

1 **Nonparametric Bayesian Methods**
2 **in Robotics**

3 PROEFSCHRIFT

4 ter verkrijging van de graad van doctor
5 aan de Universiteit Leiden,
6 op gezag van de rector magnificus,
7 prof. mr. C.J.J.M. Stolker,
8 volgens besluit van het College voor Promoties
9 te verdedigen op ...2018
10 in de aula van de Universiteit
11 klokke ...

12 door

13 **Anne Cornelis van Rossum,**
14 **geboren op 30 december 1980**
15 **te Dirksland**

16 Samenstelling van de promotiecommissie

17

18

Promotoren: Prof. dr. H.J. van den Herik,
Prof. dr. ir. H. X. Lin

Copromotor: Dr. J.L.A. Dubbeldam

Promotiecommissie: Prof. dr. A. Plaat,
Prof. dr. J.N. Kok,
Prof. dr. ir. A.W. Heemink (TU Delft) ,
... (Buitenlands),
Prof. dr. ir. B.J.A. Kröse
Prof. dr. C.M. Jonker (SIKS)



19 SIKS Dissertation Series No. ...

20 The research reported in the thesis has been carried out under the auspices of SIKS, the
21 Dutch Research School for Information and Knowledge Systems.

22

23 ISBN

24 Copyright © 2018, A.C. van Rossum

25 *All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or*
26 *transmitted, in any form or by any means, electronically, mechanically, photocopying, recording*
27 *or otherwise, without prior permission of the author.*

“

28

The study of mental objects with reproducible properties is called mathematics.

29

”

30

The Mathematical Experience (Davis and Hersch, 1981)

“

31

The study of physical objects with reproducible properties is called science.

32

”

33

The dawning of the age of stochasticity, Mathematics: frontiers and perspectives

34

(Mumford, 2000)

CONTENTS

36	1 Introduction	1
37	1.1 Scope of the Thesis	1
38	1.2 Bayesian Nonparametrics	2
39	1.3 Problem Statement and Research Questions	2
40	1.4 Research Methodology	3
41	1.5 Main Contribution	4
42	1.6 Organization of the Thesis	5
43	2 Related Work	7
44	2.1 Probability Theory	8
45	2.1.1 Measure Theory	8
46	2.1.2 Bayesian Inference	16
47	2.1.3 Model Composition	21
48	2.1.4 General Random Elements	23
49	2.1.5 Plate Notation	23
50	2.1.6 Completely Random Measure and Lévy Measure	25
51	2.1.7 Exchangeability	26
52	2.1.8 Stick-breaking Representation	26
53	2.2 Five Random Processes	28
54	2.2.1 Dirichlet Process	28
55	2.2.2 Beta Process	31
56	2.2.3 Gamma Process	33
57	2.2.4 Pitman-Yor Process	36
58	2.2.5 Hierarchical Dirichlet Process	36
59	2.2.6 Comparison of Random Processes	37
60	2.3 Inference	38
61	2.3.1 Inverse Transform Sampling	38
62	2.3.2 Rejection Sampling	39
63	2.3.3 Approximate Bayesian Computation	40
64	2.3.4 Gibbs Sampling	41
65	2.3.5 Metropolis-Hastings Sampling	43
66	2.3.6 Split-Merge MCMC Sampling	43
67	2.3.7 Comparison of the Six Inference Methods	45
68	2.4 Chapter Conclusions	45

69	3 Nonparametric Bayesian Line Detection	47
70	3.1 Infinite Line Model	48
71	3.1.1 Posterior Predictive for a Line given Other Lines	49
72	3.1.2 Likelihood of Data given a Line	50
73	3.1.3 Conjugate Prior for a Line	51
74	3.1.4 Posterior Predictive for a Line given Data	52
75	3.2 Inference for the Infinite Line Model	53
76	3.3 Accelerating Inference for the Infinite Line Model	54
77	3.4 Performance of the Infinite Line Model	56
78	3.4.1 Clustering Performance	56
79	3.4.2 Two Examples	58
80	3.5 Chapter Conclusions	59
81	4 Nonparametric Bayesian Segment Estimation	61
82	4.1 Pareto Pairs	62
83	4.1.1 Pareto Prior	62
84	4.1.2 Posterior for a Pareto pair	64
85	4.2 Generative Process to Create a Line Segment	66
86	4.2.1 Dirichlet Process Prior	66
87	4.2.2 Likelihood of Data given Segment Parameters	66
88	4.3 Inference over a Line Segment	68
89	4.4 Results	69
90	4.5 Chapter Conclusions	70
91	5 Triadic Split-Merge Sampler	71
92	5.1 The Class of Split-Merge Samplers	71
93	5.2 Conventional Split-Merge Sampler	72
94	5.2.1 Acceptance for the Split Step	73
95	5.2.2 Acceptance for the Merge Step	74
96	5.2.3 Sequentially-Allocated Merge-Split Sampler	75
97	5.3 Triadic split-merge sampler	75
98	5.3.1 Acceptance for the Split Step	76
99	5.3.2 Acceptance for the Merge Step	78
100	5.4 Results	79
101	5.4.1 Implementation	80
102	5.4.2 Comparison	81
103	5.5 Chapter Conclusions	82
104	6 Adversarially Trained MCMC Kernels	83
105	6.1 Data-Driven Inference	83
106	6.2 Learning the Transition Operator	84
107	6.2.1 Adversarial Training	84
108	6.2.2 Variational Autoencoders	85
109	6.2.3 Infusion Training	85
110	6.3 Volumetric Models	86
111	7 Recommender Engine	89
112	7.1 Application	89

113	7.2 Model of Individuals	90
114	7.2.1 Multi-modal Normal-Uniform Distribution Model	90
115	7.2.2 Multi-modal Von-Mises-Uniform Distribution Model	91
116	7.2.3 Hyperparameters	92
117	7.3 Model of Groups	93
118	7.4 Inference	93
119	7.5 Results	94
120	7.5.1 Artificial Dataset	94
121	7.5.2 Real-world Dataset	95
122	7.6 Discussion	96
123	8 Discussion and Conclusions	97
124	References	99
125	Appendices	
126	A Probabilistic Concepts	107
127	A.1 Common Inequalities	108
128	A.1.1 Markov's Inequality	108
129	A.1.2 Chebyshev's Inequality	109
130	A.1.3 Weak Law of Large Numbers	109
131	A.1.4 Strong Law of Large Numbers	110
132	A.1.5 Common Distributions	110
133	B Dirichlet-Multinomial Interpretations	113
134	B.1 Dirichlet-Categorical	114
135	B.2 Dirichlet-Multinomial Distribution	115
136	B.3 Dirichlet-N Categorical Distributions	116
137	C Gibbs Sampling	119
138	C.1 Gibbs Sampling of Parameters	121
139	D Glossary	125
140	List of Figures	127
141	List of Tables	131
142	Summary	134
143	Samenvatting	136
144	Acknowledgments	139
145	Curriculum Vitae	140
146	Publications	142

147	SIKS Dissertation Series	144
-----	---------------------------------	------------

INTRODUCTION

Contents

The thesis addresses nonparametric Bayesian methods in robotic vision. Nonparametric Bayesian models can be simultaneously employed to perform inference over the number of entities observed and over the shape or nature of these entities. This chapter introduces nonparametric Bayesian models, the research methodology based on the Bayesian methodology, the main contribution towards robotic vision, and the general organization of the thesis.

Outline

The scope of thesis is to apply nonparametric Bayesian methods to robotic vision (Section 1.1). Bayesian nonparametric models define entities together with noise in such a way that inference can be performed in an optimal manner (Section 1.2). Particular problems in robotic vision that can benefit from Bayesian nonparametric methods are formulated and detailed (Section 1.3). The research methodology is described (Section 1.4). Our main contribution is to introduce nonparametric Bayesian models in robotic vision (Section 1.5). At the end of this chapter the organization of the thesis is given (Section 1.6).

1.1 Scope of the Thesis

In the thesis, modern Bayesian nonparametric methods are used to answer long-standing questions within computer vision and robotics. The following three challenging questions are typical examples. Is there a Bayesian form of line detection rather than applying the traditional Hough transform? Which of the nonparametric Bayesian priors can be used to detect multiple features simultaneously? What are efficient inference methods for these priors?

The scope of the thesis is the transfer of knowledge on Bayesian nonparametrics to well-described application domains. It will not establish a new body of work around a new family

of stochastic processes. The detailed application of complex models towards robotic vision is expected to help and encourage people in entirely different application domains, such as collaborative filtering, search engine optimization, and audio processing. All these different applications do not always need dedicated algorithms, but do deserve and can exploit the same optimal general inference techniques from Bayesian nonparametrics.

1.2 Bayesian Nonparametrics

In robotic vision (computer vision and depth perception) traditionally custom-made algorithms have been developed for a given task. There are specific methods to detect corners (e.g., Förstner and Gülch, 1987; Harris and Stephens, 1988; Shi and Tomasi, 1994), to detect edges (e.g., Sobel, 1970; Canny, 1986), to detect features (e.g., Hough, 1962), and to describe features (e.g., Lowe, 1999; Dalal and Triggs, 2005; Bay et al., 2006).

On the one hand, it is desirable that such sophisticated methods are generalizable to other application domains. On the other hand, it is important to take particular information about an application domain into account. The methods described in the previous paragraph are limited to their specific task. An example of limited generalizability can be found in the Hough transform. The Hough transform can be used to detect lines, but the way inference is performed in the algorithm does limit its application to basic forms of object detection. An example of limited specificity can be found in linear regression. Linear regression does not take into account real-world statistics.

Both generalization and specificity are formalized by a Bayesian model. A Bayesian model is general because it can be solved with general inference methods. One of such general inference methods is a Markov-Chain Monte Carlo method. It does not know anything about real-world statistics. A Bayesian model is also specific in that it can incorporate application-specific know-how by the definition of priors.

Typical problems in robotic vision will be about the recognition of several objects, multiple shapes, or objects that have multiple parts. Models that represent such objects do not have knowledge about the number of such objects, shapes, or parts. To incorporate application-specific know-how on the number of objects it is possible to define a prior that assigns a probability to this quantity. The number of objects can even be potentially infinite. The Bayesian models that define a prior on the number of objects, shapes, or parts are called nonparametric Bayesian models. This means that in contrast with conventional methods such as k -means clustering (Forgy, 1965; Lloyd, 1982) the number of objects does not need to be predefined.

1.3 Problem Statement and Research Questions

Many methods in robotics - and in particular in robotic vision - have been developed in times where computational resources were limited. Then, highly optimized algorithms have been developed, leveraging peculiarities of the application domain. Recent advances in Bayesian

212 methods, both with respect to concept development, as well as computational efficient so-
 213 lution strategies, now open up new ways to solve old problems. However, extending only
 214 the old methods themselves would lead to ad-hoc solution strategies that will miss benefits
 215 from potential optimal and more widely applicable algorithms.

216 This observation leads us to the formulation of our problem statement (PS).

217 **PS:** *How can robotic problems effectively be generalized and their structure
 exploited in a wider Bayesian framework?*

218 The problem statement is rather general. In our research, we focus on robotic vision, in the
 219 form of point cloud recognition and depth perception. In particular, we look at objects, lines,
 220 line segments, and more complex shapes.

221 The problem statement is divided into three research questions (RQs).

RQ 1 How can we estimate the number of objects simultaneously with the
 fitting of these objects?

222 **RQ 2** How can we estimate the number of lines simultaneously with line fitting
 in computer vision?

RQ 3 How can we recognize more general 3D objects?

223 1.4 Research Methodology

224 The research methodology advocated in the thesis follows the Bayesian methodology (cf.
 225 Savage, 1972; Jaynes, 2003). So, our research methodology exists out of two phases. In the
 226 first phase a Bayesian model is defined. This model exists of (1) a definition of parameters
 227 and relations between these parameters, (2) a definition of the noise, and (3) the data. In
 228 the second phase, the Bayesian method dictates all remaining unknowns, from the number
 229 of parameters to the values of the parameters. To perform Bayesian inference efficiently new
 230 methods are required if the model is complex (as is in the case of robotic vision).

231 The Bayesian methodology aims to establish the rationale for practical questions. The fol-
 232 lowing two questions are clear examples.

- 233 ◦ If we observe a single point in an image, can we expect it to be part of a line?
- 234 ◦ If we have two lines and we live in a world with squares, what are we able to infer?

235 The two questions tap into our capabilities to define models that makes our prior knowledge
 236 explicit. Moreover, if we are able to quickly assign (1) points to segments, (2) segments to
 237 lines, (3) objects to categories, we can enrich it with all corresponding group properties
 238 without the need to have them observed for this individual.

In robotic vision we take as an example the task of line detection. Both the Hough transform (Hough, 1962) and the RANSAC method (Bolles and Fischler, 1981) do detect lines, but they do not explicitly take noise into account. By applying Bayesian methodology to these tasks, the inference method becomes optimal in an information-theoretic sense. Also frequentist statisticians agree that nonparametric Bayesian models are consistent in the sense that they approach the underlying true distribution (Wasserman, 1998). There is no need to search for another method to infer lines in a line detection task. If someone would find a method that outperforms a Bayesian method it is either (1) because the signal or noise has not been correctly modeled after all, or (2) because the method overfits with respect to the available data. If approximations are used with respect to optimal Bayesian inference (either variational approximations or Markov-Chain Monte Carlo), there are theoretical guarantees on convergence.

A well known problem with nonparametric Bayesian models is the curse of dimensionality. Compared to maximum likelihood methods or other non-probabilistic methods that do not take noise into account at all, the nonparametric Bayesian models require significant computational resources. Our research methodology first establishes the correct models, even if solving them seems computationally infeasible. Our approach is to develop subsequently approximations using more sophisticated samplers, so that the theoretical guarantees on convergence are preserved.

Due to the fact that the models are optimal by construction, there are no experiments required to address the optimality in particular. However, experiments are still required to establish whether the models make sense. Yet, the methodology does also have limitations. For instance, we will not search over different noise models and limit priors to a particular hierarchical level.

1.5 Main Contribution

Our contribution to robotic vision can be subdivided into three parts that correspond with the three research questions.

The first part addresses the problem of inference about objects from a nonparametric Bayesian perspective. Contemporary methods in robotic vision do not allow for astute statements about their performance. In practice, this means that when using computer vision to detect cells under a microscope, someone cannot be confident about the number of detected cells. An autonomous cleaning robot in a supermarket cannot be confident about the aisle it is driving into. To be able to properly take into account models and uncertainty simultaneously, Bayesian models have found mainstream adoption. State-of-the-art Bayesian methods that reason about the number of objects alongside object models are a recent object of study (cf. Ferguson, 1973; Hjort, 1990; Lijoi and Prünster, 2010; Joho et al., 2011). The thesis applies such nonparametric Bayesian models towards the applications of robotic vision and depth perception. Models such as the infinite line model and the infinite line segment model are introduced.

278 The second part addresses the problem of high-dimensional data. To efficiently sample
279 more complex geometric structures, new MCMC (Markov-Chain Monte Carlo, Section 2.3.4)
280 methods are required. The thesis introduces such an MCMC sampler, namely a new Split-
281 Merge sampler, and applies it to complex geometric structures.

282 The third part addresses more complex robotic vision problems, in the form of object recog-
283 nition of point clouds in 3D. It combines nonparametric Bayesian inference with models
284 from deep learning.

285 1.6 Organization of the Thesis

286 **Chapter 1** (this chapter) introduces the problem of contemporary methods in computer
287 vision and depth perception. Due to the fact that these methods are not optimal
288 by construction, it is hard to articulate how they perform. The need for a Bayesian
289 methodology is sketched briefly. The problem statement and the research questions
290 are formulated. Moreover, the research methodology is described and the organization
291 of the thesis is outlined.

292 **Chapter 2** describes (1) probability theory using measure theory, (2) random measures
293 known as random processes of which five are described as nonparametric Bayesian
294 models, and (3) six inference methods that infer model parameters of such nonpara-
295 metric Bayesian models given the data. It is followed by a discussion that indicates
296 which parts will be most useful for chapters 3 and 4.

297 **Chapter 3** examines a first nonparametric Bayesian model, i.e., the infinite line model. The
298 infinite line model represents a countably infinite set of lines. Gibbs sampling is used
299 to perform simultaneous inference over (1) the number of lines and (2) line parameter
300 values such as slope and intercept.

301 **Chapter 4** examines a second nonparametric Bayesian model, i.e., the infinite line segment
302 model. The infinite line segment model represents a countably infinite set of line
303 segments. A split-merge MCMC sampling method is used to perform simultaneous
304 inference over (1) the number of line segments and (2) line segment parameter values
305 such as slope, intercept, and segment size. Chapters 2 to 4 answer the first research
306 question.

307 **Chapter 5** investigates a new MCMC method, the Triadic Split-Merge sampler. It is tailored
308 to clustering problems and accelerates inference of the models in Chapters 3 and 4.
309 This chapter answers the second research question.

310 **Chapter 6** examines a third nonparametric Bayesian model, particularly aimed at volumet-
311 ric inference. This chapter answers the third research question.

312 **Chapter 7** applies the hierarchical sampling method to the domain of recommender en-
313 gines. It estimates simultaneously the number of user types with a fitting procedure
314 for the individual user. I want to change this chapter to something relevant to point
315 clouds.

316 **Chapter 8** discusses the relevance of the developed models and inference methods. The
317 answers to the research questions are discussed. Then the problem statement is an-
318 swered and conclusions are formulated. Finally, recommendations are given and fu-
319 ture research is envisaged.

RELATED WORK

320

321

Contents

323

324

325

326

327

328

329

330

331

In robotics depth sensors generate point clouds. The tasks of robotic object recognition, positioning, and navigation require models that represent such point clouds. It is unclear whether the current methods that perform inference over point clouds are appropriate for these tasks. The current models do not model uncertainty explicitly. This chapter presents models that can be used for point cloud modeling and that represent uncertainty. This (partially) answers research question RQ1. The chapter concludes with recommendations for the development of point cloud inference models. They will be implemented in a new model for line inference in Chapter 3 and line segment inference in Chapter 4.

Outline

333

334

335

336

337

338

339

340

341

342

343

344

This chapter describes probability theory, and in particular, measure theory underlying random processes (Section 2.1). Five random processes are described, the Beta process, the Gamma process, the Dirichlet process, the Pitman-Yor process, and the hierarchical Dirichlet process. The random processes are presented as a Poisson process with a Lévy measure, a stick-breaking construction, and a sequential presentation (Section 2.2). These representations give rise to different inference methods. Six inference methods are described: Inverse transform sampling, rejection sampling, approximate Bayesian computation, Gibbs sampling, Metropolis-Hastings, and Split-Merge Markov chain Monte Carlo (Section 2.3). Inference about point clouds in the chapters to follow will use adaptations of the described models and inference methods for which some recommendations are given (Section 2.4).

2.1 Probability Theory

Modern probability is based on measure theory (Section 2.1.1). Measure theory will provide the means to formally describe random variables, random processes, and most generally, random measures. A model represented by random measures can be fitted to the data using Bayesian inference (Section 2.1.2). We give three typical examples of Bayesian model compositions, among which an infinite mixture model (Section 2.1.3). A number of processes are described that can be used with (for example as prior distribution) infinite mixture models (Section 2.1.4). We introduce plate notation which visualizes infinite models particularly well (Section 2.1.5). We investigate completely random measures and Lévy measures (Section 2.1.6), exchangeability (Section 2.1.7), and stick-breaking processes (Section 2.1.8), which will form the basis of the processes defined in Section 2.2.

2.1.1 Measure Theory

A random variable is a *function* that assigns values to a *set* of possible outcomes. The formal definition requires concepts such as “measurable function” and “probability space” from *measure theory* (Feller, 1950). Measure theory is used to generalize the notion of a random variable to that of a “random process”.

Informally, a measure generalizes the notion of size of Euclidean objects to sets and subsets. The definition of a measure is based on the definition of a σ -algebra. A σ -algebra ascribes a value to a sum of individual disjoint sets, even if they are infinite in number.

▼ Definition 2.1 — σ -algebra

A σ -algebra is a subset $\Sigma \in 2^X$, with X a set and 2^X its powerset, with three requirements:

- Σ is non-empty: at least one $A \in X$ is in Σ ;
 - Σ is closed under complementation: if A in Σ , so is its complement A^c ;
 - Σ is closed under countable unions: if A_1, A_2, \dots in Σ , so is $A = A_1 \cup A_2 \cup \dots$
-

The members of a σ -algebra are called *measurable sets*. Let $X = \{1, 2, 3, 4\}$ and let us define a σ -algebra $\Sigma = \{\emptyset, \{1\}, \{4\}, \{2, 3\}, \{1, 4\}, \{1, 2, 3\}, \{2, 3, 4\}, \{1, 2, 3, 4\}\}$. Here \emptyset denotes the empty set. The complement of A is defined with respect to X : $A \cup A^c = X$. An example of closure under complementation: let $A_1 = \{1\}$, then $A_1^c = \{2, 3, 4\}$ and A_1^c is indeed a member of Σ : $A_1^c \in \Sigma$. An example of closure under countable unions: let $A_1 = \{1\}$ and $A_2 = \{2, 3\}$, then $A_1 \cup A_2 = \{1, 2, 3\}$ and $A_1 \cup A_2 \in \Sigma$.

▼ Definition 2.2 — *generated* σ -algebra

A **generated σ -algebra**, with X a set and $B \in 2^X$, is the smallest σ -algebra $\sigma(B)$ that contains all sets of B .

Let $X = \{1, 2, 3, 4\}$ and $B = \{\{1\}, \{2, 3\}\}$, then the generated σ -algebra is the set $\sigma(B) = \{\emptyset, \{1\}, \{2, 3\}, \{1, 4\}, \{2, 3, 4\}, \{1, 2, 3, 4\}\}$. Here the sets in B are completed to the sets in $\sigma(B)$ by obeying the requirements of a σ -algebra of closure under complementation and countable unions by addition (e.g., $\{1, 4\}$ is added due to closure under completion with respect to $\{2, 3\}$).

The notion of a σ -algebra (Fremlin, 2000) can be applied to solve the so-called Banach-Tarski paradox (Banach and Tarski, 1924). This paradox describes how a unit-ball in \mathbf{R}^3 can be partitioned into a finite number of disjoint infinite sets (scattering of points) and then can be reassembled into two unit-balls again. This violates the intuitive notion of preservation of volume. If the measure μ of the union of two disjoint sets is equal to the sum of the measures of the two sets, this is called *finite additivity*: $\mu(\bigcup_{i=1}^N A_i) = \sum_{i=1}^N \mu(A_i)$. In probability theory σ -*additivity* extends this to infinite disjoint sets: $\mu(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mu(A_i)$. Measure theory solves the Banach-Tarski paradox by only assigning a measure to subsets that are measurable sets (Tao, 2011).

A *measure* assigns values to measurable sets (as stated before, measurable sets are members or subsets of Σ).

▼ Definition 2.3 — *measure*

A **measure** μ is a function from Σ to $[-\infty, +\infty]$, with three requirements:

- μ is non-negative: $\mu(A) \geq 0$ for $\forall A \in \Sigma$;
- μ has a null empty set: $\mu(\emptyset) = 0$;
- μ is σ -additive: $\mu(\bigcup_{i \in I_{\Sigma}} A_i) = \sum_{i \in I_{\Sigma}} \mu(A_i)$ for A_i disjoint.

The first statement defines that a measure μ only assigns non-negative values to sets in Σ . The second statement equals the measure of the empty set \emptyset to 0. The third statement defines that σ -additivity is required. For any two sets in Σ the measure of the union of the sets equals the sum of the measures of the individual sets. Here I_{Σ} defines an index over sets in Σ .

Informally, a measure relates the concepts of *sets* and *subsets* to notions of size. A measure can be seen as a *monotonically* increasing function. Let the set A in X be the interval $[0, 1]$, an uncountable (infinite) set of real numbers. Define the σ -algebra $\{\emptyset, A\}$. The empty set has measure 0, the set A has measure 1. Let us define the σ -algebra $\{\emptyset, A_{0,0.5}, A_{0.5,1}, A\}$. The set $A_{0,0.5}$ corresponds to the interval $[0, 0.5)$ and $A_{0.5,1}$ to $[0.5, 1]$. Both sets are assigned measure 0.5 and their union has measure 1. This examples shows that with σ -additive unions, measures can be assigned to sets that are uncountable.

A *measurable space* (X, Σ) is defined as a pair.

▼ Definition 2.4 — measurable space

A **measurable space** (X, Σ) is a pair with:

- X a set;
 - Σ a σ -algebra over X .
-

A **measure space** (X, Σ, μ) is defined as a triple.

▼ Definition 2.5 — measure space

A **measure space** (X, Σ, μ) is a triple with:

- X a set;
 - Σ a σ -algebra over X ;
 - μ a measure from Σ to $[-\infty, \infty]$.
-

A finite measure μ assigns a finite real number to all A .

▼ Definition 2.6 — finite measure

A **finite measure** μ is a measure from Σ to $[0, \infty)$:

- μ is non-negative: $\mu(A) \geq 0$ for $\forall A \in \Sigma$;
 - μ has a null empty set: $\mu(\emptyset) = 0$;
 - μ is σ -additive: $\mu(\bigcup_{i \in I_\Sigma} A_i) = \sum_{i \in I_\Sigma} \mu(A_i)$ for A_i disjoint;
 - μ for the whole sample space, X , is finite: $\mu(X) = N$.
-

A σ -finite measure allows A to be a countable union of sets with finite measure.

▼ Definition 2.7 — σ -finite measure

A **σ -finite measure** μ is a finite measure with:

- X is a countable union of sets with finite measures.
-

We will now define five measures: (A) the *probability measure* (Definition 2.8), (B) the *counting measure* (Definition 2.10), (C) the *Borel measure* (Definition 2.12), (D) the *Lebesgue measure* (Definition 2.17), and (E) the *random measure* (Definition 2.18). These measures are important because they are fundamental to different branches of mathematics. In probability theory a σ -algebra is interpreted as a collection of events to which probabilities are assigned. Counting measures play a fundamental role in discrete probability distributions. In integration theory a σ -algebra corresponding to the Borel and Lebesgue measures are relevant for integration in the Euclidean space \mathcal{R}^n . In statistics a σ -algebra formally defines

the concept of sufficient statistics and generalizes random variables to random functions and measures.

A: Probability measure

A probability measure, \mathbb{P} , is a finite measure that assigns non-negative values \mathbb{P} , called probabilities, to sets A , called events (see Definition 2.8).

▼ Definition 2.8 — probability measure

A probability measure \mathbb{P} is a measure μ with:

- \mathbb{P} is non-negative: $\mathbb{P}(A) \geq 0$ for $\forall A \in \Sigma$;
- \mathbb{P} has a null empty set: $\mathbb{P}(\emptyset) = 0$;
- \mathbb{P} is σ -additive: $\mathbb{P}(\bigcup_{i \in I_\Sigma} A_i) = \sum_{i \in I_\Sigma} \mu(A_i)$ for A_i disjoint;
- \mathbb{P} for the whole sample space, X , is unity: $\mathbb{P}(X) = 1$.

The four requirements are called the Kolmogorov axioms (Kolmogorov, 1933). The probability measure is an actual *measure*. It therefore obeys the three requirements: (1) non-negativity for any set, (2) the existence of a null empty set, and (3) σ -additivity. Here we note that a *probability* measure compared to a general measure obeys a fourth requirement, namely the restriction of the measure for the whole space X to 1. This can be seen as some kind of normalization. It influences how two probability measures have to be summed to become again a probability measure.

In Figure 2.1 the probability measure is visualized as a mapping from the probability space to the unit interval $[0, 1]$.

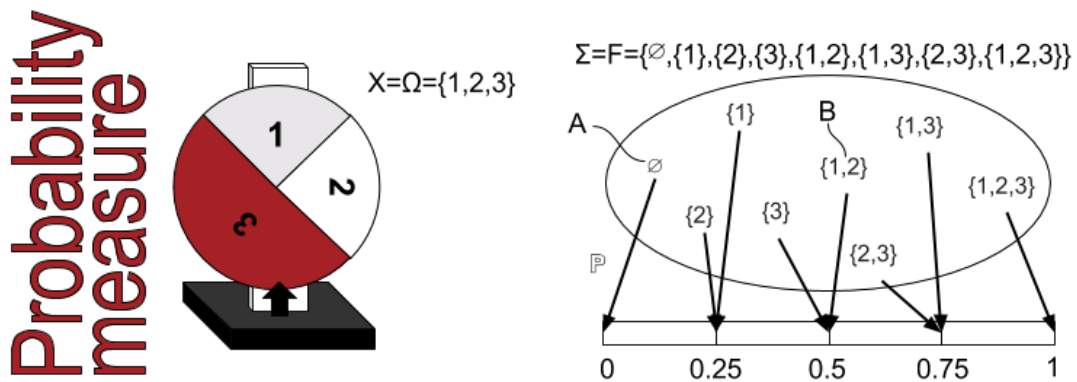


Figure 2.1: A probability measure \mathbb{P} mapping the probability space for 3 events to the unit interval. Left: a turning wheel representing three possible outcomes of which the third is twice as likely as the other two outcomes. Right: a probability measure \mathbb{P} assigned to each outcome. The empty set, $A = \emptyset$, has probability measure 0. The set of encountering either 1 or 2, $B = \{1, 2\}$, has probability measure 0.5. Taken from Wikipedia.

A probability space (X, Σ, \mathbb{P}) is a measure space (X, Σ, μ) with the probability measure \mathbb{P} as its measure μ .

▼ Definition 2.9 — *probability space*

A probability space (X, Σ, \mathbb{P}) is a triple with:

- X a set;
 - Σ a σ -algebra over X ;
 - \mathbb{P} a probability measure from Σ to $[0, 1]$.
-

The triple for the probability space (X, Σ, \mathbb{P}) is also written as $(\Omega, \mathbb{F}, \mathbb{P})$. The space X is the event space Ω , the set of *elementary outcomes*. The σ -algebra over subsets of Ω is denoted by \mathbb{F} . The probability measure \mathbb{P} assigns a value on the unit interval $[0, 1]$ to every event in \mathbb{F} .

B: Counting measure

The *counting measure* forms the basis for the definition of discrete probabilities (Schilling, 2005).

▼ Definition 2.10 — *counting measure*

A counting measure ν on a space X is a measure μ with:

- ν is non-negative and integer-valued for $\forall A \in \Sigma$;
 - $\nu < \infty$ for $\forall A \in \Sigma$ if A bounded (of finite size);
 - $\nu = \infty$ if $\exists A \in \Sigma$ with A unbounded (infinite).
-

A counting measure is a measure that is integer-valued. Every set A has a measure that is a positive integer or zero. The set A is unbounded if and only if its counting measure is infinite.

C: Borel measure

The *Borel σ -algebra* defines a σ -algebra for the real line \mathbb{R} .

▼ Definition 2.11 — *Borel σ -algebra*

A Borel σ -algebra \mathbb{B}_σ on \mathbb{R} is the smallest σ -algebra that contains all open subsets of \mathbb{R} :

- $\mathbb{B} = \Sigma(U)$ with $U = \bigcup U \subseteq \mathbb{R}$: U is open.
-

508 The Borel σ -algebra contains all open subsets of \mathbb{R} . The property of closure under comple-
 509 mentation of a σ -algebra means that it also contains the closed subsets of \mathbb{R} . If $A = (0, 1)$,
 500 then $A^c = \{[-\infty, 0], [1, \infty]\}$.

501 A Borel measure assigns values to subsets of \mathbb{B}_σ .

502 **▼ Definition 2.12 — Borel measure**

503

A Borel measure μ is a function from $\Sigma = \mathbb{B}_\sigma$ to $[-\infty, +\infty]$, with the three measure
 504 requirements:
 505

- 506 ◦ μ is non-negative: $\mu(A) \geq 0$ for $\forall A \in \Sigma$;
 - 507 ◦ μ has a null empty set: $\mu(\emptyset) = 0$;
 - 508 ◦ μ is σ -additive: $\mu(\bigcup_{i \in I_\Sigma} A_i) = \sum_{i \in I_\Sigma} \mu(A_i)$ for A_i disjoint.
- 509

510 The Borel space is a measurable space with a Borel σ -algebra rather than a general σ -
 511 algebra.

512 **▼ Definition 2.13 — Borel space**

513

A Borel space (X, \mathbb{B}_σ) is a pair with:
 514

- 515 ◦ X a set;
 - 516 ◦ \mathbb{B}_σ a Borel σ -algebra over X .
- 517

518 A complete measure space is a measure space in which every subset of every null set is mea-
 519 surable.

520 **▼ Definition 2.14 — complete measure space**

521

A complete measure space (X, Σ, μ) :
 522

- 523 ◦ $S \subseteq N \in \Sigma$ and $\mu(N) = 0 \Rightarrow S \in \Sigma$.
- 524

525 The Borel space is not a complete measure space. There are sets in the Borel σ -algebra that
 526 are of measure zero and that contain subsets that are undefined.

527 **D: Lebesgue measure**

528 The Lebesgue measure defines a size to subsets of \mathbb{R}^n that completes the Borel measure
 529 (Lebesgue, 1902). It makes use of the notion of an outer measure.

530 **▼ Definition 2.15 — outer measure**

531

An outer measure ϕ on a space \mathbb{R} is a measure μ with:
 532

- 533 ◦ ϕ is non-negative and real-valued for $\forall A \in \Sigma$;
 - 534 ◦ ϕ has a null empty set: $\phi(\emptyset) = 0$;
 - 535 ◦ ϕ is σ -subadditive: $\phi(\bigcup_{i \in I_\Sigma} A_i) \leq \sum_{i \in I_\Sigma} \mu(A_i)$ for $\forall A_i$;
 - 536 ◦ ϕ is monotone: $A \subseteq B$ implies $\phi(A) \leq \phi(B)$;
 - 537 ◦ ϕ is translation-invariant: $\phi(A + x) = \phi(A)$ for $\forall A \in \Sigma$ and $\forall x \in \mathbb{R}$.
 - 538
-

539 An outer measure relaxes σ -additivity of disjoint sets of X to σ -subadditivity for any se-
 540 quence of sets. Intuitively, the outer measure of a set is an upper bound on the size of a
 541 set.

542 ▼ **Definition 2.16 — Lebesgue outer measure**

543 A **Lebesgue outer measure** λ on a space \mathbb{R}^n is an outer measure ϕ with:

- 545 ◦ $\lambda(A) = \inf \left\{ \sum_{k=1}^{\infty} l(I_k) : (I_k)_{k \in \mathbb{N}} \text{ is a sequence of open intervals with } A \subseteq \bigcup_{k=1}^{\infty} I_k \right\}$.
 - 546
-

547 Here $A \subseteq \mathbb{R}$ is a subset of the real line. The Lebesgue outer measure λ is the infimum (greatest
 548 lower bound) of the sum of the lengths $l(I) = b - a$ of the intervals $I = [a, b]$.

549 The *Lebesgue measure* is defined through the Lebesgue outer measure.

550 ▼ **Definition 2.17 — Lebesgue measure**

551 A **Lebesgue measure** m on a space \mathbb{R}^n is a Lebesgue outer measure λ with:

- 553 ◦ $m(B) = \lambda(B \cup A) + \lambda(B \cup A^c)$.
 - 554
-

555 **E: Random measure and random process**

556 A measurable function is defined between two measurable spaces.

557 ▼ **Definition 2.18 — measurable function**

558 A **measurable function** $f : X \rightarrow Y$ fulfills:

- 560 ◦ $f^{-1}(E) \in \Sigma$ for $\forall E \in T$,

561 with both (X, Σ) and (Y, T) measurable spaces.

562

563 A measurable function *preserves the structure* of the corresponding measurable spaces
 (captured through the σ -algebras).

564 A random variable is a measurable function between two measurable spaces, with as domain
 565 a measurable space that is a probability space.

566 ▼ **Definition 2.19 — general random variable**

567 A (X, Σ) -valued random variable X is a measurable function from probability space
 568 $(\Omega, \mathbb{F}, \mathbb{P})$ to measurable space (X, Σ) .
 569

571 A random variable is a (X, Σ) -valued random variable with a choice for the codomain and
 572 σ -algebra (see Definition 2.20).

573 ▼ **Definition 2.20 — random variable**

574 A random variable X is a measurable function from probability space $(\Omega, \mathbb{F}, \mathbb{P})$ to the
 575 real line with the Borel σ -algebra $(\mathbb{R}, \mathbb{B}_{\mathbb{R}})$.
 576

578 The codomain is the real line \mathbb{R} and the Borel σ -algebra.

579 Random variables can be generalized to complex random variables or random elements of any type. A complex random variable is a measurable function from Ω to \mathbb{C} . A random elephant is a measurable function from Ω to a suitable space of elephants (Kingman, 1993).

580 ▼ **Definition 2.21 — random measure**

581 A random measure is a function $\xi : \Omega \times X \rightarrow [0, +\infty]$ from probability space $(\Omega, \mathbb{F}, \mathbb{P})$
 582 to measurable space (X, Σ) such that $\xi(\cdot, X)$ is a random variable on $(\Omega, \mathbb{F}, \mathbb{P})$ and $\xi(\omega, \cdot)$
 583 is a measure on Σ .
 584

586 We have encountered a random variable, and a probability measure \mathbb{P} on the original probability space. Now, one might wonder whether probabilities are logically assigned to elements on the measurable space that is the codomain of this random variable. Why does
 587 it map to a measurable space and not a measure space actually? This is because (through
 588 the σ -algebras of both spaces, or more precisely the random variable itself) the probability
 589 measure is *induced* on the target space. This is known as a *probability distribution*:
 590
 591

592 ▼ **Definition 2.22 — probability distribution**

593 Given a random variable X from $(\Omega, \mathbb{F}, \mathbb{P})$ to $(\mathbb{R}, \mathbb{B}_{\sigma})$, the **probability distribution** μ is
 594 the induced probability measure: $\mu(B) = \mathbb{P}(X^{-1}(B))$ for all Borel sets $B \in \mathbb{B}_{\sigma}$.
 595

596 The measurable function X is inverted: $X^{-1}(\cdot)$. The measure μ exists on $(\mathbb{R}, \mathbb{B}_{\sigma})$ just as \mathbb{P}
 597 exists on (Ω, \mathbb{F}) . The notation for the measure μ does not include the original probability
 598 space or σ -algebra. The complete notation for the probability distribution μ can be written
 599 as a function f of X :
 600

$$f_X(x) = f_{X, (\Omega, \mathbb{F}, \mathbb{P}), (\mathbb{R}, \mathbb{B}_{\sigma})}(x). \quad (2.1)$$

At the left X denotes the random variable, $x \in \Omega$ are the (elementary) outcomes on the sample space Ω . At the right the complete notation adds \mathbb{F}, \mathbb{P} and $\mathbb{R}, \mathbb{B}_\sigma$. The shorthand notation at the left will be used to indicate the real line with a Borel σ -algebra as codomain.

A random variable X is *distributed as* $f_X(x)$, notation:

$$X \sim f_X(x). \quad (2.2)$$

A *random process* is an *ordered* set of random variables. The set can be a sequence of random variables in a time series. It can be a series of steps in the spatial domain, called a random field.

▼ **Definition 2.23 — random process**

A **random process** X is a collection $\{X_t : t \in T\}$ with X_t an (S, Σ) -valued random variable on Ω and $(\Omega, \mathbb{F}, \mathbb{P})$ a probability space, (S, Σ) a measurable space, and T a totally ordered set.

A random process is a probability distribution with a domain that is a set of probability distributions. A random process is a distribution over distributions, a hierarchy over distribution.

2.1.2 Bayesian Inference

Let x be a (S, Σ_S, μ_S) -valued random variable¹, y a (T, Σ_T, μ_T) -valued random variable, then we can construct z , a (C, Σ_C, μ_C) -valued random variable with the latter being a subset of the product set of x and y : $C \in S \otimes T$.

▼ **Definition 2.24 — product space**

A **product space** $(S \otimes T, \Sigma_{S \otimes T})$ has σ -algebra $\Sigma_{S \otimes T} = \sigma(F \otimes G : F \in \Sigma_S, G \in \Sigma_T)$ with (S, Σ_S, μ_S) and (T, Σ_T, μ_T) two σ -finite measure spaces.

▼ **Definition 2.25 — product measure**

A **product measure** $\mu_{S \otimes T}$ is a measure $\mu_{S \otimes T}(F \otimes G) = \mu_S(F) \otimes \mu_T(G)$ with (S, Σ_S, μ_S) and (T, Σ_T, μ_T) two σ -finite measure spaces.

The **joint probability distribution** P_C is a probability measure on the product σ -algebra Σ_C with $C \in S \otimes T$. As function of the random variables x and y the joint probability distribution is written as $x_{,Y}(x, y)$, $f(x, y)$, or $p(x, y)$.

A σ -algebra is *independent* in the following sense.

¹The lowercase x is used instead of X in the context of probability distributions as in Eq. 2.1.

▼ **Definition 2.26 — independent σ -algebra**

Let (Ω, \mathbb{F}, P) be a probability space and \mathbb{A} and \mathbb{B} be sub- σ -algebras of \mathbb{F} . \mathbb{A} and \mathbb{B} are **independent σ -algebras** if:

- $P(A \cap B) = P(A)P(B) \quad \forall A \in \mathbb{A} \text{ and } B \in \mathbb{B}.$

Two random variables x and y are independent if and only if the σ -algebras that they generate are independent.

▼ **Definition 2.27 — conditional probability distribution**

Let (Ω, \mathbb{F}, P) be a probability space, $\mathbb{G} \subseteq \mathbb{F}$ a sub- σ -algebra of \mathbb{F} , and $X : \Omega \rightarrow \mathbb{R}$ a real-valued random variable (\mathbb{F} -measurable with respect to the Borel σ -algebra \mathbb{B}_σ on \mathbb{R}). There exists a function $\mu : \mathbb{B}_\sigma \times \Omega \rightarrow \mathbb{R}$ such that $\mu(\cdot, \omega)$ is a probability measure on \mathbb{B}_σ for each $\omega \in \Omega$ and $\mu(H, \cdot) = P(X \in H | \mathbb{G})$ (almost surely) for every $H \in \mathbb{B}_\sigma$. For any $\omega \in \Omega$, the function $\mu(\cdot, \omega) : \mathbb{B}_\sigma \rightarrow \mathbb{R}$ is called a **conditional probability distribution** of X given \mathbb{G} .

Informally, a conditional probability is described with a sub- σ -algebra which only presents part of the structure of the full σ -algebra. As function of the random variables x and y the conditional probability distribution of y given x is written as $f_{Y|X}(y|x)$, $f(y|x)$, or $p(y|x)$.

The random variables x and θ define a Bayesian model with observations x and parameters θ .

▼ **Definition 2.28 — Bayesian model**

A **Bayesian model** $f(x, \theta)$ defines a function between observations x and parameters θ with both x and θ random variables.

In a **supervised learning** task both x and θ are known. In an **unsupervised learning** task x is known, but θ is unknown. The random variable θ is called a hidden or latent variable. The random variable θ can be any random element: a random vector, a random matrix, a random process.

Let the observations x be a sequence x_0, x_1, \dots , then the observations x_i can be distributed *independent and identically*.

▼ **Definition 2.29 — independent and identically distributed**

A collection of random variables $x = \{x_0, x_1, \dots\}$ is **independent and identically distributed (i.i.d.)** if:

- the probability distribution $p(x_i)$ is the same for $\forall x_i \in x$;
 - each x_i is independent with respect to x_j with $i \neq j$.
-

The observations x_i can be distributed in an *exchangeable* sequence in which any order is equally likely.

▼ **Definition 2.30 — *exchangeable***

A sequence of random variables $x = \{x_0, x_1, \dots\}$ is **exchangeable** if for any finite permutation ρ of the indices $0, 1, \dots$:

- the joint probability distribution of the permuted sequence $p(x_{\rho(0)}, x_{\rho(1)}, \dots)$ equals that of the original sequence $p(x_0, x_1, \dots)$.
-

The joint probability distribution of i.i.d. observations given parameters can be written as a product:

$$p(x_0, \dots, x_{k-1} | \theta) = \prod_{i=0}^{k-1} p(x_i | \theta). \quad (2.3)$$

▼ **Definition 2.31 — *likelihood function***

The **likelihood function** is defined as:

$$\mathcal{L}(\theta | x) = p(x | \theta). \quad (2.4)$$

The likelihood of the parameters θ given observations x is the probability of these observations given the parameter values.

The definition of the likelihood function allows us to find an optimal set of parameter values given the observations. The probability $p(x | \theta)$ can be maximized (Aldrich and Others, 1997).

▼ **Definition 2.32 — *maximum likelihood***

Maximum likelihood is defined as:

$$\theta^* \in \operatorname{argmax}_{\theta} \prod_{i=0}^{k-1} p(x_i | \theta). \quad (2.5)$$

The maximum likelihood method finds the maximum of $\prod_{i=0}^{k-1} p(x_i | \theta)$ for all possible parameter values θ . The maximum in maximum likelihood does not need to be unique (Steel, 1994). The notation makes this explicit by writing θ^* as a member (denoted by the \in symbol) of the outcomes of the argmax operation (and does not use the equal sign).

A function $f(\cdot)$ and the logarithm of a function $\log f(\cdot)$ have the same maxima. This is due to the fact that the logarithm is a monotonic function (a monotonically increasing function). The log of a product of logarithms is equal to the sum of the individual logarithms.

▼ **Definition 2.33 — maximum log-likelihood**

Maximum log-likelihood is defined as:

$$\theta^* \in \operatorname{argmax}_{\theta} \sum_{i=0}^{k-1} \log p(x_i | \theta). \quad (2.6)$$

In the case we have information about the parameters θ we can model this with a probability distribution.

▼ **Definition 2.34 — prior probability distribution**

A **prior probability distribution** defines a probability distribution $p(\theta)$ to parameters θ without a dependency on the observations x .

Given the definition of a prior probability distribution, we can define *maximum a posteriori* estimation.

▼ **Definition 2.35 — maximum a posteriori**

maximum a posteriori estimation is defined as:

$$\theta^* \in \operatorname{argmax}_{\theta} \sum_{i=0}^{k-1} \log p(x_i | \theta) + \log p(\theta). \quad (2.7)$$

If we are not only interested in the parameter θ^* that maximizes $p(x|\theta)$ and $p(\theta)$, but in the complete distribution for $p(\theta)$ we need Bayes' theorem described by Laplace (1820).

▼ **Definition 2.36 — Bayesian inference**

Bayesian inference using Bayes' theorem is defined as:

$$f(\theta|x) = p(\theta|x) = \frac{\overbrace{p(x|\theta)}^{\text{likelihood}} \overbrace{p(\theta)}^{\text{prior}}}{\underbrace{p(x)}_{\text{normalization constant}}} = \frac{p(x|\theta)p(\theta)}{\int p(x|\theta)p(\theta)d\theta}. \quad (2.8)$$

Bayes' theorem describes the posterior probability $p(\theta|x)$ as the likelihood times the prior probability distribution divided by a normalization constant, also called the evidence. The normalization constant is not a function of the parameters θ . If a function is known except for the normalization constant, it is indicated by the "proportional to" symbol \propto .

$$f(\theta|x) \propto p(x|\theta)p(\theta) \quad (2.9)$$

In Bayesian inference $p(\theta|x)$ is calculated. In contrast, in maximum likelihood and maximum a posteriori only parts of Eq. 2.8 are calculated, respectively $p(x|\theta)$ and $p(x|\theta)p(\theta)$. In Section 2.3 inference methods will be described that approximate Bayesian inference. Approximation is required in the case closed-form expressions are not available. If the inference task only requires maximum a posteriori, approximation methods are also available (Daume, 2007), but this is outside of the scope of the current thesis.

There are two supervised learning models, a generative model and a discriminative model. Below we provide their definitions and in Figure 2.2 we give three examples for each model.

▼ **Definition 2.37 — generative model**

A **generative** model defines the joint probability distribution $p(x, \theta)$ and uses Bayes rule to define $p(x|\theta)$.

▼ **Definition 2.38 — discriminative model**

A **discriminative** model defines the conditional probability distribution $p(x|\theta)$ directly.

Figure 2.2 shows three generative and three discriminative models. They are chosen for their structure; from left to right, in both cases, the structure between the random variables is enriched, first in the form of a sequence structure, then in the form of a graph structure. Figure 2.2 visualizes three generative models: (1) the Naive Bayes Model (Russell et al., 1995), (2) the Hidden Markov Model (Baum and Petrie, 1966), and (3) the Directional Model (Koller and Friedman, 2009). It shows also three discriminative models: (1) Logistic Regression, (2) Linear-chain Conditional Random Fields, and (3) general Conditional Random Fields.

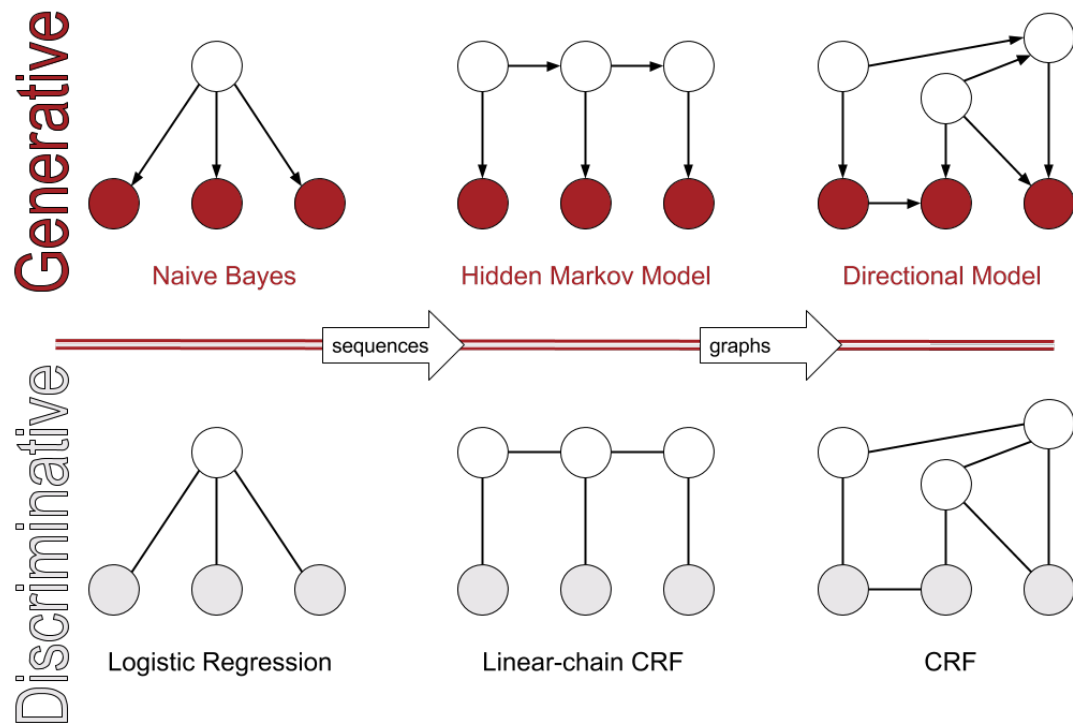


Figure 2.2: Generative models: Naive Bayes Model, Hidden Markov Model, and Directional Model. Discriminative models: Logistic Regression, Linear-chain Conditional Random Fields, and general Conditional Random Fields. Figure adapted from Sutton and McCallum (2011).

There is no definitive reason to use a generative model rather than a discriminative model or vice-versa. Here we confine ourselves to two remarks. First, a discriminative model has lower asymptotic error, but a generative model approaches its asymptotic error faster in the case of a Naive Bayes classifier versus Logistic Regression (Jordan, 2002). However, Xue and Titterington (2008) doubt the existence of such precisely defined regimes. According to them the asymptotic error denotes the error with an increasing number of samples. Second, the prior $p(\theta)$ in the generative model provides a principled way to handle missing information, while the direct modeling of decision boundaries in a discriminative model often leads to better performance in a classification task (Jaakkola et al., 1999). Apart from generative models and discriminative models, there are also hybrid models (Bouchard and Triggs, 2004; Raina et al., 2003; Bosch et al., 2008). In the thesis we will limit ourselves to generative models.

2.1.3 Model Composition

A model can be composed out of a set of probability distributions. We list three of such possible compositions. The Naive Bayes model is a *product* of probability distributions with a prior distribution (Definition 2.39). The finite mixture model is a *sum* over a finite number

of probability distributions where each one is weighted (Definition 2.40). The infinite mixture model is a *sum* over an infinite number of probability distributions where each one is weighted (Definition 2.41).

▼ **Definition 2.39 — *naive Bayes model***

The **naive Bayes model** is a product over a finite number $k \neq \infty$ of probability distributions $p(x_i|\theta)$ multiplied by the prior distribution $p(\theta)$:

$$p(\theta|x) \propto p(\theta) \prod_{i=0}^{k-1} p(x_i|\theta). \quad (2.10)$$

A finite mixture model is a sum over a finite number of probability distributions.

▼ **Definition 2.40 — *finite mixture model***

A **finite mixture model** is a sum over a finite number $k \neq \infty$ of probability distributions $p(x_i)$, with each distribution weighted by a factor w_i with $\sum_i w_i = 1$.

$$p(x) = \sum_{i=0}^{k-1} w_i p(x_i). \quad (2.11)$$

The mixture model is finite in the sense that there are only $k \neq \infty$ distributions summed up. The weights of the individual distributions $p(x_i)$ are normalized (sum up to one) such that the weighted sum over the probability distributions is itself a probability distribution.

An infinite mixture model is a sum over an infinite number of probability distributions.

▼ **Definition 2.41 — *infinite mixture model***

A **infinite mixture model** is a sum over an infinite number of probability distributions $p(x_i)$, with each distribution weighted by a factor w_i with $\sum_i w_i = 1$.

$$p(x) = \sum_{i=0}^{\infty} w_i p(x_i). \quad (2.12)$$

The infinite mixture model is a sum over an infinite number of probability distributions with weights that sum up to one. In this way it assigns a finite value to a countably infinite set of functions. In the thesis we will encounter infinite mixture models in Chapters 3 and 4.

A mixture model relates one latent cause (cluster) to each data point. A factorial model or feature allocation model assigns multiple factors to a data point.

▼ Definition 2.42 — *feature model*

A **feature model** is a sum over a number k of sets of probability distributions $p(x_{C_i})$.

$$p(x) = \sum_{i=0}^{k-1} \sum_{j=0}^{C_i} p(x_j). \quad (2.13)$$

A counting model allows each feature to occur multiple times.

▼ Definition 2.43 — *counting model*

A **counting model** is a sum over a number k of sets of probability distributions $p(x_{C_i})$ with each feature occurring a number c_j of times.

$$p(x) = \sum_{i=0}^{k-1} \sum_{j=0}^{C_i} c_j p(x_j). \quad (2.14)$$

2.1.4 General Random Elements

In section 2.1.1 random elements were described in general. Random elements can vary from random vectors, random distributions, random clusters (partitions), to random trees. Table 2.1 describes the random elements and the corresponding examples of random processes in the literature. Below we mention them with the appropriate references.

The Gaussian Process (Rasmussen and Williams, 2006) describes a distribution on functions. The Beta Process (Hjort, 1990), the Gamma Process (Ferguson, 1974), the Dirichlet Process and the Polya Tree (Ferguson, 1973) describe a distribution on distributions. The Chinese Restaurant Process (Aldous, 1985) and Pitman-Yor Process (Pitman and Yor, 1997) describe a distribution on partitions (in the form of cluster assignments). The Stick-breaking Process describes a distribution on partition sizes (with no information on assignments themselves). The Dirichlet Diffusion Tree (Neal, 2001) and Kingman's coalescence (Kingman, 1965) describe a distribution on hierarchical partitions. The Indian Buffet Process (Ghahramani and Griffiths, 2005) describes a distribution over sparse binary matrices. The Gamma-Poisson Process (Titsias, 2008) describes a distribution over integer-valued matrices. The Mondrian Process (Roy and Teh, 2009) describes a distribution over kd-trees.

2.1.5 Plate Notation

Random processes and mixture models are visually represented by a method called *plate notation* (cf. Buntine, 1994; Koller and Friedman, 2009). Sets of variables are represented in a plate, a rectangular region (see Figure 2.3).

Table 2.1: A list of seven mathematical structures and for each of these structures one or more random processes that can generate the structure. For example, a distribution on distributions can be generated by a Beta Process, Gamma Process, Dirichlet Process, or a Polya Tree.

Structure	Example
Distribution on functions	Gaussian Process
Distribution on distributions	Beta Process
	Gamma Process
	Dirichlet Process
	Polya Tree
Distribution on partition assignments	Chinese Restaurant Process Pitman-Yor Process
Distribution on partition sizes	Stick-breaking Process
Distribution on hierarchical partitions	Dirichlet Diffusion Tree Kingman’s coalescence
Distribution on sparse binary matrices	Indian Buffet Process
Distribution on integer-valued matrices	Gamma-Poisson Process
Distribution on kd-trees	Mondrian Process

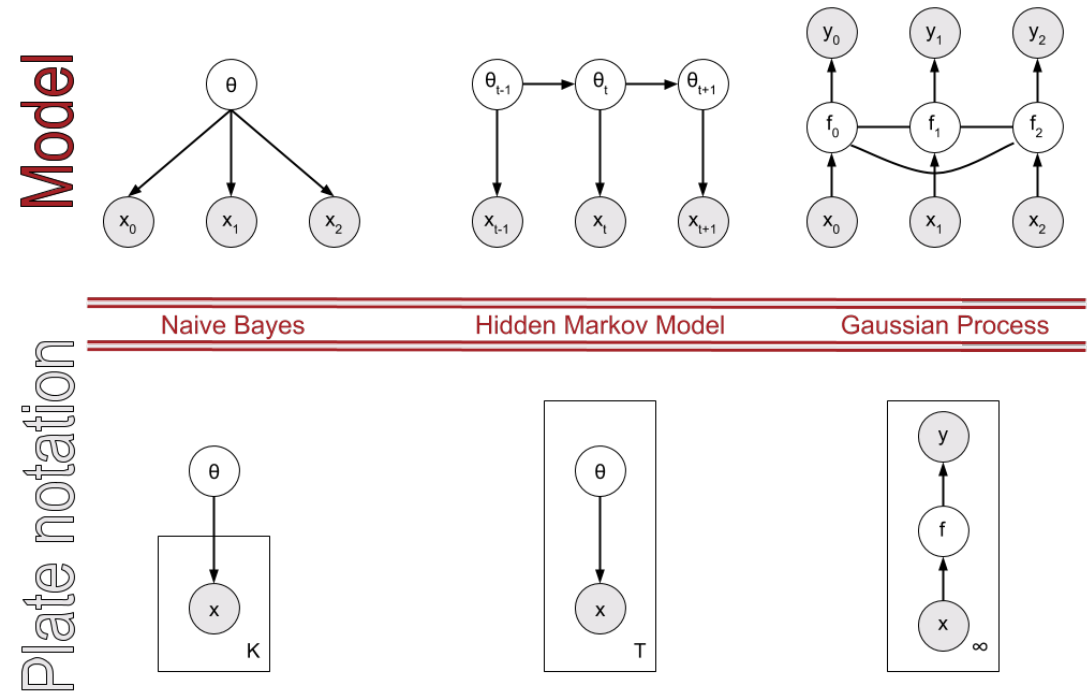


Figure 2.3: Top: graphical model of a Naive Bayes, hidden Markov model, and Gaussian process. Bottom: corresponding plate notation of the Naive Bayes, hidden Markov model, and Gaussian process. Observed variables are denoted by a circle that is shaded.

842 Plate notation is a representation that does not preserve all dependencies between variables.

For example, the dependencies between the states in the Hidden Markov Model (e.g., between θ_0 and θ_1) are not represented.

2.1.6 Completely Random Measure and Lévy Measure

Some random process are mathematically represented by a completely random measure (Kingman, 1967), which is defined as follows.

▼ Definition 2.44 — *completely random measure*

A **completely random measure** is a random measure $\mu : \Omega \times X \rightarrow [0, +\infty]$ from probability space $(\Omega, \mathbb{F}, \mathbb{P})$ to measurable space (X, Σ) with

- for any collection of disjoint sets $A_1, \dots, A_k \in \Sigma$ and $A_i \cap A_j = \emptyset$ for $i \neq j$ a mutual independency between $\mu(A_1), \dots, \mu(A_k)$.

Kingman (1967) shows that a completely random measure can be decomposed into three components:

1. a deterministic function;
2. a countable set of non-negative random masses at deterministic locations;
3. a countable set of non-negative random masses at random locations.

The first component is a deterministic function. The second component has non-negative random masses, also called atoms, on deterministic locations. The third component is the one of interest. It has a set of random masses (atoms) that can be represented as a Poisson random measure on $\mathbb{R}^+ \otimes X$ with mean measure ν which is known as the Lévy intensity measure (Favaro et al., 2013).

Table 2.2: Lévy measure of the Beta Process (Wang and Carin, 2012), Gamma Process (Knowles et al., 2014), the Dirichlet Process (Lijoi and Prünster, 2010) (indirectly through $F = 1 - e^{-\nu}$).

Random Process	Lévy measure
Beta Process	$\nu(da, dw) = H(da)\alpha w^{-1}(1-w)^{\alpha-1}dw$
Gamma Process	$\nu(da, dw) = H(da)w^{-1}e^{-\alpha w}dw$
Dirichlet Process	$\nu(da, dw) = H(da)e^{-w\alpha(x, \infty)}(1-e^{-w})^{-1}dw$

For Lévy measure decompositions of other processes such as the Indian buffet process, we refer to Wang and Carin (2012).

2.1.7 Exchangeability

Here we recall Definition 2.30 for exchangeable sequences. De Finetti's theorem states that there is parameter θ such that the data x_i is conditionally independent given this parameter for exchangeable sequences (cf. De Finetti, 1937).

▼ Definition 2.45 — De Finetti's theorem

A sequence $\{x_0, x_1, \dots\}$ of (X, Σ_X) -valued random variables is an infinitely exchangeable sequence if and only if there exist a measure $\mu(d\theta)$ on θ such that

$$p(x_0, \dots, x_{k-1}) = \int_{\Sigma_X(X)} \prod_{i=0}^{k-1} p(x_i|\theta) \mu(d\theta) \quad \forall k \geq 1. \quad (2.15)$$

In words, de Finetti's theorem states that if we have *exchangeable* data, we have a parameter θ , a likelihood $p(x|\theta)$, and some measure μ on θ , such that the data (x_0, \dots, x_{k-1}) is *conditionally independent*. Hence, although the data is not i.i.d., there are underlying, unobservable, quantities that are i.i.d. and exchangeable sequences are mixtures of these quantities. The theorem proofs that if the observations are exchangeable, they must be a random sample from some model and there must exist a prior probability distribution over the parameters of that model, hence requiring a Bayesian approach.

The theorem is not limited to exchangeable *sequences*. In contrast, there are similar theorems for other exchangeable objects (Orbanz and Roy, 2015). Five examples (see Table 2.3) of exchangeable structures that have a theorem that describes an underlying measure that can be sampled i.i.d. are: (1) exchangeable sequences (de Finetti, 1930), (2) increments (Bühlmann, 1960), (3) partitions (Kingman, 1978), (4) arrays (Aldous, 1981), and (5) Markov chains (Diaconis and Freedman, 1980).

Table 2.3: Five exchangeable structures and their theorems.

Mathematical Object	Theorem
Exchangeable Sequence	de Finetti
Exchangeable Increment	Bühlmann
Exchangeable Partition	Kingman
Exchangeable Array	Aldous-Hoover
Exchangeable Markov Chain	Diaconis-Freedman

2.1.8 Stick-breaking Representation

Now we introduce the *stick-breaking representation* by Freedman and Diaconis (1983), also known as the residual allocation model (Sawyer and Hartl, 1985; Hoppe, 1986).

▼ **Definition 2.46 — stick-breaking**

An infinite sequence of random variables $\phi = \{\phi_0, \phi_1, \dots\}$ has a **stick-breaking representation** with parameters α and β denoted by $\phi \sim GEM(\alpha, \beta)$.

$$w_k \stackrel{i.i.d.}{\sim} \text{Beta}(1 - \beta, \alpha + k\beta) \quad k = 1, \dots, K \quad (2.16)$$

$$\phi_k = w_k \prod_{i=1}^{k-1} (1 - w_i) \quad (2.17)$$

893

894

895 The stick-breaking process samples repeatedly from a $\text{Beta}(1 - \beta, \alpha + k\beta)$ distribution. The
 896 result of the process is a vector of k weights ϕ_k . The abbreviation *GEM* stands for Griffiths,
 897 Engen, and McCloskey (Ewens, 1990; Ethier, 1990). There is also a variant of GEM with a
 898 single parameter α which can be obtained by setting $\beta = 0$. In that case w_k are drawn from
 899 a $\text{Beta}(1, \alpha)$ distribution.

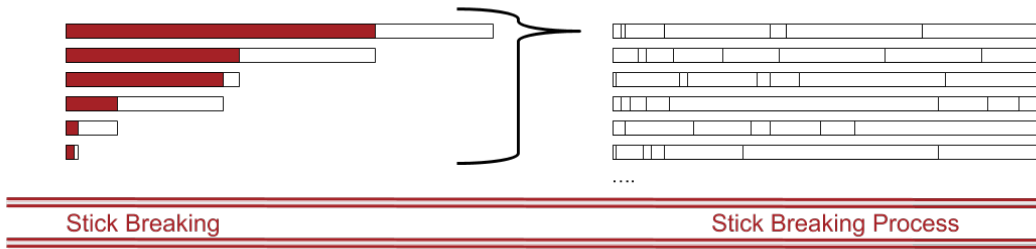


Figure 2.4: The stick-breaking representation. Left: at the first row, the stick is broken at x_0 , at the next rows the remaining part of the stick is broken x_i with $i > 0$. Only six iterations are shown. Right: samples of a stick-breaking process. The first row shows the stick ratios from the stick-breaking representation at the left. The next rows show other samples from the same process.

900 Figure 2.4 visualizes the stick-breaking process. A stick of fixed length 1 gets broken at a
 901 position w_0 sampled from a Beta distribution. The remainder of the stick is broken again at
 902 position $w_1(1 - w_0)$. This process continues for an infinite number of times. In this manner
 903 generates a stick-breaking process a sequence of non-negative values that sum up to one.
 904 The stick-breaking representation can on itself give rise to more sophisticated stochastic
 905 processes (Dunson et al., 2012). Computationally it can also fulfill a useful rule. Namely,
 906 it is possible to approximate a distribution over partitions by truncating a stick-breaking
 907 process. The stick-breaking is then only performed a limited number of times (Kurihara
 908 et al., 2007).

909 In Section 2.2.1 the relevance of the stick-breaking process for the Dirichlet process will be
 910 shown. In that case the values generated by the stick-breaking process represent the weights
 911 of the partitions induced by the Dirichlet Process.

2.2 Five Random Processes

In this section we investigate five random processes. The Dirichlet process (Section 2.2.1), the Beta process (Section 2.2.2), the Gamma process (Section 2.2.3), the Pitman-Yor process (Section 2.2.4), and the hierarchical Dirichlet process (Section 2.2.5). We compare the random processes in Section 2.2.6.

2.2.1 Dirichlet Process

The Dirichlet process is presented as a measure (Section A), is shown to have a sequential representation in the form of the Chinese restaurant process (Section B), and a stick-breaking representation (Section C).

A: Dirichlet Process as a Measure

The Dirichlet process (DP) is a distribution over distributions (Ferguson, 1973).

▼ Definition 2.47 — Dirichlet process

A Dirichlet process DP over a set S can be used to draw sample paths X :

$$X \sim DP(\alpha, H)$$

with α the dispersion parameter and H a measure on S and for which any measurable partition $\{B_0, \dots, B_{n-1}\} \in S$ is drawn from a Dirichlet distribution:

$$(X(B_0), \dots, X(B_{n-1})) \sim \text{Dirichlet}(\alpha H(B_0), \dots, \alpha H(B_{n-1}))$$

The Lévy intensity of the Dirichlet process is complicated, because it is a so-called normalized process, see Regazzini et al. (2003).

The Dirichlet process can be used as a prior for a mixture model (Definition 2.41). This is visualized in (Figure 2.5).

	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	sum
data 0	0	0	1	0	0	0	0	0	0	...	1
data 1	1	0	0	0	0	0	0	0	0	...	1
data 2	1	0	0	0	0	0	0	0	0	...	1
data 3	0	1	0	0	0	0	0	0	0	...	1
data 4	1	0	0	0	0	0	0	0	0	...	1
data 5	0	1	0	0	0	0	0	0	0	...	1
data 6	0	0	0	0	1	0	0	0	0	...	1

Figure 2.5: Matrix representation of a mixture model. At the horizontal axis the latent variables (potentially infinite clusters). At the vertical axis the data items. The rows sum up to one. A data item is only assigned to one cluster.

929 B: Chinese Restaurant Process

930 De Finetti's theorem (Definition 2.45) can be used to establish the existence of an infinitely
 931 exchangeable sequence. In the particular case of the Dirichlet process the sequence is an
 932 exchangeable *distribution over partitions* and is called the CRP.

933 ▼ Definition 2.48 — Chinese restaurant process

934
 935 A **Chinese restaurant process** is a sequential process that is an exchangeable distribu-
 936 tion over partitions:

$$p(z_i = k | z_0, \dots, z_{i-1}) = \begin{cases} \frac{n_k}{\alpha + i} & \text{if } k \leq K_+ \\ \frac{\alpha}{\alpha + i} & \text{if } k > K_+ \end{cases} \quad (2.18)$$

937

938

939 The conditional probability of a cluster assignment z_i for data item y_i given the cluster
 940 assignments z_0, \dots, z_{i-1} is proportional to the number of data items n_k assigned to an existing
 941 cluster k , or proportional to α for a new cluster.

942 The Chinese Restaurant Process is visualized in Figure 2.6.

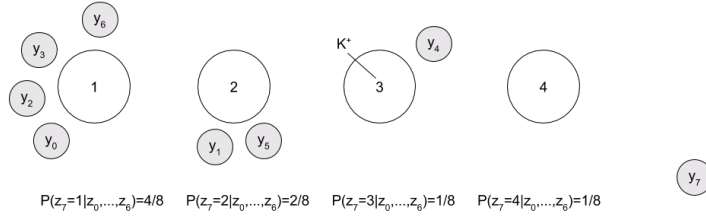


Figure 2.6: The Chinese Restaurant Process with i customers already sitting down. A new customer $y_{i=7}$ arrives and gets assigned, z_i . This is an existing table $\{1, 2, 3\}$ with a probability proportional to the number of customers n_i sitting at that table: $n_i/(\alpha + i)$, or a new, empty table 4 with probability $1/(\alpha + i)$. In the visualized Chinese restaurant process the dispersion factor $\alpha = 1$.

943 C: Stick-breaking Representation of the Dirichlet process

▼ Definition 2.49 — stick-breaking representation of the Dirichlet process

The stick-breaking representation of the Dirichlet process states that if

$$\phi_k \sim GEM(\alpha, 0) \quad (2.19)$$

$$\theta_k \sim H \quad (2.20)$$

$$G = \sum_{k=1}^{\infty} \phi_k \delta(\theta, \theta_k) \quad (2.21)$$

944 then $G \sim DP(\alpha, H)$.

945

946 The weights ϕ_k are sampled from the stick-breaking process $GEM(\alpha, 0)$ (see Definition 2.46).

947 The parameter values θ_k are sampled from the base measure H . To sample from the Dirichlet

948 Process we have to sample these parameters with the given weights.

949 If the stick-breaking process is used as a prior for a mixture, then the cluster assignments z_i

950 are sampled according to the mixing proportions ϕ :

$$\phi \sim GEM(\alpha, 0) \quad (2.22)$$

$$\theta_k \sim H \quad (2.23)$$

$$G = \sum_{k=1}^{\infty} \phi_k \delta(\theta, \theta_k) \quad (2.24)$$

$$z_i \sim Mult(\phi) \quad (2.25)$$

$$x_i \sim F(\theta_{z_i}) \quad (2.26)$$

951 Here $\theta_k = \theta_{z_i}$ for observation with index i and cluster assignment k : $z_i = k$.

952 The Dirichlet process, the Dirichlet process as prior for a mixture model, the Chinese restau-

953 rant process and the stick-breaking representation can be compared in Figure 2.7.

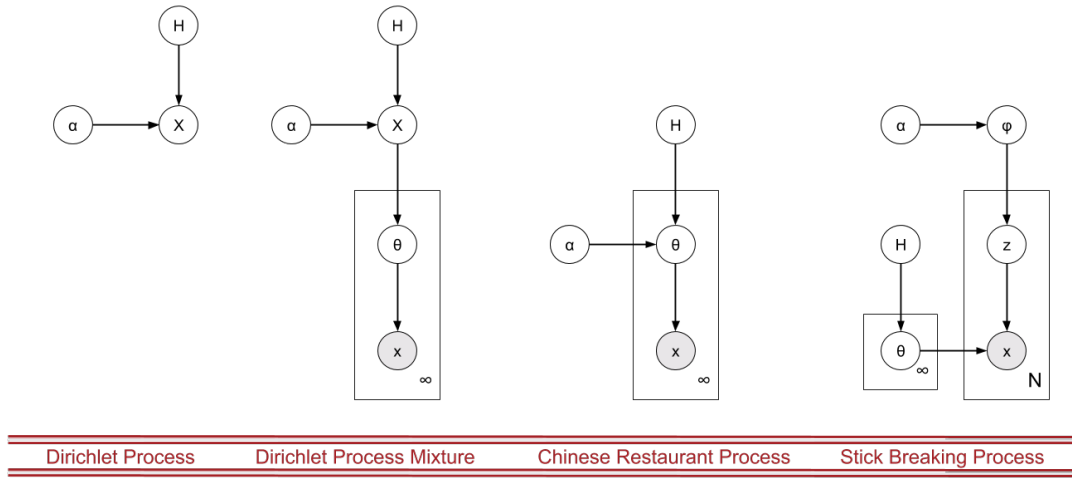


Figure 2.7: From left to right: (1) The Dirichlet process $X \sim DP(\alpha, H)$; (2) The Dirichlet mixture model with X the prior for a sum $\sum_i w_i p(x_i | \theta_i)$; (3) the Chinese restaurant process with X marginalized out; and (4) the Stick-breaking process with a distribution over partition sizes π and indicator variables z_i .

954 2.2.2 Beta Process

955 The Beta process is presented as a measure (Section A), is shown to have a sequential rep-
 956 resentation in the form of the Indian buffet process (Section B), and a stick-breaking repre-
 957 sentation (Section C).

958 A: Beta Process as a Measure

959 A Beta process (Hjort, 1990) is a random process with a countably infinite collection of
 960 weighted atoms in a space (X, \mathbb{B}) with weights that are in between $[0, 1]$.

961 ▼ Definition 2.50

962 Let (X, \mathbb{B}) be a Borel space, ν a finite measure, and $\alpha > 0$ a scale parameter, then a
 963 **Beta process** is a Lévy process on (X, \mathbb{B}) with its Lévy measure ν corresponding to the
 964 density:
 965

$$966 \nu(dw) = \alpha w^{-1} (1-w)^{\alpha-1} dw \quad (2.27)$$

966 with $w > 0$.

967
 968 In Figure 2.8 the Beta Process is generated from a Completely Random Measure (see Sec-
 969 tion 2.1.6) with a Lévy intensity defined on $\Omega \otimes (0, 1)$ (Thibaux and Jordan, 2007). In this
 970 case Ω is the so-called base measure B_0 and is assumed uniform over a bounded region. The
 971 $(0, 1)$ space is equipped with an improper Beta distribution. It is called improper or degen-
 972 erate because the scale parameter of the standard Beta distribution is set to zero. This has
 973 the consequence that the integral is infinite: $\nu(\Omega \otimes (0, 1)) = \infty$. It is due to the fact that

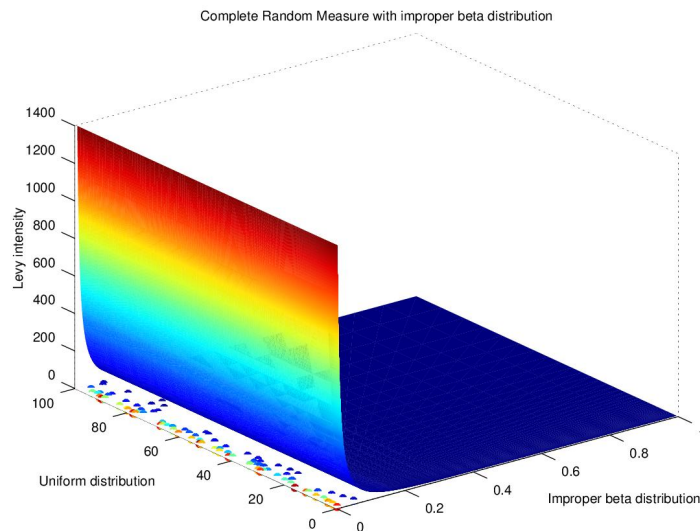


Figure 2.8: A Completely Random Measure with a Lévy intensity defined on the product space $\Omega \otimes (0, 1)$. Here Ω is a bounded interval on which the base measure $B_0 = U(0, 100)$ is defined. On $(0, 1)$ we define an improper beta distribution $\alpha w^{-1}(1 - w)^{\alpha-1}$. In this example $\alpha = 10$. This is how a Beta Process can be generated from a nonhomogeneous spatial Poisson point process. This has been visualized before (Jordan, 2010). The image is produced by rejection sampling using a homogeneous Poisson point process at $\max(\nu)$ over $w = [0.01, 0.9]$. For $w \rightarrow 0$ this maximum would go to ∞ and all points would be rejected. Hence, the points should be denser for w around 0 and should be seen as an approximation of the actual process.

974 the density $w^{-1}(1 - w)^{\alpha-1}$ goes to infinity for $w \rightarrow 0$. That means that a countable infinite
 975 number of points can be obtained from the Poisson process.

976 The Beta process can be used as a prior for a feature or factorial model (Definition 2.42).
 977 This is visualized in (Figure 2.9).

	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	sum
data 0	1	0	1	0	0	0	0	0	0	...	2
data 1	1	1	0	0	1	0	0	0	0	...	3
data 2	1	0	0	0	0	0	0	0	0	...	1
data 3	0	0	0	0	0	0	1	1	1	...	3
data 4	0	1	0	1	1	0	0	0	1	...	4
data 5	0	0	0	0	1	0	0	0	1	...	2
data 6	0	0	1	0	0	0	0	0	0	...	1

Figure 2.9: Matrix representation of a factorial model. At the horizontal axis the latent variables (potentially infinite number of features). At the vertical axis the data items. The rows sum up to nonnegative integers. A data item has multiple features.

978 B: Indian Buffet Process

979 The Beta process has a sequential representation in the form of the Indian Buffet Process:

980 ▼ Definition 2.51 — *Indian buffet process*

981 An **Indian buffet process** is a sequential process that is an exchangeable distribution
982 over sparse binary matrices:
983

$$p(z_{i,j} = k | z_{0,0}, \dots, z_{i-1,K_+}) = \begin{cases} \frac{n_{-i,k}}{i} & \text{if } k \leq K_+ \\ \frac{\lambda^{k_{new}} e^{-\lambda}}{k_{new}!} & \text{if } k > K_+ \end{cases} \quad (2.28)$$

984
985
986 Here $\lambda = \alpha/i$, $k_{new} = K_+ - k$. The i 'th data item samples an existing column with a probability
987 of the number of times it has been sampled before divided by its index, $n_{-i,k}/i$. It samples
988 a new column with a probability according to a Poisson distribution, $\lambda^{k_{new}} e^{-\lambda}/k_{new}!$. The
989 conditional form of the sequential presentation describes a closed-form solution for Gibbs
990 sampling, section 2.3.4 (Ghahramani and Griffiths, 2005).

991 C: Stick-breaking Representation of the Beta process

▼ Definition 2.52 — *stick-breaking representation of the Beta process*

The **stick-breaking representation** of the Beta process states that if

$$\phi_{k,j} \sim GEM(\alpha, 0) \quad (2.29)$$

$$C_k \sim \text{Poisson}(\gamma) \quad (2.30)$$

$$\theta_{k,j} \sim \frac{1}{\gamma} H \quad (2.31)$$

$$G = \sum_{k=1}^{\infty} \sum_{j=1}^{C_k} \phi_{k,j} \delta(\theta, \theta_{k,j}) \quad (2.32)$$

992 then $G \sim BP(\alpha, H)$.
993

994 The Beta process is used in linguistics (He et al., 2013; Vanhainen and Salvi, 2012), computer
995 vision (Zhou et al., 2011; Gao and Sun, 2013), risk assessment (Li et al., 2014), and medicin
996 (Ross et al., 2014).

997 2.2.3 Gamma Process

998 The Gamma process is presented as a measure (Section A), is shown to have a sequential
999 representation in the form of a multi-scoop Indian buffet process (Section B), and a stick-
1000 breaking representation (Section C).

A: Gamma Process as a Measure

A Gamma process is a random process with independent gamma distributed increments (Ferguson, 1974). Below we provide a formal definition.

▼ Definition 2.53 — Gamma process

Let (X, \mathbb{B}) be a Borel space, ν a finite measure, and $\alpha > 0$ a scale parameter, then a **Gamma process** is a Lévy process on (X, \mathbb{B}) with its Lévy measure ν corresponding to the density:

$$\nu(dw) = w^{-1}e^{-\alpha w}dw \quad (2.33)$$

with $w > 0$.

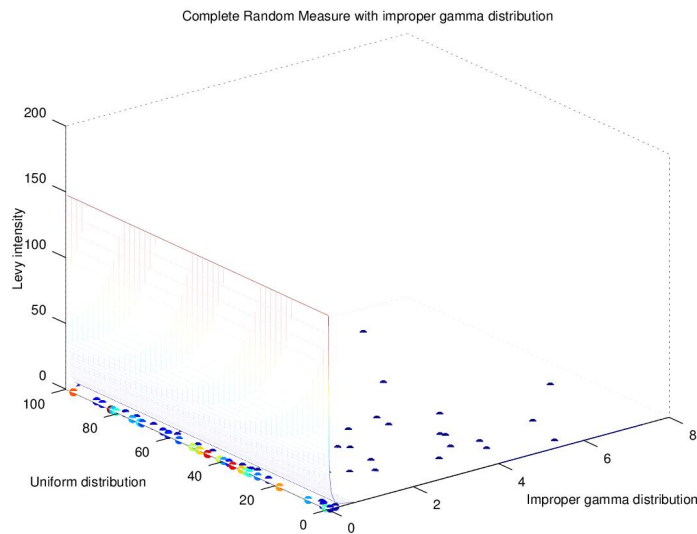


Figure 2.10: A Completely Random Measure with a Lévy intensity defined on the product space $\Omega \otimes \mathbb{R}$. Here Ω is a bounded interval on which the base measure $B_0 = U(0, 100)$ is defined. On \mathbb{R} we define an improper gamma distribution $w^{-1}e^{-\alpha w}$. In this example $\alpha = 1$. This is how a Gamma Process can be generated from a nonhomogeneous spatial Poisson point process. The image is produced by rejection sampling in the same way as Figure 2.8.

The Gamma process can be used as a prior for a counting model (Definition 2.43). This is visualized in (Figure 2.11).

	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	sum
data 0	4	0	3	0	0	0	0	0	0	...	7
data 1	1	1	0	0	1	0	0	0	0	...	3
data 2	2	0	0	0	0	0	0	0	0	...	2
data 3	0	0	0	0	0	4	1	0	0	...	5
data 4	0	0	0	2	0	0	0	0	0	...	2
data 5	0	0	0	0	0	4	0	0	6	...	10
data 6	0	0	0	0	0	4	0	0	0	...	4

Figure 2.11: Matrix representation of a counting model. At the horizontal axis the latent variables (potentially infinite number of features). At the vertical axis the data items. Each data item has one or more features and these features can occur multiple times.

1013 B: Multi-Scoop Indian Buffet Process

1014 Find sequential representation. It is an Indian Buffet Process where the customers can pick
1015 up multiple scoops from the same dish.

1016 (Zhou et al., 2012)

1017 C: Stick-breaking Representation of the Gamma process

1018 The stick-breaking representation of the Gamma process introduces an additional Gamma
1019 distribution that defines the number of times a feature is represented in a data object com-
1020 pared to the Beta process, see Definition 2.54, (Roychowdhury and Kulis, 2015).

▼ Definition 2.54 — *stick-breaking representation of the Gamma process*

The **stick-breaking representation** of the Gamma process states that if

$$\phi_{k,j} \sim GEM(\alpha, 0) \quad (2.34)$$

$$G_{k,j} \sim \text{Gamma}(\alpha + 1, c) \quad (2.35)$$

$$C_k \sim \text{Poisson}(\gamma) \quad (2.36)$$

$$\theta_{k,j} \sim \frac{1}{\gamma} H \quad (2.37)$$

$$G = \sum_{k=1}^{\infty} \sum_{j=1}^{C_k} G_{k,j} \phi_{k,j} \delta(\theta, \theta_{k,j}) \quad (2.38)$$

1021 then $G \sim \text{GamP}(\alpha, H)$.

1022

The Gamma process is used in risk theory (Dufresne et al., 1991), spatial statistics (Wolpert and Ickstadt, 1998; Rao and Teh, 2009), erosion (Singpurwalla, 1997; Abdel-Hameed, 2012), and finance (Madan and Seneta, 1990; K  chler and Tappe, 2008).

2.2.4 Pitman-Yor Process

The Pitman-Yor process (PYP) introduces another parameter d with respect to the Dirichlet process. It has been developed by Pitman and Yor as the two-parameter Poisson-Dirichlet distribution (Pitman and Yor, 1997). The Pitman-Yor process has the following definition.

▼ Definition 2.55 — Pitman-Yor process

A Pitman-Yor process PY over a set S can be used to draw sample paths X :

$$X \sim PY(d, \alpha, H)$$

with $\alpha > -d$ a strength parameter, $0 \leq d < 1$ a discount parameter, and H a measure on S .

The Pitman-Yor process generalizes the Dirichlet process. The Pitman-Yor process has a stick-breaking representation in which sticks are drawn from $GEM(\alpha, \beta)$. The Dirichlet process has a stick-breaking representation in which sticks are drawn from $GEM(\alpha, 0)$, see Def. 2.52.

▼ Definition 2.56

The stick-breaking representation of the PYP states that if

$$\phi_k \sim GEM(\alpha, \beta) \tag{2.39}$$

$$\theta_k \sim H \tag{2.40}$$

$$G = \sum_{k=1}^{\infty} \phi_k \delta(\theta, \theta_k) \tag{2.41}$$

then $G \sim PYP(\alpha, H)$.

The Pitman-Yor process is used in quite a few applications, such as language models (Teh et al., 2006), scene segmentation (Sudderth and Jordan, 2009), speech induction (Blunsom and Cohn, 2011), and time series (Bassetti et al., 2014).

2.2.5 Hierarchical Dirichlet Process

The HDP extends the Dirichlet mixture model with a hierarchical structure (Teh et al., 2006).

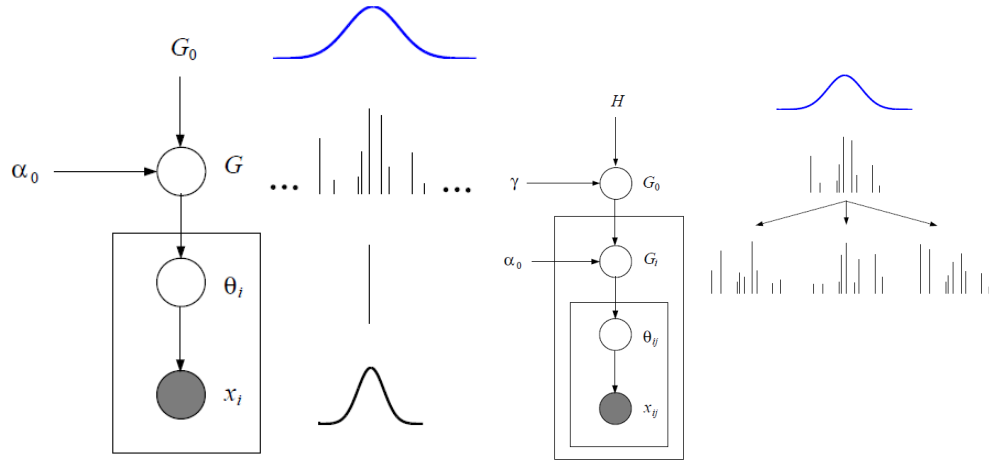
▼ Definition 2.57

A Hierarchical Dirichlet process HDP over a set S can be used to draw sample paths X :

$$G_0 \sim DP(\gamma, H)$$

$$X_i \sim DP(\alpha_0, G_0) \text{ for each group } i$$

with a Dirichlet Process with a general γ dispersion parameter and base distribution H as a measure on S of which the generated distributions G_0 are used as base distribution for each group distribution X_i .



(a) Dirichlet Process Mixture Model. Each draw from the process corresponds to a parameter. Each parameter is associated with a distribution (in this case a Gaussian). The only freedom left to express by G_i is in the weights of those atoms. This reflects a decomposition in a structural and non-structural component.

(b) Hierarchical Dirichlet Process. Observe that the location of the atoms are fixed through the highest layer G_0 . The only freedom left to express by G_i is in the weights of those atoms. This reflects a decomposition in a structural and non-structural component.

Figure 2.12: The difference visualized between a Dirichlet Process mixture and a hierarchical Dirichlet process. It illustrates also that the input of a Dirichlet process does not have to be a continuous function. If it is a continuous distribution it will become a discrete distributed almost surely. If it is a discrete distribution, it will have atoms at the locations where the discrete distribution had its probability mass concentrated.

The hierarchical Dirichlet process uses the outcome of a Dirichlet Process as a starting point to define multiple distributions with atoms at the same locations, while they come equipped with different weights. So, the Dirichlet process on the lower level uses not a continuous distribution as input, but a discrete one, generated by the DP at the top layer. Note, that the Dirichlet Process will *generate* an almost surely (a.s.) discrete distribution, but it can also have a discrete distribution as *prior* H .

2.2.6 Comparison of Random Processes

The Dirichlet process (Section 2.2.1), the Beta process (Section 2.2.2), the Gamma process (Section 2.2.3), the Pitman-Yor process (Section 2.2.4), and the hierarchical Dirichlet process (Section 2.2.5) are not be compared on performance or computationally efficiency. The

processes each represent different assumptions on the model structure. The Dirichlet process is suited for clustering problems where observations have to be assigned to a single class. The Beta process is a good model for applications with combinatorial structure: features that are shared among multiple objects. The Gamma process is a model for applications with counting: features are shared among multiple objects and there is a number of features per object. The Pitman-Yor process fits clustering problems where the distribution over clusters sizes obeys a power law. The hierarchical Dirichlet process is a typical example of a process that can model additional structure within a cluster.

2.3 Inference

There will be six inference methods described, all sampling methods. Inverse transform sampling is described in Section 2.3.1. Rejection sampling in Section 2.3.2. Approximate Bayesian computation in Section 2.3.3. Gibbs sampling in Section 2.3.4. Metropolis-Hastings in Section 2.3.5. Split-Merge MCMC in Section 2.3.6. We rapport for every inference method the corresponding algorithm in pseudo code. We compare the inference methods in Section 2.3.7.

2.3.1 Inverse Transform Sampling

Let $p(x)$ be a discrete probability distribution with two possible values $x = f$ and $x = g$. The probability distribution sums up to one: $\sum_v p(x = v) = 1$. Sample from a uniform distribution $u \sim U(0, 1)$. If $u < p(x = f)$ generate f , else generate g . This procedure samples f with probability $p(x = f)$ and g with probability $p(x = g)$. This can be readily generalized to more than two values by making use of the cumulative distribution function. In Algorithm 1 we sample from $f(x)$ by making use of the inverse cumulative distribution.

Algorithm 1 Inverse transform sampling for $f(x)$

```

1: procedure INVERSE TRANSFORM SAMPLING( $f(x)$ )           ▷ Distribution to sample from.
2:    $F(x) = P(X \leq x) \quad \forall x \in X$                    ▷ Create cumulative distribution function  $F(x)$ .
3:   for  $t = 1 \rightarrow T$  do
4:      $u \sim U(0, 1)$                                      ▷ Sample from uniform distribution.
5:      $x \sim F^{-1}(u)$                                      ▷ Sample  $x$  from (the inverse)  $F^{-1}(x)$ .
6:      $X = X \cup x$ 
7:   end for
8:   return  $X$                                              ▷  $X$  will have the distribution of  $f(x)$ .
9: end procedure

```

The term “inverse” stems from the fact that we return x (or $f(x)$) given u . Inverse transform sampling is a common component in sampling methods. When one of the steps in an algorithm samples from a uniform distribution, it is often an inverse transform sampling step.

1083 2.3.2 Rejection Sampling

1084 Let $f(x)$ be a complicated function from which it is hard to take samples. Let $g(x)$ be a
 1085 simple function that is easy to sample from. Then we can sample from $f(x)$ by making
 1086 sure $Mg(x) \geq f(x)$. The function $Mg(x)$ is an *envelope* function. This sampling method S
 1087 generates the sample set X using $f(x)$ and $g(x)$.

$$X = S(f(x), g(x)) \quad (2.42)$$

1088 The rejection sampling method (Halperin and Burrows, 1960) for $f(x)$ is described in Algo-
 1089 rithm 2.

Algorithm 2 Rejection sampling for $f(x)$

```

1: procedure REJECTION SAMPLING( $f(x), g(x)$ )      ▷ Target and proposal distribution.
2:   for  $t = 1 \rightarrow T$  do
3:      $x^t \sim g(x)$                                 ▷ Generate  $x^t$  from  $g(x)$ 
4:      $u \sim U(0, 1)$                                 ▷ Inverse transform sampling
5:      $p_0 = f(x)/(Mg(x))$ 
6:     if  $u < p_0$  then
7:        $X = X \cup x^t$                                 ▷ Accept
8:     end if
9:   end for
10:  return  $X$                                           ▷  $X$  will have the distribution of  $f(x)$ 
11: end procedure

```

1090 We can use rejection sampling to *sample* from the *posterior* $f(\theta|x)$ given that we know
 1091 the *exact* likelihood function and that we can *sample* from the prior. We know that we can
 1092 sample from the posterior by sampling from $p(\theta)p(x|\theta)$. Moreover, we know that the prior
 1093 $p(\theta)$ necessarily has to be larger than $p(\theta)p(x|\theta)$ for any observation, because $p(x|\theta)$ is a
 1094 probability density function, hence for each x and θ it is smaller than one. Hence we can
 1095 use rejection sampling with $Mg(x) \geq f(x)$ with $M = 1$, $p(\theta) = g(x)$ and $p(x|\theta) = f(x)$.

1096 We introduce the following notation. We make explicit that we need $p(x|\theta)$ for each com-
 1097 bination of observations and parameters, but that we only need to *sample* from the prior,
 1098 which we indicate by a tilde, $\sim p(\theta)$.

$$\Theta = S(\sim p(\theta), p(x|\theta), x) \quad (2.43)$$

Algorithm 3 Rejection sampling for $f(\theta|x)$

```

1: procedure REJECTION SAMPLING( $p(\theta), p(x|\theta), x$ )      ▷ Requires prior, likelihood and
   observations.
2:   for  $t = 1 \rightarrow T$  do
3:      $\theta^t \sim p(\theta)$                                 ▷ Generate  $\theta^t$  from prior
4:      $u \sim U(0, 1)$                                     ▷ Inverse transform sampling
5:      $p_0 = p(x|\theta)$ 
6:     if  $u < p_0$  then
7:        $\Theta = \Theta \cup \theta^t$                             ▷ Accept
8:     end if
9:   end for
10:  return  $\Theta$                                           ▷  $\Theta$  will have the distribution of  $f(\theta|x)$ 
11: end procedure

```

1099 In Algorithm 3 the envelope distribution $p(\theta)$ and the target distribution $p(\theta)p(x|\theta)$, cancel
 1100 in such way that only $p(x|\theta)$ remains.

1101 Most examples illustrate rejection sampling by estimating the area of a circle, but let us
 1102 visualize the method in the context of sampling (Figure 2.13).

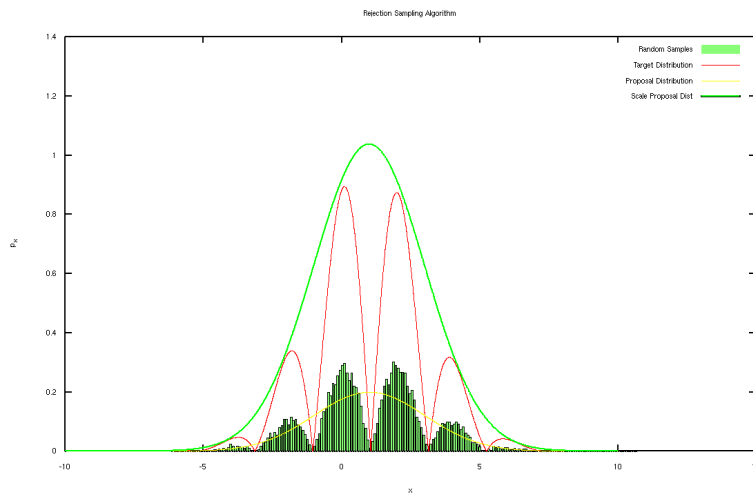


Figure 2.13: A Gaussian is placed over the complex target probability density function. Subsequently the samples that fall in between these two ‘envelopes’ are rejected. This results in a sampling scheme that follows exactly the more complicated probability density function. Note that if the function is scaled by a factor, the sampling scheme stays the same. Such a scaling factor is only important if we want, for example, to know the area under the graph.

1103 2.3.3 Approximate Bayesian Computation

1104 In approximate Bayesian computation (ABC) (Rubin and Others, 1984) the likelihood func-
 1105 tion does not need to be calculated² (Sisson and Fan, 2011). In contrast, it is assumed

²ABC is also called likelihood-free computation

that there is a model available that allows to generate observations given the (searched for) parameters. In ABC for each configuration of parameters a set of observations is generated.

$$\Theta = S(\sim p(\theta), X, \sim M(\theta), d(X^t, X), \epsilon) \quad (2.44)$$

Approximate Bayesian computation uses many tuning parameters. Its most salient characteristic though, is that it generates pseudo-observations through $M(\theta)$ (see Algorithm 4).

Algorithm 4 Approximate Bayesian computation

```

1: procedure APPROXIMATE BAYESIAN COMPUTATION( $p(\theta), X, M, d, \epsilon$ )    ▷ Requires prior,
   observations, model, distance function, and threshold.
2:   for  $t = 1 \rightarrow T$  do
3:      $\theta^t \sim p(\theta)$                                               ▷ Generate  $\theta$  from prior
4:      $X^t \sim M(\theta)$                                               ▷ Simulate observations  $X^t$  from model  $M$ 
5:      $\rho = d(X^t, X)$     ▷ Calculate distance between simulated and actual observations
6:     if  $\rho \leq \epsilon$  then
7:        $\Theta = \Theta \cup \theta^t$     ▷ Accept  $\theta^t$  if distance falls under threshold  $\epsilon$ .
8:     end if
9:   end for
10:  return  $\Theta$     ▷  $\Theta$  will have the distribution of  $f(\theta|X)$ 
11: end procedure

```

The term Bayesian reflects the fact that a prior is involved. The weight of this prior can be manipulated by the threshold ϵ . If this threshold is set very low, the prior plays no role and only observations are taken into account. If ϵ is set extremely high, all θ coming from the prior will be accepted, and the actual observations are not used in the process. There are several disadvantages to approximate Bayesian computation.

- A set of simulated observations has to be compared with the actual observations. This becomes unwieldly if there are many observations.
- It is possible to use summary statistics rather than the observations themselves. If these are sufficient statistics there will be no information loss. If not, there will be information loss in practice.
- The distance function suffers from the curse of dimensionality. In the case that the dimensionality of the individual observations becomes high, or the number of parameters becomes large, it gets increasingly difficult to come up with a distance function which is efficient and accurate at the same time.

2.3.4 Gibbs Sampling

Gibbs sampling (Geman and Geman, 1984) is similar to the *coordinate descent* optimization algorithm (Wright, 2015). In coordinate descent a local minimum of a function is found by iteratively performing a line search along one coordinate direction at a time. Gibbs sampling

1128 optimizes over one variate in the multivariate probability distribution at a time. The update
 1129 value is set and fixed. Then, the next variate is chosen in a round-robin like manner.

$$\Theta = S(X, \sim p(\theta_i | \theta_{-i}, X), \sim p(\theta), B) \quad (2.45)$$

1130 The Gibbs algorithm is given in Algorithm 5. Some explanation on the notation is as follows.
 1131 The multiple parameters in the multivariate probability distribution are denoted by θ . The
 1132 parameters are denoted individually with θ_i . The set of all parameters except for i is denoted
 1133 by θ_{-i} . If we sample a parameter we write θ^t with t the iteration or sampling round. The
 1134 set of parameter samples has capital letter Θ .

Algorithm 5 Gibbs sampling

```

1: procedure GIBBS SAMPLING( $p(\theta_i | \theta_{-i}, X), p(\theta), X, B$ )           ▷ Requires parameters,
   observations and burn-in.
2:    $\theta^0 \sim p(\theta)$                                            ▷ Set parameters to some initial value
3:   for  $t = 1 \rightarrow T$  do
4:     for  $i = 1 \rightarrow k$  do
5:        $\theta_i^t \sim p(\theta_i^{t-1} | \theta_{-i}^t, X)$            ▷ Generate  $\theta_i^t$  from the full conditional probability
6:     end for
7:      $\Theta = \Theta \cup \theta^t$ 
8:   end for
9:    $\Theta_{B:T} \in \Theta$                                            ▷ Get  $\Theta_T$  set, from burn-in  $B$  to end of run  $T$ 
10:   $\Theta \sim \Theta_{B:T}$                                            ▷ Sample  $\Theta$  from correlated  $\Theta_{B:T}$ 
11:  return  $\Theta$ 
12: end procedure

```

1135 Gibbs samples are Markovian. This means that the conditional probability only takes into
 1136 account values at the previous time step $t - 1$. When running the Gibbs sampling algorithm
 1137 long enough, it will visit all possible states eventually. The Markovian property has an un-
 1138 desired side effect. It makes subsequent steps correlated. Hence when finally extracting the
 1139 parameter probabilities, it is important to skip multiple steps to remove the temporal cor-
 1140 relations. It is also important to run the algorithm for a while after its start. In that case it
 1141 does not suffer from a bad choice of initial parameter values. Disregarding the first samples
 1142 is called burn-in. In words, Gibbs sampling works by having the algorithm spend time in
 1143 parts of the space proportionally to the probability of getting into that part of the space.

1144 In the physics literature Gibbs sampling is known as Glauber dynamics or the heat bath al-
 1145 gorithm. First, observe that Gibbs sampling does not necessary require an actual calculation
 1146 of the conditional probability in all cases. The obvious exception is for the observations,
 1147 which are already known. Second, observe that a neat optimization procedure arises when
 1148 conjugate priors are used. A conjugate prior leads to a posterior distribution that can be de-
 1149 scribed analytically. In such a case it is computationally unnecessary to perform sampling.
 1150 It is much faster to use the actual available analytic description. This is commonly called
 1151 collapsed Gibbs sampling.

1152 2.3.5 Metropolis-Hastings Sampling

1153 Metropolis-Hastings (Metropolis et al., 1953) is one of the most well-known Markov chain
 1154 Monte Carlo (MCMC) algorithms. An MCMC algorithm uses a Markov chain (see Gibbs
 1155 sampling, Section 2.3.4) and combines this with a stochastic (Monte Carlo) component.
 1156 This sampling method can be used for high-dimensional distributions. Metropolis-Hastings
 1157 calculates an acceptance factor α which takes into account if a step should be taken according
 1158 to a predefined proposal distribution. In case this step is not accepted, the current sample is
 1159 resampled (see Algorithm 6).

$$\Theta = S(X, \theta^0, Q(\theta^{t+1}|\theta^t), f(\theta, X)) \quad (2.46)$$

1160 Here we need $Q(\theta^{t+1}|\theta^t)$ explicitly as well as samples from it.

Algorithm 6 Metropolis-Hastings sampling

```

1: procedure METROPOLIS-HASTINGS SAMPLING( $\theta^0, X, Q, f$ )  $\triangleright$  Requires initial parameters,
   observations, proposal distribution, and function proportional to desired distribution
2:   for  $t = 1 \rightarrow T$  do
3:      $\theta^{t+1} \sim Q(\theta^{t+1}|\theta^t)$   $\triangleright$  Sample from proposal distribution  $Q$ 
4:      $\alpha = \frac{f(\theta^{t+1}, X^{t+1})Q(\theta^t|\theta^{t+1})}{f(\theta^t, X^t)Q(\theta^{t+1}|\theta^t)}$   $\triangleright$  Calculate acceptance
5:      $u \sim U(0, 1)$   $\triangleright$  Inverse transform sampling
6:     if  $\alpha > u$  then
7:        $\Theta = \Theta \cup \theta^{t+1}$   $\triangleright$  Accept  $\theta^{t+1}$ 
8:     else
9:        $\Theta = \Theta \cup \theta^t$   $\triangleright$  Reuse previous sample (note, different from rejection)
10:    end if
11:  end for
12:  return  $\Theta$   $\triangleright$   $\Theta$  will be samples from the distribution  $f(\theta|x)$ 
13: end procedure

```

1161 A particular choice of a Metropolis-Hastings step is that of a proposal distribution that does
 1162 not depend on the state of the chain. This is already suggested by Hastings and is called the
 1163 independence sampler.

1164 2.3.6 Split-Merge MCMC Sampling

1165 The discussed sampling methods do not assume much structure in the model. This means
 1166 that in hierarchical models sampling either occurs through updating the to-be-estimated
 1167 quantities by iterating over every single observation or over every single cluster. This has
 1168 a disadvantage, the procedure in which a cluster is split into two clusters is very slow by
 1169 moving data points one by one from an old to a new cluster. Much more efficient sampling
 1170 methods can be designed if we would be able to handle large chunks of cluster assignments
 1171 at once.

Split-merge samplers are such methods that can update cluster assignments for multiple observations at once. These samples adjust the acceptance method in the Metropolis-Hastings algorithm. Split-Merge sampling is described in Algorithm 7.

Algorithm 7 Split-Merge MCMC sampling

```

1: procedure SPLIT-MERGE MCMC SAMPLING( $\theta^0, X, Q, f$ )  ▷ Requires initial parameters,
   observations, proposal distribution, and function proportional to desired distribution
2:   for  $t = 1 \rightarrow T$  do
3:      $i \sim D(0, N - 1)$   ▷ Sample observation  $i$  discretely
4:      $j \sim D(0, N - 1)$   ▷ Sample observation  $j$  discretely
5:     if  $c_i == c_j$  then
6:        $c_{old} = c_i$ 
7:        $\theta_{c_{new}}^{t+1} \sim Q(\theta^{t+1} | \theta^t)$   ▷ Sample from proposal distribution  $Q$ 
8:       for  $k \in c_{old}$  do
9:          $c_k \sim C(c_{old}, c_{new})$   ▷ Assign to new cluster categorically
10:      end for
11:     else
12:        $c_{merge} = c_i$ 
13:       for  $k \in c_j$  do
14:          $c_k = c_{merge}$   ▷ Assign all to first cluster
15:       end for
16:     end if
17:      $\alpha = \frac{f(\theta^{t+1}, X^{t+1})Q(\theta^{t+1} | \theta^t)}{f(\theta^t, X^t)Q(\theta^t | \theta^{t+1})}$   ▷ Calculate acceptance
18:      $u \sim U(0, 1)$   ▷ Inverse transform sampling
19:     if  $\alpha > u$  then
20:        $\Theta = \Theta \cup \theta^{t+1}$   ▷ Accept  $\theta^{t+1}$ 
21:     else
22:        $\Theta = \Theta \cup \theta^t$   ▷ Reuse previous sample (note, different from rejection)
23:     end if
24:   end for
25:   return  $\Theta$   ▷  $\Theta$  will be samples from the distribution  $f(\theta | x)$ 
26: end procedure

```

The exact acceptance probability depends on the model. For the mixture model with a Dirichlet Process as prior, its performance is further improved by adjusting the assignment process from random to observation-supported by introducing intermediate restricted Gibbs sampling steps (Jain and Neal, 2004, 2007). Similarly, there are other variants that incorporate data fit to the splitting step. Labels can for example be calculated sequentially (Dahl, 2003) or methods can be used that postulate subcluster structure within clusters to optimize inference over split and merge sets (Chang and Fisher III, 2013).

1182 **2.3.7 Comparison of the Six Inference Methods**

1183 In robotic vision the type of data we are obtaining from depth sensors are point clouds. To
1184 perform inference over objects made out of point clouds, clustering algorithms benefit from
1185 two sampling strategies. If conjugate probability densities are used, Gibbs sampling, or col-
1186 lapsed Gibbs sampling can be used (Section 2.3.4). If the model becomes more complicated
1187 and nonconjugate split-merge sampling will likely accelerate the inference (Section 2.3.6).

1188 **2.4 Chapter Conclusions**

1189 Chapter 3 describes Gibbs sampling to perform inference over an infinite set of lines. Gibbs
1190 sampling requires conditional probabilities. These are given in closed-form because there
1191 is a conjugate description of the line parameters given the points that form the lines in this
1192 application.

1193 Chapter 4 describes Split-Merge MCMC sampling to perform inference over an infinite set
1194 of line segments. The parameters for line segments do not have a conjugate description.
1195 Metropolis-Hastings can be used to perform inference over the line segments, but the search
1196 space is quite large. The Split-Merge MCMC method performs faster inference than Metropolis-
1197 Hastings because it is able to split and merge line segments (with multiple points ascribed
1198 to them) at once.

1199

1200

1201

NONPARAMETRIC BAYESIAN LINE DETECTION

1202 Contents

1203

1204

1205

1206

1207

1208

1209

1210

In this chapter the nonparametric Bayesian models from the literature (Chapter 2) are applied to perform inference over point clouds. The point cloud under study will be a point cloud distributed over lines in a two-dimensional space. Traditionally, RANSAC and the Hough transform have been used to perform inference over such lines. We use a nonparametric Bayesian model to perform inference over a countably infinite number of lines. Given a prior with respect to the noise and the distribution of points over the lines, Bayesian inference describes the optimal procedure to perform line fitting.

1211 Published in

1212

1213

1214

1215

1216

A.C. van Rossum, H.X. Lin, J. Dubbeldam, and H.J. van den Herik. Nonparametric Bayesian Line Detection. *International Conference on Pattern Recognition and Methods*, ICPRAM 2016, Rome, Italy, February 24-26, 2016. Best paper award in theory and methods track.

A.C. van Rossum, H.X. Lin, J. Dubbeldam, and H.J. van den Herik. *Fundamentals of Nonparametric Bayesian Line Detection*. Springer, 2017.

1217 Outline

1218

1219

1220

1221

1222

1223

1224

The infinite line model describes a collection of lines with a Dirichlet process as prior (Sect. 3.1). Inference in the infinite line model is performed through Gibbs sampling (Sect. 3.2). As is known, Gibbs sampling over *parameters* converges slowly, however it can be accelerated through sampling over *clusters* (Sect. 3.3). The results by the inference method are assessed using clustering performance measures (Sect. 3.4). The chapter summarizes the findings (Sect. 3.5) and introduces extensions which will be handled in the next chapters.

1225 In computer vision and particularly in robotics, traditionally the task of line detection has
1226 been performed through sophisticated, but ad-hoc methods. Here we mention two examples

of such methods, RANSAC and the Hough transform. RANSAC (Bolles and Fischler, 1981) is a method that iteratively tests a hypothesis. A line is fitted through a subset of points. Then other points that are in consensus with this line (according to a certain loss function) are added to the subset. This procedure is repeated till a certain performance level is obtained. The Hough transform (Hough, 1962) is a deterministic approach which maps points in the image space to curves in the so-called Hough space of slopes and intercepts. A line is extracted by getting the maximum in the Hough space.

There are four main problems with these methods. First, the extension of RANSAC or Hough to the detection of multiple lines is nontrivial (Zhang and Křsecká, 2007; Gallo et al., 2011; Chen et al., 2001). Second, the noise level is hardcoded into model parameters and it is not possible to incorporate knowledge about the nature of the noise. Third, it is hard to extend the model to hierarchical forms, for example, to lines that form more complicated structures such as squares or volumetric forms. Fourth, there are no results known with respect to any form of optimality of the mentioned algorithms.

In this chapter we postulate a method to perform inference over the number of lines and over the fitting of points on that line using the nonparametric Bayesian methods from chapter 2. The method aims at overcoming the four main problems mentioned above.

3.1 Infinite Line Model

The Dirichlet process has been previously described as prior for a mixture distribution (in Figure 2.6, see Section 2.2.1). It will be used in our model as a prior for the *distribution of points over a countably infinite set of lines*. From now on we will refer to this model as the infinite line model (ILM).

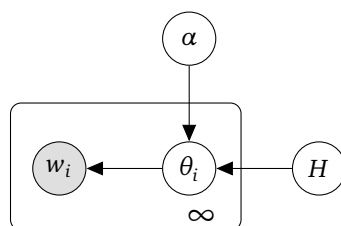


Figure 3.1: The infinite line model using the Chinese restaurant process representation (compare with Figure 2.6). Top: α , the concentration parameter of the Dirichlet process. Bottom, left to right: w_i , the observation, an individual point in a 2D space; θ_i , the parameters (intercept, slope) of the line belonging to observation w_i ; H , the base distribution from which line parameter values are sampled.

The infinite line model is visualized in Figure 3.1 using plate notation (see Section 2.1.5). In Section 3.1.1 it is described how θ_i is sampled from H and α . In Section 3.1.2 it is described how w_i is sampled from θ_i . In Section 3.1.3 the prior H for θ_i is described. In Section 3.1.4 it is described how line parameters θ_i can be updated given the data w_i .

1253 3.1.1 Posterior Predictive for a Line given Other Lines

1254 Let us start reiterating the definition of the Dirichlet process. Let H be a distribution over Θ
 1255 and let α be scalar. The Dirichlet process generates a distribution $G \sim DP(\alpha, H)$:

$$G(\theta_1, \dots, \theta_\infty) \sim DP(\alpha, H(\theta_1, \dots, \theta_\infty)). \quad (3.1)$$

1256 A Dirichlet process assigns a Dirichlet distribution to every parameter partition $\Theta_1, \dots, \Theta_r$:

$$(G(\Theta_1), \dots, G(\Theta_r)) \sim Dir(\alpha H(\Theta_1), \dots, \alpha H(\Theta_r)). \quad (3.2)$$

1257 The Dirichlet is conjugate to the categorical:

$$(G(\Theta_1), \dots, G(\Theta_r)) \mid \theta_1, \dots, \theta_n \sim Dir(\alpha H(\Theta_1) + n_1, \dots, \alpha H(\Theta_r) + n_r),$$

$$n_k = \sum_{j=1}^n \delta_{\theta_j}(\Theta_k). \quad (3.3)$$

1258 To study the details about the conjugacy of the Dirichlet distribution with respect to multi-
 1259 nomial and categorical distributions we refer to the derivations in Appendix B.

1260 In the above notation, $\delta_{\theta_j}(\Theta_k)$ is a Dirac measure (a generalization of the Dirac delta func-
 1261 tion), also known as an indicator function. Given a set Θ_k with a σ -algebra over subsets of
 1262 Θ :

$$\delta_{\theta_j}(\Theta_k) = 1_{\Theta_k}(\theta_j) = \begin{cases} 1 & \text{if } \theta_j \in \Theta_k \\ 0 & \text{if } \theta_j \notin \Theta_k \end{cases}. \quad (3.4)$$

1263 The posterior for the Dirichlet process base distribution and dispersion parameter is a Dirich-
 1264 let process with adjusted parameters:

$$G(\cdot) \mid \theta_1, \dots, \theta_n \sim DP\left(\alpha + n, \frac{\alpha}{\alpha + n} H(\cdot) + \frac{n}{\alpha + n} \frac{\sum_{j=1}^n \delta_{\theta_j}(\cdot)}{n}\right). \quad (3.5)$$

1265 The posterior base distribution G is a weighted average between the prior base distribution
 1266 H and the empirical distribution $n^{-1} \sum_{j=1}^n \delta_{\theta_j}$ with the weights respectively α and n (nor-
 1267 malized). The dispersion parameter α is updated to $\alpha + n$. Note that $\delta_{\theta_j}(\cdot)$ is a distribution,
 1268 the Dirac measure Eq. 3.4.

1269 The posterior predictive for a new parameter θ_{n+1} has the form:

$$P(\theta_{n+1} \in \Theta_k \mid \theta_1, \dots, \theta_n) = \frac{1}{\alpha + n} \left(\alpha H(\Theta_k) + \sum_{j=1}^n \delta_{\theta_j}(\Theta_k) \right). \quad (3.6)$$

In other words, the posterior predictive of θ_{n+1} given the parameters $\theta_1, \dots, \theta_n$ in Eq. 3.6 has exactly the same form as the posterior base distribution G given the parameters $\theta_1, \dots, \theta_n$ (Blackwell and MacQueen, 1973) in Eq. 3.5, namely:

$$\theta_{n+1} \mid \theta_1, \dots, \theta_n \sim \frac{1}{\alpha + n} \left(\alpha H(\theta_{n+1}) + \sum_{j=1}^n \delta(\theta_j - \theta_{n+1}) \right). \quad (3.7)$$

A normal Dirac delta function $\delta(\theta_j - \theta_{n+1})$ can be used here, which is only non-zero when θ_j is equal to θ_{n+1} .

Equivalently, if we describe θ_n conditioned on $\theta_1, \dots, \theta_{n-1}$ we have to run over $n-1$ rather than n parameters:

$$\theta_n \mid \theta_1, \dots, \theta_{n-1} \sim \frac{1}{\alpha + n - 1} \left(\alpha H(\theta_n) + \sum_{j=1}^{n-1} \delta(\theta_j - \theta_n) \right). \quad (3.8)$$

Due to the exchangeability property we can also consider any other parameter update (Neal, 2000):

$$\theta_i \mid \theta_{-i} \sim \frac{1}{\alpha + n - 1} \left(\alpha H(\theta_i) + \sum_{j \neq i} \delta(\theta_j - \theta_i) \right). \quad (3.9)$$

The notation θ_{-i} means every parameter θ except for the one equal to θ_i .

3.1.2 Likelihood of Data given a Line

The likelihood of data given line parameters is defined to be according to the **Bayesian linear regression** model. The Bayesian linear regression model for a single line (Box and Tiao, 2011) assumes a linear relationship between the independent x_i and dependent variables y_i with Gaussian noise added in the y -direction. The individual points i are drawn from a Normal distribution:

$$y_i \sim \mathcal{N}(x_i \beta, \sigma^2). \quad (3.10)$$

The (column) vector β maps the (row) vector with independent variables x_i to the dependent variable y_i . The noise is normally distributed with standard deviation σ along the dimension of the dependent variable.

In a 2D point cloud the point p is represented by (x_p, y_p) . The points are mapped into an intercept-slope representation through defining $X_i = [1 \ x_p]$ and $y_i = y_p$. The vector β will then contain the y -intercept as the first value, the slope as the second value.

1292 All observations that belong to the same single line lead to a likelihood function that corre-
 1293 sponds to a normally distributed random variable with y and X as parameters:

$$p(y | X, \beta, \sigma^2) \propto \sigma^{-n} \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)^T(y - X\beta)\right). \quad (3.11)$$

1294 The dependent variable is now a column vector of values y and each observation has a row
 1295 of independent variables in X . The vector β and the standard deviation σ are shared across
 1296 all observations. The term $y - X\beta$ is written out like this:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} - \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}. \quad (3.12)$$

1297 Note that Eq. 3.11 has exactly the same form for a single point or for multiple points that be-
 1298 long to the same line. Hence, we have the probability of a point w_i given the line parameters
 1299 $\theta_k = (\beta_k, \sigma_k^2)$:

$$F(w_i, \theta_k) = p(w_i | \theta_k) = p(w_i | \beta_k, \sigma_k^2) = p(y_i | X_i, \beta_k, \sigma_k^2). \quad (3.13)$$

1300 To get the full distribution $p(w_i, \beta, \sigma^2)$ we will need also $p(\beta, \sigma^2)$.

1301 3.1.3 Conjugate Prior for a Line

1302 The conjugate prior for the likelihood in Eq. 3.11 is a product of a prior for the standard
 1303 deviation $p(\sigma)$ and the conditional probability of the line coefficients given the standard
 1304 deviation $p(\beta | \sigma^2)$.

$$p(\sigma^2, \beta) = p(\sigma^2)p(\beta | \sigma^2). \quad (3.14)$$

1305 The standard deviation σ is sampled from an Inverse-Gamma (IG) distribution:

$$p(\sigma) \propto (\sigma^2)^{-(\nu_0/2+1)} \exp\left(-\frac{1}{2\sigma^2} \nu_0 s_0^2\right). \quad (3.15)$$

1306 This is an $IG(a_0, b_0)$ with $a_0 = \nu_0/2$ and $b_0 = 1/2 \nu_0 s_0^2$. The conditional with respect to the
 1307 line coefficients has a normal distribution as prior:

$$p(\beta | \sigma^2) \propto \sigma^{-n} \exp\left(-\frac{1}{2\sigma^2}(\beta - \mu_0)^T \Lambda_0(\beta - \mu_0)\right). \quad (3.16)$$

1308 Let us collect $\Lambda_0, \mu_0, a_0, b_0$ into λ_0 , we have now a description of our base distribution H :

$$H(\theta_k) = NIG(\theta_k; \lambda_0). \quad (3.17)$$

1309 The Normal-Inverse-Gamma (NIG) is a distribution that combines a Normal and an Inverse
 1310 Gamma distribution. The line coefficients are sampled from a Normal distribution and the
 1311 standard deviation is sampled from the Gamma distribution with a_0 and b_0 as hyperparam-
 1312 eters.

$$\begin{aligned} \sigma_k &= \tau_k^{-1/2} & \tau_k &\sim \mathcal{G}(a_0, b_0), \\ \mu_k &\sim \mathcal{N}(\mu_0, \sigma^2 \Lambda_0^{-1}). \end{aligned} \quad (3.18)$$

1313 3.1.4 Posterior Predictive for a Line given Data

1314 Due to the fact that the NIG is a conjugate prior with respect to the normal distribution (with
 1315 unknown mean and variance), we have a simplified description for updating the hyperpa-
 1316 rameters, given a set of observations. The hyperparameters are updated¹ according to (c.f.
 1317 Denison, 2002):

$$\begin{aligned} \Lambda_n &= \Lambda_0 + X^T X, \\ \mu_n &= \Lambda_n^{-1}(\Lambda_0 \mu_0 + X^T y), \\ a_n &= a_0 + n/2, \\ b_n &= b_0 + 1/2(y^T y + \mu_0^T \Lambda_0 \mu_0 - \mu_n^T \Lambda_n \mu_n). \end{aligned} \quad (3.19)$$

1318 Let us collect $\Lambda_0, \mu_0, a_0, b_0$ into λ and $\Lambda_n, \mu_n, a_n, b_n$ into λ^* . Let us collect a set of our obser-
 1319 vations and $(X, y)_k$ into w_k . The update for the hyperparameters can then be summarized
 1320 as:

$$\lambda^* = U_{up}(\lambda, w_k). \quad (3.20)$$

1321 Removing observations does lead to similar updates (“downdates”) for the hyperparameters:

$$\begin{aligned} \Lambda_n &= \Lambda_0 - X^T X, \\ \mu_n &= \Lambda_n^{-1}(\Lambda_0 \mu_0 - X^T y), \\ a_n &= a_0 - n/2, \\ b_n &= b_0 - 1/2(y^T y + \mu_n^T \Lambda_n \mu_n - \mu_0^T \Lambda_0 \mu_0). \end{aligned} \quad (3.21)$$

1322 The downdate for the hyperparameters can then be summarized as:

$$\lambda^* = U_{down}(\lambda, w_k). \quad (3.22)$$

¹In comparison with the notation of Denison (2002), we update Λ rather than $V = \Lambda^{-1}$ and subsequently use Λ_n at the right-hand side to simplify the notation for μ_n and b_n .

1323 If we combine this update with sampling θ_k from λ_n according to Eq. 3.17, then we obtain:

$$p(\theta_k | \lambda_0, w_k) \propto F(w_k, \theta_k)H(\theta_k; \lambda_0) = p(\theta_k | \lambda_n) = NIG(\theta_k; \lambda_n). \quad (3.23)$$

1324 Sampling of $NIG(\theta_k; \lambda_n)$ is as in Eq. 3.18, but with λ_n rather than λ_0 .

$$\begin{aligned} \sigma_k &= \tau_k^{-1/2} & \tau_k &\sim \mathcal{G}(a_n, b_n), \\ \mu_k &\sim \mathcal{N}(\mu_n, \sigma^2 \Lambda_n^{-1}). \end{aligned} \quad (3.24)$$

1325 Let us integrate over θ (through the function H):

$$Q(w_k, \lambda_0) = \int_{\Theta} F(w_k, \theta) dH(\theta; \lambda_0). \quad (3.25)$$

1326 3.2 Inference for the Infinite Line Model

1327 The posterior predictive for parameters (see Eq. 3.9) combined with observations w_i is de-
1328 scribed by:

$$p(\theta_i | \theta_{-i}, w_i) \propto r_i H_i(\theta_i) + \sum_{j \neq i} L_{i,j} \delta(\theta_j - \theta_i). \quad (3.26)$$

1329 Eq. 3.26 defines the posterior as proportional (indicated by \propto) to three terms. First, the
1330 α -weighted posterior r_i for a new *cluster*. Second, the posterior with respect to the prior
1331 distribution over the parameter values $H_i(\theta_i)$ given observation w_i . Third, a term that sums
1332 over the likelihood of w_i given existing line θ_j , indicated by $L_{i,j}$.

1333 The α -weighted posterior r_i defines the probability that a new cluster will be sampled:

$$r_i = \alpha Q(w_i, \lambda_0) = \alpha \int_{\Theta} F(w_i, \theta) dH(\theta). \quad (3.27)$$

1334 The posterior $H_i(\theta)$ is the normalized product of the prior distribution $H(\theta_i)$ with the like-
1335 lihood $F(w_i, \theta_i)$ for a single observation w_i .

$$H_i(\theta_i) \propto H(\theta_i) F(w_i, \theta_i). \quad (3.28)$$

1336 The likelihood $F(w_i, \theta_i)$ we will denote $L_{i,i}$ or, in general, for observation w_i and line θ_j :

$$L_{i,j} = F(w_i, \theta_j). \quad (3.29)$$

1337 Sampling a new cluster parameter from $H_i(\theta_i)$ is done with probability:

$$p(\theta_{new}) = \frac{r_i}{r_i + \sum_{j \neq i} L_{i,j}}. \quad (3.30)$$

1338 We can use these equations to derive the parameters θ_i with Gibbs sampling.

Algorithm 8 Gibbs sampling over parameters θ_i

```

1: procedure GIBBS ALGORITHM 1( $w, \lambda_0, \alpha$ )           ▷ Accepts points  $w$ , hyperparameters  $\lambda_0, \alpha$  and
   returns  $k$  line coordinates
2:   for all  $t = 1 : T$  do
3:     for all  $i = 1 : N$  do
4:        $r_i = \alpha Q(w_i, \lambda_0)$            ▷ Posterior predictive of  $w_i$  given hyper parameters (Eq. 3.27)
5:       for all  $j = 1 : N, j \neq i$  do
6:          $L_{i,j} = F(w_i, \theta_j)$            ▷ Likelihood for a line given observation (Eq. 3.29)
7:       end for
8:        $p(\theta_{new}) = \frac{r_i}{r_i + \sum_{j \neq i} L_{i,j}}$    ▷ Probability of sampling a new parameter (Eq. 3.30)
9:        $u \sim U(0, 1)$ 
10:      if  $p(\theta_{new}) > u$  then                               ▷ Sample with probability  $p(\theta_{new})$ 
11:         $\lambda_n = U_{up}(w_i, \lambda_0)$            ▷ Update hyperparameters with  $w_i$  (Eq. 3.20)
12:         $\theta_i \sim NIG(\theta_i; \lambda_n)$            ▷ Sample  $\theta_i$  from NIG (Eq. 3.24)
13:      else
14:         $\theta_i$  sampled from existing clusters           ▷ Sample old cluster
15:      end if
16:    end for
17:  end for
18:  return summary on  $\theta_k$  for  $k$  lines
19: end procedure

```

1339 This Gibbs algorithm is earlier described in its general form (see algorithm 1 in Neal, 2000).
1340 As shown in Algorithm 8 we perform a loop in which for T iterations each θ_i belonging to
1341 observation w_i is updated in succession. The loop consists of four steps. First, the posterior
1342 predictive for w_i given the hyperparameters $p(w_i | \lambda_0)$ is calculated. Second, the likelihood
1343 $L_{i,j}$ for all θ_j given w_i (with $j \neq i$) is calculated. Third, the fraction with r_i defines the
1344 probability for θ_i to be sampled from a new or existing cluster. Fourth, depending on the
1345 probability u , (1) a new cluster is sampled, the hyperparameters are updated with informa-
1346 tion on w_i and thereafter θ is sampled from a Normal-Inverse-Gamma distribution with the
1347 updated hyperparameters, or (2) an existing cluster is sampled.

1348 3.3 Accelerating Inference for the Infinite Line Model

1349 Gibbs sampling of this model might be accelerated. In Figure 3.2 we use plate notation to
1350 show the stick-breaking representation of the infinite line model.

1351 In the previous section we sampled over individual parameters. It is possible to iterate only
1352 over the clusters. The derivation takes a few steps (Neal, 2000) but leads to a simple update
1353 for the component indices that only depends on the number of data items per cluster, the
1354 parameter α , and the available data.

1355 The probability to sample from an existing cluster depends on the number of items in that
1356 cluster (the current data item excluded). This is expressed in equation 3.31.

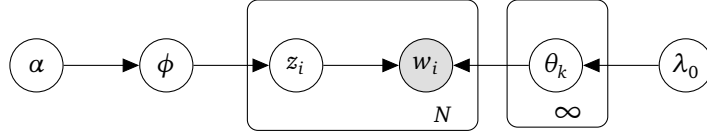


Figure 3.2: The infinite line model in the stick-breaking representation (compare with Figure 3.1). From left to right: α , the concentration parameter of the Dirichlet process; (ϕ_1, \dots, ϕ_k) , the partition of points over lines; z_i , the assignment parameters that link observation w_i with line k ; w_i , the observation, an individual point with x and y coordinates; θ_k , the parameters of line k ; λ_0 , the base measure from which the line parameter values are sampled.

Algorithm 9 Gibbs sampling over clusters c_k

```

1: procedure GIBBS ALGORITHM 2( $w, \lambda_0, \alpha$ )    ▷ Accepts points  $w$  and hyperparameters  $\lambda_0$  and  $\alpha$ ,
   returns  $k$  line coordinates
2:   for all  $t = 1 : T$  do
3:     for all  $i = 1 : N$  do
4:        $c = \text{cluster}(w_i)$                     ▷ Get cluster  $c$  currently assigned to observation  $w_i$ 
5:        $\lambda_c = U_{\text{down}}(w_i, \lambda_c)$           ▷ Adjust cluster hyperparameters on removing  $w_i$  (Eq. 3.22)
6:        $m_c = m_c - 1$                         ▷ Adjust cluster size  $m_c$ 
7:       for all  $k = 1 : K$  do
8:          $L_k = m_k F(w_i, \theta_k)$             ▷ Likelihood for cluster  $k$  given  $w_i$  (Eq. 3.33)
9:       end for
10:       $r_i = Q(w_i, \lambda_0)$                   ▷ Posterior predictive of  $w_i$  given hyper parameters
11:       $p(\text{new}) = \frac{r_i}{r_i + \sum_k L_k}$           ▷ Sample new or old?
12:      if  $p(\text{new})$  then
13:         $\lambda_k = U_{\text{up}}(w_i, \lambda_0)$         ▷ Update hyperparameters with observation  $w_i$ 
14:         $\theta_i \sim \text{NIG}(\lambda)$                 ▷ Sample  $\theta_i$  from NIG
15:      else
16:         $k$  sampled from existing clusters
17:         $\lambda_k = U_{\text{up}}(w_i, \lambda_k)$         ▷ Restore hyperparameters with observation  $w_i$ 
18:      end if
19:       $m_k = m_k + 1$                         ▷ Increment cluster size  $m_k$ 
20:    end for
21:    for all  $k = 1 : K$  do
22:       $\theta_k \sim \text{NIG}(\lambda_k)$                 ▷ Sample  $\theta_k$  from NIG
23:    end for
24:  end for
25:  return summary on  $\theta_k$  for  $k$  lines
26: end procedure

```

$$p(c_i = c \text{ and } c_i = c_j \text{ and } i \neq j \mid c_{-i}, w_i, \alpha, \theta) \propto \frac{n_{c,-i}}{\alpha + n - 1} F(w_i \mid \theta_i). \quad (3.31)$$

1357 The probability to sample a new cluster only depends on α and the total number of data
 1358 items. This is formally described in equation 3.32.

$$p(c_i \in \Omega(c) \text{ and } c_i \neq c_j \text{ and } i \neq j \mid c_{-i}, \alpha) \propto \frac{\alpha}{\alpha + n - 1} \int F(w_i \mid \theta_i) dH(\theta). \quad (3.32)$$

1359 Here $\Omega(c)$ denotes all admitted values for c_i . The importance of conjugacy is obvious from
 1360 Eq. 3.32, it will lead to an analytic form of the integral. The inference method using Eqs. 3.31

and 3.32 is described in Section 3.1.

One benefit of iterating over clusters rather than non-unique parameters is that we can calculate the likelihood by multiplying it with the number of observations at that cluster (rather than per parameter). If we write the number of observations as $n_{c,-i} = m_k$, we can update the likelihood on a cluster level like this:

$$L_k = m_k F(w_i, \theta_k). \quad (3.33)$$

Directly sampling over the clusters is described in its general form (see algorithm 2 in Neal, 2000). Rather than updating each θ_i per observation w_i , an entire cluster θ_k is updated. In Algorithm 8 the update of a cluster would require a first observation to generate a new cluster at θ_j and then moving all observations of the old cluster θ_i to θ_j . In contrast, in Algorithm 9 when a data item either is added or deleted from a cluster, the cluster parameters are updated for all data items in that cluster at once. For this algorithm this means that when w_i is excluded from calculating the likelihood we have to “downdate” the corresponding hyperparameters (as previously mentioned). In Algorithm 9 after all observations have been iterated over and assigned the corresponding cluster k , an outer loop iterates over all clusters to obtain new parameters θ from the NIG prior.

3.4 Performance of the Infinite Line Model

The infinite line model (see Section 3.1) is able to fit an infinite number of lines through a point cloud in two dimensions. These lines are no line segments, but infinite lines. However, to test the model a variable number of lines are generated of a length that is considerably larger compared to the spread caused by the standard deviation of points from that line.

As described earlier, Gibbs sampling leads to correlated samples. Our choice is to get the Maximum A Posterior estimates for the clusters by picking the median values for all the parameters involved. In Section 3.4.1 we discuss the clustering performance and in Section 3.4.2 we provide two clustering examples.

3.4.1 Clustering Performance

The results of the clustering algorithms are measured using conventional metrics. For instance, we may use the Rand Index. It describes the accuracy of cluster assignments (Rand, 1971) by:

$$R = \frac{a + b}{a + b + c + d}. \quad (3.34)$$

Here a counts the pair of points that belong to the same cluster, both at ground truth as well as after the inference procedure. Likewise b numbers the pair of points that belong to

different clusters in both sets. The values c and d describe discrepancies between the ground truth and the results after inference. A Rand Index of one means that there have been no mistakes.

The clustering performance is quite different from the line estimation performance. If the points are not properly assigned, the line will not be estimated correctly. Due to the fact that line estimation has this secondary effect, line estimation performance is not taken into account. Moreover, from lines that generated only a single, or very few points, we can extract point assignments, but line coefficients are impossible to derive. In fact, any derivation would lead to introducing a threshold for the number of points per cluster. Then the performance would need to be measured by weighting the fitting versus the assignment.

The performance of Algorithm 8 can be seen in Figure 3.3 and is rather disappointing. On average the inference procedure agrees upon the ground truth for 75% of the cases considering the Rand Index. Even worse, if we adjust for chance as with the Adjusted Rand Index, the performance would then drop to only having 25% correct cases!

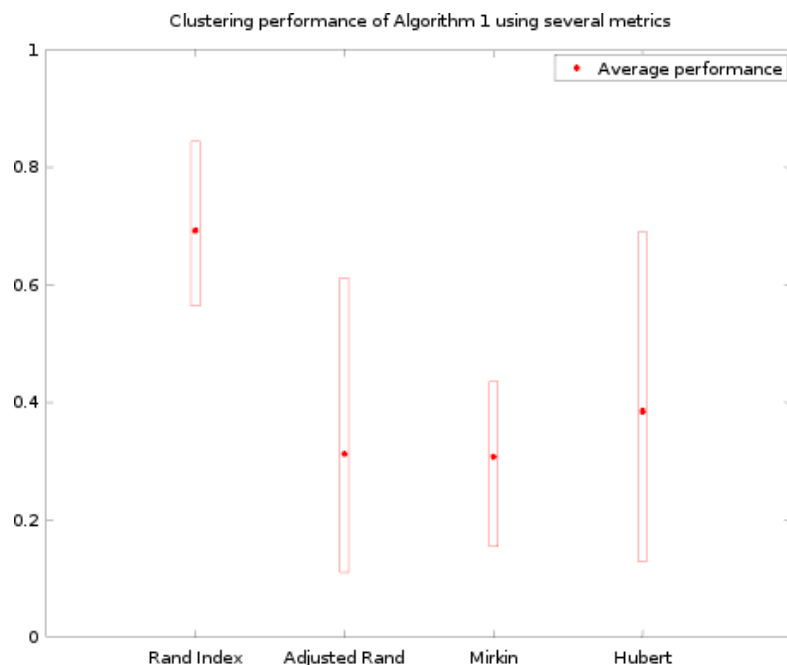


Figure 3.3: The performance of Algorithm 8 with respect to clustering is measured using the Rand Index, the Adjusted Rand Index, the Mirvin metric, and the Hubert metric. A score of 1 means perfect clustering for all metrics, except Mirvin's where 0 denotes perfect clustering.

Algorithm 9 leads to stellar performance measures (Figure 3.4). Apparently, updating entire clusters at once with respect to their parameter values leads at times to perfect clustering, bringing the performance metrics close to their optimal values (see also van Rossum et al., 2016b).

The lack of performance of Algorithm 8 is not only caused by slow mixing. Even when allowing it ten times the number of iterations of Algorithm 8, it does not reach the same performance levels. A line seems to form local regions of high probability, making it difficult for points to postulate slightly changed line coordinates.

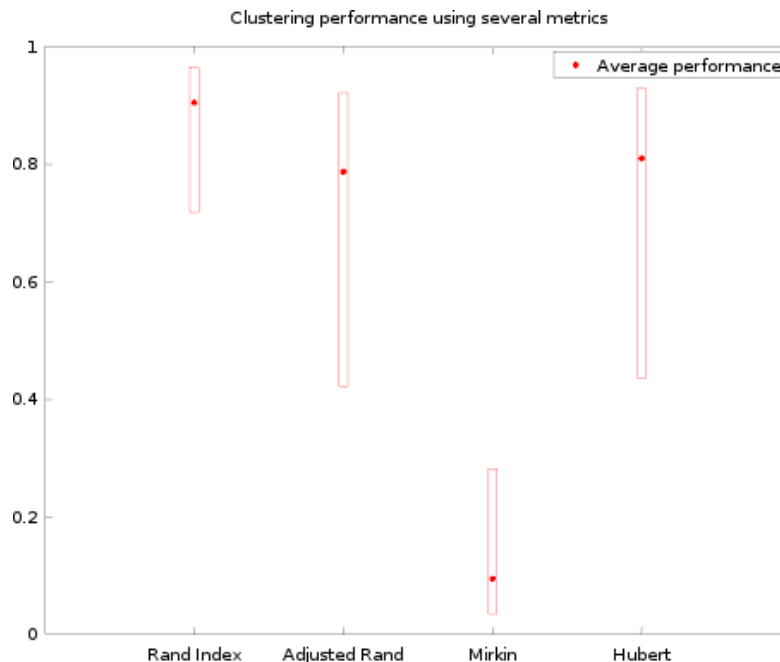


Figure 3.4: The performance of Algorithm 9 with respect to clustering is measured using the Rand Index, the Adjusted Rand Index, the Mirvin metric, and the Hubert metric. A score of 1 means perfect clustering for all metrics, except Mirvin's where 0 denotes perfect clustering.

3.4.2 Two Examples

In the following we show two examples to understand the inference process better. The first example is seen in Figure 3.5. It shows the assignment after a single Gibbs step in Algorithm 8. There is a single line that is represented by two clusters. Algorithm 8 does not have merge or split steps to perform inference about sets of data points, it thus has to move each data point one by one. In passing we mention that there are split-merge algorithms that take these more sophisticated Gibbs steps into account (Jain and Neal, 2004) and we will see these in the following two chapters.

The second example in Figure 3.6 shows that a single point as an outlier is not a problem for our method. A single point might throw off Bayesian linear regression, but because there are multiple lines to be estimated in our Infinite Line Mixture Model, this single point is assigned its own line.

The extension to more points as outliers would, of course, require us to postulate a distribution for these outlier points as well. For instance, a uniform distribution might be used in tandem with the proposed model. However, this would lead to a non-conjugate model and hence it would require different inference methods.

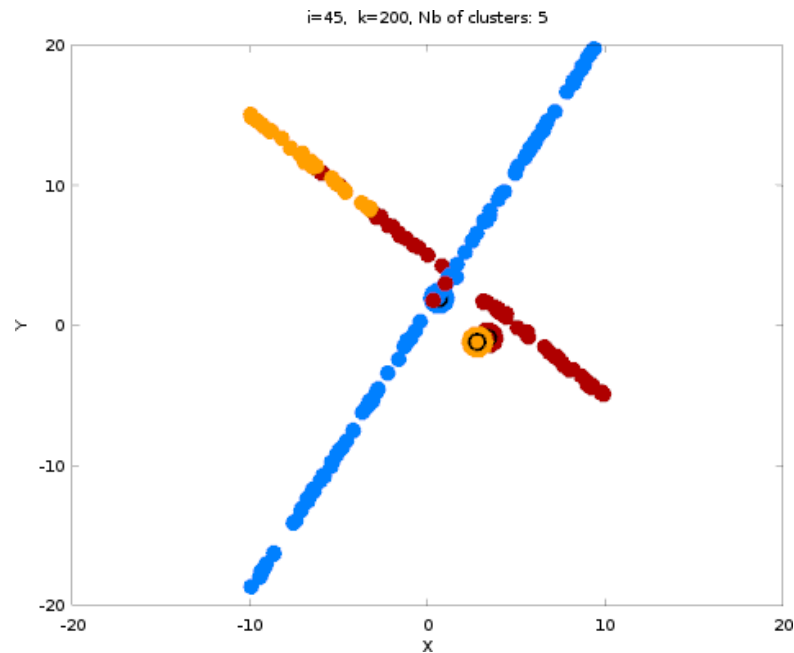


Figure 3.5: One of the Gibbs steps in the inference of two particular lines. The points are roughly distributed according to the lines, but one line exists out of two large clusters. The line coordinates are visualized by a double circle. The x-coordinate is the y-intercept of the line, the y-coordinate is the slope.

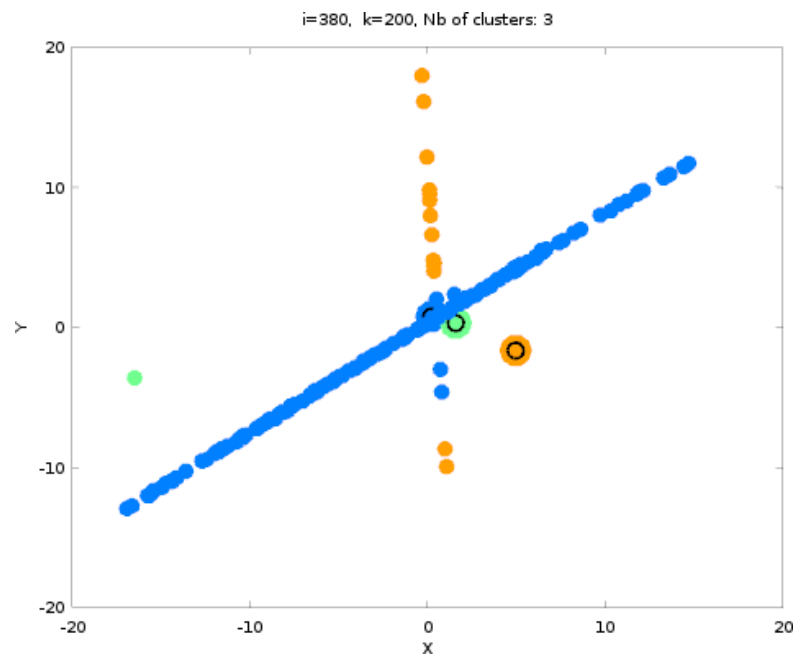


Figure 3.6: The assignment of a line to a single point. In the figure, there are three clusters found, rather than only the two obvious clusters.

1429 3.5 Chapter Conclusions

1430 The infinite line model that is proposed extends the familiar Bayesian linear regression model
 1431 to an infinite number of lines using a Dirichlet Process as prior. The model is a full Bayesian
 1432 method to detect multiple lines. A full Bayesian method, in contrast to ad-hoc methods

such as RANSAC or the Hough transform, means optimal inference (Zellner, 1988) given the model and noise definition.

Results in section 3.4 show high values for different performance metrics for clustering, such as the Rand Index, the Adjusted Rand Index, and other metrics (van Rossum et al., 2016b,a). The Bayesian model is solved through two types of algorithms. Algorithm 8 iterates over all observations and suffers from slow mixing. The individual updates makes it hard to reassign large number of points at the same time. Algorithm 9 iterates over entire clusters. This allows updates for groups of points leading to much faster mixing. Note, that even optimal inference may occasionally result in misclassifications. The dataset is generated by a random process. Hence, occasionally two lines are generated with almost the same slope and intercept. Points on these lines are impossible to assign to the proper line.

The essential contribution of this chapter is the introduction of a fully Bayesian method to infer lines. For such a model, it holds that there are two ways in which it can be extended for full-fledged inference in computer vision as required in robotics. First, the extension of lines in 2D to planes in 3D. This is quite a trivial extension that does not change anything of the model except for the dimension of the data points. Second, somehow a prior needs to be incorporated to limit the lines of infinite length, to line segments. To restrict points on the lines to a uniform distribution of points over a line segment, a symmetric Pareto distribution can be used as prior (see next Chapter). This would subsequently allow for a hierarchical model in which these end points are on their turn part of more complicated objects. Hence, the Infinite Line Mixture Model is an essential step towards the use of Bayesian methods (and thus properly formulated priors) for robotic computer vision.

1455

1456

1457

NONPARAMETRIC BAYESIAN SEGMENT ESTIMATION

Contents

1458

1459

1460

1461

1462

1463

1464

1465

The nonparametric Bayesian model for line estimation (Chapter 3) does not take into account lines that are of finite length. In this chapter, we introduce a Bayesian method to perform inference over such line segments. In this model our prior for the length of the line segment is a symmetric Pareto distribution. Due to the fact that the prior and likelihood do not form a conjugate pair, a more general inference method is used (than the inference methods for the conjugate model in Chapter 3), namely Gibbs sampling with auxiliary variables.

Published in

1466

1467

1468

1469

A.C. van Rossum, H.X. Lin, J. Dubbeldam, and H.J. van den Herik. Non-parametric Segment Detection. *Proceedings of the Eighth European Starting AI Researcher Symposium, STAIRS 2016*, the Hague, the Netherlands, August 26-27, 2016.

Outline

1470

1471

1472

1473

1474

1475

1476

1477

1478

The model is using both a Normal-Inverse-Gamma distribution and a Pareto distribution as priors for an individual line segment (Sect. 4.1.1). The parameters for the line segments are generated through a Dirichlet process (Sect. 4.2). The generative Dirichlet process is used to perform inference using Gibbs sampling over auxiliary variables (Sect. 4.3). The results for inference over line segments are compared with those for lines (Sect. 4.4). Finally, weak aspects of the current MCMC method are established (Sect. 4.5). They will form the basis for new inference methods in the next chapters.

4.1 Pareto Pairs

Lines in a two-dimensional space are mathematical objects that can be described by *two* parameters. To restrict a line to a line segment, a total of *four* parameters are required. Two parametrizations will then come to mind. First, a center-point parametrization, in which parameters describe the center of a line segment, the slope of the line through the center, and the size of the line segment. Second, an endpoint parametrization, in which parameters describe the locations of the two endpoints. The two parametrizations are *equivalent*, but generalizations can either be intuitive or cumbersome. The reason is that the generalization to a line segment from a two-dimensional space to a three-dimensional space, requires the endpoints to be positions in a 3D space. Here we see that the center-point parametrization would require a nonintuitive description of the angles in particular directions. In contrast, the generalization to squares and rectangles or shapes with many endpoints, might benefit from the center-point parametrization.

As far as we know there is no statistical description of data points distributed over a line segment that has a conjugate prior form. A line segment itself, however, has a conjugate form! Assume that we have a prior for the location of endpoints on the x-axis. Given the data, we then update the location of the endpoints. By disregarding the distribution of the data points over the segment, we can update the location of the endpoints by a conjugate Bayesian construction.

4.1.1 Pareto Prior

Assume that the data is distributed according to a symmetric uniform distribution. Hence, the likelihood is given by:

$$p(x | a) \sim \mathcal{U}(-a, a) = \begin{cases} \frac{1}{2a} & \text{for } x \leq |a| \\ 0 & \text{otherwise} \end{cases}. \quad (4.1)$$

Here the uniform distribution is centered around 0 and extends with size a in both directions. It is possible to shift the entire distribution over a distance b . For now, let us continue with one endpoint at a and one endpoint at $-a$.

A prior for the (endpoints of a) symmetric uniform distribution is a symmetric Pareto distribution:

$$p(a) \sim \mathcal{P}_s(\lambda, k) = \begin{cases} \frac{1}{2} k \lambda^k |a|^{-k-1} & |a| \geq \lambda \\ 0 & \text{otherwise} \end{cases}. \quad (4.2)$$

1506 The factor $\frac{1}{2}$ stems from the fact that the symmetric Pareto distribution is now mirrored
 1507 across the y-axis. Hence, the probability density is half of that of the normal Pareto distri-
 1508 bution for the positive x-axis.

1509 If we would just sample from a symmetric Pareto distribution, we can sample multiple times
 1510 from the positive x-axis. To actually sample endpoints of segments we have to sample pairs
 1511 of points.

$$p(a, b) \sim \mathcal{P}_p(\lambda_m, \lambda_n, k) \quad (4.3)$$

1512 This process can be described in two steps. First, we sample a and b from a categorical
 1513 distribution to decide which one will be the left endpoint and which one the right endpoint.
 1514 Second, we sample the right endpoint from a normal Pareto distribution and the left endpoint
 1515 from a mirrored Pareto distribution.

1516 The sampling of Pareto pairs is visualized in Fig. 4.1.

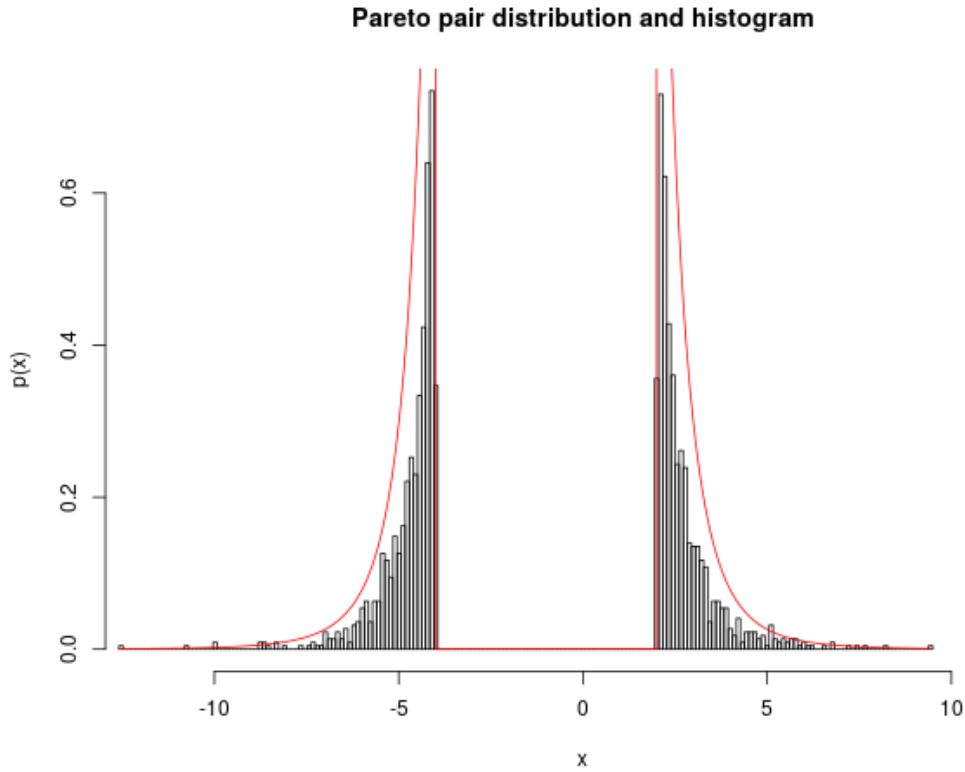


Figure 4.1: Sampling of Pareto pairs. The parameters are $\lambda_m = -4$, $\lambda_n = 2$, $k = 5$, and we have sampled $N = 1000$ pairs. The position parameters λ_m and λ_n define the positions of the endpoints. The shape parameter k defines the variance in the exact positions on both sides. The Pareto pairs always sample endpoints both at the "left" and the "right" (not both at the left or the right side).

1517 4.1.2 Posterior for a Pareto pair

1518 The Pareto distribution is a conjugate prior for the uniform distribution, with updated hy-
 1519 perparameters:

$$p(a \mid D) = \mathcal{P}(c, N + k). \quad (4.4)$$

1520 The data is denoted by $D = \{x_0, \dots, x_{N-1}\}$, the parameter k is adjusted with the number of
 1521 data points N , and the parameter c is the maximum of $\{m, \lambda\}$ with m the maximum value
 1522 in D .

1523 The posterior for a Pareto pair can be found by sampling in parallel for the endpoint at
 1524 the "right" and the one at the "left". The endpoint at the right is sampled from a Pareto
 1525 distribution $\mathcal{P}(c_n, N + k_n)$ with (1) c_n the maximum of the data points D and λ_n , (2) N
 1526 the number of Pareto pairs, and (3) k_n the scale hyperparameter. The endpoint at the left
 1527 is sampled from a Pareto distribution $\mathcal{P}(c_m, N + k_m)$ with (1) c_m the minimum of the data
 1528 points D and λ_m , (2) N the number of Pareto pairs, and k_m the scale hyperparameter.

1529 If $k_n \neq -k_m$ the distribution is shifted such that $k'_n = -k'_m$. This makes the form of the
 1530 probability distribution symmetric with respect to the y axis. In the end, the results are
 1531 shifted back. This transformation makes sense for pairs of points. We do not want the two
 1532 scale parameters of the Pareto distribution to influence the symmetry (the shape) of the
 1533 overall distribution.

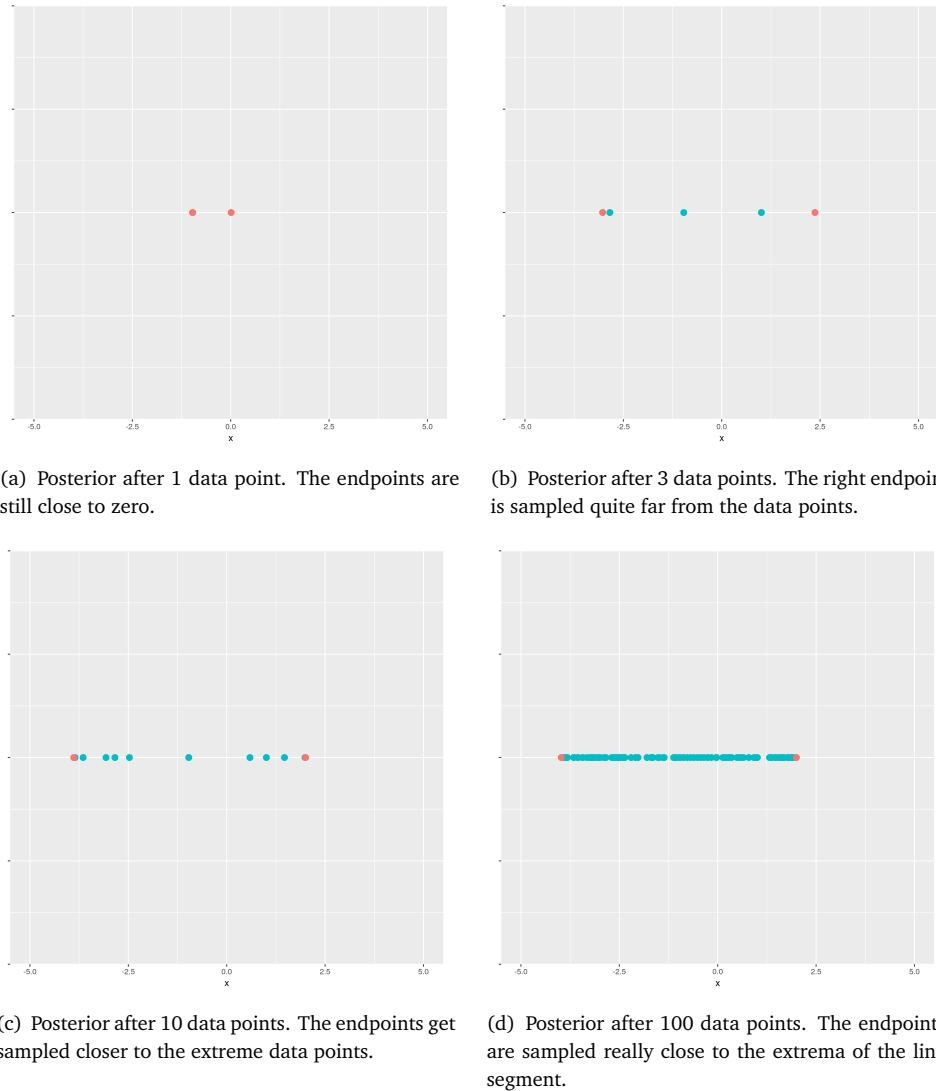


Figure 4.2: Consider (1) the data uniformly distributed on a line segment and (2) a symmetric Pareto prior for both endpoints, then we can update the estimate for the endpoints given the data as visualized. Each subfigure shows an adjustment of the endpoints given more data points (1, 3, 10, and 100 data points). The y-axis does not have a significance in these plots.

1534 Sampling from the Pareto distribution is through inverse transform sampling. By sampling
 1535 from $U(0, 1)$ with 1 included, we transform according to $k/U^{1/a}$.

1536 Figure 4.2 shows how the endpoints are updated given the data. An uninformative prior is
 1537 used. In this case the hyperparameters k_n and k_m are set close to 0, thus the data will wash
 1538 out the prior immediately. Naturally, it is possible to define large k_n and k_m . In that case it
 1539 should be noted that the data will never be able to “correct for” this prior: if the true length
 1540 of the segment is smaller than $|k_n - k_m|$ this will not be inferred properly. Here we note also
 1541 that the maximum and minimum operators are quite sensitive to outliers as well.

4.2 Generative Process to Create a Line Segment

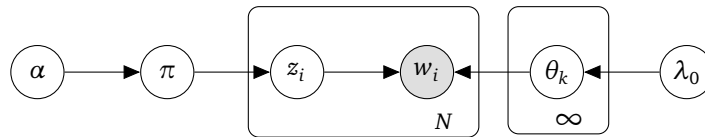


Figure 4.3: The Bayesian linear regression model for multiple line segments in plate notation is the same as for the Infinite Line Model. The Dirichlet process is defined at the left with concentration parameter α . It generates the partitions (π_1, \dots, π_k) with assignment parameters z_i that denote which observation w_i belongs to which cluster k . The cluster is summarized through the parameter set θ_k and has λ_0 as its hyperparameter. The parameter set θ_k includes parameters that signify the line itself such as slope and y-intercept, plus the parameters that denote the extent of the segment.

To be able to perform inference over a line segment in a two-dimensional space, we will have to map somehow these points to a one-dimensional space (see Figure 4.2).

In the case of a *line* we can sample θ_i from a Normal-Inverse-Gamma distribution with hyperparameter λ_{temp} . The update for the hyperparameter we obtain in closed form given observations (described in the previous chapter, Sections 3.1.3 and 3.1.4). However, in the case of a *line segment* there is no known conjugate prior available. Here we will introduce a non-conjugate models to describe line segments. It will make use of a Dirichlet process as prior for the assignment of points towards multiple segments (Section 4.2.1). It will use a uniform distribution (Section 4.2.2) to describe the distribution of points across the line segment.

4.2.1 Dirichlet Process Prior

Let us reiterate the Dirichlet process for our nonparametric line segment model:

$$\begin{aligned} G &\sim DP(\alpha, H), \\ \theta_i &| G \sim G, \\ w_i &| \theta_i \sim F(w_i, \theta_i). \end{aligned} \tag{4.5}$$

The likelihood function F describes the mapping from parameters θ_i to observations w_i . In the previous chapter this has been a likelihood function that describes observations on lines.

4.2.2 Likelihood of Data given Segment Parameters

The likelihood $F(w_i, \theta_i)$ describes the mapping from parameters θ_i to observations w_i . We will again use an intercept-slope representation through defining $w_i = (X_i, y_i)$ with $X_i = [1, x_i]$. The column vector $\beta = [\beta_0, \beta_1]$ contains two parameters: the y-intercept β_0 and the slope parameter β_1 (compare Section 3.1.2).

1562 We assume a normally distributed random variable across $y - X\beta$, the same as in the line
 1563 model (Eq. 3.11):

$$p(y | X, \beta, \sigma^2) \propto \sigma^{-n} \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)^T(y - X\beta)\right). \quad (4.6)$$

1564 However, in contrast to the line model we restrict the variable $X_i = [1 \ x_i]$ to a subset of
 1565 all possible values. The data points on the segment are to be generated from a uniform
 1566 distribution:

$$\begin{aligned} F(y_i, X_i | \beta_i, \sigma_i, a_i, b_i) &= \mathcal{N}(y_i - X_i \beta_i, \sigma_i), \\ X_i &= [1 \ x_i], \\ x_i &\sim \mathcal{U}(a_i, b_i). \end{aligned} \quad (4.7)$$

1567 Fig. 4.4 displays the adjustment with points generated uniformly over the line segment.

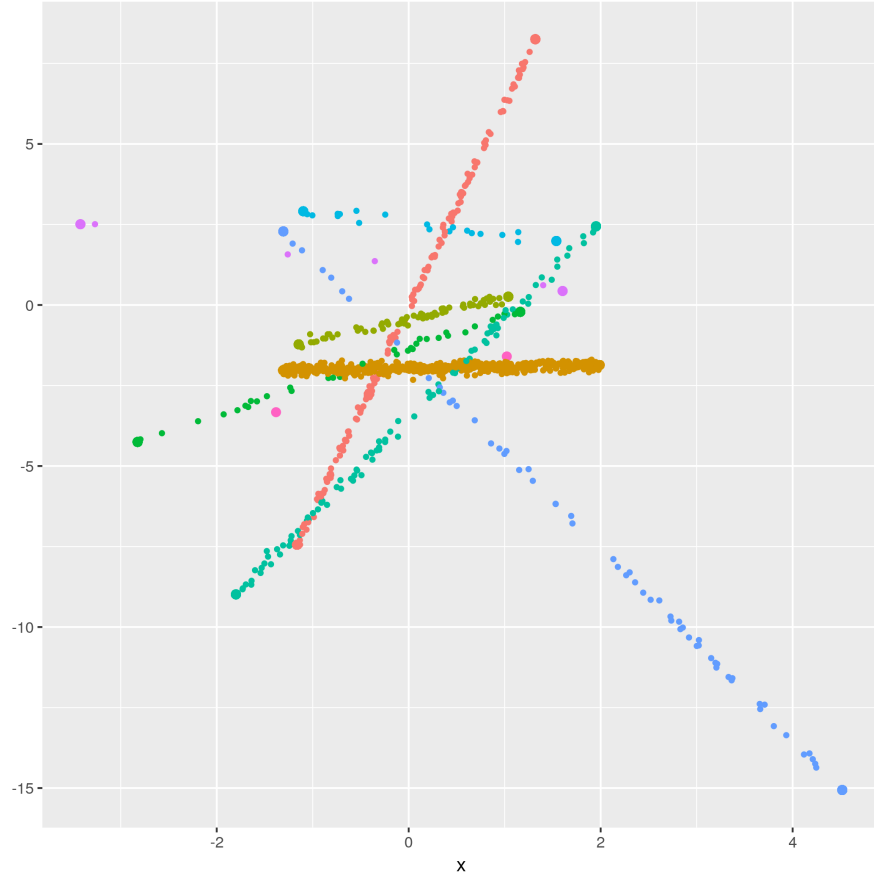


Figure 4.4: Line segments generated through a Dirichlet Process. The line parameters are generated from a Normal-Inverse-Gamma distribution. The endpoints of the line segment are generated from a symmetric Pareto distribution. The points are generated uniformly over the line segments.

The description in Eq. 4.7 is not conjugate. This means that inference over line segments requires more complicated sampling strategies than the inference methods in the previous chapter that require conjugacy.

4.3 Inference over a Line Segment

The line segment model is not conjugate. This therefore requires a sampling algorithm that does not make use of conjugacy. One algorithm that does not assume conjugacy is described in its general form before (Neal, 2000) and is called *Gibbs sampling over auxiliary variables*. The sampling process proposes m new values for the parameters from the hyperparameters. The m values are called auxiliary parameters. Now, to establish to which cluster a certain observation w_i needs to be assigned, the likelihood of each existing and new clusters alike are compared. The weight of an old cluster is defined through the number of data points assigned to it. The weight of a new cluster is defined through α/m .

After every data item is assigned a cluster, the cluster parameters themselves are updated given the assigned data items. In a conjugate model the sufficient statistics can be updated at once, given such observations. In a nonconjugate model we will need to update θ_j by sampling from $p(\theta_j | y)$.

Algorithm 10 Gibbs sampling over auxiliary variables (a θ_i)

```

1: procedure GIBBS ALGORITHM WITH AUXILIARY VARIABLES( $w, \lambda_0, \alpha$ )  $\triangleright$  Accepts points  $w$ ,
   hyperparameters  $\lambda_0, \alpha$ , number of auxiliary variables  $m$ , and returns  $k$  line coordinates
2:   for all  $t = 1 : T$  do
3:     for all  $i = 1 : N$  do
4:       for all  $j = 1 : m$  do
5:          $\theta_j \sim NIG(\lambda_0)$   $\triangleright$  Sample  $\theta_j$  from NIG
6:       end for
7:       for all  $j = 1 : K + m, j \neq i$  do
8:          $L_j = \text{likelihood}(w_i, \theta_j)$   $\triangleright$  Update likelihood for all theta (except  $\theta_i$ ) given
           observation  $w_i$ 
9:       end for
10:       $P_{-i=1:K} = b \sum_{-i} L_{-i}$   $\triangleright$  Calculate probability of existing cluster
11:       $P_{-i=K:K+m} = b\alpha/m L_m L_{-i}$   $\triangleright$  Calculate probability of new cluster
12:       $\theta_i = \theta_j$  according to above  $P_{-i}$   $\triangleright$  Sample  $\theta_i$  accord. to above prob
13:      Remove unused clusters
14:    end for
15:    for all  $j = 1 : K$  do
16:       $\theta_j \sim p(\theta_j | y)$   $\triangleright$  Update  $\theta_j$ 
17:    end for
18:  end for
19:  return summary on  $\theta_k$  for  $k$  line segments
20: end procedure

```

1584 4.4 Results

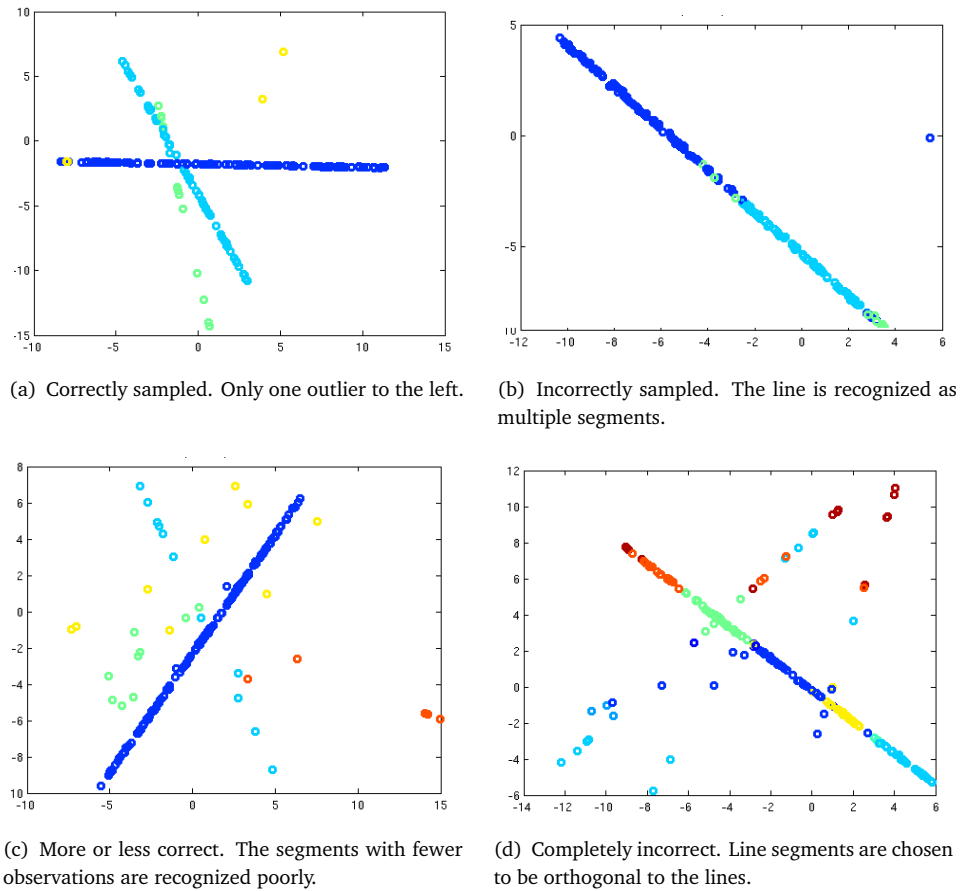


Figure 4.5: Bayesian point estimates of the sampling process with varying outcomes.

1585 There is one phenomenon that is very noticeable in Fig. 4.5. Line segments that form a larger
 1586 line segment are not recognized as such by the inference method.

1587 The results over a larger dataset can be measured with clustering metrics as visualized in
 1588 Fig. 4.6. The Rand Index, Adjusted Rand Index, and Hubert metrics show all reduced per-
 1589 formance compared to line detection where there are no constraints on segment size.

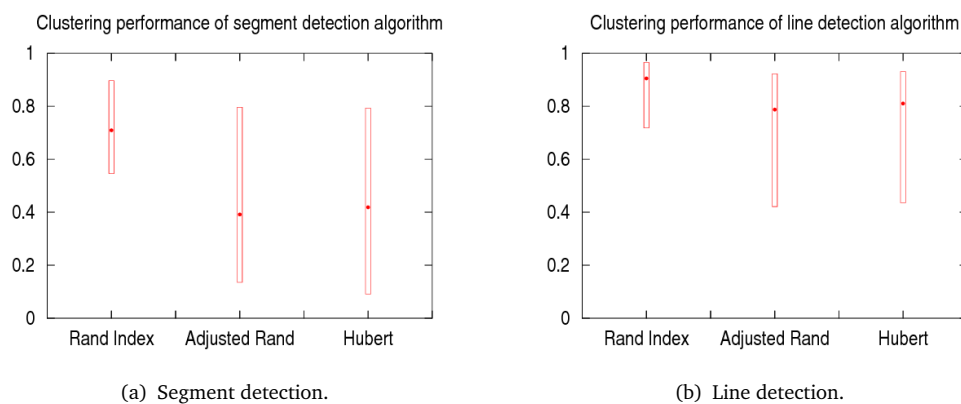


Figure 4.6: Segment detection performs much worse than line detection across all three clustering performance indicators. Perfect clustering is indicated by 1.0 for Rand Index, Adjusted Rand Index, and Hubert.

4.5 Chapter Conclusions

Segment estimation is a much harder problem than line estimation (Chap. 3). In this chapter we used an advanced method, namely MCMC sampling with auxiliary variables to perform inference over an infinite set of line segments. The parameters for line segments do not have a conjugate description. Metropolis-Hastings has been used to perform inference over the line segments, but the search space is quite large. The auxiliary variable MCMC method is indeed faster than ordinary Metropolis-Hastings thanks to postulating multiple new lines than only one.

However, the segment estimation problem is a challenge for the current inference methods. The target probability density has a lot of modes that each needs to be found and are separated by very low probability regions. In Chapter 5 we will introduce new sampling methods that will cope with these challenges.

TRIADIC SPLIT-MERGE SAMPLER

1604	Contents	The nonparametric Bayesian model for line estimation, the infinite line model (Chapter 3) thanks to its conjugate properties has been solved with moderately straightforward sampling methods. The additional constraints that limit lines to line segments (Chapter 4) reduced convergence of the underlying MCMC sampling method (a Gibbs method with auxiliary variables) to sub-par results.
1605		
1606		
1607		
1608		
1609		
1610		This chapter introduces a new sampling method called the triadic split-merge sampler.
1611		
1612	Published in	A.C. van Rossum, H.X. Lin, J. Dubbeldam, and H.J. van den Herik. Triadic Split-Merge Sampler. <i>The 10th International Conference on Machine Vision, ICMV 2017, Vienna, Austria, November 13-November 15, 2017.</i>
1613		
1614		
1615	Outline	The class of split-merge samplers, part of MCMC samplers, are introduced (Sect. 5.1). A conventional split-merge sampler, labeled the dyadic split-merge sampler is detailed (Sect. 5.2). The new split-merge sampler, the triadic split-merge sampler is introduced (Sect. 5.3). The results for inference over lines is compared between the conventional and the new sampler (Sect. 5.4). Finally, although this sampler already improves on the state-of-the-art we see in the chapter conclusions (Sect. 5.5) how we further improve the inference procedure, which will be the basis of the next chapter.
1616		
1617		
1618		
1619		
1620		
1621		
1622		
1623		

5.1 The Class of Split-Merge Samplers

We will consider a Dirichlet process as a prior on the distribution over parameters G . The form of this model is:

$$\begin{aligned}
y_i | \theta_i &\sim F(\theta_i) \\
\theta_i | G &\sim G \\
G &\sim DP(H, \alpha)
\end{aligned} \tag{5.1}$$

5.2 Conventional Split-Merge Sampler

The conventional split-merge sampler Jain and Neal (2004) splits a single cluster into two clusters, and merges two clusters into a single cluster. Hence, this split-merge sampler operates on two clusters at each time step, for which reason we will call it a dyadic split-merge sampler in contrast with our approach.

Algorithm 11 Dyadic split-merge sampler

```

1: procedure DYADIC SPLIT-MERGE SAMPLER( $c$ )                                ▷ Accepts cluster assignments
    $c$  of length  $N$  (besides Metropolis-Hastings acceptance factors  $a(c', c)$  and a split procedure e.g.
   SIMPLERANDOMSPLIT) and returns a (potentially) updated cluster assignment vector  $c'$ .
2:    $i \sim U(1, N)$                                                          ▷ Sample  $i$  random uniformly over cluster assignments.
3:    $j \sim U(1, N) \cap i$                                                  ▷ Sample  $j$  also random uniformly, but with  $j \neq i$ .
4:    $S_R = \{c_i, c_j\}$                                                      ▷ Sampled clusters  $c_i, c_j$ .
5:    $S_I = \{c_x\}$  with  $c_x \in S_R$  for  $x \in \{1, \dots, N\}$                 ▷ All data in clusters  $c_i, c_j$ .
6:    $S_E = S \cap S_R$                                                      ▷ All data in clusters  $c_i, c_j$  excluding  $S_R$ .
7:    $N_S = \text{unique}(S_R)$ 
8:   if  $N_S = 1$  then                                                     ▷ Case:  $i, j$  belong to the same cluster.
9:      $c_i^{(2)} = c_k$  with  $c_k \notin \{c_1, \dots, c_N\}$                     ▷ Sample new cluster for  $c_i^{(2)}$ .
10:     $c_j^{(2)} = c_j^{(1)}$                                                   ▷ Keep  $c_j$  the same.
11:     $c_e^{(2)} = \text{SPLITPROCEDURE}(S_E, c_i^{(2)}, c_j^{(2)})$                 ▷ After  $c_i^{(2)}, c_j^{(2)}$  assign  $S_E$ .
12:    for all  $m \notin S_I$  do
13:       $c_m^{(2)} = c_m^{(1)}$                                                ▷ Data points in clusters other than  $c_i, c_j$  are not adjusted.
14:    end for
15:     $c' = \{c_i^{(2)}, c_j^{(2)}, c_e^{(2)}, c_m^{(2)}\}$ 
16:     $a = a_{\text{split}}(c', c)$  according to Eq. 5.3                        ▷ MH acceptance for a split.
17:  else                                                                 ▷ Case:  $i, j$  belong to different clusters  $c_i \neq c_j$  ( $N_S = 2$ ).
18:    for all  $q \in S_I$  do
19:       $c_q^{(1)} = c_j^{(2)}$                                              ▷ Assign all data points in  $c_i$  and  $c_j$  to  $c_j$ .
20:    end for
21:    for all  $m \notin S_I$  do
22:       $c_m^{(1)} = c_m^{(2)}$                                                ▷ Data points in clusters other than  $c_i, c_j$  are not adjusted.
23:    end for
24:     $c' = \{c_q^{(1)}, c_m^{(1)}\}$ 
25:     $a = a_{\text{merge}}(c', c)$  according to Eq. 5.10                      ▷ MH acceptance for a merge.
26:  end if
27:   $u \sim U(0, 1)$                                                          ▷ Sample  $u$  between 0 or 1 uniformly.
28:  if  $a < u$  then
29:     $c' = c$                                                              ▷ Reject  $c'$  by setting it to  $c$ 
30:  end if
31:  return  $c'$ , the (updated) cluster assignment vector:  $c \rightarrow c'$ .
32: end procedure

```

1632 In algorithm 11 the notation $c_i^{(2)}$ is used to signify that the cluster assignment c_i has 2 clusters
 1633 under consideration. In the dyadic algorithm we could have used c_i^{merge} and c_i^{split} , however
 1634 in the triadic algorithm (see algorithm 14) with multiple split and merge operations the
 1635 latter notation would become confusing.

Algorithm 12 Simple random split

```

1: procedure SIMPLERANDOMSPLIT( $S, c_0, c_1$ )  $\triangleright$  Accepts unassigned set  $S$  and cluster indices  $c_0, c_1$ ,
   returns cluster assignment  $c'_m$ .
2:   for all  $m \in S$  do
3:      $c'_m \sim Cat(c_0, c_1)$  with equiprobable  $p(c_0) = p(c_1) = \frac{1}{2}$ .
4:   end for
5:   return  $c'_m$ , the cluster assignment for  $S$ .
6: end procedure
  
```

1636 The dyadic split-merge sampler in Algorithm 11 samples two distinct data items. If the data
 1637 items belong to the same cluster a split step is attempted. If the data items belong to different
 1638 clusters a merge step is attempted. The split procedure itself is the so-called simple random
 1639 split (Algorithm 12) that assigns data items with the same probability to one of the parts of
 1640 the splitted cluster without consideration for data fit.

1641 5.2.1 Acceptance for the Split Step

1642 The acceptance ratio contains the Metropolis ratio to step from c to c' :

$$\frac{P(c')L(c'|y)}{P(c)L(c|y)} \quad (5.2)$$

1643 Additionally, the Hastings correction is applied because of the asymmetry of the proposal
 1644 distribution in the form of $q(c|c')/q(c'|c)$:

$$a_{split}(c^{(2)}, c^{(1)}) = \min \left[1, \frac{q(c^{(1)}|c^{(2)})}{q(c^{(2)}|c^{(1)})} \frac{P(c^{(2)})}{P(c^{(1)})} \frac{L(c^{(2)}|y)}{L(c^{(1)}|y)} \right] \quad (5.3)$$

1645 The notation $c^{(2)}$ is used to indicate that the cluster index vector is referencing 2 unique
 1646 clusters (in this case after the split step).

1647 The prior distribution is represented by a Chinese Restaurant Process with concentration
 1648 parameter α and no discount factor. Data not yet assigned is assigned with probability
 1649 $\alpha/(n + \alpha)$ to a new cluster and with probability $n_c/(n + \alpha)$ to an existing cluster c . Here n
 1650 are the total number of assigned data points, n_c are the number of data points assigned to
 1651 cluster c . There are D clusters. Hence, the prior over clusters:

$$P(c) = \frac{\Gamma(\alpha)}{\Gamma(\alpha + n)} \alpha^D \prod_{c_l} \Gamma(n_{c_l}) = \alpha^D \frac{\prod_{c_l} (n_{c_l} - 1)!}{\prod_{k=1}^n (\alpha + k - 1)} \quad (5.4)$$

1652 In the prior distribution ratio before and after the split step many of the factors drop out.
 1653 There is one factor α remaining and the number of data points in the splitted cluster is part

of the equation. There is no dependency on other clusters or the total number of data points and we can simplify the formula using the beta function $B(a, b)$:

$$\frac{P(c^{(2)})}{P(c^{(1)})} = \alpha \frac{(n_{c_i^{(2)}} - 1)!(n_{c_j^{(2)}} - 1)!}{(n_{c_i^{(1)}} - 1)!} = \alpha B(n_{c_i^{(2)}}, n_{c_j^{(2)}}) \quad (5.5)$$

The likelihood can be written as:

$$L(c|y) = \prod_{c=1}^D \prod_{k:c_k=c} p(y_k|\phi) \quad (5.6)$$

Here we assume no conjugacy between $F(y_k, \phi)$ and prior distribution $H(\phi)$ and hence write $p(y_k|\phi)$ rather than the conjugate construction $\int F(y_k, \phi) dH_{k,c}(\phi)$ (see Dahl (2005)). The likelihood ratio becomes:

$$\frac{L(c^{(2)}|y)}{L(c^{(1)}|y)} = \frac{\prod_{k:c_k^{(2)}=c_i^{(2)}} p(y_k|\phi) \prod_{k:c_k^{(2)}=c_j^{(2)}} p(y_k|\phi)}{\prod_{k:c_k^{(1)}=c_i^{(1)}} p(y_k|\phi)} \quad (5.7)$$

The split step determines the probability of a particular split. Given that already two data points are assigned to distinct clusters, only the remaining ones have to be assigned with equal probability to $c_i^{(2)}$ and $c_j^{(2)}$:

$$q(c^{(2)}|c^{(1)}) = \left(\frac{1}{2}\right)^{-2+n_{c_i^{(2)}}+n_{c_j^{(2)}}} = \left(\frac{1}{2}\right)^{-2+n_{c_i^{(1)}}} \quad (5.8)$$

The probability of the reverse of the split operation is exactly 1. There is only one way in which a single cluster could have risen from a split cluster, hence:

$$\frac{q(c^{(1)}|c^{(2)})}{q(c^{(2)}|c^{(1)})} = \frac{1}{\left(\frac{1}{2}\right)^{n_{c_i^{(2)}}+n_{c_j^{(2)}}-2}} = 2^{-2+n_{c_i^{(1)}}} \quad (5.9)$$

5.2.2 Acceptance for the Merge Step

Acceptance of a merge step consists of the same components as that of the split step.

$$a_{merge}(c^{(1)}, c^{(2)}) = \min \left[1, \frac{q(c^{(2)}|c^{(1)})}{q(c^{(1)}|c^{(2)})} \frac{P(c^{(1)})}{P(c^{(2)})} \frac{L(c^{(1)}|y)}{L(c^{(2)}|y)} \right] \quad (5.10)$$

$$\frac{P(c^{(1)})}{P(c^{(2)})} = \alpha^{-1} \frac{(n_{c_i^{(1)}} - 1)!}{(n_{c_i^{(2)}} - 1)!(n_{c_j^{(2)}} - 1)!} = \frac{1}{\alpha B(n_{c_i^{(2)}}, n_{c_j^{(2)}})} \quad (5.11)$$

$$\frac{L(c^{(1)}|y)}{L(c^{(2)}|y)} = \frac{\prod_{k:c_k^{(1)}=c_i^{(1)}} p(y_k|\phi)}{\prod_{k:c_k^{(2)}=c_i^{(2)}} p(y_k|\phi) \prod_{k:c_k^{(2)}=c_j^{(2)}} p(y_k|\phi)} \quad (5.12)$$

1669

$$\frac{q(c^{(2)}|c^{(1)})}{q(c^{(1)}|c^{(2)})} = \left(\frac{1}{2}\right)^{-2+n_{c_i^{(1)}}} = 2^{2-n_{c_i^{(1)}}} \quad (5.13)$$

1670 The ratios of the merge step are the inverse of the ratios of the split step.

1671 5.2.3 Sequentially-Allocated Merge-Split Sampler

1672 A variant on the conventional split-merge sampler is the Sequentially Allocated Merge-Split¹
 1673 (SAMS) sampler Dahl (2003). The simple random split procedure of Algorithm 12 is re-
 1674 placed by a procedure that sequentially assigns observations to clusters rather than splitting
 1675 the data random uniformly over the splitted clusters.

Algorithm 13 Sequentially Allocated Merge-Split

```

1: procedure SAMS( $S, c_0, c_1$ )    ▷ Accepts unassigned set  $S$ , cluster indices  $c_i$ , and  $p(y_k|\theta_{c_i})$  with
    $i = 0, 1$ , returns cluster assignment  $c'_m$ .
2:    $T = \text{random\_shuffle}(S)$ 
3:   for all  $m \in T$  do
4:      $p(c_m = c_0 | c_0, c_1, \theta_{c_0}, \theta_{c_1}) = \frac{N_0 p(y_k | \theta_0)}{N_0 p(y_k | \theta_0) + N_1 p(y_k | \theta_1)}$ 
5:      $p(c_m = c_1 | c_0, c_1, \theta_{c_0}, \theta_{c_1}) = 1 - p(c_m = c_0 | c_0, c_1, \theta_{c_0}, \theta_{c_1})$ 
6:      $c'_m \sim p(c_m | c_0, c_1, \theta_{c_0}, \theta_{c_1})$ 
7:   end for
8:   return  $c'_m$ , the cluster assignment for  $S$ .
9: end procedure

```

1676 In contrast to the simple random split, observations y_k are used in the SAMS to obtain cluster
 1677 assignments that correspond with the data rather than cluster assignments independent of
 1678 the data.

1679 5.3 Triadic split-merge sampler

1680 The triadic split-merge sampler uses up to three clusters for a split or merge step (Fig. 5.1).

¹In the naming of split-merge or merge-split samplers, the order of merge split does not bear any significance.

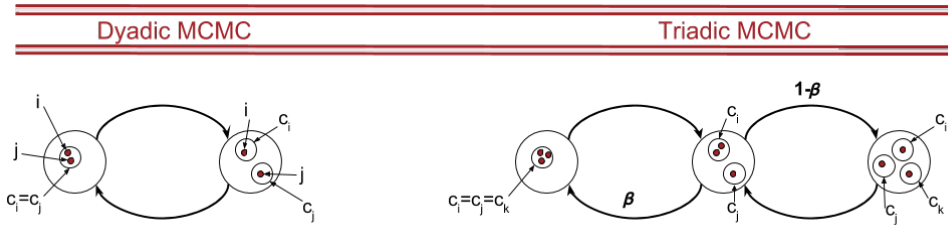


Figure 5.1: Right: dyadic MCMC picks two data items i, j random uniformly. If both are in the same cluster a split towards two clusters is attempted. If both are in distinct clusters a merge towards one cluster is attempted. Left: triadic MCMC picks three data items i, j, k random uniformly. If all three are in the same cluster a split towards two clusters is attempted. If the three items are in two clusters either a split into three (with probability $1 - \beta$) or a merge into a single cluster (with probability β) is attempted. If the three data items are in three distinct clusters a merge is attempted. There are no direct transitions from a single cluster to three clusters or the other way around.

The intuition behind the triadic split-merge sampler is twofold:

- In the dyadic sampler there is a large asymmetry between split and merge steps. There is only one way in which two clusters can be merged into one single cluster, while there are many ways in which one single cluster can be split into two clusters. This asymmetry is reduced by transitioning between two and three clusters. This is a straightforward improvement in balancing split and merge steps (for alternatives, see Wang and Russell (2015)).
- In practical optimization problems it might be useful to form a third cluster out of subsets of two other clusters. The dyadic MCMC sampler requires immediate steps in which (1) one of these clusters is split into two, (2) the other is split into two, and (3) the two new clusters are merged. This means that (a) mixing and hence convergence will be slow and (b) the intermediate steps might have very low probability and function as an unnecessary barrier between high probable states.

Sampling random uniformly for three unique items is implemented through a random shuffle algorithm, in particular the modern version of the Fisher-Yates shuffle introduced by Durstenfeld (1964) and picking the first three items.

5.3.1 Acceptance for the Split Step

In the triadic split-merge sampler there are two splitting steps. It is possible to split according to the dyadic split-merge sampler. However, given two clusters there are (split) jumps to three states as well as (merge) jumps to single states again. To account for this asymmetry another Hastings correction is applied to establish detailed balance.

$$a_{split}(c^{(2)}, c^{(1)}) = \min \left[1, \frac{r(c^{(1)}|c^{(2)})}{r(c^{(2)}|c^{(1)})} \frac{q(c^{(1)}|c^{(2)})}{q(c^{(2)}|c^{(1)})} \frac{P(c^{(2)})}{P(c^{(1)})} \frac{L(c^{(2)}|y)}{L(c^{(1)}|y)} \right] \quad (5.14)$$

Algorithm 14 Triadic split-merge sampler

```

1: procedure TRIADIC SPLIT-MERGE SAMPLER( $c$ )                                ▷ Accepts
   cluster assignments  $c$  of length  $N$  (besides Metropolis-Hastings acceptance factors  $a(c', c)$  and a
   split procedure) and returns a (potentially) updated cluster assignment vector  $c'$ .
2:    $i \sim U(1, N)$                                 ▷ Sample  $i$  random uniformly over cluster assignments.
3:    $j \sim U(1, N) \cap i$                             ▷ Sample  $j$  also random uniformly, but with  $j \neq i$ .
4:    $k \sim U(1, N) \cap \{i, j\}$                         ▷ Sample  $k$  random uniformly, but with  $k \neq j, k \neq i$ .
5:    $S_R = \{c_i, c_j, c_k\}$                                 ▷ Sampled clusters  $c_i, c_j, c_k$ .
6:    $S_I = \{c_x\}$  with  $c_x \in S_R$  for  $x \in \{1, \dots, N\}$         ▷ All data in clusters  $c_i, c_j, c_k$ .
7:    $S_E = S_I \cap S_R$                                 ▷ All data in clusters  $c_i, c_j, c_k$  excluding  $S_R$ .
8:    $N_S = \text{unique}(S_R)$ 
9:    $u \sim U(0, 1)$                                 ▷ Sample  $u$  between 0 or 1 uniformly.
10:  if  $N_S = 1$  then                                ▷ Case:  $i, j, k$  belong to the same cluster.
11:    return  $c' = \text{DYADIC SPLIT-MERGE SAMPLER}(c)$ 
12:  else if  $N_S = 2$  and  $u < \beta$  then                ▷ Case: a cluster with one item and one with two items and
    $u < \beta$ .
13:    return  $c' = \text{DYADIC SPLIT-MERGE SAMPLER}(c)$ 
14:  else if  $N_S = 2$  and  $u \geq \beta$  then                ▷ Case: a cluster with one item and one with two items and
    $u \geq \beta$ .
15:     $c_i^{(3)} = c_k$  with  $c_k \notin \{c_1, \dots, c_N\}$         ▷ Sample new cluster for  $c_i^{(3)}$ .
16:     $c_j^{(3)} = c_j^{(2)}$                                 ▷ Keep  $c_j$  the same.
17:     $c_e^{(3)} = \text{SPLITPROCEDURE}(S_E, c_i^{(3)}, c_j^{(3)})$         ▷ After  $c_i^{(3)}, c_j^{(3)}$  assign  $S_E$ .
18:    for all  $m \notin S_I$  do
19:       $c_m^{(3)} = c_m^{(2)}$                                 ▷ Data points in clusters other than  $c_i, c_j$  are not adjusted.
20:    end for
21:     $c' = \{c_i^{(3)}, c_j^{(3)}, c_e^{(3)}, c_m^{(3)}\}$ 
22:     $a = a_{\text{split}}(c', c)$  according to Eq. 5.14        ▷ MH acceptance for a split.
23:  else                                ▷ Case:  $i, j, k$  belong to three different clusters  $c_i \neq c_j \neq c_k$  ( $N_S = 3$ ).
24:     $S_L = S_I \cap \{c_i^{(3)}, c_j^{(3)}\}$                 ▷ Data in clusters  $c_i, c_j, c_k$  except for  $i$  and  $j$  itself.
25:     $\{c_i^{(2)}, c_j^{(2)}\} = \text{SAMS}(S_L, c_i^{(3)}, c_j^{(3)})$         ▷ Assign data points in  $c_i, c_j, c_k$  to  $c_i, c_j$ .
26:    for all  $m \notin S_L$  do
27:       $c_m^{(2)} = c_m^{(3)}$                                 ▷ Data points in clusters other than  $S_L$  are not adjusted.
28:    end for
29:     $c' = \{c_i^{(2)}, c_j^{(2)}, c_m^{(2)}\}$ 
30:     $a = a_{\text{merge}}(c', c)$  according to Eq. 5.21        ▷ MH acceptance for a merge.
31:  end if
32:   $u \sim U(0, 1)$                                 ▷ Sample  $u$  between 0 or 1 uniformly.
33:  if  $a < u$  then
34:     $c' = c$                                 ▷ Reject  $c'$  by setting it to  $c$ 
35:  end if
36:  return  $c'$ , the (updated) cluster assignment vector:  $c \rightarrow c'$ .
37: end procedure

```

Here we have one additional term compared to the split step from one cluster to two clusters:

$$\frac{r(c^{(1)}|c^{(2)})}{r(c^{(2)}|c^{(1)})} = \frac{\beta}{1} \quad (5.15)$$

The parameter β is free to control, as long as $0 < \beta < 1$ (to maintain ergodicity). The transition from two states to three states is another split step:

$$a_{split}(c^{(3)}, c^{(2)}) = \min \left[1, \frac{r(c^{(2)}|c^{(3)})}{r(c^{(3)}|c^{(2)})} \frac{q(c^{(2)}|c^{(3)})}{q(c^{(3)}|c^{(2)})} \frac{P(c^{(3)})}{P(c^{(2)})} \frac{L(c^{(3)}|y)}{L(c^{(2)}|y)} \right] \quad (5.16)$$

The fraction with r :

$$\frac{r(c^{(2)}|c^{(3)})}{r(c^{(3)}|c^{(2)})} = \frac{1}{1 - \beta} \quad (5.17)$$

The fraction with q uses the total number of data points n_c in the clusters:

$$\frac{q(c^{(2)}|c^{(3)})}{q(c^{(3)}|c^{(2)})} = \frac{\left(\frac{1}{2}\right)^{n_c-2}}{\left(\frac{1}{3}\right)^{n_c-3}} = (3^{n_c-3})(2^{2-n_c}) = \left(\frac{3}{2}\right)^{n_c} \frac{2^2}{3^3} \quad (5.18)$$

To move from 2 clusters to 3 clusters the probability is a $1/3$ for each cluster index in vector c (except for the three data items already selected randomly, hence $n_c - 3$). To move back, the probability is a $1/2$ and there are only two data items randomly assigned beforehand. The fraction with P uses the number of data points in each of the clusters before and after the step:

$$\frac{P(c^{(3)})}{P(c^{(2)})} = \alpha \frac{(n_{c_i^{(3)}} - 1)!(n_{c_j^{(3)}} - 1)!(n_{c_k^{(3)}} - 1)!}{(n_{c_i^{(2)}} - 1)!(n_{c_j^{(2)}} - 1)!} = \alpha \frac{B(n_{c_i^{(3)}}, n_{c_j^{(3)}}, n_{c_k^{(3)}})}{B(n_{c_i^{(2)}}, n_{c_j^{(2)}})} \quad (5.19)$$

Here we introduced a generalized Beta function $B(a, b, c) = \Gamma(a)\Gamma(b)\Gamma(c)/\Gamma(a + b + c)$ with $\Gamma(x) = (x - 1)!$ the Gamma function. The likelihood ratio becomes:

$$\frac{L(c^{(3)}|y)}{L(c^{(2)}|y)} = \frac{\prod_{m:c_m^{(3)}=c_i^{(3)}} P(y_m|\phi) \prod_{m:c_m^{(3)}=c_j^{(3)}} P(y_m|\phi) \prod_{m:c_m^{(3)}=c_k^{(3)}} P(y_m|\phi)}{\prod_{m:c_m^{(2)}=c_i^{(2)}} P(y_m|\phi) \prod_{m:c_m^{(2)}=c_j^{(2)}} P(y_m|\phi)} \quad (5.20)$$

5.3.2 Acceptance for the Merge Step

The merge step from two to one cluster is analogous to the split step:

$$a_{merge}(c^{(1)}, c^{(2)}) = \min \left[1, \frac{r(c^{(2)}|c^{(1)})}{r(c^{(1)}|c^{(2)})} \frac{q(c^{(2)}|c^{(1)})}{q(c^{(1)}|c^{(2)})} \frac{P(c^{(1)})}{P(c^{(2)})} \frac{L(c^{(1)}|y)}{L(c^{(2)}|y)} \right] \quad (5.21)$$

The merge step from three clusters to two clusters is:

$$a_{merge}(c^{(2)}, c^{(3)}) = \min \left[1, \frac{r(c^{(3)}|c^{(2)})}{r(c^{(2)}|c^{(3)})} \frac{q(c^{(3)}|c^{(2)})}{q(c^{(2)}|c^{(3)})} \frac{P(c^{(2)})}{P(c^{(3)})} \frac{L(c^{(2)}|y)}{L(c^{(3)}|y)} \right] \quad (5.22)$$

Note that all the fractions in Eq. 5.22 are the reverse of the fractions in Eq. 5.16. Inverting Eq. 5.17–5.20 will be left to the reader.

One additional issue we have to consider. When merging three clusters into two we can (1) distribute the data over all three clusters or (2) alternatively, keep the data in two clusters assigned to these clusters and only distribute the data in the third cluster over the other two clusters. The second and alternative option however would introduce unnecessary asymmetry with the merge step. In other words, Eq. 5.23 is not the inverse of Eq. 5.18. In contrast, the equation is similar to splitting one cluster across two as in Eq. 5.9:

$$\frac{q_{alt}(c^{(3)}|c^{(2)})}{q_{alt}(c^{(2)}|c^{(3)})} = 2^{-2+n_c} \quad (5.23)$$

Hence the first option is entertained and the q -fraction is exactly the inverse of Eq. 5.18.

Another choice has been made, namely to exclude direct operations between a single cluster and three clusters. This is because factors like:

$$\frac{P(c^{(3)})}{P(c^{(1)})} = \alpha^2 \frac{(n_{c_i^{(3)}} - 1)!(n_{c_j^{(3)}} - 1)!(n_{c_k^{(3)}} - 1)!}{(n_{c_i^{(1)}} - 1)!} \quad (5.24)$$

become very small and although compensated by a large q fraction, remain further away from an acceptance factor of 1. Note that by the ability to split a single cluster into two and then into three, there is no ergodic argument to introduce also the immediate step.

5.4 Results

The problem we use to test our sampler is a well-known problem in computer vision, namely that of the inference of line parameters (slope and intercept) given data points. Rather than ordinary linear regression, in computer vision there is a mixture of lines that have to be estimated. Moreover, the number of lines is not known beforehand. To solve this problem we use the Dirichlet process mixture (Eq. 5.1) with a normal distribution $N(0, \sigma_0)$ to generate the line parameters and a likelihood function that defines points to be uniformly distributed across a line of length 20 and deviating from the line according to a normal distribution $N(0, \sigma_1)$.

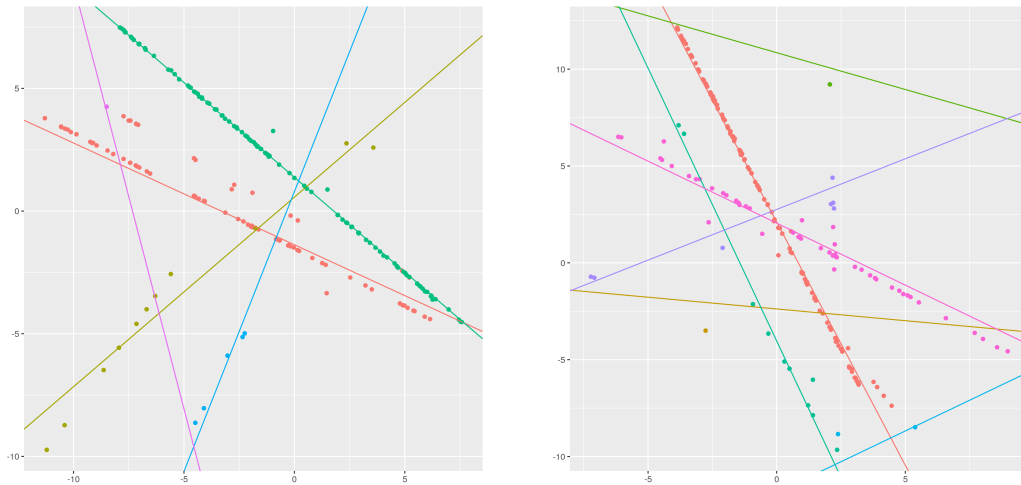


Figure 5.2: Two examples of fitting a mixture of lines to data items scattered over a two-dimensional space. The lines drawn are inferred using one of the methods in this chapter. The lines are not the ground truth, but are meant to demonstrate the typical errors made by fitting methods. Note for example that there are mistakes in both the assignment of points to lines as well as the line parameters (slope and intercept).

5.4.1 Implementation

The sampler is open-source² implemented in C++ which means that (a) it is computationally fast, (b) it can be run on embedded devices if a cross-compiler is available and the Eigen3 library is ported. Note, that due to the fact that the simulator uses a lot of random numbers the system should use a modern compiler (g++-6 or newer) and should have enough entropy available³. Rather than a random scan, the implementation uses a fixed scan as advocated in the literature MacEachern (2007).

To speed up the sampler most calculations are done in log-space. Consider $v = u + 1$. The ratio with probabilities (Eq. 5.5 and 5.19) becomes:

$$\log \frac{P(c^{(v)})}{P(c^{(u)})} = \log(\alpha) + \sum_i \log \Gamma(n_{c_i^{(v)}}) - \sum_i \log \Gamma(n_{c_i^{(u)}}) \quad (5.25)$$

The fraction with $q(\cdot)$ (Eq. 5.9 and 5.18) becomes:

$$\log \frac{q(c^{(v-1)}|c^{(v)})}{q(c^{(v)}|c^{(v-1)})} = (v - n_c - 1) \log(v - 1) - (v - n_c) \log(v) \quad (5.26)$$

The fraction with r becomes for example (Eq. 5.17):

$$\log \frac{r(c^{(2)}|c^{(3)})}{r(c^{(3)}|c^{(2)})} = -\log(1 - \beta) \quad (5.27)$$

²Code can be found at <https://code.annevanrossum.nl/noparama>.

³On Linux this can be checked in `/proc/sys/kernel/random/entropy_avail`.

The log-probability to calculate the likelihood given by a multivariate Normal distribution is well-known.

5.4.2 Comparison

The Triadic sampler using SAMS is compared with the Jain-Neal Dyadic sampler using SAMS and an auxiliary variable sampler with $m = 3$ (see algorithm 8 in Neal (2000)).

Method	Purity	Rand Index	Adjusted Rand Index
Dyadic sampler	0.80960	0.80580	0.56382
Auxiliary variables	0.87235	0.85879	0.68224
Triadic sampler	0.86405	0.87188	0.71067

Table 5.1: The purity, rand index, and adjusted rand index establishing the quality of the clustering method. The closer the values to one, the better the method performed. The purity metric assigns high values to clusters that do not have data points from other clusters (but does not penalize the number of clusters). The rand index computes similarity between clusters taking false negatives and false positives into account. The adjusted rand index accounts for chance. The adjusted rand index is most useful in our comparison.

In Table 5.1 the line estimation problem is compared for the dyadic sampler, an auxiliary variables sampler, and the proposed triadic sampler. The simulation is run with $\beta = 0.1$ so that a significant number of steps are tried between two and three clusters (rather than only between one and two clusters).

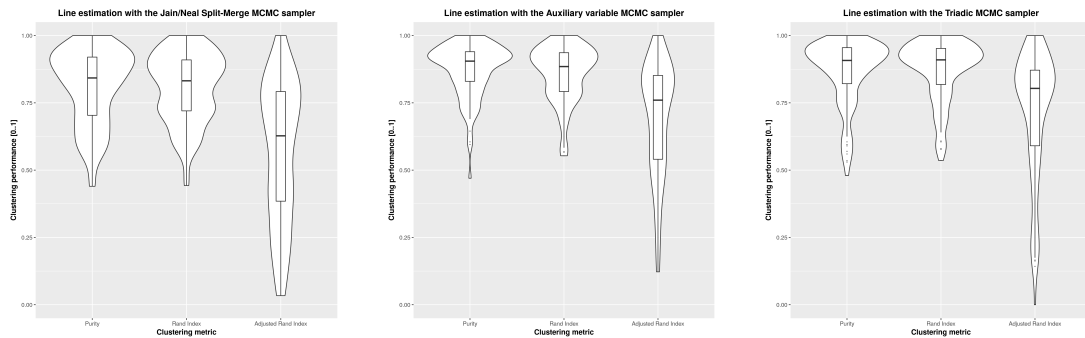


Figure 5.3: The same results as in Table 5.1, but visualized in a violin plot. The distribution over metric values are displayed in a vertical fashion. From left to right the distribution shifts to one, signifying better clustering performance.

In Fig. 5.3 the different metrics are visualized in the form of violin plots. The improvement in clustering is especially visible with the adjusted rand index.

5.5 Chapter Conclusions

A new split-merge sampler has been introduced, implemented, and applied to the computer vision problem of line estimation. The sampler outperforms existing samplers, such as the ordinary (dyadic) split-merge sampler Jain and Neal (2004) and auxiliary variable sampler Neal (2000).

Although the proposed split-merge sampler is able to mix considerably faster through a mixture model, it does not use global jumps directly based on the data. It is reasonable to suggest that MCMC methods benefit from combining the local jumps with global jumps, for example by a mixture of the local Metropolis-Hastings sampler with a Metropolized independence sampler Jampani et al. (2015). We will introduce such a sampler in chapter 6.

ADVERSARIALLY TRAINED MCMC KERNELS

Contents To use MCMC for volumetric inference it is necessary to be able to accelerate the algorithms even further. Volumetric objects exhibit more structure, which is reflected by symmetry.

Outline We describe MCMC methods that cope with symmetric objects.

6.1 Data-Driven Inference

There are three aspects we would like to address in our inference engine.

The first aspect aims to have structure within our inference engine. The proposal distribution in a Markov chain, although moderately complex in the previous chapter, does not have much knowledge about the model. An artificial border is maintained that does not allow the inference engine to have knowledge about the model. The purpose of this is never articulated in particular. However, it is logical from a separation of concern. Such an inference engine (1) does not need to receive any information about the model and (2) is guaranteed to be general in the sense that it is not tailored to a particular model. This is nicely articulated by Tran et al. (2017) from which we quote.

“Many existing probabilistic programming languages treat the inference engine as a black box, abstracted away from the model. These cannot capture probabilistic inferences that reuse the model’s representation - a key idea in recent advances in variational inference, generative adversarial networks, and also in more classic inference.

The second aspect concerns the data. In MCMC the position for the chain is driven by (1) the prior, (2) the prior and the likelihood, (3) a sequence of priors and likelihood, (4) a

sequence of priors, likelihood and proposal distributions, basically anything, except for the data itself. Data-driven approaches would namely destroy the convergence of the Markov chain. To start an MCMC sampler in a data-driven manner and continue in a data-oblivious manner is a possible solution (Zhang and Perez-Cruz, 2017). Even better, it is possible to use a Metropolized independence sampler Jampani et al. (2015). Such a sampler samples independently from the previous state and uses global information. However, to work well its proposal distribution needs to match the target distribution quite well. Although, when combined with a local sampler, it might be sufficient to just be able to match the modes of the target distribution well.

The third aspect concerns the way we build our MCMC engine. The split-merge sampler of the previous chapter has been meticulously designed. If we admit a data-driven approach, we might as well adjust our MCMC engine using training samples. Note, that this training will be across a set of line, box, or scenes mixtures. The MCMC engine will not be able to learn just the parameters of a particular visual object. It will learn how to jump around (optionally, adaptively) from one visual object to the next or from one cluster configuration to the next. In other words, it will be able to teach itself to become a Triadic Split-Merge sampler if that happens to be a good engine. Is it possible to constrain the search through MCMC kernels such that its result is always converging in an MCMC sense? If we aim to learn the transition operator of our Markov chains, there is new literature that makes use of deep nets.

6.2 Learning the Transition Operator

There are multiple methods that can be used in a generative setting. We will discuss the three most prominent ones: (1) generative adversarial networks, (2) variational autoencoders, and (3) infusion training. This is far from an extensive categorization, worth studying are variational walkback (Goyal et al., 2017), stacked generative adversarial networks (Huang et al., 2016), generative latent optimization (Bojanowski et al., 2017), deep learning through the use of non-equilibrium thermodynamics (Sohl-Dickstein et al., 2015), denoising autoencoders, or generative stochastic networks, to name just a few.

6.2.1 Adversarial Training

Adversarial training has been extensively studied since the article on generative adversarial networks by Goodfellow et al. (2014). A particular adversarial setup for training an MCMC has been suggested as well (Song et al., 2017). The generator samples from a Markov chain. A discriminator subsequently needs to judge if its incoming data comes from the generator or if it is sampled from the actual data set. To start the process, the generator can run the chain from the model as well as from the data.

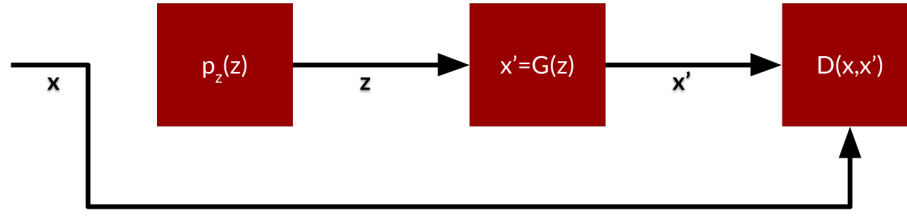


Figure 6.1: Left: $p_z(z)$ is a (prior) random distribution that generates random variables z . Middle: the generator G maps the random variables z to simulated data points x' . Right: the discriminator $D(x, x')$ compares the simulated data x' with the real data x . The generator tries to generate samples in such way that the discriminator has difficulties distinguishing the simulated from the real data.

6.2.2 Variational Autoencoders

Variational autoencoders (Kingma and Welling, 2013; Rezende et al., 2014) are ordinary autoencoders with additional constraints on the latent variables. The latent variables in autoencoder parlance are called the code. In a variational autoencoder the latent variables are forced to approximately describe a unit Gaussian distribution. The autoencoder is trained using a loss function that is composed out of (1) a generative loss, a mean squared error that measures how accurately the network reconstructs its input, and (2) a latent loss, a KL-divergence that measures how closely the latent variables match a unit Gaussian. To optimize the KL divergence a reparameterization trick is applied. The encoder does not generate a vector with real values, but generates a vector with means and standard deviations instead.

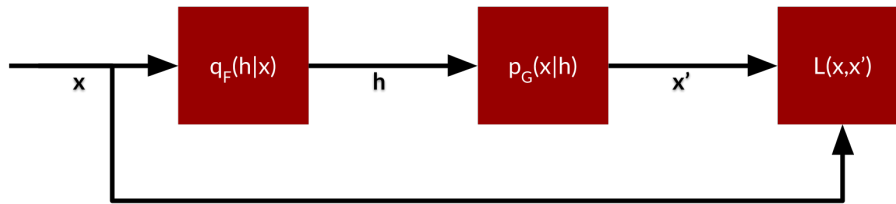


Figure 6.2: Left: $q_F(h|x)$ maps the data x to (hidden) random variables h . Middle: $p_G(x|h)$ maps the hidden random variables to reconstructed data x' . Right: $L(x, x')$ measures the similarity between x and x' .

6.2.3 Infusion Training

The transition operator can also be learned directly through infusion training. In infusion training we gradually adjust totally unstructured noise to a target distribution as well. In this method a particular data point is ‘infused’ into the Markov chain to bias the model sampling to move towards this particular data point (and not another). In contrast to a generative loss that is a mean squared error, this promises to have less blurry reconstructions.

6.3 Volumetric Models

In image processing autoencoders have been used for 2D shape recognition. To apply the same type of models to 3D point clouds, these point clouds are represented through voxels. The application of data-driven deep learning techniques, be it autoencoders, generative adversarial networks, or adversarial autoencoders promises similar good results in these 3D settings than in the current computer vision tasks.

The 3D ShapeNet model (Wu et al., 2015) exists of 3D voxel input that is piped through several stages with an increasing number of filters. The used voxel representation is a binary tensor. It assigns a value of 1 to each voxel that is inside the 3D object mesh and a value 0 to each voxel outside the mesh (empty space). The voxel sizes are fixed as well as the grid size (in this particular model the grid exists of 30x30x30 voxels). The inference model is a Deep Belief Network (DBN). Convolution operators, in the form of filters over small neighbourhoods, are used to reduce the number of model parameters (30x30x30 fully connected would be really many weights). The DBN is used in a supervised setting where shapes are trained with object labels. The model subsequently learns to generate shapes given an object label.

An unsupervised method in the form of a convolutional (volumetric) autoencoder (Sharma et al., 2016) has been applied to the same type of data. This (denoising) autoencoder, coined VConv-DAE maps from an entire voxel grid to another voxel grid. This work uses a combination of standard techniques, a dropout layer, a deconvolution layer, ReLu as well as sigmoid activation, but it is not in particular tailored to 3D point clouds.

Other representations than voxels are used. For example collections of 2D views and transformation parameters (Dosovitskiy et al., 2017). The most interesting are methods that work with raw data, the point cloud themselves. This alleviate the need to process the data and does not inadvertently increase the data dimensions, for example by artificially introduce voxels where there is no object present.

PointNet (?) directly operates on point clouds. To handle the input as a set of points (unordered), it uses a symmetric function over n input vectors and outputs a vector that is invariant to the input order. Typically sum and multiplication operators are such symmetric functions. After input and feature transforms by multi-layer perceptrons, a max pooling operator is used to map the input to a global feature. In the ModelNet40 shape classification benchmark there are more than 12000 CAD models from 40 object categories. PointNet achieves state of the art results compared to volumetric methods for a fraction of the computational costs.

Point clouds are also directly used in so-called deep kd-networks (Klokov and Lempitsky, 2017). A kd-tree is constructed by recursively picking the coordinate axis with the largest range of point coordinates and splitting the set of points into two subsets of equal size. These subsets are recursed into successively. The recursion stops at a particular level, depth D . The kd-networks are purported to outperform for example PointNet amongst other model architectures.

1888 A deep permutation equivariant (for semisupervised learning) and permutation invariant
1889 (for supervised learning) network has also been directly applied to point clouds (Ravanbakhsh
1890 et al., 2016). It does not reach the ModelNet40 accuracy levels from PointNet or the kd-
1891 networks though.

1892 PointNet++ Qi et al. (2017) introduces hierarchical structure to PointNet. This fits better
1893 non-uniform point distributions and seems to surpass kd-nets again on the ModelNet40 task.

1894

1895

RECOMMENDER ENGINE

Contents

1897

1898

1899

The described nonparametric Bayesian models (Chapter 3, 4, 5, and 6) are not limited to computer vision tasks. This chapter describes a recommender engine in which groups of runners are extracted from data collected from social media.

Outline

1901

1902

1903

1904

1905

We (1) introduce the form of the data at hand, (2) describe a multi-modal Von Mises-Uniform distribution to model the individual runners, (3) use a Dirichlet Process prior to group people, (4) use the previously described MCMC methods to perform inference, (5) show the results on an artificial and real-world data set, and (6) discuss ways with which the model can be expanded.

7.1 Application

1906

The data of people exercising can be considered binary (someone is either exercising or not in a particular timeslot). We do have however more information available. We know how often people have been exercising in a timeslot. This data has the form as visualized in Table 7.1.

1910

Table 7.1: Example of the type of data about the timing of exercising. A person is represented by row, her preferences by column. There is not a predefined number of users or groups of users.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
0	0	0	0	4	23	38	9	12	6	2	7	2	3	2	7	5	3	2	0	3	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	11	4	4	2	1	2	0	0	0
0	0	0	0	0	1	0	0	1	2	0	0	0	1	0	2	3	1	1	3	2	0	0	0
0	0	0	0	0	0	5	4	1	3	11	3	7	3	3	4	3	10	23	3	0	0	0	0
0	0	0	0	0	0	8	35	23	12	4	41	14	11	8	7	14	38	36	6	5	1	2	0
0	0	0	0	0	0	0	1	9	2	0	3	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	29	9	8	7	5	3	0	2	0	1	8	6	1	4	1	2	0	0
0	0	0	0	0	2	14	12	7	1	0	2	0	2	0	3	4	6	4	3	9	2	0	1
0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	3	3	2	0	1	0	0	0
0	0	0	0	0	0	0	1	1	0	1	1	0	0	0	2	1	2	3	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	4	5	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	3	2	1	0	1	1	0	0	0	0	3	11	12	0	0	0
0	0	0	0	0	0	0	0	0	2	1	0	0	0	1	0	3	5	4	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	18	4	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	2	0	0	1	9	14	14	2	1	4	15	4	0	0
0	0	0	0	0	0	0	0	2	4	2	1	3	5	2	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	3	3	1	3	2	3	0	0
0	0	0	0	0	0	0	1	2	1	3	2	0	1	2	0	0	0	0	1	2	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	8	4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	4	6	2	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	1	1	6	2	1	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	2	2	3	12	18	12	12	2	4	3	5	6	43	37	16	1	2	0

1911 The first row defines the time of day, starting at midnight till the last timeslot from 23.00 till
 1912 00.00. In this particular case it shows that nobody is running from midnight to four o'clock
 1913 in the morning, understandably so.

1914 7.2 Model of Individuals

1915 7.2.1 Multi-modal Normal-Uniform Distribution Model

1916 We first postulate the likelihood function for the moments at which people exercise through
 1917 the day as defined in Eq. 7.1.

$$f(x|\theta) = w_0 \mathcal{U}(a, b) + \sum_{i=1}^2 w_i \mathcal{N}(\mu_i, \sigma_i) \quad (7.1)$$

1918 The likelihood (Eq. 7.1) is built up out of three probability density functions: one Uniform dis-
 1919 tribution $\mathcal{U}(a, b)$ with a and b as parameters and two Normal distributions $\mathcal{N}(\mu_i, \sigma_i)$ with
 1920 mean μ_i and σ_i . The distributions are weighted by the factors w_0, w_1, w_2 . The collection of
 1921 parameters for the likelihood function is referred to by $\theta = \{a, b, \mu_1, \sigma_1, \mu_2, \sigma_2, w_0, w_1, w_2\}$.

1922 This probability density function $f(x|\theta)$ will have the form as in Fig. 7.1. The uniform
 1923 distribution generates values here between 00:00 and 24:00. There are on top of that the
 1924 two Normal distributions that form peaks at certain moments during the day.

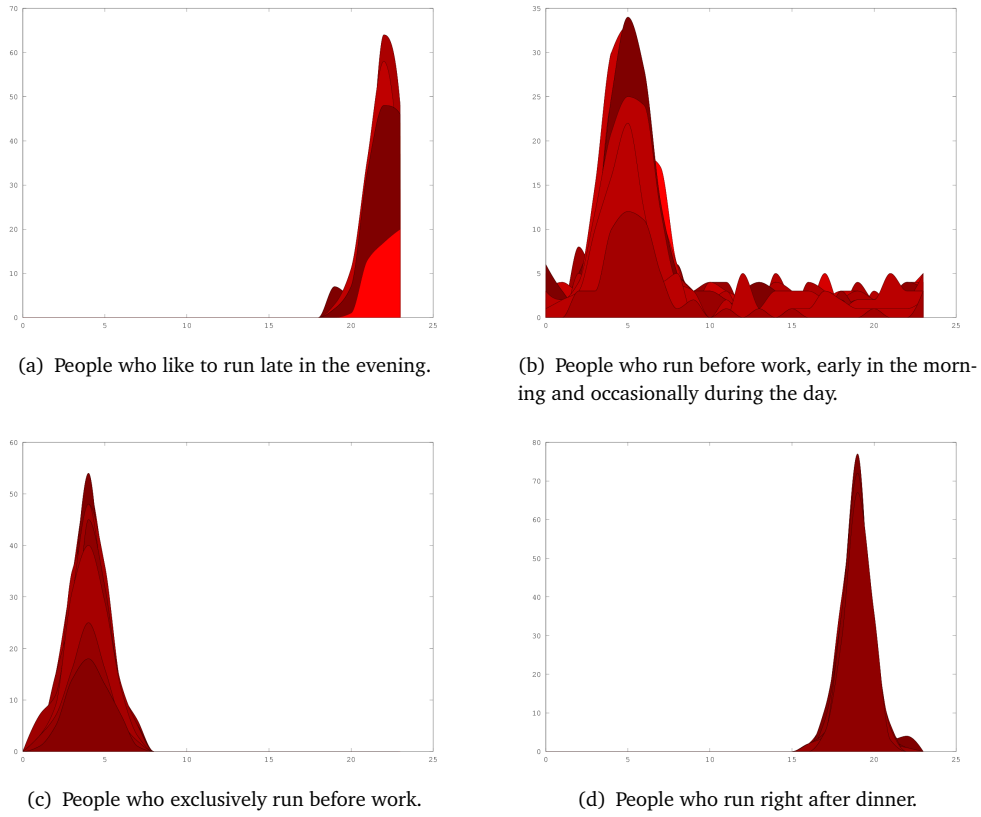


Figure 7.1: The likelihood function for the moments at which people decide to exercise during the day. On the horizontal axis time, on the vertical axis the frequency of exercising.

There is something noticable in Fig. 7.1, namely that the 24 hours of a day cause the Normal distribution to be cut off. Especially in Fig. 7.1 (a) there should be some considerable likelihood of running in the wee hours of the morning between 00:00 and 01:00.

7.2.2 Multi-modal Von-Mises-Uniform Distribution Model

There are several options to define a distribution over a limited range T . A so-called wrapped distribution is a distribution defined over the unity circle. By just multiplying it with $T/(2\pi)$ it can be used to define a probability density function over a day ($T = 24$).

$$f(x|\theta) = w_0 \mathcal{U}(a, b) + \sum_{i=1}^2 w_i \mathcal{VM}(\mu_i, \kappa_i) \quad (7.2)$$

The likelihood (Eq. 7.2) is again built up out of three probability density functions: one Uniform distribution $\mathcal{U}(a, b)$ with a and b as parameters and two Von Mises distributions $\mathcal{VM}(\mu_i, \kappa_i)$ with mean μ_i and κ_i . The parameters μ_i will be scaled and shifted with $[a, b]$ so all variables within this range fall on the unity circle. The parameter κ_i plays the same role as σ_i for the Normal distribution. The distributions are weighted by the factors w_0, w_1, w_2 . The collection of parameters for the likelihood function is referred to by $\theta = \{a, b, \mu_1, \kappa_1, \mu_2, \kappa_2, w_0, w_1, w_2\}$.

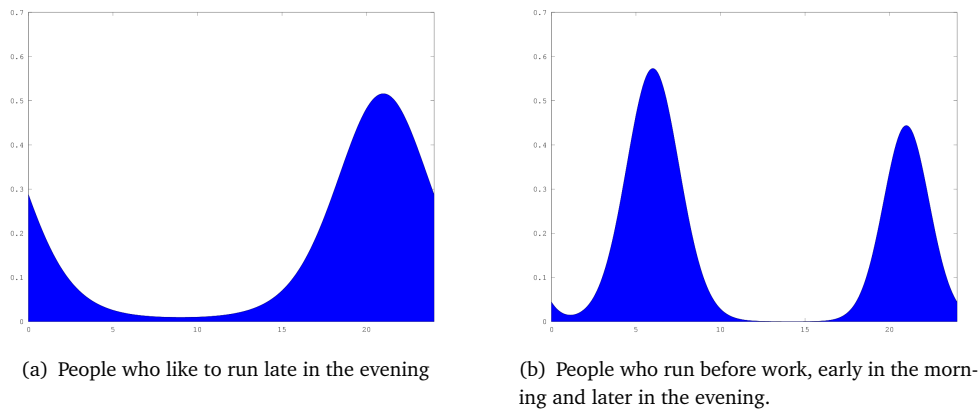


Figure 7.2: The improved likelihood function for the moments at which people decide to exercise during the day using the Von Mises distribution. On the horizontal axis time, on the vertical axis the frequency of exercising.

The likelihood with Von Mises distributions rather than Normal distributions is visualized in Fig. 7.2. The Von Mises distributions capture behavior in deviations from a standard exercise time. It does not take into account structured deviations, for example running late from work every Thursday, or weekends, or a person either running in the morning or in the evening, but never both at the same day.

7.2.3 Hyperparameters

The parameters θ for each user j are either fixed or generated from prior distributions. The hyperparameters a and b for the Uniform distribution are set to 0 and 24. People can run potentially any time of the day. The hyperparameters μ_i, κ_i for the Von Mises distributions are generated from a Uniform-Exponential distribution (Eq. 7.3). The Uniform distribution reflects the fact that if people run on regular times, this time can be any time of the day. The Exponential distribution defines a prior on how much people deviate from such a regular time to exercise.

$$\begin{aligned} f(\mu_i|a, b) &= \mathcal{U}(a, b) \\ f(\kappa_i|\lambda) &= \mathcal{E}(\lambda) \end{aligned} \tag{7.3}$$

The Uniform distribution generates μ_i between a and b . The parameter κ_i is generated from an Exponential distribution with hyperparameter λ . If λ is set to be small (< 0.5) we have a high likelihood that κ can be large and we have pronounced peaks. In contrary, if λ is set to be large, the Von Mises distribution likely approaches the Uniform distribution due to a higher chance of sampling a small value for κ .

The weights we sample from a normalized product of a zero-deflated Bernoulli distribution and a Dirichlet distribution (Eq. 7.4).

$$f(w_i|p, \alpha_i) = \mathcal{B}(p)\mathcal{D}(\alpha_i)/Z \quad (7.4)$$

The Bernoulli distribution samples zeros and ones with probability of $p = 0.5$, leading to 3-vectors like 001, 101, etc. The distribution is corrected in such way that the chance to sample 000 is zero. The Bernoulli distribution only gives weights $w_i = 0$ or $w_i = 1$, hence it is multiplied with a Dirichlet distribution. The Dirichlet samples a 3-vector with weights between zero and one where the weights $\sum_i w_i = 1$. A symmetric Dirichlet distribution with $\alpha = 1$ is similar to the Uniform distribution over the simplex. It is set slightly more towards favoring particular distributions with $\alpha_i = 1/3$ (we assume that people sample one or two distributions, and rarely from all three). The product with the zero-deflated Bernoulli distribution is made up to sum up to one again by normalizing the result.

We can combine Eq. 7.3 and Eq. 7.4 in Eq. 7.5:

$$\theta \sim \mathcal{U}(a, b)\mathcal{E}(\lambda)\mathcal{B}(p)\mathcal{D}(\alpha_i)/Z \quad (7.5)$$

To sample the parameters θ this is the base distribution we will encounter in the next section 7.3.

7.3 Model of Groups

Each person's exercise schedule is represented by a Von-Mises-Uniform distribution. People that are similar do have exercise schedules that can be represented by the same Von-Mises-Uniform distribution. To group similar schedules we define a nonparametric discrete distribution over a potentially infinite number of groups with each person assigned to a group.

A Dirichlet Process (Eq. 7.6) is a distribution over distributions that can be used as a prior for such a nonparametric discrete distribution.

$$DP(\alpha, H) \quad (7.6)$$

The Dirichlet Process has (1) a hyperparameter α , which defines the likelihood that there are many clusters versus few clusters (although it doesn't say anything about its actual count), and (2) a base distribution H , the distribution that generates θ (Eq.7.5).

7.4 Inference

The implementation of the model makes use of Gibbs sampling with auxiliary variables Jain and Neal (2007).

Details on this algorithm can be found in Jain and Neal (2007) and previous work of the authors van Rossum et al. (2016b,c).

Algorithm 15 Gibbs sampling over auxiliary variables

```

1: procedure GIBBS ALGORITHM WITH AUXILIARY VARIABLES( $w, \lambda_0, \alpha$ )      ▷ Accepts schedule  $w$ ,
   hyperparameters  $\lambda_0, \alpha$ , number of auxiliary variables  $m$ , and returns  $k$  groups
2:   for all  $t = 1 : T$  do
3:     for all  $i = 1 : N$  do
4:       for all  $j = 1 : m$  do
5:          $\theta_j \sim H(\lambda_0)$       ▷ Sample  $\theta_j$  from base distribution  $H$  in Eq. 7.5
6:       end for
7:       for all  $j = 1 : K + m, j \neq i$  do
8:          $L_j = \text{likelihood}(w_i, \theta_j)$       ▷ Update likelihood for all theta (except  $\theta_i$ ) given  $w_i$ 
9:       end for
10:       $P_{-i=1:K} = b \sum_{-i} L_{-i}$       ▷ Calculate probability of existing cluster
11:       $P_{-i=K:K+m} = b\alpha / m L_m L_{-i}$       ▷ Calculate probability of new cluster
12:       $\theta_i = \theta_j$  according to above  $P_{-i}$       ▷ Sample  $\theta_i$  accord. to above prob
13:      Remove unused clusters
14:    end for
15:    for all  $j = 1 : K$  do
16:       $\theta_j \sim p(\theta_j | y)$       ▷ Update  $\theta_j$ 
17:    end for
18:  end for
19:  return summary on  $\theta_k$  for  $k$  groups of runners
20: end procedure

```

7.5 Results

The algorithm is run first on an artificial dataset of which we know the ground truth (Sect. 7.5.1) and next on the real-world dataset from Twitter (Sect. 7.5.2).

7.5.1 Artificial Dataset

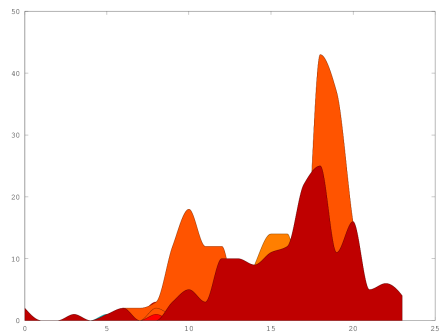
The algorithm has been used on a self-generated dataset (generated from the probability density function as in Fig. 7.2). In this case we have the ground truth that establishes which exercise schedule comes from which probability density function. Using this ground truth we can calculate how often our algorithm makes a mistake, grouping a person with people that belong to another group. The results with this dataset are perfect (Rand Index equal to one: perfect clustering). The results are of the form of Table 7.2 (except by a permutation of indices), hence are not shown (the indices have no intrinsic meaning).

Table 7.2: Top row: A sequence of cluster indices indicates the ground truth. Each cluster index represents a multi-modal Von-Mises-Uniform distribution with different parameters θ . Bottom row: A sequence of cluster indices that are the result of the described algorithm. Each cluster index represents again a multi-modal Von-Mises distribution. Errors would be represented by an inconsistent mapping from the top row to the bottom row.

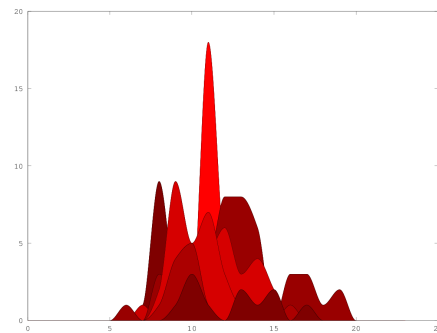
1	4	4	5	2	2	2	1	3	2	1	5	1	1	4	4	3	1	3	1	2	...
4	5	5	1	2	2	2	4	3	2	4	1	4	4	5	5	3	4	3	4	2	...

1996 7.5.2 Real-world Dataset

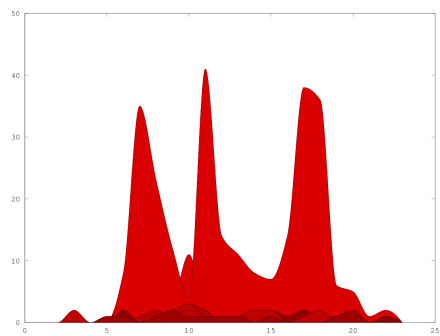
1997 The results when we actually use the collected Twitter dataset can be best visualized (Fig. 7.3).
 1998 This dataset consists of around 4000 moments at which people decide to run. The dataset
 1999 is subsequently filtered on regulars, people that at least have run a few times.



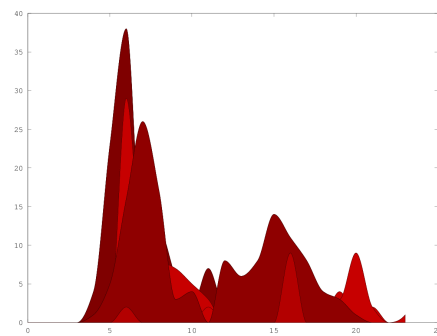
(a) People who predominally seem to like to run in the early evening.



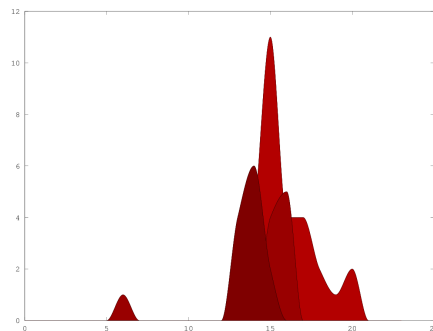
(b) People who run during the day, not later than at 20:00 o'clock.



(c) People who seem to run all the time (interesting collection though!).



(d) People who run early in the morning.



(e) People who run around 16:00 o'clock and almost never run at another time.

Figure 7.3: After running the algorithm, the above figures show the different categories of runners that have been found.

2000 The results in Fig. 7.3 show that there is a categorization of people indeed. After the al-
 2001 gorithm has done its work it is possible to assign a certain label in a post-hoc manner, e.g.
 2002 “People who run around 16:00 o'clock and almost never run at another time.”.

7.6 Discussion

The postulated model allows to reason about groups of people performing exercising, without predefining what these groups constitute apart from very general characteristics such as that there might be preferred times of day to exercise. A Dirichlet process is used as a nonparametric Bayesian model in which both the number of groups and the assignment of people exercising are learnt from the data.

There are several directions in which this research can be extended. First, more data would be very helpful. Only a limited number of people are posting consistently their training data online. Tapping into the data of current fitness promoting companies would allow the model to wash out the prior a bit more and adjust to the data. Second, we also collected weather data over this time period and also expect the day of the week to have significant influence. When there will be more data available these are logically dimensions to include in the dataset. Third, it would be interesting to study if people relate to the group of people they have been categorized with. Can this help or support their exercise regime?

2017



2018

DISCUSSION AND CONCLUSIONS

- 2021 M Abdel-Hameed. Optimal replacement policies for devices subject to a gamma wear process. *The*
 2022 *theory and applications of reliability*, pages 397–412, 2012.
- 2023 David J Aldous. Representations for partially exchangeable arrays of random variables. *Journal of*
 2024 *Multivariate Analysis*, 11(4):581–598, 1981.
- 2025 David J Aldous. *Exchangeability and related topics*. Springer, 1985.
- 2026 John Aldrich and Others. R.A. Fisher and the making of Maximum Likelihood 1912-1922. *Statistical*
 2027 *Science*, 12(3):162–176, 1997.
- 2028 Charles E Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric
 2029 problems. *The annals of statistics*, pages 1152–1174, 1974.
- 2030 Stefan Banach and Alfred Tarski. Sur la décomposition des ensembles de points en parties respec-
 2031 tivement congruentes. *Fund. math*, 6(1):924, 1924.
- 2032 Federico Bassetti, Roberto Casarin, and Fabrizio Leisen. Beta-product dependent Pitman–Yor pro-
 2033 cesses for Bayesian inference. *Journal of Econometrics*, 180(1):49–72, 2014.
- 2034 Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov
 2035 chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.
- 2036 Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European*
 2037 *conference on computer vision*, pages 404–417. Springer, 2006.
- 2038 Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2016.
- 2039 David Blackwell and James B MacQueen. Ferguson distributions via Pólya urn schemes. *The annals*
 2040 *of statistics*, pages 353–355, 1973.
- 2041 Phil Blunsom and Trevor Cohn. A hierarchical Pitman-Yor process HMM for unsupervised part of
 2042 speech induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational*
 2043 *Linguistics: Human Language Technologies-Volume 1*, pages 865–874. Association for Computa-
 2044 tional Linguistics, 2011.
- 2045 Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space
 2046 of generative networks. *arXiv preprint arXiv:1707.05776*, 2017.
- 2047 Robert C Bolles and Martin A Fischler. A RANSAC-Based Approach to Model Fitting and Its Application
 2048 to Finding Cylinders in Range Data. In *IJCAI*, volume 1981, pages 637–643, 1981.
- 2049 Anna Bosch, Andrew Zisserman, and Xavier Muñoz. Scene classification using a hybrid generative/dis-
 2050 criminative approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(4):712–
 2051 727, 2008.

- Guillaume Bouchard and Bill Triggs. The tradeoff between generative and discriminative classifiers. In *16th IASC International Symposium on Computational Statistics (COMPSTAT'04)*, pages 721–728, 2004.
- George E P Box and George C Tiao. *Bayesian inference in statistical analysis*, volume 40. John Wiley & Sons, 2011.
- H Bühlmann. *Austauschbare stochastische Variablen und ihre Grenzwertsätze*. PhD thesis, ETH Zürich, 1960.
- Wray L Buntine. Operations for learning with graphical models. *JAIR*, 2:159–225, 1994.
- John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.
- Jason Chang and John W Fisher III. Parallel sampling of dp mixture models using sub-cluster splits. In *Advances in Neural Information Processing Systems*, pages 620–628, 2013.
- Haifeng Chen, Peter Meer, and David E Tyler. Robust regression for data with multiple structures. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I—1069. IEEE, 2001.
- David B Dahl. An Improved Merge-Split Sampler for Conjugate Dirichlet Process Mixture Models. Technical report, University of Wisconsin–Madison, November 2003.
- David B Dahl. Sequentially-Allocated Merge-Split Sampler for Conjugate and Nonconjugate Dirichlet Process Mixture Models. Technical report, Texas A&M University, November 2005.
- Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.
- Hal Daume. Fast search for Dirichlet process mixture models. In *International Conference on Artificial Intelligence and Statistics*, pages 83–90, 2007.
- B de Finetti. Funzione caratteristica di un fenomeno aleatorio. *Atti Reale Accademia Nazionale dei Lincei*, VI:86–133, 1930.
- Bruno De Finetti. La prévision: ses lois logiques, ses sources subjectives. In *Annales de l'institut Henri Poincaré*, volume 7, pages 1–68, 1937.
- David GT Denison. *Bayesian methods for nonlinear classification and regression*, volume 386. John Wiley & Sons, 2002.
- Persi Diaconis and David Freedman. de Finetti’s theorem for Markov chains. *The Annals of Probability*, pages 115–130, 1980.
- Alexey Dosovitskiy, Jost Tobias Springenberg, Maxim Tatarchenko, and Thomas Brox. Learning to generate chairs, tables and cars with convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):692–705, 2017.
- François Dufresne, Hans U Gerber, and Elias S W Shiu. Risk theory with the gamma process. *Astin Bulletin*, 21(02):177–192, 1991.
- David B Dunson, Ya Xue, and Lawrence Carin. The matrix stick-breaking process. *Journal of the American Statistical Association*, 2012.
- Richard Durstenfeld. Algorithm 235: Random Permutation. *Communications of the ACM*, 7(7):420, July 1964. ISSN 0001-0782. doi: 10.1145/364520.364540. URL <http://doi.acm.org/10.1145/364520.364540>.
- Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.
- Stewart N Ethier. The distribution of the frequencies of age-ordered alleles in a diffusion model. *Advances in Applied Probability*, pages 519–532, 1990.
- Warren John Ewens. Population genetics theory-the past and the future. In *Mathematical and statistical developments of evolutionary theory*, pages 177–227. Springer, 1990.

- 2100 Stefano Favaro, Yee Whye Teh, and Others. MCMC for normalized random measure mixture models.
2101 *Statistical Science*, 28(3):335–359, 2013.
- 2102 William Feller. An introduction to probability theory and its applications. Vol. I. 1950.
- 2103 Thomas S Ferguson. A Bayesian analysis of some nonparametric problems. *The annals of statistics*,
2104 pages 209–230, 1973.
- 2105 Thomas S Ferguson. Prior distributions on spaces of probability measures. *The annals of statistics*,
2106 pages 615–629, 1974.
- 2107 E.W. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications.
2108 *Biometrics*, 21:768–769, 1965.
- 2109 Wolfgang Förstner and Eberhard Gülch. A fast operator for detection and precise location of distinct
2110 points, corners and centres of circular features. In *Proc. ISPRS intercommission conference on fast*
2111 *processing of photogrammetric data*, pages 281–305, 1987.
- 2112 David Freedman and Persi Diaconis. On inconsistent Bayes estimates in the discrete case. *The Annals*
2113 *of Statistics*, pages 1109–1118, 1983.
- 2114 David Heaven Fremlin. *Measure theory*, volume 4. Torres Fremlin, 2000.
- 2115 Orazio Gallo, Roberto Manduchi, and Abbas Rafii. CC-RANSAC: Fitting planes in the presence of
2116 multiple surfaces in range data. *Pattern Recognition Letters*, 32(3):403–410, 2011.
- 2117 Qing-Bin Gao and Shi-Liang Sun. Human activity recognition with beta process hidden Markov
2118 models. In *Machine Learning and Cybernetics (ICMLC), 2013 International Conference on*, volume 2,
2119 pages 549–554. IEEE, 2013.
- 2120 Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian
2121 restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–
2122 741, 1984.
- 2123 Zoubin Ghahramani and Thomas L Griffiths. Infinite latent feature models and the Indian buffet
2124 process. In *Advances in neural information processing systems*, pages 475–482, 2005.
- 2125 Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron
2126 Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information*
2127 *processing systems*, pages 2672–2680, 2014.
- 2128 Anirudh Goyal, Nan Rosemary Ke, Surya Ganguli, and Yoshua Bengio. Variational walkback: Learning
2129 a transition operator as a stochastic recurrent net. *arXiv preprint arXiv:1711.02282*, 2017.
- 2130 Max Halperin and G L Burrows. The Effect of Sequential Batching for Acceptance–Rejection Sam-
2131 pling Upon Sample Assurance of Total Product Quality. *Technometrics*, 2(1):19–26, 1960.
- 2132 Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*,
2133 volume 15, page 50. Citeseer, 1988.
- 2134 Li He, Hairong Qi, and Russell Zaretzki. Beta process joint dictionary learning for coupled feature
2135 spaces with application to single image super-resolution. In *Proceedings of the IEEE Conference on*
2136 *Computer Vision and Pattern Recognition*, pages 345–352, 2013.
- 2137 Nils Lid Hjort. Nonparametric Bayes estimators based on beta processes in models for life history
2138 data. *The Annals of Statistics*, pages 1259–1294, 1990.
- 2139 Fred M Hoppe. Size-biased filtering of Poisson-Dirichlet samples with an application to partition
2140 structures in genetics. *Journal of Applied Probability*, pages 1008–1012, 1986.
- 2141 Paul V.C. Hough. Method and Means for Recognizing Complex Patterns, Dec 1962. URL <https://www.google.com/patents/US3069654>. Patent US 3069654 A.
- 2143 Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. Stacked generative
2144 adversarial networks. *arXiv preprint arXiv:1612.04357*, 2016.
- 2145 Tommi S Jaakkola, David Haussler, and Others. Exploiting generative models in discriminative clas-
2146 sifiers. *Advances in neural information processing systems*, pages 487–493, 1999.
- 2147 Sonia Jain and Radford M. Neal. A Split-Merge Markov Chain Monte Carlo Procedure for the Dirichlet
2148 Process Mixture Model. *Journal of Computational and Graphical Statistics*, 13(1):158–182, 2004.

- ISSN 10618600. URL <http://www.jstor.org/stable/1391150>.
- Sonia Jain and Radford M Neal. Splitting and Merging Components of a Nonconjugate Dirichlet Process Mixture Model. *Bayesian Analysis*, 2(3):445–472, 2007.
- Varun Jampani, Sebastian Nowozin, Matthew Loper, and Peter V Gehler. The informed sampler: A discriminative approach to bayesian inference in generative computer vision models. *Computer Vision and Image Understanding*, 136:32–44, 2015.
- Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.
- Dominik Joho, Gian Diego Tipaldi, Nikolas Engelhard, Cyrill Stachniss, Wolfram Burgard, Martin Senk, Felix Faber, Maren Bennewitz, Clemens Eppner, Attila Görög, and Others. Unsupervised Scene Analysis and Reconstruction Using Nonparametric Bayesian Models. *Robotics and Autonomous Systems (RAS)*, 59(5):319–328, 2011.
- A Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in neural information processing systems*, 14:841, 2002.
- Michael I Jordan. Hierarchical models, nested models and completely random measures. *Frontiers of Statistical Decision Making and Bayesian Analysis: in Honor of James O. Berger*. New York: Springer, 2010.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- J F C Kingman. *Poisson processes*, volume 3 of *Oxford Studies in Probability*. The Clarendon Press Oxford University Press, New York, 1993. ISBN 0-19-853693-3.
- J F C Kingman. Some further analytical results in the theory of regenerative events. *Journal of Mathematical Analysis and Applications*, 11:422–433, 1965.
- J F C Kingman. Random partitions in population genetics. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 361, pages 1–20. The Royal Society, 1978.
- John Kingman. Completely random measures. *Pacific Journal of Mathematics*, 21(1):59–78, 1967.
- Roman Klovov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 863–872. IEEE, 2017.
- David Knowles, Zoubin Ghahramani, and Konstantina Palla. A reversible infinite HMM using normalised random measures. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1998–2006, 2014.
- Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- A Kolmogorov. *Grundbegriffe der wahrscheinlichkeitsrechnung*, volume 2 of *Ergebnisse der Mathematik und ihrer Grenzgebiete*. Springer-Verlag, 1933.
- Uwe Küchler and Stefan Tappe. Bilateral Gamma distributions and processes in financial mathematics. *Stochastic Processes and their Applications*, 118(2):261–283, 2008.
- Kenichi Kurihara, Max Welling, and Yee Whye Teh. Collapsed Variational Dirichlet Process Mixture Models. In *IJCAI*, volume 7, pages 2796–2801, 2007.
- Pierre-Simon Laplace. *Théorie analytique des probabilités*. V. Courcier, 1820.
- Henri Lebesgue. Intégrale, longueur, aire. *Annali di Matematica Pura ed Applicata (1898-1922)*, 7(1):231–359, 1902.
- Zhidong Li, Bang Zhang, Yang Wang, Fang Chen, Ronnie Taib, Vicky Whiffin, and Yi Wang. Water pipe condition assessment: a hierarchical beta process approach for sparse incident data. *Machine learning*, 95(1):11–26, 2014.
- Antonio Lijoi and Igor Prünster. Models beyond the Dirichlet process. *Bayesian nonparametrics*, 28: 80, 2010.

- 2197 S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inform. Theory*, 28:129–137, 1982. Originally
2198 as an unpublished Bell laboratories Technical Note (1957).
- 2199 David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999.*
2200 *The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee,
2201 1999.
- 2202 Steven N MacEachern. Comment on "Splitting and Merging Components of a Nonconjugate Dirichlet
2203 Process Mixture Model" by Jain and Neal. *Bayesian Analysis*, 2(3):483–494, 2007.
- 2204 Steven N MacEachern and Peter Müller. Estimating mixture of Dirichlet process models. *Journal of*
2205 *Computational and Graphical Statistics*, 7(2):223–238, 1998.
- 2206 Dilip B Madan and Eugene Seneta. The variance gamma (VG) model for share market returns. *Journal*
2207 *of business*, pages 511–524, 1990.
- 2208 Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward
2209 Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*,
2210 21(6):1087–1092, 1953. doi: 10.1063/1.1699114.
- 2211 Radford M Neal. Defining Priors for Distributions Using Dirichlet Diffusion Trees. Technical report,
2212 University of Toronto, Toronto, Ontario, Canada, 2001.
- 2213 Radford M Neal. Markov Chain Sampling Methods for Dirichlet Process Mixture Models. *Journal of*
2214 *Computational and Graphical Statistics*, 9(2):249–265, 2000.
- 2215 Peter Orbanz and Daniel M Roy. Bayesian models of graphs, arrays and other exchangeable random
2216 structures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(2):437–461, 2015.
- 2217 Jim Pitman and Marc Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable
2218 subordinator. *The Annals of Probability*, pages 855–900, 1997.
- 2219 Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature
2220 learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*,
2221 pages 5105–5114, 2017.
- 2222 Rajat Raina, Yirong Shen, Andrew McCallum, and Andrew Y Ng. Classification with hybrid gen-
2223 erative/discriminative models. In *Advances in neural information processing systems*, page None,
2224 2003.
- 2225 William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American*
2226 *Statistical association*, 66(336):846–850, 1971.
- 2227 Vinayak Rao and Yee W Teh. Spatial normalized gamma processes. In *Advances in neural information*
2228 *processing systems*, pages 1554–1562, 2009.
- 2229 Carl Edward Rasmussen and Christopher K I Williams. *Gaussian processes for machine learning*, vol-
2230 ume 2. 2006.
- 2231 Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Deep learning with sets and point clouds.
2232 *arXiv preprint arXiv:1611.04500*, 2016.
- 2233 Eugenio Regazzini, Antonio Lijoi, and Igor Prünster. Distributional results for means of normalized
2234 random measures with independent increments. *Annals of Statistics*, pages 560–585, 2003.
- 2235 Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and
2236 approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.
- 2237 James C Ross, Peter J Castaldi, Michael H Cho, and Jennifer G Dy. Dual beta process priors for latent
2238 cluster discovery in chronic obstructive pulmonary disease. In *Proceedings of the 20th ACM SIGKDD*
2239 *international conference on Knowledge discovery and data mining*, pages 155–162. ACM, 2014.
- 2240 Daniel M Roy and Yee W Teh. The mondrian process. In *Advances in neural information processing*
2241 *systems*, pages 1377–1384, 2009.
- 2242 Anirban Roychowdhury and Brian Kulis. Gamma processes, stick-breaking, and variational inference.
2243 In *Artificial Intelligence and Statistics*, pages 800–808, 2015.
- 2244 Donald B Rubin and Others. Bayesianly justifiable and relevant frequency calculations for the applied
2245 statistician. *The Annals of Statistics*, 12(4):1151–1172, 1984.

- 2246 Stuart Russell, Peter Norvig, and Artificial Intelligence. *Artificial Intelligence: A modern approach.*
2247 *Artificial Intelligence.* Prentice-Hall, Englewood Cliffs, 25:27, 1995.
- 2248 Leonard J Savage. *The foundations of statistics.* Courier Corporation, 1972.
- 2249 Stanley Sawyer and Daniel Hartl. A sampling theory for local selection. *Journal of Genetics*, 64(1):
2250 21–29, 1985.
- 2251 René L Schilling. *Measures, integrals and martingales*, volume 13. Cambridge University Press, 2005.
- 2252 Abhishek Sharma, Oliver Grau, and Mario Fritz. Vconv-dae: Deep volumetric shape learning without
2253 object labels. In *European Conference on Computer Vision*, pages 236–250. Springer, 2016.
- 2254 Jianbo Shi and Carlo Tomasi. Good features to track. In *Computer Vision and Pattern Recognition,*
2255 *1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600. IEEE,
2256 1994.
- 2257 Nozer Singpurwalla. Gamma processes and their generalizations: an overview. In *Engineering prob-*
2258 *abilistic design and maintenance for flood protection*, pages 67–75. Springer, 1997.
- 2259 Scott A Sisson and Yanan Fan. Likelihood-free MCMC. *Handbook of Monte Carlo*, ed. Brooks, A.,
2260 Gelman, A., Jones, GL, Meng XL, pages 313–335, 2011.
- 2261 I Sobel. *Camera Models and Perception.* PhD thesis, Stanford University, Stanford, CA, 1970.
- 2262 Jascha Sohl-Dickstein, Eric A Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised
2263 learning using nonequilibrium thermodynamics. *arXiv preprint arXiv:1503.03585*, 2015.
- 2264 Jiaming Song, Shengjia Zhao, and Stefano Ermon. A-nice-mc: Adversarial training for mcmc. *arXiv*
2265 *preprint arXiv:1706.07561*, 2017.
- 2266 Mike Steel. The maximum likelihood point for a phylogenetic tree is not unique. *Systematic Biology*,
2267 43(4):560–564, 1994.
- 2268 Erik B Sudderth and Michael I Jordan. Shared segmentation of natural scenes using dependent
2269 Pitman-Yor processes. In *Advances in Neural Information Processing Systems*, pages 1585–1592,
2270 2009.
- 2271 Charles Sutton and Andrew McCallum. An introduction to conditional random fields. *Machine Learn-*
2272 *ing*, 4(4):267–373, 2011.
- 2273 Terence Tao. *An introduction to measure theory*, volume 126. American Mathematical Soc., 2011.
- 2274 Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical Dirichlet processes.
2275 *Journal of the American statistical association*, 101(476), 2006.
- 2276 Romain Thibaux and Michael I Jordan. Hierarchical beta processes and the Indian buffet process. In
2277 *International Conference on Artificial Intelligence and Statistics*, pages 564–571, 2007.
- 2278 Michalis K Titsias. The infinite gamma-Poisson feature model. In *Advances in Neural Information*
2279 *Processing Systems*, pages 1513–1520, 2008.
- 2280 Dustin Tran, Matthew D Hoffman, Rif A Saurous, Eugene Brevdo, Kevin Murphy, and David M Blei.
2281 Deep probabilistic programming. *arXiv preprint arXiv:1701.03757*, 2017.
- 2282 Anne C van Rossum, Hai Xiang Lin, Johan Dubbeldam, and H Jaap van den Herik. Fundamentals of
2283 nonparametric bayesian line detection. In *International Conference on Pattern Recognition Applica-*
2284 *tions and Methods*, pages 175–193. Springer, 2016a.
- 2285 Anne C. van Rossum, Hai Xiang Lin, Johan Dubbeldam, and H. Jaap van den Herik. Nonparametric
2286 Bayesian Line Detection - Towards Proper Priors for Robotic Computer Vision. In *Proceedings of*
2287 *the 5th International Conference on Pattern Recognition Applications and Methods*, pages 119–127,
2288 Feb 2016b. ISBN 978-989-758-173-1. doi: 10.5220/0005673301190127.
- 2289 Anne C. van Rossum, Hai Xiang Lin, Johan Dubbeldam, and H. Jaap van den Herik. Nonparametric
2290 Segment Detection. In *Proceedings of the 8th European Starting AI Researcher Symposium (STAIRS)*,
2291 pages 203–208, Aug 2016c. ISBN 978-989-758-173-1. doi: 10.5220/0005673301190127.
- 2292 Niklas Vanhainen and Giampiero Salvi. Word Discovery with Beta Process Factor Analysis. In *INTER-*
2293 *SPEECH*, pages 799–802, 2012.

- 2294 Wei Wang and Stuart Russell. A Smart-dumb/Dumb-smart Algorithm for Efficient Split-merge MCMC.
2295 In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI15, pages
2296 902–911, Arlington, Virginia, United States, 2015. AUAI Press. ISBN 978-0-9966431-0-8. URL
2297 <http://dl.acm.org/citation.cfm?id=3020847.3020940>.
- 2298 Yingjian Wang and Lawrence Carin. Levy measure decompositions for the beta and gamma processes.
2299 *arXiv preprint arXiv:1206.4615*, 2012.
- 2300 Larry Wasserman. Asymptotic properties of nonparametric Bayesian procedures. In *Practical non-*
2301 *parametric and semiparametric Bayesian statistics*, pages 293–304. Springer, 1998.
- 2302 Robert L Wolpert and Katja Ickstadt. Poisson/gamma random field models for spatial statistics.
2303 *Biometrika*, 85(2):251–267, 1998.
- 2304 Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.
- 2305 Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong
2306 Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE*
2307 *conference on computer vision and pattern recognition*, pages 1912–1920, 2015.
- 2308 Jing-Hao Xue and D Michael Titterton. Comment on “On discriminative vs. generative clas-
2309 sifiers: A comparison of logistic regression and naive Bayes”. *Neural processing letters*, 28(3):
2310 169–187, 2008.
- 2311 Arnold Zellner. Optimal information processing and Bayes’s theorem. *The American Statistician*, 42
2312 (4):278–280, 1988.
- 2313 Michael M Zhang and Fernando Perez-Cruz. Accelerated inference for latent variable models. *arXiv*
2314 *preprint arXiv:1705.07178*, 2017.
- 2315 Wei Zhang and Jana Kösecká. Nonparametric estimation of multiple structures with outliers. In
2316 *Dynamical Vision*, pages 60–74. Springer, 2007.
- 2317 Mingyuan Zhou, Hongxia Yang, Guillermo Sapiro, David B Dunson, and Lawrence Carin. Dependent
2318 hierarchical beta process for image interpolation and denoising. In *International conference on*
2319 *artificial intelligence and statistics*, pages 883–891, 2011.
- 2320 Mingyuan Zhou, Lauren A Hannah, David B Dunson, and Lawrence Carin. Beta-negative binomial
2321 process and poisson factor analysis. *Journal of Machine Learning Research*, 2012.



PROBABILISTIC CONCEPTS

TODO: What follows down here is old and has to be adapted to the measure-theoretic realizations.

Let us introduce the notation of expectation for random variable X :

$$E_p[X] = \sum_i^k p(X = x_i)x_i \quad (\text{A.1})$$

For the continuous case, if $f_X(x)$ is a properly defined probability density function, the expected value becomes:

$$E_f[X] = \int_{-\infty}^{\infty} x f_X(x) dx \quad (\text{A.2})$$

From the context it can be seen that the object X at the left is a different object than x at the right. The former is random variable (a one-dimensional function, an (in)finite vector), the latter is a value of a random variable (a scalar). The term dx is a measure, in this case it assigns volume to *subsets of random variables values*. A proper notation would incorporate this aspect, but not much will be gained by creating such complex notations.

Let us also introduce the conditional probability:

$$p(X|Y) = \frac{p(X, Y)}{p(Y)} \quad (\text{A.3})$$

Suppose X is a discrete random variable, then the object $p(X)$ is a vector of finite size k , $p(Y)$ is a vector of finite size l , and $p(X|Y)$ as well as $p(X, Y)$ are matrices of size $k \times l$. A conditional probability hence ‘divides’ a matrix by a vector. It is a proper measure if $p(Y) \neq 0$ (for all values of - or events in - Y).

2338 In for example importance sampling (Sect. ??), the expectation is taken over a function of
 2339 a random variable. A function, if measurable, can be taken the expectation over using the
 2340 so-called law of the unconscious statistician:

$$E_f[g(X)] = \int_{-\infty}^{\infty} g(x)f_X(x)dx \quad (\text{A.4})$$

2341 Here $g(X)$ is a general measurable function, and not restricted to a probability density func-
 2342 tion.

2343 The notation above is an indefinite integral. We can approach this integral by Monte Carlo
 2344 integration (Sect. ??).

$$E_f[g(X)] = \frac{1}{k} \sum_{i=1}^k g(x_i) \quad \text{with} \quad x_i \sim f_X(x) \quad (\text{A.5})$$

2345 Rather than summing over $f_X(x_i)$, we now sample $x_i \sim f_X(x)$.

2346 A.1 Common Inequalities

2347 A.1.1 Markov's Inequality

2348 Markov's inequality comes up with an upper bound for the probability that an non-negative
 2349 random variable X exceeds some constant positive threshold a .

$$p(X \geq a) \leq \frac{E[X]}{a} \quad (\text{A.6})$$

2350 The proof in classical probability theory uses an indicator variable:

$$aI(X \geq a) \leq X \quad (\text{A.7})$$

2351 Here $I(X \geq a) = 1$ if the event $X \geq a$ occurs, setting the left-hand side to a (which is of
 2352 course smaller than X). And $I(X \geq a) = 0$ on the event $X < a$, which is naturally smaller
 2353 than the non-negative X .

2354 Expectations obey the inequality: if $(X \leq Y)$, then $E[X] \leq E[Y]$, hence:

$$E[aI(X \geq a)] \leq E[X] \quad (\text{A.8})$$

2355 And because expectations add up linearly:

$$E[aI(X \geq a)] = aE[I(X \geq a)] = a(1 \cdot p(X \geq a) + 0 \cdot p(X < a)) = a \cdot p(X \geq a) \quad (\text{A.9})$$

2356 So, we have Markov's inequality combining Eq. A.8 and A.9:

$$a \cdot p(X \geq a) \leq E[X] \quad (\text{A.10})$$

2357 A.1.2 Chebyshev's Inequality

2358 Now, Chebyshev's inequality defines in a similar way (Chebyshev was a teacher of Markov¹)
 2359 an upper bound on the deviation from the mean for a random variable. Recall the definition
 2360 of the variance of X and assume it is finite:

$$\text{Var}(X) = E[(X - E[X])^2] = \sigma^2 \quad (\text{A.11})$$

2361 Consider now the random variable $(X - E[X])^2$ and constant $a = (\sigma k)^2$ and write down
 2362 Markov's inequality:

$$p((X - E[X])^2 \geq (\sigma k)^2) \leq \frac{E[(X - E[X])^2]}{(\sigma k)^2} \quad (\text{A.12})$$

2363 Taking the square root of the inequality at the left, and using the definition of σ^2 at the
 2364 right, leads to:

$$p(|X - E[X]| \geq \sigma k) \leq \frac{1}{k^2} \quad (\text{A.13})$$

2365 A.1.3 Weak Law of Large Numbers

2366 Chebyshev's inequality can be used to prove the weak law of large numbers. Given that we
 2367 have a series of random variables, all with the same finite expectation, $E[X_i] = \mu$, then this
 2368 law states that the sample average $\bar{X} = \frac{1}{n}(X_1 + \dots + X_n)$ converges in probability towards
 2369 the expected value:

$$\bar{X} \xrightarrow{P} \mu \quad \text{for} \quad n \rightarrow \infty \quad (\text{A.14})$$

2370 We can use the independence assumption between variables X_i to write down the variance
 2371 and expectation of \bar{X} :

$$\text{Var}(\bar{X}) = \text{Var}\left(\frac{1}{n}(X_1 + \dots + X_n)\right) = \frac{1}{n^2} \text{Var}(X_1 + \dots + X_n) = \frac{\sigma^2}{n} E[\bar{X}] = \mu \quad (\text{A.15})$$

¹There were many mathematically gifted Markov's. This is Andrey Andreyevich Markov Sr., known from the Markov chains and Markov processes. Jr. is known from Markov's principle, Markov's rule and the Markov algorithm.

2372 And now we can apply Chebyshev's inequality on \bar{X} :

$$p(|\bar{X} - \mu| \geq \epsilon) \leq \frac{\sigma^2}{n\epsilon^2} \quad (\text{A.16})$$

2373 Convergence in probability towards X is the case if for all ϵ :

$$\lim_{n \rightarrow \infty} p(|\bar{X} - X| \geq \epsilon) = 0 \quad (\text{A.17})$$

2374 This is the case for $n \rightarrow \infty$ indeed.

2375 A.1.4 Strong Law of Large Numbers

2376 The strong law incorporates the weak law. Rather than convergence *in probability*, it states
2377 convergence *almost surely* towards the expected value.

$$\bar{X} \xrightarrow{a.s.} \mu \quad \text{for} \quad n \rightarrow \infty \quad (\text{A.18})$$

2378 The strong law states that with probability 1, for any $\epsilon > 0$, the inequality $|X - \mu| < \epsilon$
2379 holds for large enough n . The weak law states only that the average \bar{X} is likely near μ , but
2380 $|X - \mu| \geq \epsilon$ can still happen, even for large n .

2381 A.1.5 Common Distributions

2382 One of the probability distributions that is interesting to us is the beta-distribution. A normal
2383 distribution might be a reasonable prior for a continuous variable such as human heights in
2384 a population. If this variable however is itself a probability, a reasonable prior is the beta-
2385 distribution. The beta-distribution can be described as:

$$f(x, \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1-x)^{\beta-1} = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1} \quad (\text{A.19})$$

2386 Here B is the beta function and Γ is the gamma function, the continuous extension of the
2387 factorial function: $\Gamma(n) = (n-1)!$. Naturally, there are many of such extensions. The
2388 gamma function extends the factorial in a specific sense. It obeys the recurrence relation
2389 $f(x+1) = xf(x)$ with $f(1) = 1$. Its description is defined with an improper integral:

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx \quad (\text{A.20})$$

2390 The expected value of a random variable X with a beta-distribution:

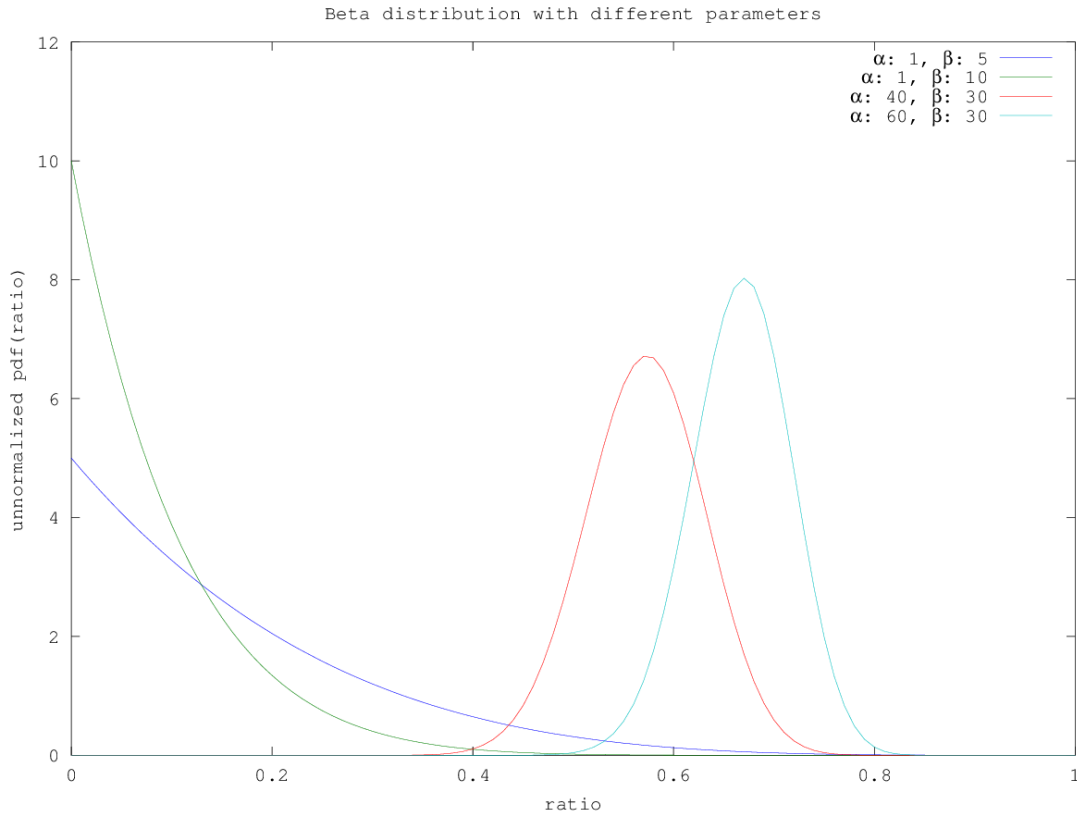


Figure A.1: The beta-distribution with different parameters. The x-axis is the quantity modelled, for example a ratio between wins and losses in a soccer season. The y-axis is the corresponding unnormalized density function. Setting $\alpha = 1$ shows a monotonically decreasing density function. Having a variable α allows probability mass to shift to the end.

$$E_f[X] = \int_0^1 x f(x, \alpha, \beta) dx = \int_0^1 \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^\alpha (1-x)^{\beta-1} dx = \frac{\alpha}{\alpha + \beta} \quad (\text{A.21})$$

2391 Let us illustrate the effects of the parameters of the beta-distribution.

2392 The stick-breaking presentation shows that a Dirichlet process consists of beta processes with
 2393 $\alpha = 1$. The Pitman-Yor process has α left variable. Informally, the location where the stick
 2394 will be broken (iteratively) for the Dirichlet process is ‘quite close to the beginning’ with high
 2395 probability. In the case α becomes larger, the breaking can also occur likely at the end of the
 2396 stick. This means for breaking a stick, say 20 times, the distribution of stick lengths for the
 2397 Pitman-Yor process has a much larger support. The difference between the large and small
 2398 sticks is much larger.

DIRICHLET-MULTINOMIAL INTERPRETATIONS

There are multiple interpretations of the compound compound "Dirichlet-multinomial" distribution. We will derive different related compound distributions and show how each of these compound distributions can be used as a likelihood function that is conjugate to the Dirichlet distribution. Let us recall the Dirichlet distribution (Eq. B.1).

$$p(\theta|\alpha) = \frac{1}{B(\alpha)} \prod_i \theta_i^{\alpha_i-1}. \quad (\text{B.1})$$

The normalizing constant $1/B(\alpha)$ contains the Beta function:

$$B(\alpha) = \frac{\prod_i \Gamma(\alpha_i)}{\Gamma(\sum_i \alpha_i)}. \quad (\text{B.2})$$

Regarding notation, the Γ function can be written as a factorial $\Gamma(x+1) = x!$. Steps like $\Gamma(1+x) = x\Gamma(x)$ and $\frac{\Gamma(x+n)}{\Gamma(x)} = x^n$ will not be made explicit in the following text.

We would like to find a conjugate distribution for the Dirichlet that will result in a Dirichlet distribution after performing Bayesian inference. The shape of this conjugate distribution is of the form:

$$p(z|\theta) = C \prod_i \theta_i^{z_i}. \quad (\text{B.3})$$

Here C is a normalization constant that depends on the interpretation of the random variable z .

There are three ways to interpret the Dirichlet-multinomial, namely as (1) a compound Dirichlet-categorical distribution (Appendix B.1), (2) a compound Dirichlet-multinomial distribution (Appendix B.2), and as (3) compound Dirichlet with N categorical distributions (Appendix B.3). Each of these interpretations are different. However, if they are used in a Bayesian context, they will all result in a posterior that is a Dirichlet distribution.

B.1 Dirichlet-Categorical

Often, the Dirichlet-multinomial is actually not a compound Dirichlet and *multinomial* distribution, but a compound Dirichlet and *categorical* distribution. Informally, the categorical distribution is a distribution over categories. More formally, it is a generalization of a Bernoulli distribution that assigns probabilities to a random variable with two possible outcomes to a random variable with more than two possible outcomes. The categorical distribution has the form as in Eq. B.4.

$$p(z|\theta) = \prod_i \theta_i^{z_i}. \quad (\text{B.4})$$

This notation defines only *a single* categorical variable, not a set of variables. It is using a 1-of- K encoded representation (cf. Bishop, 2016). Assume a regular 6-faced dice. The face with one pip is represented by $z_0 = [1, 0, 0, 0, 0, 0]$. The face with two pips $z_1 = [0, 1, 0, 0, 0, 0]$. There is only one category non-zero, $\sum_j z_{i,j} = 1$. The probabilities of each category are represented by θ_i and are one sixth for a regular dice.

Let us take the product of the Dirichlet distribution in Eq. B.1 and the categorical distribution in Eq. B.4:

$$p(z|\theta)p(\theta|\alpha) = \frac{1}{B(\alpha)} \prod_i \theta_i^{z_i + \alpha_i - 1}. \quad (\text{B.5})$$

The compound distribution is created by integrating over θ :

$$p(z|\alpha) = \int p(z|\theta)p(\theta|\alpha)d\theta = \frac{1}{B(\alpha)} \int \prod_i \theta_i^{z_i + \alpha_i - 1} d\theta. \quad (\text{B.6})$$

The multivariate Beta function (Eq. B.2) stems from normalizing the Dirichlet distribution and can also be written as an integral over the unnormalized Dirichlet distribution:

$$B(\alpha) = \int \prod_i \theta_i^{\alpha_i - 1} d\theta. \quad (\text{B.7})$$

Now we can use the 1-of- K representation to write the integral as follows:

$$\int \prod_i \theta_i^{z_i + \alpha_i - 1} d\theta = B(\alpha + z). \quad (\text{B.8})$$

2437 The compound distribution of the Dirichlet-categorical henceforth can be written as:

$$p(z|\alpha) = \int p(z|\theta)p(\theta|\alpha)d\theta = \frac{B(\alpha + z)}{B(\alpha)}. \quad (\text{B.9})$$

2438 We can now perform Bayes' rule that shows the benefit of the closed-form description of the
2439 compound distribution in the conjugate context:

$$p(\theta|z, \alpha) = \frac{p(z|\theta, \alpha)p(\theta|\alpha)}{p(z|\alpha)} \quad (\text{B.10})$$

2440 In this case the likelihood $p(z|\theta, \alpha) = p(z|\theta)$ has observations z that only depend on θ (and
2441 θ depends on α).

$$p(\theta|z, \alpha) = \frac{p(z|\theta)p(\theta|\alpha)}{p(z|\alpha)} \quad (\text{B.11})$$

2442 Hence the posterior of the Dirichlet prior with the categorical likelihood is a Dirichlet poste-
2443 rior (use Eq. B.5 and Eq. B.9 in Eq. B.11).

$$p(\theta|z, \alpha) = \frac{1}{B(\alpha + z)} \prod_i \theta_i^{z_i + \alpha_i - 1}. \quad (\text{B.12})$$

2444 Note again that we run i over the entries in our categorical variable z represented as a vector.
2445 This is different from a multinomial distribution over a set of variables!

2446 B.2 Dirichlet-Multinomial Distribution

2447 In case of an actual multinomial distribution, the topic of consideration are *counts* of z . Let's
2448 write the counts as $n(z)$.

$$p(z|\theta) = \frac{(\sum_k n(z_k))!}{\prod_k (n(z_k))!} \prod_k \theta_k^{n(z_k)}. \quad (\text{B.13})$$

2449 We now run over k unique variables, not over a vectorized categorical variable. The fol-
2450 lowing example clarifies the multinomial. Suppose we have a bag with k colored balls. The
2451 multinomial is the random variable that assigns a probability to the vector $[n(z_0), \dots, n(z_{k-1})]$
2452 with $n(z_i)$ the number of balls of color i . The distribution also represents the task of throwing
2453 a k -faced dice n times and counting the number of times $n(z)$ category z is occurring.

2454 The product of the Dirichlet distribution with the multinomial can be performed along the
 2455 same lines as in Appendix B.1.

$$p(z|\theta)p(\theta|\alpha) = \frac{\sum_k n(z_k) \Gamma(\sum_k n(z_k))}{\prod_k n(z_k) \prod_k \Gamma(n(z_k))} \frac{1}{B(\alpha)} \prod_k \theta_k^{n(z_k)+\alpha_k-1}. \quad (\text{B.14})$$

2456 This we can simplify to:

$$p(z|\theta)p(\theta|\alpha) = \frac{\sum_k n(z_k)}{\prod_k n(z_k)} \frac{1}{B(n(z_k))} \frac{1}{B(\alpha)} \prod_k \theta_k^{n(z_k)+\alpha_k-1}. \quad (\text{B.15})$$

2457 We will get the following result for the compound distribution (using Eq. B.7 with the Beta
 2458 function as known result for the integral):

$$p(z|\alpha) = \int p(z|\theta)p(\theta|\alpha)d\theta = \frac{\sum_k n(z_k)}{\prod_k n(z_k)} \frac{1}{B(n(z_k))} \frac{1}{B(\alpha)} B(\alpha + n(z_k)). \quad (\text{B.16})$$

2459 The Dirichlet prior with the multinomial likelihood results in a Dirichlet posterior as well:

$$p(\theta|z, \alpha) = \frac{p(z|\theta)p(\theta|\alpha)}{p(z|\alpha)} = \frac{1}{B(\alpha + n(z_k))} \prod_k \theta_k^{n(z_k)+\alpha_k-1}. \quad (\text{B.17})$$

2460 B.3 Dirichlet-N Categorical Distributions

2461 The third interpretation of a Dirichlet-multinomial distribution is the one in which the term
 2462 "multinomial" refers to a distribution of a *sequence* of categorical variables. Recall that the
 2463 multinomial assigns probabilities to the *number* of extracted balls (in an experiment getting
 2464 n balls out of a bag with k ball types). A sequence of categorical variables has a form for its
 2465 probability distribution that has no normalization factor:

$$p(z|\theta) = \prod_k \theta_k^{z_k}. \quad (\text{B.18})$$

2466 Here k runs over the categories. Note that this form is exactly the same as that of Eq. B.4.
 2467 The compound distribution will henceforth have the same form as the Dirichlet-categorical
 2468 distribution:

$$p(z|\alpha) = \int p(z|\theta)p(\theta|\alpha)d\theta = \frac{B(\alpha + z)}{B(\alpha)}. \quad (\text{B.19})$$

2469 The Dirichlet prior with the n-categorical distribution as likelihood results in a Dirichlet
 2470 posterior:

$$p(\theta|z, \alpha) = \frac{p(z|\theta)p(\theta|\alpha)}{p(z|\alpha)} = \frac{1}{B(\alpha + z)} \prod_k \theta_k^{z_k + \alpha_k - 1}. \quad (\text{B.20})$$



GIBBS SAMPLING

2471

2472

2473 Notation:

$$\int dF(x) = F(x) \quad (\text{C.1})$$

2474 A mixture model:

$$L(x) = \int dF(x) \mu(x) \quad (\text{C.2})$$

2475 If we have a particular form of $F(x)$, namely it admits a decomposition of a sum of individual
 2476 values x_i :

$$F(x) = \sum_i \delta_{x_i} = \delta(x = x_0) + \delta(x = x_1) + \dots \quad (\text{C.3})$$

2477 Then our mixture model can be written as:

$$L(x) = \int dF(x) \mu(x) = \sum_i \mu(x_i) \quad (\text{C.4})$$

2478 Let $x_0 = 3$, $x_1 = 4$, $\mu(x) = x^2$, then $L(x) = 3^2 + 4^2 = 25$.2479 Walker with $P = F$, $\mu(x) = N(y|\theta)$, $i = j$ and giving each θ_j a weight ω_j :

$$f_P(y) = \int dP(\theta) N(y|\theta) \quad P = \sum_j \omega_j \delta_{\theta_j} \quad (\text{C.5})$$

2480 Then:

$$f_{\omega,j}(y) = \sum_j \omega_j N(y|\theta_j) \quad (\text{C.6})$$

2481 The likelihood:

$$L(w | \alpha, \lambda_0) = p(\phi | \alpha) \prod_{i=0}^{N-1} \int p(w_i | \theta_i, \phi) dG_0(\theta_i) \quad (\text{C.7})$$

2482 Here the index runs over all data points w_i . Each data point corresponds to a line with
 2483 parameters θ_i . Here the parameters θ_i and θ_j for data point w_i and w_j can be the same and
 2484 thus reflect the same line. The index for θ runs over the N data points, not over the K lines.

2485 The distribution G_0 does have hyperparameters λ_0 .

2486 We can also group all data points that belong to the same line k together by reordering the
 2487 product terms:

$$L(w | \alpha, \lambda_0) = p(\phi | \alpha) \prod_k \prod_{i:z_i=k} \int p(w_i | \theta_i, \phi) dG_0(\theta_i) \quad (\text{C.8})$$

2488 Here the factors that belong to line k are multiplied. The index still runs over the data points.

2489 It is also possible not to limit the second product to only the data points i that are assigned
 2490 to line k .

$$L(w | \alpha, \lambda_0) = p(\phi | \alpha) \prod_k \prod_i p(z_i | \phi) \int p(w_i | \theta_i, \phi) dG_0(\theta_i) \quad (\text{C.9})$$

2491 Now, we are gonna introduce the stick-breaking sum, which turns our integral into a discrete
 2492 sum.

2493 $G = \sum_l p_l \delta_{Z_l}$ with Z_l iid from G_0 and p_l defined as a product of beta distributions.

$$L(w | \alpha, \lambda_0) = p(\phi | \alpha) \sum_k \prod_i p(z_i | \phi) p(w_i | \theta_k) p(\theta_k | \lambda_0) \quad (\text{C.10})$$

2494 The first term at the right hand side, $p(\phi | \alpha)$, generates the partition ϕ by the Dirichlet
 2495 process with concentration parameter α . The second term $p(z_i | \phi)$ defines indices z_0, \dots, z_N
 2496 to link observations w_0, \dots, w_N with the parameters $\theta_0, \dots, \theta_K$. The probability $p(w_i | \theta_k)$
 2497 corresponds to the likelihood equations 3.10 and 3.11 with w_i the tuple of x_i and y_i and
 2498 θ_k the line parameters σ_k^2 and β_k . The probability $p(\theta_k | \lambda_0)$ corresponds to the prior from
 2499 equation 3.14. The parameters θ_k (that is, σ_k^2 and β_k) are generated from hyperparameters
 2500 λ_0 . The hyperparameters $\lambda_0 = \{\mu_0, \Lambda_0, a, b\}$ are the parameters from the Normal-Inverse-
 2501 Gamma prior.

2502 The Dirichlet process can be used as a mixture model (Antoniak, 1974; Escobar and West,
 2503 1995; MacEachern and Müller, 1998) in which it generates (non-unique) parameters that
 2504 subsequently generate observations:

$$\begin{aligned} G &\sim DP(\alpha, G_0) \\ \theta_i &| G \sim G \\ w_i &| \theta_i \sim F(\theta_i) \end{aligned} \tag{C.11}$$

2505 Here F describes the mapping from parameters θ_i to observations w_i . It is possible to inte-
 2506 grate over G and sample the parameters directly from the base distribution G_0 .

2507 It is possible to integrate over G and get a description in the form of conditionals over the
 2508 parameters (Blackwell and MacQueen, 1973):

$$\theta_{n+1} | \theta_1 \dots \theta_{n-1} \sim \frac{1}{\alpha + n} (\alpha G_0 + \sum_{i=1}^n \delta_{\theta_i}) \tag{C.12}$$

2509 C.1 Gibbs Sampling of Parameters

2510 Algorithm, we will draw $\theta_i | \theta_{-i}, y_i$ for all i .

2511 And that continuously. So, that's how we get theta.

2512 Gibbs sampling requires the conditional probabilities of all entities involved (Geman and
 2513 Geman, 1984). Gibbs sampling just as other Markov chain Monte Carlo methods generates
 2514 a sequence of correlated samples. Subsequently, if necessary, the Maximum A Posteriori
 2515 estimation of a value can be found through picking the mode (most common occurring
 2516 value) of a parameter.

2517 The derivation of the conditional probabilities of parameters with respect to the remain-
 2518 ing parameters has been described in the literature (Neal, 2000). Such a derivation uses
 2519 an important property of the Dirichlet process, namely that it is the conjugate prior of the
 2520 multinomial distribution. Thanks to conjugacy the following equations have closed-form
 2521 descriptions. The conditional probabilities are sampled from the base distribution G_0 and
 2522 the other parameters θ_i in the following way:

$$\theta_{n+1} | \theta_1 \dots \theta_{n-1} \sim \frac{1}{\alpha + n} (\alpha G_0 + \sum_{i=1}^n \delta_{\theta_i}) \tag{C.13}$$

2523 If we include the observations themselves, we need to include the likelihood as well:

$$\theta_i | \theta_{-i}, w_i \sim C \left\{ \sum_{j, j \neq i} F(w_i, \theta_j) \delta_{\theta_j} + \alpha H_i \int F(w_i, \theta) dG_0(\theta) \right\} \tag{C.14}$$

2524 The constant C is a normalization factor to make the above a proper probability density
2525 (summing to one). The entity H_i is the posterior density of θ given G_0 as prior and y_i as
2526 observation. The notation θ_{-i} describes the set of all parameters Θ with θ_i excluded. The
2527 integral over $dG_0(\theta)$ is a Lebesgue-Stieltjes integral that weighs the contribution of $F(w_i, \theta)$
2528 with the base distribution $G_0(\theta)$.

2529 Equation C.14 can be used to perform inference directly with all (non-unique) parameters
2530 θ_i tied to observations w_i . Details on inference will be provided in Sect. ??.



GLOSSARY

2531

2532

2533

2534 **burn-in** running a Markov chain Monte Carlo for a while before starting to sample from it,
2535 so the results are not depending on its initial random starting position. 42

2536 **collapsed Gibbs sampling** Rao-Blackwellized Gibbs sampling. Certain sampling steps are
2537 replaced by steps where one or more variables are integrated out. This is can be thanks
2538 to analytic descriptions that arise from the use of conjugate priors. 42

2539 **slow mixing** a high degree of correlation between subsequent samples in a Markov chain
2540 leading to a long time before the chain converges. 57

LIST OF FIGURES

2542	2.1	Probability measure	11
2543	2.2	Generative vs Discriminative	21
2544	2.3	Plate notation	24
2545	2.4	Stick-breaking representation	27
2546	2.5	Matrix Representation	29
2547	2.6	Chinese Restaurant Process	30
2548	2.7	Dirichlet Process	31
2549	2.8	Beta Process	32
2550	2.9	Matrix Representation	32
2551	2.10	Gamma Process	34
2552	2.11	Matrix Representation	35
2553	2.12	The difference visualized between a Dirichlet Process mixture and a hierar-	
2554		chical Dirichlet process. It illustrates also that the input of a Dirichlet process	
2555		does not have to be a continuous function. If it is a continuous distribution	
2556		it will become a discrete distributed almost surely. If it is a discrete distribu-	
2557		tion, it will have atoms at the locations where the discrete distribution had its	
2558		probability mass concentrated.	37
2559	2.13	Rejection Sampling	40
2560	3.1	The infinite line model using the Chinese restaurant process representation	
2561		(compare with Figure 2.6). Top: α , the concentration parameter of the Dirich-	
2562		let process. Bottom, left to right: w_i , the observation, an individual point in	
2563		a 2D space; θ_i , the parameters (intercept, slope) of the line belonging to ob-	
2564		servation w_i ; H , the base distribution from which line parameter values are	
2565		sampled.	48
2566	3.2	The infinite line model in the stick-breaking representation (compare with	
2567		Figure 3.1). From left to right: α , the concentration parameter of the Dirich-	
2568		let process; (ϕ_1, \dots, ϕ_k) , the partition of points over lines; z_i , the assignment	
2569		parameters that link observation w_i with line k ; w_i , the observation, an indi-	
2570		vidual point with x and y coordinates; θ_k , the parameters of line k ; λ_0 , the	
2571		base measure from which the line parameter values are sampled.	55
2572	3.3	The performance of Algorithm 8 with respect to clustering is measured using	
2573		the Rand Index, the Adjusted Rand Index, the Mirvin metric, and the Hubert	
2574		metric. A score of 1 means perfect clustering for all metrics, except Mirvin's	
2575		where 0 denotes perfect clustering.	57

2576	3.4	The performance of Algorithm 9 with respect to clustering is measured using the Rand Index, the Adjusted Rand Index, the Mirvin metric, and the Hubert metric. A score of 1 means perfect clustering for all metrics, except Mirvin's where 0 denotes perfect clustering.	58
2577			
2578			
2579			
2580	3.5	One of the Gibbs steps in the inference of two particular lines. The points are roughly distributed according to the lines, but one line exists out of two large clusters. The line coordinates are visualized by a double circle. The x-coordinate is the y-intercept of the line, the y-coordinate is the slope. . . .	59
2581			
2582			
2583			
2584	3.6	The assignment of a line to a single point. In the figure, there are three clusters found, rather than only the two obvious clusters.	59
2585			
2586			
2587	4.1	Sampling of Pareto pairs. The parameters are $\lambda_m = -4$, $\lambda_n = 2$, $k = 5$, and we have sampled $N = 1000$ pairs. The position parameters λ_m and λ_n define the positions of the endpoints. The shape parameter k defines the variance in the exact positions on both sides. The Pareto pairs always sample endpoints both at the "left" and the "right" (not both at the left or the right side). . . .	63
2588			
2589			
2590			
2591	4.2	Consider (1) the data uniformly distributed on a line segment and (2) a symmetric Pareto prior for both endpoints, then we can update the estimate for the endpoints given the data as visualized. Each subfigure shows an adjustment of the endpoints given more data points (1, 3, 10, and 100 data points). The y-axis does not have a significance in these plots.	65
2592			
2593			
2594			
2595			
2596	4.3	The Bayesian linear regression model for multiple line segments in plate notation is the same as for the Infinite Line Model. The Dirichlet process is defined at the left with concentration parameter α . It generates the partitions (π_1, \dots, π_k) with assignment parameters z_i that denote which observation w_i belongs to which cluster k . The cluster is summarized through the parameter set θ_k and has λ_0 as its hyperparameter. The parameter set θ_k includes parameters that signify the line itself such as slope and y-intercept, plus the parameters that denote the extent of the segment.	66
2597			
2598			
2599			
2600			
2601			
2602			
2603			
2604	4.4	Line segments generated through a Dirichlet Process. The line parameters are generated from a Normal-Inverse-Gamma distribution. The endpoints of the line segment are generated from a symmetric Pareto distribution. The points are generated uniformly over the line segments.	67
2605			
2606			
2607			
2608	4.5	Bayesian point estimates of the sampling process with varying outcomes. . .	69
2609	4.6	Segment detection performs much worse than line detection across all three clustering performance indicators. Perfect clustering is indicated by 1.0 for Rand Index, Adjusted Rand Index, and Hubert.	70
2610			
2611			
2612	5.1	Dyadic vs triadic MCMC	76
2613	5.2	Two examples of fitting a mixture of lines to data items scattered over a two-dimensional space. The lines drawn are inferred using one of the methods in this chapter. The lines are not the ground truth, but are meant to demonstrate the typical errors made by fitting methods. Note for example that there are mistakes in both the assignment of points to lines as well as the line parameters (slope and intercept).	80
2614			
2615			
2616			
2617			
2618			
2619	5.3	The same results as in Table 5.1, but visualized in a violin plot. The distribution over metric values are displayed in a vertical fashion. From left to right the distribution shifts to one, signifying better clustering performance. . . .	81
2620			
2621			
2622	6.1	Generative Adversarial Network	85

2623	6.2	Varational Autoencoder	85
2624	7.1	The likelihood function for the moments at which people decide to exercise	
2625		during the day. On the horizontal axis time, on the vertical axis the frequency	
2626		of exercising.	91
2627	7.2	The improved likelihood function for the moments at which people decide to	
2628		exercise during the day using the Von Mises distribution. On the horizontal	
2629		axis time, on the vertical axis the frequency of exercising.	92
2630	7.3	After running the algorithm, the above figures show the different categories	
2631		of runners that have been found.	95
2632	A.1	Beta distribution	111

LIST OF TABLES

2634	2.1	Structures and Processes	24
2635	2.2	Levy measure	25
2636	2.3	Exchangeable structures	26
2637	5.1	The purity, rand index, and adjusted rand index establishing the quality of the clustering method. The closer the values to one, the better the method performed. The purity metric assigns high values to clusters that do not have data points from other clusters (but does not penalize the number of clusters). The rand index index computes similarity between clusters taking false negatives and false positives into account. The adjusted rand index accounts for chance. The adjusted rand index is most useful in our comparison.	81
2644	7.1	Example of the type of data about the timing of exercising. A person is represented by row, her preferences by column. There is not a predefined number of users or groups of users.	90
2647	7.2	Top row: A sequence of cluster indices indicates the ground truth. Each cluster index represents a multi-modal Von-Mises-Uniform distribution with different parameters θ . Bottom row: A sequence of cluster indices that are the result of the described algorithm. Each cluster index represents again a multi-modal Von-Mises distribution. Errors would be represented by an inconsistent mapping from the top row to the bottom row.	94

2653 Acronyms

- 2654 **ABC** approximate Bayesian computation
- 2655 **a.s.** almost surely
- 2656 **BP** Beta process
- 2657 **CRP** Chinese restaurant process
- 2658 **DMM** Dirichlet mixture model
- 2659 **DP** Dirichlet process
- 2660 **GP** Gamma process
- 2661 **GEM** Griffiths, Engen, and McCloskey
- 2662 **HDP** hierarchical Dirichlet process
- 2663 **IBP** Indian buffet process
- 2664 **ILM** infinite line model
- 2665 **MAP** maximum a posteriori
- 2666 **MCMC** Markov chain Monte Carlo
- 2667 **ML** maximum likelihood
- 2668 **MLL** maximum log-likelihood
- 2669 **NB** naive Bayes
- 2670 **NIG** Normal-Inverse-Gamma
- 2671 **PYP** Pitman-Yor process

2672

SUMMARY

2673 Summary....

2674

SAMENVATTING

2675 Samenvatting...

2676

ACKNOWLEDGMENTS

2677 I want to thank...

2678

CURRICULUM VITAE

2679 Anne van Rossum

2680

PUBLICATIONS

2681 The investigations performed during my Ph.D. research resulted in the following publica-
2682 tions.

2683 ◦ A.C. van Rossum.

SIKS DISSERTATION SERIES

2685	1998	2716	2000
2686	1 Johan van den Akker (CWI ¹) <i>DEGAS - An Active</i>	2717	1 Frank Niessink (VU) <i>Perspectives on Improving Soft-</i>
2687	<i>Temporal Database of Autonomous Objects</i>	2718	<i>ware Maintenance</i>
2688	2 Floris Wiesman (UM) <i>Information Retrieval by</i>	2719	2 Koen Holtman (TU/e) <i>Prototyping of CMS Storage</i>
2689	<i>Graphically Browsing Meta-Information</i>	2720	<i>Management</i>
2690	3 Ans Steuten (TUD) <i>A Contribution to the Linguistic</i>	2721	3 Carolien M.T. Metselaar (UvA) <i>Sociaal-</i>
2691	<i>Analysis of Business Conversations within the Lan-</i>	2722	<i>organisatorische Gevolgen van Kennistechnologie;</i>
2692	<i>guage/Action Perspective</i>	2723	<i>een Procesbenadering en Actorperspectief</i>
2693	4 Dennis Breuker (UM) <i>Memory versus Search in</i>	2724	4 Geert de Haan (VU) <i>ETAG, A Formal Model of Com-</i>
2694	<i>Games</i>	2725	<i>petence Knowledge for User Interface Design</i>
2695	5 Eduard W. Oskamp (RUL) <i>Computerondersteuning</i>	2726	5 Ruud van der Pol (UM) <i>Knowledge-Based Query</i>
2696	<i>bij Straftoemeting</i>	2727	<i>Formulation in Information Retrieval</i>
2697	1999	2728	6 Rogier van Eijk (UU) <i>Programming Languages for</i>
		2729	<i>Agent Communication</i>
		2730	7 Niels Peek (UU) <i>Decision-Theoretic Planning of Clin-</i>
2698	1 Mark Sloof (VU) <i>Physiology of Quality Change Mod-</i>	2731	<i>ical Patient Management</i>
2699	<i>elling; Automated Modelling of Quality Change of</i>	2732	8 Veerle Coupé (EUR) <i>Sensitivity Analysis of Decision-</i>
2700	<i>Agricultural Products</i>	2733	<i>Theoretic Networks</i>
2701	2 Rob Potharst (EUR) <i>Classification using Decision</i>	2734	9 Florian Waas (CWI) <i>Principles of Probabilistic</i>
2702	<i>Trees and Neural Nets</i>	2735	<i>Query Optimization</i>
2703	3 Don Beal (UM) <i>The Nature of Minimax Search</i>	2736	10 Niels Nes (CWI) <i>Image Database Management Sys-</i>
2704	4 Jacques Penders (UM) <i>The Practical Art of Moving</i>	2737	<i>tem Design Considerations, Algorithms and Architec-</i>
2705	<i>Physical Objects</i>	2738	<i>ture</i>
2706	5 Aldo de Moor (KUB) <i>Empowering Communities: A</i>	2739	11 Jonas Karlsson (CWI) <i>Scalable Distributed Data</i>
2707	<i>Method for the Legitimate User-Driven Specification</i>	2740	<i>Structures for Database Management</i>
2708	<i>of Network Information Systems</i>		
2709	6 Niek J.E. Wijngaards (VU) <i>Re-Design of Composi-</i>	2741	2001
2710	<i>tional Systems</i>		1 Silja Renooij (UU) <i>Qualitative Approaches to Quan-</i>
2711	7 David Spelt (UT) <i>Verification Support for Object</i>	2742	<i>tifying Probabilistic Networks</i>
2712	<i>Database Design</i>	2743	2 Koen Hindriks (UU) <i>Agent Programming Lan-</i>
2713	8 Jacques H.J. Lenting (UM) <i>Informed Gambling;</i>	2744	<i>guages: Programming with Mental Models</i>
2714	<i>Conception and Analysis of a Multi-Agent Mecha-</i>	2745	
2715	<i>nism for Discrete Reallocation</i>		

¹Abbreviations: SIKS - Dutch Research School for Information and Knowledge Systems; CWI - Centrum voor Wiskunde en Informatica, Amsterdam; EUR - Erasmus Universiteit, Rotterdam; KUB - Katholieke Universiteit Brabant, Tilburg; KUN - Katholieke Universiteit Nijmegen; OU - Open Universiteit; RUL - Rijksuniversiteit Leiden; RUN - Radboud Universiteit Nijmegen; TUD - Technische Universiteit Delft; TU/e - Technische Universiteit Eindhoven; UL - Universiteit Leiden; UM - Universiteit Maastricht; UT - Universiteit Twente, Enschede; UU - Universiteit Utrecht; UvA - Universiteit van Amsterdam; UvT - Universiteit van Tilburg; VU - Vrije Universiteit, Amsterdam.

- 2746 3 Maarten van Someren (UvA) *Learning as Problem Solving* 2802
2747 2803
- 2748 4 Evgueni Smirnov (UM) *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets* 2804
2749 2805
- 2750 5 Jacco van Ossenbruggen (VU) *Processing Structured Hypermedia: A Matter of Style* 2806
2751 2807
- 2752 6 Martijn van Welie (VU) *Task-Based User Interface Design* 2808
2753 2808
- 2754 7 Bastiaan Schonhage (VU) *Diva: Architectural Perspectives on Information Visualization* 2809
2755 2809
- 2756 8 Pascal van Eck (VU) *A Compositional Semantic Structure for Multi-Agent Systems Dynamics* 2810
2757 2811
- 2758 9 Pieter Jan 't Hoen (RUL) *Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes* 2812
2759 2813
2760 2814
- 2761 10 Maarten Sierhuis (UvA) *Modeling and Simulating Work Practice BRAHMS: a Multiagent Modeling and Simulation Language for Work Practice Analysis and Design* 2815
2762 2816
2763 2817
2764 2818
- 2765 11 Tom M. van Engers (VU) *Knowledge Management: The Role of Mental Models in Business Systems Design* 2819
2766 2820
2767 2821
- 2768 **2002** 2822
2823
2824
- 2769 1 Nico Lassing (VU) *Architecture-Level Modifiability Analysis* 2825
2770 2826
- 2771 2 Roelof van Zwol (UT) *Modelling and Searching Web-based Document Collections* 2827
2772 2828
- 2773 3 Henk Ernst Blok (UT) *Database Optimization Aspects for Information Retrieval* 2829
2774 2830
- 2775 4 Juan Roberto Castelo Valdueza (UU) *The Discrete Acyclic Digraph Markov Model in Data Mining* 2831
2776 2832
- 2777 5 Radu Serban (VU) *The Private Cyberspace Modeling Electronic Environments Inhabited by Privacy-Concerned Agents* 2833
2778 2834
2779 2835
- 2780 6 Laurens Mommers (UL) *Applied Legal Epistemology; Building a Knowledge-based Ontology of the Legal Domain* 2836
2781 2837
2782 2838
- 2783 7 Peter Boncz (CWI) *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications* 2839
2784 2840
- 2785 8 Jaap Gordijn (VU) *Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas* 2841
2786 2842
- 2787 9 Willem-Jan van den Heuvel (KUB) *Integrating Modern Business Applications with Objectified Legacy Systems* 2843
2788 2844
2789 2845
- 2790 10 Brian Sheppard (UM) *Towards Perfect Play of Scrabble* 2846
2791 2847
- 2792 11 Wouter C.A. Wijngaards (VU) *Agent Based Modelling of Dynamics: Biological and Organisational Applications* 2848
2793 2849
2794 2849
- 2795 12 Albrecht Schmidt (UvA) *Processing XML in Database Systems* 2850
2796 2850
- 2797 13 Hongjing Wu (TU/e) *A Reference Architecture for Adaptive Hypermedia Applications* 2851
2798 2852
- 2799 14 Wieke de Vries (UU) *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems* 2853
2800 2854
2801 2855
- 15 Rik Eshuis (UT) *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*
- 16 Pieter van Langen (VU) *The Anatomy of Design: Foundations, Models and Applications*
- 17 Stefan Manegold (UvA) *Understanding, Modeling, and Improving Main-Memory Database Performance*
- 2003**
- 1 Heiner Stuckenschmidt (VU) *Ontology-Based Information Sharing in Weakly Structured Environments*
- 2 Jan Broersen (VU) *Modal Action Logics for Reasoning About Reactive Systems*
- 3 Martijn Schuemie (TUD) *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*
- 4 Milan Petkovic (UT) *Content-Based Video Retrieval Supported by Database Technology*
- 5 Jos Lehmann (UvA) *Causation in Artificial Intelligence and Law – A Modelling Approach*
- 6 Boris van Schooten (UT) *Development and Specification of Virtual Environments*
- 7 Machiel Jansen (UvA) *Formal Explorations of Knowledge Intensive Tasks*
- 8 Yong-Ping Ran (UM) *Repair-Based Scheduling*
- 9 Rens Kortmann (UM) *The Resolution of Visually Guided Behaviour*
- 10 Andreas Lincke (UT) *Electronic Business Negotiation: Some Experimental Studies on the Interaction between Medium, Innovation Context and Culture*
- 11 Simon Keizer (UT) *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*
- 12 Roeland Ordelman (UT) *Dutch Speech Recognition in Multimedia Information Retrieval*
- 13 Jeroen Donkers (UM) *Nosce Hostem – Searching with Opponent Models*
- 14 Stijn Hoppenbrouwers (KUN) *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*
- 15 Mathijs de Weerd (TUD) *Plan Merging in Multi-Agent Systems*
- 16 Menzo Windhouwer (CWI) *Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouse*
- 17 David Jansen (UT) *Extensions of Statecharts with Probability, Time, and Stochastic Timing*
- 18 Levente Kocsis (UM) *Learning Search Decisions*
- 2004**
- 1 Virginia Dignum (UU) *A Model for Organizational Interaction: Based on Agents, Founded in Logic*
- 2 Lai Xu (UvT) *Monitoring Multi-party Contracts for E-business*
- 3 Perry Groot (VU) *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*

- 2856 4 Chris van Aart (UvA) *Organizational Principles for* 2910
2857 *Multi-Agent Architectures* 2911
2858 5 Viara Popova (EUR) *Knowledge Discovery and* 2912
2859 *Monotonicity* 2913
2860 6 Bart-Jan Hommes (TUD) *The Evaluation of Busi-* 2914
2861 *ness Process Modeling Techniques* 2915
2862 7 Elise Boltjes (UM) *Voorbeeld_{IG} Onderwijs; Voor-* 2916
2863 *beeldgestuurd Onderwijs, een Opstap naar Abstract* 2917
2864 *Denken, vooral voor Meisjes* 2918
2865 8 Joop Verbeek (UM) *Politie en de Nieuwe Inter-* 2919
2866 *nationale Informatiemarkt, Grensregionale Politie* 2920
2867 *Gegevensuitwisseling en Digitale Expertise* 2921
2868 9 Martin Caminada (VU) *For the Sake of the Argu-* 2922
2869 *ment; Explorations into Argument-based Reasoning* 2923
2870 10 Suzanne Kabel (UvA) *Knowledge-rich Indexing of* 2924
2871 *Learning-objects* 2925
2872 11 Michel Klein (VU) *Change Management for Dis-* 2926
2873 *tributed Ontologies* 2927
2874 12 The Duy Bui (UT) *Creating Emotions and Facial Ex-* 2928
2875 *pressions for Embodied Agents* 2929
2876 13 Wojciech Jamroga (UT) *Using Multiple Models of* 2930
2877 *Reality: On Agents who Know how to Play* 2931
2878 14 Paul Harrenstein (UU) *Logic in Conflict. Logical Ex-* 2932
2879 *plorations in Strategic Equilibrium* 2933
2880 15 Arno Knobbe (UU) *Multi-Relational Data Mining* 2934
2881 16 Federico Divina (VU) *Hybrid Genetic Relational* 2935
2882 *Search for Inductive Learning*
2883 17 Mark Winands (UM) *Informed Search in Complex* 2936
2884 *Games*
2885 18 Vania Bessa Machado (UvA) *Supporting the Con-* 2937
2886 *struction of Qualitative Knowledge Models* 2938
2887 19 Thijs Westerveld (UT) *Using generative probabilis-* 2939
2888 *tic models for multimedia retrieval* 2940
2889 20 Madelon Evers (Nyenrode) *Learning from Design* 2941
2890 *facilitating multidisciplinary design teams* 2942
2943
2891 **2005** 2944
2945
2892 1 Floor Verdenius (UvA) *Methodological Aspects of* 2947
2893 *Designing Induction-Based Applications* 2948
2894 2 Erik van der Werf (UM) *AI techniques for the game* 2949
2895 *of Go* 2950
2896 3 Franc Grootjen (RUN) *A Pragmatic Approach to the* 2951
2897 *Conceptualisation of Language* 2952
2898 4 Nirvana Meratnia (UT) *Towards Database Support* 2953
2899 *for Moving Object data* 2954
2900 5 Gabriel Infante-Lopez (UvA) *Two-Level Probabilis-* 2955
2901 *tic Grammars for Natural Language Parsing* 2956
2902 6 Pieter Spronck (UM) *Adaptive Game AI* 2957
2903 7 Flavius Frasinca (TU/e) *Hypermedia Presentation* 2958
2904 *Generation for Semantic Web Information Systems* 2959
2905 8 Richard Vdovjak (TU/e) *A Model-driven Approach* 2960
2906 *for Building Distributed Ontology-based Web Appli-* 2961
2907 *cations* 2962
2908 9 Jeen Broekstra (VU) *Storage, Querying and Infer-* 2963
2909 *encing for Semantic Web Languages* 2964
2965
10 Anders Bouwer (UvA) *Explaining Behaviour: Using*
Qualitative Simulation in Interactive Learning Envi-
ronments
11 Elth Ogston (VU) *Agent Based Matchmaking and*
Clustering - A Decentralized Approach to Search
12 Csaba Boer (EUR) *Distributed Simulation in Indus-*
try
13 Fred Hamburg (UL) *Een Computermodel voor het*
Ondersteunen van Euthanasiebeslissingen
14 Borys Omelayenko (VU) *Web-Service configuration*
on the Semantic Web; Exploring how semantics
meets pragmatics
15 Tibor Bosse (VU) *Analysis of the Dynamics of Cog-*
nitive Processes
16 Joris Graaumanns (UU) *Usability of XML Query Lan-*
guages
17 Boris Shishkov (TUD) *Software Specification Based*
on Re-usable Business Components
18 Danielle Sent (UU) *Test-selection strategies for prob-*
abilistic networks
19 Michel van Dartel (UM) *Situated Representation*
20 Cristina Coteanu (UL) *Cyber Consumer Law, State*
of the Art and Perspectives
21 Wijnand Derks (UT) *Improving Concurrency and*
Recovery in Database Systems by Exploiting Appli-
cation Semantics
2006
1 Samuil Angelov (TU/e) *Foundations of B2B Elec-*
tronic Contracting
2 Cristina Chisalita (VU) *Contextual issues in the de-*
sign and use of information technology in organiza-
tions
3 Noor Christoph (UvA) *The role of metacognitive*
skills in learning to solve problems
4 Marta Sabou (VU) *Building Web Service Ontologies*
5 Cees Pierik (UU) *Validation Techniques for Object-*
Oriented Proof Outlines
6 Ziv Baida (VU) *Software-aided Service Bundling -*
Intelligent Methods & Tools for Graphical Service
Modeling
7 Marko Smiljanic (UT) *XML schema matching – bal-*
ancing efficiency and effectiveness by means of clus-
tering
8 Eelco Herder (UT) *Forward, Back and Home Again*
- Analyzing User Behavior on the Web
9 Mohamed Wahdan (UM) *Automatic Formulation of*
the Auditor's Opinion
10 Ronny Siebes (VU) *Semantic Routing in Peer-to-Peer*
Systems
11 Joeri van Ruth (UT) *Flattening Queries over Nested*
Data Types
12 Bert Bongers (VU) *Interactivation - Towards an e-*
cology of people, our technological environment, and
the arts
13 Henk-Jan Lebbink (UU) *Dialogue and Decision*
Games for Information Exchanging Agents

- 2966 14 Johan Hoorn (VU) *Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change* 3020-3022
- 2967
- 2968
- 2969 15 Rainer Malik (UU) *CONAN: Text Mining in the Biomedical Domain* 3023-3024
- 2970
- 2971 16 Carsten Riggelsen (UU) *Approximation Methods for Efficient Learning of Bayesian Networks* 3025-3026
- 2972
- 2973 17 Stacey Nagata (UU) *User Assistance for Multitasking with Interruptions on a Mobile Device* 3027-3028
- 2974
- 2975 18 Valentin Zhizhkun (UvA) *Graph transformation for Natural Language Processing* 3029-3030
- 2976
- 2977 19 Birna van Riemsdijk (UU) *Cognitive Agent Programming: A Semantic Approach* 3031-3032
- 2978
- 2979 20 Marina Velikova (UvT) *Monotone models for prediction in data mining* 3033-3034
- 2980
- 2981 21 Bas van Gils (RUN) *Aptness on the Web* 3035-3036
- 2982
- 2983 22 Paul de Vrieze (RUN) *Fundamentals of Adaptive Personalisation* 3037-3038
- 2984
- 2985 23 Ion Juvina (UU) *Development of Cognitive Model for Navigating on the Web* 3039-3040
- 2986
- 2987 24 Laura Hollink (VU) *Semantic Annotation for Retrieval of Visual Resources* 3041-3042
- 2988
- 2989 25 Madalina Drugan (UU) *Conditional log-likelihood MDL and Evolutionary MCMC* 3043-3044
- 2990
- 2991 26 Vojkan Mihajlovic (UT) *Score Region Algebra: A Flexible Framework for Structured Information Retrieval* 3045-3046
- 2992
- 2993 27 Stefano Bocconi (CWI) *Vox Populi: generating video documentaries from semantically annotated media repositories* 3047-3048
- 2994
- 2995
- 2996 28 Borkur Sigurbjornsson (UvA) *Focused Information Access using XML Element Retrieval* 3049-3050
- 2997
- 2998 **2007** 3051-3052
- 2999 1 Kees Leune (UvT) *Access Control and Service-Oriented Architectures* 3053-3054
- 3000
- 3001 2 Wouter Teepe (RUG) *Reconciling Information Exchange and Confidentiality: A Formal Approach* 3055-3056
- 3002
- 3003 3 Peter Mika (VU) *Social Networks and the Semantic Web* 3057-3058
- 3004
- 3005 4 Jurriaan van Diggelen (UU) *Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach* 3059-3060
- 3006
- 3007
- 3008 5 Bart Schermer (UL) *Software Agents, Surveillance and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance* 3061-3062
- 3009
- 3010
- 3011 6 Gilad Mishne (UvA) *Applied Text Analytics for Blogs* 3063-3064
- 3012
- 3013 7 Natasa Jovanovic' (UT) *To Whom It May Concern Addressee Identification in Face-to-Face Meetings* 3065-3066
- 3014
- 3015 8 Mark Hoogendoorn (VU) *Modeling of Change in Multi-Agent Organizations* 3067-3068
- 3016
- 3017 9 David Mobach (VU) *Agent-Based Mediated Service Negotiation* 3069-3070
- 3018
- 3019 10 Huib Aldewereld (UU) *Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols* 3071-3072
- 3020
- 3021
- 3022
- 3023
- 3024
- 3025
- 3026
- 3027
- 3028
- 3029
- 3030
- 3031
- 3032
- 3033
- 3034
- 3035
- 3036
- 3037
- 3038
- 3039
- 3040
- 3041
- 3042
- 3043
- 3044
- 3045
- 3046
- 3047
- 3048
- 3049
- 3050
- 3051
- 3052
- 3053
- 3054
- 3055
- 3056
- 3057
- 3058
- 3059
- 3060
- 3061
- 3062
- 3063
- 3064
- 3065
- 3066
- 3067
- 3068
- 3069
- 3070
- 3071
- 3072
- 3073
- 3074
- 3075
- 11 Natalia Stash (TU/e) *Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System*
- 12 Marcel van Gerven (RUN) *Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty*
- 13 Rutger Rienks (UT) *Meetings in Smart Environments; Implications of Progressing Technology*
- 14 Niek Bergboer (UM) *Context-Based Image Analysis*
- 15 Joyca Lacroix (UM) *NIM: a Situated Computational Memory Model*
- 16 Davide Grossi (UU) *Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems*
- 17 Theodore Charitos (UU) *Reasoning with Dynamic Networks in Practice*
- 18 Bart Orriens (UvT) *On the development and management of adaptive business collaborations*
- 19 David Levy (UM) *Intimate relationships with artificial partners*
- 20 Slinger Jansen (UU) *Customer Configuration Updating in a Software Supply Network*
- 21 Karianne Vermaas (UU) *Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005*
- 22 Zlatko Zlatev (UT) *Goal-oriented design of value and process models from patterns*
- 23 Peter Barna (TU/e) *Specification of Application Logic in Web Information Systems*
- 24 Georgina Ramírez Camps (CWI) *Structural Features in XML Retrieval*
- 25 Joost Schalken (VU) *Empirical Investigations in Software Process Improvement*
- 2008**
- 1 Katalin Boer-Sorbán (EUR) *Agent-Based Simulation of Financial Markets: A modular, continuous-time approach*
- 2 Alexei Sharpanskykh (VU) *On Computer-Aided Methods for Modeling and Analysis of Organizations*
- 3 Vera Hollink (UvA) *Optimizing hierarchical menus: a usage-based approach*
- 4 Ander de Keijzer (UT) *Management of Uncertain Data - towards unattended integration*
- 5 Bela Mutschler (UT) *Modeling and simulating causal dependencies on process-aware information systems from a cost perspective*
- 6 Arjen Hommersom (RUN) *On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective*
- 7 Peter van Rosmalen (OU) *Supporting the tutor in the design and support of adaptive e-learning*
- 8 Janneke Bolt (UU) *Bayesian Networks: Aspects of Approximate Inference*
- 9 Christof van Nimwegen (UU) *The paradox of the guided user: assistance can be counter-effective*

- 3076 10 Wauter Bosma (UT) *Discourse oriented Summariza-* 3134
3077 *tion* 3135
- 3078 11 Vera Kartseva (VU) *Designing Controls for Network*
3079 *Organizations: a Value-Based Approach* 3136
- 3080 12 Jozsef Farkas (RUN) *A Semiotically oriented Cogni-*
3081 *tive Model of Knowledge Representation* 3137
- 3082 13 Caterina Carraciolo (UvA) *Topic Driven Access to* 3138
3083 *Scientific Handbooks* 3139
- 3084 14 Arthur van Bunningen (UT) *Context-Aware Query-* 3140
3085 *ing; Better Answers with Less Effort* 3141
- 3086 15 Martijn van Otterlo (UT) *The Logic of Adaptive Be-* 3142
3087 *havior: Knowledge Representation and Algorithms* 3143
3088 *for the Markov Decision Process Framework in First-* 3144
3089 *Order Domains* 3145
- 3090 16 Henriette van Vugt (VU) *Embodied Agents from a* 3146
3091 *User's Perspective* 3147
- 3092 17 Martin Op't Land (TUD) *Applying Architecture and* 3148
3093 *Ontology to the Splitting and Allying of Enterprises* 3149
- 3094 18 Guido de Croon (UM) *Adaptive Active Vision* 3150
- 3095 19 Henning Rode (UT) *From document to entity re-* 3151
3096 *trieval: improving precision and performance of fo-* 3152
3097 *cused text search* 3153
- 3098 20 Rex Arendsen (UvA) *Geen bericht, goed bericht. Een* 3154
3099 *onderzoek naar de effecten van de introductie van* 3155
3100 *elektronisch berichtenverkeer met een overheid op de* 3156
3101 *administratieve lasten van bedrijven* 3157
- 3102 21 Krisztian Balog (UvA) *People search in the enter-* 3158
3103 *prise* 3159
- 3104 22 Henk Koning (UU) *Communication of IT* 3160
3105 *architecture* 3161
- 3106 23 Stefan Visscher (UU) *Bayesian network models for* 3162
3107 *the management of ventilator-associated pneumonia* 3163
- 3108 24 Zharko Aleksovski (VU) *Using background knowl-* 3164
3109 *edge in ontology matching* 3165
- 3110 25 Geert Jonker (UU) *Efficient and Equitable exchange* 3166
3111 *in air traffic management plan repair using spender-* 3167
3112 *signed currency* 3168
- 3113 26 Marijn Huijbregts (UT) *Segmentation, diarization* 3169
3114 *and speech transcription: surprise data unraveled* 3170
- 3115 27 Hubert Vogten (OU) *Design and implementation* 3171
3116 *strategies for IMS learning design* 3172
- 3117 28 Ildiko Flesh (RUN) *On the use of independence re-* 3173
3118 *lations in Bayesian networks* 3174
- 3119 29 Dennis Reidsma (UT) *Annotations and subjective* 3175
3120 *machines- Of annotators, embodied agents, users,* 3176
3121 *and other humans* 3177
- 3122 30 Wouter van Atteveldt (VU) *Semantic network anal-* 3178
3123 *ysis: techniques for extracting, representing and* 3179
3124 *querying media content* 3180
- 3125 31 Loes Braun (UM) *Pro-active medical information re-* 3181
3126 *trieval* 3182
- 3127 32 Trung B. Hui (UT) *Toward affective dialogue man-* 3183
3128 *agement using partially observable markov decision* 3184
3129 *processes* 3185
- 3130 33 Frank Terpstra (UvA) *Scientific workflow design,* 3186
3131 *theoretical and practical issues* 3187
- 3132 34 Jeroen de Knijf (UU) *Studies in Frequent Tree Min-* 3188
3133 *ing* 3189
- 35 Benjamin Torben-Nielsen (UvT) *Dendritic mor-*
phology: function shapes structure
- 2009**
- 1 Rasa Jurgelenaite (RUN) *Symmetric Causal Inde-*
pendence Models
- 2 Willem Robert van Hage (VU) *Evaluating Ontology-*
Alignment Techniques
- 3 Hans Stol (UvT) *A Framework for Evidence-based*
Policy Making Using IT
- 4 Josephine Nabukenya (RUN) *Improving the Qual-*
ity of Organisational Policy Making using Collabo-
ration Engineering
- 5 Sietse Overbeek (RUN) *Bridging Supply and De-*
mand for Knowledge Intensive Tasks - Based on
Knowledge, Cognition, and Quality
- 6 Muhammad Subianto (UU) *Understanding Classi-*
fication
- 7 Ronald Poppe (UT) *Discriminative Vision-Based Re-*
covery and Recognition of Human Motion
- 8 Volker Nannen (VU) *Evolutionary Agent-Based Pol-*
icy Analysis in Dynamic Environments
- 9 Benjamin Kanagwa (RUN) *Design, Discovery and*
Construction of Service-oriented Systems
- 10 Jan Wielemaker (UvA) *Logic programming for*
knowledge-intensive interactive applications
- 11 Alexander Boer (UvA) *Legal Theory, Sources of Law*
& the Semantic Web
- 12 Peter Massuthe (TU/e, Humboldt-Universität zu
Berlin) *Operating Guidelines for Services*
- 13 Steven de Jong (UM) *Fairness in Multi-Agent Sys-*
tems
- 14 Maksym Korotkiy (VU) *From ontology-enabled ser-*
vices to service-enabled ontologies (making onto-
logies work in e-science with ONTO-SOA)
- 15 Rinke Hoekstra (UvA) *Ontology Representation -*
Design Patterns and Ontologies that Make Sense
- 16 Fritz Reul (UvT) *New Architectures in Computer*
Chess
- 17 Laurens van der Maaten (UvT) *Feature Extraction*
from Visual Data
- 18 Fabian Groffen (CWI) *Armada, An Evolving*
Database System
- 19 Valentin Robu (CWI) *Modeling Preferences, Strate-*
gic Reasoning and Collaboration in Agent-Mediated
Electronic Markets
- 20 Bob van der Vecht (UU) *Adjustable Autonomy: Con-*
trolling Influences on Decision Making
- 21 Stijn Vanderlooy (UM) *Ranking and Reliable Clas-*
sification
- 22 Pavel Serdyukov (UT) *Search For Expertise: Going*
beyond direct evidence
- 23 Peter Hofgesang (VU) *Modelling Web Usage in a*
Changing Environment
- 24 Annerieke Heuvelink (VU) *Cognitive Models for*
Training Simulations
- 25 Alex van Ballegooij (CWI) *"RAM: Array Database*
Management through Relational Mapping"

- 3191 26 Fernando Koch (UU) *An Agent-Based Model for the* 3211 35 Wouter Koelewijn (UL) *Privacy en Politiegegevens;*
 3192 *Development of Intelligent Mobile Services* 3212 *Over geautomatiseerde normatieve informatie-*
 3193 27 Christian Glahn (OU) *Contextual Support of social* 3213 *uitwisseling*
 3194 *Engagement and Reflection on the Web* 3214 36 Marco Kalz (OUN) *Placement Support for Learners*
 3195 28 Sander Evers (UT) *Sensor Data Management with* 3215 *in Learning Networks*
 3196 *Probabilistic Models* 3216 37 Hendrik Drachsler (OUN) *Navigation Support for*
 3197 29 Stanislav Pokraev (UT) *Model-Driven Semantic In* 3217 *Learners in Informal Learning Networks*
 3198 *tegration of Service-Oriented Applications* 3218 38 Riina Vuorikari (OU) *Tags and self-organisation: a*
 3199 30 Marcin Zukowski (CWI) *Balancing vectorized query* 3219 *metadata ecology for learning resources in a multi-*
 3200 *execution with bandwidth-optimized storage* 3220 *lingual context*
 3201 31 Sofiya Katrenko (UvA) *A Closer Look at Learning* 3221 39 Christian Stahl (TUE, Humboldt-Universität zu
 3202 *Relations from Text* 3222 Berlin) *Service Substitution – A Behavioral Ap-*
 3203 32 Rik Farenhorst and Remco de Boer (VU) *Architec* 3223 *proach Based on Petri Nets*
 3204 *tural Knowledge Management: Supporting Archi* 3224 40 Stephan Raaijmakers (UvT) *Multinomial Language*
 3205 *tects and Auditors* 3225 *Learning: Investigations into the Geometry of Lan-*
 3206 33 Khiet Truong (UT) *How Does Real Affect Affect Affect* 3226 *guage*
 3207 *Recognition In Speech?* 3227 41 Igor Berezhnyy (UvT) *Digital Analysis of Paintings*
 3208 34 Inge van de Weerd (UU) *Advancing in Software* 3228 42 Toine Bogers (UvT) *Recommender Systems for So-*
 3209 *Product Management: An Incremental Method En* 3229 *cial Bookmarking*
 3210 *gineering Approach*