ind}}{\sim} \text{Poisson}(F_{nbm}(a, r, c)). \tag{2.19}$$

2.2 Inference

In this section we explain how we apply the Celeste model to astronomical image data sets, to draw inferences about unknown quantities.

In principle, all parameters could be learned by variational inference. But we reuse some estimates from the existing photometric pipeline that are not thought to limit performance. The background noise level $\bar{\epsilon}_{bn}$ is set by the existing photometric pipeline, based on a heuristic: most pixels in each image receive photons primarily from background radiation. The calibration constant $\bar{\iota}_{bn}$ is set by first calibrating overlapping images relative to each other, and then by calibrating some images absolutely, based on benchmark stars [22]. The image-specific parameters of the point spread function, $(\bar{\alpha}_{nb}, \bar{\eta}_{nb}, \bar{\tau}_{nb})_{k=1}^K$, are set by the existing photometric pipeline; a mixture of Gaussians is fit to known stars (considered point sources, before convolution with the PSF) in each image that are not near other celestial objects.

Galaxy profiles' parameters, $(\bar{\eta}_{ij}, \bar{\nu}_{ij})_{j=1}^{J}$, are set a priori too, by fitting mixtures of Gaussians to Sérsic profiles—widely used models of galaxy profiles. We fit $J = 8$-component mixtures for de Vaucouleurs galaxies ($i = 1$) and $J = 6$-component mixtures for exponential galaxies ($i = 2$). These approximations are good enough for the current version of the model: the misfit versus Sérsic profiles is smaller than the misfit between Sérsic profiles and actual galaxies.

The prior distributions also could be learned within the variational procedure, by empirical Bayes. But, because much data is available in existing astronomical catalogs, we estimate their parameters a priori. Fitting Φ, Υ, and Λ by maximum likelihood to existing catalogs is straightforward. To fit the prior on color (c_s) to existing catalogs, we use the expectation-maximization algorithm, initialized by k-means [23]. Though $D = 64$ minimized held-out test error, we set $D = 2$, to work around a limitation of our present optimizer—it only supports box constraints. Figure 2.3 shows a two-dimensional slice of the $(B-1)$-dimensional data set used to construct the color prior.

The remaining quantities are estimated by variational inference.

2.2.1 Variational approximation of the posterior distribution

Let $\Theta = (a, r, k, c)$ be the latent random variables in the Celeste model, and let

$$x = (x_{111}, \ldots, x_{NBM})$$

be the pixel intensities. Computing the posterior $p(\Theta|x)$ is intractable: to apply Bayes's rule exactly, we need to evaluate

$$p(x) = \int \prod_{n=1}^{N} \prod_{b=1}^{B} \prod_{m=1}^{M} p(x_{nbm}|\Theta) \prod_{s=1}^{S} p(\Theta_s)\, d\Theta. \tag{2.20}$$

But the S-dimensional integral in (2.20) does not decompose into a product of low-dimensional integrals, and therefore cannot be computed efficiently.

Instead, we use optimization to find a distribution that best approximates the posterior. For any distribution q over Θ,

$$\begin{aligned}
\log p(x) &= \log \int p(x, \Theta) d\Theta && (2.21)\\
&= \log \int p(x, \Theta) \frac{q(\Theta)}{q(\Theta)} d\Theta && (2.22)\\
&= \log \mathbb{E}_q \left[\frac{p(x, \Theta)}{q(\Theta)} \right] && (2.23)\\
&\geq \mathbb{E}_q \left[\log p(x|\Theta) \right] - D_{\mathrm{KL}} \left(q(\Theta), p(\Theta) \right) && (2.24)\\
&=: \mathcal{L}(q). && (2.25)
\end{aligned}$$

Here $\mathbb{E}_q$ is expectation with respect to q. We call $\mathcal{L}$ the evidence lower bound (ELBO). To find a distribution $q^\star$ that approximates the exact posterior, we maximize the ELBO over a set $\mathcal{Q}$ of candidate q's. We restrict $\mathcal{Q}$ to distributions of the factored form

$$q(\Theta) = \prod_{s=1}^{S} q(a_s) q(r_s|a_s) q(k_s|a_s) \prod_{b=1}^{B-1} q(c_{sb}|a_s). \tag{2.26}$$

Furthermore, for all $s = 1, \dots, S$, $b = 1, \dots B-1$, and $i \in \{0,1\}$, we set

$$q(a_s) \sim \text{Bernoulli}(\chi_s), \tag{2.27}$$
$$q(r_s|a_s = i) \sim \text{Gamma}(\gamma_s^{(i)}, \zeta_s^{(i)}), \tag{2.28}$$
$$q(k_s|a_s = i) \sim \text{Categorical}(\kappa_s^{(i)}), \tag{2.29}$$
$$q(c_{sb}|a_s = i) \sim \text{Normal}(\beta_{sb}^{(i)}, \lambda_{sb}^{(i)}). \tag{2.30}$$

Then finding $q^\star$ is equivalent to finding the optimal $(\chi_s, \gamma_s, \zeta_s, \kappa_s, \beta_s, \lambda_s)$ for each celestial body. By design, most of the expectations in $\mathcal{L}$ can be evaluated analytically. We proceed by decomposing each of the two expectation in equation (2.24).

2.2.1.1 Expected log likelihood

From the density function for the Poisson distribution,

$$\mathbb{E}_q[\log p(x|\Theta)] = \sum_{n=1}^{N} \sum_{b=1}^{B} \sum_{m=1}^{M} \Big\{ x_{nbm} \mathbb{E}_q[\log F_{nbm}] - \mathbb{E}_q[F_{nbm}] - \log(x_{nbm}!) \Big\}. \tag{2.31}$$

Here

$$\mathbb{E}_q\left[F_{nbm}\right] = \bar{\iota}_{nb} \mathbb{E}_q\left[G_{nbm}\right] \tag{2.32}$$

and

$$\mathbb{E}_q\left[G_{nbm}\right] = \epsilon_{nb} + \sum_{s=1}^{S} \left\{ (1-\chi_s) \mathbb{E}_q\left[\ell_s | a_s = 0\right] f_{s0}(m) + \chi_s \mathbb{E}_q\left[\ell_s | a_s = 1\right] f_{s1}(m) \right\} \tag{2.33}$$

and

$$\mathbb{E}_q\left[\ell_s | a_s = i\right] \tag{2.34}$$
$$= \mathbb{E}_q\left[r_s | a_s = i\right] \mathbb{E}_q\left[\prod_{j=1}^{B-1} \exp\left\{I_b(j)\, c_{sb}\right\} | a_s = i\right] \tag{2.35}$$
$$= \exp\Big\{\gamma_s^{(i)} + \zeta_s^{(i)}/2\Big\} \prod_{j=1}^{B-1} \mathbb{E}_q\Big[\exp\left\{I_b(j)\, c_{sb}\right\} \Big| a_s = i\Big], \tag{2.36}$$

and, for $\tau \in \mathbb{R}$,

$$\mathbb{E}_q\left[\exp\left\{\tau c_b\right\}|k_s = d, a_s = i\right] = \exp\left\{\tau\beta_{sb}^{(i)} + \frac{1}{2}\tau^2\lambda_{sb}^{(i)}\right\}. \tag{2.37}$$

We approximate $\mathbb{E}_q\left[\log F_{nbm}\right]$ by replacing the logarithm with its second-order Taylor expansion around $\mathbb{E}_q\left[F_{nbm}\right]$:

$$\log(x) \quad = \quad \log\mathbb{E}_q[x] + \frac{1}{\mathbb{E}_q[x]}\left(x - \mathbb{E}_q[x]\right) - \frac{1}{2\mathbb{E}_q[x]^2}\left(x - \mathbb{E}_q[x]\right)^2 + \ldots$$

Therefore,

$$\mathbb{E}_q\left[\log F_{nbm}\right] \approx \log\mathbb{E}_q\left[F_{nbm}\right] - \frac{\mathbb{V}_q\left[F_{nbm}\right]}{2\mathbb{E}_q\left[F_{nbm}\right]^2}, \tag{2.38}$$

where

$$\begin{aligned}\mathbb{V}_q\left[G_{nbm}\right] &= \sum_{s=1}^{S}\mathbb{V}_q\left[\ell_{sb}f_{sa_s}(m)\right] \\ &= \sum_{s=1}^{S}\mathbb{E}_q\left[\ell_{sb}^2 f_{sa_s}(m)^2\right] + \mathbb{E}_q\left[\ell_{sb}f_{sa_s}(m)\right]^2,\end{aligned} \tag{2.39}$$

and

$$\mathbb{E}_q\left[\ell_{sb}^2 f_{sa_s}(m)^2\right] = (1-\chi_s)\,f_{s0}(m)^2\,\mathbb{E}_q\left[\ell_{sb}^2|a_s = 0\right] + \chi_s f_{s1}(m)^2\,\mathbb{E}_q\left[\ell_{sb}^2|a_s = 1\right], \tag{2.40}$$

and

$$\mathbb{E}_{q_s}\left[\ell_{sb}^2|a_s = i\right] = \mathbb{E}_q\left[r_s^2|a_s = i\right]\prod_{j=1}^{B-1}\mathbb{E}_q\left[\exp\left\{2I_b(j)\,c_{sb}\right\}\Big|a_s = i\right], \tag{2.41}$$

and

$$\mathbb{E}_q\left[r_s^2|a_s = i\right] = \exp\left\{2\left(\gamma_s^{(i)} + \zeta_s^{(i)}\right)\right\}. \tag{2.42}$$

This technique is known as delta-method variational inference [24, 25]. Because z_{nbm} is a sum over celestial bodies, whose corresponding random variables are treated as independent in q, the computational complexity of the approximation scales linearly in the number of celestial bodies:

$$\mathbb{V}_q[z_{nbm}] = \sum_{s=1}^{S}\mathbb{V}_q[\ell_{sb}\breve{f}_{sa_s}(m)]. \tag{2.43}$$

2.2.1.2 Kullback-Leibler divergence

The second expectation in equation (2.24) is $D_{\mathrm{KL}}\left(q(\Theta), p(\Theta)\right)$, the Kullback-Leibler divergence between the variational distribution and the prior. Intuitively, this term penalizes variational distributions that deviate from the prior, even though they may fit the data more closely. To compute it, we make use of the factorization of the variational distribution:

$$\begin{aligned}
&D_{\mathrm{KL}}\left(q(\Theta), p(\Theta)\right) \\
&\quad = \int \left[\log q - \log p\right] \prod_{s=1}^{S} q(a_s) q(r_s|a_s) q(k_s|a_s) \prod_{b=1}^{B-1} q(c_{sb}|a_s) && (2.44)\\
&\quad = \sum_{s=1}^{S} D_{\mathrm{KL}}\left(q\left(a_s, r_s, k_s, c_s\right), p_s\left(a_s, r_s, k_s, c_s\right)\right) && (2.45)\\
&\quad = \sum_{s=1}^{S} D_{\mathrm{KL}}\left(q(a_s), p(a_s)\right) + \sum_{i=1}^{2} q(a_s = i) && (2.46)\\
&\qquad\qquad \cdot \left[D_{\mathrm{KL}}\left(q(r_s|a_s = i), p(r_s|a_s = i)\right) + D_{\mathrm{KL}}\left(q\left(k_s, c_s|a_s = i\right), p_s\left(k_s, c_s|a_s = i\right)\right)\right].
\end{aligned}$$

Then, the KL-divergence between color vector c_s and color prior indicator k_s (an auxilliary variable) is

$$\begin{aligned}
&D_{\mathrm{KL}}\left(q\left(k_s, c_s|a_s = i\right), p_s\left(k_s, c_s|a_s = i\right)\right) \\
&\quad = \int \left[\log q(k_s|a_s = i) + \log q\left(c_s|k_s, a_s = i\right) - \log p(k_s|a_s = i) - \log p\left(c_s|k_s, a_s = i\right)\right] && (2.47)\\
&\qquad\qquad \cdot q(k_s|a_s = i) q(c_s|k_s, a_s = i)\\
&\quad = D_{\mathrm{KL}}\left(q(k_s|a_s = i), p(k_s|a_s = i)\right) && (2.48)\\
&\qquad\qquad + \int \left[\log q\left(c_s|k_s, a_s = i\right) - \log p\left(c_s|k_s, a_s = i\right)\right] q(k_s|a_s = i) q(c_s|k_s, a_s = i)\\
&\quad = D_{\mathrm{KL}}\left(q(k_s|a_s = i), p(k_s|a_s = i)\right) && (2.49)\\
&\qquad\qquad + \sum_{d=1}^{D} q\left(k_s = d|a_s = i\right) D_{\mathrm{KL}}\left[q\left(c_s|a_s = i\right), p\left(c_s|k_s = d, a_s = i\right)\right].
\end{aligned}$$

The KL-divergence between astronomical object type indicator a_s is

$$D_{\mathrm{KL}}\left(q(a_s), p(a_s)\right) = \chi_s \log \frac{\chi_s}{\Delta} + (1 - \chi_s) \log \frac{1 - \chi_s}{1 - \Delta}. \tag{2.50}$$

The KL-divergence between conditional brightness in the reference band is

$$D_{\mathrm{KL}}\left(q(r_s|a_s = i), p(r_s|a_s = i)\right) = \log \frac{\Psi^{(i)}}{\zeta_s^{(i)}} + \frac{\zeta_s^{(i)} + \left(\gamma_s^{(i)} - \Upsilon^{(i)}\right)^2}{2\Psi^{(i)}} - \frac{1}{2}. \tag{2.51}$$

The KL-divergence between the color prior mixture component indicators is

$$D_{\mathrm{KL}}\left(q(k_s|a_s = i), p(k_s|a_s = i)\right) = \sum_{d=1}^{D} \kappa_{sd}^{(i)} \left[\log \kappa_{sd}^{(i)} - \log \Psi^{(i,d)}\right]. \tag{2.52}$$

And finally, the KL-divergence between color vectors is

$$
\begin{aligned}
&D_{\mathrm{KL}}\left(q\left(c_s|a_s=i\right), p\left(c_s|k_s=d, a_s=i\right)\right) \\
&\quad = \frac{1}{2}\left\{\left(\sum_{b=1}^{4}\left[\left(\Lambda^{(i,d)}\right)^{-1}\right]_{bb}\lambda_{sb}^{(i)}\right) + \left(\Omega^{(i,d)} - \beta_s^{(i)}\right)^{\intercal}\left(\Lambda^{(i,d)}\right)^{-1}\left(\Omega^{(i,d)} - \beta_s^{(i)}\right)\right. \\
&\qquad\qquad \left. - 4 - \sum_{b=1}^{4}\log\lambda_{sb}^{(i)} + \log\left|\Lambda^{(i,d)}\right|\right\}.
\end{aligned}
$$

2.2.2 Optimization

Once the expectations in the ELBO are replaced with analytic expressions, maximizing it becomes a standard optimization problem, amenable to various techniques. We use L-BFGS-B [26].

When possible, we use existing star and galaxy catalogs for initialization. When no catalog is suitable, we convolve the images with matched filters to increase the signal-to-noise ratio [27]. We find each pixel whose value exceeds both the values of its neighboring pixels and an upper bound on the number of photons that could come from sky noise. We initialize the center of each such pixel as a celestial object. The number of such pixels determines S, the number of objects we assume are present in the image. Modeling S as random is the subject of ongoing research.

We compute derivatives alongside the evaluation of the objective, with little overhead: results for the most computationally expensive operations required to evaluate the objective, like exponentiation, can be reused for evaluating the derivative. We have not found the speed of automatic differentiation toolkits competitive with manually coded derivatives, so our current results use the latter. Validating manually derived derivatives against approximations from numeric differentiation is essential.

To get a feel for the scale of the computation, consider how we produced the results in the next section. The objective and its derivative are summations over pixels (Equation (2.31)). At each pixel, for each nearby celestial body, we evaluate 45 Gaussian densities, to compute the quantities in Equations (2.15) and (2.17). In the model, every celestial body can contribute photons to every pixel. In practice, we truncate these Gaussians. Most stars and galaxies contribute photons to fewer than 100 pixels, and no pixels are thought to receive photons from more than 10 celestial bodies.

With these techniques, evaluating the ELBO takes several seconds on a 2000-pixel image containing a few celestial bodies, and roughly 5 minutes on a 4-megapixel image with hundreds of celestial bodies. The calculation for a single image could be parallelized, though instead we elect to process images in parallel, on separate processors.

2.3 Experiments

For real astronomical images, ground truth is unknown. However, a region of the sky known as “Stripe 82” has been imaged more than 30 times by modern telescopes, whereas most of the sky has been imaged through all five filter bands of our chosen photometric system just once. “Photo” [6, 28] is the current state of the art for detection and characterization of celestial bodies. We henceforth refer to Photo, limited to just one image in each band, as “Primary,” and Photo run on the complete collection of replicated Stripe 82 images as “Coadd.” Coadd serves as the ground truth in our subsequent analysis. However: (1) any systematic biases in the Photo software are shared by both Coadd and Primary, and (2) the composition of images taken through different atmospheric conditions can create its own biases. Nonetheless, with predictions based on at least 30 times more data, we expect that Coadd accurately characterizes any celestial body detected by either Primary or Celeste.

We compare Celeste to Primary on 654 celestial bodies from Stripe 82, selected based on Coadd (the ground truth), that were not so bright as to be trivial to detect, but not so dim as to be impossible to detect. The data are sets of $B = 5$ images, called “stamps,” each centered on a selected celestial body. For these experiments, stamps substitute for fields in Figure 2.2. Unlike fields, the stamps do not overlap, and a celestial body appearing in one stamp does not contribute photons to other stamps that we analyze. Multiple celestial bodies contribute photons to most stamps.

We initialize Celeste to the output of Primary, so that we can assess the marginal improvement obtained from our inference procedure. Results appear in Table 2.1.

Sample size varies by row because the galaxy models for Celeste and Photo are not always comparable: both fit exponential and de Vaucouleurs light-kernel prototypes to each galaxy, but in Celeste both prototypes are constrained to have the same rotation and scaling applied to them. Hence, for rotation and scaling measures, we only compare using galaxies where Coadd puts all the mixing weight on one of the two prototypes.

Primary (Photo) is a carefully hand-tuned heuristic. Yet, Celeste matches or improves Photo on most metrics; only for reference-band brightness and scale is Celeste worse by more than two standard errors. This result comes on a data set of difficult celestial bodies, with ground truth set by Photo itself. For each color, Celeste reduces Primary’s error, making Celeste (initialized by Photo) state-of-the-art for color detection. Whereas Primary estimates each filter band’s brightness independently, Celeste predicts band brightnesses jointly, and regularizes these predictions based on prior information.

Celeste significantly outperforms Primary for position too. The 9.8% (+/- 2.0%) smaller position error is of practical importance to astronomers.

2.3.1 Synthetic images

To gauge the extent to which Celeste’s performance is limited by model misfit, or by errors in Coadd (the ground truth), we also test Celeste with synthetic images. For each real image, we generate a synthetic image with the same properties, with the locations, celestial object

Table 2.1: Columns 1 and 2 are the average error for Primary and Celeste on celestial bodies from Stripe 82; column 4 is the average error for Celeste on synthetic images. **Lower is better.** "Improve" is the improvement from Celeste relative to Primary, with SE in parentheses. Celeste on synthetic data is compared to Primary on real data; see the text. "N" is sample size. "Position" is error, in pixels, for the location of the celestial bodies' centers. "Missed gals" counts galaxies labeled as stars. "Missed stars" counts stars labeled as galaxies. "Brightness" measures the reference band (r-band) brightness in nanomaggies. "Colors" are log-ratios of brightnesses in consecutive bands. "Profile" is a proportion indicating whether a galaxy is de Vaucouleurs or exponential. "Eccentricity" is the ratio between the lengths of a galaxy's minor and major axes. "Scale" is the effective radius of a galaxy in arcseconds. "Angle" is the orientation of a galaxy in degrees.

dataset	real			synthetic		
model	primary	celeste	improve	celeste	improve	n
position	0.22	**0.20**	.02 (.00)	0.08	.14 (.01)	654
missed gals	28 / 654	**15 / 654**	.02 (.01)	14 / 654	.02 (.01)	654
missed stars	**8 / 654**	31 / 654	-.04 (.01)	6 / 654	.00 (.01)	654
brightness	**0.76**	1.60	-.83 (.12)	0.29	.47 (.08)	654
color u-g	1.10	**0.49**	.61 (.04)	0.20	1.06 (.05)	582
color g-r	0.16	**0.09**	.07 (.01)	0.05	.43 (.02)	654
color r-i	0.09	**0.06**	.03 (.00)	0.04	.25 (.01)	654
color i-z	0.25	**0.10**	.15 (.01)	0.08	.31 (.02)	654
profile	0.19	0.23	-.04 (.02)	0.16	.03 (.02)	237
eccentricity	0.17	**0.13**	.04 (.01)	0.11	.05 (.01)	237
scale	**0.37**	1.28	-.91 (.17)	0.23	.14 (.04)	237
angle	19.40	18.10	1.40 (.80)	14.90	4.50 (.80)	237

types, and reference band bright of the celestial bodies set to Coadd, but with the colors and pixel values drawn from our model. For each synthetic image, we initialize Celeste to the predictions from Primary (run on the real images, not the synthetic images).

We would like to also test Photo on the synthetic images. Running Photo on new data, however, rather than using the catalogs from former runs of Photo, exceeds our capacity; Photo is a long, intricate hand-tuned pipeline that has not been compiled in 6 years. Comparing the results for Celeste on synthetic data to Primary is nonetheless informative, since the synthetic data mirrors Coadd (the ground truth).

For position, brightness, color, and all 4 properties of galaxies, we see large reductions in error by Celeste from using synthetic images rather than real images. The number of galaxies we misclassify as stars, and vice versa, is also reduced.

Table 2.2: Average error for Celeste's predictions, for real astronomical images, binned into quartiles by estimated uncertainty. SEs in parentheses. Bins of more uncertain predictions have greater average error, without exception.

	Q1	Q2	Q3	Q4
brightness	.27 (.02)	.53 (.04)	.94 (.10)	4.04 (.50)
color u-g	.17 (.01)	.43 (.04)	.65 (.07)	.85 (.09)
color g-r	.05 (.00)	.07 (.01)	.09 (.01)	.15 (.01)
color r-i	.03 (.00)	.04 (.00)	.06 (.01)	.08 (.01)
color i-z	.04 (.00)	.09 (.01)	.10 (.01)	.17 (.01)

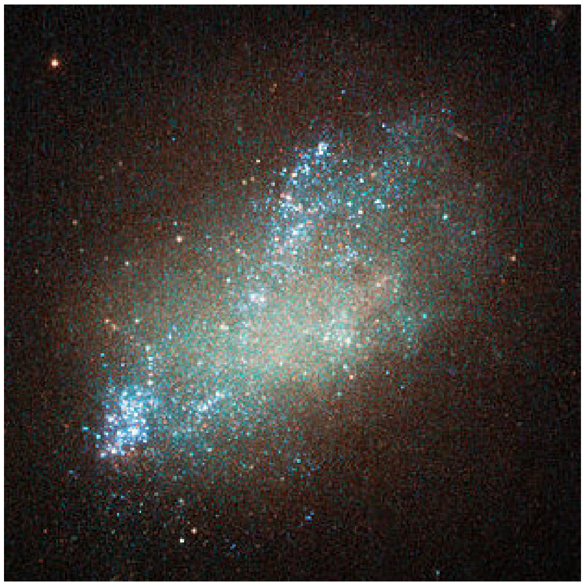


Figure 2.6: An irregular galaxy from the constellation Leo. Because the brightest areas of this galaxy are not near its center,Celeste may misfit this galaxy. credit: NASA

2.3.2 Uncertainty quantification

Celeste is fairly certain (≤1% uncertainty) about the classification (star vs. galaxy) for 526 out of 573 celestial objects. Of these classifications, 4% are wrong. Of the remaining classifications (>1% uncertainty), 45% are wrong.

For brightness and each of the four colors, observed error rates also correlate directly with reported uncertainty (Table 2.2). This correlation holds for synthetic data too.

2.3.3 Model misfit

On real data, galaxy model misfit may have constrained Celeste's performance across the board. Celeste's two-component galaxy model is based on the successful models from [29] and [17]. However, Celeste is a fully generative model, so any misfit between the modeled galaxy and the actual galaxy must be explained by Poisson randomness. The Poisson process is well-suited to modeling the variation in photon count when the underlying rate is correct. But, for galaxies, the rate itself may deviate greatly from the model. With this type of model misfit, the galaxy shape that optimizes the ELBO may be unduly influenced by irregularities in the underlying rate. Figure 2.6 illustrates a galaxy that would be difficult to fit with any simple parametric model, even if sampled at low spatial resolution. Enhancing Celeste's galaxy model is a promising direction of ongoing research.

Modeling the galaxy-specific parameters (ρ_s, φ_s, σ_s, and θ_s) as constants to be learned rather than as random variables with prior distributions likely also worsens performance. In some cases galaxies scales' (σ_s) were much too large; the optimizer may have been using a galaxy to explain a background noise rate that exceeded ϵ_{nb}. These cases likely explain why Primary outperformed Celeste in determining brightness and scale. Though the underlying issue is model misfit (as Celeste's good performance on synthetic data suggests), constraining σ_s with a prior distribution could mitigate this effect. Also, a model that assigns more degrees of freedom to galaxies—unconstrained by prior distributions—than stars biases the classification in favor of galaxies. Treating all unknown quantities in Celeste as random is the subject of ongoing research.

2.4 Conclusion

Posterior inference in a fully generative model works for astronomical object detection. Variational approximation in conjunction with techniques like the delta method, and modeling choices that lead to analytic expectations, make inference tractable. Treating pixel values as observed Poisson random variables, whose rate parameters are a function of latent variables, is feasible. It leads to performance that improves the state of the art for locating celestial bodies. Constraining star and galaxy colors with a Gaussian mixture prior, as well as modeling all colors jointly in the posterior, is also both feasible and effective: Celeste reduces the error rate for color detection as much as 60%. Characterization of galaxy shapes and profiles is feasible with the Celeste model too, but performance does not yet improve upon heuristic methods. Extending the Celeste model with a richer galaxy model and constraining more quantities with prior distributions are promising directions for future research.

Chapter 3

A deep generative model for astronomical images of galaxies

Galaxies in astronomical images (Figure 3.1) often resemble galaxy prototypes (Figure 3.2) and possess shared characteristics like spiral "arms," a "bar," or a "bulge." Even irregular galaxies—typically resulting from the collision of two regular galaxies—have shapes greatly constrained by physics.

Simple parametric models (e.g., [30, 31, 32, 33, 34]) suffice to describe many idealized galaxy shapes, but severely misfit actual galaxies: they are not sufficiently flexible [34]. The popular program GALFIT copes with the limitations of simple parametric models by allowing users to fit an arbitrary number of mixture components [35, 36]. These mixtures are not learned from actual galaxies, so the models cannot provide meaningful uncertainty estimates. To our knowledge, no existing galaxy models are learned from a training set, which would allow for such uncertainty estimates. Indeed, only [31] attempts a Bayesian treatment of galaxy shapes,[1] albeit one based on just a few manually specified parameters. Yet modeling galaxies is an important part of learning about the universe from large-scale astronomical sky surveys [33, 34, 37, 38], and billions of images of galaxies are available for training.

Neural networks—high-dimensional parametric models—have enjoyed great success for classifying images [39]. They have been effective at discriminating between stars and galaxies [40] and for labeling images of galaxies as possessing or lacking certain features, such as a "bar" or spiral "arms" [41]. However, to our knowledge no one has yet reported on melding the flexibility of neural networks and a generative probabilistic model for galaxies. Recent advances in variational inference for non-conjugate models [42, 43] make this possible. To our knowledge, this is the first publication to report on applying these advances to a problem in the physical sciences.[2]

[1]The galaxy models in [33] and [34] are deterministic though they are embedded in probabilistic models.

[2] We are informed by a review of reverse citations and personal correspondence with Diederik Kingma (10/8/2015).

Figure 3.1: The Whirlpool galaxy—a classic spiral galaxy. credit: ESA Hubble / NASA.

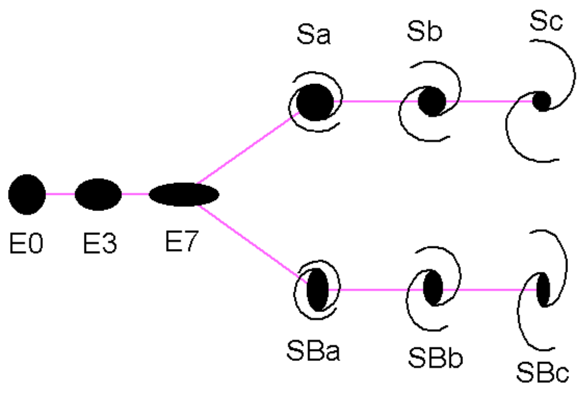


Figure 3.2: The Hubble "tuning fork" of galaxy morphology. credit: Todd Thompson.

3.1 The model

For a particular image x of a galaxy, let z be a low-dimensional latent random vector, distributed as a multivariate standard normal. Given z, we model the observed intensities of the image's pixels $x = (x_1, \ldots, x_m)$, as

$$x|z \sim \mathcal{N}\left(f_\mu(z), f_\Sigma(z)\right). \tag{3.1}$$

We take the deterministic functions f_μ and f_Σ to be neural networks that share some weights. We constrain f_Σ to produce diagonal covariance matrices. As shorthand, let the neural network $f(z) := (f_\mu(z), f_\Sigma(z))$.

3.1.1 Inference

Given an image's pixel intensities $x = (x_1, \ldots, x_m)$, we aim to infer the posterior distribution of $z = (z_1, \ldots, z_n)$. Unfortunately, integrating z out of the joint distribution (x, z) to compute the marginal likelihood of x is intractable due to the nonlinear form of f. Therefore, we turn to variational inference. In keeping with the approaches in [42] and [43], let the variational approximate posterior take the form

$$q(z|x) = \mathcal{N}(g_\mu(x), g_\Sigma(x)), \tag{3.2}$$

where g_μ and g_Σ are neural networks that map x to a mean vector and a diagonal covariance matrix, respectively. As shorthand, let neural network $g(x) \coloneqq (g_\mu(x), g_\Sigma(x))$. By the standard construction of the variational lower bound,

$$\begin{aligned} \log p(x) &\geq \log p(x) - D_{\mathrm{KL}}\left[q(z|x), p(z|x)\right] &(3.3)\\ &= \mathbb{E}_q\left[\log p(x|z)\right] - D_{\mathrm{KL}}\left[q(z|x), p(z)\right]. &(3.4)\end{aligned}$$

Therefore, the distribution q that maximizes (3.4) minimizes $D_{\mathrm{KL}}\left[q(z|x), p(z|x)\right]$: this q is the best approximation of form (3.2) to the posterior. Let W_f and W_g be the weights of neural networks f and g, respectively. Maximizing over $W = (W_f, W_g)$ simultaneously finds the q that best approximates the posterior and the model p that assigns the highest probability to our data.

The normal-normal KL-divergence $D_{\mathrm{KL}}\left[q(z|x), p(z)\right]$ is closed form, but $\mathbb{E}_q\left[\log p(x|z)\right]$ is not. We can nonetheless efficiently compute unbiased estimates of its gradient, and therefore maximize (3.4) by stochastic gradient optimization.

The stochastic gradient described in [42, 43, 44] and example 5.1 of [45], based on "the reparameterization trick," has the lowest variance among all unbiased estimators. Let $\epsilon \sim \mathcal{N}(0, I)$. Then

$$\begin{aligned} \frac{\partial}{\partial W}\mathbb{E}_q\left[\log p(x|z)\right] &= \frac{\partial}{\partial W}\mathbb{E}_\epsilon\left[\log p\left(x|z = g_\Sigma(x)\epsilon + g_\mu(x)\right)\right] &(3.5)\\ &= \mathbb{E}_\epsilon\left[\frac{\partial}{\partial W}\log p\left(x|z = g_\Sigma(x)\epsilon + g_\mu(x)\right)\right]. &(3.6)\end{aligned}$$

Hence, for e sampled from ϵ,

$$\frac{\partial}{\partial W}\log p\left(x|z = g_\Sigma(x)e + g_\mu(x)\right) \tag{3.7}$$

is an unbiased estimate of the derivative of $\mathbb{E}_q\left[\log p(x|z)\right]$.

3.2 Experiments

We apply our model to preprocessed 424×424-pixel images of galaxies from the Sloan Digital Sky Survey [46, 47]. In keeping with the approach of [41], we crop each image to surround just the most prominent galaxy and downscale these subimages to 69×69 pixels. Based on a blob detection routine, we exclude images where the most prominent galaxy overlaps with other bright astronomical objects,[3] leaving 43,444 images for training.

[3]"Deblending" astronomical objects is a related problem, likely facilitated by an accurate galaxy model, but beyond the scope of this work.

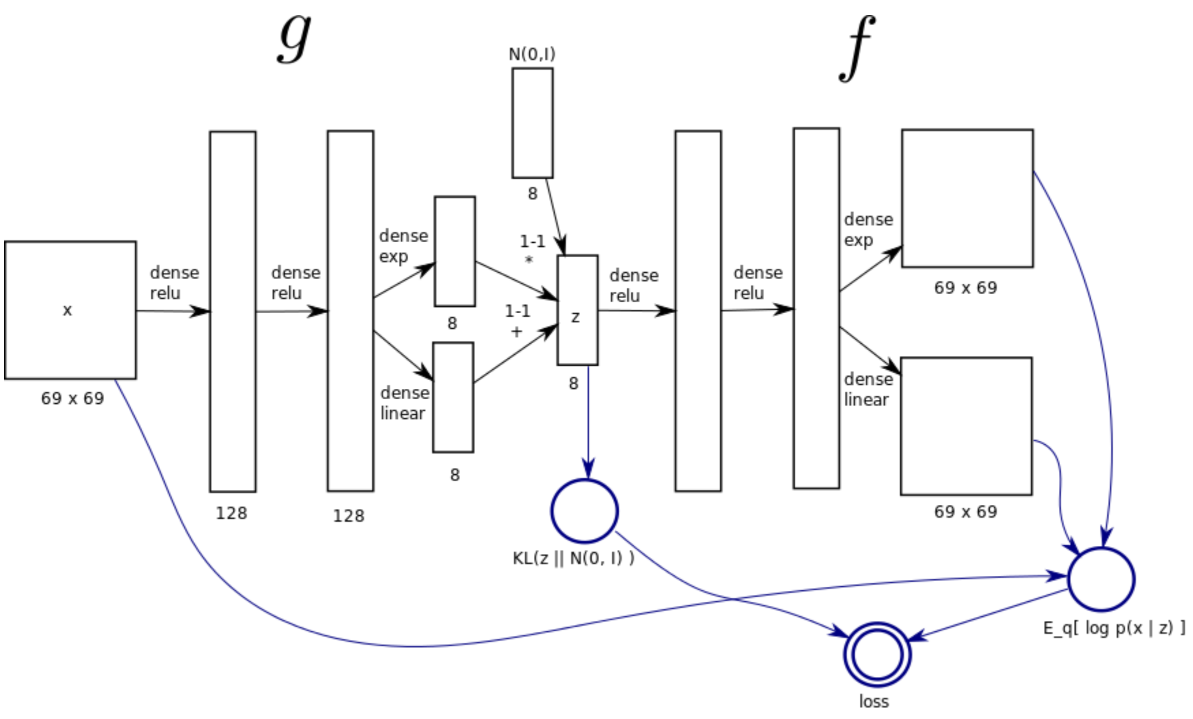


Figure 3.3: The architecture for the proposed generalized denoising autoencoder.

3.2.1 Implementation

Fitting our model by stochastic gradient descent involves simultaneously training two neural networks: f, for specifying the generative model p, and g, for specifying the variational distribution q.

An alternative perspective is helpful for implementing the fitting procedure: Both f and g are components in a single neural network called a "generalized denoising autoencoder" (GDAE) [48, 49]. An image x is input to g, yielding $g_\mu(x)$ and $g_\Sigma(x)$. The next layer in the GDAE corrupts $g_\mu(x)$. Its inputs are $g_\mu(x)$, $g_\Sigma(x)$, and a sample e from $\mathcal{N}(0, I)$. Its output is $g_\mu(x) + g_\Sigma(x)e$. This output z serves as the input to f. The output of f is penalized by the expected negative reconstruction error: $-\log p\,(x|z)$. As a form of regularization, the output of g is penalized too, according to $D_{\text{KL}}\,[q(z|x), p(z)]$.

This perspective facilitates adapting existing neural network software to learn the proposed generative model. Mocha.jl [50] is a neural network toolkit written in Julia, inspired by Caffe [51]. We reuse the basic framework from Mocha.jl, but augment it with new types of layers to compute the proposed loss function. The parts of the network corresponding to f and g each have two fully connected hidden layers composed of 128 nodes each, all with rectified linear units. The parts corresponding to the output layers of f and g each use exponential nonlinearities to ensure that variances are strictly positive. We set the dimension of z to eight. Figure 3.3 diagrams the architecture. On an Nvidia Tesla K20X GPU, our network performs roughly 200 iterations per second. (Each iteration involves forward and backward propagation for one image.) Parameter-specific learning rates are set adaptively [52].

3.2.2 Results

First, we examine the trained model qualitatively. Figure 3.4 shows sample input images to the trained autoencoder from a held-out set, and the resulting output. The mean of the reconstruction $f_\mu(z)$ resembles a smoothed version of the input. The variance of the reconstruction $f_\Sigma(z)$ is low for the backgrounds, which by construction is nearly black in the original images. Variance is higher for the foreground, particularly near the borders of each galaxy—presumably z cannot store enough information to represent slight differences in galaxies' sizes. The intensity of the third galaxy's center is particularly uncertain, which may reflect that some but not all galaxies have a prominent "bulge" in the center.

Figure 3.5 shows a two-dimensional embedding of a held-out set of galaxies, generated by applying t-SNE [53] to the 8-dimensional means $g_\mu(x)$ of the galaxies' variational distributions. At this resolution, galaxies are clearly grouped by their orientations. Some clustering of spiral galaxies is apparent too.

Figure 3.6 shows $f_\mu(z)$, that is, the mean of $p(x|z)$, for values of z selected by a one-at-a-time experimental design.

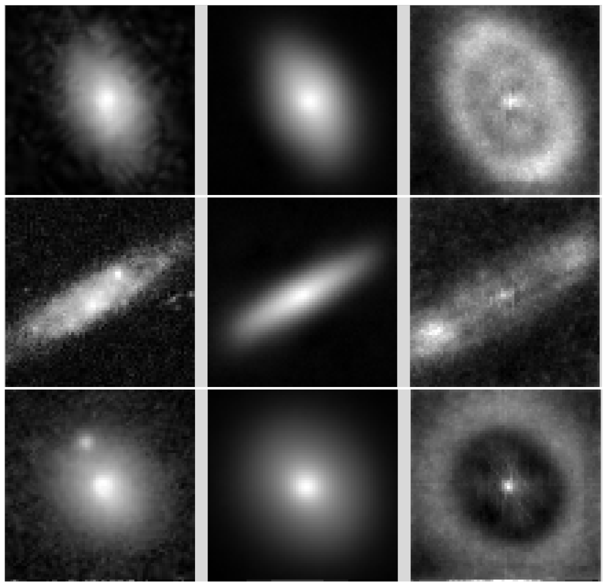


Figure 3.4: Each row corresponds to a different example from a test set. The left column shows the input x. The center column shows the output $f_\mu(z)$ for a z sampled from $\mathcal{N}(g_\mu(x), g_\Sigma(x))$. The right column shows the output $f_\Sigma(z)$ for the same z.

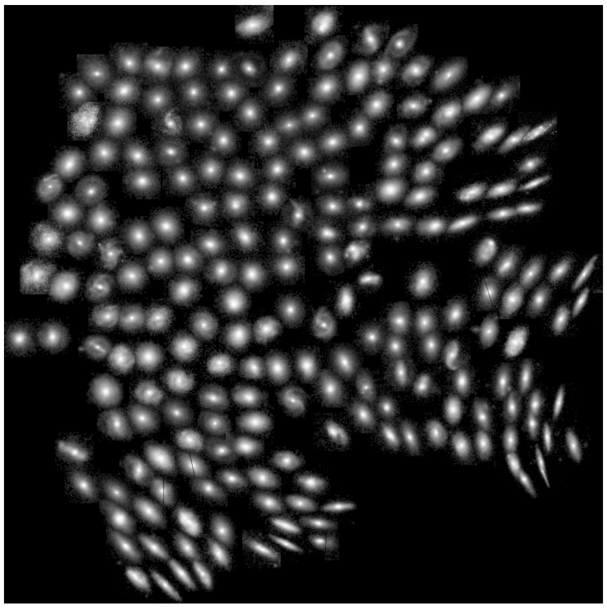


Figure 3.5: Galaxies embedded in two dimensions based on the means of their variational distributions, $f_\mu(x)$.

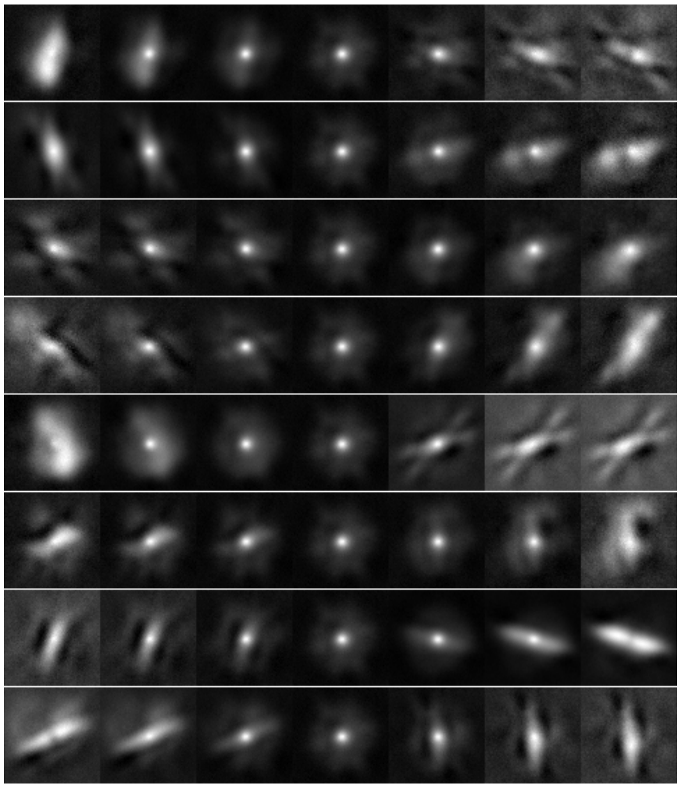


Figure 3.6: $f_\mu(z)$ for z values sampled according to a one-at-a-time experimental design. In each row, from left to right, one dimension of z is incremented by one standard deviation per column, while the other dimensions are fixed at zero. The center column in each row is $f_\mu(0,\ldots,0)$. The leftmost and rightmost columns are 3 standard deviations from the mean and thus highly unlikely; we show these extremes to highlight the effect of each dimension of z.

Because the model we propose is, to our knowledge, the first galaxy model learned from a training set, comparing it to existing galaxy models is not straightforward. Comparison is also challenging because the most common galaxy models do not explicitly model uncertainty.

We also compare the proposed galaxy model to a current common practice: fitting a scaled bivariate Gaussian density function to each imaged galaxy. On a held-out dataset of 1000 images of galaxies, we compute $f_\mu(g_\mu(x))$. This amounts to running the proposed autoencoder with the layer for sampling z replaced with the mean of z. For each image, we also fit a scaled bivariate Gaussian density to minimize squared error averaged over pixels. The optimization was performed with BFGS over six unconstrained parameters: two for the mean, three for the Cholesky decomposition of the covariance, and one for the scale. For 971 of 1000 images, $f_\mu(g_\mu(x))$ fit x more closely than the best scaled bivariate Gaussian density. In some sense this is not surprising, since only the proposed model uses training data. On the other hand, the parameters of the proposed model are only optimized by a feed-forward recognition model rather than an iterative algorithm, and only the scaled bivariate Gaussian model is explicitly trained to minimize residual sum of squares.

Fitting a function to minimize residual squared error averaged across pixels is analogous to maximizing the likelihood of the data for a model where all pixels have a Gaussian distribution with a common variance. This interpretation lets us compare our proposed model, conditioned on a particular z, to the scaled bivariate Gaussian density function in terms of log likelihood. Now for each image, in addition to fitting a scaled bivariate Gaussian density function to each held-out image, we learn the variance shared by all pixels that assigns the highest likelihood to the image. (The solution is closed form.) We compare this to the likelihood assigned to the data by the model we propose, for a particular z. For 972 of 1000 images, the model we propose better explains the data. Both models treat each pixel intensity as Gaussian, but only the model we propose assigns different variances to different pixel.

3.3 Future work

The proposed model shows little sign of overfitting our existing training set, and billions of additional images of galaxies are freely available. By increasing the dimension of z and by making our network deeper, we could almost certainly improve accuracy on held-out data. Augmenting f with intermediate latent layers [43] would also likely improve accuracy on held-out data and better model uncertainty about the structure of the galaxies, rather than just uncertainty at the level of individual pixels.

We could also exploit the rotational and reflective symmetries of galaxies, either through data augmentation or with a network architecture that explicitly enforces it.

Our immediate focus, however, is on embedding the current galaxy model into the model for raw astronomical images (not cropped around galaxies) described in [34]. Augmenting the broader model with this data-adaptive galaxy model likely will improve its performance across the board.

Part II

Author disambiguation

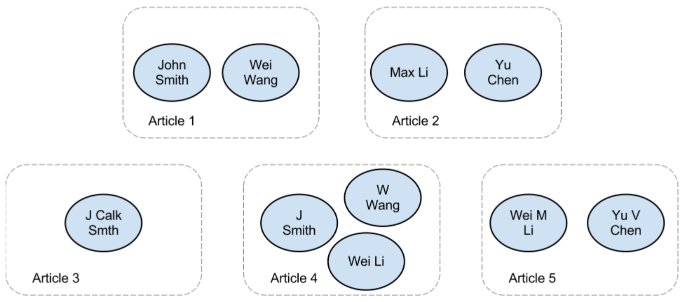
Chapter 4

Problem overview and preliminaries

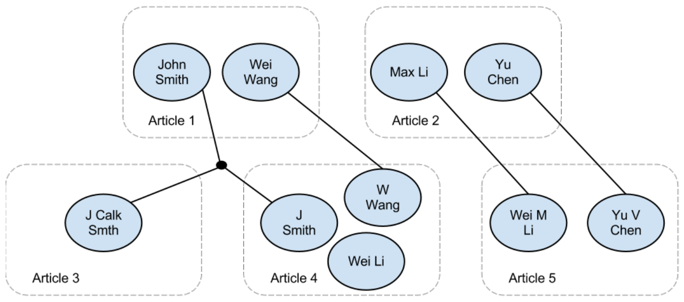
In a large collection of academic publications, many authors' names will not be unique. For instance, a collection may contain hundreds of authors named "J. Smith." Moreover, a single author may be referred to in multiple ways, perhaps due to misspellings, name changes, or spelling variations. The problem of *author disambiguation* is to determine who wrote what [3]. We refer to each instance of an author's name string as a *mention*, along with attributes such as the containing article's keywords and topics (explicit or inferred), and the name strings of other authors of the article. Figure 4.1 illustrates a collection of mentions without attributes besides name strings, before and after author disambiguation.

In addition to being an important problem in its own right, author disambiguation is representative of a broader class of clustering problems, known as *disambiguation* problems. Entity resolution, citation matching, noun-phrase coreference resolution, database hardening and record linkage are examples of disambiguation problems. Figure 4.2 illustrates the relationship between several of these research areas. In disambiguation problems, given a collection of items (e.g., author name strings, customer records, or noun phrases), we seek to find a clustering of the items such that each cluster contains only coreferent items, and no items in different clusters are coreferent. In disambiguation, as opposed to clustering in general, the true number of clusters (e.g., authors, customers, or anaphoric sets) usually grows linearly in the number of items.

General-purpose clustering algorithms are poorly suited to large-scale disambiguation problems. Agglomerative clustering, k-means, and traditional implementations of spectral clustering all have (at least) quadratic runtime on disambiguation problems because the number of clusters k scales linearly with the number of items n. When faced with an algorithm whose runtime is quadratic in the number of items, practitioners frequently use a computationally inexpensive pre-processing algorithm to divide the items into blocks [54]. Then they process each block separately. But this approach has numerous drawbacks, and often some blocks' sizes will still grow linearly in the total number of items. Some researchers have proposed techniques for applying spectral clustering to large datasets without blocking. While an $n \times n$ matrix encoding the similarities between every pair of items cannot be explicitly constructed, these researchers seek to develop space-efficient alternatives, through



(a) before



(b) after

Figure 4.1: Bibliographic data before and after author disambiguation. Ovals represent author mentions, rectangles represent articles, and (hyper)edges represent unique authors.

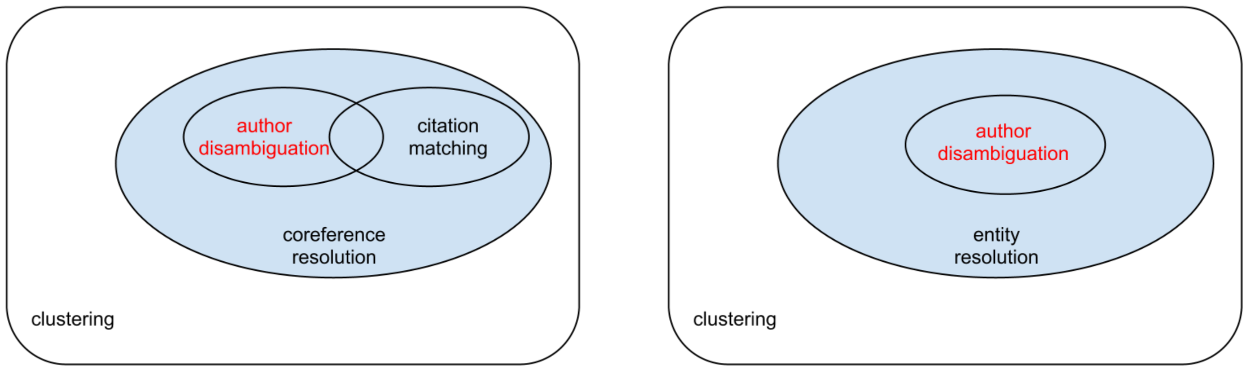


Figure 4.2: Relationship between author disambiguation and other research areas

subsampling [55, 56] or low-rank approximation [57]. But the number of clusters k to recover effectively lower bounds the rank of a similarity matrix that is sufficient to recover the correct clusters, and for disambiguation problems, k grows linearly in n. Furthermore, determining the support of a sparse similarity matrix, at least with existing approaches, would itself take quadratic time.

Section 4.1 reviews existing approaches to author disambiguation, none of which go beyond blocking to address the issue of quadratic runtime. Section 4.2 introduces and analyzes two manually disambiguated datasets that we use throughout. Section 4.3 proposes a novel technique for disambiguating authors in nearly linear time without blocking. This technique makes tractable the approaches to author disambiguation throughout Part II. Section 4.4 develops a sophisticated-but-heuristic approach to author disambiguation, which serves as a baseline for comparison to the model-based approaches in subsequent chapters. Also, it illustrates the details that accurate author disambiguation must account for.

4.1 Related work

Author disambiguation has been extensively studied, though it remains an open problem. Several studies are representative of the state of the field.

In [58], records with identical person names are disambiguated. For a general solution, we would need to generalize the notion of identical names to include all names that could conceivably refer to the same author. Like most approaches to author disambiguation, [58] uses agglomerative clustering and a statistic like a Jaccard coefficient [59] to measure similarity between authors. Novel contributions include 1) the use of a PageRank-style algorithm to determine the importance of each edge in the co-author network; 2) generalization of the notion of co-authorship to include any relationship, such as appearing in the same conference or journal; and 3) a way of generating training examples automatically, where rare names provide the positive examples and very different names provide the negative examples. The authors train an SVM with these examples, and apply it to test data to combine different set resemblances (basically Jaccard coefficients) to create a single similarity measure.

[60] proposes evaluating sets of mentions, and determining the likelihood that all mentions in the set refer to the same author entity. When the cardinality of these sets is 2, their method recreates existing pairwise approaches to author disambiguation. But when larger sets are evaluated, their method allows for similarity measures that favor partitions where each author entity 1) belongs to only a few institutions, 2) has only 1 or 2 different email addresses, and 3) publishes fewer than 30 publications per year. Their method could easily be extended to also favor clustering where each author writes papers filed under just 1 or 2 subjects. The proposed inference procedure is more principled than many, as it involves learning the parameters of a scoring function. This scoring function maps from a partition of the mentions to a real value representing the partition's goodness. This goodness score is not computed with respect to the ground truth, but with respect to the parameters of a scoring function. Also, the authors propose using a classifier in conjunction with something like a

perceptron in order to optimize the scoring function's parameters' values. Agglomerative clustering is still used as the final step of the procedure, to partition the test data.

[61] and [62] are widely cited articles that propose two methods for citation matching. One is based on a naive Bayes classifier and the other on an SVM. These methods are supervised, and training examples are needed for *every* author. Supervised methods can be useful for large-scale disambiguation if training examples are not needed for each author. But these supervised methods seemingly cannot be easily modified to work without such extensive training data, so these techniques are not directly relevant to large-scale name disambiguation.

[63] presents an architecture for large-scale author disambiguation. Mentions are blocked, so that pairwise comparisons can be performed efficiently A similarity function compares any two mentions in a block. The similarity function outputs a vector of similarities. These similarities are computed on text fields. Jaccard similarity, soft-TFIDF, and edit distance are all employed. A trained SVM takes these similarity vectors as input, and outputs a real-valued distance measurement. A novel clustering algorithm, DBSCAN, forms clusters based on the distance measurements. DBSCAN is similar to agglomerative clustering, except it resolves some transitivity violations. Additionally, this work is noteworthy for using LASVM, an online (as opposed to batch) SVM library. This facilitates training via active learning, and simplifies the process of folding in new publications. However, the co-authorship network is not used in this work.

[64] proposes two graphical models, based closely on probabilistic latent semantic indexing (pLSA) and latent Dirichlet allocation (LDA), that generate a collection of publications, including words and mentions. Gibbs sampling is used for inference. The better-performing LDA-based model specifies a distribution over topics for each mention. Euclidean distance in topic space measures the pairwise similarity among mentions. The distance between non-singleton clusters is the maximum distance between mentions in the two clusters. Levenshtein distance is also considered, though it is not combined with topic-based distance to create a single distance function. Rather, an agglomerative clustering algorithm merges the most topically similar clusters if and only if randomly chosen names from each cluster are closer than some arbitrary threshold with respect to Levenshtein distance. This approach works for disambiguating mentions on web pages too, because the co-authorship network is not used. The results suggest that such an approach substantially outperforms [63] with respect to F-measure. However, the authors refer to [63] as an unsupervised model, when in fact [63] is supervised and requires training data. Therefore, the comparison may not be apt. Interestingly, topic code helps their routine disambiguate. The best performance is attained from topic distance, as computed in [64], as a feature in the model presented in [63].

[65] presents a model tuned for disambiguating Medline data. For a candidate pair of mentions, a similarity vector is formed based on first name, last name, middle initial, suffix, article title words, affiliation, journal name, language and co-authors. One entry in feature vector X is the number of shared co-authors' names on the two papers in question, when each co-author's name is reduced to the first letter of the first name and the last name. Given a similarity vector x, the probability of a common author $\mathbb{P}(M|x)$ is computed using

Bayes's rule:

$$\mathbb{P}(M|x) \propto \mathbb{P}(x|M)\mathbb{P}(M).$$

Critically, $\mathbb{P}(M)$ is not the marginal probability that *any* two mentions are a match. Rather, $\mathbb{P}(M)$ is the prior probability that two mentions *sharing a particular name* are a match. The conditional probability $\mathbb{P}(M|x)$ is monotonic in x, in that if $x \leq y$ component-wise, then

$$\mathbb{P}(M|x) \leq \mathbb{P}(M|y).$$

This property facilitates interpolation and extrapolation of estimates for unobserved similarity vectors, based on a lookup table. Independence between some features is also exploited to produce a lookup table mapping similarity to vectors to probabilities. Quadratic programming for least squares minimization enforces the monotonicity property. Pairs of mentions that share rare names serve as positive examples. Randomly selected pairs of mentions that have different names serve as negative examples.

In [66], graphical models are used for author disambiguation. The proposed model is based on Latent Dirichlet Allocation, where hidden groups (or communities) generate authors. For each mention, a latent random variable is added to LDA, to model which author generates it. A noise model determines how mentions are generated from author entities. It determines how likely names are to be dropped, converted to initials, or misspelled. A Gibbs sampler performs inference. In addition to assigning author entities to communities, and mentions to author entities, it is also necessary to determine the most likely name for each author entity at each iteration of the Gibbs sampler. The number of author entities is also unknown, but it can be modeled by a Dirichlet Process. The resulting sampling procedure would be prohibitively slow if it were to operated on each mention at every iteration. Instead, the authors propose permitting only split and merge operations, or equivalently, performing block assignment of mentions to author entities. An advantage of their approach is that disambiguation is collective and decisions are not pairwise. The approach does not require training data, since it is largely unsupervised, though there are several parameters that need to be manually set. Disadvantages include the runtime and the need to guess the values of hyper-parameters, an initial assignment, and the true number of hidden community variables. Several tricks make the Gibbs sampler mix better, such as forcing estimates of name corruption probabilities to evolve slowly and introducing a scalar to control the rate of cluster merges. Still, the Gibbs sampler had convergence issues according to personal correspondence (1/29/2009) with the first author (Indrajit Bhattacharya).

In [67], the authors propose a pairwise similarity metric based not only on the attributes of mentions, but on the entities that have already been determined. Initially, each mention is its own cluster. During a bootstrapping phase, only mentions with very high attribute similarity are merged into the same cluster. For example, two mentions with rare yet matching names would have high attribute similarity. After bootstrapping, merges are based on a linear combination of attribute similarity and common co-authors. A variety of normalizations on the number of co-authors is performed. For example, dividing the number of co-authors in common by the total number of co-authors for two clusters yields a metric between 0 and 1.

Such a metric could be interpreted as a probability. On real datasets, [67] reports that the choice of normalization is not particularly important. Results from 3 datasets are presented, 2 of which are publicly available. For the `arXiv HEP` dataset, discussed subsequently, the best pairwise F-measure [68] is 0.985.

Many other approaches to author disambiguation and coreference resolution have also been attempted [69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87].

Finally, [3] provides an excellent overview of the field.

4.2 Exploratory data analysis

We base our analysis on two manually disambiguated collections of academic articles: `arXiv HEP` and `MathSciNet`. The former, `arXiv HEP`, contains 29,555 articles from the High-Energy Physics section of arXiv. These articles contain 58,515 mentions, referencing 9200 unique authors. The dataset contains only a modest number of authors in relation to the number of unique author names in dataset; it is representative of bibliographic data with low ambiguity. This dataset is publicly available and disambiguation results on this dataset are reported elsewhere.

Our second dataset, `MathSciNet`, comes from the American Mathematical Society. No manually disambiguated public dataset comes close to its scale. It contains 2,041,269 articles about mathematics, including title, subject code, journal, and year published. These articles contain 3,366,842 mentions, referencing 512,506 unique authors. Because of its large scale, many authors have the same names.

4.2.1 Name string variations

By examining the set of name strings that refer to the same author in these two databases, we assemble a list of the common variations. Accounting for these variations is an essential aspect of author disambiguation.

Name encoding. Name strings that contain non-ASCII characters are often transcoded to ASCII (e.g., $\tilde{n}$ to n) for some instances but not others. Name strings may have uppercase letters sometimes and title case others (e.g., JOHN and John). Name strings may have non-standard spacing, such as a trailing space, a tab character rather than a space, or two spaces rather than one (e.g., “John Smith ” and “John Smith”). Names may appears sometimes with the last name followed by a comma, but not others (e.g., “Smith, John” and “John Smith”).

Name decorators. Names strings may include titles (e.g., Dr., Mr., Mrs., Ms.), while others, for the same author, do not. Name strings may include suffixes (e.g., Jr, III, IV), while others, for the same author, do not. Short, uncapitalized name pieces (e.g., van, de, del, da, do, el, la, di, von, der) are frequently omitted.

Name completeness. First names and middle names are often initialized (John and J.) The middle name may be initialized but not the first name, vice verse, both, or neither. First names also get truncated quite often in these datasets (Chan to Ch.), and not necessarily to just the first two characters.

Nick names. First names and middle names are often replaced with nick names (e.g., Samantha to Sam). But this only happens for tens of common names.

Omitted first names. An author's first name may be omitted entirely, but only if a middle name is present to take the place of the first name (e.g. "s j fuchs" and "jurgen fuchs" and "c g ducati" and "g ducati"). Concluding that two authors have the same name based on this pattern can lead to errors, however. In `MathSciNet`, the names "alfredo t suzuki" and "takesi suzuki" are not coreferent.

Appended last names. An author may publish both before and after marrying. Rather than dropping the original last name, the author may append a hyphen and an additional last name to the original last name (e.g., "antonio j segui-santonja" and "a j segui"; and "marcia e knutt-wehlau" and "marcia e knutt"). Concluding that two authors have the same name based on this pattern can lead to errors, however. In `MathSciNet`, the names "a lima-santos" and "a f lima" are not coreferent.

Misspellings. There are two main types of spelling errors: typos and character encoding issues. The second type of error in some cases may be corrected during pre-processing. We observe character insertions, deletions, transpositions, and substitutions in `MathSciNet`—all the standard types of spelling mistakes. Examples from `MathSciNet` include "metin gurses" and "matin gurses," and "m b johnson" and "m b jonnson." However, correcting suspected misspellings, even with just one character edit, may incorrectly suggest coreference (e.g., "m sakaguchi" and "m sawaguchi").

4.2.2 Author productivity

Lotka's law states that the number of publications by each author follows a Zipf distribution [88]. Figure 4.3 illustrates how closely Lotka's law fits the empirical distribution of author productivities in `MathSciNet`. Some systematic bias is apparent. Nonetheless, Lotka's law may work well as an approximation if the quality of author disambiguation is not highly sensitive to misspecification of the author productivity distribution.

Figure 4.4 illustrates that the approximation holds for a subset of the authors in `MathSciNet` selected based on their first name. Lotka's law approximates the author productivity distribution for this subset too, with roughly the same rate parameter. In general, subsets of datasets selected by criteria that do not correlated strongly with author productivity should be well modeled by the author productivity distribution for the full dataset.

4.2.3 Baseline partitioning procedures

Before considering more complex disambiguation schemes, we verify that straightforward partitioning algorithms do not suffice. Figure 4.5 shows the accuracy of 4 simple partitioning schemes. The `singletons` partitioning scheme places each mention in its own cluster. Because 43% of authors only wrote only one publication (as suggested by Lotka's law), the

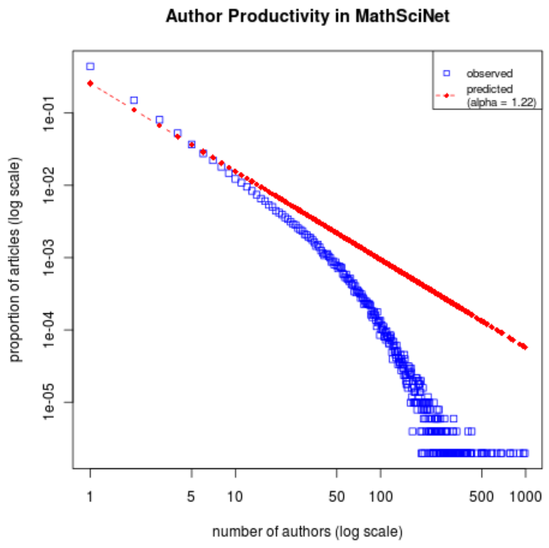


Figure 4.3: The empirical distribution of author productivity in `MathSciNet` is approximately log-log linear, as predicted by Lotka's law.

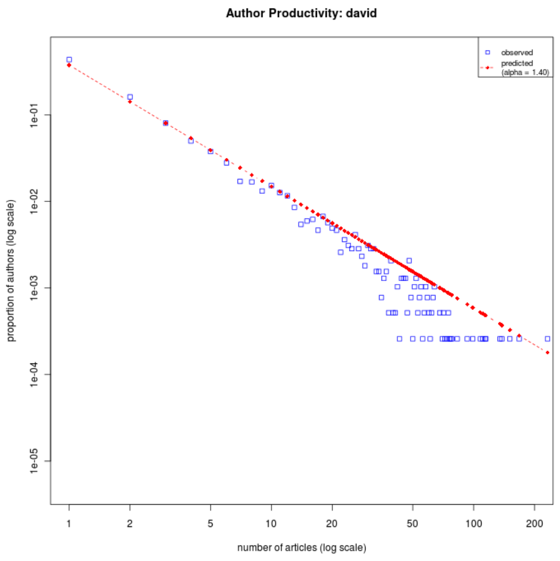


Figure 4.4: The empirical distribution of author productivity in `MathSciNet` for authors named "David" is also approximately log-log linear, with roughly the same slope. Lotka's law applies to subsets of authors selected by attributes not highly correlated with author productivity, too.

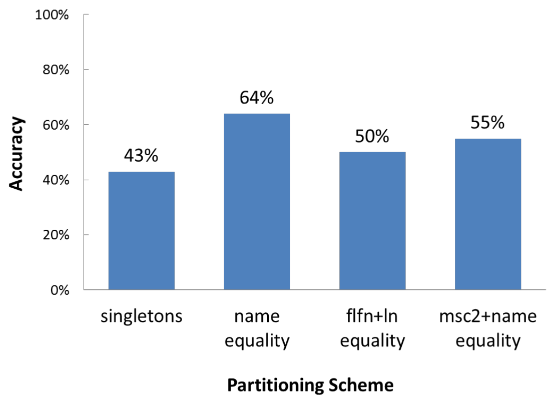


Figure 4.5: Proportion of unique authors in `MathSciNet` disambiguated without *any* mistakes, by each of 4 straightforward partitioning schemes.

singleton scheme disambiguates 43% of authors perfectly. The `name equality` partitioning scheme groups mentions having the exact same name, after some simple preprocessing suggested by the analysis in Section 4.2.1. Still, more than one third of all authors are incorrectly disambiguated—way too many for most applications. One reason is that many authors have their first names initialized sometimes but not others, or have a middle name included sometimes but not others. Either inconsistency places their mentions in multiple parts. The `flfn+ln` partitioning scheme avoids such issues by partitioning based on just the first letter of the first name concatenated with the last name. Though this scheme matches mentions that differ due to initialization, or due to the presence and absence of a middle name, it "over merges" authors in the process. The overall accuracy of `flfn+ln` is substantially lower than `name equality`. Part of the reason for this is that `MathSciNet` is so large: it contains publications spanning many areas, and many authors working in unrelated areas have the same name. In `msc2+name equality`, the partition is induced by the author name appended with a high-level subject code (e.g., MSC60: Probability). Unfortunately, this too fails to improve on `name equality`: too many authors write papers that span multiple subjects. More sophisticated disambiguation algorithms are required.

4.3 A feature-space representation of records for efficient disambiguation without blocking

Disambiguation algorithms typically compare every pair of records that could be coreferent. Thus, for computational reasons, even moderately sized datasets are often grouped into (possibly overlapping) blocks, and only references assigned to the same block are compared. For example, for person names, records (i.e., mentions) may be grouped by the first letter of the first name together with the full last name. But blocking has two major drawbacks:

merges across blocks are not possible, and blocks' sizes typically still grow quadratically in the size of the full dataset.

This section proposes an alternative to blocking. Rather than assigning records to blocks, a graph encodes the pairs of records that could conceivably be coreferent. The graph has a number of edges and vertices that is linear in the number of records. Only pairs of records connected by a short path must be considered. Our approach explicitly deals with missing fields (censoring) as well as corrupted entries. First, we develop a method for records in general, and then apply the method to person names. On a real dataset of person names, our method reduces the number of pairs of records to consider by a factor of 100, relative to a standard blocking procedure, without any loss of accuracy.

The standard technique of disambiguation within a block, or without any blocking (which compares every pair of mentions), may be viewed as a kernel method. Like kernel methods, it has at least quadratic runtime. Our proposed approach is based on a feature space representation of the mentions. Our approach has runtime linear in the dimension of the feature representation, which is bounded and low. Our approach may be thought of as exploiting the "inverse" of the kernel trick: rather that operating in an implicit feature space to determine which pairs of mentions have support, as a kernel method would, we explicitly form a low-dimensional feature vector for each mention.

4.3.1 Notation and terminology

A record $r : K_r \to V_r$ maps field names (keys) to values. Let $r|_K$ denote the restriction of r to K. Let

$$Q(r) := \{(r|_K, K_r \setminus K) : K \subset K_r\} \tag{4.1}$$

For any record r, $q \in Q(r)$ is a *censored record.* In words, a censored record contains a subset of the fields of the original record, along with the field names that were excluded. Let T be a symmetric relation on censored records. Then T induces a symmetric relation on records too: Records r_1 and r_2 are T*-conceivable* if there exist $q_1 \in Q(r_1)$ and $q_2 \in Q(r_2)$ satisfying $q_1 T q_2$.

In practice, we select T so that only T-conceivable records are coreferent; T governs what discrepancies are minor enough that the pair of records could nonetheless refer to the same author or entity.

4.3.2 Efficiently enumerating T-conceivable record pairs

Let M be a set of records. Set

$$A := \bigcup_{r \in M} Q(r). \tag{4.2}$$

In words, A is all the censored records of the records M. Let

$$X := \{(r, q) \in M \times A : q \in Q(r)\}. \tag{4.3}$$

In words, X consists of the edges connecting each record to its censored record. Let

$$B \coloneqq \{(q_1, q_2) \in A \times A : q_1 T q_2\}. \tag{4.4}$$

In words, B maps censored records to other censored records that could refer to the same person, according to T.

Let the graph G have vertices $V = M \cup A$ and (undirected) edges $E = X \cup B$. (Remark: $M \subset V$ is an independent set in G.) By construction, records $r_1, r_2 \in M$ are *T-conceivable* iff they are connected in G by a path of length 3. Moreover, $|V|$ and $|E|$ are linear in $|M|$. Therefore, T-conceivable records can be efficiently enumerated.

4.3.3 Person name disambiguation

For person name disambiguation, records are mentions. A mention r must have a last name and a first initial. r may also have a first name, a middle name, and a middle initial. If a first name is present, so is a first initial that agrees with the first name; if a middle name is present, so is a middle initial that agrees with the middle names. Figure 4.6 shows the graph G for 4 mentions.

Let T include all the relationships suggested by all the name variations detailed in Section 4.2.1. In other words, T relates mentions to each possible 1 and 2 character spelling mistake (i.e., addition, substitution, and deletion), as well as all the other types of name variations (e.g., drop hyphenated part of last name). We omit certain relations from T, such as those that initialize a first or middle name, or that change the first letter of an initialized name. These transformations generate an excessive number of false positive matches.

Table 4.1 lists results. The results show that one proposed criterion (conceivability) attains nearly the same recall as a coarse blocking scheme based on partitioning two initials (first name and last name initials), while reducing the number of pairs of mentions to consider by 2 orders of magnitude. Moreover, while the common `flfn-ln` (first letter of the first name, concatenated with the last name) blocking scheme incorrectly splits 4.8% of authors among blocks—errors that cannot be recovered from subsequently—`conceivability` (connected authors) splits just 2.4% of authors, effectively halving the rate of `flfn-ln`, all while making a modest reduction in the number of author pairs to consider.

4.4 AuthorshipToolkit

This section describes AuthorshipToolkit, a novel approach to author disambiguation. The feature-space representation of mentions from Section 4.3 limits the number of mention pairs to consider. AuthorshipToolkit uses agglomerative clustering to partition the test data: Initially, all mentions belong to singleton clusters. Then the clusters are merged iteratively, in a greedy fashion, until some stopping criterion is met. At each iteration, potential merges are assessed based on a model of the probability that a pair of clusters is coreferent, given that each cluster contains only coreferent mentions.

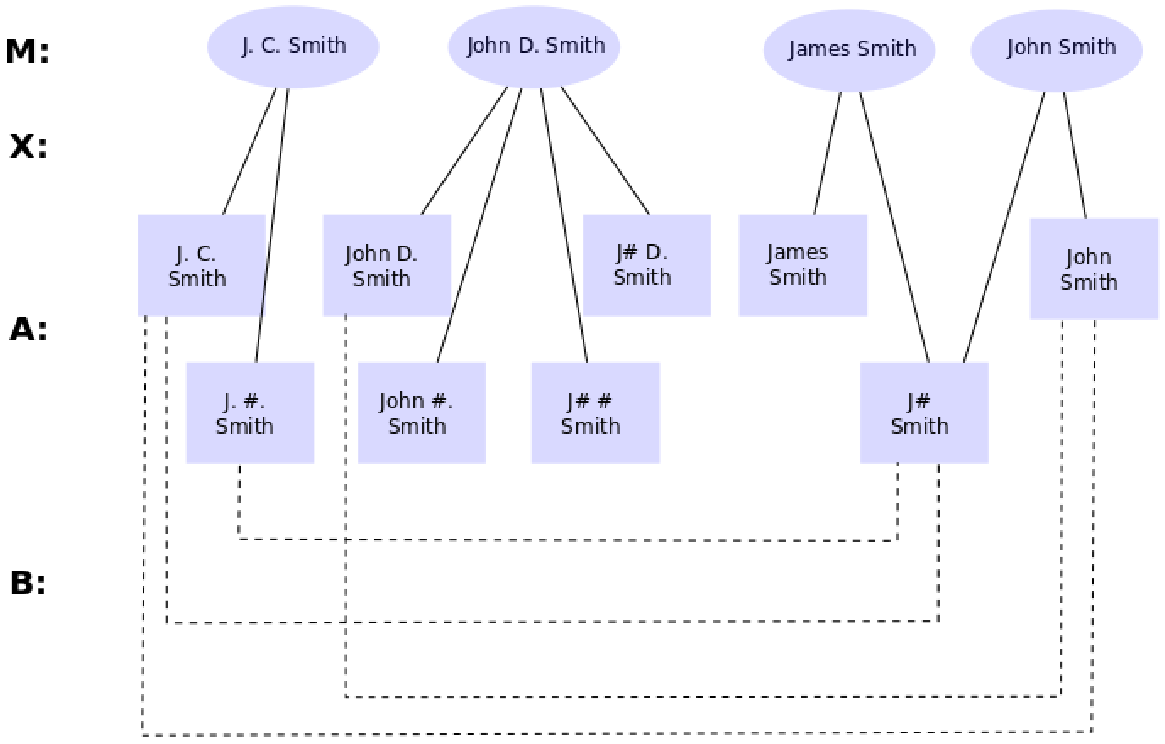


Figure 4.6: An example of G for 4 mentions. The mentions (records) are represented by vertices M. Censored records are vertices A. Here the # sign indicates a censored field. Edges X link the mentions to their censored records. Edges B connect censored records that could have been assigned to the same author, given the set T of name string transformations to consider. In this example, we set T to account for initialization of the first name and middle name, and omission of the middle name.

While each merge is guided by a probabilistic model, the overall clustering procedure is not based on any explicit probabilistic model of the full dataset. Thus, AuthorshipToolkit is a heuristic, despite having some grounding in probability. The AuthorshipToolkit serves as an advanced baseline procedure for comparison with the model-based approaches in Chapter 5 and Chapter 6. Additionally, it illustrates some benefits and limitations of heuristics for author disambiguation.

4.4.1 A probabilistic similarity metric

A probabilistic similarity metric resembling a naive Bayes classifier guides the agglomerative clustering procedure. Let M be a Bernoulli random variable indicating that two clusters, A_1 and A_2, contain only mentions that refer to the same author, given that the mentions in A_1

	# pairs to consider	recall		
		pairwise	clique	connected
		out of 69,442,989 pairs	out of 512,506 authors	
no blocking	5,662,691,033,860	100%:	100%	
flfn-flln	17,200,258,884	98.2%	97.6%	
flfn-ln	257,756,359	96.0%	95.2%	
conceivability	219,271,291	98.1%	96.5%	97.6%
compatibility	89,556,833	92.6%	91.1%	92.4%

Table 4.1: Recall and computational burden for `MathSciNet` for various schemes of selecting mention pairs to consider for disambiguation. Each row corresponds to a different scheme for selecting mention pairs. The first row ("no blocking") considers all pairs. The second row ("flfn-flln") considers pairs sharing first and last initials. The third row ("flfn-ln") considers pairs sharing a first initial and a last name. The fourth row ("conceivability") is the proposed method, with T containing all discussed transformations. The fifth row ("compatibility") is the proposed method with T that accounts for initialization and omission of the middle name. Recall is true positives: predicted coreferent pairs (resp. clusters) considered as a proportion of actual coreferent pairs (resp. authors). A "clique author" is a cluster of mentions where all pairs are joined a path of length 3. A "connected author" is a cluster of mentions where a path joins every pair. A cluster must exactly match an author to be marked correct.

all refer to one author and the mentions in A_2 all refer to one author. By Bayes' rule,

$$\mathbb{P}(M|A_1, A_2) \propto \mathbb{P}(A_1, A_2|M)\mathbb{P}(M).$$

Computing the normalization constant for the right-hand side is tractable because M is binary. The agglomerative clustering procedure terminates when $P(M = 1|A_i, A_j) < P(M = 0|A_i, A_j)$ for all $i \neq j$.

The marginal probability of a match, $\mathbb{P}(M)$, is approximated by dividing the number of authors in the dataset by the number of clusters at the current iteration. The number of authors in the dataset, in turn, is approximated by dividing the number of mentions in the dataset by an estimate of the average number of articles published by authors in the dataset.

We model the likelihood of generating two clusters, A_1 and A_2, as

$$\mathbb{P}(A_1, A_2|M) = \prod_{i=1}^{F} \mathbb{P}(f_i(A_1, A_2)|M)$$

The f_i compute features of the pair of clusters. We select feature functions that summarize aspects of the data thought to be largely unrelated, given M, to minimize any model misspecification stemming from our assumption of independence. Our choice of features, and the class-conditional likelihoods we assign to these features, account for several factors.

4.4.1.1 Name frequencies

Mentions of relatively rare names are more likely to refer to the same author than mentions of more common names. We use a large external dataset of person names to fit the model. Our model consists of several lookup tables for different name parts. Each row lists the proportion of the authors in the external dataset with that name. One table lists first names, along with proportions. Another table lists last names, along with proportions. A third table list pairs of first name and last names that are common, and that show up much more or much less frequently than would be expected if first and last names were independent, according to the product of the corresponding entries in the other two tables.

We use these empirical distributions, along with an estimate of the number of authors in the dataset to estimate the prior probability that two mentions with the same name string refer to the same author. Alternatively, we can discretize this real-valued feature: mention pairs may share a name string that is either very rare, rare, common, or very common. We learn the probabilities for each level from training data.

We also update these name priors at test time, to account for additional information in the test set. While our external training set of author names may suggest that there is at most one author in our test set named "A. Zinheizter," if our test set contains both "Aaron Zinheizter" and "Adam Zinheizter," then our estimate of the number of distinct A. Zinheizter's should be at least two.

4.4.1.2 Misspellings and other name-string transformations

Names that do not match exactly can nonetheless refer to the same author in our model if some sequence of transformations connects them (Section 4.3). The sequence of transformations that makes a pair of mentions' name strings equal is a statistic of the pair. If mentions could share a particular name only through some sequence of transformations, then the prior probability that two authors with exactly that name string are coreferent should be revised downward to account for the necessary transformations.

4.4.1.3 Co-authorship network

Each mention is associated with a set of co-authors from the article it appears in, though this set is unknown a priori. If two mentions share many co-authors, then they are more likely refer to the same author. Though at disambiguation time we do not have access to mentions' co-authors, we may approximate it based on either a partition of the mentions made without use of co-author information or simply the partition of the mentions induced thus far in an iterative algorithm. In either case, by applying Bayes's rule, we can incorporate co-author information into our estimate. In practice, we find that if at least several co-authors are shared, two mentions are virtually certain to be the same. Thus, it is not necessary to consider cases where more than a few co-authors are shared. For $n \in 0, 1, 2, 3$, for both $M = 0$ and $M = 1$, we estimate

$$\mathbb{P}(n \, \text{shared co-authors} \,|\, M)$$

from a training dataset similar in structure to our test set. The probability of different author entities sharing n co-authors is readily computed from a manually disambiguated training set, either by sampling or with a brute force approach. The probability of two mentions that refer to the same entity sharing n co-authors is also readily computed from a training set.

Though readily estimated, these probabilities are not perfectly suited to clustering. Consider an extreme case: Initially every mention is assigned to a singleton cluster. According to this partition, no one co-authors papers with someone who also co-authored a different paper with someone else, and therefore no pairs of mentions have shared co-authors. Towards the end of the clustering procedure, once clusters are starting to resemble actual authors, the estimates may become reasonable. But no fixed set of conditional probabilities remains valid throughout the procedure.

To mitigate this problem, we perform clustering in two phases with two different similarity metrics. During the first phase (bootstrapping), our similarity metric is based solely on attribute similarity. The mentions with the rarest, most unambiguous names are merged. After this phase, co-authorship becomes meaningful, though some authors are still spread across more than one cluster. During the second phase, attribute similarity is updated based on the current co-authorship network.

4.4.2 Results

We apply AuthorshipToolkit to both of our manually disambiguated datasets: `arXiv HEP` (Figure 4.7) and `MathSciNet` (Figure 4.8). Results on `arXiv HEP` are reported in terms of the harmonic mean of pairwise precision and recall (pairwise F-measure) for comparison with other publications. (ROC curves typically are not reported in author disambiguation literature, thus limiting their usefulness for comparing to existing work.) For the metric we consider, AuthorshipToolkit reduces error by one third relative to [66]. These results are encouraging, though we designed the AuthorshipToolkit with `arXiv HEP` in mind. That said, AuthorshipToolkit has few parameters, most of which were set in a way that should generalize to any subset of arXiv.

Results on `MathSciNet` are reported in terms of accuracy: the proportion of authors disambiguated perfectly. Though not sensitive to some changes in the disambiguation, accuracy has the advantage of being easily interpreted. For `MathSciNet`, the test data was not examined before testing, and the test set is so large anyway that it would be overfit. In this high ambiguity dataset, we see that AuthorshipToolkit outperforms all baselines by a large margin. Also, we see that there is much room for improvement.

4.4.3 Limitations

AuthorshipToolkit accounts for many aspects of author disambiguation. However, it fails to fully exploit available information. Modeling the probability that two mention clusters are coreferent, rather than the probability of a particular partition of the mentions, simplifies inference in some respects, but makes it difficult to account for many features in a principled

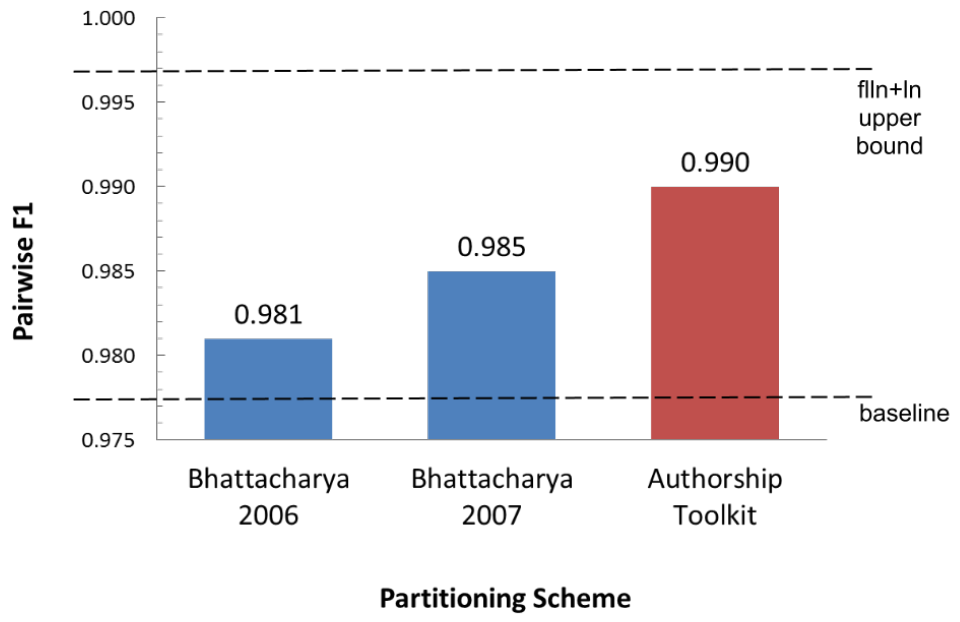


Figure 4.7: Results for `arXiv HEP`, a dataset with low name ambiguity. The `fifn+ln upper bound` line indicates the best score that could be attained without merging mentions that do not have the same first initial and last name. The `baseline` line represents partition by name string, after basic preprocessing.

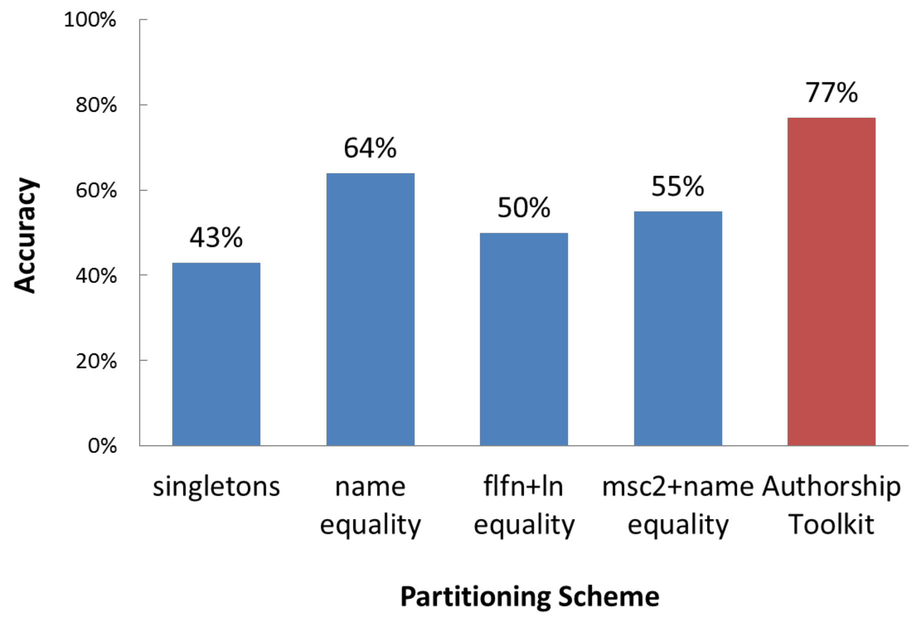


Figure 4.8: Results for `MathSciNet`, a large dataset with high name ambiguity. The blue bars show results for the baseline procedures introduced in Figure 4.5.

way. To learn probabilities from training data, we assume in some cases that the probability that two clusters are coreferent is similar to the probability that two mentions with certain properties are coreferent. But that only holds exactly when each cluster is a singleton, not throughout the clustering process. In other cases, we approximate probabilities about pairs of clusters as if each cluster contained all the references for a particular author, but this is unlikely to be the case.

Incorporating features about clusters other than the pair which may be merged is particularly problematic: we can form features based on the other clusters in the partition. But the probability of observing these features, conditional upon whether the pair of clusters is coreferent, is unknown. Any estimate likely would not remain constant throughout the clustering procedure. We have no means of learning how such conditional probabilities vary during a greedy, heuristic clustering procedure.

To correct these shortcomings, the next chapter develops an approach to author disambiguation that explicitly models the probability of any partition of the mentions.

Chapter 5

Conditional random fields for author disambiguation

This chapter presents a probabilistic model for a partition of mentions. Higher probabilities are assigned to partitions that correctly disambiguate mentions. The model is discriminative rather than generative: it models the probability the partition is correct directly, wihout also modeling the probability of the data. Inference is performed by a Metropolis-Hastings sampler. For parameter learning, we consider both SampleRank [89] and contrastive divergence [90].

5.1 Model

Let M be a set of mentions. Let $\mathcal{F} \equiv \{f_1, \dots, f_n\}$ be a collection of feature functions, each mapping a set of mentions to a real value. Let θ be an $(n+1)$-dimensional real-valued vector. Let A be a partition of M.

We model the conditional probability given M that A is the correct disambiguation of the authors as

$$\mathbb{P}(A;\theta) \equiv \frac{1}{Z(\theta)} \prod_{E \in A} \Psi(E) \tag{5.1}$$

where

$$\Psi(E) \equiv \exp\{\theta_0 + \sum_{i=1}^{n} \theta_i f_i(E)\} \tag{5.2}$$

and

$$Z(\theta) \equiv \sum_{A'} \prod_{E \in A'} \Psi(E). \tag{5.3}$$

Here $E \in A$ denotes a set of mentions that is one part in the partition A. Because we condition on M, and use an explicit model for the probability of the latent variables, our model is discriminative. Because the model is a product of functions of subsets of our latent variables, it can be conceived of as an undirected graph. Also, our model is log-linear. Thus, our model defines a conditional random field (CRF) [91, 92]. Let $G = (V, E)$ be a graph such that $\mathbf{Y} = (\mathbf{Y}_v)_{v \in V}$, so that $\mathbf{Y}$ is indexed by the vertices of G. Then $(\mathbf{X}, \mathbf{Y})$ is a conditional random field, since, conditioning on $\mathbf{X}$, the random variables $\mathbf{Y}_v$ obey the Markov property with respect to the graph:

$$p(\mathbf{Y}_v | \mathbf{X}, \{\mathbf{Y}_w : w \neq v\}) = p(\mathbf{Y}_v | \mathbf{X}, \{\mathbf{Y}_w, w \sim v\}), \tag{5.4}$$

where $w \sim v$ means that w and v are neighbors in G [93]. CRFs have been successfully applied to many problems [81, 60, 93, 83, 87].

We restrict the form of the feature functions. For $i = 1, \dots, n$, there must exist ϕ_i s.t. for any $\{m_1, \dots, m_p\} \subset M$,

$$f_i(\{m_1, \dots, m_p\}) = \sum_{j=1}^{p} \phi_i(m_j, \{m_1, \dots m_{j-1}\}). \tag{5.5}$$

In words, a feature function f_i imposes an arbitrary ordering on entities' mentions, and decomposes additively over successively longer sequences of mentions. We restrict f_i to functions that are invariant under re-orderings of their arguments. The feature functions in $\mathcal{F}$ are meant to measure the qualities of a good partition of M.

5.1.1 Feature functions

Our model judges latent-variable configurations by the kinds of properties first introduced in Section 4.4.1. We compute the follow classes of features:

Name commonality. Mentions with rare names are more likely refer to the same author. Name frequencies are computed based on the common part of compatible name strings and then discretized into 5 levels.

Topic similarity. Mentions from papers about the same topic are more likely to refer to the same author. Topic is determined by Mathematics Subject Classification (MSC) code, a 5-digit code with some structure: the first 2 digits indicate a high-level subject (e.g., probability), the first 3 digits indicate a mid-level subject (e.g., Markov processes), the full code indicate a low-level subject (e.g., Brownian motion). MSC codes are tested for equality on their first 2 characters, their first 3 characters, and all 5 characters.

Name string compatibility. Mentions with character-equivalent name strings most likely to refer to the same person, but truncation of first or middle names, or omission of middle names, may only lower the probability of a match slightly.

Misspelling likelihood. Some spelling errors are more likely than others. Mentions that require unusual spelling errors to be coreferent are least likely to match.

Coauthorship network Authors tend to work with each other repeatedly, and to select co-authors from their community. To encourage these configurations, we include a feature function that reports the number of co-authors for each author. In the future, we may want to either 1) introduce latent variables representing academic communities, and favor configurations where authors have fewer communities; or 2) favor configuration with fewer distinct co-authors among authors with similar names.

5.1.2 Computationally efficient relative probabilities

During both inference and learning, we compute the relative probabilities of partitions of mentions that differ only slightly. In particular, suppose that A_1 and A_2 are partitions of M satisfying

$$A_1 = \{E_1 \cup \{m\}, E_2, E_3, \dots E_d\} \tag{5.6}$$

and

$$A_2 = \{E_1, E_2 \cup \{m\}, E_3, \dots, E_d\}. \tag{5.7}$$

In words, A_1 and A_2 differ only by the part containing a particular mention. Then we can efficiently compute the relative probability as

$$\begin{aligned}\frac{\mathbb{P}(A_2)}{\mathbb{P}(A_1)} &= \frac{\Psi(E_1)}{\Psi(E_1 \cup \{m\})} \cdot \frac{\Psi(E_2 \cup \{m\})}{\Psi(E_2)} && (5.8)\\ &= \exp\{\sum_{i=1}^{n} \theta_i [f_i(E_1) - f_i(E_1 \cup \{m\}) + f_i(E_2 \cup \{m\}) - f_i(E_2)]\} && (5.9)\\ &= \exp\{\sum_{i}^{n} \theta_i [\phi_i(m, E_2) - \phi_i(m, E_1)]\}. && (5.10)\end{aligned}$$

5.2 Parameter learning

The model contains a parameter θ with dimension equal to the cardinality of $\mathcal{F}$. To learn this parameter, we consider 1) maximum likelihood estimation, 2) contrastive divergence [90], and 3) SampleRank [89].

5.2.1 Maximum likelihood estimation

Maximum likelihood estimation is the standard approach to estimating θ. However, due to the normalization term, computing the MLE of our model is intractable. To see this, consider the log likelihood:

$$\begin{aligned}\ell(A; \theta) &\equiv \log \mathbb{P}(A; \theta) && (5.11)\\ &= \sum_{E \in A} \sum_{i=1}^{n} \theta_i f_i(E) - \log \sum_{A'} \prod_{E \in A'} \prod_{i=1}^{n} \exp\{\theta_i f_i(E)\}. && (5.12)\end{aligned}$$

The derivative of the log likelihood is

$$\frac{\partial}{\partial\theta_i}\ell(A;\theta) = \sum_{E\in A} f_i(E) - \frac{\partial}{\partial\theta_i}\log\sum_{A'}\prod_{E\in A'}\prod_{j=1}^{n}\exp\{\theta_j f_j(E)\} \tag{5.13}$$

$$= \sum_{E\in A} f_i(E) - \frac{\frac{\partial}{\partial\theta_i}\sum_{A'}\prod_{E\in A'}\prod_{j=1}^{n}\exp\{\theta_j f_j(E)\}}{\sum_{A'}\prod_{E\in A'}\prod_{j=1}^{n}\exp\{\theta_j f_j(E)\}} \tag{5.14}$$

$$= \sum_{E\in A} f_i(E) - \frac{\sum_{A'}\frac{\partial}{\partial\theta_i}\prod_{E\in A'}\prod_{j=1}^{n}\exp\{\theta_j f_j(E)\}}{Z(\theta)} \tag{5.15}$$

$$= \sum_{E\in A} f_i(E) - \sum_{A'}\frac{1}{Z(\theta)}\left[\prod_{\substack{j=1\\ j\neq i}}^{n}\prod_{E\in A'}\exp\{\theta_j f_j(E)\}\right]\frac{\partial}{\partial\theta_i}\prod_{E\in A'}\exp\{\theta_i f_i(E)\} \tag{5.16}$$

$$= \sum_{E\in A} f_i(E) - \sum_{A'}\frac{1}{Z(\theta)}\left[\prod_{\substack{j=1\\ j\neq i}}^{n}\prod_{E\in A'}\exp\{\theta_j f_j(E)\}\right]\frac{\partial}{\partial\theta_i}\exp\{\theta_i\sum_{E\in A'} f_i(E)\} \tag{5.17}$$

$$= \sum_{E\in A} f_i(E) - \sum_{A'}\frac{1}{Z(\theta)}\left[\prod_{E\in A'}\prod_{j=1}^{n}\exp\{\theta_j f_j(E)\}\right]\sum_{E\in A'} f_i(E) \tag{5.18}$$

$$= \sum_{E\in A} f_i(E) - \sum_{A'}\mathbb{P}(A';\theta)\sum_{E\in A'} f_i(E) \tag{5.19}$$

$$= \sum_{E\in A} f_i(E) - \mathbb{E}_{A'}\left[\sum_{E\in A'} f_i(E)\right]. \tag{5.20}$$

Unfortunately it is difficult to reliably estimate the expectation in the second term; even if correctly distributed samples $A_1,\ldots A_m$ can be drawn, presumably via some MCMC method, the variance among $\sum_{E\in A_k} f_i(E)$ due to θ will be swamped by the variance of $|A_k|$ [90].

5.2.2 Contrastive divergence

Finding parameters θ that maximize the likelihood is equivalent to minimizing the Kullback-Leibler divergence between the empirical distribution of the data, P^0, and the modeled distribution of the data. The latter distribution is termed the equilibrium distribution, denoted $P_\theta^\infty \equiv \mathbb{P}$; it is the distribution sampled from by an MCMC sampler run infinitely long [90]. However, it is no easier to compute gradients of $D_{KL}(P^0\|P_\theta^\infty)$:

$$-\frac{\partial}{\partial\theta_i}D_{KL}(P^0\|P_\theta^\infty) = -\frac{\partial}{\partial\theta_i}\mathbb{E}_{P^0}\left[\log\frac{P^0(A)}{P_\theta^\infty(A)}\right] \tag{5.21}$$

$$= \mathbb{E}_{P^0}\left[-\frac{\partial}{\partial\theta_i}\log\frac{P^0(A)}{P_\theta^\infty(A)}\right] \tag{5.22}$$

$$= \mathbb{E}_{P^0}\left[\frac{\partial}{\partial\theta_i}\log P_\theta^\infty(A)\right] \tag{5.23}$$

$$= \mathbb{E}_{P^0}\left[\frac{\partial}{\partial\theta_i}\ell(A;\theta)\right] \tag{5.24}$$

$$= \mathbb{E}_{P^0}\left[\sum_{E\in A} f_i(E) - \mathbb{E}_{P_\theta^\infty}\left[\sum_{E\in A'} f_i(E)\right]\right] \tag{5.25}$$

$$= \mathbb{E}_{P^0}\left[\sum_{E\in A} f_i(E)\right] - \mathbb{E}_{P_\theta^\infty}\left[\sum_{E\in A} f_i(E)\right]. \tag{5.26}$$

Fortunately, minimizing $D_{KL}(P^0\|P_\theta^\infty)$ is equivalent to minimizing the difference between $D_{KL}(P^0\|P_\theta^1)$ and $D_{KL}(P_\theta^1\|P_\theta^\infty)$, where P_θ^1 is the distribution obtained by running Gibbs sampling – initialized with P_0—for a single iteration. Minimizing the latter objective function is more tractable, since

$$-\frac{\partial}{\partial\theta_i}D_{KL}(P_\theta^1\|P_\theta^\infty) = -\frac{\partial}{\partial\theta_i}\mathbb{E}_{P_\theta^1}\left[\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right] \tag{5.27}$$

$$= -\frac{\partial}{\partial\theta_i}\sum_A\left[P_\theta^1(A)\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right] \tag{5.28}$$

$$= -\sum_A\frac{\partial}{\partial\theta_i}\left[P_\theta^1(A)\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right] \tag{5.29}$$

$$= -\sum_A\frac{\partial}{\partial\theta_i}\left[P_\theta^1(A)\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right] \tag{5.30}$$

$$= -\sum_A\left[\left(\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right)\frac{\partial}{\partial\theta_i}P_\theta^1(A) + \left(P_\theta^1(A)\frac{\partial}{\partial\theta_i}\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right)\right] \tag{5.31}$$

$$= -\sum_A\left[\left(\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right)\frac{\partial P_\theta^1(A)}{\partial\theta_i}\right] - \mathbb{E}_{P_\theta^1}\left[\frac{\partial}{\partial\theta_i}\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right] \tag{5.32}$$

$$= -\sum_A\left[\left(\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right)\frac{\partial P_\theta^1(A)}{\partial\theta_i}\right] - \mathbb{E}_{P_\theta^1}\left[\frac{\partial}{\partial\theta_i}\log P_\theta^1(A)\right] + \mathbb{E}_{P_\theta^1}\left[\frac{\partial}{\partial\theta_i}\log P_\theta^\infty(A)\right] \tag{5.33}$$

$$= -\sum_A\left[\left(\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right)\frac{\partial P_\theta^1(A)}{\partial\theta_i}\right] + \mathbb{E}_{P_\theta^1}\left[\frac{\partial}{\partial\theta_i}\log P_\theta^\infty(A)\right] \tag{5.34}$$

$$= \mathbb{E}_{P_\theta^1}\left[\sum_{E\in A} f_i(E)\right] - \mathbb{E}_{P_\theta^\infty}\left[\sum_{E\in A} f_i(E)\right] - \sum_A\left[\left(\log\frac{P_\theta^1(A)}{P_\theta^\infty(A)}\right)\frac{\partial P_\theta^1(A)}{\partial\theta_i}\right] \tag{5.35}$$

$$= \mathbb{E}_{P_\theta^1}\left[\sum_{E\in A} f_i(E)\right] - \mathbb{E}_{P_\theta^\infty}\left[\sum_{E\in A} f_i(E)\right] - \frac{\partial P_\theta^1}{\partial\theta_i}\frac{\partial D_{KL}(P_\theta^1\|P_\theta^\infty)}{\partial P_\theta^1}, \tag{5.36}$$

and therefore the expectations over P_θ^∞ cancel:

$$\begin{aligned}
&-\frac{\partial}{\partial\theta_i}[D_{KL}(P^0\|P_\theta^\infty) - D_{KL}(P_\theta^1\|P_\theta^\infty)] \\
&\quad = \left\{-\frac{\partial}{\partial\theta_i}D_{KL}(P^0\|P_\theta^\infty)\right\} - \left\{-\frac{\partial}{\partial\theta_i}D_{KL}(P_\theta^1\|P_\theta^\infty)\right\} \qquad (5.37) \\
&\quad = \mathbb{E}_{P^0}\left[\sum_{E\in A} f_i(E)\right] - \mathbb{E}_{P_\theta^1}\left[\sum_{E\in A} f_i(E)\right] + \frac{\partial P_\theta^1}{\partial\theta_i}\frac{\partial D_{KL}(P_\theta^1\|P_\theta^\infty)}{\partial P_\theta^1}. \qquad (5.38)
\end{aligned}$$

Computing the leftmost term is straightforward, and the middle term can be reliably approximated through Gibbs sampling. The rightmost term supposedly does not have much effect and may be ignored [90].

However, it seems a bit odd to frame the minimization in terms of a difference between KL divergences, since upon ignoring the third term, the objective function is simply $D_{KL}(P^0\|P_\theta^1)$.

Regardless, this approximation of the derivative suggests the following update step:

$$\Delta\theta_i^{(t)} \propto \mathbb{E}_{P^0}\left[\sum_{E\in A} f_i(E)\right] - \mathbb{E}_{P_\theta^1}\left[\sum_{E\in A} f_i(E)\right]. \qquad (5.39)$$

Let A^* be the correct disambiguation of the training data. Since P^0 is just a point mass at A^*, our single observation of a correction disambiguation,

$$\Delta\theta_i^{(t)} \propto \mathbb{E}_{P_\theta^1}\left[\sum_{E\in A^*} f_i(E) - \sum_{E\in A} f_i(E)\right]. \qquad (5.40)$$

Contrastive divergence generates samples from P_θ^1 by updating each latent variable via a single iteration of a Gibbs sampler. However, any definition of P_θ^1 should suffice, as long as P_θ^1 is the distribution of some non-trivial step of a Markov chain initialized to A^*, converging to P_θ^∞. Thus, Metropolis-Hastings sampling may be substituted for Gibbs sampling. Also, as suggested by [89], P_θ^1 may be a single Metropolis-Hastings step, rather than an update of all latent variables. The latent variables in our model may be parameterized in three ways: as $\frac{|M|(|M|-1)}{2}$ binary coreference indicators, as $|M|$ integers, or as one partition. If we take our latent variables to be $|M|$ integers, we get the following update procedure: Let

$$A^* = \{E_1 \cup \{m\}, E_2, E_3, \ldots E_d\}, \qquad (5.41)$$

where the parts' ordering is arbitrary. Then samples from P_θ^1 have the form

$$\hat{A} = \{E_1, E_2 \cup \{m\}, E_3 \ldots E_d\}. \qquad (5.42)$$

For such samples,

$$\begin{aligned}
\Delta\theta_i^{(t)} &\propto [f_i(E_1 \cup \{m\}) - f_i(E_1)] + [f_i(E_2) - f_i(E_2 \cup \{m\})] \qquad (5.43) \\
&= \phi_i(m, E_1) - \phi_i(m, E_2). \qquad (5.44)
\end{aligned}$$

Contrastive divergence maximizes the joint probability of observed and unobserved variables, without access to complete data. However, our model is discriminative, and our training data is fully observed. This simplifies sampling from P_θ^1. To apply the ideas from [90], we consider the attributes of the mentions to be fixed exogenously, rather than conditioned upon. Then the data vector is A^*. We get our "reconstructed" data vector $\hat{A}$ directly, by running a sampler initialized to A^*.

5.2.3 SampleRank

SampleRank was developed to improve on contrastive divergence (CD) [94, 89]. Like CD, SampleRank learns weights from samples drawn by Metropolis-Hastings. With CD, the sampler's Markov chain is reset to the ground truth whenever the parameters are updated, whereas with SampleRank, when the parameters are updated, the Markov chain continues. SampleRank evaluates a loss function on each sample, and trains a classifier to rank samples according to the loss. While similar in form to CD, SampleRank appears to lack CD's theoretical justification and convergence guarantees. Nonetheless, the ability to optimize for an arbitrary loss function makes SampleRank an approach to consider.

5.3 Inference

Upon learning the parameters from training data, either by CD or SampleRank, we collect samples from the posterior distribution by running a Metropolis-Hastings sampler on the test set. Proposed moves transfer one mention at a time between authors, though split-merge proposals [95] could improve runtime, and may be necessary in practice. Otherwise the sampler may need to pass through many low-probability states to get from one likely partition to another. Proposal canopies [96], another way to improve runtime, essentially amount to blocking. The approach in Section 4.3 is an even more efficient alternative to blocking, and could be employed.

We consider several approaches to combining multiple samples, to obtain a single "average" partition of the mentions. We can solve for either the modal partition or the "median" partition. To solve for the mode, we introduce a "temperature" parameter to the proposal distribution. The temperature is gradually lowered, over the iterations, leading the sampler to configurations with higher probability. (Lower temperatures make moves to lower probability samples less frequent.) To solve for a median partition rather than the mode, consensus clustering [97] may be applied to combine MCMC samples. Finally, if we just want to determine whether a particular set of mentions is probably coreferent, we can count the fraction of samples that place them all in the same cluster.

5.4 Results

On synthetic data generated from the model, with highly informative feature functions, the MAP estimate of the partition recovers the true partition of the synthetic data. Inference works, at least when the feature functions are very informative and the assumptions of the model are met.

On real author data from `MathSciNet`, results are less encouraging. The parameters learned by CD assign high probability to the true partition of the training set, compared to partitions that are one Metropolis-Hastings step away from the truth. However, they assign even higher probability to degenerate partitions, i.e., where every mention is in a singleton cluster or all mentions are in one cluster.

With SampleRank, the parameters we learn optimize for a loss function rather than maximize the likelihood. The parameters learned by SampleRank also do not induce good partitions of even the training data, much less the test data: parameters can be manually "guessed" through trial and error that perform better than SampleRank even with respect to the loss function that SampleRank optimizes for. Because SampleRank lacks theoretical guarantees, it is hard to draw general conclusions about the problem from these empirical results.

The results from CD suggest that our model is misspecified. While our model leaves some parameters free, it also make some strong assumptions about the data. For example, consider a binary feature that indicates whether an author publishes in only one subject area. Presumably the parameter θ_i for this feature will be positive: partitions where more authors publish in only one area tend to be accurate. But if this were the only feature, a MAP partition would be all singleton clusters: every author has just one subject area. We might try encouraging merges by adding a feature that reports the size of each cluster. If CD assigns the parameter θ_j for this feature a positive weight, it may counteract the other feature's preference for small clusters, to some extent. But just counteracting the other feature does not make the model correct: a precise balance between the two effects is required, but the second feature can only scale the log probability linearly in the size of each cluster. There is no reason to think that *any* combination of θ_i and θ_j accurately models the data.

Chapter 6

Spectral author disambiguation

This chapter presents spectral disambiguation, a procedure for author disambiguation based on spectral clustering and a learned pairwise similarity metric. Section 4.3 proposes a feature-space representation of mentions and a technique for efficiently finding pairs of mentions with non-zero similarity. This chapter extends that technique, to not only find pairs of mentions with non-zero similarity, but to encode the similarity between all pairs, including those with non-zero similarity. We report on applying both a procedure to learn the similarity metric (the "learning procedure") and a procedure to partition the mentions (the "inference procedure") to `MathSciNet`.

Unlike AuthorshipToolkit (Section 4.4), spectral disambiguation is based on optimizing a loss function over partitions of the mentions. Unlike the approach of Chapter 5, spectral disambiguation is not probabilistic. Both AuthorshipToolkit and spectral disambiguation include greedy routines, but only the latter's greedy routine is based on optimizing an explicitly stated objective function.

6.1 Inference procedure

Each mention possesses a modest and bounded number (i.e., 10–100) of attributes, regardless of the number of mentions n to disambiguate. Attributes include things like name string and article keywords, as well as derived attributes, like possible misspellings of an author's name. The precise set of attributes is detailed in subsequent sections. Our choice of attributes follows from the form of our similarity matrix, described shortly, not the other way around. The total number of unique attributes p grows roughly linearly with n, though some attributes are shared by $O(n)$ mentions. Let the sparse binary $n \times p$ matrix X encode the attributes possessed by each mention. Let the non-negative sparse $p \times p$ matrix B encode attribute-attribute similarity. Let the mention-mention similarity matrix be

$$W = XBX^\top. \qquad (6.1)$$

Then the entries of W encode a Mahalanobis distance between pairs of mentions.

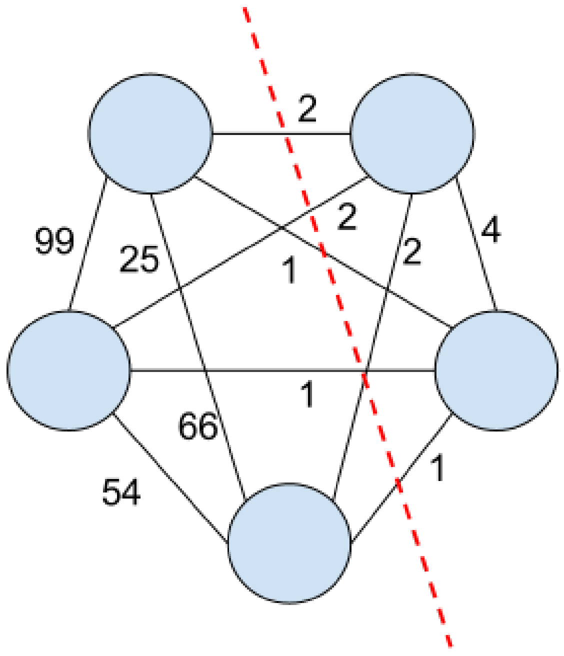


Figure 6.1: The `2-min-cut` of a weighted graph.

Inference amounts to finding a partition of the mentions that minimizes the edge weights (entries in W) that cross parts. This is known as the `k-min-cut` problem [98]. Figure 6.1 gives an example.

Solving `k-min-cut` is computationally intractable, but good approximations exist. K-way spectral clustering solves a relaxation of the `k-min-cut` objective [12]. Unfortunately, K-way spectral clustering has $O(n^3)$ runtime for author disambiguation because K is $O(n)$ for author disambiguation. Even with our parameterization of W as a product of sparse matrices with $O(n)$ fill, K-way spectral clustering has $O(n^2)$ runtime for fixed K. We need to determine K, too.

Recursive spectral bipartitioning [99, 100] instead solves a series of `2-min-cut` problems approximately, through 2-way spectral clustering. It recovers $O(n)$ clusters in $O(\log n)$ steps, as shown in Figure 6.2. Each step takes $O(n)$ runtime, so overall runtime is $O(n \log n)$—a scalable algorithm.

To compute each bipartition in $O(n)$ time, we find the eigenvector corresponding to the second smallest eigenvalue of the (unnormalized) Laplacian matrix

$$L = D - W$$

where D is a diagonal matrix with $D_{ii} = \sum_{j=1}^{n} W_{ij}$, i.e., $D = W\mathbf{1}$. For a vector v, we can compute

$$Lv = Dv - X\left(B\left(X^\top v\right)\right)$$

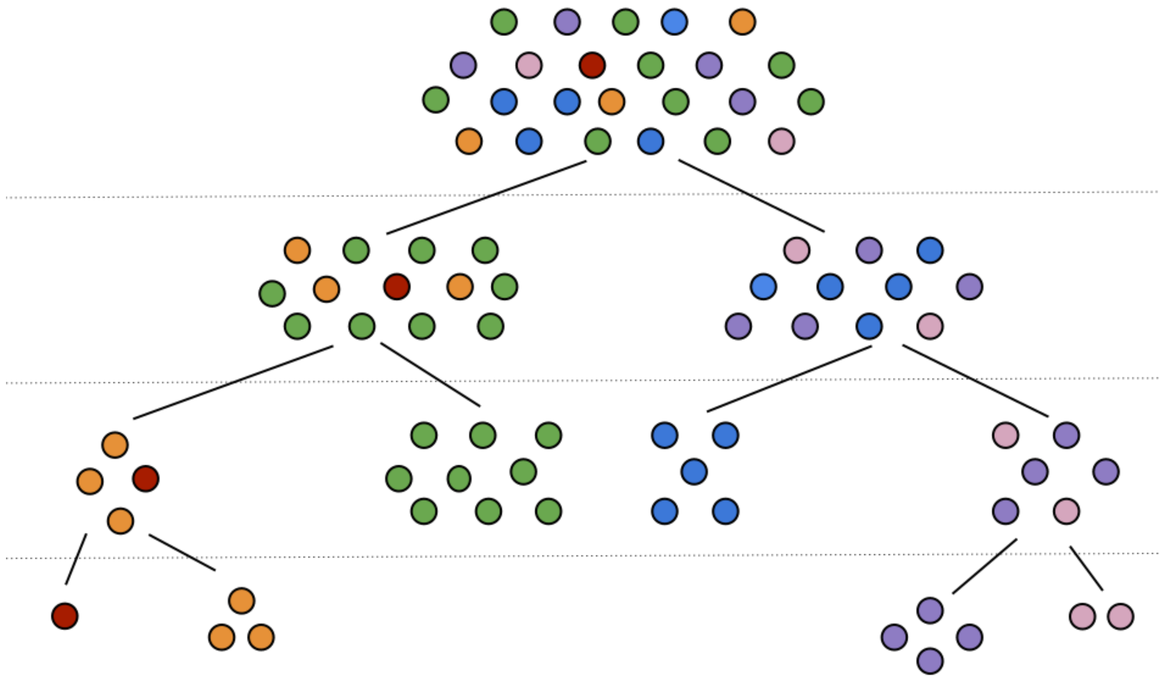


Figure 6.2: Recursive bipartition recovers $O(n)$ authors for n mentions in $O(\log n)$ steps, as long as the bipartitions are roughly balanced.

in $O(n)$ time by right-multiplying, as indicated by the parentheses. Using an Arnoldi method [101], and our formula for Lv, each bipartition can be computed rapidly [99]. We ensure that each bipartition is roughly balanced, so that the number of recurrences is at most $O(\log n)$. Then, the overall runtime of the inference procedure is $O(n \log n)$. Recursion stops when the average edge weight in the graph associated with Laplacian L cut by the proposed bipartition exceeds some threshold t.

6.1.1 Encoding mention pairs' similarities through Mahalanobis distance

Name strings are by far the most important attributes for author disambiguation. For now let us focus on mentions with only name strings as attributes. Some pairs of name strings are identical, and clearly $W = XBX^\intercal$ should assign these pairs non-zero similarity. Other pairs are not identical, but the difference could be exclusively due to censoring, e.g., "J. Smith" and "John Calhoon Smith." These pairs should also have non-zero similarity. Now consider the name strings "John Smith," "Jane Smith," and "J. Smith." Both "John Smith" and "Jane Smith" should have non-zero similarity with "J. Smith," but they should have zero similarity with each other—barring an egregious misspelling. Good spelling correction, and correction for other types of name mutations, is essential for good disambiguation. Table 6.1

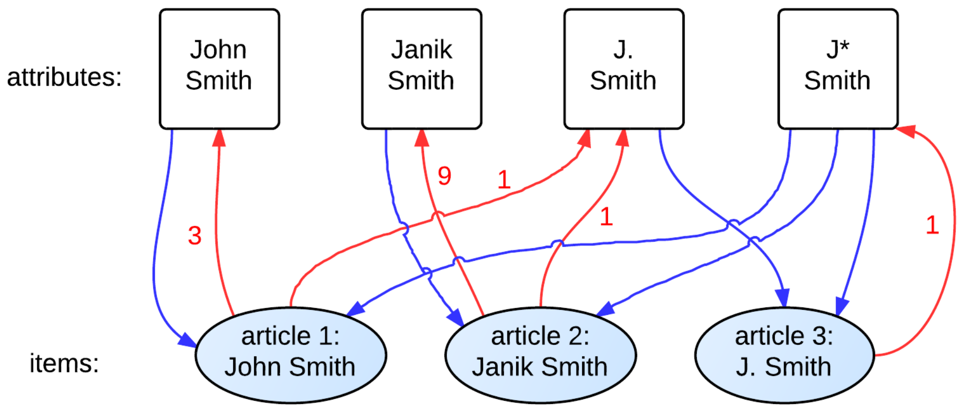


Figure 6.3: A graphical representation of similarity matrix $W == XBX^{\intercal}$, for 3 mentions. The blue edges correspond to the binary attribute-mention matrix X. The red edges correspond to the real-valued mention-attribute matrix $BX^{\intercal}$. Only mentions connected by a path of length 2 have non-zero similarity in W. For mentions connected by a path of length 2, their similarity is the weight along the red edge on the path.

lists the types of name pairs that $W = XBX^{\intercal}$ assigns non-zero similarity. This pattern of non-zero entries can be encoded by manipulating the entries of B, while maintaining sparsity. Figure 6.3 shows how entries of B can be set to encode real-valued similarities between name strings.

Pair type	Example
exact match	“Sally Smith” ∧ “Sally Smith”
compatible names	“R J Clark” ∧ “Ronald Clark”
hyphenated last name	“Jane Tyler-Nussbaum” ∧ “Jane Tyler”
truncated first name	“Yurii Nesterov” ∧ “Yu Nesterov”
first name dropped	“J. Carl Lovejoy” ∧ “Carl Lovejoy”
first and last names swapped	“Erin Kotsov” ∧ “Kotsov Erin”
nick name	“Jim Hunt” ∧ “James Hunt”
misspelling	“G Honderd” ∧ “G Khonderd”
modified word boundaries	“Xiao Fan Wang” ∧ “Xiaofan Wang”
compound mutation	“Xiao Fan Wang” ∧ “Xiaofan Wangg”

Table 6.1: Types of name string pairs with support

Now suppose mentions are associated with their articles keywords and subject codes, indicating the Mathematics Subject Classification (MSC). (MSC codes are introduced in

Section 5.1.1.) Then the B matrix is block diagonal with the following form:

$$B = \begin{bmatrix} B_{name} & 0 & 0 \\ 0 & B_{topic} & 0 \\ 0 & 0 & B_{keyword} \end{bmatrix}.$$

Matrix B_{name} indicates the similarity between name strings, as previously discussed.

Matrix B_{topic} indicates the similarity between topic codes. It is also a block diagonal matrix:

$$B_{topic} = \begin{bmatrix} B_{t2} & 0 & 0 \\ 0 & B_{t3} & 0 \\ 0 & 0 & B_{t5} \end{bmatrix}.$$

Matrices B_{t2}, B_{t3}, and B_{t5} indicate the affinities between top-level, mid-level and bottom-level MSC codes, respectively. For $j = 2, 3, 5$, B_{tj} has 1 row and 1 column for each k-digit topic code prefix (the full code if $j = 5$). For now,

$$B_{tj} = \alpha_j I$$

where α_j is a scalar. Alternatively, if sparsity is too restrictive, B_{tj} may be a low rank matrix.

$B_{keyword}$ indicates the similarity between keywords. Like B_{tj}, it could be either a sparse matrix or a low rank matrix. We have not yet implemented this component.

6.1.2 Parameterizing the B matrix

The B matrix has $O(p^2)$ entries—far too many to learn from $O(n)$ training examples. (Recall p grows roughly linearly in n.) Moreover, different datasets possess different attributes, so each dataset requires a different B matrix. Fortunately, many entries of B are related. We define B in terms of a much smaller vector θ, and a 3-dimensional tensor E whose entries are indices of θ:

$$B_{ij} = \sum_{\ell=1}^{p} \mathbf{1}\{\ell \in E_{ij}\}\theta_\ell. \tag{6.2}$$

The values of θ are constrained by the requirement that B be non-negative. The vector θ has roughly 500 entries. Each of the first 5 entries of θ stores a weight for a class of name string defined by its rarity. For example, the entries of B on the diagonal corresponding to common names like "John Smith" are assigned similarity θ_0, whereas pairs of rare names like "Babbage Z. Gorzak" are assigned similarity θ_4. Off-diagonal entries of B corresponding to pairs of author names are the sum of two elements of θ: One element of θ corresponds to the rarity of the name string common to both names (e.g., "J. C. Smith" and "John Smith" shared the name string "J. Smith"). The other element of θ penalizes the discrepancy between the pair of name strings (e.g., due to a misspelling). The last few hundred entries of θ correspond to penalties for various types of frequently mistaken spelling substitutions (e.g., "yu"→"ju," "e"→"a").

6.2 Learning procedure

In the previous section, we set the similarity matrix $W = XBX^\intercal$, where the matrix B is parameterized by a fixed vector θ. In this section, we present a procedure for learning θ from a large collection of manually disambiguated publications, which will lead to good performance of our specific inference procedure on future datasets. Future datasets will not necessarily contain any of the same authors as the manually disambiguated publications. But the relative frequency of various types of spelling mistakes, the relative importance topics in relation to name strings, etc., should generalize from the manually disambiguated dataset to future datasets.

6.2.1 Designing the objective function

Learning θ to exactly optimize the quality of output from our specific inference procedure is likely intractable. Due to the heuristic nature of recursive bipartitioning, it is difficult to reason about the final output of the recursions.

Instead, we optimize an objective function that favors good parameters for minimizing average cut, the problem that spectral bipartitioning approximately solves. Let A_i denote the mentions of a particular author. Let A be the collection of all authors' mention-sets, that is $A = \{A_1, \ldots, A_{|A|}\}$, where $|A|$ is the number of authors in the training set. Let M be the collection of mentions in the training set; that is $M = \cup_{i=1}^{|A|} A_i$. Let M_i denote the i^{th} mention. For sets of mentions $U, V \subset M$, define

$$\text{avgCut(U, V)} = \frac{\sum_{i,j} W_{ij}}{|U| \times |V|}.$$

This is the average weight of the edges between mentions in U and mentions in V. Spectral bipartitioning approximately solves

$$\min_{V \subset M} \text{avgCut}(V, M \backslash V).$$

Therefore, intuitively, a good θ would have relatively low edge weight between pairs of non-coreferent mentions, and relatively high edge weight between pairs of coreferent mentions. The absolute edge weights are unimportant, since a constant scaling of all edge weights does not change bisection minimizing the average cut.

6.2.1.1 Average edge weight

We first consider minimizing the difference between the total weight of edges across authors, and the total weight of edges within authors:

$$\mathcal{L}_1(\theta) = \frac{1}{(|M|)^2} \sum_{i=1}^{|M|} \sum_{j=1}^{|M|} \varphi(M_i, M_j) W_{ij}, \tag{6.3}$$

where $\varphi(M_i, M_j)$ returns 1 if M_i and M_j are coreferent and −1 otherwise. Evaluating $\mathcal{L}_1$ naively would require computing a quadratic number of terms ($|M|$ is approximately 2,000,000) but an alternate formulation with a linear number of terms exists: the total edge weight within authors (the intra-author edge weight) can be computed in linear time because no author has more than a bounded number of publications (about 1000). Additionally, the total edge weight (both within authors and between authors) can be computed in linear time by exploiting the form of $W = XBX^\intercal$, where X and B are sparse, by both left and right multiplying W by a ones vector. Thus,

$$\mathcal{L}_1(\theta) = \frac{1}{(|M|)^2}\left[\mathbf{1}W\mathbf{1}^\intercal - \sum_{i=1}^{|A|}\sum_{j=1}^{|A_i|} W_{iA_{ij}}\right],$$

where (with slight abuse of notation) we let A_{ij} denote the index in M of the j^{th} mention of the i^{th} author. Therefore, $\mathcal{L}_1$ can be evaluated in linear time. Also, we can minimize (6.3) without ever evaluating all of its terms, using a stochastic gradient method and treating each edge as an example.

Without regularization, however, $\min_\theta \mathcal{L}_1$ is unbounded if any attribute appears more often between coreferent mentions than non-coreferent mentions; increasing the weight of such an attribute will always decrease $\mathcal{L}_1$. And, if any attribute occurs more often between non-coreferent mentions than coreferent mentions, setting this attribute's weight to 0 will minimize $\mathcal{L}_1$ (W must be non-negative for spectral clustering.) Adding an L_2 regularizer to $\mathcal{L}_1$ yields

$$\mathcal{L}_2(\theta) = \frac{1}{(|M|)^2}\left[\sum_{i=1}^{|M|}\sum_{j=1}^{|M|} W_{ij}\left(1 - 2\delta(M_i, M_j)\right) + \frac{\lambda}{2}\|\theta\|_2^2\right].$$

This remedies the first problem: the addition of a non-negative quadratic term ($\lambda > 0$) bounds $\min_\theta \mathcal{L}_2$ from below. But the second deficiency points to a deeper flaw with the design of this objective function, which cannot be addressed by regularization: Most attributes will more often be shared between non-coreferent pairs, because the vast majority of pairs of mentions are not coreferent. For example, the vast majority of mentions sharing a particular topic code are not coreferent, but, mentions with the same topic are nonetheless more likely to be coreferent than those with different topics. Hence, topic is still useful for guiding spectral clustering. It should receive weight strictly greater than 0. Objectives $\mathcal{L}_1$ and $\mathcal{L}_2$ will not do so. Rather than picking θ to minimize average weights, perhaps we could find θ that accentuates the (relative) similarity between pairs of mentions most likely to be confused.

6.2.1.2 Minimax edge weight

Let α and γ be non-negative tuning parameters. Recall that t is the threshold used by the inference procedure. Cuts with average weight above t will not be accepted. Because a constant scaling of all edge weights does not affect the solution of spectral clustering, the choice of t is arbitrary. We set $t = 1$ for both learning and inference. We propose minimizing

$$\mathcal{L}_3(\theta) = \frac{1}{|A|}\sum_{i=1}^{|A|}\left[\ell(\xi_i) + \alpha\ell(\zeta_i) + \frac{\lambda}{2}\|\theta\|_2^2\right], \tag{6.4}$$

where

$$\ell(x) = \frac{1}{2}\left[(x+\gamma)_+\right]^2, \tag{6.5}$$

and

$$\xi_i = t - \min_{V \subset A_i} \operatorname{avgCut}_\theta(V, A_i \setminus V),$$

and

$$\zeta_i = \max_{U \subset M \setminus A_i} \operatorname{avgCut}_\theta(A_i, U) - t. \tag{6.6}$$

For author i, ξ_i measures the amount by which threshold t exceeds every bipartition of A_i, and ζ_i measures the amount by which some cut between the mentions of other authors and A_i exceeds threshold t. If ξ_i and ζ_i are both negative, our inference routine should disambiguate author A_i perfectly.

Like $\mathcal{L}_1$ and $\mathcal{L}_2$, objective $\mathcal{L}_3$ penalizes a θ that heavily weights edges between non-coreferent mentions, relative to the edge weights between coreferent mentions. While $\mathcal{L}_1$ and $\mathcal{L}_2$ penalize θ based on the average of edge weights within and between authors, $\mathcal{L}_3$ compares the minimum average cut within each author to the maximum average cut between each author and any set of mentions by other authors. In this sense, it more closely mirrors the "average cut" perspective on spectral bipartitioning. (Spectral bipartitioning solves a relaxation of minimum average cut.)

6.2.1.3 Alternative objective functions

We consider two alternatives to $\mathcal{L}_3$. Let

$$\mathcal{L}_4(\theta) = \max_{i=1,\dots,|A|}\left[\ell(\xi_i) + \alpha\ell(\zeta_i)\right] + \frac{\lambda}{2}\|\theta\|_2^2.$$

Like $\mathcal{L}_3$, objective $\mathcal{L}_4$ focuses on learning good θ for the most problematic pairs of mentions, rather than the overall average of intra- and inter- author edge weights. However, $\mathcal{L}_4$ fixates on a particular author, whose intra-author edge weights are most exceeded by inter-author edge weights. If perfectly disambiguating all authors were possible, $\mathcal{L}_4$ might be a reasonable objective function, but in practice disambiguating some authors is hopeless. With $\mathcal{L}_4$, learning focuses entirely on authors that we cannot get right, who likely are not representative of the vast majority of the authors. With $\mathcal{L}_3$ the most difficult authors have much less influence.

We also consider

$$\mathcal{L}_5(\theta) = \max_{i=1,\ldots,|A|} \left[e(\xi_i) + \alpha e(\zeta_i)\right] + \frac{\lambda}{2}\|\theta\|_2^2,$$

where

$$e(x) = (x+\gamma)_+ .$$

Like ℓ (truncated quadratic loss), e (hinge loss) penalizes values of x more that exceed $-\gamma$ by more. But e penalizes values in proportion to the amount they exceed $-\gamma$, while ℓ penalizes them quadratically. Thus, while $\mathcal{L}_4$ focuses more than $\mathcal{L}_3$ on the most difficult authors, $\mathcal{L}_5$ focuses less on them than $\mathcal{L}_3$. $\mathcal{L}_5$ should lead to disambiguated results more in line with subjective disambiguation quality. However, minimizing $\mathcal{L}_3$ is faster than minimizing $\mathcal{L}_5$, due to its greater convexity. Getting consistent convergence with $\mathcal{L}_5$ necessitates setting λ to higher values than desired, whereas with $\mathcal{L}_3$ we are free to pick any non-negative value for λ. Implementation concerns ultimately dictate our preference for $\mathcal{L}_3$ over $\mathcal{L}_5$.

6.2.2 Minimization

Objective $\mathcal{L}_3$ is convex because it is a non-negative weighted sum of convex functions. To show that $\ell(\xi_i)$ and $\ell(\zeta_i)$ are convex in θ, because ℓ is convex and non-decreasing, it suffices to show that ξ_i and ζ_i are convex in θ. Observe that

$$\begin{aligned}
\mathrm{avgCut}_\theta(V, A_i \setminus V) &= \left[|V|\left(|A|_i - |V|\right)\right]^{-1} \sum_{i\in V}\sum_{j\in A_i\setminus V} W_{ij} \\
&= \left[|V|\left(|A|_i - |V|\right)\right]^{-1} \sum_{i\in V}\sum_{j\in A_i\setminus V} X_{i\cdot} B \left(X_{j\cdot}\right)^\top \\
&= \left[|V|\left(|A|_i - |V|\right)\right]^{-1} \sum_{i\in V}\sum_{j\in A_i\setminus V}\sum_{k\in X_i}\sum_{\ell\in X_j} B_{k\ell} \qquad (6.7)\\
&= \left[|V|\left(|A|_i - |V|\right)\right]^{-1} \sum_{i\in V}\sum_{j\in A_i\setminus V}\sum_{k\in X_i}\sum_{\ell\in X_j}\sum_{q\in E_{k\ell}} \theta_q .
\end{aligned}$$

Hence avgCut_θ and $-\mathrm{avgCut}_\theta$ are linear functions of θ. Now

$$\zeta_i = \max_{U\subset M\setminus A_i} \mathrm{avgCut}_\theta(A_i, U) - t \qquad (6.8)$$

and

$$\begin{aligned}
\xi_i &= t - \min_{V\subset A_i} \mathrm{avgCut}_\theta(V, A_i \setminus V) \qquad (6.9)\\
&= t + \max_{V\subset A_i} -\mathrm{avgCut}_\theta(V, A_i \setminus V)
\end{aligned}$$

are maxima of linear (and thus convex) functions of θ, and hence are also convex.

6.2.2.1 Stochastic subgradient descent

The maximum in (6.8) and the minimum in (6.9) over cuts are continuous but not differentiable. Each author can serve as an example. Hence, we use stochastic subgradient descent to minimize objective $\mathcal{L}_3$. Let ∂_k denote the subdifferential with respect to θ_k. Then

$$\partial_k \mathcal{L}(\theta) = \frac{1}{|A|} \sum_{i=1}^{|A|} \left[\partial_k \ell(\xi_i) + \alpha \partial_k \ell(\zeta_i) + \lambda \theta_k \right]. \tag{6.10}$$

For $x \in \{\xi_1, \ldots, \xi_{|A|}, \zeta_1, \ldots, \zeta_{|A|}\}$,

$$\partial_k \ell(x) = \begin{cases} \{0\}, & \text{if } x \leq -\delta \\ (x + \delta)\,(\partial_k x), & \text{otherwise.} \end{cases} \tag{6.11}$$

Also

$$\partial_k \xi_i = -\partial_k \min_{V \subset A_i} \operatorname{avgCut}(V, A_i \setminus V) \tag{6.12}$$

and

$$\partial_k \zeta_i = \partial_k \max_{U \subset M \setminus A_i} \operatorname{avgCut}(A_i, U). \tag{6.13}$$

To find a particular subgradient in $\partial_k \mathcal{L}$, we approximate a subgradient from (6.12) and compute a subgradient from (6.13).

6.2.2.2 Approximating the intra-author subgradient

Though few authors write more than several hundred publications, finding the exact minimum in (6.12) seems intractable for more than a few tens of publications: the number of possible partitions grows exponentially. Fortunately, a minimizer can be well approximated for a particular θ, using spectral bipartitioning, followed by a greedy "clean up" procedure that considers swapping individual mentions between parts.

To perform spectral bipartitioning, we find the eigenvector associated with the second smallest eigenvalue[1] (the "Fielder vector") of the Laplacian matrix

$$L_i = \operatorname{diag}(W_i \mathbf{1}) - W_i,$$

where W_i is the submatrix of W corresponding to the affinities between the elements of A_i. Though we only need to find one eigenvector, we find that in practice computing the full SVD of L_i is faster than using iterative methods. Though finding the SVD takes $O(n^3)$ time,

[1]For authors with disconnected similarity graphs, bisecting based on an eigenvector is neither necessary nor advisable. Any bisection that does not cut any connected component is a minimizer.

here n is $|A_i|$, not $|M|$. Since the number of publications per author is bounded, the runtime of this step is linear in the size of the training set.

The Fiedler vector gives an ordering of an author's mentions. To decide where to cut the sequence of ordered mentions, we consider each of the $n-1$ cuts. By exploiting the structure of $W = XBX^\intercal$, we compute all $n-1$ cut weights efficiently. Then, we iterate repeated through the mentions, and propose moving each mention to the other part of the bipartition, accepting moves that reduce the average cut. Again, by exploiting the structure of W, we can efficiently evaluate the change in average cut weight for any particular move.

Upon determining the bipartition of A_i that minimizes the average cut weight, denoted $(V^\star, A_i \backslash V^\star)$, we compute a subgradient $g_\xi(\theta) \in \partial\ell(\xi_i)|_\theta$. In two special (but not uncommon) cases, we can avoid much of the computation needed to compute $g_\xi(\theta)$. First, if author i has only written 1 publication, then trivially $0 \in \partial\xi_i|_\theta$. Computing the minimum average cut can be avoided. In most collections, roughly half the authors have written only a single publication (author productivity follows a Zipf distribution), so this rule leads to a significant speedup. Second, if the minimum average cut weight of A_i is greater than $t+\gamma$, then $0 \in \partial\xi_i|_\theta$, and computing $g_\xi(\theta)$ is unnecessary. This condition usually holds after a small number of iterations of the stochastic subgradient method, and hence this rule too yields a sizable speed up.

If neither of these special cases apply, $\ell(\xi_i) > 0$ and $\ell(\xi_i)$ is differentiable for the current value of θ, and

$$\begin{aligned}
g_\xi(\theta) &= \frac{\partial}{\partial\theta}\ell(\xi_i) \\
&= \frac{\partial}{\partial\theta}\frac{1}{2}\left[(\xi_i+\gamma)\right]^2 \\
&= (\xi_i+\gamma)\frac{\partial}{\partial\theta}\xi_i \\
&= \left([t-\text{avgCut}_\theta(V^\star, A_i \setminus V^\star)]+\gamma\right)\frac{\partial}{\partial\theta}\left[t-\text{avgCut}_\theta(V^\star, A_i \setminus V^\star)\right] \\
&= -\left[t+\gamma-\text{avgCut}_\theta(V^\star, A_i \setminus V^\star)\right]\frac{\partial}{\partial\theta}\text{avgCut}_\theta(V^\star, A_i \setminus V^\star).
\end{aligned}$$

For any $U, V \subset M$,

$$\begin{aligned}
\frac{\partial}{\partial\theta}\text{avgCut}(U,V) &= \left[|U|\,(|V|)\right]^{-1}\sum_{i\in U}\sum_{j\in V}\sum_{k\in X_i}\sum_{\ell\in X_j}\sum_{q\in E_{k\ell}}\frac{\partial}{\partial\theta}\theta_q && (6.14)\\
&= \left[|U|\,(|V|)\right]^{-1}\sum_{i\in U}\sum_{j\in V}\sum_{k\in X_i}\sum_{\ell\in X_j}E_{k\ell}. && (6.15)
\end{aligned}$$

In (6.14) we use $E_{k\ell}$ to denote the indexes of the entries of θ summed to compute $B_{k\ell}$. In (6.15) we reused the notation $E_{k\ell}$ to denote a binary vector with the same length as θ, indicating the entries of θ summed to compute $B_{k\ell}$. In practice we compute

$$\sum_{i\in V^\star}\sum_{j\in A_i\backslash V^\star}\sum_{k\in X_i}\sum_{\ell\in X_j}E_{k\ell}$$

without explicitly forming any $E_{k\ell}$, by counting the number of times each entry of θ contributes to the weight of edges that cross the cut $(V^\star, A_i \backslash V^\star)$.

6.2.2.3 Computing the inter-author subgradient

For a particular author A_i, once a maximizer $U^\star$ of (6.13) is known, finding a subgradient $g_\zeta(\theta) \in \partial\ell(\zeta_i)|_\theta$ of the inter-author loss is straightforward. If $\text{avgCut}_\theta(A_i, U^\star) \le t - \gamma$, then inter-author loss is the constant 0 for all θ, so 0 is a subgradient. Otherwise,

$$\begin{aligned} g_\zeta(\theta) &= \frac{\partial}{\partial\theta}\left[\frac{1}{2}\left[(\text{avgCut}_\theta(A_i, U^\star) - t + \gamma)\right]^2\right] \\ &= (\text{avgCut}_\theta(A_i, U^\star) - t + \gamma)\,\frac{\partial}{\partial\theta}\text{avgCut}_\theta(A_i, U^\star), \end{aligned}$$

where the formula for $\frac{\partial}{\partial\theta}\text{avgCut}_\theta(A_i, U^\star)$ is given in (6.15).

Algorithm (1) shows how to find

$$U^\star = \arg\max_{U \subset M \smallsetminus A_i} \text{avgCut}(A_i, U).$$

The following are the key ideas underlying it. For any maximizer U of (6.13), there exists a singleton $\{m_u\} \subset U$ that is also a maximizer. We aim to find an $m_u \in M \backslash A_i$ without considering most of the edges that have end points in A_i (and without considering any of the edges lacking end points in A_i). In the worst case, we would have to consider all edges with exactly one end point in A_i, but on average, we can consider many fewer edges, by exploiting the structure of $W = XBX^\top$ (where X and B are sparse).

First we partition the mentions into connected components in linear time. This partitioning only needs to be done once, regardless of how many subgradients we ultimately compute. We consider only author's name string when forming connected components, not topic. If A_i belongs only to connected components exclusively composed of mentions in A_i, then their inter-author loss $\ell(\zeta_i)$ is trivially 0, for all θ. Hence 0 is a subgradient in this common special case. If A_i belongs to a connected component containing multiple authors, we count how often each attribute appears in the mentions in A_i. Most attributes do not appear at all, and these 0 counts are not explicitly stored. Then we compute the weight with which these attributes target other attributes. Again most attributes are not targeted by any attribute of any mention in A_i.

Upon building these data structures, we consider targeted attributes, in descending order according to the ratio of their weight to their number of attributes in the connected component sharing this attribute. Thus, we first consider the attributes that are most important in relation to the amount of work required to process them. Initially, as we iterate through the attributes, we maintain a sum of the attribute weights for each mention in $M \backslash A_i$ that shares attributes encountered. The total weight on the remaining attributes upper bounds how similar any mention could be to A_i. If the upper bound becomes less than $t - \gamma$, then we know the inter-author loss $\ell(\zeta_i)$ is 0, and $0 \in \partial\ell(\zeta_i)|_\theta$; considering additional attributes is unnecessary.

Even if the inter-author loss is not 0, we still avoid considering many mentions with non-zero similarity to mentions in A_i. After related mentions have been identified by considering the most promising attributes, considering additional mentions may be unnecessary; we only aim to find the mention m_u *most* similar to A_i. Therefore, if the combined weight of all remaining attributes is less than the weight already tabulated for some mention, the most similar mention must have already been encountered. Alternatively, if the weight remaining is less than $t - \gamma$, then no unseen mention could accrue enough weight to have non-zero loss. In either case, as we iterate through the remaining attributes, we only need to tally the weights for mentions that have already been visited.

6.2.2.4 Adaptive subgradient method for stochastic optimization (AdaGrad)

Stochastic subgradient methods' updates typically have the following form:

$$\theta^{(t+1)} = \theta^{(t)} + \alpha_t G_t g(\theta^{(t)}),$$

where $g(\theta)$ is a noisy subgradient at θ, $G_t = I$, and $(\alpha_t)_1^\infty$ is a divergent series but a square summable sequence. These updates use the same step size α_t for all coordinates of θ. Some coordinates of θ, such as those corresponding to topic codes, are updated almost every iteration. Other coordinates of θ, such as those corresponding to rare spelling mistakes, are updated infrequently. Yet the same step size schedule $(\alpha_t)_1^\infty$ will apply to all coordinates.

A subgradient method called AdaGrad [102] instead sets G_t to a diagonal matrix that approximates the inverse Hessian of the objective function. Specifically,

$$G_t = \left[\operatorname{diag}\left(\sum_{i=1}^{t} g(\theta^{(t)})\right)\right]^{-1/2},$$

and, for some constant c,

$$\alpha_t = \frac{c}{\sqrt{t}}.$$

This effectively implements a step size schedule $(\alpha_t G_t)_1^\infty$ that, all else equal, cools more quickly for the coordinates of θ that are updated often, and more slowly for the coordinates of θ that are updated rarely. We find that AdaGrad converges more quickly than a vanilla stochastic subgradient method.

6.3 Results

We apply the learning procedure and the inference procedure to both synthetic data and `MathSciNet`, a collection of roughly 2 million manually disambiguated publications.

Our synthetic data is a collection of items belonging to roughly 500 true clusters. Each item is a vector of 10 integers, the first 9 entries formed by corrupting the corresponding cluster's true identifier with progressively more noise. The final entry is intentionally misleading—if two entries share this integer, they likely do not belong to the same cluster. This synthetic dataset tests both the learning and inference procedures, without testing the string processing logic required for author disambiguation. Also, by using synthetic data, we can draw multiple datasets from exactly the same distribution.

6.3.1 Learning parameters from synthetic data

On synthetic data, the learning procedure converges within a few minutes. At the optimum,

$$\theta = \begin{bmatrix}0.168 & 0.158 & 0.107 & 0.079 & 0.065 & 0.048 & 0.048 & 0.039 & 0.036 & 0.001\end{bmatrix}^\top.$$

An application of the learning procedure to the same data, but with a different random selection of examples, yields

$$\theta = \begin{bmatrix}0.168 & 0.158 & 0.106 & 0.079 & 0.068 & 0.050 & 0.050 & 0.038 & 0.038 & 0.001\end{bmatrix}^\top.$$

Additional trials confirm that the learning procedure indeed converges to the same value of θ. Moreover, the relative values of the optimizer θ_i are in accordance with the dataset's construction:

$$\theta_0 \geq \theta_1 \geq \ldots \geq \theta_8 \geq \theta_9.$$

The weight on each successive parameter decreases as the amount of noise increases. Also, the final parameter recovered, which weights an intentionally misleading feature function, is as small as the constraints allow. (We use projection to enforce $\theta_i \geq 0.001$ for all i.)

6.3.2 Inferring an optimal partition of synthetic data

Given the optimal θ, we turn to testing our inference procedure. Since most author disambiguation to date has been performed by agglomerative clustering, we use agglomerative clustering as a baseline. We use the similarity matrix $W = XBX^\top$, where each entry of B is a sum of elements of θ (as discussed earlier), for both agglomerative clustering and for our disambiguation procedure ("spectral," in Table 6.2). Then, we generate six datasets from the same distribution, learn the optimal θ from the first dataset, and disambiguate the remaining datasets using that θ.

Our disambiguation procedure easily scales to processing millions of items on a single machine, whereas agglomerative clustering—which has $O(n^2 \log n)$ runtime on this type of data—nears its limit on a synthetic dataset of modest size (approximately 3000 items). If it can disambiguate real author data well enough, then performance on synthetic data is relevant only for validating the proper functioning of the procedures.

Figure 6.4 shows that inference for spectral disambiguation scales as expected: nearly linearly in n. Agglomerative clustering, on the other hand, exhibits quadratic runtime.

We compare the disambiguation algorithms' performance using two metrics: accuracy and F-score. Accuracy is the proportion of authors perfectly disambiguated—to get any credit,

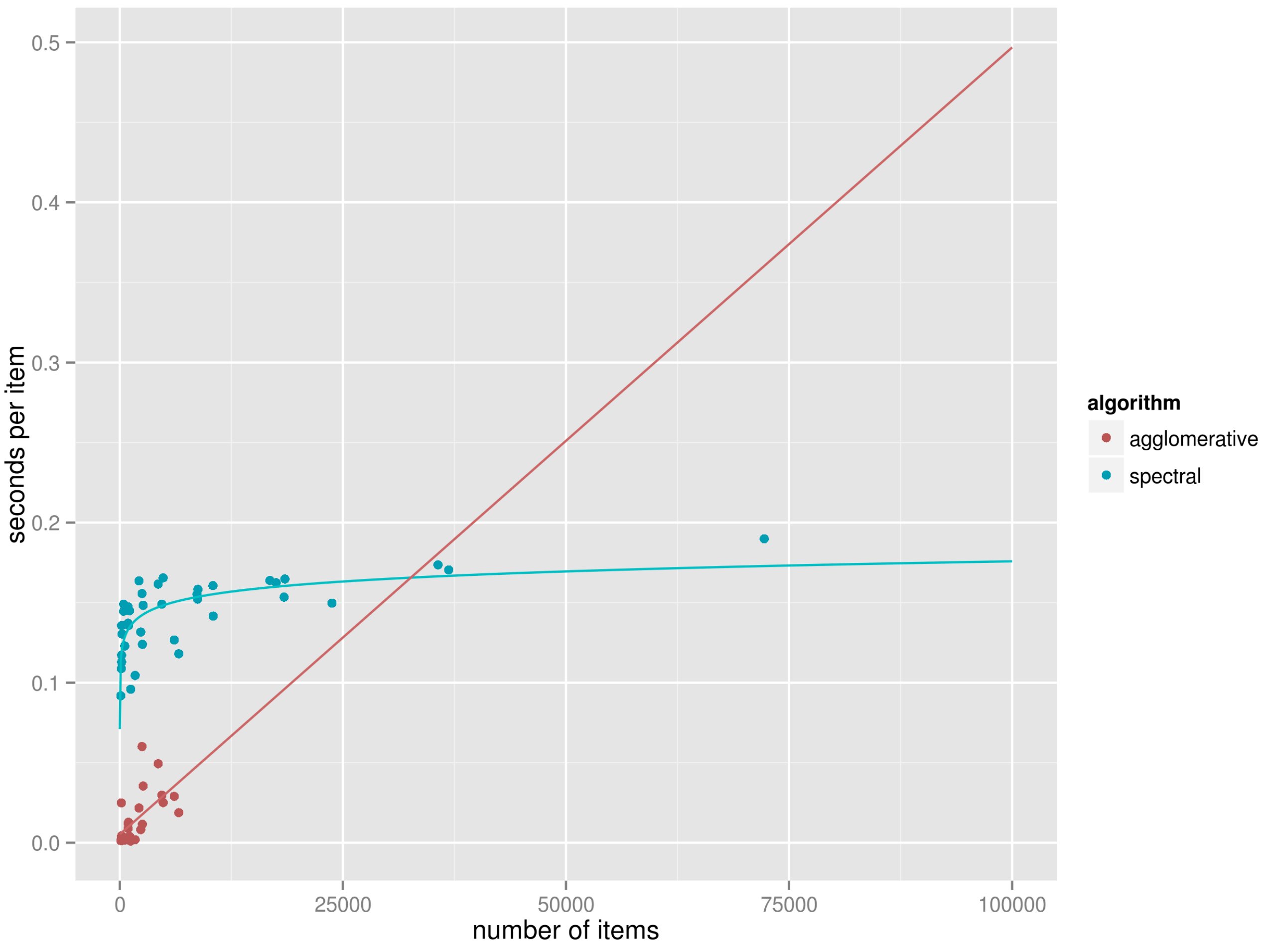


Figure 6.4: Scaling results for inference. Spectral disambiguation exhibits nearly constant runtime *per item*, whereas the runtime for agglomerative clustering *per item* grows linearly.

no mention can be excluded and no other author's mentions may be included. Accuracy is strict, but also highly interpretable. F-score is the harmonic mean of pairwise recall and pairwise precision. For both metrics, larger is better.

As Table 6.2 shows, agglomerative clustering consistently outperforms our spectral disambiguation procedure. We see two possible explanations: 1) the value of θ that minimizes objective function $\mathcal{L}_3$ is more suitable for agglomerative clustering than for our spectral disambiguation algorithm; or 2) our disambiguation procedure does not work as well as agglomerative clustering on these data for any value of θ.

trial #	agglomerative F-score	agglomerative accuracy	spectral F-score	spectral accuracy
1	97.6%	90.7%	95.6%	83.6%
2	98.8%	93.3%	94.8%	81.1%
3	98.1%	93.0%	95.7%	83.6%
4	98.1%	92.3%	95.3%	82.5%
5	97.8%	91.6%	94.7%	80.4%

Table 6.2: Inference performance on synthetic data

6.3.3 Learning parameters from `MathSciNet`

We test many different parameterizations of the B matrix for `MathSciNet`. Without spelling correction, for example, we find

$$\theta = \begin{bmatrix}\theta_{frequency} & \theta_{topic} & \theta_{mutation} & \theta_{truncation}\end{bmatrix}^{\intercal},$$

where

$$\begin{aligned}
\theta_{frequency} &= \begin{bmatrix}0.4066 & 0.4731 & 0.5294 & 0.5917 & 0.6148\end{bmatrix}\\
\theta_{topic} &= \begin{bmatrix}0.0565 & 0.0103 & 0.046\end{bmatrix}\\
\theta_{mutation} &= \begin{bmatrix}-0.0619 & -0.1999 & -0.4728 & -0.4021 & -0.0277 & -0.1105 & 0. & -0.0001\end{bmatrix}\\
\theta_{truncation} &= \begin{bmatrix}-0.1663 & -0.1851 & -0.1851 & -0.2046 & -0.186 & -0.15 & -0.0865 & -0.0662\end{bmatrix}.
\end{aligned}$$

Here $\theta_{frequency}$ stores the weights of increasingly rare name strings. The first entry corresponds to very common names, like “J. Smith,” whereas the last entry corresponds to very rare names, which are almost certain to be coreferent. As desired,

$$\theta_0 < \theta_1 < \theta_2 < \theta_3 < \theta_4.$$

The rarer the name, the more likely mentions sharing it are coreferent. θ_{topic} stores the weights for topics codes of 3 levels of specificity. The top-level topic code (e.g., “MSC62: Statistics”) gets the most weight, as desired: most authors publish in only a small number of top-level topic codes, whereas the mid-level (e.g., “Nonparametric inference”) and bottom-level (e.g., “Hypothesis testing”) are far less indicative of coreference. $\theta_{mutation}$ stores penalties for different kinds of mismatches (other than misspelling) between name strings. The last entry of $\theta_{mutation}$ corresponds to the penalty for adding a hyphen between the middle and last names, if the mention otherwise would have multiple middle names (e.g., “Juan Carlos Rodrigez Diaz” →“Juan Carlos Rodrigez-Diaz”). This mutation rule is unlikely to generate false positives, so the penalty is insignificant. The third entry of $\theta_{mutation}$, on the other hand, is a large penalty. It corresponds to dropping the first name, and using the middle name as a first name (e.g., “D. Radko Mesiar” → “Radko Mesiar”). Though sometimes authors do intermittently identify by their middle names, intuitively, this rule is likely to lead to false positives, so the high penalty makes sense. Finally, $\theta_{truncation}$ stores penalties for truncating a first name to various lengths (e.g., "Yurii Nesterov" → "Yu Nesterov"). Truncations are surprisingly common in `MathSciNet`.

In another test, as a sanity check, we include the ground truth as an attribute of each mention. In this case, the learning procedure correctly assigns high weight to the ground truth (0.6469) and low weight to all other features ($\theta_i < 0.02$).

6.3.4 Inferring an optimal partition of `MathSciNet`

Table 6.3 gives results from disambiguating `MathSciNet`.[2] Each row reports results from including a different subset of the attributes. Using name frequency attributes exclusively—without mutations, truncations, spelling correction, and while ignoring topic codes—yields 71.8% accuracy. This is the baseline score to beat, to show that our combined procedure (learning + inference) can make effective use of additional attributes. Including mutations, truncations, spelling correction, but still excluding topic code increases accuracy to 73.9%. This increase, though not insignificant, is less than expected based on manually examining the errors made by the baseline procedure. Including topic code attributes (the third row of table 6.3) made the performance much worse—worse than even the baseline. Additionally, while including the ground truth increases performance, over 11% of authors are still not perfectly recovered. Removing the topic code, but leaving all the other attributes—including the ground truth—increases the accuracy to 98.1%. Why would including topic codes be particularly deleterious?

included attributes						spectral	spectral
frequency	topic	mutation	truncation	spelling	truth	F-score	accuracy
√						91.6%	71.8%
√		√	√	√		90.9%	73.9%
√	√	√	√	√		87.4%	63.8%
√	√	√	√	√	√	95.4%	88.8%
√		√	√	√	√	99.6%	98.1%

Table 6.3: Performance on `MathSciNet`

Most attributes are shared only by a modest number of mentions. Topics codes are an exception. Hundreds of thousands of mentions share some top-level topic codes. The learning procedure learns weights that are good for the final split of the disambiguation by recursive bisection. Indeed, the learned weights seem to work well in practice for final splits, but the weights that are good for the final split are not necessarily good for earlier

[2]We use the same dataset for both training and testing at this time, to postpone dealing with differences in distribution between the training and testing datasets. With so much data and so few parameters, we do not expect that overfitting will prevent validating the procedure.

splits—particularly when some attributes are shared by many mentions. In a dataset of only two authors, when ground truth is an attribute with weight 0.6469, then topic weights, which are less than 0.02, do not substantively affect the split. But, in a dataset with many thousands of mentions with the same topic code, the topic weight contributes to billions of edges connecting thousands of mentions (the number of edges grows quadratically), whereas the number edges affected by the ground truth scales linearly as new authors are added to the dataset. Thus, for early splits, topic similarity overshadows the influence of the ground truth: the inference algorithm prefers to split authors (each of whom comprises relatively few mentions) than to split topics (which are shared by many mentions).

Recursive spectral bipartitioning is a well-established clustering algorithm, but as far as we know, this is the first work that attempts to learn the similarity matrix for it from labeled data. Our experience suggests a negative result: no single set of pairwise similarities leads to good splits at every recursion. Unfortunately, we see no straightforward way to extend our approach to learn a variable similarity matrix. Generative modeling may be more appropriate for this application, with an Indian Buffet Process as a prior on the binary author-article assignment matrix.

Algorithm 1 Find $m \in M \backslash A_i$ maximizing avgCut(m, A_i)

```
def max_external_cut(self):
    assert(self.num_true > 1)
    neighbors = defaultdict(float)

    w_at2 = []
    total_w = 0.
    for at_id, count in self.at_counts_global.iteritems():
        for at2_id, poses_lst in self.sg_component.at_to_at2(at_id):
            prop = count / float(self.sg_local.n())
            w = self.model.get_affinity(poses_lst) * prop * 2.
            w_at2.append((w, at2_id))
            total_w += w
    visited_w = 0.
    max_it, max_ws = None, 0.

    def goodness(w_at2):
        return w_at2[0] / len(self.sg_component.at_to_it(w_at2[1]))

    for w, at2_id in sorted(w_at2, reverse=True, key=goodness):
        left_w = total_w - visited_w
        # for speed, if there's no chance of incurring inter author
        # error, just return 0 immediately.
        if max_ws + left_w <= (learn_t - delta):
            # print "short circuit"
            return None, 0.
        visited_w += w
        if left_w > max_ws and left_w > (learn_t - delta) \
            and len(self.sg_component.at_to_it(at2_id)) <= max_external_nodes:
            # "breadth loop"
            for it2_id in self.sg_component.at_to_it(at2_id):
                if self.sg_component.items[it2_id].true_id != self.true_id:
                    neighbors[it2_id] += w
                    if neighbors[it2_id] > max_ws:
                        max_it, max_ws = it2_id, neighbors[it2_id]
        else:
            # "depth loop"
            for it2_id in neighbors.keys():
                if at2_id in self.sg_component.it_to_at(it2_id):
                    neighbors[it2_id] += w
                    if neighbors[it2_id] > max_ws:
                        max_it, max_ws = it2_id, neighbors[it2_id]
    return max_it, max_ws
```

Part III

Computer experiments

Chapter 7

Confident contouring

This chapter addresses the problem of determining a level set of a real-valued function f, based on a limited number of evaluations of f. This problem is common when f is computationally expensive computer simulator, modeling a physical process, which maps settings for unknown constants to a real-valued outcome.

We are particularly concerned with contouring a climate simulator, based on a dataset from the Lawrence Livermore National Laboratory (LLNL) containing results from their climate simulations. Each simulation has 21 parameters as its inputs. The output of interest is the simulated global average upwelling longwave flux (FLUT) approximately 50 years in the future. Henceforth, we refer to the mapping from the parameter space to FLUT induced by the climate simulation as the *simulation function*. Evaluating the simulation function was computationally expensive, with each run taking several days on a supercomputer. The dataset contains results from approximately 1000 runs, and any additional datasets would likely be of a similar scale. The 21 parameters are scaled between 0 and 1. This range contains all parameter settings considered reasonable. We aim to determine which parameter settings map to FLUT values in excess of some threshold t.

We focus on two settings. In the first, the dataset is fixed. We aim to impute a level set based on a finite collection of observations. Henceforth, we refer to the task of determining points' memberships with regard to the level set as *classification*. In the second setting, given an existing set of observations of f, we get to adaptively select parameter settings to test, so as to maximize knowledge of the simulation function while minimizing the number of evaluations of the simulation function.

7.1 Inference from a fixed set of observations

Clearly no finite collection contains every possible combination of the 21 real-valued parameters. In fact, no dataset of the scale anticipated contains all combinations of a high setting and a low setting for each parameter. (2^{21} runs would be required to simulate the "corners" of the parameter space.)

While it may be tempting to approximate the simulation function with a simplier parametric model, such an approximation is unlikely to be accurate: the simulation function is highly nonlinear, with many interactions between parameters. One may also be tempted to suppose some prior distribution over the parameter space, but the model's true parameters really might lie anywhere: no decisive prior information is available.

Continuity conditions can reasonably be expected to hold: minor perturbations in the inputs are unlikely to trigger major changes in the output. Though other types of continuity conditions might also suffice, for simplicity, we suppose Lipschitz continuity. More precisely, let $f : [0,1]^{21} \to \mathbb{R}$ be the simulation function. Let d_X be some distance metric on the parameter space. We supposed that for all $x_1, x_2 \in [0,1]^{21}$, there exists a scalar K_f satisfying

$$\frac{|f(x_2) - f(x_1)|}{d_X(x_1, x_2)} \leq K_f.$$

Without evidence that some other distance metric would be more appropriate, we set d_X to the distance metric induced by the sup-norm. Let S be the level set desired. Let t be the threshold for inclusion in S. Let g satisfy

$$g(x) = f(x) - t.$$

Then g has the same level set with respect to 0 as f has with respect to t. Thus, without loss of generality, we may let $t = 0$. Then

$$S = \{x : f(x) \geq 0\}.$$

Let h satisfy

$$h(x) = f(x)/K_f.$$

Then h has the same level set with respect to 0 as f has with respect to 0, and $K_h = 1$. Thus, without loss of generality, we may let $K_f = 1$. Now, for $x \in [0,1]^{21}$, if $f(x) \geq 0$, then

$$\bar{B}(x, f(x)) \cap [0,1]^{21} \subset S,$$

where $\bar{B}$ is the closed ball centered at x of radius $f(x)$. Similarly, if $f(x) < 0$, then

$$B(x, |f(x)|) \cap [0,1]^{21} \subset S^c,$$

where B is the open ball centered at x of radius $|f(x)|$. Thus, setting d_X to sup-norm allows classifying cubical subsets of the domain with regard to their membership in S. This is convenient because a cube can be naturally represented in a computer program by the coordinates of its lowest corner and its highest corner, and this storage format scales linearly in the dimensionality of the domain of f.

We also experimented with setting d_X to the distance metric induced by the L_1-norm, or, equivalently, the Manhattan distance.

Under any distance metric, classification is most difficult where f is near 0. To formalize this intuition, let μ be Lebesgue measure. In the worst case, where there exists $A \subset [0,1]^{21}$ satisfying $\mu(A) > 0$ and $f|_A = 0$, no finite or countable number of evaluations of f would let us classify a non-measure-zero subset of A; if $x \in A$, then we classify $B(x, f(x))$, and

$$\mu(B(x, f(x))) = \mu(B(x, 0)) = 0.$$

To avoid this situation, redefine the classification problem slightly: Fix $\epsilon > 0$. Rather than classifying points as either in or out of S, classify each point into the following categories:

1. $\{x : f(x) > 0\}$
2. $\{x : |f(x)| < \epsilon\}$
3. $\{x : f(x) < 0\}$

These categories are not mutually exclusive. The classification problem is solved when every point in the domain is assigned to at least one category.

7.2 Inference from an adaptively selected dataset

At present, LLNL tests combinations of parameter settings in accordance with Latin hypercube designs, one-at-a-time designs, and several variants thereof. In other words, the experimental design is fixed before any simulations take place. Yet batches of runs are performed serially, so there is the potential to use the results from earlier runs to select the inputs for subsequent simulations.

The optimal sequence of evaluations would allow us to classify the entire domain with the fewest evaluations of the simulation function. However, a Bayes optimal approach would require knowing unknowable distributions. Instead, we propose a greedy strategy. At each iteration, we select the point in the parameter space to maximize the anticipated volume of the domain classified during that iteration, according to a surrogate model, trained on the observations thus far. Intuitively, a good selection has the following qualities:

1. It lies away from the borders and from regions of the domain that has already been classified, so as to maximize the volume of neighboring domain that can be classified.
2. The simulation function maps it to a value far from the level set's threshold, so that a large neighborhood around it may be determined to have the same classification.

Optimizing for the second quality requires modeling the simulation function. At any point in the parameter space, we want estimates not just of the expected value of the simulation function, but estimates of the distribution of the simulation function. A Gaussian process model gives us these estimates. Gaussian process models are frequently used to model nearly-deterministic computer simulations [103], in part because their fitted values exactly match the true responses at observed points. Henceforth, let $\hat{f}(x)$ be the estimate of $f(x)$ from

Algorithm 2 `gasp`

```
1: for $i = 1, \dots, n_1$ do
2:     Draw point $x_i$ at random uniformly from $[0,1]^N$.
3:     Draw scalars $y_i^{(1)}, \dots, y_i^{(n_2)}$ from $\mathcal{N}(\hat{f}(x_i), \hat{\sigma}(x_i))$.
4:     For $j = 1, \dots, n_2$, draw points $z_{ij}^{(1)}, \dots, z_{ij}^{(n_3)}$ at random uniformly from
           $$B(x_i, y_i^{(j)}) \cap [0,1]^N.$$
5:     Let $R \subset [0,1]^N$ be the region already classified and let $\mathbf{1}$ be the indicator function.
       Set
           $$\pi_i = \frac{\sum_j^{n_2} \sum_k^{n_3} \mathbf{1}(z_{ij}^{(k)} \in R)}{n_2 n_3},$$
6: Find $i^\star = \arg\max_{i'}\{\pi_{i'}\}$.
7: Evaluate $f(x_{i^\star})$.
```

the Gaussian process model, and let $\hat{\sigma}(x)$ be the standard error of this estimate. Then $\mathcal{N}(\hat{f}(x), \hat{\sigma}(x)^2)$ is the distribution of the Gaussian process model at x.

However, even with the ability to estimate the simulation function's value at any point, finding the optimal point in a high-dimensional space for a single iteration of a greedy algorithm is intractable. The modeling assumptions so far enable comparing points, determining which was better, but not searching an uncountable set for an optimal point. So we again use Monte Carlo methods. Let N be the dimensionality of the domain. Fix whole numbers n_1, n_2, n_3. Then our selection procedure is given by algorithm 2, henceforth called `gasp`. For each $i = 1, \dots, n_1$, `gasp` forms an estimate of the additional volume that would be classified during the current iteration if $f(x_i)$ were evaluated.

7.3 Experiments with low-dimensional data

To validate the proposed selection procedure, `gasp`, we introduce several baseline procedures and compare them to `gasp`. The procedure `random` picks a point uniformly from the domain at each iteration, without considering what has already been classified or what values the simulation function has returned thus far. The procedure `minimax` picks the point that has the fewest classified points within an ϵ radius of it. The procedure `omniscient` cheats. It is identical to `gasp`, except that in step 2, rather than drawing points from $\mathcal{N}(\hat{f}(x_i), \hat{\sigma}(x_i))$, it draws the exact value of the simulation function at the point in question.

We test these selection procedures on three functions: `flat`, `slab` and `saddle`. `flat` is $f : [0,1]^2 \to \mathbb{R}$ satisfying $f(x) = 0.1$. `slab` and `saddle` are shown in figures 7.1 and 7.2, respectively. We aim to test how rapidly each selection procedure classified the domain. Because our selection procedures are stochastic, and because our method for determining how well each procedure performed is also stochastic, we base our conclusions on at least 10 runs of each combination of selection procedure and test function.

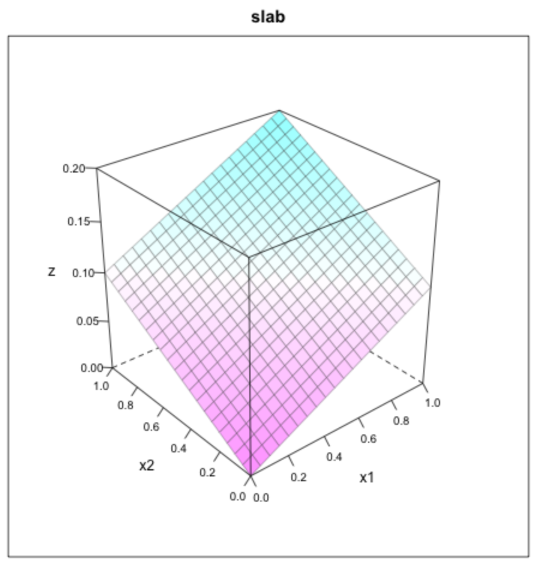


Figure 7.1: Wire frame plot of $f : [0,1]^2 \to \mathcal{R}$ s.t $f(x) = \frac{x_1+x_2}{10}$

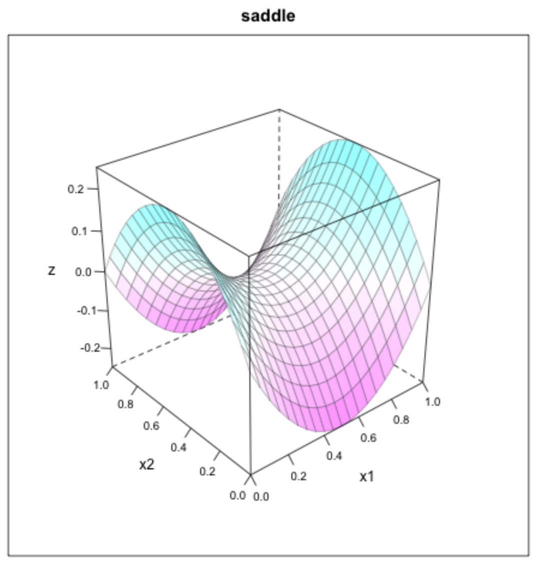


Figure 7.2: Wire frame plot of $f : [0,1]^2 \to \mathcal{R}$ s.t $f(x) = (x_1 - 0.5)^2 - (x_2 - 0.5)^2$

A box plot (figure 7.3) shows that when contouring `flat`, the strategies `minimax`, `gasp` and `omniscient` each classify virtually all of the domain after 75 iterations. `random` is significantly slower. The true optimal strategy would require just 25 iterations, since on `flat`, choosing points in accordance with an appropriately spaced grid will classify 4% of the domain per iteration; upon determining that $f(x)$ evaluates to 0.1, we can classify the sup-norm ball $B(x, 0.1)$, and the volume of this ball is 0.04. After 25 iterations, both `gasp` and `omniscient` classify roughly 85% of the domain.

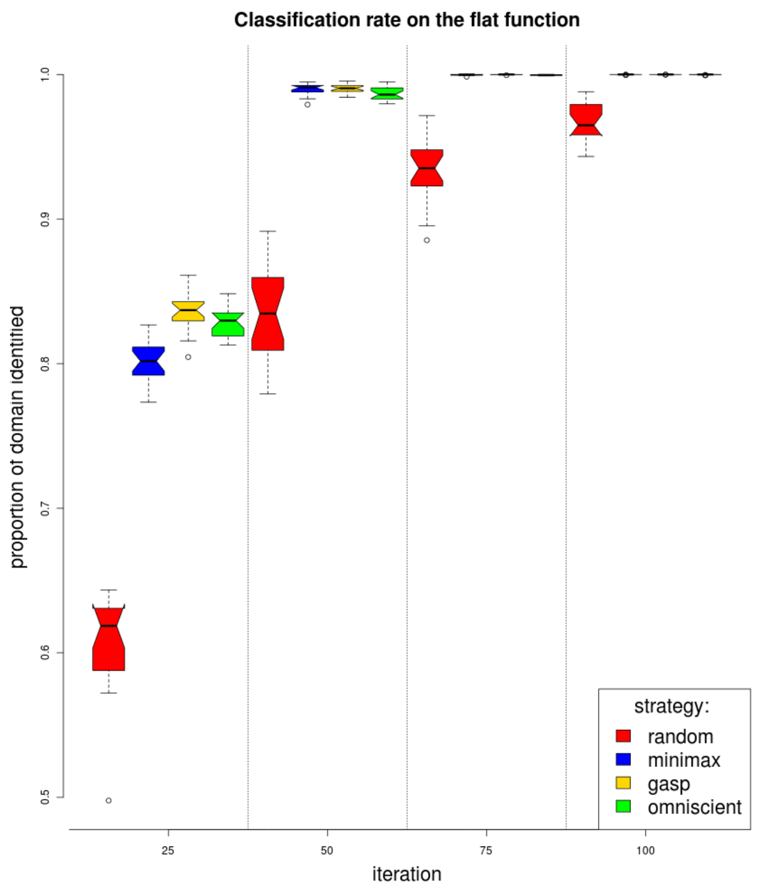


Figure 7.3: The proportion of the domain classified when various strategies are employed to select points while contouring the flat function $f(x) = 0.1$

A box plot (figure 7.4) shows that when contouring `saddle`, `minimax` outperforms `random` at all iterations, while `gasp` and `omniscient` both outperform `minimax` at all iterations. `omniscient` outperforms `gasp` during the first 50 iterations, but after that the best method is less clear. After 50 iterations, `gasp` makes estimates that are nearly perfect.

Figure 7.5 shows which points `omniscient` selects at various iterations while contouring the `saddle`. During early iterations, points mapping to values far from the threshold are preferentially selected, since initially such selections classify the greatest volume of the domain. However, after 10 iterations, points with $f(x)$ closer to the threshold start to be preferred, since the rest of the domain has already been classified. This progression is clearer in figure 7.6, which displays the points selected by `omniscient` while contouring `slab`.

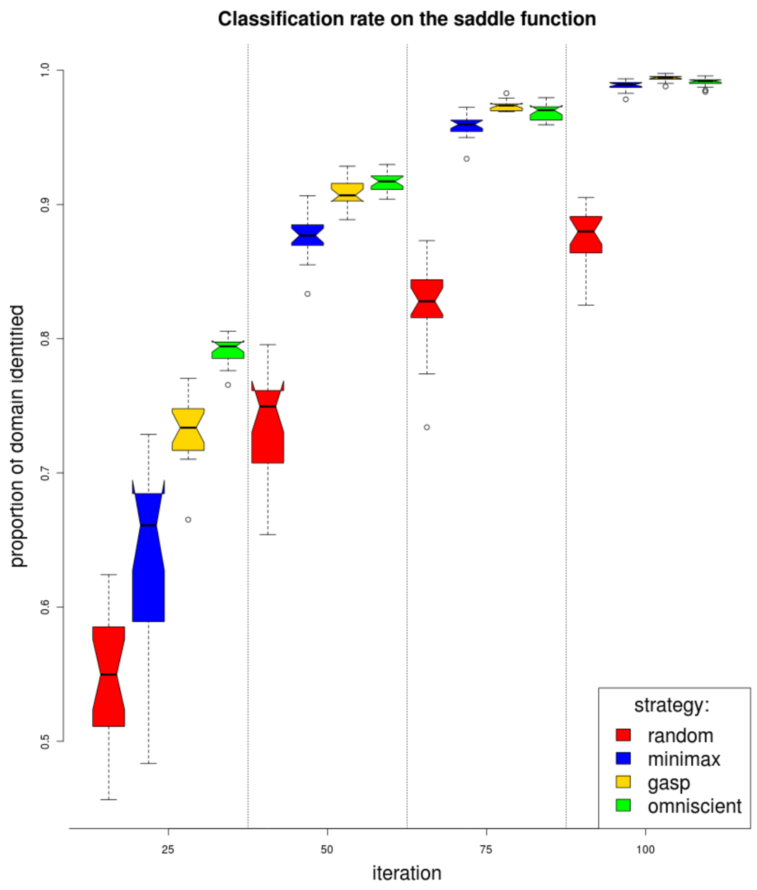


Figure 7.4: The proportion of the domain classified when various strategies are employed to select points while contouring the saddle function $f(x) = (x_1 - 0.5)^2 - (x_2 - 0.5)^2$

7.4 Experiments with high-dimensional data

With all routines performing well in low-dimensional space, we turn to testing them in high-dimensional spaces. Our first high-dimensional test function is $f : [0,1]^{25} \to \mathbb{R}$ satisfying $f(x) = 0.57$. It is henceforth referred to as `flat25`. Picking a high-dimensional constant test function is challenging, because we need a height such that the strategies would finish classifying the domain within a tractable number of iterations, but without finishing on the first iteration. Figure 7.7 shows that `flat25` strikes such a balance, and that as in low-dimensional space, `gasp` and `omniscient` perform the best. However, the difficulty of selecting 0.57 was concerning; a slightly lower value lets even `random` finish in 1 iteration, while a slightly higher value prevents even `omniscient` from making measurable progress. Any real application seemed unlikely to strike such a delicate balance.

Upon validating our algorithm and implementation on a variety of closed form functions, we tested it on a function that more closely resembled the actual simulation function. We created such a test function by fitting a Gaussian process model to the LLNL dataset, and using the resulting model as a map from any point in the parameter space. Because we were taking 0 to be the level set's threshold, we figured that shifting the test function to have mean 0 would make for a substantive test scenario. The test function would also need to be scaled so it would have Lipschitz constant 1. It would have been convenient if an expert with domain knowledge could have given an upper bound on the simulation function's Lipschitz constant. Better yet, such an expert would provide separate Lipschitz constants for each of the 21 parameters. However, we was not able to obtain any such constants, so we estimated the Lipschitz constant from the dataset itself, by comparing every pair of runs. Under sup-norm, the dataset never exhibited a slope greater than 35. Under L_1-norm, the dataset never exhibited a slope greater than 15. Note that the L_1-norm is always greater than the sup-norm, so the slope under it is always less. However, the maximum distances between two points under sup-norm and L_1-norm are 1 and 21, respectively. It is also important to note that our estimates of Lipschitz constants are lower bounds; we only compared some subset of the pairs of points in the true parameter space (i.e., the points for which we had data). The true Lipschitz constant could only be higher, which could only make it more difficult to identify level sets.

We then tested our algorithm on this test function, scaled by optimistic estimates of the Lipschitz constant. After any feasible number of function evaluations, our estimate of the proportion of the domain classified remained at 0. As the dimension N increases, the volume of cubes with a constant radius decreases geometrically. For example, in 21 dimensions, the cube with radius 0.8 occupies less than 1% of the volume of the unit cube, while the cube with radius 0.5 occupies less than 0.00005% of the volume of the unit cube. Had we fallen victim to the curse of dimensionality?

7.5 Conclusions

With additional conditions on f, we might be able to classify the entire domain of f based on only a modest number of observations of f. For example, if f were known to be linear with respect to some polynomial basis having only low-order interaction terms, with observations corrupted by additive Gaussian noise, then standard design of experiments methodolgy applies. Fractional factorial design would detect interactions, and extraneous interaction terms could be removed from the model. Once the model had been pruned sufficiently, we might iteratively select points (or batches of points) from the parameter space that lead to the D-optimal or G-optimal design. After exausting our budget of simulations, we could compute the variance of the fitted model at any point in the parameter space by analyzing the residuals. There is some reason to think that f could approximately have a polynomial basis: a polynomial can approximate any function that has a Taylor expansion arbitrarily well. But we do not know that a Taylor approximation composed of only low-order terms would suffice.

Another direction for future research would be a change the problem: rather than identify level sets of f, directly solve the problem finding level sets was thought relevant to answering. Ultimately climate modelers concerned about global warming want to know how much Earth's temperature will increase. Moreover, the ranges of parameter values deemed reasonable was itself determined by running the models, and comparing the output to historical data. Perhaps we could bound the simulation function's output by a function of models' differences on estimates of historical data, and avoid finding level sets entirely.

Rather pursuing either possible remedy, the following chapter developes negative results about learning black-box functions in high-dimensional spaces from any tractable number of function evaluations.

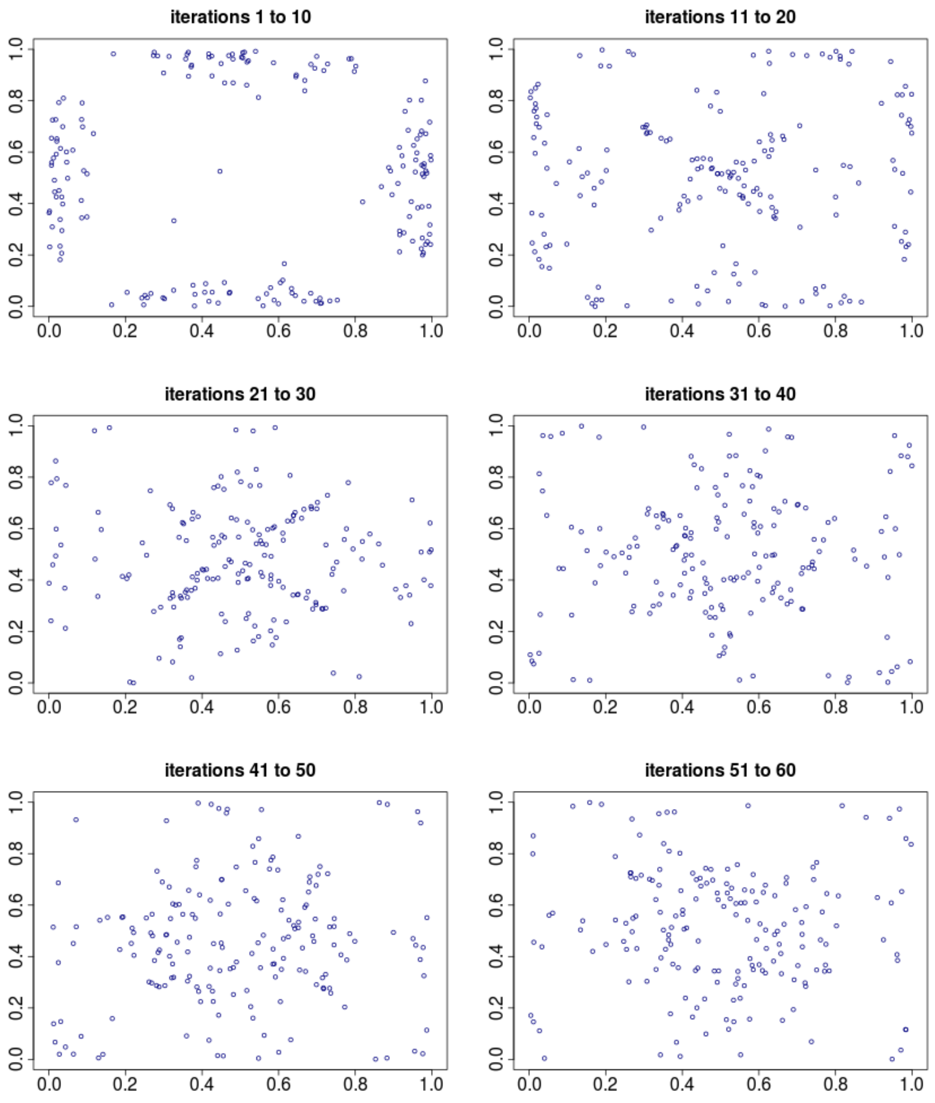


Figure 7.5: The points `omniscient` selects at various iterations while contouring the saddle function during 10 replications

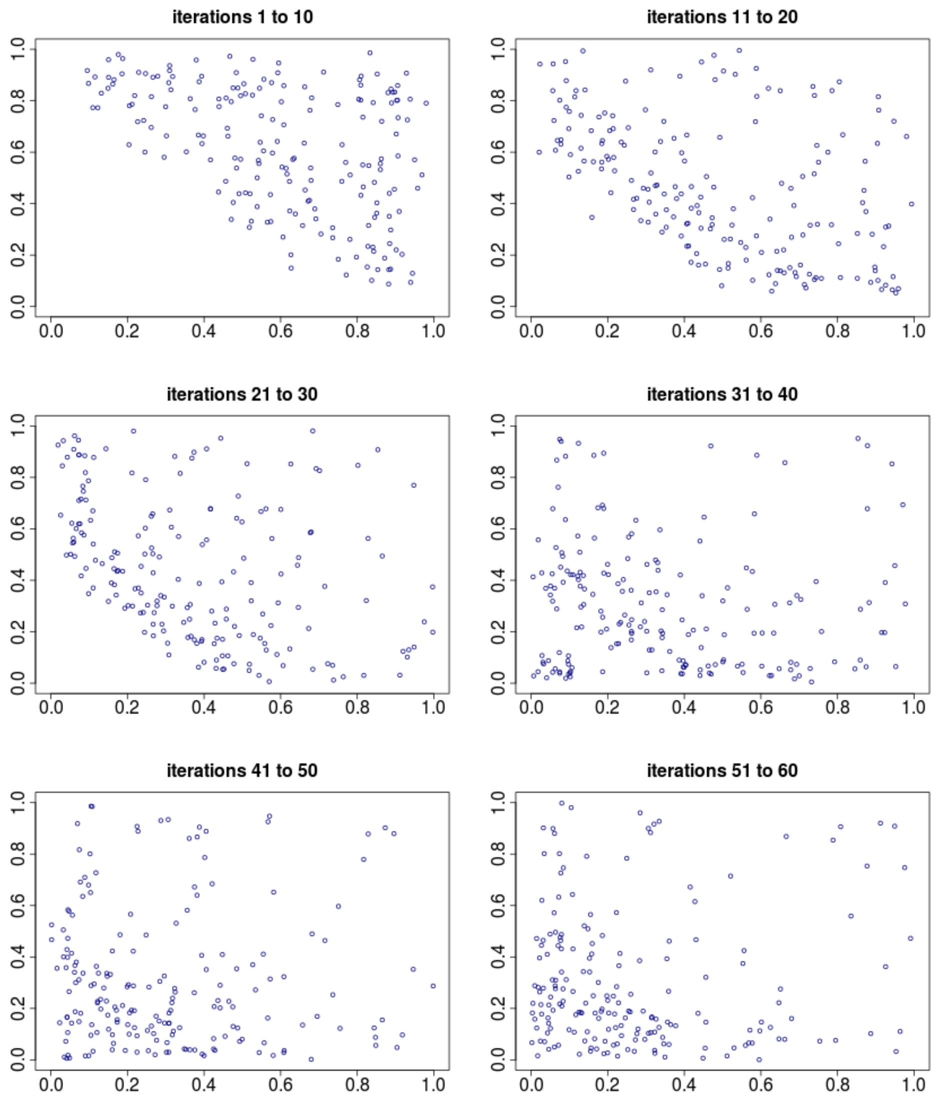


Figure 7.6: The points `omniscient` selects at various iterations while contouring the slab function during 10 replications

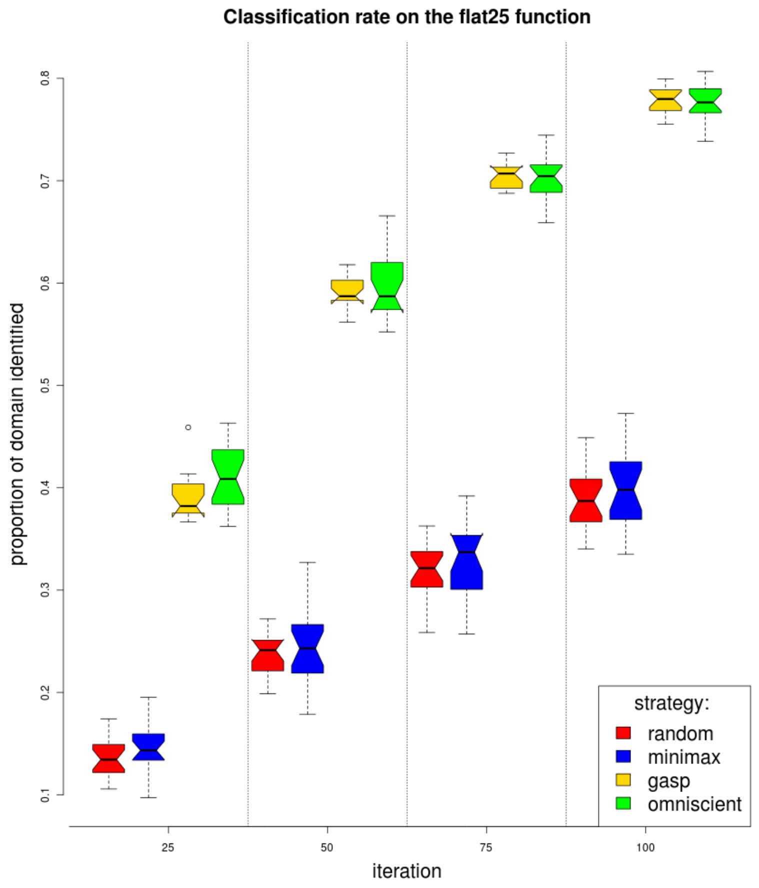


Figure 7.7: The proportion of the domain classified when various strategies are employed to select points while contouring the 25-dimensional flat function $f(x) = 0.57$

Chapter 8

Mini-Minimax Uncertainty Quantification for Emulators

This chapter studies the accuracy of emulators, also known as surrogate functions and meta-models. Emulators are important tools for approximating functions that have been observed only partially. Kriging, Multivariate Adaptive Regression Splines (MARS), Projection Pursuit Regression, Polynomial Chaos Expansions (PC), Gaussian Process models (GP), and other Bayesian modeling techniques are common methods for constructing emulators [7, 8, 9]. We find error bounds for emulators in general—including the "best possible" method—rather than focusing on any particular emulation method.

Emulators are frequently used to approximate expensive computer models, which are often deterministic functions.[1] Resources limit the number of times the computer model can be run, though typically an intractable number of inputs are possible—for instance if any input parameter is a floating point number. By fitting an emulator to the output of a tractable number of runs for different inputs, one can approximate the computer model inexpensively; the issue is the accuracy of that approximation.

Computer models known as *HEB* [104] may be particularly difficult to emulate: They depend on *H*igh-dimensional inputs; they are *E*xpensive to run; and they are effectively *B*lack boxes that are not amenable to closed-form, analytic study. Because such models have high-dimensional inputs, it takes prohibitively many runs to explore their domains: to attain a given sample density, the number grows exponentially in the dimension. Because the models are expensive, performing many runs is impractical or impossible. And because the models are black boxes, there are few (if any) constraints to ensure that the error in

[1]They might not be entirely deterministic; for instance, they could involve Monte Carlo simulations. Moreover, in distributed parallel computations, numerical results can depend on the order in which subproblems happen to complete. These cases can be thought of as observing the function with noise. We do not address noise here; however, uncertainty in the observations makes accurate approximation more difficult. Because we focus on lower bounds on the difficulty of approximating the function accurately, our results generally remain lower bounds when the observations are not only incomplete, but also noisy. To extend our methods to include noise would involve finding a lower confidence bound on the regularity of the function.

extrapolating from inputs actually tried to inputs not sampled is small.

HEB problems arise often in practice, for instance:

- Climate models: [105] (21–28-dimensional domains; 1154 simulations; Kriging and MARS)
- Automobile crashes: [106] (15-dimensional domain; 55 simulations; polynomial response surfaces and artificial neural networks).
- Chemical reactions: [107] (30–50-dimensional domain; boosted surrogate models) and [108] (46-dimensional domain; seconds per simulation).
- Aircraft design: [109] (25-dimensional domain; 500 simulations; response surfaces and Kriging), [110] (22-dimensional domain; minutes per simulation; response surfaces and Kriging), and [111] (31-dimensional domain; 20 minutes to several days per simulation; Kriging).
- Electric circuits: [112] (60-dimensional domain; 216 simulations; Kriging).

How accurately can a function f be emulated from a given set of data? How many evaluations of f are required to guarantee that f can be emulated to a given level of accuracy?

Since f is a "black box," we do not know how rough it might be: extrapolating beyond the data could entail arbitrarily large errors. We assume that f is regular and find the resulting uncertainty in emulating f. If the regularity assumption fails, the uncertainty would be larger. We measure the regularity of f by its *absolute condition number* or *Lipschitz constant* K. Similar results could be derived for other measures of regularity, but Lipschitz bounds are particularly amenable to analysis.

The observations impose a lower bound $\hat{K}$ on K. Suppose, optimistically, that the true Lipschitz constant of f is equal to this lower bound. Then f might be any member of the set $\mathcal{F}_{\hat{K}}$ of functions that agree with the observations and have Lipschitz constant no greater than $\hat{K}$. If an emulator is guaranteed to do well no matter which member of $\mathcal{F}_{\hat{K}}$ f happens to be, then the uncertainty of that emulator is low. On the other hand, if there are elements of $\mathcal{F}_{\hat{K}}$ that an emulator cannot approximate well, the uncertainty is large.

Consider *all* emulators that can be computed from the observations alone, without additional knowledge of f; this collection includes emulators constructed using GP, PC, MARS, and all the other methods mentioned above. Viewed as a function of w in the domain of f, the minimax error among such emulation methods over the set $\mathcal{F}_{\hat{K}}$ of functions that agree with the observations and have Lipschitz constant no greater than $\hat{K}$ is the *mini-minimax uncertainty* $\mathcal{E}_{\hat{K}}(w)$ in the title of this chapter.

The first "mini" refers to the regularity condition: since K is not smaller than $\hat{K}$, $\mathcal{E}_{\hat{K}}(w)$ is a lower bound on the minimax uncertainty for functions that are as regular as f. The second "mini" refers to emulators: this is the uncertainty for the best emulator—including all the standard ones. The "max" is over functions that agree with f at the observations and satisfy the optimistic regularity condition. That is, $\mathcal{E}_{\hat{K}}(w)$ is the smallest that the uncertainty at w could be, for the best emulator, over the set of functions that have the highest degree of regularity consistent with the observations and that agree with the observations. The maximum of $\mathcal{E}_{\hat{K}}(w)$ over w in the domain of f is an attainable lower bound on the maximum uncertainty of any emulator $\hat{f}$ of f.

If K were known, this would be a standard problem in information-based complexity [113, 114, 115]. We derive bounds on the uncertainty using the lower bound $\hat{K}$ computed from the observed variation of f. Section 8.2 derives a lower bound on the number of additional observations that might be necessary to learn f. Section 8.3 derives two lower bounds on the maximum uncertainty for approximating f from a fixed set of observations: a purely empirical bound and a bound expressed as a fraction of the unknown Lipschitz constant. The latter yields conditions under which emulating f by a constant function, equal to the value of f at the centroid of its domain, has smaller maximum uncertainty than any emulator based on the n actual observations.

Section 8.4 applies these bounds to two closed-form functions (a high-dimensional cone and the *borehole function* [116]) and to a black-box function (the Community Atmosphere Model [105]). Section 8.5 extends the results for the maximum error to quantiles of the error and the mean of the error over the domain of f. Section 8.6 gives our conclusions.

8.1 Notation and problem formulation

The function f is a fixed unknown real-valued function on $[0,1]^p$, the p-dimensional unit cube. The space of real-valued continuous functions on $[0,1]^p$ is $\mathcal{C}[0,1]^p$. The Roman letters i, j, p, q, and M_ϵ denote integers. Lowercase Greek letters denote real scalars, with the exception of μ, which denotes Lebesgue measure. Uppercase Roman letters such as X and D denote subsets of $[0,1]^p$; X is a fixed finite subset of $[0,1]^p$. Lowercase Roman letters from the end of the alphabet, such as v, w, x, y, and z, denote points in $[0,1]^p$. The lowercase Roman letters e, f, g, and h denote real-valued functions on (subsets of) $[0,1]^p$. The domain of a function g is $\mathrm{dom}(g)$. The restriction of a function g to $D \subset \mathrm{dom}(g)$ is denoted $g|_D$. The observations from which f is to be emulated are $f|_X$; that is, we observe f on the set X. An emulator $\hat{f}$ is a real-valued function on $[0,1]^p$. Let $\|h\|_\infty \equiv \sup_{w\in\mathrm{dom}(h)} |h(w)|$, the infinity-norm of h. This chapter studies how large $|\hat{f}(w) - f(w)|$ and $\|\hat{f} - f\|_\infty$ could be, for the best $\hat{f}$ chosen on the basis of the data—without other information about f.

Let d be a metric on $\mathrm{dom}(g)$. The (best) Lipschitz constant for g is

$$\mathrm{Lip}(g) \equiv \sup\left\{\frac{g(v) - g(w)}{d(v,w)} : v, w \in \mathrm{dom}(g) \text{ and } v \neq w\right\}. \tag{8.1}$$

If $f \notin \mathcal{C}[0,1]^p$, then $\mathrm{Lip}(f) \equiv \infty$. Define

$$\mathcal{F}_\kappa(g) \equiv \{(h : [0,1]^p \to \mathfrak{R}) : \mathrm{Lip}(h) \leq \kappa \text{ and } h|_{\mathrm{dom}(g)} = g\}.$$

symbol	meaning
f	unknown function on $[0,1]^p$ to be emulated
$\hat{f}$	an emulator
X	finite subset of $[0,1]^p$ where f is observed
$g\vert_Y$	the restriction of the function g to the set $Y \subset [0,1]^p$
$f\vert_X$	*the data*: the restriction of f to X
K	Lipschitz constant of the function f
$\hat{K}$	smallest Lipschitz constant of any function that interpolates the data
$\mathcal{F}_{\kappa,Y}$	all functions that interpolate $f\vert_Y$ and have Lipschitz constant no larger than κ.
$\mathcal{F}_\kappa$	$\mathcal{F}_{\kappa,X}$
$e^+_\kappa(w)$	maximum value at w among functions in $\mathcal{F}_\kappa$
$e^-_\kappa(w)$	minimum value at w among functions in $\mathcal{F}_\kappa$
$\hat{f}_\kappa(w)$	mean of $e^+_\kappa(w)$ and $e^-_\kappa(w)$; the minimax emulator at the point w over functions in $\mathcal{F}_\kappa$
$\mathcal{E}_{\kappa,Y}(w;\hat{f})$	*maximum uncertainty of $\hat{f}$ at w*: uncertainty of $\hat{f}$ at the point w over functions in $\mathcal{F}_{\kappa,Y}$
$\mathcal{E}_{\kappa,Y}(w)$	*minimax uncertainty at w*: uncertainty of the best possible emulator at the point w over functions in $\mathcal{F}_{\kappa,Y}$
$\mathcal{E}_{\kappa,Y}(\hat{f})$	*maximum uncertainty of $\hat{f}$*: maximum (over $w \in [0,1]^p$) uncertainty of $\hat{f}$ over functions in $\mathcal{F}_{\kappa,Y}$
$\mathcal{E}_{\kappa,Y}$	*minimax uncertainty*: maximum (over $w \in [0,1]^p$) uncertainty of the best possible emulator over functions in $\mathcal{F}_{\kappa,Y}$
$\mathcal{E}_\kappa(\cdots)$	when $Y = X$, we generally suppress X from the subscript, viz., $\mathcal{E}_\kappa(w;\hat{f})$, $\mathcal{E}_\kappa(w)$, $\mathcal{E}_\kappa(\hat{f})$, and $\mathcal{E}_\kappa$
M_ϵ	*minimum computational burden*: a lower bound on the number of additional observations needed to guarantee that the minimax uncertainty is no larger than ϵ

Table 8.1: Summary of key notation

Then $\mathcal{F}_\infty(f|_X)$ is the space of (possibly discontinuous) functions that fit the n data. Some of our results involve values of f at points other than the points X at which f was observed; Y denotes a generic set of points in the domain of f. To simplify notation, we set

$$\mathcal{F}_{\kappa,Y} \equiv \mathcal{F}_\kappa(f|_Y).$$

When $Y = X$, we generally write $\mathcal{F}_\kappa$ in place of $\mathcal{F}_{\kappa,X}$.

Definition. The *uncertainty at w of $\hat{f} : [0,1]^p \to \mathfrak{R}$ over the set of functions $\mathcal{F}_{\kappa,Y}$* is

$$\mathcal{E}_{\kappa,Y}(w;\hat{f}) \equiv \sup_{g\in\mathcal{F}_{\kappa,Y}} |\hat{f}(w) - g(w)|$$

The *minimax uncertainty at w over the set of functions $\mathcal{F}_{\kappa,Y}$* is

$$\mathcal{E}_{\kappa,Y}(w) \equiv \inf_{\hat{f}:[0,1]^p\to\mathfrak{R}} \mathcal{E}_{\kappa,Y}(w;\hat{f}).$$

The *maximum uncertainty of $\hat{f} : [0,1]^p \to \mathfrak{R}$ over the set of functions $\mathcal{F}_{\kappa,Y}$* is

$$\mathcal{E}_{\kappa,Y}(\hat{f}) \equiv \sup_{w\in[0,1]^p} \mathcal{E}_{\kappa,Y}(w;\hat{f}) = \sup_{g\in\mathcal{F}_{\kappa,Y}} \|\hat{f} - g\|_\infty.$$

The *minimax maximum uncertainty over the set of functions $\mathcal{F}_{\kappa,Y}$* is

$$\mathcal{E}_{\kappa,Y} \equiv \inf_{\hat{f}:[0,1]^p\to\mathfrak{R}} \mathcal{E}_{\kappa,Y}(\hat{f}).$$

The emulator $\hat{f}$ approximates f within $\mathcal{E}_\infty(w;\hat{f})$ at the point w if f is in $\mathcal{F}_\infty$, the set of functions that agree with the observations. However, $\mathcal{E}_\infty(w;\hat{f})$ is infinite for every $\hat{f}$ unless $w \in X$, even if f is guaranteed to be continuous.[2] To guarantee that the uncertainty is finite requires stronger regularity than mere continuity.

Let $K \equiv \mathrm{Lip}(f)$ and $\hat{K} \equiv \mathrm{Lip}(f|_X)$. Because $X \subset [0,1]^p$, $\hat{K} \le K$, as illustrated in figure 8.1. (There and in subsequent figures, $p = 1$ and the bold black dots represent $f|_X$, the observations of f at $x \in X$.)

Define

$$e^+_\kappa(w) \equiv \min_{x\in X} [f(x) + \kappa d(x,w)]$$

and

$$e^-_\kappa(w) \equiv \max_{x\in X} [f(x) - \kappa d(x,w)].$$

The mean of the two is

$$\hat{f}_\kappa(w) \equiv \hat{f}_\kappa(w;X,\kappa) \equiv \frac{e^-_\kappa(w) + e^+_\kappa(w)}{2}.$$

Figures 8.2 and 8.3 illustrate these definitions. The proof of Proposition 8.1 shows that the function $\hat{f}_\kappa(w)$ is the minimax emulator for pointwise error over the class $\mathcal{F}_\kappa$ of functions that agree with the data and have Lipschitz constant no greater than κ. The minimax emulator $\hat{f}_\kappa(w)$ interpolates (rather than smooths) the data.

[2]The set X is not dense in $[0,1]^p$, so for any $c > 0$, there exists some function $g \in \mathcal{F}_\infty(f|_X)$ satisfying $\|f - g\|_\infty > c$.

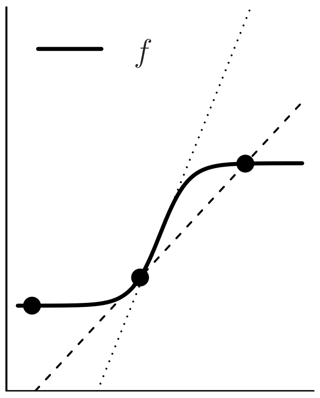


Figure 8.1: Illustration of the difference between the true Lipschitz constant K and the empirical lower bound $\hat{K}$ for K. The dotted line is tangent to f where f attains its Lipschitz constant: it has slope K. The dashed line is the steepest line that intersects any pair of observations: it has slope $\hat{K} \leq K$.

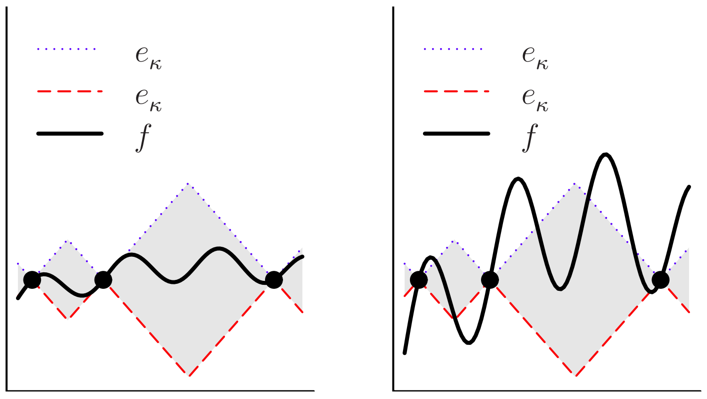


Figure 8.2: Illustration of the upper and lower envelope functions e_κ^- and e_κ^+ when $\kappa = K$ and when $\kappa < K$, and the derived estimate of In both panels, the optimal interpolant $\hat{f}_\kappa$ is constant. In the left panel $\kappa = K$; in the right, $\kappa < K$. If $\kappa \geq K$ then $e_\kappa^- \leq f \leq e_\kappa^+$, and, equivalently, $f \in \mathcal{F}_\kappa$.

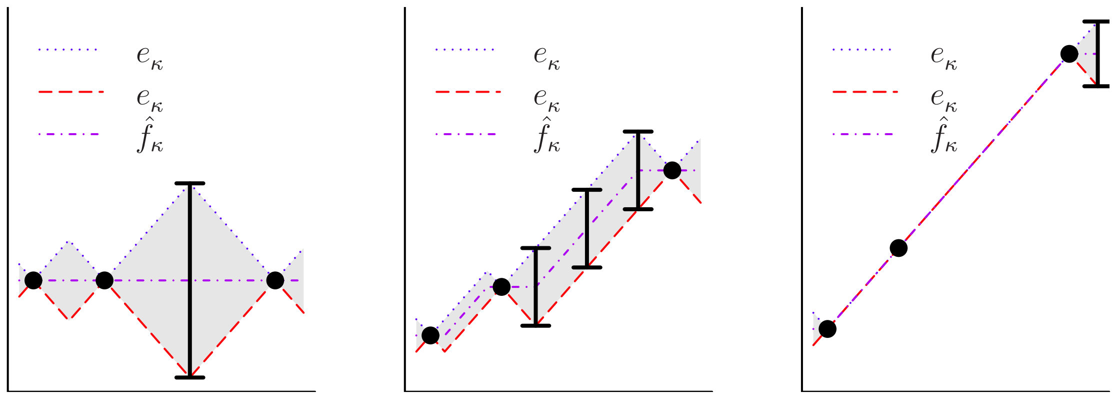


Figure 8.3: Illustration of how the pointwise uncertainty depends on the observed variation of f: the uncertainty is smaller where the data require f to vary rapidly. The vertical distance between the blue and red curves is twice the uncertainty at the corresponding abscissa. The black error bars are at some points where the uncertainty is largest. The succession of panels shows that as the slope between observations approaches κ, $\mathcal{E}_\kappa(w)$ approaches 0 for points w between observations, and the maximum uncertainty decreases.

8.2 Bounds on the number of observations needed to approximate f well

In this section we construct a function $\bar{f}$ that agrees with the data $f|_X$, has Lipschitz constant $\hat{K}$ (the smallest Lipschitz constant consistent with the data), and yet would require a large number M_ϵ of additional observations $f|_Y$ to estimate f within ϵ on $[0,1]^p$.[3] The function $\bar{f}$ is not intended to be an emulator—it is a technical device. Since f could in fact be $\bar{f}$, this gives a lower bound on the number of additional observations that might be required to estimate f well, even if f is no rougher than the original data $f|_X$ reveal it to be.

Let $B(x,\delta)$ denote the open ball in $\mathbb{R}^p$ centered at x with radius δ. Since f has Lipschitz constant K, $f(y)$ is guaranteed to be within ϵ of $f(x)$ if $y \in B(x,\epsilon/K)$. But depending on f and X, it can happen that $\hat{f}_{\hat{K}}$ is guaranteed to be within $\pm\epsilon$ of every $g \in \mathcal{F}_K$ for parts of the domain not contained in $\cup_{x \in X} B(x,\epsilon/K)$. To see this, consider $p = 1$, $f(x) = x$, and let X be the two-element set $\{0,1\}$. Then $K = \hat{K} = 1$. In this case, the observations $f|_X$ determine f exactly: the only function in $\mathcal{F}_K$ is f. In this example, for a function g to agree with the observations requires it to attain the Lipschitz constant K everywhere. A function cannot agree with the observations and "run away" from f very far.

More generally, if f varies on X, then for a function g to agree with f at the observations, g must vary too. That required variation "spends" some of g's Lipschitz constant, preventing g from running as far away from f as it could if f_X were constant. We now quantify this

[3]We do not discuss the choice of $\epsilon > 0$ in detail: scientific context should inform the choice. In examples below, we set ϵ to be an absolute tolerance, a fraction of $\hat{K}$, and a fraction of K. One might also consider relating ϵ to the "typical value" of f (e.g., the mean of f or of $f|_X$).

intuition to construct a function $\bar{f}$ that requires many additional observations to estimate well. The function $\bar{f}$ is constant "as much as possible" subject to the constraint that it interpolates the data and has Lipschitz constant $\hat{K}$. Since estimating $\bar{f}$ where it is constant is hard (as illustrated in figure 8.3), the size of the set where $\bar{f}$ could be constant gives a lower bound on the number of additional observations that might be required.

Define $\bar{\gamma} \equiv \arg\min_{\gamma\in\mathbb{R}} \sum_{x\in X} |f(x) - \gamma|^p$. Computing $\bar{\gamma}$ is straightforward because the objective function is univariate and convex.[4] Let $X^+ \equiv \{x \in X : f(x) \geq \bar{\gamma}\}$ and let $X^- \equiv \{x \in X : f(x) < \bar{\gamma}\}$. Let

$$Q_+ \equiv \bigcup_{x\in X^+} \left\{ B\left(x, \frac{f(x) - \bar{\gamma}}{\hat{K}}\right) \bigcap [0,1]^p \right\}$$

and

$$Q_- \equiv \bigcup_{x\in X^-} \left\{ B\left(x, \frac{\bar{\gamma} - f(x)}{\hat{K}}\right) \bigcap [0,1]^p \right\}.$$

Then $Q_+ \cap Q_- = \varnothing$.[5]

Define

$$\begin{aligned} \bar{f} &: [0,1]^p \to \mathbb{R} \\ w &\mapsto \begin{cases} e^-_{\hat{K}}(w), & w \in Q_+ \\ e^+_{\hat{K}}(w), & w \in Q_- \\ \bar{\gamma}, & \text{otherwise.} \end{cases} \end{aligned}$$

Figure 8.4 illustrates this definition. If we know $f|_X$, we know $\bar{f}$. By construction, $\bar{f} \in \mathcal{F}_{\hat{K}} \subset \mathcal{F}_K$.

Let $\bar{Q} \equiv [0,1]^p \setminus (Q_+ \cup Q_-)$. Let μ be Lebesgue measure. By the union bound, because $\mu([0,1]^p) = 1$,

$$\mu(\bar{Q}) \geq 1 - \sum_{x\in X} \mu\left(B\left(x, |f(x) - \bar{\gamma}|/\hat{K}\right)\right).$$

Let $C_2 \equiv \frac{\pi^{p/2}}{\Gamma(p/2+1)}$ and $C_\infty \equiv 2^p$, where Γ is the gamma function. Then, for $q \in \{2, \infty\}$,

$$\mu(\bar{Q}) \geq 1 - C_q \sum_{x\in X} \left(|f(x) - \bar{\gamma}|/\hat{K}\right)^p.$$

[4]Alternatively, we could set $\bar{\gamma} \equiv \frac{1}{\#X}\sum_{x\in X} f(x)$, where $\#X$ is the size of X. The resulting lower bound may not be as tight.

[5]Fix $x^+ \in X^+$ and $x^- \in X^-$. Then $|f(x^+) - f(x^-)|/d(x^+, x^-) \leq \hat{K}$. Equivalently, $d(x^+, x^-) \geq |f(x^+) - f(x^-)|/\hat{K}$. Let $B^+ = B\left(x^+, [f(x) - \bar{\gamma}]\hat{K}\right)$ and $B^- = B\left(x^-, [\bar{\gamma} - f(x)]/\hat{K}\right)$. Let a be the sum of the radii of B^+ and B^-. Then $a = (f(x^+) - \bar{\gamma})/\hat{K} + (\bar{\gamma} - f(x^-))/\hat{K} = (f(x^+) - f(x^-))/\hat{K}$, and $a \leq d(x^+, x^-)$. Therefore, $B^+ \cap B^- = \varnothing$. Because our selection of $x^+ \in X^+$ and $x^- \in X^-$ was arbitrary, $Q^+ \cap Q^- = \varnothing$.

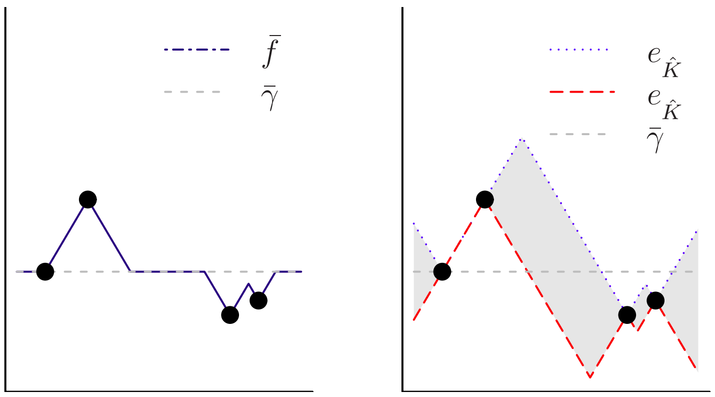


Figure 8.4: A function that agrees with the data, has Lipschitz constant $\hat{K}$, and is hard to estimate because it is often constant. The function $\bar{f}$ (shown in the left panel) is comprised of segments of $e^{+}_{\hat{K}}$, $e^{-}_{\hat{K}}$ and the constant function $\bar{\gamma}$ (all shown in the right panel). It is constant over roughly half of the domain. No function between $e^{-}_{\hat{K}}$ and $e^{+}_{\hat{K}}$ (inclusive) is constant over a larger fraction of the domain.

If there is some $x \in X$ for which for all $g \in \mathcal{F}_{\hat{K},\{x\}}$, $|g(y) - f(x)| < \epsilon$ for all $y \in A \subset \bar{Q}$, then $\mu(A) \leq \mu(B(0, \epsilon/\hat{K}))$. Hence, because $\bar{f} \in \mathcal{F}_K$,

$$\begin{aligned} M_\epsilon &\geq \left\lceil \frac{\mu(\bar{Q})}{\mu(B(0,\epsilon/\hat{K}))} \right\rceil \\ &\geq \left\lceil \epsilon^{-p} \left[\frac{\hat{K}^p}{C_q} - \sum_{x \in X} |f(x) - \bar{\gamma}|^p \right] \right\rceil . \end{aligned} \tag{8.2}$$

Section 8.4 shows that this lower bound, the *minimum computational burden*, can be extremely large for even modest problem dimensions p.

8.3 Bounds on the maximum uncertainty for a fixed experimental design

The previous section gave lower bounds on the number of additional observations of f required to attain a desired maximum uncertainty ϵ. This section gives two lower bounds on the maximum uncertainty $\mathcal{E}_K(\hat{f})$ for a fixed experimental design X: an absolute bound and a bound expressed as a fraction of K. The bound as a fraction of K can yield a strong negative result: when a statistic—calculable from the observations—exceeds a calculable threshold, the maximum uncertainty is not less than the maximum uncertainty of the best emulator based on a single observation at the centroid of the domain. If the goal is to minimize the maximum uncertainty, we could have just approximated f as constant and saved $\#X - 1$ observations.

8.3.1 Lower bounds

Consider the set $\mathcal{F}_\kappa$ of functions g that agree with the observations $f|_X$ and have Lipschitz constant no larger than κ. Consider all possible emulators $\hat{f}$. Proposition 8.1 states that the smallest (across emulators $\hat{f}$) maximum (across functions g) error at the point $w \in [0,1]^p$ is $[e_\kappa^+(w) - e_\kappa^-(w)]/2$, and the emulator $\hat{f}_\kappa(w)$ attains this bound at every w.

Proposition 8.1. *If $\kappa \geq \hat{K}$, then*

$$\mathcal{E}_\kappa(w) = \mathcal{E}_\kappa(w; \hat{f}_\kappa) = \frac{e_\kappa^+(w) - e_\kappa^-(w)}{2}.$$

Proof.

Step 1: e_κ^+ and e_κ^- are Lipschitz continuous with constant κ.
For $v, w \in [0,1]^p$, $\exists x, y \in X$ satisfying

$$e_\kappa^+(v) = f(x) + \kappa d(x,v) \text{ and } e_\kappa^+(w) = f(y) + \kappa d(y,w).$$

Suppose without loss of generality that $e_\kappa^+(v) \geq e_\kappa^+(w)$. By construction, $e_\kappa^+(v) \leq f(y) + \kappa d(y,v)$. Hence

$$\begin{aligned} 0 \leq e_\kappa^+(v) - e_\kappa^+(w) &\leq f(y) + \kappa d(y,v) - e_\kappa^+(w) \\ &= f(y) + \kappa d(y,v) - f(y) - \kappa d(y,w) \\ &\leq \kappa(d(y,v) - d(y,w)) \\ &\leq \kappa d(v,y), \end{aligned}$$

by the triangle inequality. Hence e_κ^+ has Lipschitz constant κ. An analogous argument shows that e_κ^- also has Lipschitz constant κ.

Step 2: e_κ^+ and e_κ^- agree with f on X. (Hence, $\hat{f}_\kappa = (e_\kappa^+ + e_\kappa^-)/2$ agrees with f on X.)
We have

$$\kappa \geq \hat{K} \equiv \max_{x,y \in X : x \neq y} \frac{|f(x) - f(y)|}{d(x,y)},$$

and hence $|f(x) - f(y)| \leq \kappa d(x,y)$ for all $x, y, \in X$. Thus

$$\min_{x \in X}[f(x) + \kappa d(x,y)] = \min\left\{f(y), \min_{x \in X, x \neq y}[f(x) + \kappa d(x,y)]\right\} = f(y).$$

Similarly, $\max_{x \in X}[f(x) - \kappa d(x,y)] = f(y)$ for $y \in X$. Hence, $e_\kappa^+(y) = e_\kappa^-(y) = f(y)$ for $y \in X$. Since, as shown in step 1, e_κ^+ and e_κ^- are Lipschitz with constant κ, e_κ^+ and $e_\kappa^- \in \mathcal{F}_\kappa$.

Step 3: e_κ^- is the pointwise infimum of $\mathcal{F}_\kappa$ and e_κ^+ is the pointwise supremum of $\mathcal{F}_\kappa$.
Suppose to the contrary that there exists $w \in [0,1]^p$, $x \in X$, and $g \in \mathcal{F}_\kappa$ for which

$$g(w) > f(x) + \kappa d(x,w).$$

Recall that $g \in \mathcal{F}_\kappa$ implies that $g(x) = f(x)\ \forall x \in X$. Hence

$$g(w) - g(x) > f(x) + \kappa d(x,w) - f(x) = \kappa d(x,w).$$

That is, g has a Lipschitz constant greater than κ, a contradiction. Hence, $e_\kappa^+(w) = \sup\{g(w) : g \in \mathcal{F}_\kappa\}$ for all $w \in [0,1]^p$. The same argument, *mutatis mutandi*, shows that

$$e_\kappa^-(w) = \inf\{g(w) : g \in \mathcal{F}_\kappa\} \text{ for all } w \in [0,1]^p.$$

Step 4: The maximum uncertainty of $\hat{f}_\kappa$ at w, $\mathcal{E}_\kappa(w;\hat{f}_\kappa)$, equals $[e_\kappa^+(w) - e_\kappa^-(w)]/2$.

$$\begin{aligned}
\mathcal{E}_\kappa(w;\hat{f}_\kappa) &\equiv \sup_{g\in\mathcal{F}_\kappa(w)} \left|\hat{f}_\kappa(w) - g(w)\right| \\
&= \max\left\{\sup_{g\in\mathcal{F}_\kappa(w)} g(w) - \hat{f}_\kappa(w), \hat{f}_\kappa(w) - \inf_{g\in\mathcal{F}_\kappa(w)} g(w)\right\} \\
&= \max\left\{e_\kappa^+(w) - \hat{f}_\kappa(w), \hat{f}_\kappa(w) - e_\kappa^-(w)\right\} \qquad (8.3) \\
&= \max\left\{e_\kappa^+(w) - \frac{e_\kappa^+(w) + e_\kappa^-(w)}{2}, \frac{e_\kappa^+(w) + e_\kappa^-(w)}{2} - e_\kappa^-(w)\right\} \\
&= \frac{e_\kappa^+(w) - e_\kappa^-(w)}{2}.
\end{aligned}$$

Equality (8.3) follows from step 3.

Step 5: The minimax uncertainty at w, $\mathcal{E}_\kappa(w)$, equals $[e_\kappa^+(w) - e_\kappa^-(w)]/2$.
Suppose $\hat{f}(w) > \hat{f}_\kappa(w)$. Then

$$|\hat{f}(w) - e_\kappa^-(w)| > \frac{e_\kappa^+(w) - e_\kappa^-(w)}{2} = \mathcal{E}_\kappa(w;\hat{f}_\kappa).$$

Suppose $\hat{f}(w) < \hat{f}_\kappa(w)$. Then

$$|\hat{f}(w) - e_\kappa^+(w)| > \frac{e_\kappa^+(w) - e_\kappa^-(w)}{2} = \mathcal{E}_\kappa(w;\hat{f}_\kappa).$$

Hence, $\hat{f}_\kappa(w)$ is minimax, and $\mathcal{E}_\kappa(w) = \mathcal{E}_\kappa(w,\hat{f}_\kappa) = [e_\kappa^+(w) - e_\kappa^-(w)]/2$. □

Proposition 8.1 gives us Corollary 8.2, our next result, and contributes to the proof of Theorem 8.5.

Corollary 8.2. *For any emulator $\hat{f}$,*

$$\mathcal{E}_K(w,\hat{f}) \geq \mathcal{E}_K(w) \geq \mathcal{E}_{\hat{K}}(w;\hat{f}_{\hat{K}}). \qquad (8.4)$$

Corollary 8.2 follows from proposition 8.1 and the fact that, since $\hat{K} \le K$,

$$\mathcal{F}_{\hat{K}} \subset \mathcal{F}_K.$$

Corollary 8.2 is one of our principal results: $\mathcal{E}_{\hat{K}}$, a statistic calculable solely from the observations $f|_X$, is a lower bound on the maximum uncertainty for *any* emulator $\hat{f}$ based on the observations $f|_X$.

Theorem 8.5 gives a stronger lower bound in terms of the unknown value of K. Two lemmas contribute to the proof of Theorem 8.5, both based on the several definitions. For real χ and ρ, define the interval

$$I(\chi, \rho) \equiv \begin{cases} [\chi - \rho, \chi + \rho], & \rho \ge 0 \\ \varnothing, & \text{otherwise.} \end{cases}$$

If I is an interval, $\mu(I)$ denotes its length; for instance, $\mu(I(\chi, \rho)) = \max(0, 2\rho)$.

Lemma 8.3. Fix $\alpha \in [0,1]$, $\rho_1, \dots, \rho_n \in [0, \infty)$ and $\chi_1, \dots, \chi_n \in \mathbb{R}$. Let $I_1 \equiv \bigcap_{i=1}^n I(\chi_i, \rho_i)$ and $I_\alpha \equiv \bigcap_{i=1}^n I(\chi_i, \alpha\rho_i)$. Then $\alpha\mu(I_1) \ge \mu(I_\alpha)$.

Proof. Because the intersection of intervals is itself an interval, there exist χ_0 and ρ_0 satisfying

$$I_\alpha = I(\chi_0, \rho_0).$$

Fix $i \in 1, \dots, n$. Then

$$I(\chi_0, \rho_0) \subset I(\chi_i, \alpha\rho_i).$$

It follows that

$$\chi_0 - \rho_0 \ge \chi_i - \alpha\rho_i.$$

Then

$$\alpha\left(\rho_i - \frac{\rho_0}{\alpha}\right) \ge \chi_i - \chi_0.$$

Because $\alpha \le 1$ and $\rho_i \ge 0$,

$$\rho_i - \frac{\rho_0}{\alpha} \ge \chi_i - \chi_0.$$

Finally,

$$\chi_0 - \frac{\rho_0}{\alpha} \ge \chi_i - \rho_i.$$

By symmetric reasoning we also have

$$\chi_0 + \frac{\rho_0}{\alpha} \le \chi_i + \rho_i.$$

Therefore,

$$I\left(\chi_0, \frac{\rho_0}{\alpha}\right) \subset I(\chi_i, \rho_i).$$

Because i was arbitrary,

$$I\left(\chi_0, \frac{\rho_0}{\alpha}\right) \subset I_1.$$

Hence,

$$\mu\left(I_1\right) \geq \mu\left(I\left(\chi_0, \frac{\rho_0}{\alpha}\right)\right) = \frac{2\rho_0}{\alpha} = \frac{\mu\left(I_\alpha\right)}{\alpha}.$$

□

Lemma 8.3 is used in the proof of Theorem 8.5, below.

Lemma 8.4. *For $\kappa \geq 0$,*

$$\mathcal{E}_\kappa(w) = \frac{1}{2}\mu\left(\bigcap_{x \in X} I\left(f(x), \kappa d(x,w)\right)\right).$$

Proof.

$$\begin{aligned}
\mathcal{E}_\kappa(w) &= \frac{1}{2}\left[e_\kappa^+(w) - e_\kappa^-(w)\right] \\
&= \frac{1}{2}\left\{\min_{x \in X}\left\{f(x) + \kappa d(x,w)\right\} - \max_{x \in X}\left\{f(x) - \kappa d(x,w)\right\}\right\} \\
&= \frac{1}{2}\mu\left(\left[\max_{x \in X}\left\{f(x) - \kappa d(x,w)\right\}, \min_{x \in X}\left\{f(x) + \kappa d(x,w)\right\}\right]\right) \\
&= \frac{1}{2}\mu\left(\bigcap_{x \in X} I\left(f(x), \kappa d(x,w)\right)\right).
\end{aligned}$$

The first equality follows from proposition 8.1. □

Theorem 8.5. *For any $\lambda \in \Re^+$,* if $\mathcal{E}_{\hat{K}} \geq \lambda\hat{K}$, then $\mathcal{E}_K(\hat{f}) \geq \lambda K$.

Proof. Let $w^\star \equiv \arg\max_w \mathcal{E}_{\hat{K}}(w)$. Then

$$\begin{aligned}
\mathcal{E}_K(\hat{f}) &\geq \mathcal{E}_K(\hat{f}_K) \\
&= \mathcal{E}_K(w^\star) \\
&\geq \frac{K}{\hat{K}} \cdot \mathcal{E}_{\hat{K}}(w^\star) \qquad (8.5) \\
&\geq \frac{K}{\hat{K}} \cdot \lambda\hat{K} \qquad (8.6) \\
&= \lambda K.
\end{aligned}$$

Inequality (8.6) follows from (8.5) by hypothesis. Inequality (8.5) is a consequence of lemma 8.3: Let $\alpha = \hat{K}/K \leq 1$. For, $i = 1, \ldots, \#X$, let $\rho_i = f(x_i)$ and $\chi_i = Kd(x,w)$. Then, by lemma 8.4, $\mu(I_1)/2 = \mathcal{E}_K$ and $\mu(I_\alpha)/2 = \mathcal{E}_{\hat{K}}$. □

8.3.2 Maximum uncertainty for an emulator based on one observation

In this section we work in ℓ_∞: $d(v, w) = \|v - w\|_\infty$. This simplifies the calculations and gives a particularly strong result.

Let $z \equiv (1/2, \ldots, 1/2)$, the centroid of $[0,1]^p$, and let $Z \equiv \{z\}$. Let $\hat{g} \in \mathcal{F}_{\infty,Z}$ be the constant function $\hat{g}(w) \equiv f(z)$, $\forall w \in [0,1]^p$. The ℓ_∞ distance from z to any point on the boundary of $[0,1]^p$ is $1/2$, so

$$\mathcal{E}_{K,Z}(\hat{g}) = \frac{K}{2}.$$

That is, the maximum uncertainty of the emulator that is constant throughout $[0,1]^p$ and equal to the value of f at the centroid of the cube is $K/2$. Let $W \subset [0,1]^p$ be finite and $c \in \mathbb{R}$. Suppose f is constant on the set W and that W contains fewer than 2^p points. Let $\hat{h} \in \mathcal{F}_{\infty,W}$. By examining the corners of the domain, it follows that

$$\mathcal{E}_{K,W}(\hat{h}) \geq \frac{K}{2}.$$

Making 2^p observations of f is intractable for the Community Atmosphere Model and for many other applications. If f is nearly constant, the situation may still be hopeless.

How do we know whether $f|_X$ is too close to constant to benefit from observing it more than once, but fewer than 2^p times?

Corollary 8.6. *If* $\mathcal{E}_{\hat{K}} \geq \hat{K}/2$, *then*

$$\mathcal{E}_K(\hat{f}) \geq \frac{K}{2} \geq \mathcal{E}_{K,Z}(\hat{g}).$$

That is, if $\mathcal{E}_{\hat{K}} \geq \hat{K}/2$, no emulator based on observing $f|_X$ has smaller maximum uncertainty than the constant emulator based on a single observation—f is too nearly constant. Corollary 8.6 follows directly from theorem 8.5, taking $\lambda = \hat{K}/2$.

8.4 Applications

This section presents three examples of increasing complexity: two in which f is known analytically, and one in which f is *HEB* arising from a numerical model of climate. In this section the distance metric is $d(v, w) = \|v - w\|_\infty$, except where noted.

8.4.1 High-dimensional ℓ_∞ cone

Consider a emulating a function defined on the 21-dimensional hypercube $[0,1]^{21}$; $z \equiv (0.5, \ldots, 0.5)$ denotes the center of that hypercube. Suppose

$$f(x) \equiv \|x - z\|_\infty.$$

We observe f at z and, for $i = 1, \dots 21$, at both points satisfying $x_i \in \{0, 1\}$ and $x_j = 0.5$ for $j \neq i$. (This is a "one-at-a-time" sampling design, where one component at a time is shifted from a typical value to a more extreme value.) These 43 points constitute X. Then

$$\hat{K} = K = 1.$$

Because every point $w \in [0, 1]^{21}$ is within 0.5 of $x \in X$ satisfying $f(x) = 0.5$,

$$e^-_{\hat{K}} \geq 0.$$

Because every point $w \in [0, 1]^{21}$ is within 0.5 of z, and $f(z) = 0$,

$$e^+_{\hat{K}} \leq 0.5.$$

Hence, by corollary 8.2,

$$\mathcal{E}_{\hat{K}} \leq 0.25.$$

Had we only observed f at z but fixed $\hat{K} = 1$ (or observed f at another point in addition to z and computed $\hat{K}$ from those two points),

$$\mathcal{E}_{\hat{K}} = 0.5.$$

In this example, despite the high dimension of $\text{dom}(f)$, emulating f using a modest number of observations (43) has smaller maximum uncertainty than emulating f using just a single observation of f at z: a small number of observations may constrain a high-dimensional function globally. High-dimensional problems with small numbers of data do not necessarily have large uncertainties, as "the curse of dimensionality" would suggest. The dimension matters, but so does f itself.

To connect our results to a common emulation method, we fit a Gaussian process to $f|_X$ by maximum likelihood using the R package `mlegp` [117]. For 100,000 points selected uniformly at random from $[0, 1]^p$, the mean error is 0.02, 2% of K. The maximum error at these 100,000 points is 0.23, but the error at $(0.6, \dots, 0.6)$—which is not in the sample—is 0.38.[6] Because the error of $\hat{f}_{\hat{K}}$ is no greater than $\mathcal{E}_{\hat{K}} = 0.25$, for this f, the minimax emulator $\hat{f}_{\hat{K}}$ outperforms this Gaussian process emulator both in minimax uncertainty and in actual maximum error.

8.4.2 Borehole function

The commonly used test function

$$f_0(H_u, H_\ell, T_u, T_\ell, r, r_w, L, K_w) \equiv \frac{2\pi T_u \left(H_u - H_l\right)}{\log\left(r/r_w\right)\left(1 + \frac{2LT_u}{\log(r/r_w) r_w^2 K_w} + \frac{T_u}{T_\ell}\right)}$$

models water flow through a borehole [116]. Its input variables are described in table 8.2, which also lists the ranges of those variables. The output is water flow rate in cubic meters

Table 8.2: Borehole function domain

variable	range	description
H_u	$[990, 1110]$	potentiometric head of upper aquifer (m)
H_ℓ	$[700, 820]$	potentiometric head of lower aquifer (m)
T_u	$[63070, 115600]$	transmissivity of upper aquifer (m^2/yr)
T_ℓ	$[63.1, 116]$	transmissivity of lower aquifer (m^2/yr)
r	$[100, 50000]$	radius of influence (m)
r_w	$[0.05, 0.15]$	radius of borehole (m)
L	$[1120, 1680]$	length of borehole (m)
K_w	$[9855, 12045]$	hydraulic conductivity of borehole (m/yr)

per year. We rescale f_0 so that its inputs range over the 8-dimensional unit hypercube $[0,1]^8$; the resulting function is denoted f.

Now, by reasoning about the functional form of f, we seek a bound on its Lipschitz constant K. In ℓ_∞, because f is differentiable and $\mathrm{dom}(f)$ is convex,

$$K = \sup_{w \in \mathrm{dom}(f)} \|Df(w)\|_\infty = \sup_{w \in \mathrm{dom}(f)} \sum_{i=1}^{8} \left| \frac{\partial}{\partial w_i} f(w) \right|.$$

Let

$$H = 2\pi(H_u - H_\ell),$$
$$R = \log(r/r_w),$$
$$M = 2L/K_w,$$
$$t = T_\ell^{-1} + T_u^{-1}$$

and

$$S = M + R r_w^2 t.$$

Now

$$f_0(H_u, H_\ell, T_u, T_\ell, r, r_w, L, K_w) = \frac{H r_w^2}{S}.$$

[6]This point was found by searching the ray $c(1,\ldots,1)$; there might be points with even larger errors.

We bound each partial derivative of f using the ranges of the input variables:

$$\left|\frac{\partial f_0}{\partial H_\ell}\right| = \left|\frac{\partial f_0}{\partial H_u}\right| = \frac{2\pi r_w^2}{S} \leq 0.76 \implies \left|\frac{\partial f}{\partial H_\ell}\right| = \left|\frac{\partial f}{\partial H_u}\right| \leq 91.2$$
$$\left|\frac{\partial f_0}{\partial T_u}\right| = \frac{HRr_w^4}{S^2T_u^2} \leq 0.01 \implies \left|\frac{\partial f}{\partial T_u}\right| \leq 0.01$$
$$\left|\frac{\partial f_0}{\partial T_l}\right| = \frac{HRr_w^4}{S^2T_\ell^2} \leq 0.13 \implies \left|\frac{\partial f}{\partial T_l}\right| \leq 6.8$$
$$\left|\frac{\partial f_0}{\partial r}\right| = \frac{Hr_w^4t}{S^2r} \leq 0.01 \implies \left|\frac{\partial f}{\partial r}\right| \leq 290.8$$
$$\left|\frac{\partial f_0}{\partial r_w}\right| = \frac{Hr_w^3t}{S^2} + \frac{2H}{S(1/r_w + Rr_wt/M)} \leq 4050.2 \implies \left|\frac{\partial f}{\partial r_w}\right| \leq 405.0$$
$$\left|\frac{\partial f_0}{\partial L}\right| = \frac{2Hr_w^2}{S^2K_w} \leq 0.34 \implies \left|\frac{\partial f}{\partial L}\right| \leq 190.4$$
$$\left|\frac{\partial f_0}{\partial K_w}\right| = \frac{2LHr_w^2}{S^2K_w^2} \leq 0.06 \implies \left|\frac{\partial f}{\partial K_w}\right| \leq 123.7.$$

Summing these upper bounds for the partial derivatives of f yields

$$\sup_{w \in \mathrm{dom}(f)} \|Df(w)\|_\infty < 1200.$$

Moreover, for $w_0 = (1100, 700, 115547, 116, 100, 0.15, 1120, 12045)$,

$$\|Df(w_0)\|_\infty = 944.$$

Hence, for the rescaled borehole function f,

$$944 \leq K \leq 1200.$$

Of course, if f really were a black box, such reasoning would be impossible. We estimated $\hat{K}$ from 1000 sample points selected in two different ways:

1. Select 1000 points by Latin hypercube sampling. This yields $\hat{K} = 367$.
2. Select 100 points by Latin hypercube sampling. For each of these points, draw an additional 9 points a small distance (10^{-5}) from it in each coordinate, in a random direction. This yields $\hat{K} = 576$.

We fix $\hat{K} = 576$ for the remainder of this example; note that this is roughly half the true value of K.

Now let X contain the following 273 points: all $2^8 = 256$ corners of $[0,1]^8$, the center of the domain $(0.5, \ldots, 0.5)$, and, for $i = 1, \ldots, 8$, each of the two points satisfying $x_i \in \{0,1\}$

and $x_j = 0.5$ for $j \neq i$. (The empirical Lipschitz constant of f on this set is less than 576.) By branch-and-bound we find

$$\mathcal{E}_{576} < 207$$

which is less than 576/2. Hence, by corollary 8.2, the best emulator $\hat{f}_{576}$ based on $f|_X$ has lower maximum uncertainty than the best emulator based on $f|_{\{z\}}$ alone.

Holding X fixed, we now lower-bound M_ϵ, the minimum computational burden (Section 8.2). Convex programming finds $\bar{\gamma} = 134.7$. The union bound implies that the proportion of the domain where f could be constant is $\mu\left(\bar{Q}\right) \geq 0.76$. Then for $\epsilon = 100$ (about 20% of $\hat{K}$ or 10% of K), $M_{100} \geq 3598$ additional observations might be needed. But for $\epsilon = 10$, $M_{10} \geq 3.59 \times 10^{11}$ additional observations might be required.

For comparison, we emulate f by a Gaussian process, again estimating the parameters using the R package `mlegp` [117] from the same set X of 273 points. For 100,000 points selected at random uniformly from $[0,1]^8$, the mean error is 37.3, approximately 3% of K. The maximum error at these points is 207.9, approximately 20% of K.

8.4.3 Climate modeling

The Uncertainty Quantification Initiative at Lawrence Livermore National Laboratory[7] provided results from 1154 climate simulations using the Community Atmosphere Model (CAM) with $p = 21$ parameters. Each parameter was scaled so that the interval $[0,1]$ contained all values considered physically reasonable. The output of interest was a scalar, the simulated global average upwelling longwave flux (FLUT) averaged over the third through twelfth years of the simulation (a 10-year average after a 2-year burn-in). Each such average is deterministic: repeating a run with the same input parameters should produce the same output. The simulator amounts to a function f that maps $[0,1]^p \to \mathbb{R}$. Running the simulator was computationally expensive; each run took several days on a supercomputer. The Lawrence Livermore National Laboratory team used several approaches to choose the points $X \subset [0,1]^p$ at which to run simulations, including Latin hypercube, one-at-a-time, and random-walk multiple-one-at-a-time [105]. The 1154 simulations include all points selected by any of those approaches.

For these observations, we find $\bar{\gamma} = 232.77$, $\hat{K} = 14.20$ for $q = 2$, and $\hat{K} = 34.68$ for $q = \infty$.

8.4.3.1 Computational burden

By (8.2),

$$M_\epsilon \geq \left\lceil \epsilon^{-21} \left[\frac{1.57 \times 10^{24}}{0.0038} - 6.81 \times 10^{24} \right] \right\rceil > \epsilon^{-21} \times 10^{26}$$

[7]This dataset was provided by the Institutional Science and Technology Office at Lawrence Livermore National Laboratory under the Uncertainty Quantification Strategic Initiative Laboratory-Directed Research and Development Project 10-SI-013.

for $q = 2$. For example, if ϵ is 1% of $\hat{K}$, then $M_\epsilon \geq 10^{43}$. Even if ϵ is 50% of $\hat{K}$, $M_\epsilon > 10^8$. For $q = \infty$,

$$M_\epsilon \geq \left\lceil \epsilon^{-21} \left\lceil \frac{2.19 \times 10^{32}}{2^{21}} - 6.81 \times 10^{24} \right\rceil \right\rceil > \epsilon^{-21} \times 10^{25}.$$

These lower bounds on the minimum computational burden are extreme for a wide range of values of ϵ: there are functions that fit the 1154 observations and are as regular as the observations allow, but that cannot be approximated with useful uncertainty from any tractable number of observations. The function $\bar{f}$, which is simple to construct, attains these lower bounds on minimum computational burden. Note the contrast with the cone example, which was also 21-dimensional: the dimension of $\mathrm{dom}(f)$ does not by itself determine how hard it is to emulate f accurately.

8.4.3.2 Uncertainty

Is the maximum uncertainty of the best emulator based on observing f at the 1154 points in X lower than the maximum uncertainty of the constant emulator based on one observation of f at the centroid of $[0,1]^p$? We cannot simply compute these two maximum uncertainties, because K is unknown. But corollary 8.6 applies if we can determine whether $\mathcal{E}_{\hat{K}} \geq \hat{K}/2$. Unfortunately, determining $\mathcal{E}_{\hat{K}}$ is difficult. In ℓ_∞, if $f|_X$ is constant, finding $\mathcal{E}_{\hat{K}}$ amounts to finding a maximal empty hypercube, a problem recently shown to be NP-hard in p [118]. It is generally no easier if f varies on X. Fortunately, it suffices to bound $\mathcal{E}_{\hat{K}}$. By working in ℓ_∞, we can bound $\mathcal{E}_{\hat{K}}$ above and below by considering just the corners of $[0,1]^p$; we take $d(v,w) = \|v - w\|_\infty$ throughout this section.

Proposition 8.7. *Let* $\mathbf{0} \equiv (0,\ldots,0)$, $\mathbf{1} \equiv (1,\ldots,1)$, *and* $\tilde{d}(v) \equiv \max\left(d(v,\mathbf{0}), d(v,\mathbf{1})\right)$. *Then*

$$\mathcal{E}_{\hat{K}} \leq \frac{1}{2}\left\{\min_{x\in X}\left[f(x) + \hat{K}\tilde{d}(x)\right] - \max_{x\in X}\left[f(x) - \hat{K}\tilde{d}(x)\right]\right\}.$$

Proof. Fix $w \in [0,1]^p$. Let $w_{(i)}$ denote the ith component of w. Then

$$\begin{aligned}
d(v,w) &= \max_{i\in\{1,\ldots,p\}} \left|v_{(i)} - w_{(i)}\right| \\
&\leq \max_{i\in\{1,\ldots,p\}} \max_{\delta\in\{0,1\}} \left|v_{(i)} - \delta\right| \\
&= \max_{i\in\{1,\ldots,p\}} \max_{y\in\{\mathbf{0},\mathbf{1}\}} \left|v_{(i)} - y_{(i)}\right| \\
&= \max_{y\in\{\mathbf{0},\mathbf{1}\}} \max_{i\in\{1,\ldots,p\}} \left|v_{(i)} - y_{(i)}\right| \\
&= \max_{y\in\{\mathbf{0},\mathbf{1}\}} d(v,y) \\
&= \max(d(v,\mathbf{0}), d(v,\mathbf{1})).
\end{aligned}$$

Hence,

$$\begin{aligned}
\mathcal{E}_{\hat{K}}(w) =& \frac{1}{2}\mu\left(\bigcap_{x\in X} I\left(f(x), \hat{K}d(x,w)\right)\right) && (8.7)\\
\leq& \frac{1}{2}\mu\left(\bigcap_{x\in X} I\left(f(x), \hat{K}\tilde{d}(x)\right)\right) && (8.8)\\
=& \frac{1}{2}\left\{\min_{x\in X}\left[f(x)+\hat{K}\tilde{d}(x)\right] - \max_{x\in X}\left[f(x)-\hat{K}\tilde{d}(x)\right]\right\}
\end{aligned}$$

where (8.7) follows from lemma 8.4. Because the right-hand side of this inequality does not depend on w, the proposition follows by taking suprema. □

Using this proposition, we calculate $\mathcal{E}_{\hat{K}} \leq 20.95$ for the CAM dataset. On the other hand, the maximum over all $[0,1]^p$ is at least as large as the maximum over the corners of $[0,1]^p$:

$$\mathcal{E}_{\hat{K}} \geq \max\left\{\mathcal{E}_{\hat{K}}(w) : \forall w \in \{0,1\}^p\right\}.$$

Perhaps surprisingly, this lower bound is essentially sharp for the CAM dataset. The domain $[0,1]^p$ contains 2^p corners $\{r_i\}_{i=1}^{2^p}$. Divide $[0,1]^p$ into 2^p hypercubes $\{R_i\}_{i=1}^{2^p}$ with edge-length 1/2, disjoint interiors, each containing a different corner of $[0,1]^p$ (e.g., one such hypercube is $[0,1/2]^p$). Then the R_i are disjoint ℓ_∞-balls of radius 1/4. Because X contains only 1154 points, the vast majority of $\{R_i\}_{i=1}^{2^p}$ do not contain any element of X. Because $\mathcal{E}_{\hat{K}}(w)$ tends to increase with distance from points in X, these unoccupied hypercubes are good regions to look for points with large values of $\mathcal{E}_{\hat{K}}(w)$. Within an unoccupied hypercube R_i, no point is farther in ℓ_∞ from any point in X than the corner r_i. So, the corners $\{r_i\}_1^{2^p}$ are good places to observe $\mathcal{E}_{\hat{K}}(w)$ to find a tight lower bound on $\mathcal{E}_{\hat{K}}$.

For the CAM dataset, one corner r_j attains $\mathcal{E}_{\hat{K}}(r_j) = 20.95$. Since this is also the numerical upper bound, $\mathcal{E}_{\hat{K}} = 20.95$.

Because $\mathcal{E}_{\hat{K}} \geq \hat{K}/2 = 17.34$, theorem 8.5 says that $\mathcal{E}_K(\hat{f}) \geq K/2$ for any emulator $\hat{f}$. In other words, by the discussion in Section 8.3.2, our maximum uncertainty would have been no greater had we just observed f once, at z, and predicted $\hat{f}(w) = f(z)$ for all $w \in [0,1]^p$.

In some sense, this result is not surprising: if we had fixed $\hat{K}$ but replaced f with a constant function, and $\#X < 2^p$, then $\mathcal{E}_{\hat{K}} \geq \hat{K}/2$, with equality holding if and only if $z \in X$. By repeating the bounding procedures from the previous two sections with $\hat{K}/2 = 17.34$ fixed but f replaced with constant function c, we find $\mathcal{E}_{c,X,\hat{K}} = 26.95$. The increase in maximum uncertainty from 20.95 to 26.95 that results from replacing f with a constant shows that the observed variation in f reduces the maximum uncertainty considerably—although the maximum uncertainty remains quite large.

To connect these theoretical results to common emulation methods, we fit a Gaussian process model [117] and Multivariate Adaptive Regression Splines (MARS) [119] to the 110 CAM observations from a Latin hypercube design, leaving 1043 observations for testing. On the test set, the mean error of the Gaussian process model is 1.03 (3% of $\hat{K}$) and the maximum error is 6.73 (20% of $\hat{K}$). For MARS, the mean error on the test set is 1.59 and

Table 8.3: Confidence bounds for quantiles and the mean of the uncertainty of the minimax emulator $\hat{f}_{\hat{K}}$ for CAM

		95% lower confidence bound			
norm	units	lower quartile	median	upper quartile	average
Euclidean	$\hat{K}/2$	1.462	1.599	1.732	1.599
supremum	$\hat{K}/2$	0.648	0.716	0.781	0.715
Euclidean	$\hat{\gamma}$	0.044	0.049	0.053	0.049
supremum	$\hat{\gamma}$	0.048	0.053	0.058	0.053

Column 1: distance metric d used for the Lipschitz constant. Columns 3–5: binomial lower 95% confidence bounds for quartiles of the uncertainty, obtained by inverting binomial tests. Column 6: 95% lower 95% confidence bound for the integral of the uncertainty over the entire domain $[0,1]^p$, based on inverting z-tests. Columns 3–6 are expressed as a fraction of the quantity in column 2. Results are based on 10,000 uniform random samples from $[0,1]^p$.

maximum error is 6.21. Since the 1043 test points are all distant from many corners of $[0,1]^p$, the error of these methods over $[0,1]^p$ might be far larger; it would take many more evaluations of f to tell. Absent such data, there is no evidence that those methods have maximum error less than $\mathcal{E}_{\hat{K}} = 20.95$.

8.5 Extensions

8.5.1 Distribution of the uncertainty

By drawing independent points $W \sim \text{Uniform}([0,1]^p)$ and evaluating $\mathcal{E}_{\hat{K}}(W)$, we construct lower confidence bounds for quantiles of the uncertainty and the mean uncertainty over $[0,1]^p$. Table 8.3 shows the results for the CAM simulations based on 10,000 random samples from $[0,1]^p$. Even the lower quartiles are a large fraction of $\hat{K}$. For instance, at confidence level 95%, the uncertainty under the sup-norm metric exceeds 71.7% of $\hat{K}/2$ on at least 50% of the domain.

8.5.2 Uncertainty relative to typical values

We have focused on taking ϵ to be a fraction of K or $\hat{K}$. When ϵ is chosen that way, Section 8.2 and Section 8.3 establish conditions under which no emulator can be guaranteed to replicate the variation of f. Emulators are generally constructed to capture the complexity of the model: tracking its variability. That suggests approximating f to within a fraction of its variation, which is why we have calibrated ϵ to $\hat{K}$. If the goal were to approximate f to within a fraction of its mean, and its mean is large compared to its variation, approximating f globally by its sample mean might suffice. Then it might make sense to set ϵ to be a

Table 8.4: Minimum computational burden for the CAM model.

norm	$\epsilon/\hat{\gamma}$	lower bound on M_ϵ
Euclidean	0.02	3.6×10^{12}
	0.04	1,720,354
	0.06	345
	0.08	1
supremum	0.02	8.6×10^{10}
	0.04	413,595
	0.06	83
	0.08	1

fraction of a typical value of f, for instance, $\bar{\gamma}$ or the sample mean

$$\hat{\gamma} = \frac{1}{\#X} \sum_{x \in X} f(x).$$

The last 2 rows of Table 8.3 list confidence bounds for percentiles of the uncertainty as a fraction of $\hat{\gamma}$.

Similarly, for ϵ chosen suitably, inequality (8.2) gives a lower bound on M_ϵ for approximating f within a fraction of its typical value, rather than within a fraction of its observed variation. (Of course, the resulting bounds can be made arbitrarily small by adding a sufficiently large constant to f. One reason we think it is more interesting to calibrate ϵ as a fraction of K or $\hat{K}$ is that the results are invariant under affine transformations of f.)

For the CAM model, this lower bound on M_ϵ is trivial when ϵ is a large fraction of the typical value of f, but grows rapidly as the fraction decreases (table 8.4).

8.5.3 Other uses for e_κ^- and e_κ^+

We have primarily used e_κ^+ and e_κ^- to construct the minimax emulator and find its uncertainty. But if f is no less regular than it was observed to be, $e_{\hat{K}}^+$ is a pointwise upper bound on f and $e_{\hat{K}}^-$ is a pointwise lower bound on f. Moreover, if f is no less regular than the data require it to be, $\max_{w \in [0,1]^p} e_{\hat{K}}^+(w)$ is a global upper bound on f and $\min_{w \in [0,1]^p} e_{\hat{K}}^-(w)$ is a global lower bound on f.

Maximizing $e_{\hat{K}}^+$ or minimizing $e_{\hat{K}}^-$ exactly may not be tractable. For sup-norm, we can use the techniques from Section 8.4.3 to bound these extrema from above and below: for the CAM model, those upper and lower bounds on $e_{\hat{K}}^+$ are equal, as they are for $e_{\hat{K}}^-$. The maximum of $e_{\hat{K}}^+$ is 253.78 and the minimum of $e_{\hat{K}}^-$ is 211.88.

8.6 Conclusions

We find a lower bound on the minimum (over emulators) maximum (over functions that agree with the data and are as regular as the data allow) error of emulators of a function f based on n observations. This "mini-minimax" uncertainty is optimistic because it assumes that f has the smallest Lipschitz constant consistent with the data. The mini-minimax uncertainty is an attainable bound on the error of the best emulator of f at w: for any emulator $\hat{f}$, there is a function g that is at least as regular as f, that agrees with f at the n observations, and for which $|\hat{f}(w) - g(w)|$ is at least this mini-minimax value.

In some problems, *every* emulator based on any tractable number of observations of f has large maximum uncertainty (and the uncertainty is large over much of the domain), even if f is as regular as the data allow. That is, there are functions g and h that agree perfectly with the observations, are as regular as the observations permit, and yet differ by a large amount at some point in the domain of f.

We give sufficient conditions under which even the best possible emulator has large uncertainty. The conditions depend only on the observed values of f; they can be computed from the same observations used to train an emulator, at a cost that typically is small compared with the cost of generating those observations. The conditions are sufficient but not necessary, because f could be less regular than any finite set of observations reveals it to be. It is not possible to give necessary conditions that depend only on the observed values of f; a priori bounds on the regularity of f would be needed.

The conditions seem likely to hold for many high-consequence applications. Indeed, we show quantitatively that the conditions hold for a large climate-modeling dataset. When the maximum uncertainty in approximating f everywhere by a constant—the value of f at the center of the domain—is no larger than the maximum uncertainty in approximating f from any tractable number of observations, emulators may not be useful. No emulator can then reliably model f as a function of its input $w \in [0,1]^p$.

Common techniques for assessing the accuracy of emulators (e.g., posterior variance or performance on hold-out data) understate the true uncertainty, because they make strong assumptions about f that are based neither on the observations nor on known properties of f, or because they focus on average error rather than worst-case error. However, as Section 8.5 shows, even the average uncertainty and quartiles of the uncertainty for the CAM model are quite large.

The mini-minimax uncertainty is a one-sided tool: if this uncertainty is large, the data do not constrain f well, while if it is small, the data constrain f *only* if it is no less regular than the data collected so far show it must be. That said, if the mini-minimax uncertainty is uncomfortably large, there might be ways to reduce it. For instance, if the lower bound (8.2) on the computation burden required to reduce the uncertainty to a useful level ϵ is affordable, one might collect more data. Provided the new data do not increase $\hat{K}$ substantially, the mini-minimax uncertainty can be reduced at will. But when p is large, the lower bound is likely to be large, because it grows exponentially with p. If observing f requires a real-world experiment, new technology might be required to make a useful number of additional observations affordable. When observing f involves running a simulator, collecting enough additional data to reduce the uncertainty to a reassuring value might require not only recruiting additional computational resources but also reducing the computational cost of each simulation—substantially.

In some cases, clever strategies can reduce the cost of computing f, at least to some known degree of approximation, but that is not always so. Cost reductions of orders of magnitude might require reducing the complexity of f. Reducing the dimension p of the domain of f is especially helpful, because reducing p pays exponential dividends. But it requires scientific justification: In general, eliminating parameters from a model entails bias in the model with no *a priori* limit. It is hard to calibrate the tradeoff between fitting a model that is constrained by the data but is known or suspected to be overly simplistic—and therefore biased—and a model that has lower bias but cannot be estimated reliably from an affordable number of data. Subject-matter knowledge is key.

Without increasing the number of observations or revising the model, reducing the uncertainty of emulators requires either more information about f[8] or changing the measure of uncertainty—changing the scientific question. Finally, approximating f pointwise is not usually the ultimate scientific goal. More important questions about f might be answered more directly.[9] These tactics are application-specific: the underlying science dictates the conditions that actually hold for f and the questions about f that matter.

[8]Common additional conditions include the following: parameters have only low-order interactions; the second derivative has an upper bound; the third derivative has a limited number of knots; the integral of the squared derivative of the model is bounded [120]. There are problems in which conditions like these may reflect actual knowledge about f. However, such conditions tend to be difficult to verify: simulation is perhaps most valuable when the underlying equations are not amenable to mathematical analysis.

[9]For example, for global optimization—finding maxima or minima—a form of adaptive sampling known as multi-start methods yields good results [121].

Bibliography

[1] S. A. Cook. “An Overview of Computational Complexity.” In: *Commun. ACM* 26.6 (June 1983), pp. 400–408. ISSN: 0001-0782. DOI: `10.1145/358141.358144`. URL: `http://doi.acm.org/10.1145/358141.358144`.

[2] R. Berry and J. Burnell. *The Handbook of Astronomical Image Processing*. Willmann-Bell, 2005. ISBN: 9780943396828. URL: `https://books.google.com/books?id=OOfPPAAACAAJ`.

[3] N. Smalheiser and V. Torvik. “Author name disambiguation.” In: *Annual Review of Information Science and Technology* (2009). Ed. by B. Cronin and E.)

[4] T. J. Santner, B. J. Williams, and W. I. Notz. *The design and analysis of computer experiments*. Springer Science & Business Media, 2013.

[5] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York, 2013. ISBN: 9780387216065. URL: `https://books.google.com/books?id=yPfZBwAAQBAJ`.

[6] R. Lupton, J. Gunn, et al. “The SDSS imaging pipelines.” In: *arXiv preprint astro-ph/0101420* (2001).

[7] J Sacks et al. “Design and Analysis of Computer Experiments.” In: *Statistical Science* (1989).

[8] E. Ben-Ari and D. Steinberg. “Modeling data from computer experiments: An empirical comparison of Kriging with MARS and projection pursuit regression.” In: *Quality Engineering* (2007).

[9] R. Ghanem, A Doostan, and J Red-Horse. “A probabilistic construction of model validation.” In: *Computer Methods in Applied Mechanics and Engineering* (2008).

[10] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[11] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. “Variational inference: A review for statisticians.” In: *arXiv preprint arXiv:1601.00670* (2016).

[12] U. Von Luxburg. “A tutorial on spectral clustering.” In: *Statistics and computing* 17.4 (2007), pp. 395–416.

[13] F. Jordan and F Bach. “Learning spectral clustering.” In: *Adv. Neural Inf. Process. Syst* 16 (2004), pp. 305–312.

[14] C. Stoughton, R. H. Lupton, et al. “Sloan digital sky survey: early data release.” In: *The Astronomical Journal* 123.1 (2002), p. 485.

[15] SDSS. *Sky images observed by the SDSS telescope.* `http://classic.sdss.org/gallery/gal_data.html`. [Online; accessed January 30, 2015]. 2015.

[16] E. Bertin and S Arnouts. “SExtractor: Software for source extraction.” In: *Astronomy and Astrophysics Supplement Series* 117.2 (1996), pp. 393–404.

[17] L. Miller, C. Heymans, et al. “Bayesian galaxy shape measurement for weak lensing surveys III: Application to the Canada–France–Hawaii Telescope Lensing Survey.” In: *Monthly Notices of the Royal Astronomical Society* (2013).

[18] D. Hogg. *Theories of everything.* Slides of a talk given at the NIPS 2012 Cosmology Meets Machine Learning Workshop. 2012.

[19] Dark Energy Survey. `http://www.darkenergysurvey.org/`. [Online; accessed February 5, 2015]. 2015.

[20] Large Synoptic Survey Telescope Consortium. `http://www.lsst.org/lsst/about`. [Online; accessed October 7, 2014]. 2014.

[21] SDSS DR10. *Flux units: maggies and nanomaggies.* `https://www.sdss3.org/dr10/algorithms/magnitudes.php#nmgy`. [Online; accessed February 6, 2015]. 2015.

[22] N. Padmanabhan et al. “An improved photometric calibration of the Sloan Digital Sky Survey imaging data.” In: *The Astrophysical Journal* 674.2 (2008), p. 1217.

[23] D. van Leeuwen. *GaussianMixtures.jl: A Julia package for Gaussian Mixture Models.* `https://github.com/davidavdav/GaussianMixtures.jl`. [Online; accessed May 11, 2015]. 2015.

[24] M. Braun and J. McAuliffe. “Variational inference for large-scale models of discrete choice.” In: *Journal of the American Statistical Association* 108.504 (2010), pp. 1230–1242.

[25] C. Wang and D. M Blei. “Variational inference in nonconjugate models.” In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 1005–1031.

[26] R. H. Byrd et al. “A limited memory algorithm for bound constrained optimization.” In: *SIAM Journal on Scientific Computing* 16.5 (1995), pp. 1190–1208.

[27] D. O. North. “An analysis of the factors which determine signal/noise discrimination in pulsed-carrier systems.” In: *Proceedings of the IEEE* 51.7 (1963), pp. 1016–1027.

[28] R. H. Lupton. *SDSS Image Processing I: The Deblender.* Tech. rep. 2005.

[29] D. Lang and D. Hogg. *Tractor: Astronomical source detection, separation, and photometry.* `http://thetractor.org/`. [Online; accessed January 30, 2015]. 2015.

[30] R. Lupton, Z. Ivezic, et al. *SDSS Image Processing II: The Photo Pipelines.* Tech. rep. Princeton University, 2005.

[31] L. Miller, T. Kitching, et al. “Bayesian galaxy shape measurement for weak lensing surveys–I. Methodology and a fast-fitting algorithm.” In: *Monthly Notices of the Royal Astronomical Society* 382.1 (2007), pp. 315–324.

[32] L. Simard, C. Willmer, et al. “The deep groth strip survey. II. Hubble space telescope structural parameters of galaxies in the groth strip.” In: *The Astrophysical Journal Supplement Series* 142.1 (2002), p. 1.

[33] D. Lang and D. Hogg. *Tractor: Astronomical source detection, separation, and photometry.* `http://thetractor.org/`. [Online; accessed January 30, 2015]. 2015.

[34] J. Regier, A. Miller, J. McAuliffe, et al. “Celeste: Variational inference for a generative model of astronomical images.” In: *Proceedings of the 32nd International Conference on Machine Learning.* 2015.

[35] C. Y. Peng et al. “Detailed structural decomposition of galaxy images.” In: *The Astronomical Journal* 124.1 (2002), p. 266.

[36] C. Y. Peng et al. “Detailed decomposition of galaxy images. II. Beyond axisymmetric models.” In: *The Astronomical Journal* 139.6 (2010), p. 2097.

[37] M. Barden et al. “GALAPAGOS: From pixels to parameters.” In: *Monthly Notices of the Royal Astronomical Society* 422.1 (2012), pp. 449–468.

[38] B. Häußler et al. “MegaMorph—multiwavelength measurement of galaxy structure: complete Sérsic profile information from modern surveys.” In: *Monthly Notices of the Royal Astronomical Society* 430.1 (2013), pp. 330–369.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet classification with deep convolutional neural networks.” In: *Advances in neural information processing systems.* 2012, pp. 1097–1105.

[40] E. Bertin and S. Arnouts. “SExtractor: Software for source extraction.” In: *Astronomy and Astrophysics Supplement Series* 117.2 (1996), pp. 393–404.

[41] S. Dieleman, K. Willett, and J. Dambre. “Rotation-invariant convolutional neural networks for galaxy morphology prediction.” In: *Monthly Notices of the Royal Astronomical Society* 450.2 (2015), pp. 1441–1459.

[42] D. Kingma and M. Welling. “Auto-encoding variational bayes.” In: *arXiv preprint arXiv:1312.6114* (2013).

[43] D. J. Rezende, S. Mohamed, and D. Wierstra. “Stochastic backpropagation and approximate inference in deep generative models.” In: *arXiv preprint arXiv:1401.4082* (2014).

[44] M. Titsias and M. Lázaro-Gredilla. “Doubly stochastic variational bayes for nonconjugate inference.” In: *Proceedings of the 31st International Conference on Machine Learning.* 2014, pp. 1971–1979.

[45] J. C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2005. ISBN: 9780471441908. URL: `https://books.google.com/books?id=f66OIvvkKnAC`.

[46] C. Stoughton, R. Lupton, et al. "Sloan digital sky survey: early data release." In: *The Astronomical Journal* 123.1 (2002), p. 485.

[47] *Galaxy Zoo—The Galaxy Challenge*. `https://www.kaggle.com/c/galaxy-zoo-the-galaxy-challenge`. [Online; accessed October 1, 2015]. 2013.

[48] Y. Bengio et al. "Generalized denoising auto-encoders as generative models." In: *Advances in Neural Information Processing Systems*. 2013, pp. 899–907.

[49] S. Mohamed. *A Statistical View of Deep Learning*. `http://blog.shakirm.com/wp-content/uploads/2015/07/SVDL.pdf`. [Online; accessed October 1, 2015]. 2015.

[50] C. Zhang. `https://github.com/pluskid/Mocha.jl`. [Online; accessed October 5, 2015]. 2015.

[51] Y. Jia et al. "Caffe: Convolutional Architecture for Fast Feature Embedding." In: *arXiv preprint arXiv:1408.5093* (2014).

[52] D. Kingma and J. Ba. "Adam: A method for stochastic optimization." In: *arXiv preprint arXiv:1412.6980* (2014).

[53] L. Van der Maaten and G. Hinton. "Visualizing data using t-SNE." In: *Journal of Machine Learning Research* 9.2579-2605 (2008), p. 85.

[54] M. Bilenko and B. Kamath. "Adaptive blocking: Learning to scale up record linkage." In: *International Conference on Data Mining*. 2006. ISBN: 0-7695-2701-7.

[55] D. Yan, L. Huang, and M. I. Jordan. "Fast approximate spectral clustering." In: *Knowledge Discovery and Data Mining*. 2009. ISBN: 9781605584959.

[56] C. Fowlkes et al. "Spectral grouping using the Nyström method." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2004).

[57] F. Bach and M. I. Jordan. "Learning spectral clustering, with application to speech separation." In: *The Journal of Machine Learning Research* (2006).

[58] X. Yin and J. Han. "Object distinction: Distinguishing objects with identical names." In: *Data Engineering, 2007. ICDE 2007.* (2007), pp. 1242–1246.

[59] P.-N. Tan et al. *Introduction to data mining*. Pearson Education India, 2006.

[60] A. Culotta et al. "First-order probabilistic models for coreference resolution." In: *Proceedings of NAACL HLT*. 2007, pp. 81–88.

[61] H. Han et al. "Two supervised learning approaches for name disambiguation in author citations." In: *Proceedings of the 4th ACM/IEEE-CS joint conference on digital libraries* (2004), pp. 296–305.

[62] H. Han, W. Xu, and H. Zha. "A hierarchical naive Bayes mixture model for name disambiguation in author citations." In: *Proceedings of the 2005 ACM symposium on Applied computing* (2005), pp. 1065–1069.

[63] J. Huang, S. Ertekin, and C. L. Giles. "Efficient Name Disambiguation for Large-Scale Databases." In: *Knowledge Discovery in Databases: PKDD 2006* (2006), pp. 536–544.

[64] Y. Song et al. "Efficient topic-based unsupervised name disambiguation." In: *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries.* JCDL '10. New York, NY, USA: ACM, 2007, pp. 342–351. ISBN: 978-1-4503-0085-8.

[65] V. Torvik et al. "A probabilistic similarity metric for Medline records: a model for author name disambiguation." In: *Journal of the American Society for information science and technology* 56.2 (2005), pp. 140–158. ISSN: 1532-2882.

[66] I. Bhattacharya and L. Getoor. "A latent Dirichlet model for unsupervised entity resolution." In: *SIAM International Conference on Data Mining.* 2006.

[67] I. Bhattacharya and L. Getoor. "Collective entity resolution in relational data." In: *ACM Trans. Knowl. Discov. Data* 1.1 (2007), p. 5. ISSN: 1556-4681.

[68] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2005.

[69] C. A. D'Angelo, C. Giuffrida, and G. Abramo. "A heuristic approach to author name disambiguation in bibliometrics databases for large-scale research assessments." In: *Journal of the American Society for Information Science and Technology* 62.2 (2011), pp. 257–269. ISSN: 15322890.

[70] A. A. Ferreira et al. "Effective Self-Training Author Name Disambiguation in Scholarly Digital Libraries." In: *Proceedings of the 10th annual joint conference on Digital libraries* (2010), pp. 342–351.

[71] H. Han and H. Zha. "Name disambiguation in author citations using a K-way spectral clustering method." In: *Joint Conference on Digital Libraries* (2005).

[72] I. Kang et al. "On co-authorship for author disambiguation." In: *Information Processing & Management* 45.1 (2009), pp. 84–97. ISSN: 0306-4573.

[73] B. Malin. "Unsupervised name disambiguation via social network similarity." In: *Workshop on Link Analysis, Counterterrorism, and Security.* Vol. 1401. 2005, pp. 93–102.

[74] D. M. McRae-Spencer and N. R. Shadbolt. "Also by the same author: AKTiveAuthor, a citation graph approach to name disambiguation." In: *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries.* JCDL '06. New York, NY, USA: ACM, 2006, pp. 53–54. ISBN: 1-59593-354-9.

[75] B. On et al. “Comparative study of name disambiguation problem using a scalable blocking-based framework.” In: *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. New York, NY, USA: ACM, 2005, pp. 344–353. ISBN: 1-58113-876-8.

[76] L. Tang and J. Walsh. “Bibliometric fingerprints: name disambiguation based on approximate structure equivalence of cognitive maps.” In: *Scientometrics* 84.3 (2010), pp. 763–784. ISSN: 0138-9130.

[77] Y. F. Tan, M. Y. Kan, and D. Lee. “Search engine driven author disambiguation.” In: *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*. New York, NY, USA: ACM, 2006, pp. 314–315. ISBN: 1-59593-354-9.

[78] T. Velden, A. Haque, and C. Lagoze. “Resolving author name homonymy to improve resolution of structures in co-author networks.” In: *Joint Conference on Digital Libraries* (2011), p. 241.

[79] K.-H. Yang et al. “Author name disambiguation for citations using topic and web correlation.” In: *Research and Advanced Technology for Digital Libraries* (2008), pp. 185–196.

[80] A. Culotta et al. “Author disambiguation using error-driven machine learning with a ranking loss function.” In: *Sixth International Workshop on Information Integration on the Web (IIWeb), collocated with AAAI, 2007*. 2007.

[81] A. Culotta and A. McCallum. “Tractable learning and inference with high-order representations.” In: *ICML Workshop on Open Problems in Statistical Relational Learning*. Citeseer, 2006.

[82] L. Getoor et al. “Learning probabilistic models of link structure.” In: *JMLR* 3 (2003), p. 707.

[83] A. McCallum, K. Bellare, and F. Pereira. *A conditional random field for discriminatively-trained finite-state string edit distance*. Conference on Uncertainty in AI (UAI), 2005.

[84] H. Pasula et al. “Identity uncertainty and citation matching.” In: *NIPS*. 2002, pp. 1425–1432.

[85] S. Singh et al. “Large-scale cross-document coreference using distributed inference and hierarchical models.” In: *Association for Computational Linguistics: Human Language Technologies (ACL HLT)* (2011).

[86] A. Wellner and A. McCallum. “Conditional Models of Identity Uncertainty with Application to Noun Coreference.” In: *NIPS*. 2004.

[87] M. Wick et al. “An entity based model for coreference resolution.” In: *SIAM International Conference on Data Mining*. Citeseer, 2009, pp. 365–376.

[88] A. J. Lotka. “The frequency distribution of scientific productivity.” In: *Journal of Washington Academy Sciences* (1926).

[89] M. Wick et al. "SampleRank: Training Factor Graphs with Atomic Gradients." In: *ICML*. 2011.

[90] G. Hinton. "Training products of experts by minimizing contrastive divergence." In: *Neural Computation* 14.8 (2002), pp. 1771–1800.

[91] C. Sutton and A. McCallum. "An introduction to conditional random fields for relational learning." In: *Introduction to statistical relational learning* x (2006), pp. 95–130.

[92] C. Sutton and A. McCallum. "An Introduction to Conditional Random Fields." In: *Found. Trends Mach. Learn.* 4.4 (Apr. 2012), pp. 267–373. ISSN: 1935-8237. DOI: `10.1561/2200000013`. URL: `http://dx.doi.org/10.1561/2200000013`.

[93] J. Lafferty, A. McCallum, and F. Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data." In: *ICML*. Citeseer, 2001.

[94] M. Wick et al. "Samplerank: Learning preferences from atomic gradients." In: *Neural Information Processing Systems (NIPS), Workshop on Advances in Ranking*. Citeseer, 2009, pp. 1–5.

[95] S. Jain and R. M. Neal. "A Split-Merge Markov chain Monte Carlo Procedure for the Dirichlet Process Mixture Model." In: *Journal of Computational and Graphical Statistics* 13.1 (2004), pp. 158–182. ISSN: 1061-8600.

[96] A. McCallum, K. Nigam, and L. Ungar. "Efficient clustering of high-dimensional data sets with application to reference matching." In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2000, pp. 169–178. ISBN: 1581132336.

[97] A. Goder and V. Filkov. "Consensus clustering algorithms: Comparison and refinement." In: *Proceedings of ALENEX*. Citeseer, 2008, pp. 109–117.

[98] N. Guttmann-Beck and R. Hassin. "Approximation algorithms for minimum k-cut." In: *Algorithmica* 27.2 (2000), pp. 198–207.

[99] L. Hagen and A. B. Kahng. "New spectral methods for ratio cut partitioning and clustering." In: *IEEE Transactions on Computer-Aided Design* (1992).

[100] J. Shi and J. Malik. "Normalized cuts and image segmentation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2000).

[101] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996. ISBN: 9780801854149. URL: `https://books.google.com/books?id=mlOa7wPX6OYC`.

[102] J. Duchi, E. Hazan, and Y. Singer. "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of Machine Learning* (2010), pp. 1–23.

[103] P. Ranjan, D. Bingham, and G. Michailidis. "Sequential experiment design for contour estimation from complex computer codes." In: *Technometrics* 50.4 (2008), pp. 527–541.

[104] S Shan and G. Wang. "Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions." In: *Structural and Multidisciplinary Optimization* (2009).

[105] C Covey et al. *A new ensemble of perturbed-input-parameter simulations by the Community Atmosphere Model.* Tech. rep. Lawrence Livermore National Laboratory, 2011.

[106] D Aspenberg, J Jergeus, and L Nilsson. "Robust optimization of front members in a full frontal car impact." In: *Engineering Optimization* (2012).

[107] M Holena, D Linke, and U Rodemerck. "Generator approach to evolutionary optimization of catalysts and its integration with surrogate modeling." In: *Catalysis Today* (2011).

[108] J. Shorter, P. Ip, and H. Rabitz. "An efficient chemical kinetics solver using high dimensional model representation." In: *The Journal of Physical Chemistry A* (1999).

[109] A Srivastava et al. "A method for using legacy data for metamodel-based design of large-scale systems." In: *Structural and Multidisciplinary Optimization* (2004).

[110] P. Koch, T. Simpson, and J. Allen. "Statistical approximations for multidisciplinary design optimization: the problem of size." In: *Journal of Aircraft* (1999).

[111] A. Booker et al. "A rigorous framework for optimization of expensive functions by surrogates." In: *Optimization* (1999).

[112] R. Bates et al. "Experimental design and observation for large systems." In: *Journal of the Royal Statistical Society, Series B* (1996).

[113] E. Packel. "Do linear problems have linear optimal algorithms?" In: *SIAM Review* (1988).

[114] J Traub and H Woźniakowski. *A general theory of optimal algorithms.* 1980.

[115] J. Traub, G. Wasilkowski, and H Woźniakowski. *Information-based complexity.* 1988.

[116] S Surjanovic and D Bingham. *Virtual library of simulation experiments: test functions and datasets.* `http://www.sfu.ca/~ssurjano/emulat.html`. Online; accessed March 3, 2014.

[117] G. Dancik. *mlegp: Maximum likelihood estimates of Gaussian processes.* R package version 3.1.4. `http://cran.r-project.org/package=mlegp`. 2013.

[118] J Backer and J. Keil. "The mono- and bichromatic empty rectangle and square problems in all dimensions." In: *LATIN 2010: Theoretical Informatics.* 2010.

[119] T Hastie and R Tibshirani. *mda: Mixture and Flexible Discriminant Analysis.* R pacakage version 0.4.4. `http://cran.r-project.org/package=mda`. 2013.

[120] M Lamboni et al. "Derivative-based global sensitivity measures: general links with Sobol' indices and numerical tests." In: *arXiv preprint* (2012).

[121] F. Hickernell. "A simple multistart algorithm for global optimization." In: *OR Transactions* (1997).