CHAPTER 3

NONPARAMETRIC BAYESIAN LINE DETECTION

Contents 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.

Published in 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.

Outline The infinite line model describes a collection of lines with a Dirichlet process as prior (Section 3.2). Inference in the infinite line model is performed through Gibbs sampling (Section 3.3). As is known, Gibbs sampling over *parameters* converges slowly, however it can be accelerated through sampling over *clusters* (Section 3.4). The results by the inference method are assessed using clustering performance measures (Section 3.5). The chapter summarizes the findings (Section 3.6) and introduces extensions which will be handled in the next chapters.

3.1 Four Problems with Line Detection

In computer vision and particularly in robotics, traditionally the task of line detection has 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 (Chen et al., 2001; Zhang and Kǒsecká, 2007; Gallo et al., 2011). Second, the noise level is hard-coded 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.

3.2 Infinite Line Model

The application we would like to address in this chapter is that of the detection of multiple lines.

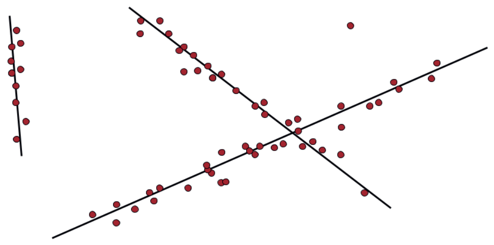


Figure 3.1: A mixture of lines. There are n points in 2D space, each point generated from a line with parameters θ_k. The number of lines k is not known beforehand.

The Dirichlet process has been previously described as prior for a mixture distribution (in Figure 2.2, see Section 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).

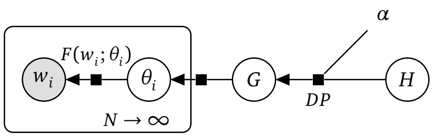


Figure 3.2: For the infinite line model we will use a Dirichlet process mixture. This visualization uses a combination of a so-called factor graph with plate notation. Compare the visualization with Eq. 3.1. The observations w_i are generated by line parameters θ_i (which do not have to be unique). The parameters θ_i are distributed according to G which is sampled from a Dirichlet process with base measure H and dispersion parameter α.

The Dirichlet process mixture nature of the infinite line model is visualized in Figure 3.2 using plate notation (see Appendix A.5). The line parameters θ_i with $i = 1, 2, \dots$ are sampled from a distribution G. This distribution is sampled from the base distribution H with dispersion parameter α. The representation should not be seen as suggesting a form for factorization. The Dirichlet process as a prior for a mixture model we summarize as follows (compare with Figure 3.2):

$$\begin{aligned} G &\sim DP(\alpha, H), \\ \theta_i \mid G &\overset{iid}{\sim} G, \\ w_i \mid \theta_i &\overset{iid}{\sim} F(w_i;\theta_i). \end{aligned} \tag{3.1}$$

Eq. 3.1 represents a mixture model due to the fact that parameters $\{\theta_i, \theta_j\}$ can be identical for $j \neq i$. In that case y_i and y_j are considered to belong to the same cluster characterized by parameter $\theta_i = \theta_j$ (see Section 2.1.3). Here $X \sim S$ means that X has the distribution S. Independence properties, such as observation y_i given parameter θ_i being independent of other observations, are written down explicitly in Eq. 3.1. They might be silently assumed further on.

We will consider models where G is integrated out, the details of which, will follow in this chapter. We also introduce hyperparameters to the base distribution H.

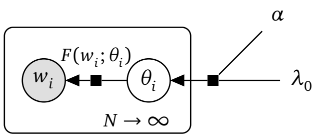


Figure 3.3: The Dirichlet process mixture with the realizations G integrated out and with a hyperparameter λ_0 for the base distribution H.

We will later on see that the base measure H will be the so-called Normal-Inverse-Gamma (NIG) distribution with hyperparameters λ_0. We will also create more detailed figures to emphasize particular aspects of the model.

As mentioned before the parameters θ_i and θ_j can be identical for $i \neq j$. We can equivalently express the Dirichlet process mixture using only k clusters and running the index k over unique clusters.

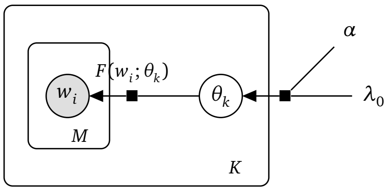


Figure 3.4: The Dirichlet process mixture over clusters with the index, k, ranging over the number of clusters. Here there are M_k observations per cluster k.

In this chapter, both θ_i and θ_k will be used. If the parameter is written as θ_i the index i runs over as many (non-unique) parameters as there are observations. If the parameter is written as θ_k the index k runs over (unique) lines.

In Section 3.2.1 it is described how θ_i is sampled from H and α. In Section 3.2.2 it is described how w_i is sampled from θ_i. In Section 3.2.3 the prior $H(\lambda_0)$ for θ_i is described. In Section 3.2.4 it is described how the hyperparameters λ_0 can be updated given the data w_i to define the posterior predictive for the line parameters, θ_i.

3.2.1 Posterior Predictive for a Line given Other Lines

The Dirichlet process (DP) is described in Section 2.1. The Dirichlet process generates a distribution $G \sim DP(\alpha, H)$ with H the so-called base distribution and α, a scalar, the dispersion parameter.

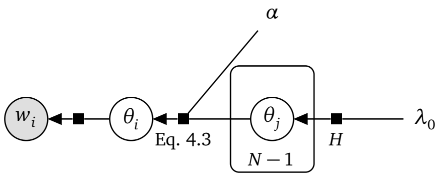


Figure 3.5: The Dirichlet process mixture highlighting the posterior predictive for the parameter θ_i given the other parameters θ_{-i}. Resampling parameters θ_{-i} is governed by Eq. 4.3. The observations w_j with $j \neq i$ with respect to θ_{-i} are not visualized.

The posterior for the Dirichlet process base distribution and dispersion parameter is a Dirichlet process with adjusted parameters:

$$G \mid \theta_1, \dots, \theta_n \sim DP\left(\alpha + n, \frac{\alpha}{\alpha + n} H + \frac{n}{\alpha + n} \frac{\sum_{j=1}^{n} \delta_{\theta_j}}{n}\right). \tag{3.2}$$

The posterior distribution G is a weighted average between the prior base distribution H and the empirical distribution $n^{-1} \sum_{j=1}^{n} \delta_{\theta_j}$ with the weights respectively α and n (normalized). The dispersion parameter α is updated to $\alpha + n$.

The posterior predictive for a new parameter θ_n has the form (see Section 2.1):

$$\theta_n \mid \theta_1, \dots, \theta_{n-1} \sim \frac{1}{\alpha + n - 1}\left(\alpha H + \sum_{j=1}^{n-1} \delta_{\theta_j}\right). \tag{3.3}$$

Due to the exchangeability property, any other parameter update can be written down equivalently (Neal, 2000). The prior distribution of parameters θ_i takes the form of conditional distributions:

$$\theta_i \mid \theta_{-i} \sim \frac{1}{\alpha + n - 1}\left(\alpha H + \sum_{j \neq i} \delta_{\theta_j}\right). \tag{3.4}$$

The notation θ_{-i} describes every other parameter than θ_i: the set of parameters, θ_j, with $j \neq i$. This representation with G marginalized has no independent draws anymore. The draws θ_i depend on previous draws θ_{-i}. This representation is known as the Pólya urn scheme (Blackwell and MacQueen, 1973). It lends itself well to Gibbs sampling (Eq. 2.16) as can be found in the literature (Escobar, 1994; Escobar and West, 1995).

3.2.2 Likelihood of Data given a Line

Each point in our point cloud $w_i = (x_i, y_i)$ we map into a intercept-slope representation using $X_i = [1; \quad x_i]$. A line k we model using ordinary linear regression with slope $\beta_{k,0}$, intercept $\beta_{k,1}$, and standard deviation σ_k. Thus, the line is parametrized by $\theta_k = \{\beta_{k,0}, \beta_{k,1}, \sigma_k\}$ or, equivalently, $\theta_k = \{\beta_k, \sigma_k\}$. The noise is normally distributed with the standard deviation σ_k as in ordinary least squares.

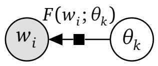


Figure 3.6: This section describes the likelihood of an observation w_i given line parameters θ_k. The likelihood is as in ordinary least squares and can be found in Eq. 3.5.

$$y_i \overset{iid}{\sim} N(X_i \beta_k, \sigma_k^2). \tag{3.5}$$

We can collect all data points $w_i = (x_i, y_i)$ on a line k, $y_i - X_i \beta_k$ for $i = 1, \dots, n$:

$$y - X\beta_k = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} - \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \begin{pmatrix} \beta_{k,0} \\ \beta_{k,1} \end{pmatrix}. \tag{3.6}$$

This allows us to write down the likelihood function as:

$$p(y \mid X, \beta_k, \sigma_k) \propto \sigma_k^{-n} \exp\left(-\frac{1}{2\sigma_k^2}(y - X\beta_k)^T(y - X\beta_k)\right). \tag{3.7}$$

Or equivalently, given the points are drawn i.i.d. from the line parameters:

$$p(y_i \mid X_i, \beta_k, \sigma_k) \propto \sigma_k^{-n} \exp\left(-\frac{1}{2\sigma_k^2}(y_i - X_i\beta_k)^T(y_i - X_i\beta_k)\right). \tag{3.8}$$

The likelihood of a data point $w_i = (x_i, y_i)$ given a line k with parameters θ_k is denoted $F(w_i; \theta_k)$ and is the ordinary linear regression model.

$$F(w_i; \theta_k) = p(y_i \mid X_i, \beta_k, \sigma_k^2). \tag{3.9}$$

We will draw the line parameters θ_k from a prior distribution. This makes this model a **Bayesian linear regression** model as can be found in the literature for single lines (Box and Tiao, 2011).

3.2.3 Conjugate Prior for a Line

We postulate the same prior as done before in the literature (Box and Tiao, 2011) for β_k and σ_k in Eq. 3.9. Those priors are defined on $p(\sigma_k^2)$ rather than $p(\sigma_k)$ which are related through a square root operation. First, we write out the joint probability as a product of the conditional probability and the marginal probability as follows:

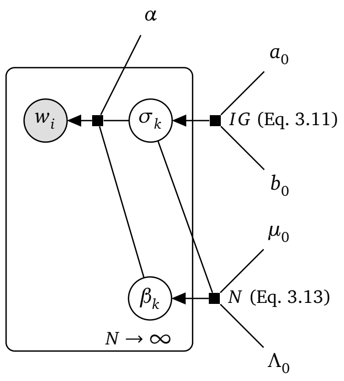


Figure 3.7: The conjugate Normal (N) and Inverse-Gamma (IG) priors for the infinite line model model. The parameter θ_k that parametrizes a line contains a slope β_k and standard deviation σ_k. The main text makes precise how σ_k or more specific, σ_k^2, will be sampled.

$$p(\beta_k, \sigma_k^2) = p(\beta_k \mid \sigma_k^2)p(\sigma_k^2). \tag{3.10}$$

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

$$\sigma_k^2 \sim IG(a_0, b_0). \tag{3.11}$$

In particular, we sample from a distribution $IG(a_0, b_0)$ with hyperparameters $a_0 = \nu_0/2$ and $b_0 = \nu_0 s_0^2/2$, see e.g. Mariotto (1989):

$$p(\sigma_k^2) \propto (\sigma_k^2)^{-(\nu_0/2+1)} \exp(-\frac{1}{2\sigma_k^2}\nu_0 s_0^2). \tag{3.12}$$

The conditional with respect to the line coefficients has a normal distribution as prior:

$$\beta_k \sim N(\mu_0, \sigma_k^2 \Lambda_0^{-1}). \tag{3.13}$$

Written out:

$$p(\beta_k \mid \sigma_k) \propto \sigma_k^{-n} \exp\left(-\frac{1}{2\sigma_k^2}(\beta_k - \mu_0)^T \Lambda_0 (\beta_k - \mu_0)\right). \tag{3.14}$$

The NIG is a distribution that combines a Normal and an Inverse Gamma distribution and can be used as a shorthand for the above exposition. Let us define $\lambda_0 = \{\Lambda_0, \mu_0, a_0, b_0\}$ and recall that $\theta_k = \{\beta_k, \sigma_k\}$, we have now a description for our base distribution H of which we can sample θ_k:

$$H = NIG(\lambda_0). \tag{3.15}$$

Summarized, the standard deviation (or more precisely, the variance, σ_k^2) is sampled from the Inverse Gamma distribution and the line line coefficients, β_k, are sampled from a Normal distribution:

$$\begin{aligned} \sigma_k^2 &\sim IG(a_0, b_0), \\ \beta_k &\sim N(\mu_0, \sigma_k^2 \Lambda_0^{-1}). \end{aligned} \tag{3.16}$$

The hyperparameters are $\lambda = \{\Lambda_0, \mu_0, a_0, b_0\}$. Given that we chose the NIG prior to be conjugate to the likelihood (Section 3.2.2), we can write down the prior predictive distribution:

$$\begin{aligned} p(w_i) &= \int F(w_i; \theta_k) p(\theta_k) d\theta_k, \\ p(y_i) &= \int p(y_i | X_i, \beta_k, \sigma_k^2) p(\beta_k, \sigma_k^2) d\beta_k d\sigma_k^2, \\ &= \int N(X_i \beta_k, \sigma_k^2) NIG(\Lambda_0, \mu_0, a_0, b_0) d\beta_k d\sigma_k^2. \end{aligned} \tag{3.17}$$

This can be written in closed form using a multivariate t-distribution (MVSt):

$$p(y_i) = MVSt_{2a_0}(X_i\mu_0, \frac{b_0}{a_0}(I + X_i\Lambda_0 X_i^T)). \tag{3.18}$$

A detailed derivation can be found in Banerjee (2008).

3.2.4 Posterior Predictive for a Line given Data

In Eq. 3.17 and Eq. 3.18 we find the probability of an observation given the prior on the hyperparameters. Similarly, we want to have an expression for the probability of an observation w_i given previous observations w_j (with $j \neq i$) and the same hyperparameters λ_0:

$$p(w_i|w_j) = \int F(w_i;\theta_k)p(\theta_k|w_j)d\theta_k \tag{3.19}$$

We will see that we arrive at an expression similar to Eq. 3.18.

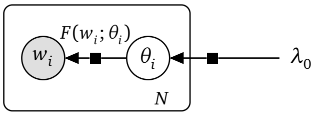


Figure 3.8: The posterior predictive can be calculated by updating the sufficient statistics $\lambda_0 \to \lambda_n$.

The NIG is a conjugate prior with respect to the normal distribution (with unknown mean and variance). Hence, we have a simplified description for updating the hyperparameters, given a set of observations. Recall, the observation $w_i = (x_i, y_i)$ can be equivalently described as (X_i, y_i) or, generalized for more observations, (X, y). The hyperparameters are updated[1] according to (cf. Denison, 2002; Walter and Augustin, 2010):

$$\begin{aligned} \Lambda_n &= \Lambda_0 + X^T X, \\ \mu_n &= \Lambda_n^{-1}(\Lambda_0\mu_0 + X^T y), \\ a_n &= a_0 + n/2, \\ b_n &= b_0 + 1/2(y^T y + \mu_0^T\Lambda_0\mu_0 - \mu_n^T\Lambda_n\mu_n). \end{aligned} \tag{3.20}$$

Let us define $\lambda = \{\Lambda_0, \mu_0, a_0, b_0\}$ and $\lambda^* = \{\Lambda_n, \mu_n, a_n, b_n\}$. Denote a new observation by w_i, then the update for the hyperparameters $\lambda \to \lambda^*$ can be summarized as:

$$\lambda^* = U_{up}(\lambda, w_i). \tag{3.21}$$

[1]In 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.

Removing observations does lead to similarly looking updates (which are at times called "downdates"[2] in the literature, see e.g. Wulsin (2013)) for the hyperparameters:

$$\begin{aligned}\Lambda_n &= \Lambda_0 - X^T X,\\ \mu_n &= \Lambda_n^{-1}(\Lambda_0\mu_0 - X^T y),\\ a_n &= a_0 - n/2,\\ b_n &= b_0 - 1/2(y^T y + \mu_n^T \Lambda_n \mu_n - \mu_0^T \Lambda_0 \mu_0).\end{aligned} \tag{3.22}$$

The downdate for the hyperparameters can then be summarized as:

$$\lambda^* = U_{down}(\lambda, w_i). \tag{3.23}$$

Hence, we can sample from $p(\theta_k \mid \lambda_0, w_i)$ by sampling from a Normal-Inverse Gamma distribution with updated hyperparameters, $NIG(\lambda_n)$. Sampling of $NIG(\lambda_n)$ is as in Eq. 3.16, but with λ_n rather than λ_0:

$$\begin{aligned}\sigma_k^2 &\sim IG(a_n, b_n),\\ \mu_k &\sim N(\mu_n, \sigma^2\Lambda_n^{-1}).\end{aligned} \tag{3.24}$$

The posterior predictive for observation w_i given other observations w_j and the line's hyperparameters becomes:

$$\begin{aligned}p(w_i|w_j) &= \int F(w_i;\theta_k)p(\theta_k)d\theta_k,\\ p(y_i|y_j,X_j) &= \int p(y_i|X_i,\beta_k,\sigma_k^2)p(\beta_k,\sigma_k^2)d\beta_k d\sigma_k^2,\\ &= \int N(X_i\beta_k,\sigma_k^2)NIG(\Lambda_n,\mu_n,a_n,b_n)d\beta_k d\sigma_k^2.\end{aligned} \tag{3.25}$$

This can be written in closed form using a multivariate t-distribution (MVSt):

$$p(y_i|y_j,X_j) = MVSt_{2a_n}(X_i\mu_n, \frac{b_n}{a_n}(I + X_i\Lambda_n X_i^T)). \tag{3.26}$$

A detailed derivation can again be found in Banerjee (2008).

[2]The terminology might have originated from the literature on rank-one updates and downdates on the Cholesky decomposition.

3.3 Inference for the Infinite Line Model

The prior distribution of the parameters can be represented in terms of successive conditional distributions as given in Eq. 4.3 is:

$$\theta_i \mid \theta_{-i} \sim \frac{1}{\alpha+n-1}\left(\alpha H + \sum_{j\neq i} \delta_{\theta_j}\right). \tag{3.27}$$

The conditional prior of θ_i given the parameters θ_j with $j \neq i$ and observation w_i is described by (Escobar, 1988; Escobar and West, 1995; MacEachern and Müller, 1998; Neal, 2000):

$$\theta_i \mid \theta_{-i}, w_i \sim r_i H_i + \sum_{j\neq i} F(w_i;\theta_j)\delta_{\theta_j}. \tag{3.28}$$

We use mainly the notation by Neal (2000), also compare Theorem 5.3 in Ghosal and Van der Vaart (2017). The α-weighted posterior r_i defines the probability that a new cluster will be sampled:

$$r_i = \alpha \int F(w_i;\theta) dH(\theta;\lambda_0). \tag{3.29}$$

In the case of the infinite line model the probability of an observation w_i given the hyperparameter λ_0 is given by Eqs. 3.17 and 3.18. In Eq. 3.29 we multiply the posterior with α which will govern the probability of a new cluster being created.

The distribution H_i is the posterior distribution for the parameter θ given base distribution H and a single observation w_i. We do not need to have calculate the probability for particular parameter values, we only have to sample them from this distribution. Sampling from H_i must be feasible.

$$\theta_i \sim H_i. \tag{3.30}$$

We can sample from H_i by performing a single update of the hyperparameters λ in Eq. 3.21 with observation w_i ($n=1$, $a_n = a_0 + 1/2$, and so on) and then sampling from $NIG(\lambda_{n=1})$, see Eq. 3.24.

The probability of sampling a new parameter is given by[3]:

$$p(\theta_{new}) = \frac{r_i}{r_i + \sum_{j\neq i} F(w_i;\theta_j)}. \tag{3.31}$$

This Gibbs algorithm[4] has been described before in the context of a Dirichlet process mixture, without particular likelihoods or priors in mind (see algorithm 1 in Neal, 2000). As shown in Algorithm 8 after initialization[5], we perform a loop in which for T iterations each

[3]This can be derived from Neal (2000) by $\sum_{i\neq j} bF(y_i,\theta_i) + b\alpha \int F(y_i;\theta_j) dG_0(\theta) = 1$, which gives an expression for the normalization factor b, which can be found as denominator in Eq. 3.31.

[4]The implementation can be found at `https://code.annevanrossum.nl/dpm` in the folder inference (gibbsDPM_algo2), written such that it is compatible with octave.

[5]Initialization details can be found in Appendix B.

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:      $\theta_i$ = GIBBS ALGORITHM 1 INITIALIZATION($w, \lambda_0, \alpha$)    ▷ See Algorithm 15.
3:      for all $t = 1 : T$ do
4:          for all $i = 1 : N$ do
5:              $r_i = \alpha \int F(w_i; \theta) dH$    ▷ Weighted posterior predictive of $w_i$ (Eq. 3.29)
6:              for all $j = 1 : N, j \neq i$ do
7:                  $L_{i,j} = F(w_i; \theta_j)$    ▷ Likelihood of a line given an observation (Eq. 3.9)
8:              end for
9:              $p(\theta_{new}) = \frac{r_i}{r_i + \sum_{j \neq i} L_{i,j}}$    ▷ Probability of sampling a new parameter (Eq. 3.31)
10:             $u \sim U(0,1)$
11:             if $p(\theta_{new}) > u$ then    ▷ Sample with probability $p(\theta_{new})$
12:                 $\lambda_n = U_{up}(w_i, \lambda_0)$    ▷ Update hyperparameters with $w_i$ (Eq. 3.21)
13:                 $\theta_i \sim NIG(\lambda_n)$    ▷ Sample $\theta_i$ from NIG (Eq. 3.24)
14:             else
15:                 $i \sim Mult(N, p(\theta_{old}))$    ▷ Sample $i$ from existing parameters, $\theta_{old}$
16:                 $\theta_i = \theta_{old=i}$    ▷ Pick $\theta_i$ given index $i$
17:             end if
18:         end for
19:     end for
20:     return summary on $\theta_k$ for $k$ lines
21: end procedure
```

θ_i belonging to observation w_i is updated in succession. The loop consists of four steps. First, the posterior predictive for w_i given the hyperparameters $p(w_i \mid \lambda_0)$ is calculated. Second, the likelihood $F(w_i; \theta_j)$ for all θ_j given w_i (with $j \neq i$) is calculated. Third, the fraction with r_i defines the probability for θ_i to be sampled from a new or existing cluster. Fourth, depending on the probability u, (1) a new cluster is sampled, the hyperparameters are updated with information on w_i and thereafter θ is sampled from a Normal-Inverse-Gamma distribution with the updated hyperparameters, or (2) an existing cluster is sampled.

3.4 Accelerating Inference for the Infinite Line Model

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

The probability to sample from an existing cluster depends on the number of items in that cluster (the current data item excluded). This is expressed in equation 3.32.

$$p(c_i = c \text{ } and \text{ } c_i = c_j \text{ } and \text{ } i \neq j \mid c_{-i}, w_i, \alpha, \theta) \propto \frac{n_{c,-i}}{\alpha + n - 1} F(w_i; \theta_i). \tag{3.32}$$

The probability to sample a new cluster only depends on α and the total number of data items. This is formally described in equation 3.33.

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:      $\theta_k, \lambda_c$ = GIBBS ALGORITHM 2 INITIALIZATION($w, \lambda_0, \alpha$)    ▷ See Algorithm 16
3:      for all $t = 1 : T$ do
4:          for all $i = 1 : N$ do
5:              $c = \text{cluster}(w_i)$    ▷ Get cluster $c$ currently assigned to observation $w_i$
6:              $\lambda_c = U_{down}(w_i, \lambda_c)$    ▷ Adjust cluster hyperparameters on removing $w_i$ (Eq. 3.23)
7:              $m_c = m_c - 1$    ▷ Adjust cluster size $m_c$ (and bookkeeping of $K$)
8:              for all $k = 1 : K$ do
9:                  $L_k = m_k \text{ F}(w_i; \theta_k)$    ▷ Likelihood for cluster $k$ given $w_i$ (Eq. 3.34)
10:             end for
11:             $r_i = \alpha \int F(w_i; \theta) dH$    ▷ Weighted posterior predictive of $w_i$ (Eq. 3.29)
12:             $p(\theta_{new}) = \frac{r_i}{r_i + \sum_k L_k}$    ▷ Calculate probability of a new parameter
13:             $u \sim U(0, 1)$
14:             if $p(\theta_{new}) > u$ then    ▷ Sample with probability $p(\theta_{new})$
15:                 $k = K + 1$    ▷ New cluster index (and bookkeeping of $K$, $K = K + 1$)
16:                 $\lambda_k = U_{up}(w_i, \lambda_0)$    ▷ Set hyperparameter $\lambda_k$ with prior pred. given $w_i$
17:                 $\theta_i \sim NIG(\lambda_k)$    ▷ Sample $\theta_i$ from NIG
18:             else
19:                 $k \sim Mult(K, L_k)$    ▷ Sample $k$ from existing clusters (weighed by $m_k$)
20:                 $\lambda_k = U_{up}(w_i, \lambda_k)$    ▷ Update hyperparameter $k$ with post. pred. given $w_i$
21:             end if
22:             $m_k = m_k + 1$    ▷ Increment cluster size $m_k$
23:         end for
24:         for all $k = 1 : K$ do
25:             $\theta_k \sim NIG(\lambda_k)$    ▷ Sample $\theta_k$ from $NIG$ with up to date $\lambda_k$
26:         end for
27:     end for
28:     return summary on $\theta_k$ for $k$ lines
29: end procedure
```

$$p(c_i \in \Omega(c) \text{ and } c_i \neq c_j \text{ and } i \neq j \mid c_{-i}, \alpha) \propto \frac{\alpha}{\alpha + n - 1} \int F(w_i; \theta_i) dH(\theta). \tag{3.33}$$

Here $\Omega(c)$ denotes all admitted values for c_i. The importance of conjugacy is obvious from Eq. 3.33, it will lead to an analytic form of the integral. The inference method using Eqs. 3.32 and 3.33 is described in Section 3.2.

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:

$$L_k = m_k F(w_i; \theta_k). \tag{3.34}$$

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[6] "downdate" the corresponding hyperparameters (as described in Eq. 3.23). 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.

3.5 Results

The infinite line model (see Section 3.2) 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 before, 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 3.5.1 we discuss the clustering performance, in Section 3.5.2 we compare with the Hough transform, in Section 3.5.3 we provide two clustering examples, and in Section 3.5.4 we inspect visually if the model converges through trace plots.

3.5.1 Clustering Performance

The results of the clustering algorithms are measured using conventional metrics. Let us first define the contingency table (see Table 3.1).

Table 3.1: Contingency table. The overlap between clusters X and Y is characterized by the numbers n_{ij} with each number denoting the number of objects common to X and Y: $n_{ij} = |X_i \cap Y_j|$.

$X \backslash Y$	Y_1	Y_2	$\dots$	Y_s	Sums
X_1	n_{11}	n_{12}	$\dots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\dots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\dots$	n_{rs}	a_r
Sums	b_1	b_2	$\dots$	b_s	

(3.35)

[6]We don't strictly have to "downdate". However, this would become quickly computationally intensive. At first sight, we might consider storing a tree of hyperparameters, with a branch for each sequence of observations that we explore. Removal of an observation assigned to a particular cluster would then correspond to backtracking. We go up the the parent node in the tree and get its previously calculated λ value. However, on reaching a leaf through adding observation w_i, we might need to update the hyperparameters by removing observation w_j. This means we need to calculate updates for λ_n for all permutations of θ_i we encounter (permutations, not sequences).

There are basically four types of object pairs possible.

- A pair of points that are placed in the same class in X and in the same class in Y.
- A pair of points that are placed in a different class in X and in a different class in Y.
- A pair of points that are placed in a different class in X and in the same class in Y.
- A pair of points that are placed in the same class in X and in a different class in Y.

Here X can be considered the ground truth, Y the inferred cluster assignment. Note that there is no mention of the indices of those classes (they are exchangeable). The first and second type can be considered "agreements". The third and fourth type can be seen as "disagreements". Let us define, the total number of distinct point pairs:

$$T = \binom{n}{2} = n(n-1)/2. \tag{3.36}$$

The agreements can be counted as (Brennan and Light, 1974):

$$A = \binom{n}{2} + \sum_{i=1}^{r}\sum_{j=1}^{s} n_{ij} - \frac{1}{2}\left(\sum_{i=1}^{r} a_i^2 + \sum_{j=1}^{s} b_j^2\right). \tag{3.37}$$

We will use four performance metrics, the Rand Index, the Mirkin index, the Hubert index, and the Adjusted Rand Index,

Let us first define the Rand Index. It describes the accuracy of cluster assignments (Rand, 1971) as a the ratio of agreements with respect to the total number of possible pairs.

▼ Definition 3.1 — *Rand index*

The Rand index RI is defined by:

$$RI = \frac{A}{T}. \tag{3.38}$$

The Mirkin index (Mirkin and Cherny, 1970) is a metric for disagreement.

▼ Definition 3.2 — *Mirkin index*

The Mirkin index MI is defined by:

$$MI = \frac{T-A}{T}. \tag{3.39}$$

The Hubert index (Hubert, 1977) is a metric takes into account the difference between agreement and disagreement.

▼ Definition 3.3 — *Hubert index*

The Hubert index HI is defined by:

$$HI = \frac{A-(T-A)}{T}. \tag{3.40}$$

However, if we randomly assign points to clusters there is a chance that we assign some of the points correctly. The adjusted Rand index (Hubert and Arabie, 1985) is a "corrected-for-chance" version of the Rand index. The correction calculates the expected index given that X and Y are chosen from a generalized hypergeometric contribution (given number of classes and objects in each):

$$E = \frac{n(n^2+1)}{2(n-1)} - \frac{n+1}{2(n-1)}\left(\sum_{i=1}^{r} a_i^2 + \sum_{j=1}^{s} b_j^2\right) + \frac{2}{2n(n-1)}\left(\sum_{i=1}^{r} a_i^2 \sum_{j=1}^{s} b_j^2\right). \tag{3.41}$$

We can then define the correction to the Rand index by making sure that $A = E$ maps to 0 and that $A = T$ maps to 1.

▼ Definition 3.4 — *Adjusted Rand index*

The Adjusted Rand index AR is defined by:

$$AR = \frac{A-E}{T-E} \tag{3.42}$$

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 3.9 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!

Algorithm 9 leads to stellar performance measures (Figure 3.10). 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

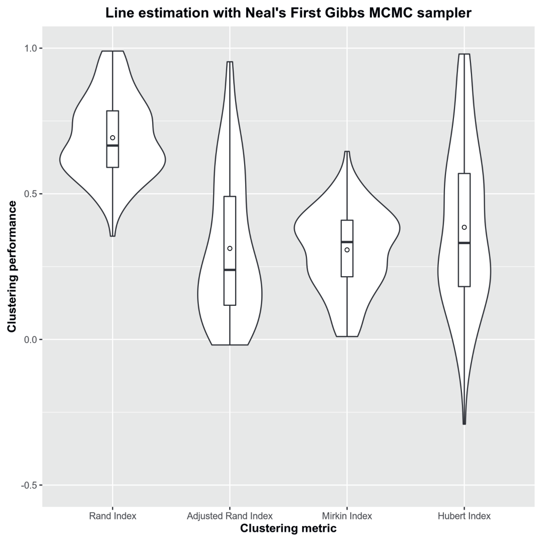


Figure 3.9: The performance of Algorithm 8 with respect to clustering is measured using the Rand Index, the Adjusted Rand Index, the Mirkin metric, and the Hubert metric. A score of 1 means perfect clustering for all metrics, except Mirkin's where 0 denotes perfect clustering.

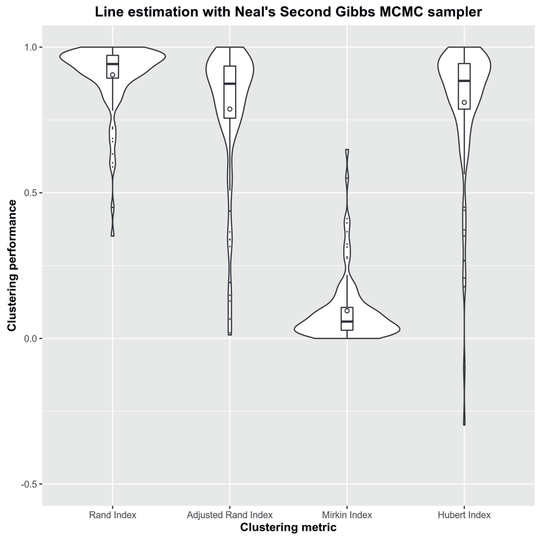


Figure 3.10: The performance of Algorithm 9 with respect to clustering is measured using the Rand Index, the Adjusted Rand Index, the Mirkin metric, and the Hubert metric. A score of 1 means perfect clustering for all metrics, except Mirkin's where 0 denotes perfect clustering.

performance levels. A line seems to form local regions of high probability, making it difficult for points to postulate slightly changed line coordinates.

3.5.2 Hough Transform

A full Bayesian method, in contrast to ad-hoc methods such as the Hough transform, means optimal inference given the model and noise definition. In practice, the model might be misspecified or the actual realization of lines might not have enough data points to benefit from the Bayesian approach. Nevertheless, it is interesting to compare with the Hough transform.

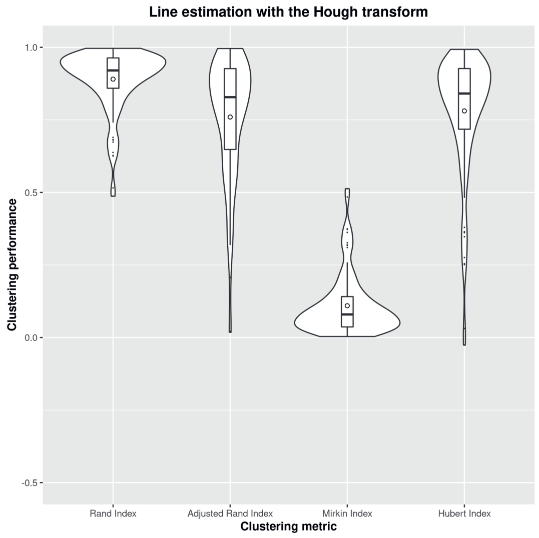


Figure 3.11: The results of the Hough transform on the same dataset. The performance of the Hough transform is slightly worse than line estimation using Neal's second MCMC sampler (Figure 3.10).

The implementation details of the Hough transform are irrelevant to the thesis[7]. We briefly summarize here the key points. The random Hough transform takes two points at random, fits a line between those points, and establishes slope and intercept of this line. A discrete object, the accumulator, exists of, in this case, 100 by 100 cells. Each cell represents an interval of slope and intercept values. For each two points, the accumulator increments a counter per cell. After running over (many or) all point pairs, those cells with large accumulated values are considered to be the detected lines. What constitutes large is determined by a threshold that is application specific.

3.5.3 Two Examples

First, we show two examples of line estimations as we would expect them (see Fig. 3.12).

In contrast to the pictures of Fig. 3.12, we show two examples with typical mistakes. These examples can guide us to understand the inference process better. The first example in seen

[7]Implementation can be found at `https://code.annevanrossum.nl/hough`

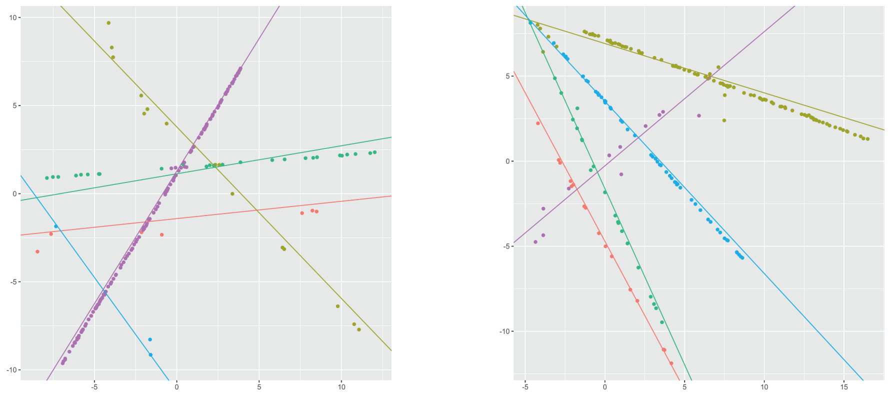
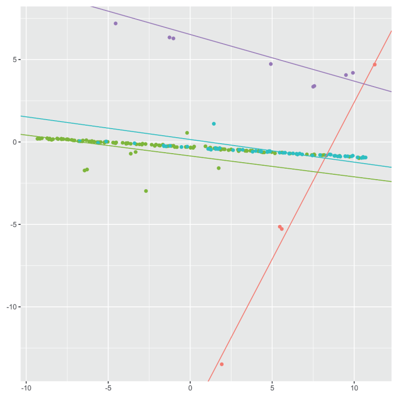


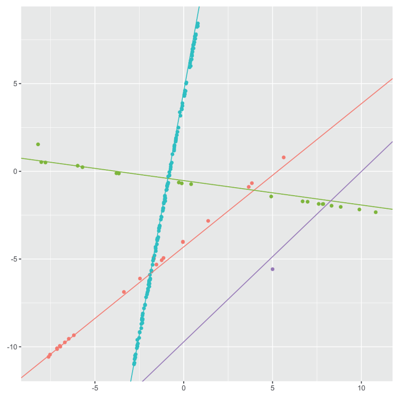
Figure 3.12: Examples of the line estimation process. Apart from slightly different angles and perhaps a few misclassifications the assignments look good.

in Figure 3.13a. It shows the assignment of two lines to a series of points that originated from a single line in the ground truth. Such an assignment can happen after a single Gibbs step in Algorithm 8 or after a long run as final assignment if the system does not convergence to underlying correct assignment.

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.



(a) This shows a mistake where a single line is fitted by two separate lines. One of the lines, the horizontal one in the center has been assigned to multiple clusters.



(b) This shows that outliers are no problem for this type of estimation. An outlier (see the purple point), even if it is a single point, can be assigned its own line.

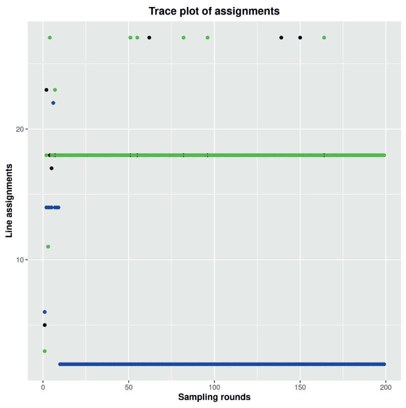
Figure 3.13: Examples of incorrect assignments in line estimation.

The second example (Figure 3.13b) 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.

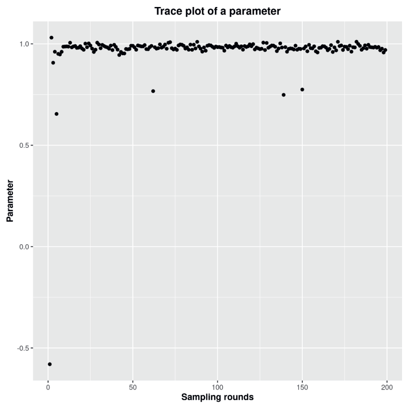
The extension to more points as outliers would, of course, require us to postulate a distribution for these outlier points as well. For instance, a uniform distribution might be used in tandem with the proposed model. However, this would lead to a non-conjugate model and hence it would require different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

3.5.4 Trace Plots

To study the convergence of parameters in an MCMC model, one of the visual aids that is in use, are so-called trace plots. A trace plot does plot values over the course of the simulation run. If we study the trace plot of individual assignments of points over lines, they are not assigned very often to other lines.



(a) This plot traces three points that are assigned to clusters (limited to around 30). Two of the points are assigned to one cluster. The other point to the cluster at the bottom. The plot only shows accepted assignments. The acceptance of a new assignment takes rarely hold.



(b) This plot traces a line parameter β_i belonging to point w_i. It exhibits exploratory behavior around a particular value (in this case 1). So now and then it shows other points (probably from w_i being assigned to a different line, compare with the plot at the right).

Figure 3.14: Two examples of trace plots. Left: a trace plot of the assignment of points to cluster (it changes not so often). Right: a trace plot of a parameter value β_i assigned to w_i.

In Figure 3.14 there are two trace plots. The first plot shows the trace plot of assignments themselves. The MCMC chain steadily assigns the same parameter to the visualized observations. At the start there is a burn-in period visible in which the assignment is more variable. After the burn-in period there are still reassignments, but they are rare. The second plot shows the trace plot of a value of one of the parameters to which on observation has been fitted.

3.6 Chapter Conclusions

The infinite line model proposed in this chapter extends the familiar Bayesian linear regression model to an infinite number of lines using a Dirichlet Process as prior. The model is a full Bayesian 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 3.5 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., 2016a,b). The Bayesian model is solved through two types of algorithms. Algorithm 8 iterates over all observations and suffers from slow mixing. The individual updates make it hard to reassign a 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. We 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.

This chapter contributes to answering our first research question.

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

We use a Bayesian method that we demonstrate on line objects. Its nonparametric nature allows for simultaneous establishing the number of lines as well as their fit.

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 to 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 an 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 cut the lines (of infinite length) to line segments. It means that we need to restrict the points on the line to a uniform distribution of points over a line segment. A symmetric Pareto distribution can be used as prior for the end points of the line segment (see next Chapter). Modeled in this manner, this would subsequently allow for a hierarchical model in which the end points of the line segment 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.