Estimation of the global maximiser of noisy functions

Omar, Fatuma M.A.

Award date:
2019

Technische Universiteit Eindhoven

Bachelor Final Project

Estimation of the global maximiser of noisy functions

By:
F.M.A Omar

Supervisor:
P.J. De Andrade Serra
Second reader:
R.M. Pires da Silva Castro

May 23, 2019

Abstract

Finding the global maximiser of a noisy function is a problem that occurs in many fields. In this thesis a linear regression model given by $\mathbf{Y} = g(\mathbf{x}) + \epsilon$ is considered, where the noise ϵ is normally distributed with $\mathbb{E}[\epsilon] = 0$ and $\mathbb{V}[\epsilon] = \mathbf{I}$. So, the function f is the true function and $\mathbf{Y}$ are the observations with noise. There are many methods to find the global maximiser of a noisy function. The first method discussed is the stochastic approximation algorithm of J.Kiefer and J.Wolfowitz. The second method involves nonparametric and Bayesian statistics and is recursive. The third method will also involve nonparametric and Bayesian statistics, but it is not recursive. For different functions different methods are best to use.

Keywords: recursive estimation, stochastic approximation, nonparametric statistics, Bayesian statistics.

Contents

Chapter 1

Introduction

Finding the global maximiser of noisy functions is a well-known problem. To evaluate these functions, only a finite amount of observations can be obtained. If parameters need to be tuned of a machine that is complex and might also be very costly to evaluate one must turn to methods that are soothed for this problem. For example , Bayesian methods are used to optimize noisy or black-box functions are used to tune parameters in Robotics in order to maximize speed and smoothness [7]. Another example is selecting a subset of a set of weather sensors optimally [4].
In this thesis the regression model given by

$$Y_i = g(x_i) + \sigma\epsilon_i, \tag{1.1}$$

for $i = 1, ..., n$ is considered. In matrix form this is given

$$\mathbf{Y} = g(\mathbf{x}) + \sigma\boldsymbol{\epsilon}. \tag{1.2}$$

Let's assume that $\mathbf{E}(\boldsymbol{\epsilon}) = 0$ and $\mathbf{V}(\boldsymbol{\epsilon}) = \mathbf{I}$. Another assumption is that all the x_i are within the bounds $[0, 1]$. When estimating the function f, it is good to have x_i uniformly distributed, so that the whole function gets estimated well. Usually, it is not the whole function that needs to be estimated, but rather the global maximum. The goal of this thesis is to find the global maximiser of the linear regression function given in equation 1.2. The nature of the true function g is not known. Also, this function isn't necessarily convex. Three methods will be used to achieve this goal. The difference between these functions will be analyzed.
The first method is the implementation of a stochastic approximation algorithm by Kiefer and Wolfowitz [6]. This is explained in section 3.1. The second method is a recursive Bayesian approach, which is described in section 3.2. And the third method is non-recursive Bayesian approach in section 3.3. The preliminaries for this method is given in section 2. These three methods will be tested on three test functions which are different in nature. The three methods will then be compared in performance. These results are seen in section 4. After that, the comparison of the two methods will be discussed in section 5.1. The thesis will be discussed in section 5.2.

Chapter 2

Preliminary

In this section, the preliminaries for the method presented in section 3.1, 3.2 and 3.3 are discussed. This entails non-parametric statistics and Bayesian statistics.

2.1 Nonparametric Statistics

Nonparametric statistics does not have a well established definition. The definition given by Wasserman [11] is that nonparametric statistics uses data in order to infer an unknown quantity while making as few assumptions as possible. In general, this is done by using models that are infinite (or high-) dimensional. This branch of statistics does not assume that data comes from a parametric family of probability distributions.
In this thesis it will be assumed that the data collected can be modelled using a nonparametric regression model. An observation Y_i from a regression model at x_i is given by

$$Y_i = g(x_i) + \epsilon_i \tag{2.1}$$

where Y_i is the response variable, x_i the covariate and g is the regression function. Also, let's assume that $\mathbb{E}(\epsilon_i) = 0$ ans $\mathbb{V}(\epsilon_i) = \sigma^2$ (not dependent of x) for $i = 1, ..., n$. As stated before, the objective is to estimate the function g with as few assumptions as possible. The estimated function is denoted by $\hat{g}_n$.

2.1.1 Smoothing

Before a function can be fitted into the data, it must be smoothed. Let $\hat{g}_n$ be the estimated function of g. Recall that the mean squared error is defined by

$$MSE = bias\left(\hat{g}_n(x)\right)^2 + \mathbb{V}\left(\hat{g}_n(x)\right), \tag{2.2}$$

where $bias\left(\hat{g}_n(x)\right)$ is the bias of $\hat{g}_n(x)$ which is given by $\mathbb{E}\left(\hat{g}_n(x) - g(x)\right)$. And $\mathbb{V}\left(\hat{g}_n(x)\right)$ is the variance of $\hat{g}_n(x)$.
The loss function is given by

$$L(g(x), \hat{g}_n(x)) = (g(x) - \hat{g}_n(x))^2. \tag{2.3}$$

In regression problems the average MSE is used, which is given by

$$R(g, \hat{g}_n) = \frac{1}{n}\sum_{i=1}^{n} L(g(x_i), \hat{g}_n(x_i)). \tag{2.4}$$

This is the average of the loss function. When smoothing data, there is a bias-variance tradeoff. The challenge is to smooth the data such that it is not oversmoothed and not undersmoothed. Oversmoothing occurs when the bias term is large and the variance small. Undersmoothing occurs when the variance term is large and the bias term is small [13].

2.1.2 Nonparametric Regression

Definition 2.1 (Linear Smoother [12])**.** *The estimator $\hat{g}_n$ of g is a linear smoother if, for each x there exists a vector $l(x) = (l_1(x), ..., l_n(x))^T$ such that*

$$\hat{g}_n(x) = \sum_{i=1}^{n} l_i(x) Y_i.$$

Let the fitted values be given by $\mathbf{g} = (\hat{g}_n(x_1), ..., \hat{g}_n(x_n))^T$. And let $\mathbf{Y} = (Y_1, ..., Y_n)^T$. Then, the fitted values can be rewritten to $\mathbf{g} = \mathbf{LY}$, where

$$\mathbf{L} = \begin{pmatrix} \vdots & & \vdots \\ l_1(x_i) & \cdots & l_n(x_i) \\ \vdots & & \vdots \end{pmatrix}.$$

This matrix $\mathbf{L}$ is the smoothing matrix. The effective degrees of freedom, which is the number of parameters, is given by the trace of $\mathbf{L}$.
The smoothers used will depend on a smoothing parameter. Let this smoothing parameter be given by λ. It would be ideal to choose λ such that it minimizes the MSE given in equation 2.4. However, this equation needs the function g, which is unknown. This means that the MSE has to be estimated. This can be done using the leave-one-out cross-validation score.

Definition 2.2 (Leave-one-out cross validation score [12])**.** *Define $\hat{g}_{-i}$ as the estimator that is obtained by omitting the i^{th} pair (x_i, Y_i)). This is given by*

$$\hat{g}_{-i}(x_i) = \sum_{j=1}^{n} Y_j l_{j,(-i)}. \tag{2.5}$$

The leave-one-out cross validation is given by

$$CV = \hat{R}(\lambda) = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{g}_{-i}(x_i)\right)^2. \tag{2.6}$$

The definition of CV can be rewritten to a much simpler version. This is described in Theorem 2.1.

Theorem 2.1 (Cross-validation score [12])**.** *Let $\hat{g}_n$ be a linear smoother. Then the leave-one-out cross validation score can be rewritten to*

$$CV = \hat{R}(\lambda) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{Y_i - \hat{g}_n(x_i)}{1 - L_{ii}}\right)^2. \tag{2.7}$$

The smoothing parameter λ will then be the value that minimizes the CV [12].

2.1.3 Penalized Regression

Let the penalized sums of squares be given by

$$M(\lambda) = \sum_{i=1}^{n} \left(Y_i - \hat{g}_n(x_i)\right)^2 + \lambda J(g), \tag{2.8}$$

where $J(g)$ is the roughness penalty [12]. Here, λ is the smoothing parameter and $\hat{g}_n$ is the function that minimizes $M(\lambda)$. Furthermore, the roughness penalty will be given by $J(g) = \int (g''(x))^2$. In order to analyze $\hat{g}_n$, splines are needed. There are many options, the one used in this thesis will be the B-spline basis which is given in Definition 2.3.

Definition 2.3 (B-spline basis)**.** *Let $\xi_0 = a$ and $\xi_{k+1} = b$. Define knots $\tau_1, .., \tau_M$ such that $\tau_1 \leq, ..., \tau_M \leq \xi_0$, $\tau_{j+M} = \xi_j$ for $j = 1, .., k$, and $\xi_{k+1} \leq \tau_{k+M+1} = ... = \tau_{k+2M}$. The basis functions are defined as*

$$B_{i,1} = \begin{cases} 1 & \text{if } \tau_i \leq x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

for $i = 1, .., k + 2M - 1$. The basis functions can be recursively defined for $m \leq M$

$$B_{i,m} = \frac{x - \tau_i}{\tau_{i+m-1} - \tau_i} B_{i,m-1} + \frac{\tau_{i+m} - x}{\tau_{i+m} - \tau_{i+1}} B_{i+1,m-1} \tag{2.10}$$

for $i = 1, .., k + 2M - m$.[12]

For Theorem 2.2, the following definition is needed.

Definition 2.4 (Cubic spline)**.** *Let the knots be defined by the set of ordered points $\xi_1 < ... < \xi_k$. A cubic spline is a continuous function g such that*

1. *g is a cubic polynomial over $(\xi_1, \xi_2), ..., (\xi_{k-1}, \xi_k)$ and*

2. g has continuous first and second derivatives at the knots. [12]

Theorem 2.2 (B-spline basis functions [12])**.** *The B-spline basis functions are the functions* $\{B_{i,4}, i = 1, ..., k\}$ *are a basis for the set of cubic splines.*

The estimated function $\hat{g}_n(x)$ minimizes $M(\lambda)$ is a natural cubic spline, which is a cubic spline that is linear beyond the boundary knots, with knots at the data points. This estimated function $\hat{g}_n(x)$ is the smoothing spline [12]. Now, the spline estimator can be described by

$$\hat{g}_n(x) = \sum_{j=1}^{n} \beta_j B_j(x), \tag{2.11}$$

where $B_1, ..., B_N$ are a basis for the natural splines. Now, β has to be estimated by minimizing $(Y - B\beta)^T(Y - B\beta) + \lambda\beta^T\Omega\beta$, where $B_{ij} = B_j(X_i)$ and $\Omega_{jk} = \int B_j''(x)B_k''(x)dx$. The estimate of β will be denoted by $\hat{\beta}$. This gives that $\hat{\beta} = (B^TB + \lambda\Omega)^{-1}B^TY$, . Now, the smoothing spline given in equation is a linear smoother where the matrix $\mathbf{L} = B(B^TB + \lambda\Omega)^{-1}B^T$ and the fitted values are given by $\mathbf{g} = \mathbf{LY}$. The parameter λ can be estimated by minimizing the CV, as explained before [12].
To get a feel for the most important concepts, an example will be given in Example 2.1.

Example 2.1 (Penalized Regression)**.** Suppose that the regression model is given by

$$Y_i = sin(x_i) + \epsilon_i,$$

where $\epsilon_i \sim \mathcal{N}(0, 0.25)$.

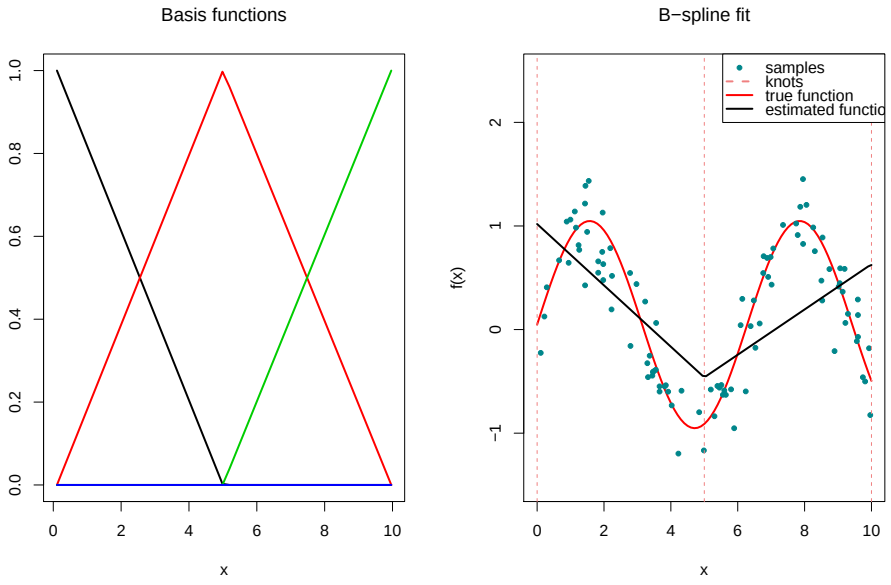

Figure 2.1: The basis function (left) and B-spline fit (right) with knots at $x = 0$, $x = 5$ and $x = 10$.

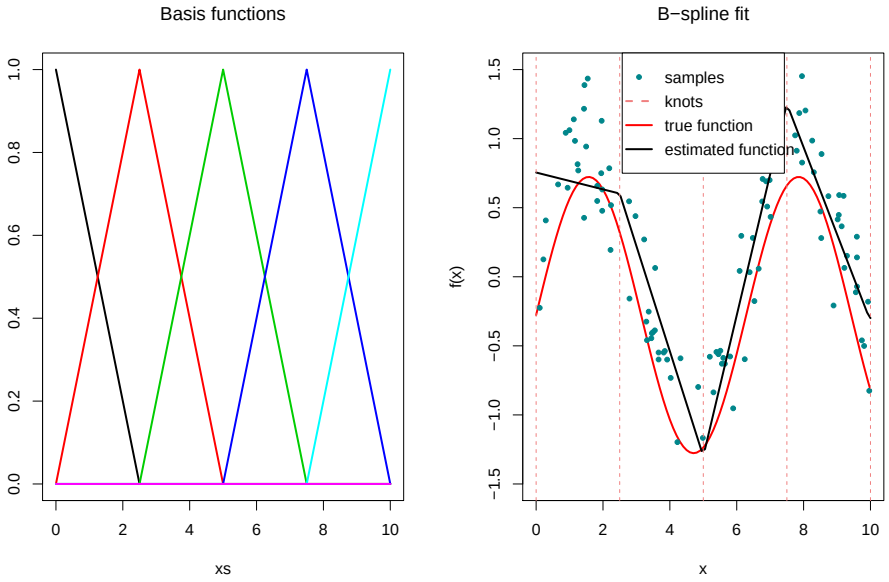

Figure 2.2: The basis function and the b-spline fit.

In Figure 2.1 the basis functions and the B-spline fit are given. The knots are at $x = 0$, $x = 5$ and $x = 10$, which is marked in Figure 2.1 and 2.2 with the orange dashed vertical lines. Since the basis functions are of degree 1, the estimation between knots will be linear functions. Note that the estimate of the true function (given in black) is a linear combination of the basis functions given in the first plot of Figure 2.1. The placement and amount of knots is important for the fit. In Figure 2.2 there are more knots and the degree of the basis functions is now 2. These knots are placed at $x = 0$, $x = 2.5$, $x = 5$, $x = 7.5$ and $x = 10$ and the basis functions are of degree 1. It can be seen that the estimation of the function has improved.

2.2 Bayesian Statistics

The idea of Bayesian statistics is based on Bayes' rule described in theorem 2.3. It states that the probability of an event A given event B is dependent on the individual probabilities and the probability of event B given A.

Theorem 2.3 (Bayes' rule)**.** *Suppose that A and B are events. Bayes' rule states that*

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)},$$

where $P(B) \neq 0$.

Consider a linear regression model. The frequentist method would be to estimate the unknown parameters with methods such as the least squares method or maximum likelihood. These methods give point estimates for the unknown parameters. This is where Bayesian approach to regression differs. Bayesian statistics treats these unknown parameters as random variables. This does not give a point estimate for the unknown parameters, but rather a distribution. These unknown random variables are derived using Bayes' rule.
Let $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_p]^T$ be all the unknown parameters of the linear regression model and let $y = [y_1, ..., y_n]^T$ be the observed data. The objective is to find a distribution of the unknown parameters given the data, this is the posterior probability distribution of $\boldsymbol{\theta}$. Applying Bayes' rule described in theorem 2.3 gives

$$p(\boldsymbol{\theta}|y) = \frac{\mathcal{L}(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta})}{p(y)}, \tag{2.12}$$

where $\mathcal{L}(y|\boldsymbol{\theta})$ is the likelihood function of the observed data, $\pi(\boldsymbol{\theta})$ is the prior probability distribution. The prior can represent the knowledge you have on $\boldsymbol{\theta}$ before observing y. One of the main justifications in using a chosen prior is that all the plausible values for the unknown parameters $\boldsymbol{\theta}$ is contained within that prior distribution. It does not have to be concentrated around the true values for $\boldsymbol{\theta}$ as the likelihood function will outweigh the chosen prior distribution. Lastly, $p(y)$ is the normalizing constant given by

$$p(y) = \int_{\boldsymbol{\theta}} \mathcal{L}(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta})d\boldsymbol{\theta}.$$

Because $p(y)$ is the normalizing constant, one can write

$$p(\boldsymbol{\theta}|y) \propto \mathcal{L}(y|\boldsymbol{\theta})\pi(\boldsymbol{\theta}). \tag{2.13}$$

In order to find the distribution of the posterior, the normalizing constant is needed. However, in many cases it is not possible to analytically solve these integrals. Another problem is that for the posterior, usually the marginal distributions are needed, but the given posterior is multivariate. This leads to the integrals

$$p(\boldsymbol{\theta}_i|y) = \int_{\boldsymbol{\theta}_{-i}} p(\boldsymbol{\theta}|y) d\boldsymbol{\theta}_{-i} \tag{2.14}$$

needing to be solved, where $\boldsymbol{\theta}_{-i}$ denotes $\boldsymbol{\theta}$ without $\boldsymbol{\theta}_i$ [10].

2.2.1 Conjugacy

There are analytically pleasant prior distributions which are conjugate. This means that the prior distribution $\pi(\boldsymbol{\theta})$ and the posterior distribution $p(\boldsymbol{\theta}|y)$ are of the same family of distributions [10]. This notion will be illustrated in example 2.2 and 2.3.

Example 2.2 (Conjugacy)**.** Suppose that the observed data, y, is exponentially distributed with rate λ and the prior distribution is defined by the gamma distribution with parameters α and β. The objective is to find a posterior distribution for λ. The likelihood function of the exponential distribution is given by

$$\mathcal{L}(y|\lambda) = \lambda^n e^{-\lambda \sum_{i=1}^n y_i}.$$

The density of the prior of λ, $\pi(\lambda)$ is given by

$$\pi(\lambda) = \frac{\beta^\alpha \lambda^{\alpha-1} e^{-\beta\lambda}}{\Gamma(\alpha)}.$$

According to the formula given in equation 2.13 it holds that

$$\begin{aligned} p(\lambda|y) &\propto \mathcal{L}(y|\lambda)\pi(\lambda) \\ &\propto \lambda^n e^{-\lambda \sum_{i=1}^n y_i} \frac{\beta^\alpha \lambda^{\alpha-1} e^{-\beta\lambda}}{\Gamma(\alpha)} \end{aligned}$$

In order to find the distribution to which the posterior distribution $p(\lambda|y)$ is proportional to, the equation does not have to depend on α and β. This gives

$$\begin{aligned} p(\lambda|y) &\propto \lambda^n e^{-\lambda \sum_{i=1}^n y_i} \lambda^{\alpha-1} e^{-\beta\lambda} \\ &\propto \lambda^{\alpha+n-1} e^{-\lambda(\beta+n\bar{y})}. \end{aligned}$$

Here, it can be seen that the posterior is gamma distributed with parameters $\hat{\alpha} = \alpha + n$ and $\hat{\beta} = \beta + n\bar{y}$. Thus,

$$\lambda|y \sim \ Gamma(\hat{\alpha}, \hat{\beta}).$$

In Figure 2.3 an example of the prior and posterior densities is given. In this example the observed data is exponentially distributed with rate 10 and the amount of observed data is 100. The prior is gamma distributed with parameters $\alpha = 2$ and $\beta = 1$. Hence, the posterior is also gamma distributed with parameters $\hat{\alpha} = n + \alpha$ and $\hat{\beta} = \beta + n\bar{y}$. The mode of the posterior is 9.94, which is close to the true value of $\lambda = 10$.

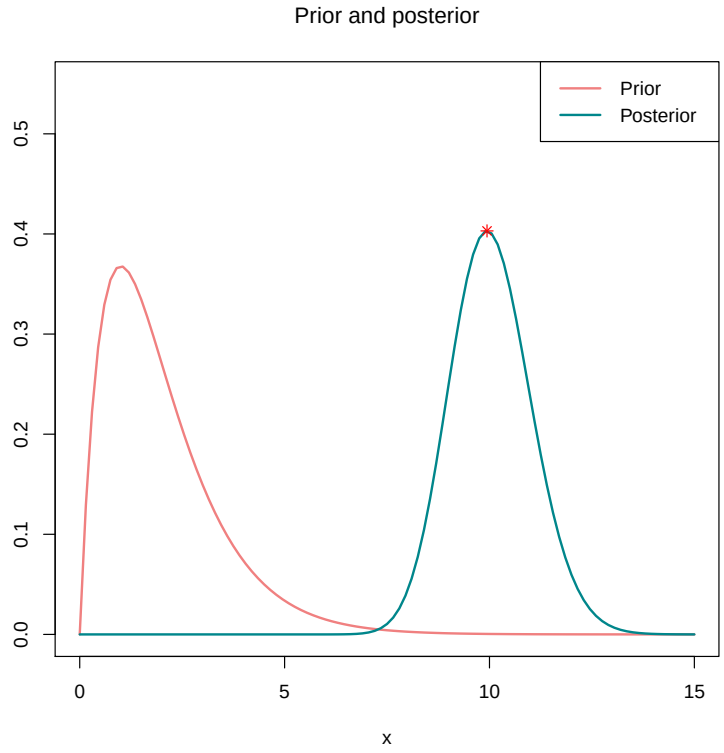

Figure 2.3: An example of the prior and posterior distribution. The red star indicates the mode of the posterior.

Example 2.3 (Normal conjugacy)**.** Let the observed data be $\mathbf{y} = (y_1, y_2, ..., y_n)$ i.i.d. from a normal distribution where the mean μ is unknown and variance σ^2 is known, so

$$y_i \overset{i.i.d.}{\sim} \mathcal{N}(\mu, \sigma^2).$$

Its likelihood is given by

$$\begin{aligned} p(x|\mu, \sigma^2) &= \Pi_{i=1}^n p(y_i|\mu, \sigma^2) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2}\right). \end{aligned}$$

Now, let the prior of μ be the normal distribution with mean μ_0 and variance σ_0^2, then the probability density function of the prior will be given by

$$p(\mu) = \frac{1}{\sqrt{2\pi\sigma_0^2}} exp\left(-\frac{(\mu - \mu_0)^2}{\sigma_0^2}\right).$$

The prior says that the observer believes that somehow the unknown parameter μ is likely to be close to μ_0. Using Bayes' theorem gives the posterior

$$\begin{aligned} p(\mu|y_i) &= \frac{1}{\sqrt{2\pi\sigma^2}} \frac{1}{\sqrt{2\pi\sigma_0^2}} exp\left(-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2}\right) exp\left(-\frac{(\mu - \mu_0)^2}{\sigma_0^2}\right) \\ &\propto exp\left(-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2}\right) exp\left(-\frac{(\mu - \mu_0)^2}{\sigma_0^2}\right) \\ &= exp\left(-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2} - \frac{(\mu - \mu_0)^2}{\sigma_0^2}\right) \\ &= exp\left(-\frac{\sum_{i=1}^n (y_i^2 - 2\mu y_i + \mu^2)}{2\sigma^2} - \frac{\mu^2 - 2\mu\mu_0 + \mu_0^2}{2\sigma_0^2}\right) \end{aligned}$$

Now, let's look at the terms in the exponent. The terms $\sum_{i=1}^n y_i^2$ and μ_0^2 do not depend on μ. This gives

$$p(\mu|y_i) \propto exp\left(-\frac{\sum_{i=1}^n (-2\mu y_i + \mu^2)}{2\sigma^2} - \frac{\mu^2 - 2\mu\mu_0}{2\sigma_0^2}\right). \tag{2.15}$$

Let $n\bar{y} = \sum_{i=1}^{n} y_i$. Rewriting equation 2.15 gives

$$\begin{aligned} p(\mu|y_i) &\propto exp\left(-\frac{-2\mu n\bar{y} + n\mu^2}{2\sigma^2} - \frac{\mu^2 - 2\mu\mu_0}{2\sigma_0^2}\right) \\ &= exp\left(-\frac{1}{2}\left(\left(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\right)\mu^2 - 2\left(\frac{n\bar{y}}{\sigma^2} + \frac{1}{\sigma_0^2}\right)\mu\right)\right). \end{aligned}$$

Let $q = \frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}$ and $r = \frac{n\bar{y}}{\sigma^2} + \frac{1}{\sigma_0^2}$. This leads to

$$\begin{aligned} p(\mu|y_i) &\propto exp\left(-\frac{1}{2}\left(q\mu^2 - 2r\mu\right)\right) \\ &= exp\left(-\frac{q}{2}\left(\mu^2 - 2\frac{r}{q}\mu\right)\right) \\ &= exp\left(-\frac{q}{2}\left(\mu^2 - 2\frac{r}{q}\mu + \frac{r^2}{q^2} - \frac{r^2}{q^2}\right)\right). \end{aligned}$$

Since $\frac{r^2}{q^2}$ does not depend on μ, it holds that

$$\begin{aligned} p(\mu|y_i) &\propto exp\left(-\frac{q}{2}\left(\mu^2 - 2\frac{r}{q}\mu + \frac{r^2}{q^2}\right)\right) \\ &= exp\left(-\frac{a}{2}\left(\mu - \frac{r}{q}\right)^2\right). \end{aligned}$$

Thus, the posterior is normally distributed with mean $\frac{r}{q} = \frac{\frac{n\bar{y}}{\sigma^2} + \frac{1}{\sigma_0^2}}{\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}}$ and variance $\frac{1}{q} = \left(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1}$, i.e.

$$\mu|\{y_i, i = 1, \ldots, n\} \sim \mathcal{N}\left(\frac{\frac{n\bar{y}}{\sigma^2} + \frac{1}{\sigma_0^2}}{\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}}, \left(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1}\right).$$

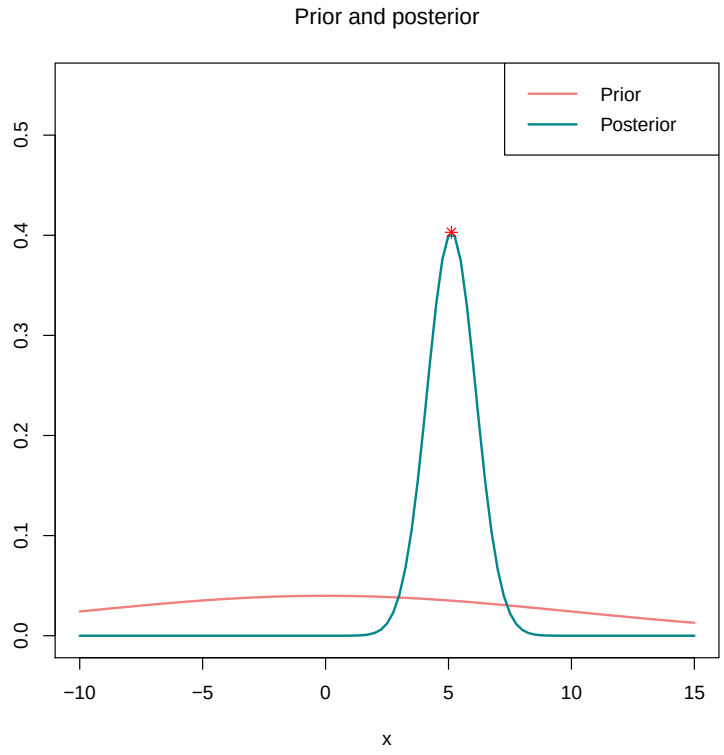

Figure 2.4: An example of the prior and posterior distribution. The red star indicates the mode of the posterior.

For $n = 100$ observations the mode of the posterior is at 5.1, this is close to the true mean of 5.

2.2.2 Bayesian Regression

Let $\mathbf{Y}$ denote the response variable. And let $\mathbf{X}$ denote the design matrix, which is given by the B-splines. Say

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \text{ and } \mathbf{X} = \begin{pmatrix} B_1(x_1) & \cdots & B_p(x_1) \\ \vdots & & \vdots \\ B_1(x_n) & \cdots & B_p(x_n) \end{pmatrix} = \left[B_j(x_i)\right].$$

The linear regression model is given by $\mathbf{Y} = \mathbf{X}\boldsymbol{\theta} + \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. The likelihood of the observations is given by

$$\mathbf{Y}|(x_1, x_2, ..., x_n), \boldsymbol{\theta} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\theta}, I)$$

Furthermore, the prior on $\boldsymbol{\theta}$ is given by

$$\boldsymbol{\theta} \sim \mathcal{N}(0, I),$$

with density The posterior will then be

$$p(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{X}) \propto exp\left(-\frac{1}{2}(\boldsymbol{\theta}^T\boldsymbol{\theta} + (\mathbf{Y} - \mathbf{X}\boldsymbol{\theta})^T(\mathbf{Y} - \mathbf{X}\boldsymbol{\theta}))\right).$$

This gives

$$\begin{aligned} p(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{X}) &\propto exp\left(-\frac{1}{2}\left(\boldsymbol{\theta}^T\boldsymbol{\theta} + (\mathbf{Y} - \mathbf{X}\boldsymbol{\theta})^T(\mathbf{Y} - \mathbf{X}\boldsymbol{\theta})\right)\right) \\ &= exp\left(-\frac{1}{2}\left(\boldsymbol{\theta}^T\boldsymbol{\theta} + \mathbf{y}^T\mathbf{y} - 2\boldsymbol{\theta}^T\mathbf{X}^T\mathbf{y} + (\mathbf{X}\boldsymbol{\theta})^T(\mathbf{X}\boldsymbol{\theta})\right)\right) \\ &= exp\left(-\frac{1}{2}\left(\boldsymbol{\theta}^T\boldsymbol{\theta} + \mathbf{Y}^T\mathbf{Y} - 2\boldsymbol{\theta}^T\mathbf{X}^T\mathbf{Y} + \boldsymbol{\theta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\theta}\right)\right) \\ &= exp\left(-\frac{1}{2}\left(\boldsymbol{\theta}^T(I + \mathbf{X}^T\mathbf{X})\boldsymbol{\theta} + \mathbf{Y}^T\mathbf{Y} - 2\boldsymbol{\theta}^T\mathbf{X}^T\mathbf{Y}\right)\right) \end{aligned}$$

The form in which this needs to be rewritten in order to recognize a normal distribution is

$$exp\left(-\frac{1}{2}((\boldsymbol{\theta} - \mu^T)\Sigma^{-1}(\boldsymbol{\theta} - \mu)\right) = exp\left(-\frac{1}{2}(\boldsymbol{\theta}^T\Sigma^{-1}\boldsymbol{\theta} - 2\mu\Sigma^{-1}\boldsymbol{\theta} + \mu^T\Sigma^{-1}\mu)\right).$$

Now, we recognize that this is a normal distribution with mean μ and Σ, where

$$\Sigma^{-1} = (\mathbf{I} + \mathbf{X}^T\mathbf{X}),$$

this means that $\Sigma = (\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}$. For the mean it then holds that

$$\begin{aligned} \mu^T\Sigma^{-1}\boldsymbol{\theta} &= \mathbf{Y}^T\mathbf{X}\boldsymbol{\theta} \\ \mu^T\Sigma^{-1} &= \mathbf{Y}^T\mathbf{X} \\ \mu^T &= \mathbf{Y}^T\mathbf{X}(\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1} \\ \mu &= (\mathbf{Y}^T\mathbf{X}(\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1})^T \\ \mu &= (\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \end{aligned}$$

Now it is clear that the posterior is normally distributed with the following parameters,

$$p(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{X}) \sim \mathcal{N}((I + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}, (I + \mathbf{X}^T\mathbf{X})^{-1}). \tag{2.16}$$

Instead of normal regression, penalized regression will be applied. It is the same idea as the ridge regression explained in the preliminaries. In the Bayesian case the prior distribution on $\boldsymbol{\theta}$ is given by

$$\boldsymbol{\theta} \sim \mathcal{N}(0, \lambda\mathbf{I}).$$

With the same likelihood function, by the previous calculations the posterior distribution is given by

$$\boldsymbol{\theta} \sim \mathcal{N}\left((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}, (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\right).$$

This will be the posterior distribution on $\boldsymbol{\theta}$ that will be used for finding the global maximiser.

Chapter 3

Estimating the global maximiser

Now that the foundation has been set, the three methods for finding the global maximiser can be given. Section 3.1 will discuss classic stochastic approximation algorithms, while the method described in section 3.2 is recursive method that is based on nonparametric and Bayesian statistics. The third method is also based on nonparametric and Bayesian statistics, but it is not recursive. This is described in section 3.3.

3.1 Method 1: Stochastic Approximation

The first approach might be to try Newton's method given in Theorem 3.1.

Theorem 3.1 (Newton's method)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real twice partially differentiable function. Consider the recursive equation*

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

As n goes to infinity, x_n will converge to the root of $f(x)$ [9].

Theorem 3.1 can be used to find the global maximiser of a given function by finding the root of the derivative. The recursion will then be given by

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}.$$

This method will not work since the objective function is observed with noise and it is not given that the function is twice differentiable which makes it difficult to find the first and second derivative at a certain point. Suppose that it is given that the function is twice differentiable, then the intuitive approach is to estimate the derivative. Since the function is observed with noise, this estimation of the derivative will be very inaccurate if it is estimated with standard numerical methods. Furthermore, the second derivative also needs to be estimated from the first derivative.
In the early 1950s Robbins and Monro [8] introduced a stochastic approximation algorithm for finding the root of a function observed with noise. This algorithm will be explained in section 3.1.1. The problem was extended by J.Kiefer and J.Wolfowitz [6] to finding the root of the first derivative i.e. finding the optimizer of a regression function $f(x)$. This algorithm will be explained in section 3.1.2.

3.1.1 Robbins-Monro Algorithm

Suppose that $f(x)$ is the expected value of the response variable $Y(x)$ for some x, which is the explanatory variable. So, $f(x) = \mathbb{E}[Y(x)]$, and $Y(x) = f(x) + \epsilon$ with ϵ being random noise. Let $f(x)$ be an unknown monotone function of x and α be a given constant. Also, let $H(Y|x)$ be the distribution function of Y. The objective is to find a unique point $x = \theta$ for which it holds that $f(\theta) = \alpha$. Define the Markov chain $\{x_n\}$ such that

$$x_{n+1} - x_n = a_n(\alpha - Y_n), \tag{3.1}$$

where Y_n is a random variable such that $P(Y_n \leq y|x_n) = H(Y|x_n)$ and $\{a_n\}$ is a fixed positive sequence where

$$0 < \sum_{n=1}^{\infty} a_n^2 < \infty.$$

The objective is to get $\lim_{n\to\infty} b_n = 0$, where $b_n = \mathbb{E}\,(x_n - \theta)$, since this implies convergence in probability by Markov's inequality. In order to obtain this limit the following theorems are introduced.

Theorem 3.2 (Robbins-Monro)**.** *Consider $f(x)$ with the following constrictions:*

1. *$f(x)$ is non decreasing,*
2. *$f(\theta) = \alpha$, and*
3. *$M'(\theta) > 0$.*

And it must hold that

$$\mathbb{P}\left(|Y(x)| \leq C\right) = \int_{-C}^{C} dH(y|x) = 1.$$

Furthermore, $\{a_n\}$ is of type $\frac{1}{n}$, which means that $\frac{c_1}{n} \leq a_n \leq \frac{c_2}{n}$ for some positive constants c_1 and c_2. Then, $\lim_{n\to\infty} b_n = 0$.

In short, in order to obtain the $x = \theta$ for which $f(x) = \alpha$, one must first choose x_1 and find the consecutive points with the Markov chain given in equation 3.1. As n goes to infinity, x_n will converge to θ if all the criteria in Theorem 3.2 are met.

Example 3.1. Suppose that the true function is given by

$$f(x) = x + 1.$$

However, data that is observed is given by

$$Y(x) = f(x) + \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0, 1).$$

The objective will be to find the root of the function $f(x)$ observed with noise. This can be done with the Robbins-Monro algorithm as this function is monotone increasing with the first derivative being positive, since $f'(x) = 1$ for all x. And the true root is given by $x = -1$, hence it exists. And the sequence a_n chosen is $a_n = \frac{1}{n}$, this is of type $\frac{1}{n}$. According to Theorem 3.2 the root can be found using the Markov chain given in equation 3.1.

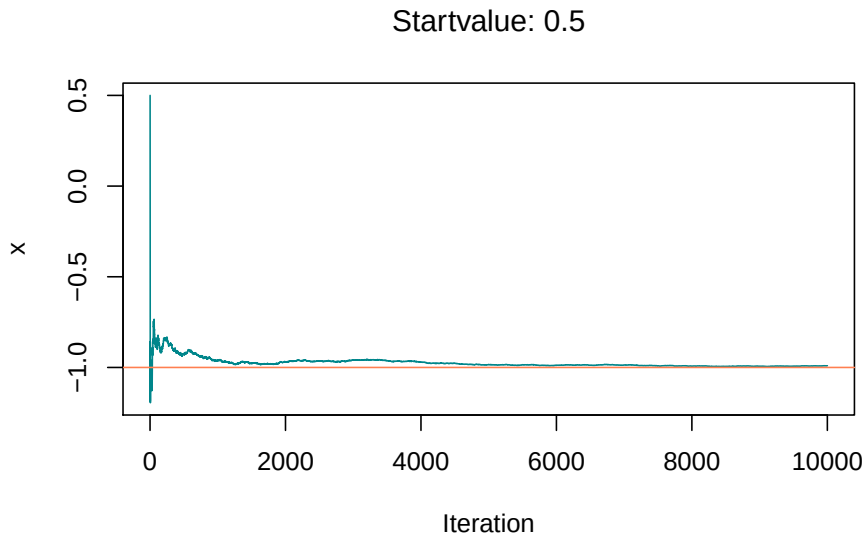

Figure 3.1: The estimated root for 10000 iterations with the true root being the horizontal line at $x = -1.0$.

In Figure 3.1 the true root is given by the horizontal line. Furthermore, for 10000 iterations, it shows that it does converge to its true root. Thus, for enough iterations, it seems accurate.

In order to be able to use the Robbins-Monro algorithm, the first derivative is needed. However, this is not available, which leads to the Kiefer-Wolfowitz algorithm.

3.1.2 Kiefer-Wolfowitz Algorithm

The problem will be extended to finding the root of the first derivative, i.e. finding the maximiser or minimiser of a regression function $f(x)$. So the objective will be to find θ such that

$$\theta = \arg\max_x f(x).$$

The solution to this problem has been given by J. Kiefer and J. Wolfowitz [6]. They considered the recursive formula given in equation 3.2.

$$x_{n+1} = x_n + a_n \frac{Y(x_n + c_n) - Y(x_n - c_n)}{2c_n} \tag{3.2}$$

The conditions for which x_n converges to the maximiser $x = \theta$ is described in Theorem 3.3.

Theorem 3.3 (Kiefer-Wolfowitz)**.** *Suppose that $f(x)$ is strictly increasing for $x < \theta$ and decreasing for $x > \theta$. And suppose that*

$$\int_{-\infty}^{\infty} (y - f(x))^2 dH(y|x) \leq S < \infty,$$

where S is a constant. Moreover, let $\{a_n\}$ and $\{c_n\}$ be sequences such that the following conditions hold:

1. $\lim_{n\to\infty} c_n = 0$,
2. $\sum_{n=1}^{\infty} a_n = \infty$,
3. $\sum_{n=1}^{\infty} a_n c_n < \infty$ *and*
4. $\sum_{n=1}^{\infty} a_n^2 c_n^{-2} < \infty$.

Then the recursive equation given in 3.2 will stochastically converge to its maximiser $x = \theta$.

So, one needs to pick a_n and c_n carefully. An example given by Kiefer and Wolfowitz is $a_n = n^{-1}$ and $c_n = n^{-1/3}$. Pick x_1, which is an arbitrary number. As n goes to infinity, x_n will converge to its maximiser $x = \theta$. This algorithm will be used to find the maximiser for the function described in the introduction. Note that if the objective is to find the global maximiser, then there cannot be another local maximiser, as the algorithm might converge to the local maximiser. So, the idea is to take an arbitrary amount of starting points uniformly distributed between the values that the maximiser might lie in. Get the estimated maximisers for these different starting points and estimate of each estimated maximiser its maximum. Then take the maximiser which gives the greatest maximum. The way to find the greatest maximum is by estimating the value at the potential maximisers. Note that this will also cost some samples. Another thing to note is that each iteration needs two samples. This means that this algorithm has to be run for $\frac{n}{2}$ iterations, since that gives an equal amount of samples used as the other methods. If p starting points are used, then the number of iterations that are allowed is equal to $\frac{n}{2p}$. The estimation of the maximiser will be given by the value of the last iteration. And the estimation of the global maximiser will be $\frac{Y(x_{n/(2p)}+c_{x_{n/(2p)}}))+Y((x_{n/(2p)}-c_{x_{n/(2p)}})}{2}$, because this does not cost more samples.

3.2 Method 2: Recursive Bayesian

The penalized regression function in section 2.2.2 will be considered again. Recall that the distribution of the posterior of θ is given by

$$p(\theta|\mathbf{X}, \mathbf{Y}) \sim \mathcal{N}\left((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}, (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\right). \tag{3.3}$$

Again, λ is the smoothing parameter that is determined by minimizing the CV score. The steps to recursively estimate the global maximiser are as following.

Step 1 Initialize the algorithm with two starting points x_0 and x_1. Both x_0 and x_1 are drawn from the uniform distribution in the interval $[0, 1]$. Thus, $x_0, x_1 \sim Unif[0, 1]$.

Step 2 Sample a function by first sampling from the posterior distribution of $\boldsymbol{\theta}$, this sample will be denoted by $\hat{\boldsymbol{\theta}}$. The estimate will then be given by $\hat{\mathbf{Y}} = \mathbf{X}\hat{\boldsymbol{\theta}}$.

Step 3 Get the maximum of the of the estimated function $\hat{\mathbf{Y}}$. The location of the maximum will be the next sample, z_2.

Step 4 With x_0, x_1 and x_2, steps 2 and 3 will be repeated, this results in x_3. Thus, the recursion will be for the $p-1$ obtained samples $x_0, x_1, ..., x_{p-1}$ steps 2 and 3 will be repeated, which gives x_p, for $p = 2, ..., n$.

Step 5 The samples x_n is the estimated maximiser.

A few things to note. First the matrix $\mathbf{X}$ will be defined by B-splines which is discussed in section 2. The reason for this basis function is its simplicity in implementation. The B-splines will be of degree 2. The position of the knots will not be a fixed number it will instead depend on the location of all the design points. Since the samples will converge around the true maximiser, it will be more desirable to have more knots around this point to get a better estimate with each iteration. Thus, the knots are defined by the quantiles of the sample points. Let n be the size of the the sample points, then the quantiles will be probabilities between 0 and 1 of length $\max(5, n^{1/3})$. It is conventional to use $n^{1/3}$. And it takes the maximum of 5 and $n^{1/3}$, since in the beginning there won't be many samples, which would otherwise give too few knots. This means that in the beginning there will be 5 knots, which will be advantageous for the exploration of the function. And later on, when there are more knots around the maximiser, there will be more exploitation. To get a feel for this method, an example has been generated in Example 3.2.

Example 3.2 (Recursive Bayesian Method)**.** Suppose that the linear regression function is given by $Y_i = \sin(\pi x_i) + \epsilon_i$ where $\epsilon_i \sim \sigma\mathcal{N}(0,1)$, where $\sigma = 0.2$.
It can be seen in Figure 3.2 that the first two samples are set (step 1). The red line is the sampled function (step 2). The maximum of this function is indicated by the red asterisk (step 3). The maximiser of this sampled function will be the next location to be sampled, which is seen in Figure 3.3. Again, the red function indicates the sampled function. The maximum of this function is indicated by the red asterisk, which will be the next location to be sampled, which is seen in Figure 3.4. Repeating this 50 times gives Figure 3.5. It can be seen that the density of the maximiser and maximum gets more concentrated around their true values (step 4). The sequence x_n of is given by all the locations of the sampled points, the black asterisks (step 5). The density of the maximiser and maximum are obtained from the maximiser and maxima of the posterior functions (the grey functions).

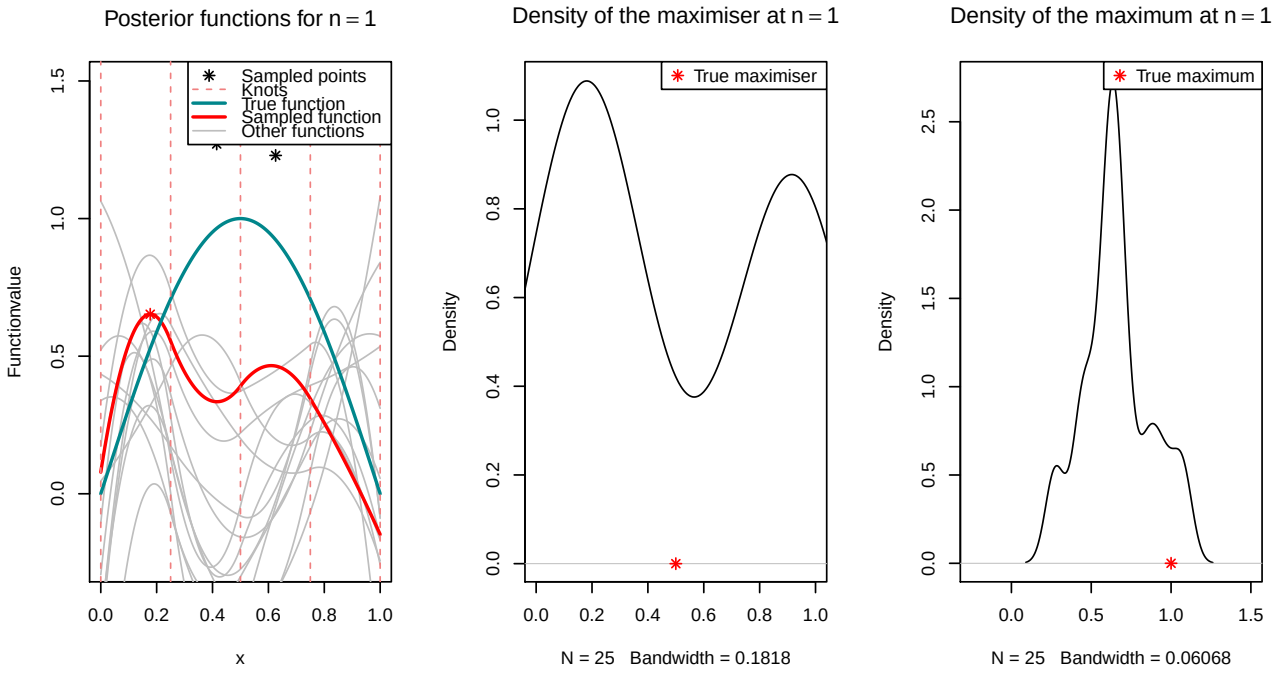

Figure 3.2: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 1$.

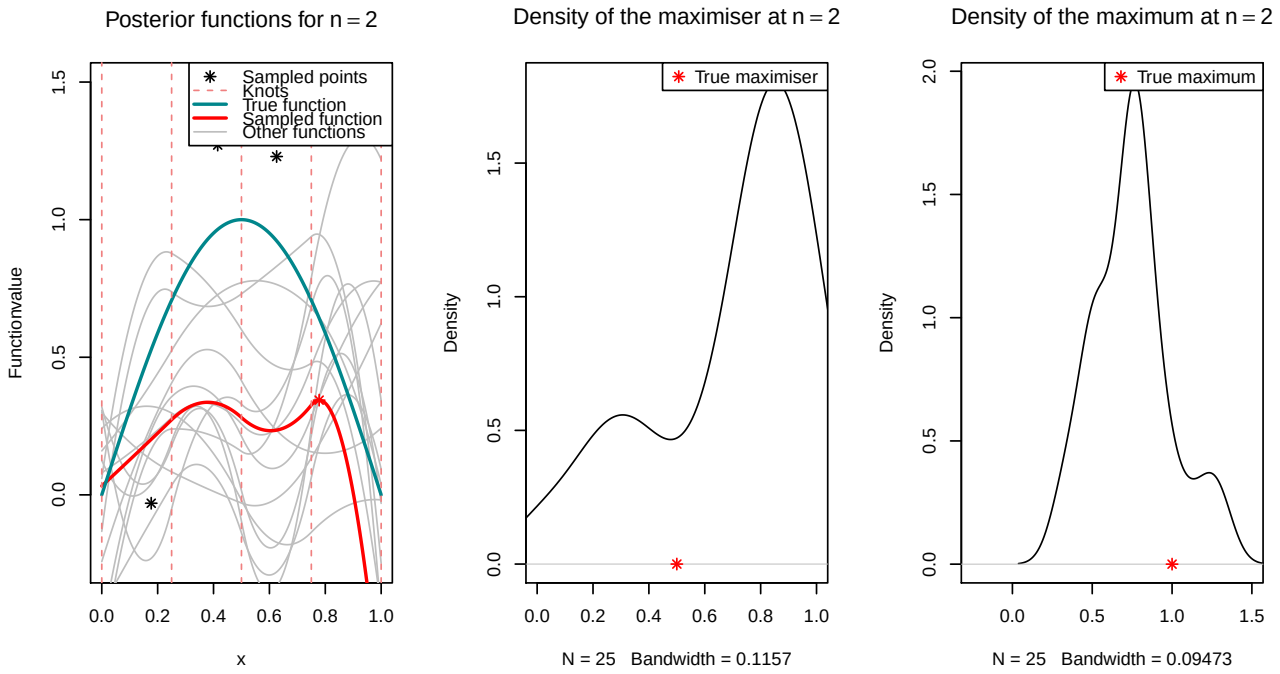

Figure 3.3: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 2$.

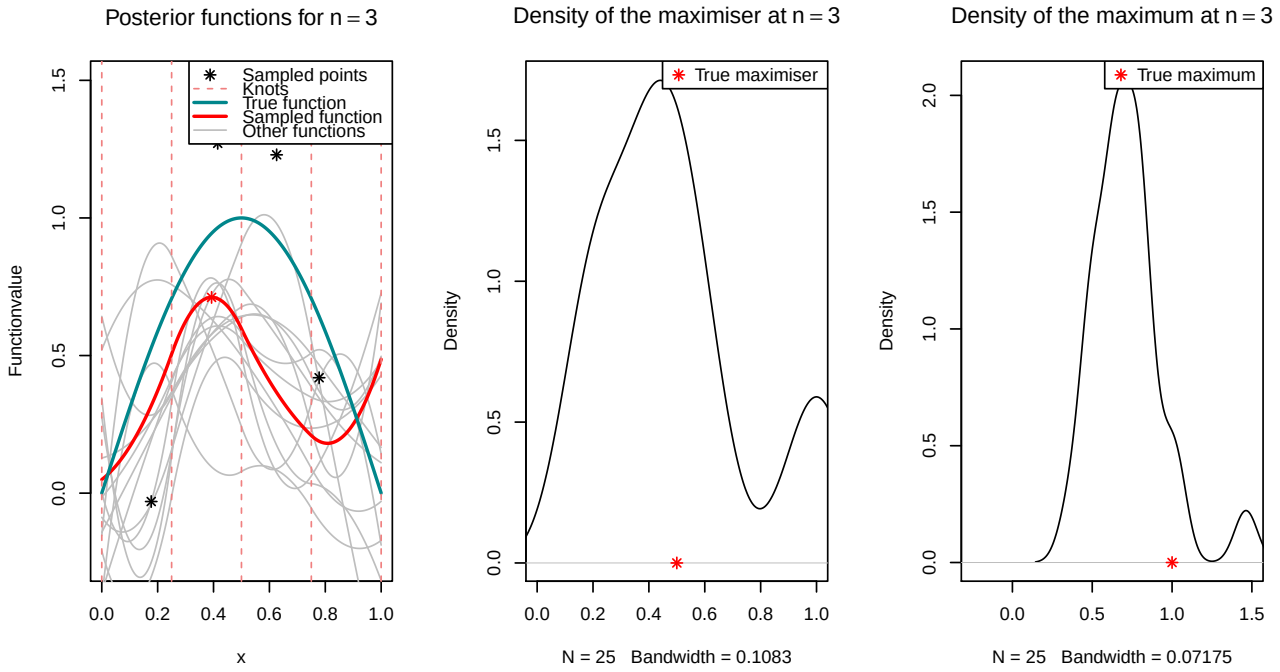

Figure 3.4: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 3$.

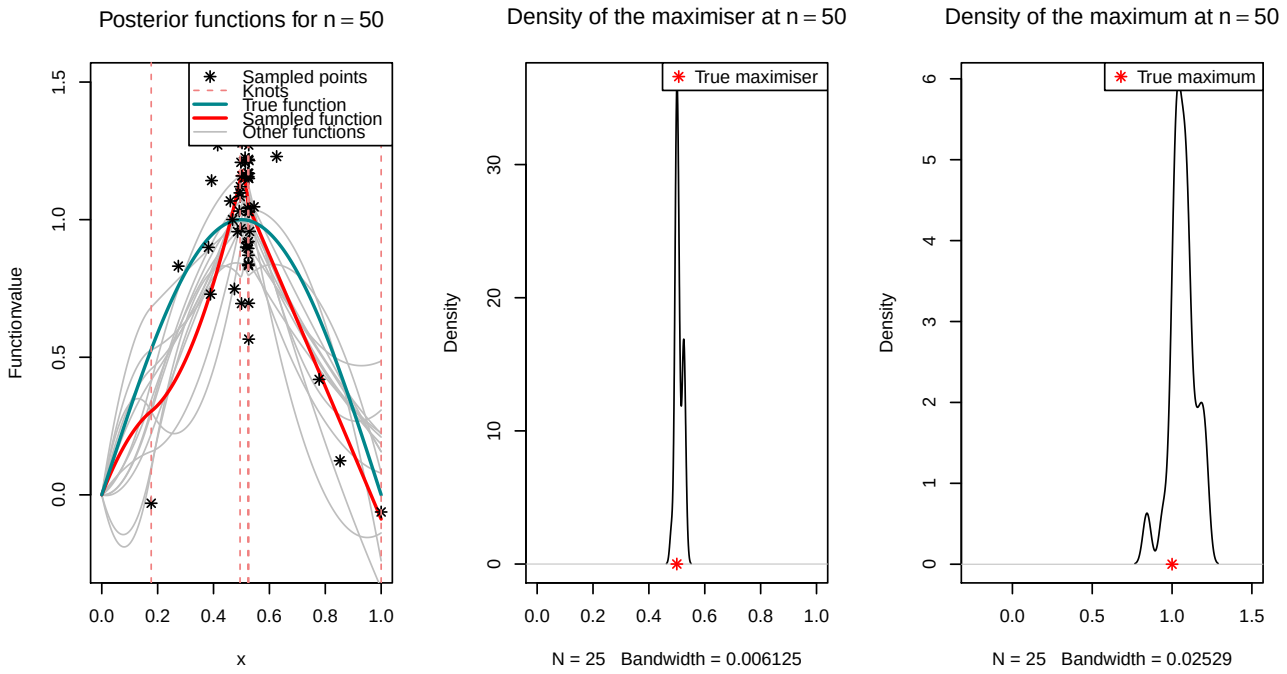

Figure 3.5: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 50$.

3.3 Method 3: Non-Recursive Bayesian

This method is in computation the same as to the one described in section 2.2.2. For this method, n samples will be taken from a uniform distribution between 0 and 1. For the estimation of the function, the mean posterior $\hat{\boldsymbol{\theta}} = (\lambda \mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$ will be used. This gives $\hat{\mathbf{Y}} = \mathbf{X}\hat{\boldsymbol{\theta}}$. The placement of the knots and the amount of knots are determined the same way they are determined with the recursive algorithm. And the matrix $\mathbf{X}$ will be again be defined by B-splines. And λ is also determined by minimizing the CV score.

Chapter 4

Results

Now that the three methods of obtaining the global maximiser are given, the three methods will be compared. The main performance that will be measured is whether the algorithms approximate both the maximiser and maximum well. The results will be given for multiple functions. The true functions that will be used are

$$\begin{aligned} g_1(x) &= C_1 \left(x \sin(10x) + x\right) \\ g_2(x) &= C_2 \left(\sin\left(\frac{13\pi x}{2}\right) + \cos\left(\frac{10\pi x}{3}\right)\right) \\ g_3(x) &= C_3 \left(\left(1 - \frac{1}{0.707107}\right)\sqrt{\left|x - \frac{1}{2}\right|}\right). \end{aligned}$$

In order to make a fair comparison, the signal to noise ratio must be equal in all three functions. The signal to noise ratio of these three functions are given by

$$SNR = \frac{\int_0^1 g_i(t)^2 dt}{\sigma^2} \text{ for } i = 1, 2, 3. \tag{4.1}$$

The constants C_1, C_2 and C_3 will be chosen so that the squared integrals of the functions g_1, g_2 and g_3 will be equal to 1. This gives that

$$C_1 = \frac{1}{\int_0^1 \left(x \sin(10x) + x\right)^2 dx} = \frac{3968 + 320\sin(10) - 199\sin(20) - 1568\cos(10) - 20\cos(20)}{8000} \approx 0.29, \tag{4.2}$$

$$C_2 = \frac{1}{\int_0^1 \left(\left(1 - \frac{1}{0.707107}\right)\sqrt{\left|x - \frac{1}{2}\right|}\right)^2 dx} = \frac{37440 + 12963\sqrt{3} + 89680\pi}{89680\pi} \approx 1.21, \tag{4.3}$$

$$C_3 = \frac{1}{\int_0^1 \left(\left(1 - \frac{1}{0.707107}\right)\sqrt{\left|x - \frac{1}{2}\right|}\right)^2 dx} = 0.166667. \tag{4.4}$$

The functions that will be analyzed are

$$\begin{aligned} g_1(x) &= \frac{x\sin(10x) + x}{\sqrt{\frac{3968 + 320\sin(10) - 199\sin(20) - 1568\cos(10) - 20\cos(20)}{8000}}} \\ g_2(x) &= \frac{\sin\left(\frac{13\pi x}{2}\right) + \cos\left(\frac{10\pi x}{3}\right)}{\sqrt{\frac{37440 + 12963\sqrt{3} + 89680\pi}{89680\pi}}} \\ g_3(x) &= \frac{\left(1 - \frac{1}{0.707107}\right)\sqrt{\left|x - \frac{1}{2}\right|}}{\sqrt{0.166667}}. \end{aligned}$$

This means that the squared integrals of g_i for $i = 1, 2, 3$ are equal to 1, which makes sure that the signal to noise ratio is equal for all functions, since σ is equal for all functions. This σ will be chosen to be $\sigma = 0.1$, which gives a signal to noise ratio of $\frac{1}{0.01}$. These scaled functions are depicted in Figure 4.1. The red stars are the locations of the global maxima. The global maximisers of g_1, g_2 and g_3 are 0.809966, 0.0615783 and 0.5, respectively. And the global maxima of g_1, g_2 and g_3 are 2.034702, 1.589668 and 2.449487.

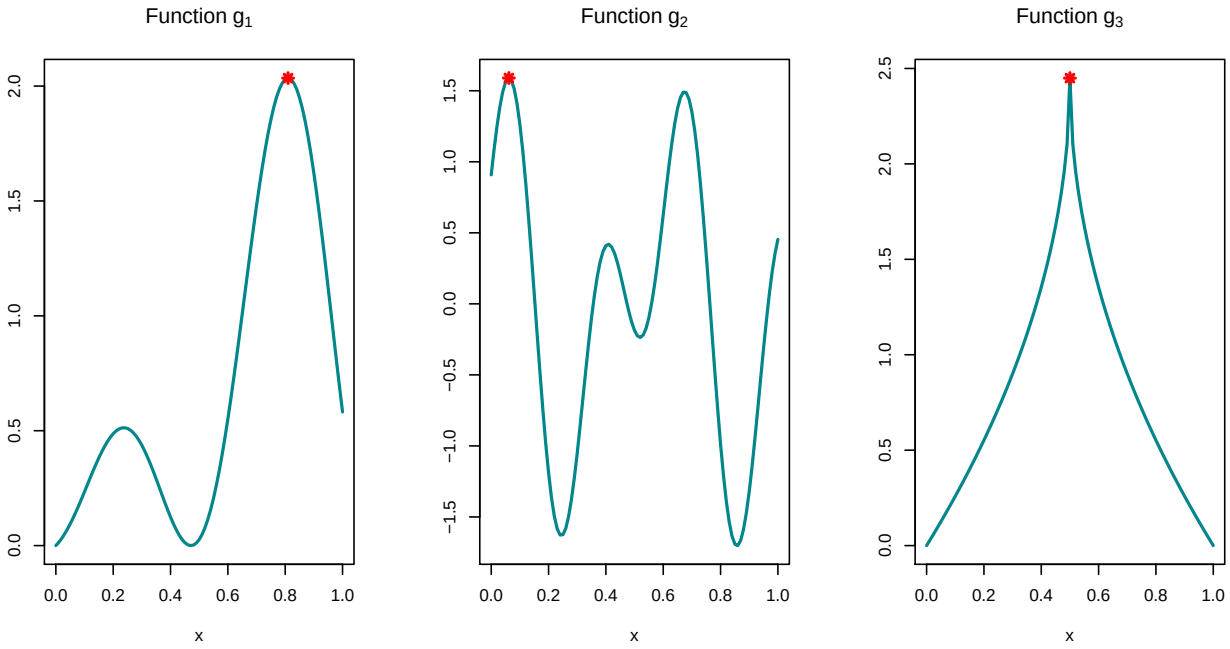

Figure 4.1: The functions g_1, g_2 and g_3, respectively. The red asterisk indicate the global maximum.

The first function is chosen because it has one local maximum and one global maximum for which the values of these maxima are not close to each other. The second function is chosen because it has a few local maximum and one global maximum. However, one of the local maximum is very close to the global maximum. And the last function is chosen, because it is a function with many local maxima that are close to the global maximum. In order to do a fair comparison between the methods, there is one restriction. The amount of samples used per method has to be equal. The methods will be run n times, which is specified in the corresponding tables. Since there is a lot of information in the second method, the results for its corresponding section will first start with results of one run. Before the results of the methods will be discussed it might be useful to look at the trajectories of the first two methods that are recursive in Figure 4.2. It is notable that all three methods quickly converge to the maximiser for functions g_1 and g_3. And for g_2, all three methods converge to either one of the boundaries.

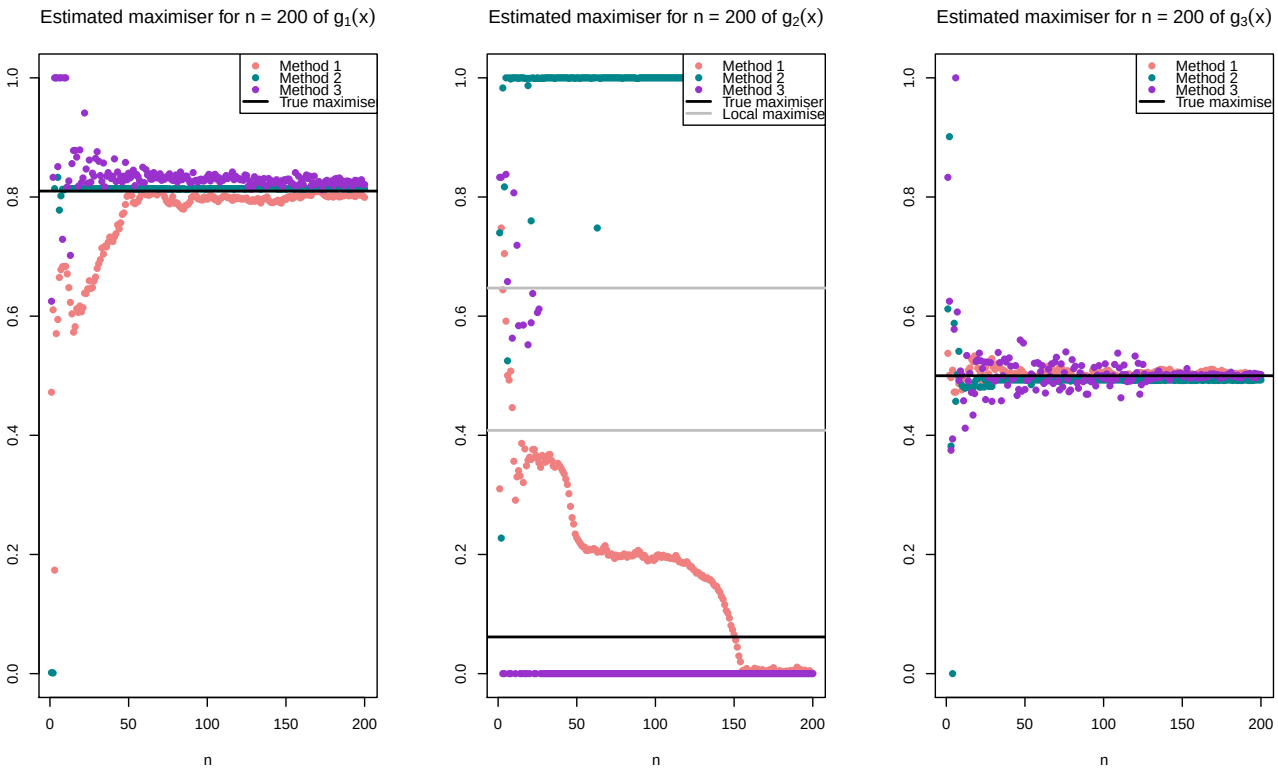

Figure 4.2: Trajectory of the estimated maximiser of functions g_1, g_2 and g_3 for method 1, method 2 and method 3, respectively, for $n = 200$ samples.

For the next sections, let x_{max} denote the true maximiser and let $\hat{x}_{max}$ denote the estimated maximiser. And let g_{max} denote the true maximum and $\hat{g}_{max}$ denote the estimated maximum.

4.1 Method 1: Stochastic Approximation

Now, the maximiser and maximum will be computed using the Kiefer-Wolfowitz algorithm. There are a few things to note before the functions can be analyzed. The algorithm only produces a local maximiser for a certain initial value. As explained before, there will be multiple initial values, which will perhaps give different maximisers. However, it is still not known which maximiser is the global one. In order to find the global maximiser, the function value of each local maximiser will be determined by the two most recent estimates of the function, namely $\frac{Y(\hat{x}_{max}+c_n)+Y(\hat{x}_{max}-c_n)}{2}$, where $\hat{x}_{max}$ is the last value of x_n. The sequences used are the ones suggested by Kiefer and Wolfowitz, namely $a_n = \frac{1}{n}$ and $c_n = n^{-\frac{1}{3}}$. Another thing to note is that this algorithm needs two samples for each iteration, $Y(x - c_n)$ and $Y(x + c_n)$. To make it a fair comparison, this algorithm can only use $\frac{n}{2}$ samples. For $n = 25$, $n = 50$ and $n = 125$ only one starting value will be used and for $n = 250$ two starting values will be used. If there are two starting values, there will be two estimates for the maximiser and maximum. The estimated maximiser corresponding to the best estimated maximum will be the estimated maximiser for $n = 250$. Each starting value will be sampled from a uniform distribution between 0 and 1.

Function g_1

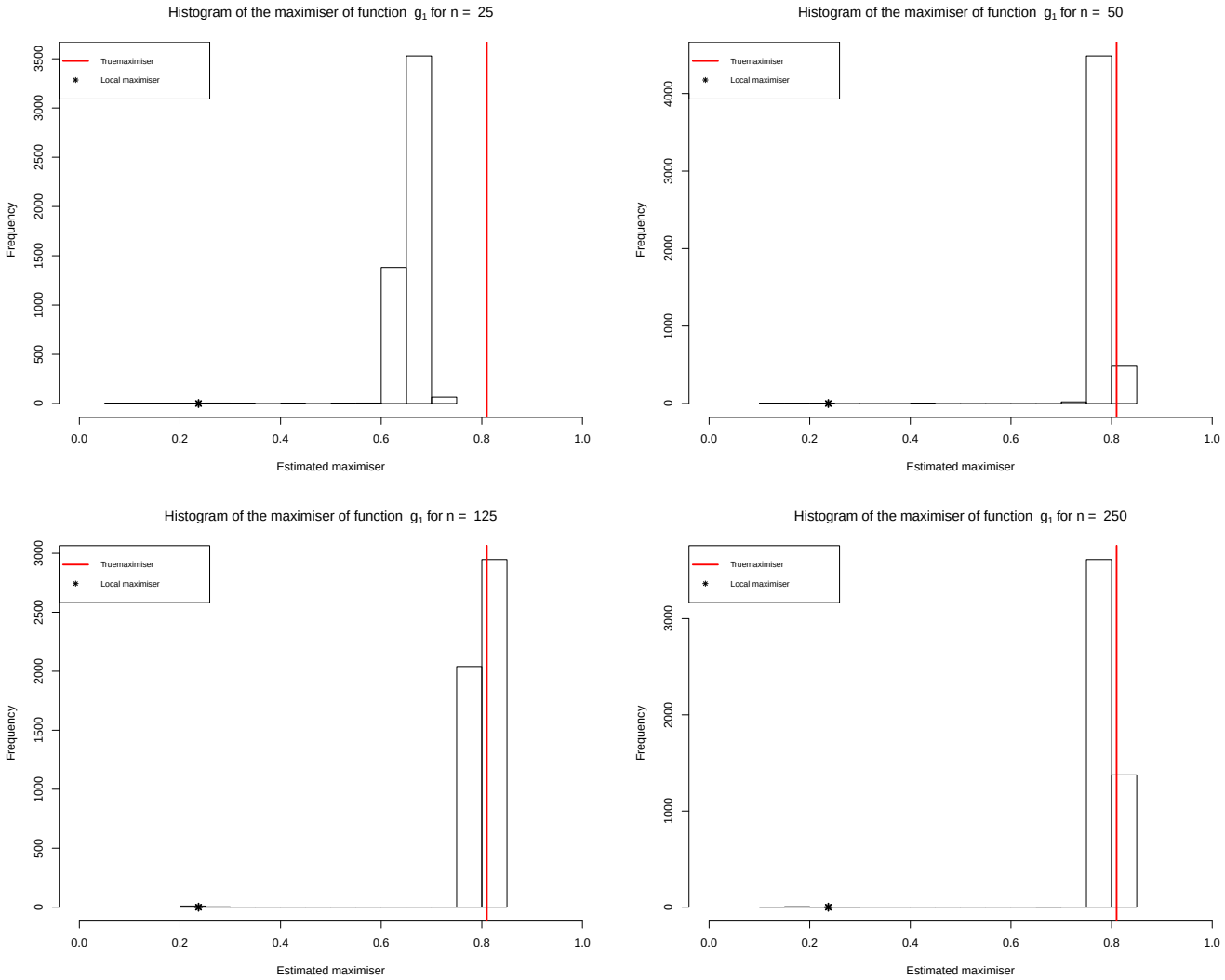

Figure 4.3: Histograms of the estimated maximiser of g_1 for values $n = 25, 50, 125$ and 500 for 5000 runs. The vertical red line indicated the true maximiser and the black dot indicates the local maximiser.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of $\hat{g}_{max}$	Mean absolute error of $\hat{g}_{max}$
25	0.657 ± 0.070	0.156	0.436 ± 0.080	1.158
50	0.776 ± 0.057	0.037	0.089 ± 0.071	1.506
125	0.801 ± 0.012	0.008	0.443 ± 0.070	1.151
250	0.791 ± 0.046	0.018	0.178 ± 0.59	1.417

Table 4.1: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_1 for $n = 25, 50, 125$ and 250 for 5000 runs.

In Table 4.1 it can be seen that for this function the mean absolute error of $\hat{x}_{max}$ is the worst for $n = 25$ and the mean estimated maximiser is given by 0.657. The histogram in Figure 4.3 shows that the algorithm always underestimates the maximiser. This means that there were not enough samples to accurately estimate the maximiser. For larger n the estimates of the maximiser are relatively accurate. The estimation of maximum at first does not seem to stabalize for larger n.

Function g_2

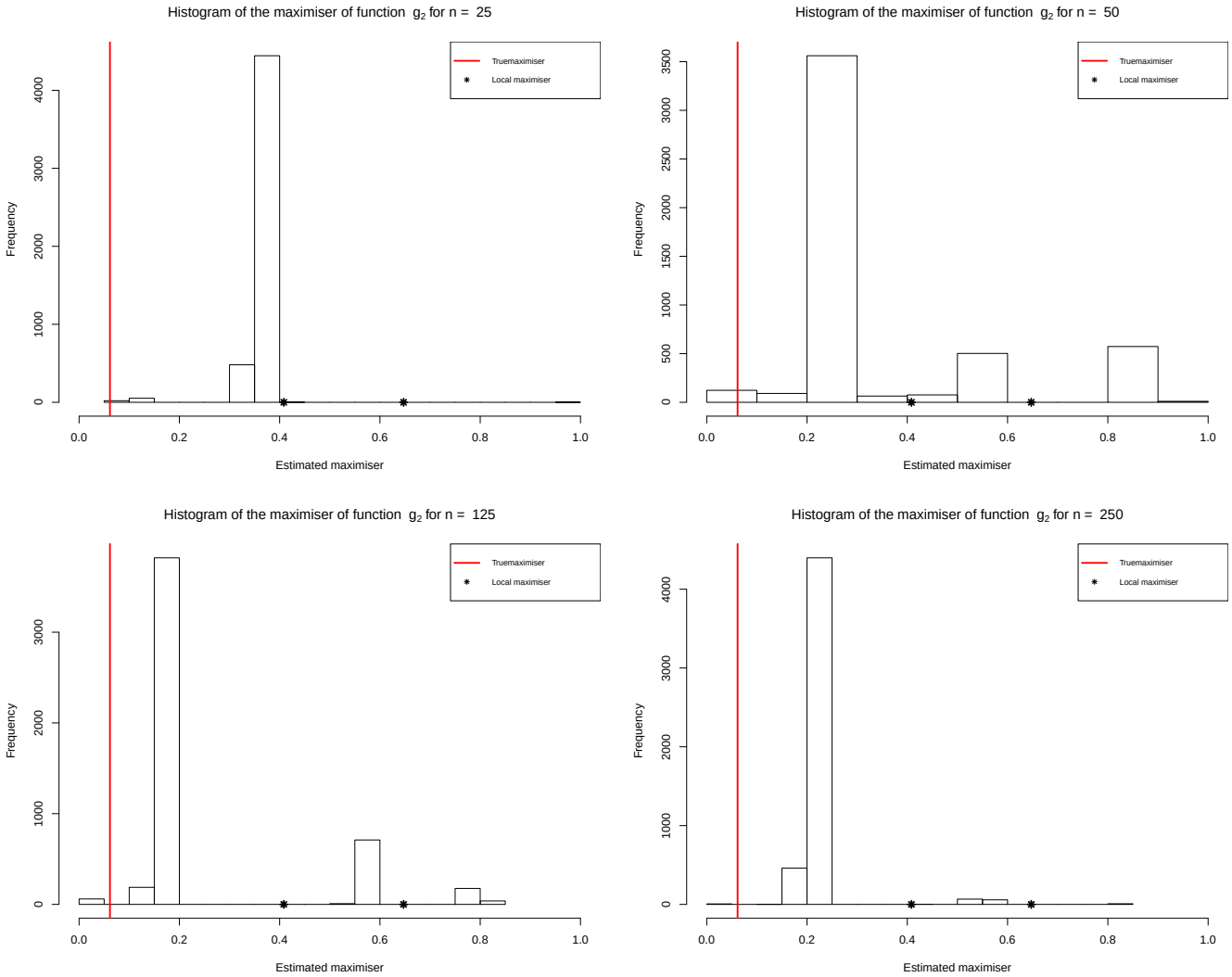

Figure 4.4: Histograms of the estimated maximiser of g_2 for values $n = 25, 50, 125$ and 250 for 5000 runs. The vertical red line indicated the true maximiser and the black dots indicates the local maximisers.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of g_{max}	Mean absolute error of g_{max}
25	0.375 $\pm$ 0.090	0.313	1.396 $\pm$ 0.119	0.354
50	0.218 $\pm$ 0.016	0.156	-0.078 $\pm$ 0.114	1.829
125	0.213 $\pm$ 0.164	0.152	0.279 $\pm$ 0.107	1.470
250	0.396 $\pm$ 0.284	0.334	0.192 $\pm$ 0.074	1.557

Table 4.2: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_2 for $n = 25, 50, 125$ and 250 for 5000 runs.

In Figure 4.4 it can be seen that for $n = 25$ the algorithm estimates the maximiser to be in between those maxima, which is the location of another local maximum. For the other values of n it seems to converge to a value that is far from both the global and local maximiser. In Table 4.2 it can be seen that the mean absolute error of both the estimated maximiser and maximum is relatively large.

Function g_3

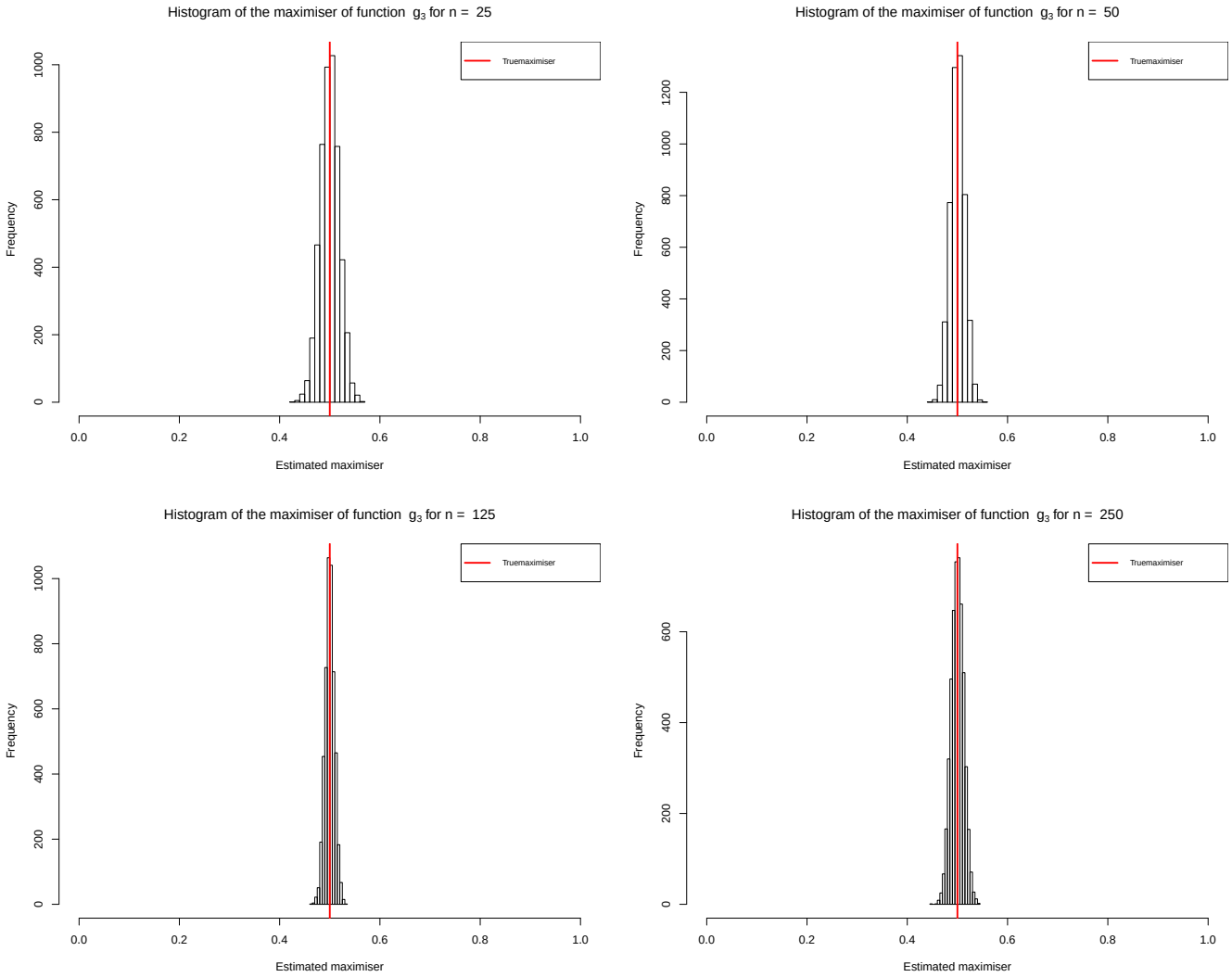

Figure 4.5: Histograms of the estimated maximiser of g_3 for values $n = 25, 50, 125$ and 250 for 5000 runs. The vertical red line indicated the true maximiser.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of $\hat{g}_{max}$	Mean absolute error of $\hat{g}_{max}$
25	0.507 ± 0.013	0.012	0.172 ± 0.070	0.827
50	0.501 ± 0.009	0.009	0.262 ± 0.070	0.737
125	0.500 ± 0.005	0.011	0.366 ± 0.070	0.633
250	0.498 ± 0.008	0.011	0.330 ± 0.058	0.669

Table 4.3: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_2 for $n = 25, 50, 125$ and 250 for 5000 runs.

Since this function is a function with only a global maximum at $x = 0.5$, this algorithm does well. This can be observed in the histogram in Figure 4.5 and Table 4.3 as the estimated maximiser is always near $x = 0.5$. For all n the mean absolute error of the estimated maximiser is low. However, for this function the estimate of the maximum is off.

4.2 Method 2: Non parametric Bayesian approach

The estimation of the maximiser will be the last sample. Note that the y values are given by $f(\hat{x}_{max}) + \sigma\epsilon$. This means that the y values are normally distributed for large n. The estimation of the maximum will then be given by the y values.

Function g_1

Figures 4.6, 4.7, 4.8, 4.9 and 4.10 show how the global maximiser gets estimated by the algorithm. It is notable that for this function, the density of both the estimated maximiser and maximum converge to their true value as the density is concentrated around the true values.

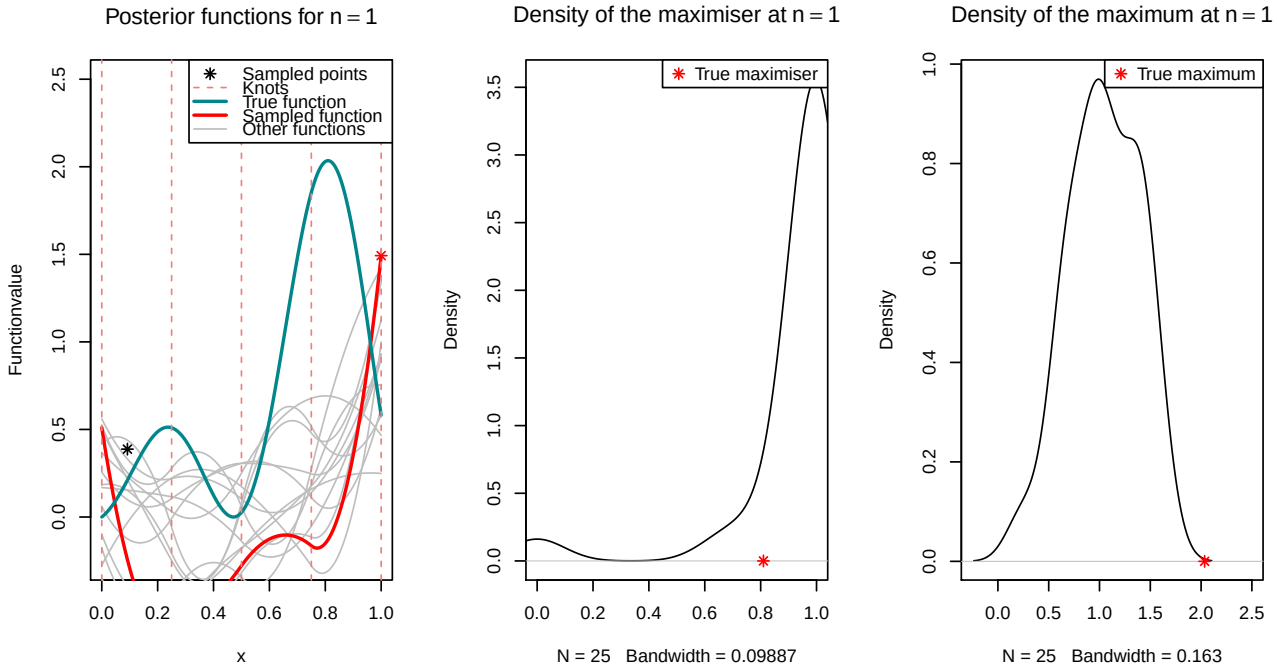

Figure 4.6: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 1$.

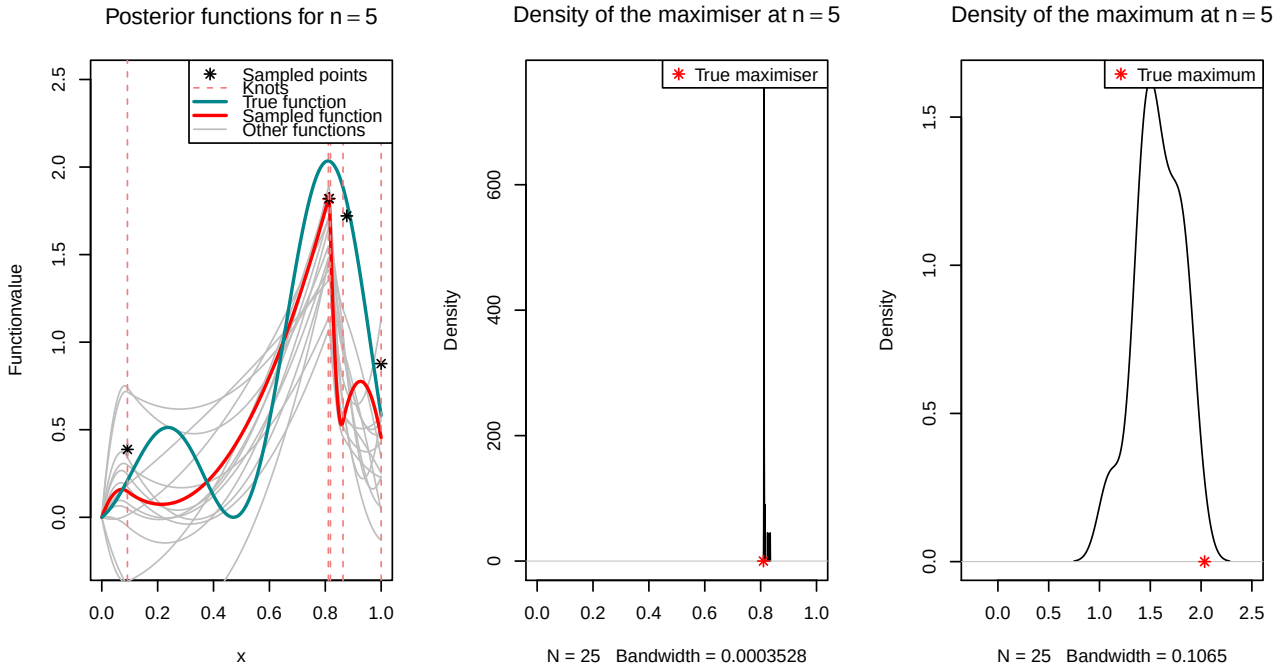

Figure 4.7: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 5$.

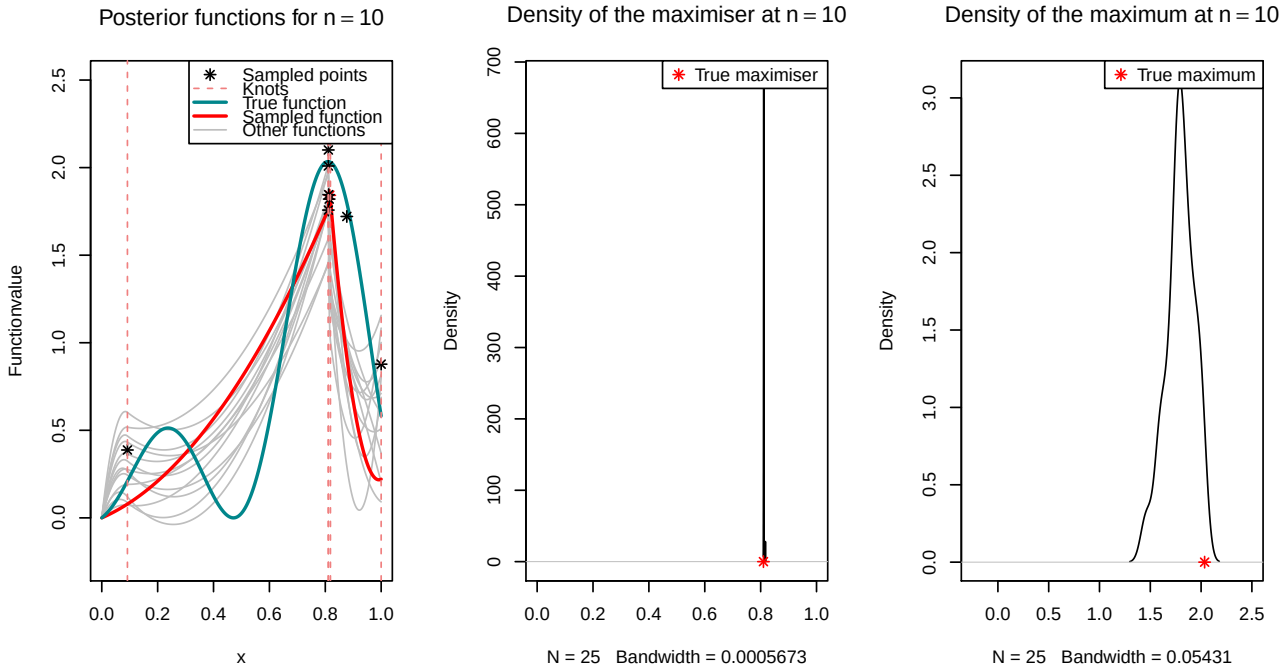

Figure 4.8: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 10$.

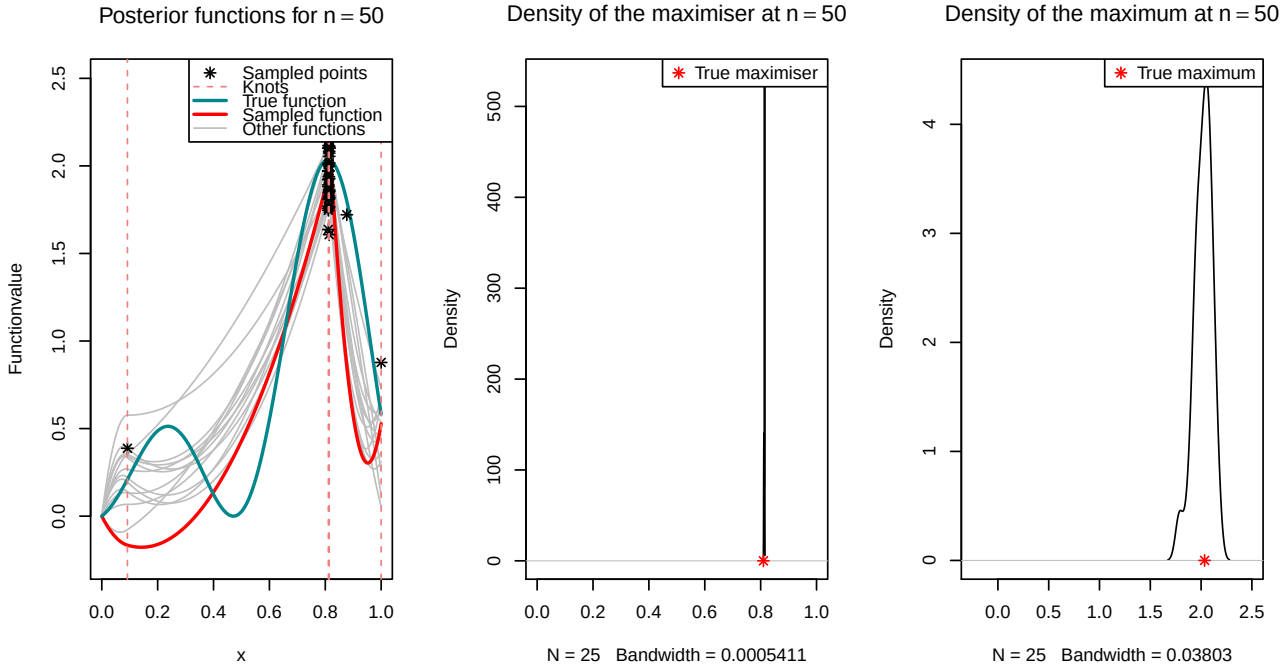

Figure 4.9: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 50$.

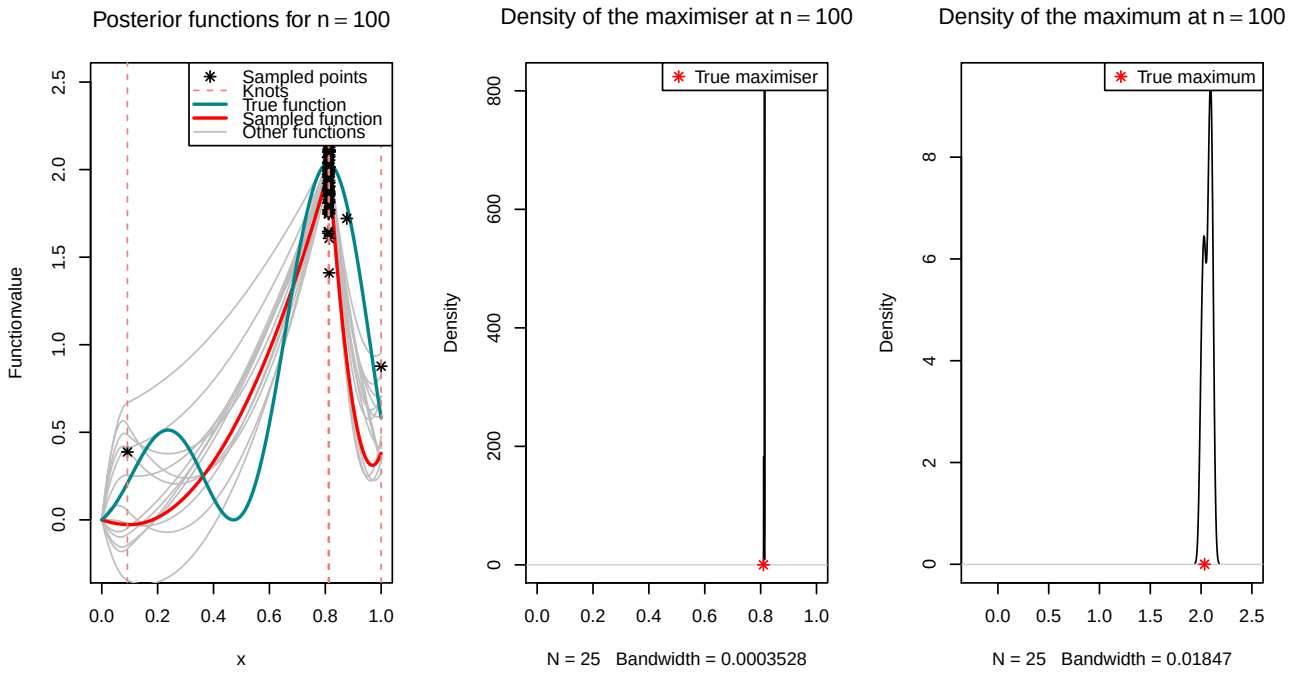

Figure 4.10: The posterior functions are the grey lines, knots, the sampled function is the red line and the sampled points are the black asterisks (left). The density of the maximiser (middle). The density of the maximum (right) at iteration $n = 100$.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of $\hat{g}_{max}$	Mean absolute error of $\hat{g}_{max}$
50	0.826 ± 0.080	0.055	1.29 ± 0.294	0.309
100	0.827 ± 0.064	0.049	1.36 ± 0.259	0.238
250	0.826 ± 0.055	0.044	1.42 ± 0.212	0.178
500	0.823 ± 0.053	0.042	1.45 ± 0.183	0.150

Table 4.4: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_1 for $n = 50, 100, 250$ and 500 for 5000 runs.

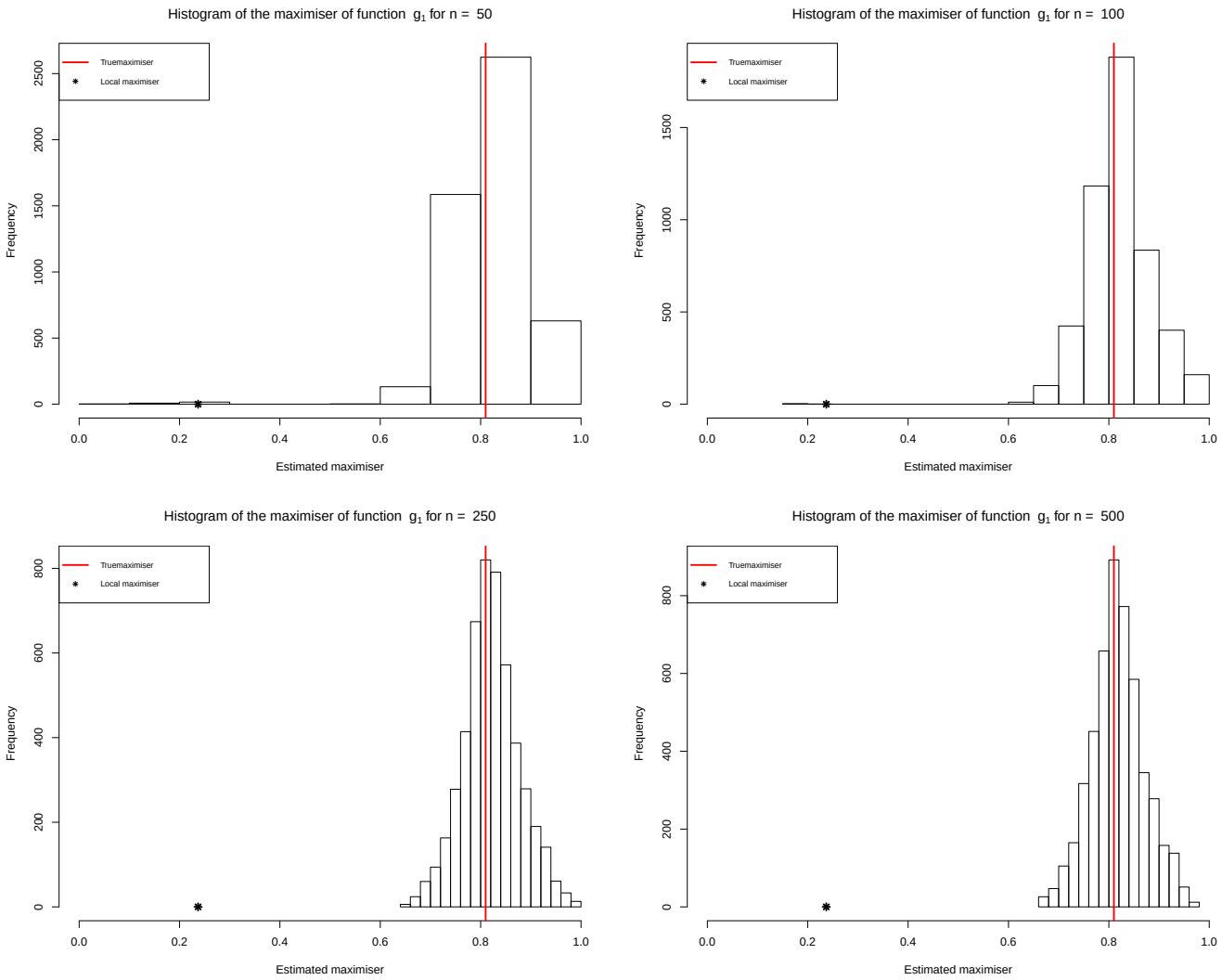

Figure 4.11: Histograms of the estimated maximiser of g_1 for values $n = 50, 100, 250$ and 500 for 5000 runs. The vertical red line indicated the true maximiser and the black dot indicates the local maximiser.

Figure 4.11 shows that the maximiser gets estimated very well. For $n = 50$ and $n = 100$ it sometimes estimated lower values close to 0.2. This does not happen for $n = 250$ and $n = 500$. In Table 4.4 it van be seen that the mean absolute error of the estimated maximiser does get smaller for larger n and this also holds for the mean estimated maximum.

Function g_2

Figures 4.12, 4.13, 4.14, 4.15 and 4.16 show how the global maximiser does not get estimated by the algorithm. It is notable that for this function, the algorithm converges to a local maximiser and its maximum value.

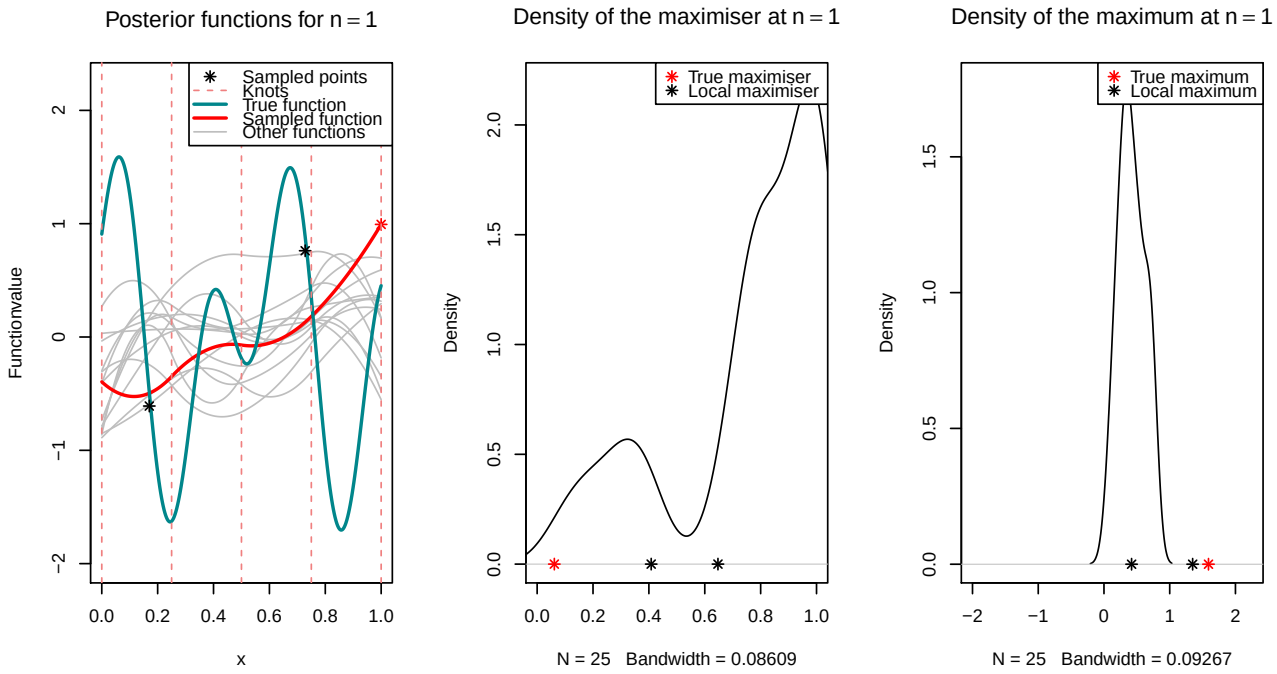

Figure 4.12: The posterior functions, knots, sampled function and sampled points of g_2 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 1$.

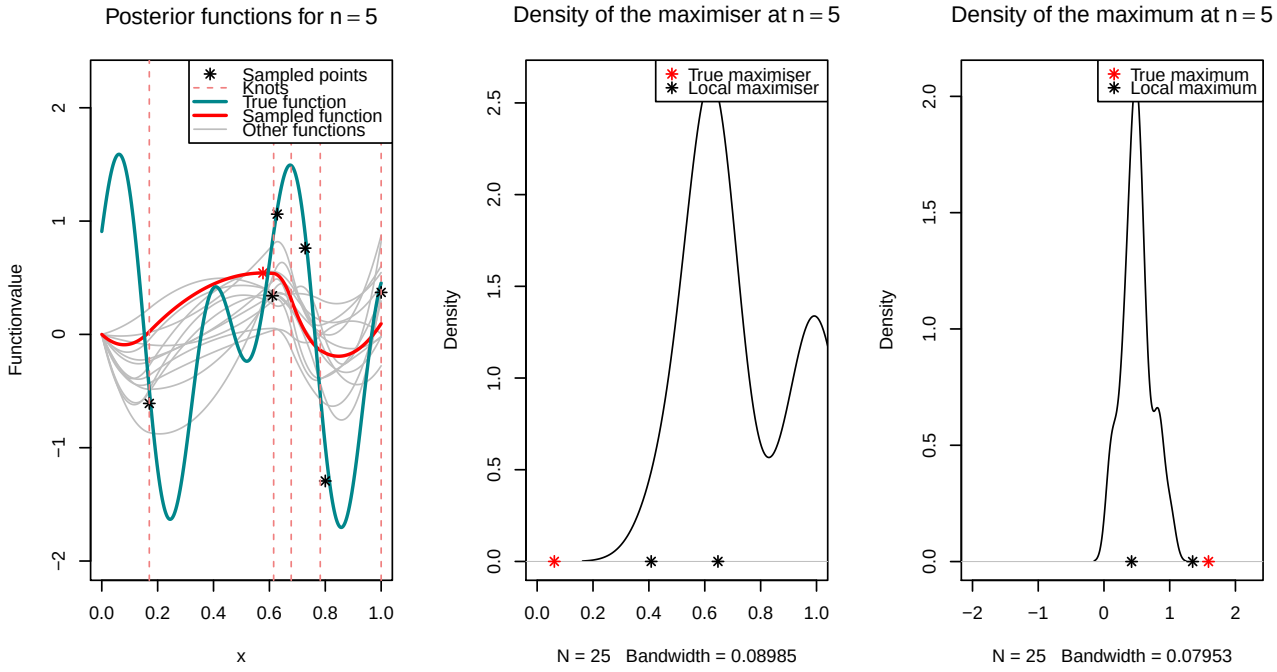

Figure 4.13: The posterior functions, knots, sampled function and sampled points of g_2 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 5$

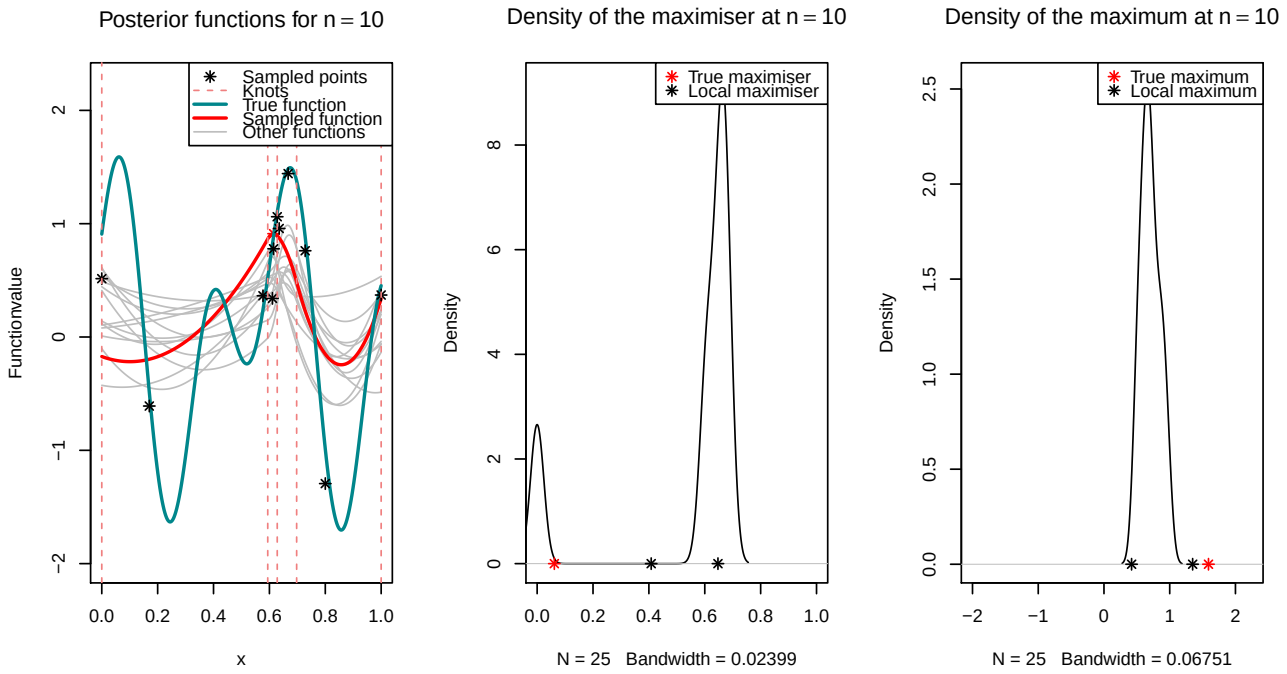

Figure 4.14: The posterior functions, knots, sampled function and sampled points of g_2 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 10$

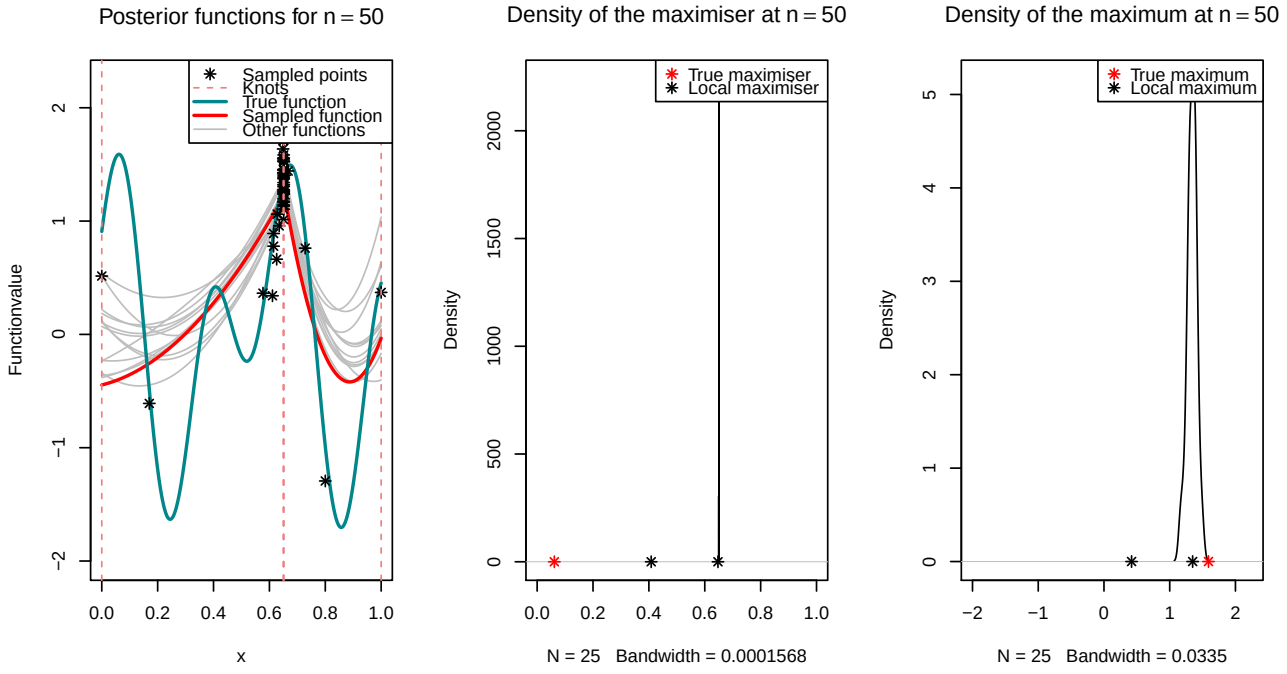

Figure 4.15: The posterior functions, knots, sampled function and sampled points of g_2 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 50$

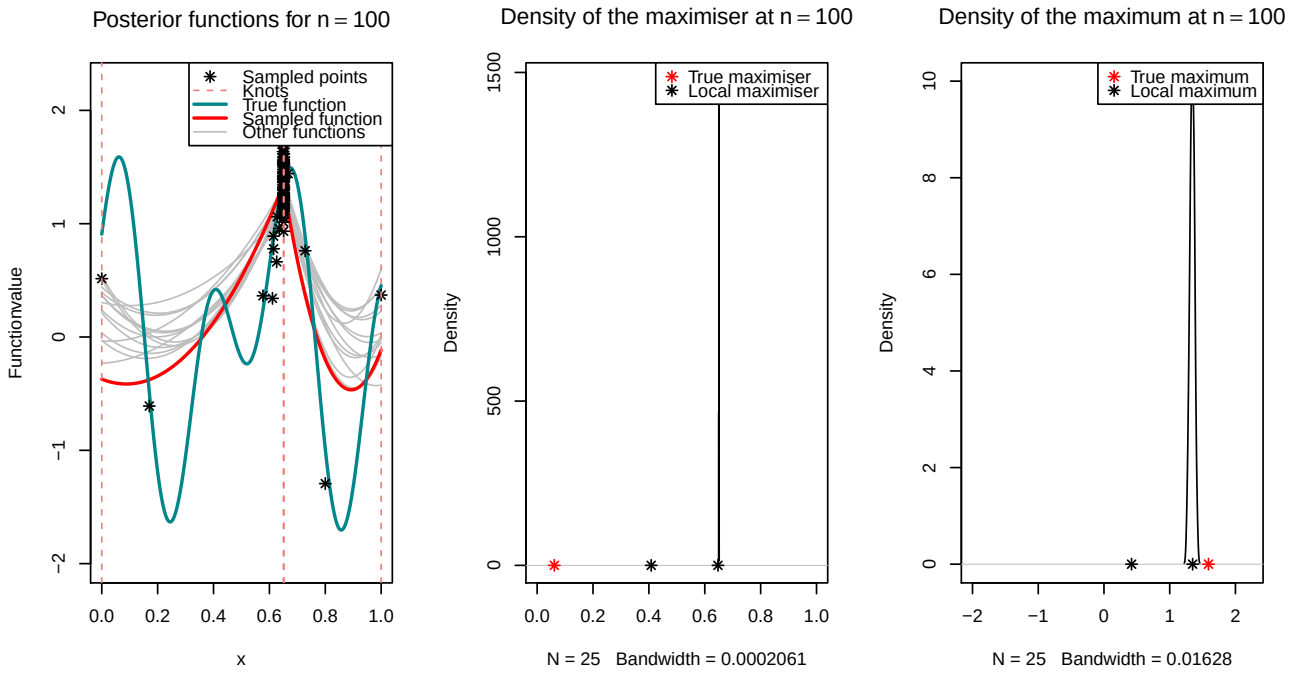

Figure 4.16: The posterior functions, knots, sampled function and sampled points of g_2 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 100$

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of g_{max}	Mean absolute error of g_{max}
50	0.356 ± 0.369	0.159	1.100 ± 0.373	0.648
100	0.362 ± 0.372	0.203	1.161 ± 0.380	0.586
250	0.354 ± 0.370	0.146	1.224 ± 0.384	0.544
500	0.355 ± 0.307	0.218	1.236 ± 0.385	0.525

Table 4.5: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_2 for $n = 50, 100, 250$ and 500 for 5000 runs.

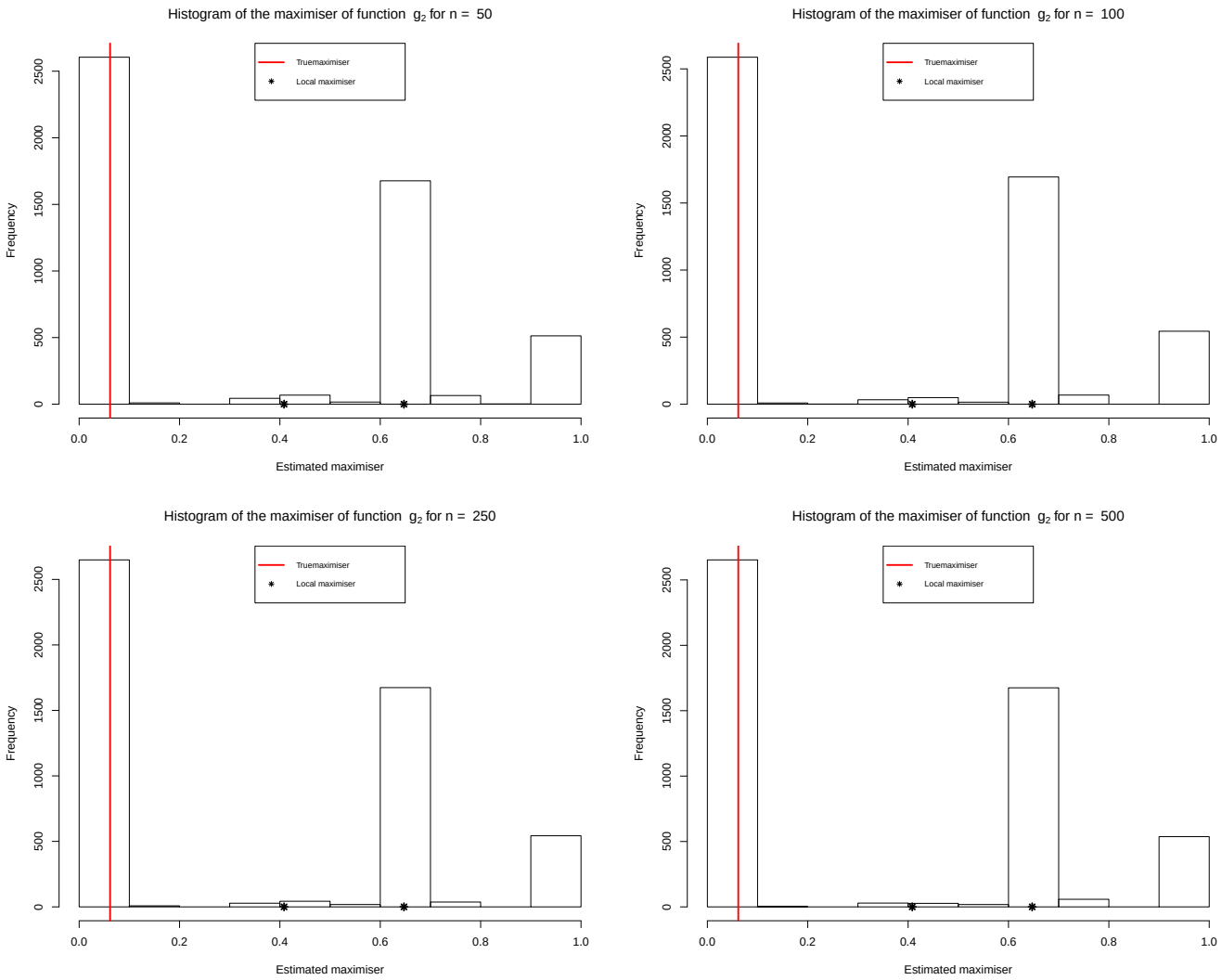

Figure 4.17: Histograms of the estimated maximiser of g_2 for values $n = 50, 100, 250$ and 500 for 5000 runs. The vertical red line indicated the true maximiser and the black dots indicates the local maximisers.

In Figure 4.17 it can be seen that most of the 5000 runs, it estimates the true maximiser. However, a large portion of the runs also estimates a local maximiser. This is in accordance with Table 4.5 since the mean is very off, since it calculates the mean between mostly those two values. The mean absolute error of the estimated maximiser does not get better for larger n. However, the mean of the estimated maximum does get better.

Function g_3

Figures 4.18, 4.19, 4.20, 4.21 and 4.22 show how the global maximiser gets estimated very well by the algorithm. However, it is notable that the maximum gets underestimated.

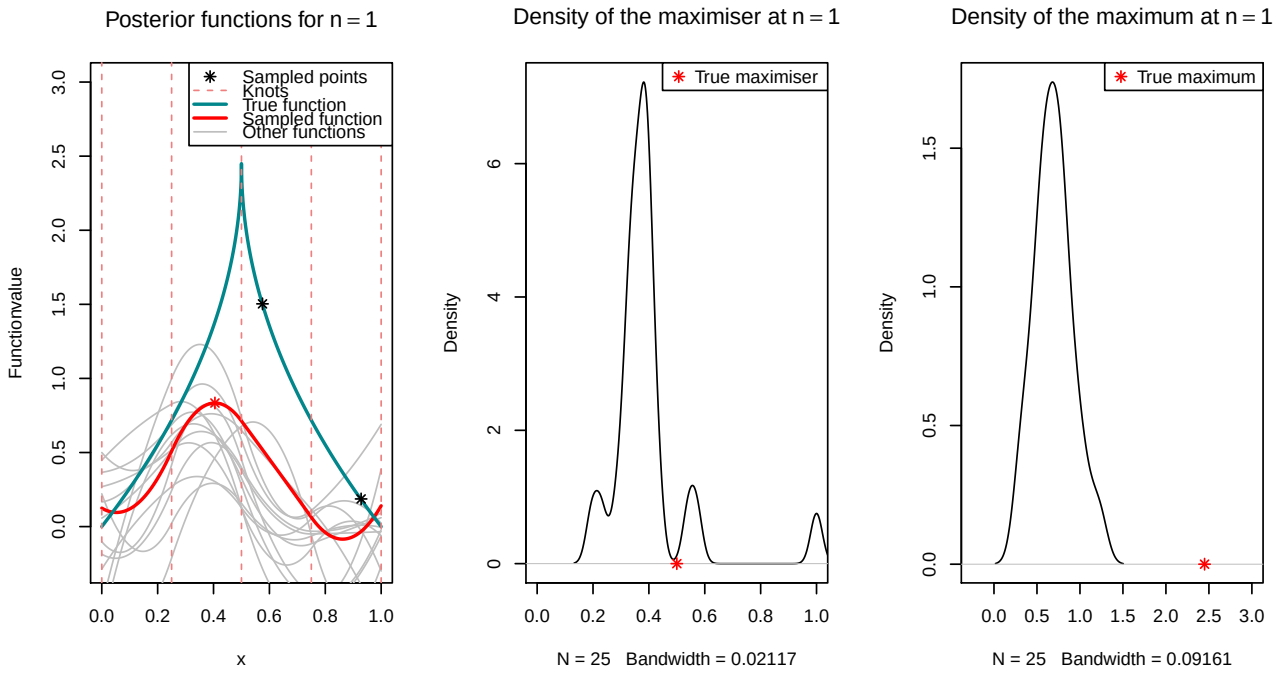

Figure 4.18: The posterior functions, knots, sampled function and sampled points of g_3 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 1$.

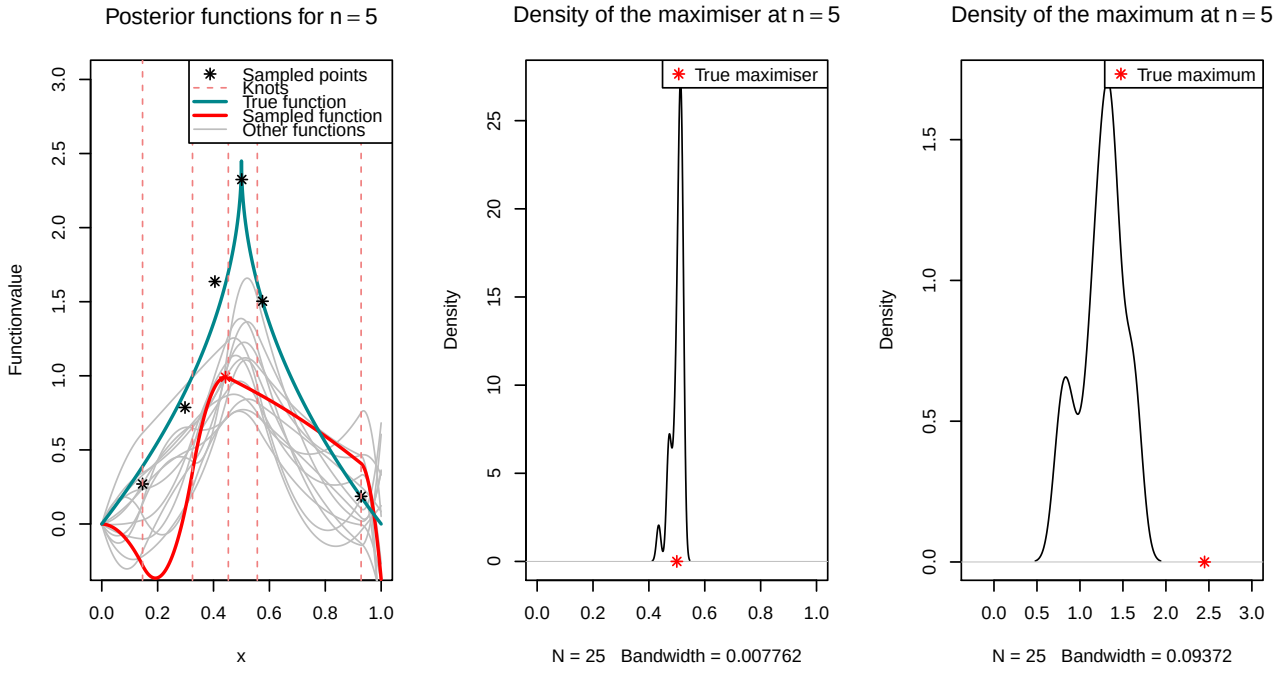

Figure 4.19: The posterior functions, knots, sampled function and sampled points of g_3 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 5$.

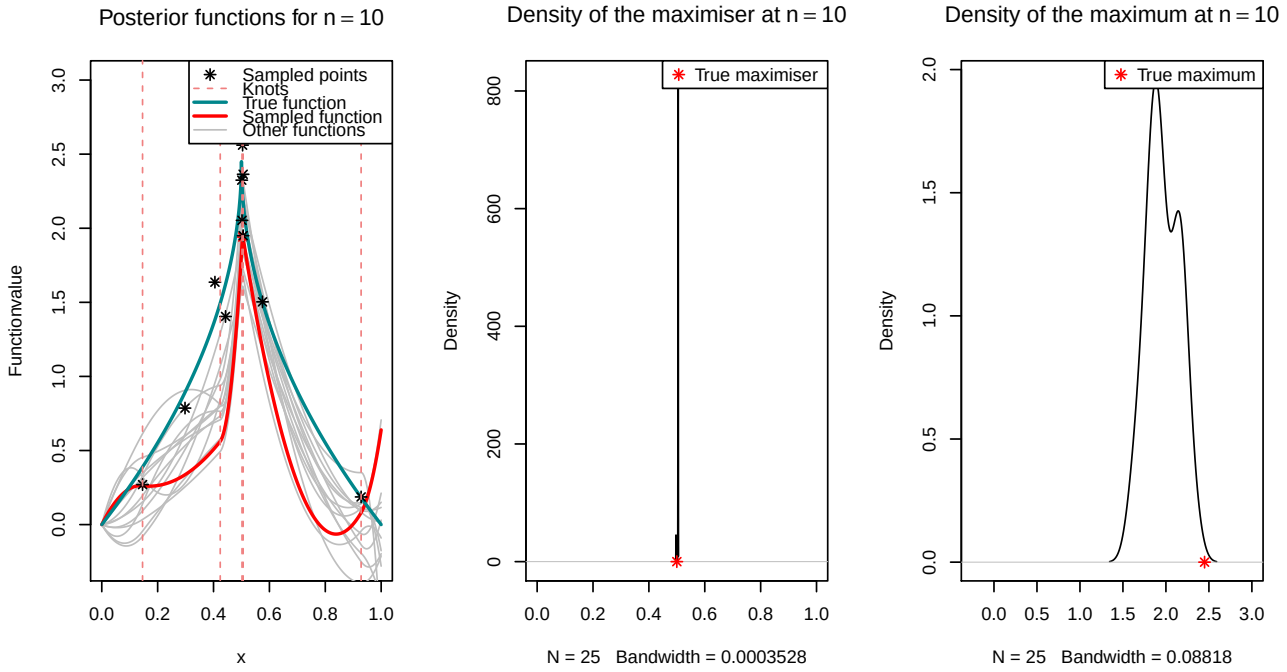

Figure 4.20: The posterior functions, knots, sampled function and sampled points of g_3 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 10$.

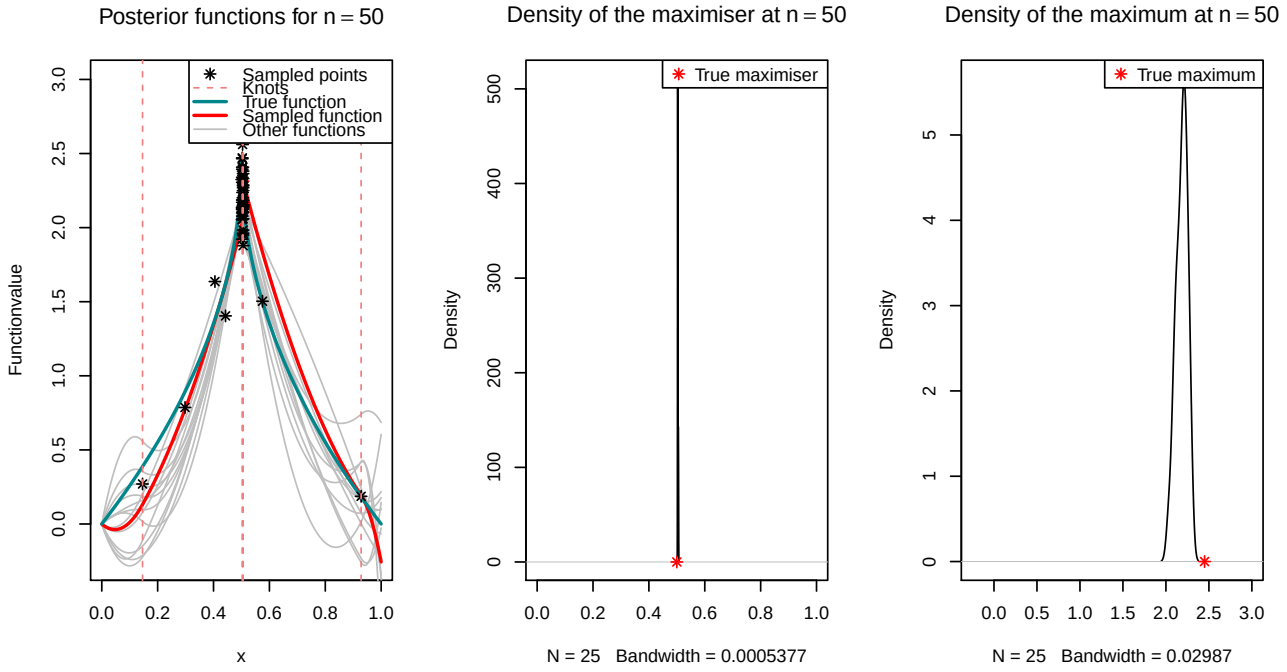

Figure 4.21: The posterior functions, knots, sampled function and sampled points of g_3 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 50$.

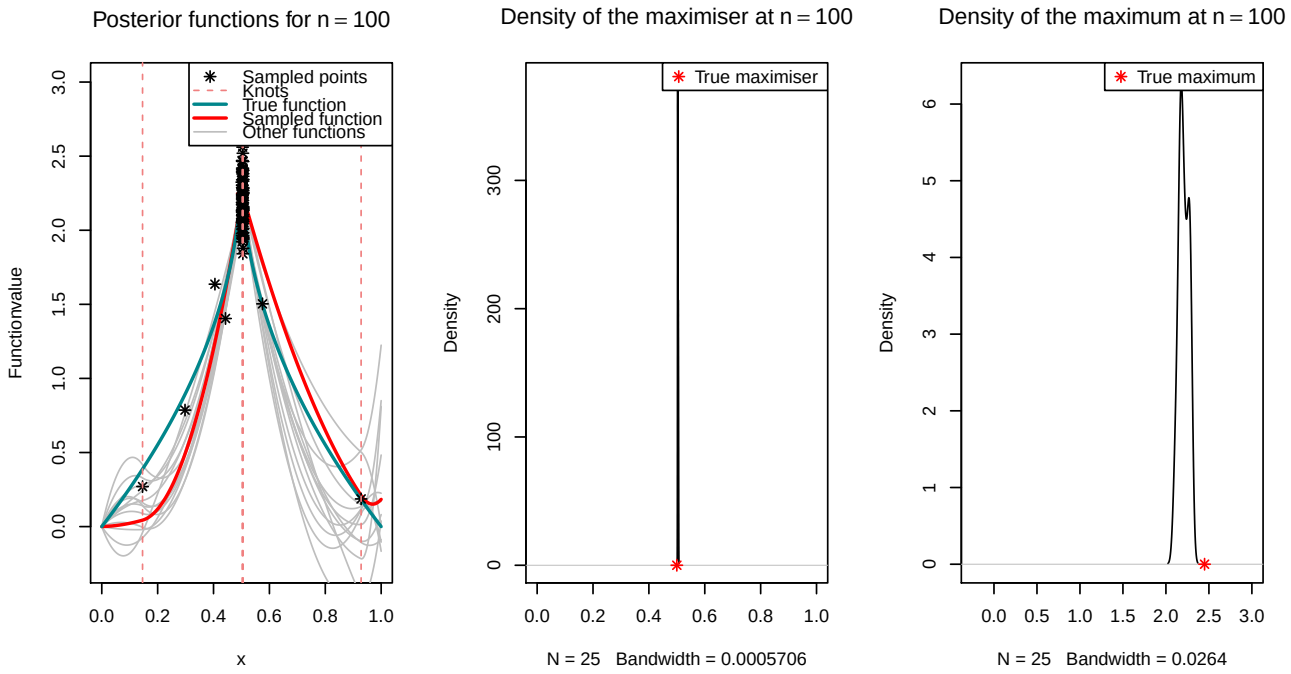

Figure 4.22: The posterior functions, knots, sampled function and sampled points of g_3 (left) with the density of the maximiser (middle) and maximum (right) at iteration $n = 100$.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of g_{max}	Mean absolute error of g_{max}
50	0.499 ± 0.063	0.039	1.55 ± 0.213	0.900
100	0.498 ± 0.047	0.030	1.74 ± 0.216	0.708
250	0.497 ± 0.031	0.019	1.86 ± 0.251	0.587
500	0.495 ± 0.023	0.014	2.04 ± 0.210	0.414

Table 4.6: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_3 for $n = 50, 100, 250$ and 500 for 5000 runs.

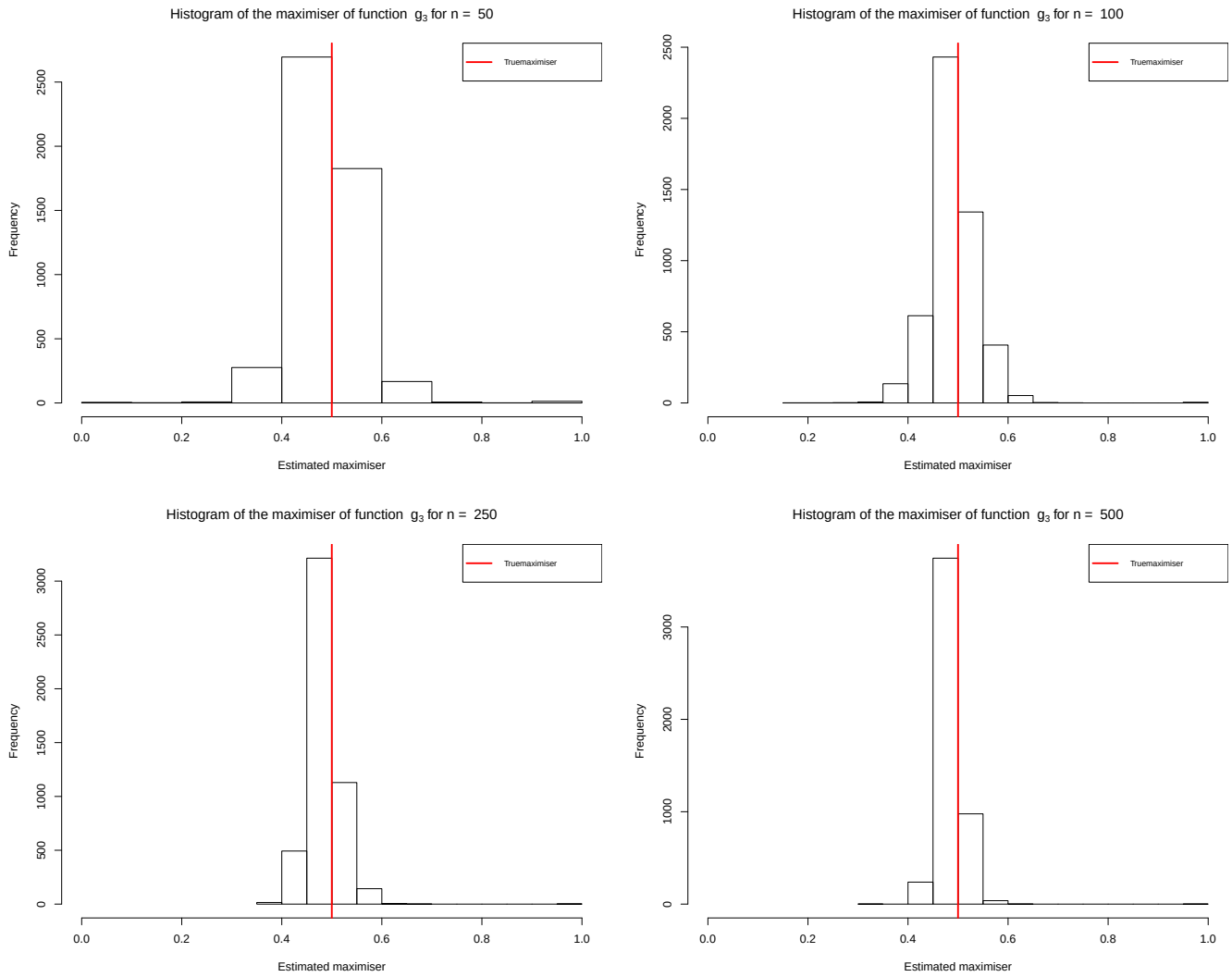

Figure 4.23: Histograms of the estimated maximiser of g_1 for values $n = 50, 100, 250$ and 500 for 5000 runs. The vertical red line indicated the true maximiser.

For this function, it can be seen in Figure 4.23 the maximiser is estimated well. Table 4.6 shows that the estimates for both the maximiser and maximum get better for larger n.

4.3 Method 3: Non recursive method

In this section, this will be run 5000 times and the mean of the maximiser and maximum will be extracted. This will be done for n samples. After that, the mean absolute error of the maximiser and Mean absolute error of the maximum will be computed. This is the difference between the mean maximiser and the true maximiser and likewise for the maximum. From these tables, the threshold for the recursive method will be determined.

Function g_1

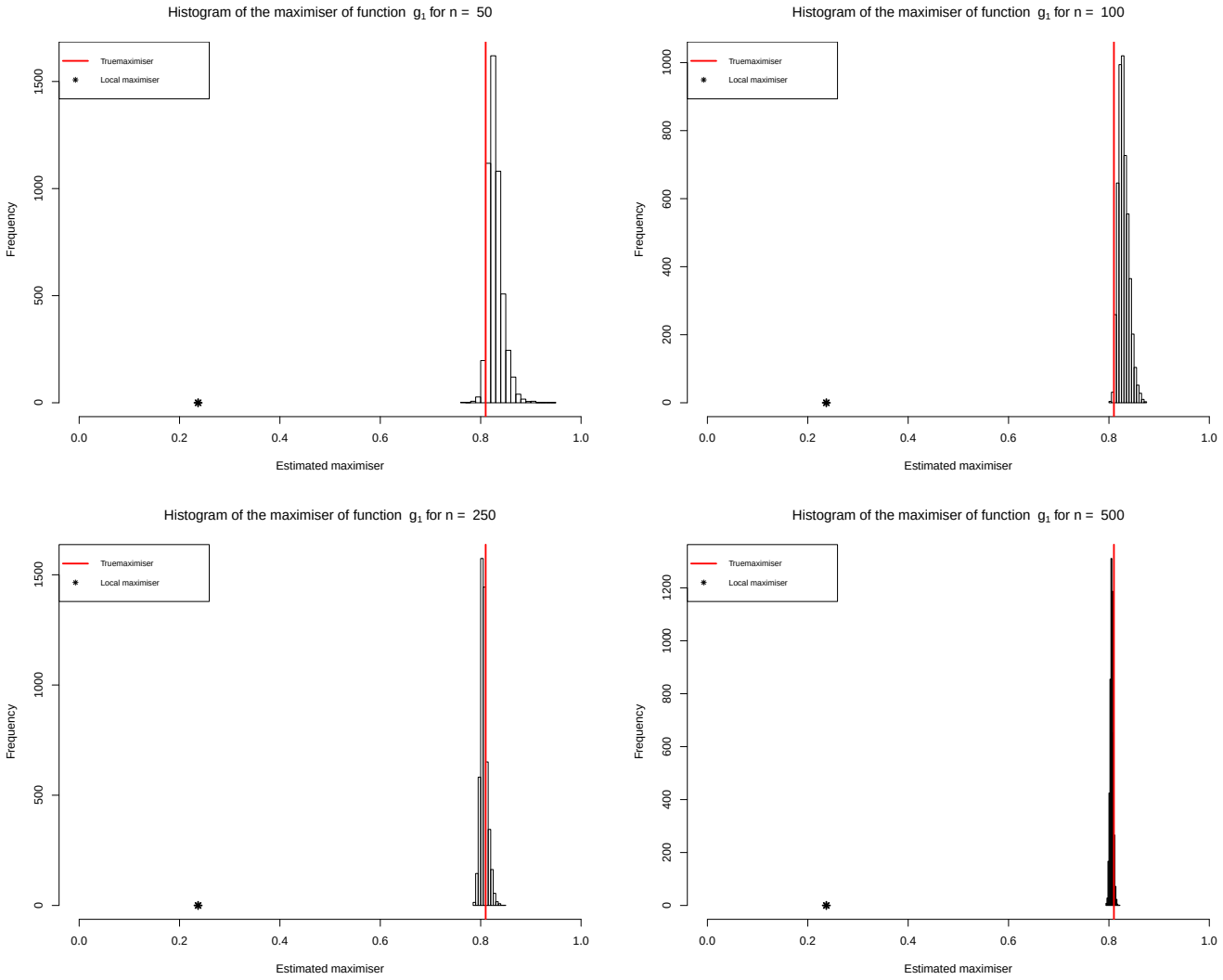

Figure 4.24: Histograms of the estimated maximiser of g_1 for values $n = 50, 100, 250$ and 500 for 5000 runs. The vertical red line indicated the true maximiser and the black dot indicates the local maximiser.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of g_{max}	Mean absolute error of g_{max}
50	0.829 ± 0.015	0.020	1.651 ± 0.122	0.382
100	0.829 ± 0.010	0.019	1.830 ± 0.070	0.204
250	0.806 ± 0.007	0.006	1.931 ± 0.045	0.103
500	0.806 ± 0.003	0.004	1.998 ± 0.029	0.039

Table 4.7: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_1 for $n = 50, 100, 250$ and 500 for 5000 runs.

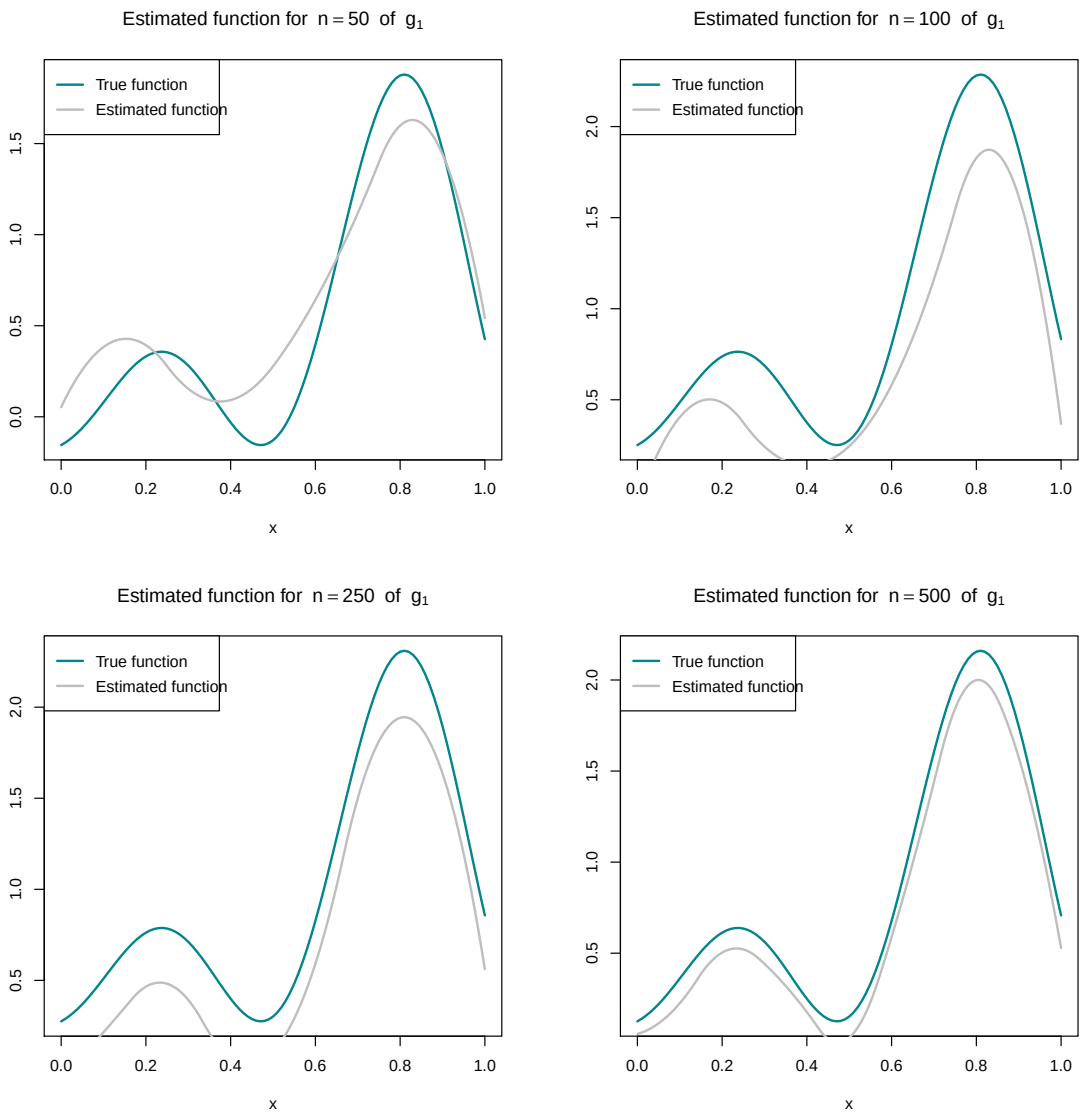

Figure 4.25: The estimated function for $n = 50, 100, 250$ and 500 samples of function g_1.

For this function it can be seen in Figure 4.25 that for larger n the function has a better estimation. This is in accordance with Table 4.7 where it can be seen that the mean absolute error of both the maximiser and maximum get smaller for larger n. In Figure 4.24 it can be seen that the estimated maximiser is concentrated around the true maximiser. And it never estimates the local maximiser at $x = 0.237$.

Function g_2

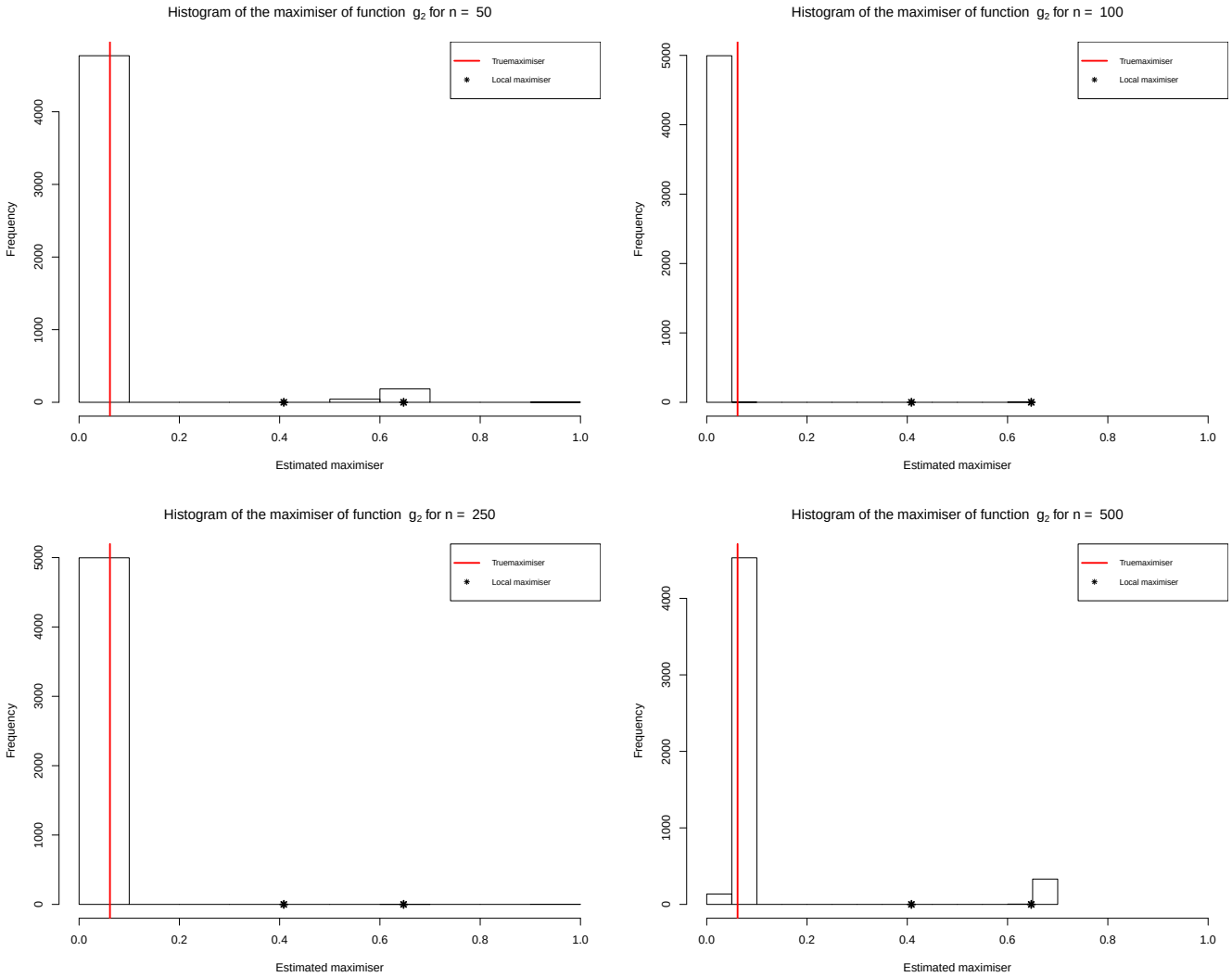

Figure 4.26: Histograms of the estimated maximiser of g_2 for values $n = 50, 100, 250$ and 500 for 5000 runs. The vertical red line indicated the true maximiser and the black dots indicates the local maximisers.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of g_{max}	Mean absolute error of g_{max}
50	0.043 ± 0.128	0.069	1.256 ± 0.221	0.337
100	0.009 ± 0.022	0.053	1.643 ± 0.189	0.155
250	0.027 ± 0.037	0.035	1.452 ± 0.155	0.182
500	0.098 ± 0.148	0.041	1.584 ± 0.091	0.075

Table 4.8: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_2 for $n = 50, 100, 250$ and 500 for 5000 runs.

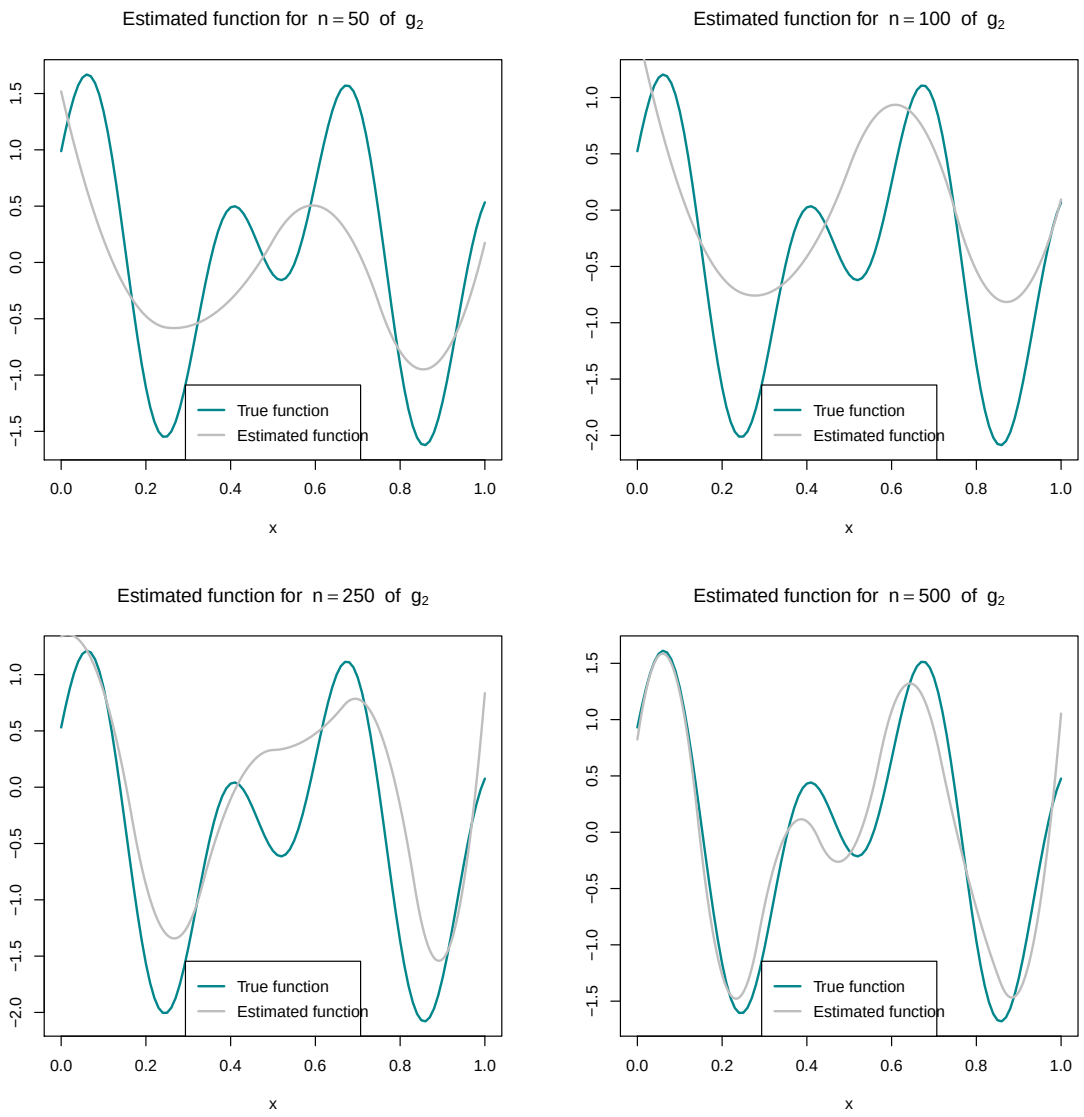

Figure 4.27: The estimated function for $n = 50, 100, 250$ and 5000 samples of function g_2.

The plots of the estimated function 4.27 it can be seen that the estimation is very off for $n = 50$ and $n = 100$ where the structure of the true function cannot be seen. For $n = 250$ this gets better. In Figure 4.24 it can be seen that for small n, in this case $n = 50$, $n = 100$ and $n = 250$, it does estimate the maximiser well for most of the 5000 runs. However, it does sometimes take other values between 0 and 1 as a estimated maximiser. In Table 4.8 it can be seen that the mean absolute error of the maximiser gets better for $n = 50$ to $n = 250$. For $n = 500$ it goes up a little. The mean absolute error of the maximum does get better for larger n.

Function g_3

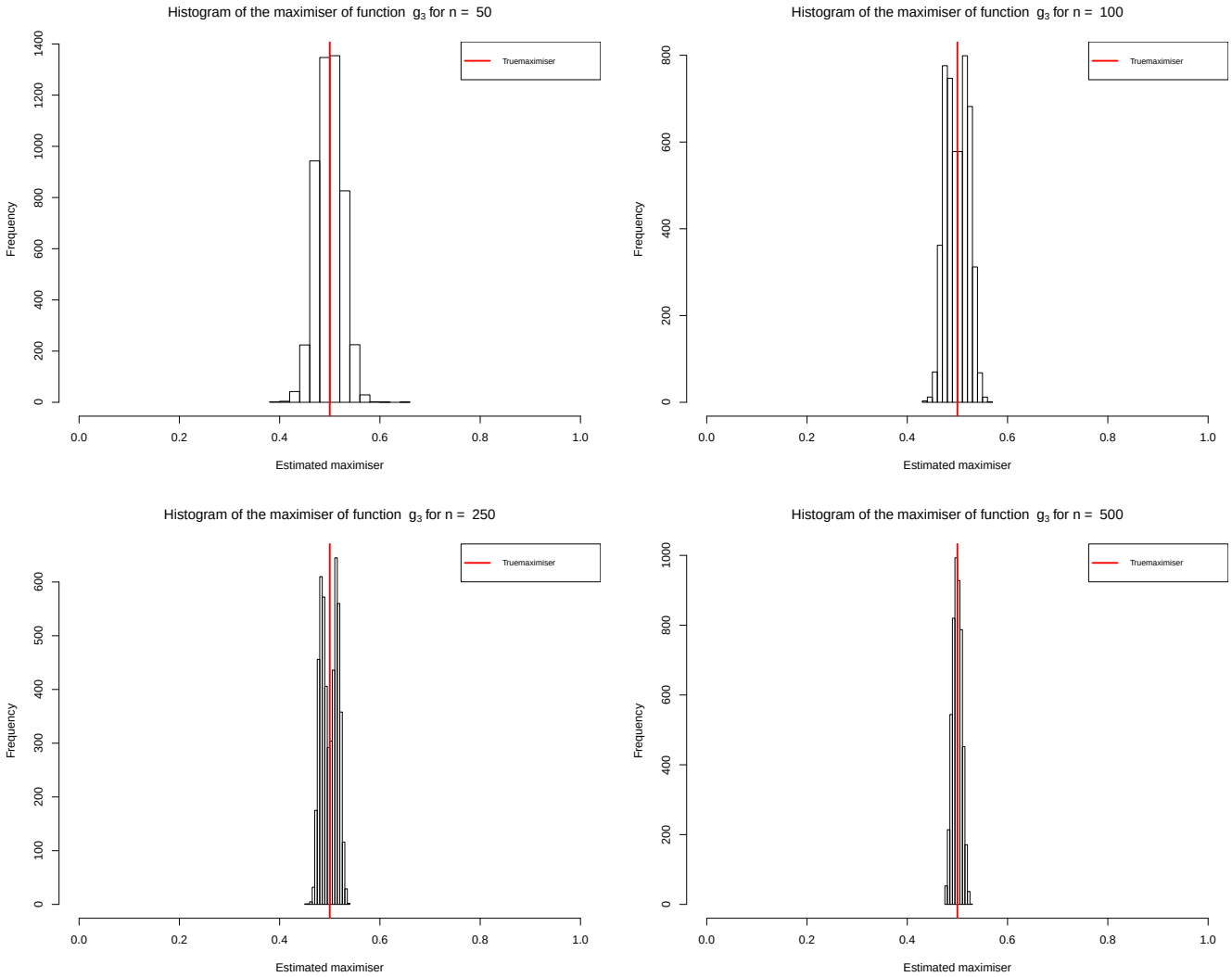

Figure 4.28: Histograms of the estimated maximiser of g_1 for values $n = 50, 100, 250$ and 500 for 500 runs. The vertical red line indicated the true maximiser.

n	Mean of $\hat{x}_{max}$	Mean absolute error of $\hat{x}_{max}$	Mean of g_{max}	Mean absolute error of g_{max}
50	0.500 ± 0.023	0.019	1.454 ± 0.112	0.994
100	0.500 ± 0.021	0.018	1.532 ± 0.078	0.917
250	0.499 ± 0.015	0.013	1.710 ± 0.060	0.738
500	0.499 ± 0.009	0.007	1.869 ± 0.044	0.579

Table 4.9: Table of the mean estimated maximiser and maximum plus-minus the standard deviation and their absolute error for function g_3 for $n = 50, 100, 250$ and 500 for 5000 runs.

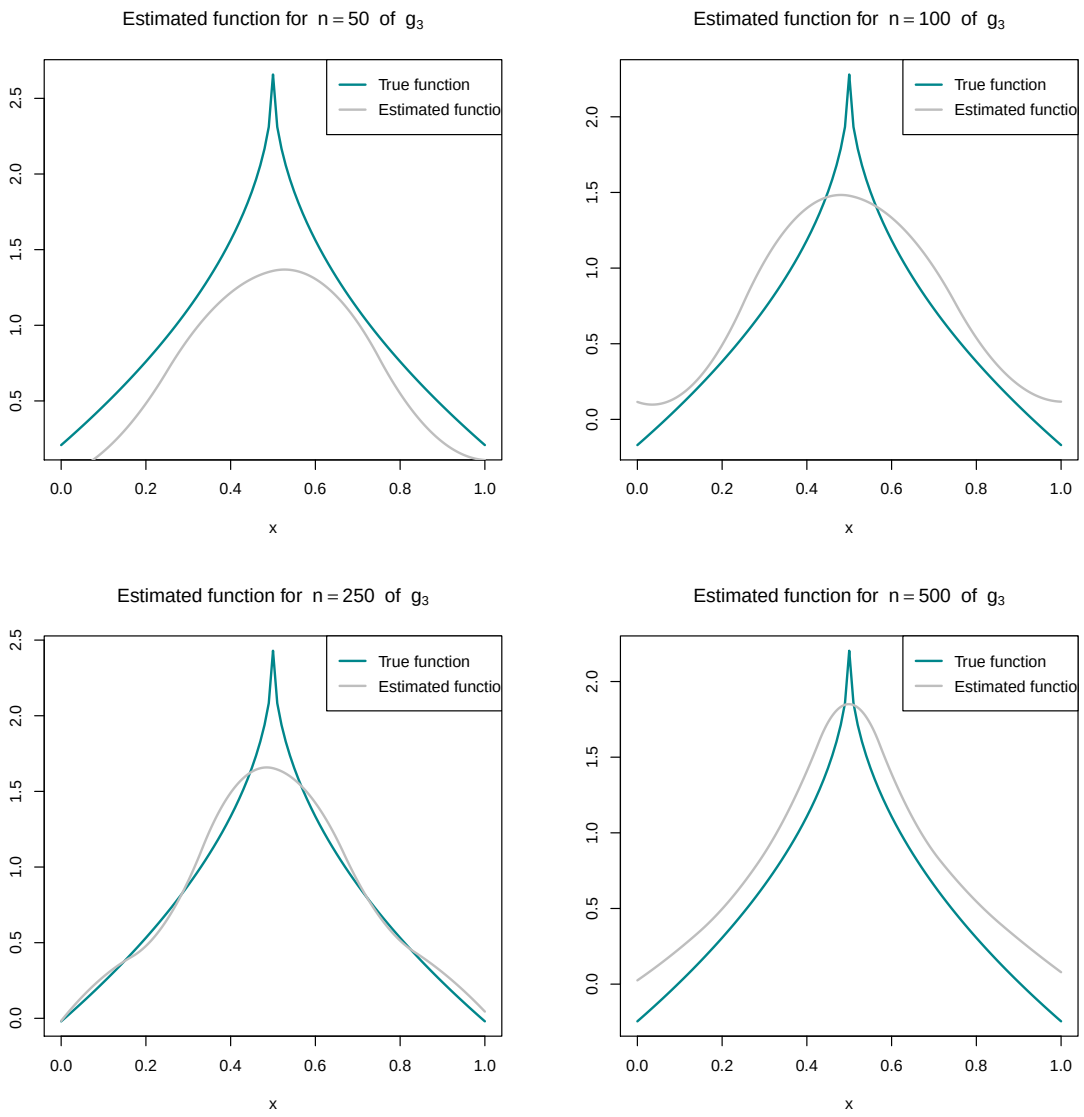

Figure 4.29: The estimated function for $n = 50, 100, 250$ and 500 samples of function g_3.

For this function, the maximiser is well estimated. However, the thing that is notable in Figure 4.28 is that for $n = 100$ and $n = 250$ the there are two peaks in the estimated maximiser just below and just above the true maximiser. This can be explained by the fact that the it is a very steep function and the peak is not smooth. The function is estimated as a smooth function, which means the maximum is a little off. This can be seen in Figure 4.29. In Table 4.9 it can be seen that both the mean absolute error of the estimated maximiser and maximum gets smaller for larger n. However, the mean error for the maximum is large in comparison with the two previous functions.

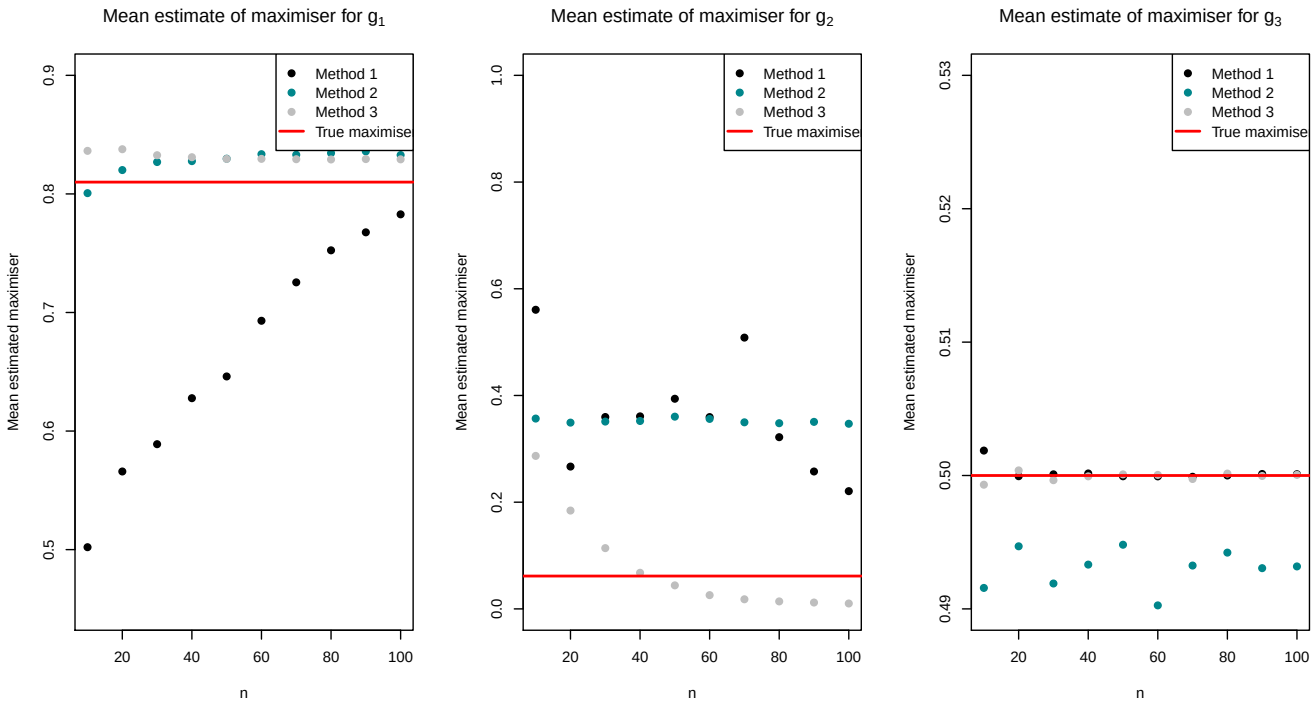

Figure 4.30: The mean estimated maximiser for 5000 runs of each method of function g_1 (left), g_2 (middle) and g_3 (right). The true maximiser is indicated by the red line.

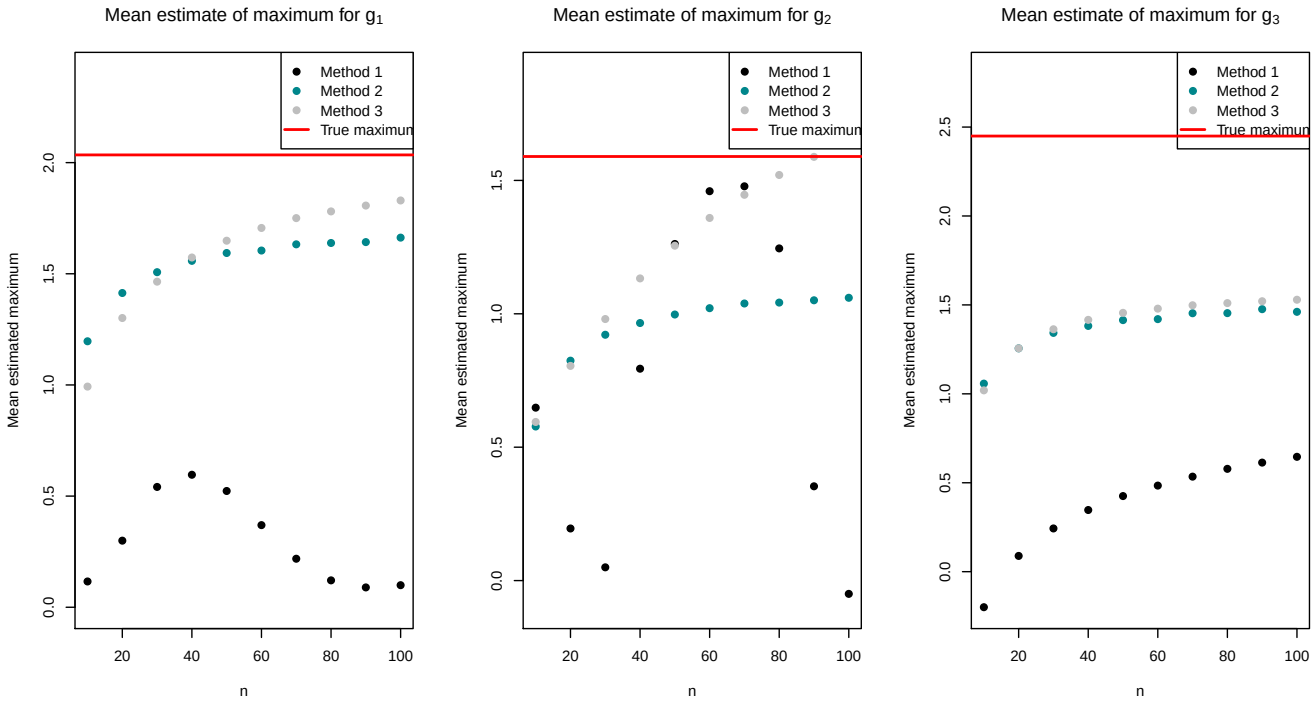

Figure 4.31: The mean estimated maximum for 5000 runs of each method of function g_1 (left), g_2 (middle) and g_3 (right). The true maximiser is indicated by the red line.

In Figure 4.30 it can be seen that for function g_1 the estimation of the maximiser for the Kiefer-Wolfowitz algorithm is very off for small n. It slowly goes to the true maximiser for larger n. The mean maximiser of the recursive and non-recursive Bayesian methods meet at $n = 40$. Thus, for $n < 40$ recursive Bayesian method is more accurate than the non-recursive method. For function g_2 the Kiefer Wolfowitz does not consistently get better results for larger n. The mean estimated maximiser for the recursive Bayesian method stays at around 0.4 for all n. The non-recursive method does get better for larger n. For function g_3 both the Kiefer-

Wolfowitz algorithm and the non-recurisive Bayesian method almost exactly estimate the maximiser. However, the recursive Bayesian method does stay behind.
In Figure 4.31 it can be seen that for function g_1 that again for $n = 40$ the mean estimated maximum coincide for the recursive and non-recursive Bayesian methods. For $n < 40$ the recursive method estimates the maximum slightly better. The Kiefer-Wolfowitz algorithm has the worst estimates. For function g_2 it can again be seen that the recursive Bayesian method tends to the true maximum faster than the non-recursive method. And the Kiefer-Wolfowitz algorithm estimates are not structured. And finally, for function g_3 it can be seen that for all three methods the estimates get better for larger n. However, for $n = 10$ the recursive Bayesian method is estimates the maximum better than the non-recursive one. This changes quickly as for $n > 20$ the non-recursive method estimates the maximum slightly better.

Chapter 5

Conclusion and discussion

5.1 Conclusion

The Kiefer-Wolfowitz algorithm and non-recursive Bayesian method are easiest to implement as opposed to the recursive Bayesian method. However, the nature of the functions affect the results for these methods. For the first function, there are no notable things that happened. The function has one clear global maximiser that got estimated well. The third method did best for both the maximiser and maximum. The Kiefer-Wolfowitz algorithm performed the worst. Moreover, the Kiefer-Wolfowitz is the worst option to estimate the maximum. The reason for this is the choice of the manner in which the maximum is estimated. This is very inaccurate. But this choice was made, because those were estimated of the maximum that were available. The second function has 2 maxima that are very close. The recursive Bayesian method was the best at estimating the maximiser. However, the non-recursive method did a better job estimating the maximum. And the Kiefer-Wolfowitz performed badly in estimating both the maximiser and maximum. And for the third function it was the fact that it had one maximum made sure that the maximiser was estimated well. But because of the nature of the function (very sharp at the maximum, not differentiable) it is not able to estimate the maximum accurately for method 3. The maximum is underestimated. Also, the choice between the recursive Bayesian method the non-recursive one depends on the amount of samples that are available. For approximately $n < 40$ the recursive Bayesian method is more accurate in estimating the maximiser for the first function. And for the for the third function, the non-recursive method is more accurate in estimating the maximum for $n > 20$.

Recommendation

Which method to use is highly depends on what is already known about the function and what the objective is. If it is a convex function that is perturbed with noise and only the maximiser is needed, then the Kiefer-Wolfowitz algorithm is most appropriate. The reason for this is its simplicity and ability to be accurate for functions with no local maxima. If the maximum is also needed, then the Kiefer-Wolfowitz algorithm is least appropriate, since the approximation of the maximum is most off. If n is large enough and the function is smooth everywhere, then it would be appropriate to use the third method. However, for functions that are not smooth at the maximum, it might be wiser to use the second method which recursively determines the maximum.

5.2 Discussion

5.2.1 Mistakes

At first, I did not understand the second method too well. I was trying too many things at once. This resulted in many mistakes. The first being the way the function was being sampled in the second method. These functions were piecewise linear functions. This resulted in the same spot being the estimated maximiser. This was solved once I understood the method better. Another mistake on the same method was the way cross validation was implemented to estimate λ. This was also fixed once I understood the concept.

5.2.2 Implementation

For the Kiefer-Wolfowitz algorithm, the one thing that went wrong is the fact that it did not converge to a maximiser between 0 and 1. The first solution that was tried was to redirect the algorithm to 0 if it was directing the estimated maximiser to be less than 0, and redirect the algorithm to 1 if it was directing the estimated maximiser to be larger than 1. This is a tricky approach, since the algorithm would then sometimes get stuck on

the boundaries. The approach used in the report gave better results as it would essentially start the algorithm over. Another thing that would be interesting to improve is the estimation of the maximum. This is something that lacked in all three functions analyzed.
There is a possibility that there is still a mistake in the implementation in R of method 2. Since it sometimes gives plots as depicted in Figure 5.1.

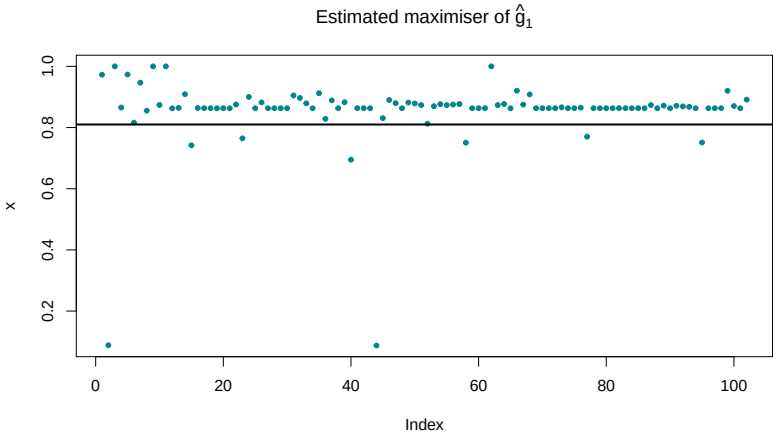

Figure 5.1: An example of potential strange behaviour by the method implemented.

Figure 5.1 shows the estimated maximiser in each iteration. It can be seen that there is a relatively large gap between each sample and the true maximiser. It is also notable that after few iterations it already starts to converge to 0.89.
Efficiency is something that can be improved for the function written for the second method. There are function in R, such as `optim`, that optimize any function that has a bounded range. However, I was not able to use this function. I suspect that the way I defined the function to be optimized is the culprit. However, this meant that I had to either redefine my function or use a less efficient way of obtaining the maximum. I tried the former, however with no result. This happened with two functions. Speeding this function up will not affect the results significantly.

5.2.3 Suggestions

The second method is based on the fact that there is conjugacy with the chosen prior and given likelihood. It might be interesting to see what happens if there is no conjugacy. In this case simulating methods are very useful. Examples of algorithms that can approximate the posterior are the Metropolis-Hastings or Gibbs algorithm [1]. It might be interesting to see how this effects the results.
In the second method, the way the location for the next sample was chosen was to just randomly get a function from the normal distribution. The global maximum of this function was located, and this would be the location of the next sample. There exist acquisition functions that determine the location of the next sample which balance exploration and exploitation [2]. It would be interesting to explore different ways of determining the location of the next sample.
For the Kiefer-Wolfowitz algorithm, as discussed in section 5.2.2, the problem of the algorithm going beyond the bound of $[0, 1]$ has been solved by essentially starting the algorithm over. This could have also been solved by setting the next sample to 0 or 1, if it would have been smaller than 0 or bigger than 1, respectively. Making the step sizes smaller might also help.
The problem for both recursive methods could also be extended to with a stopping criterion. Currently, the functions work with a fixed amount of iterations. When is the estimated maximiser close enough? For the Kiefer-Wolfowitz algorithm, this is a problem that is already studied. For example, D. L. Burkholder [3], Y. Chow and H. Robbins [5] have done this research.

Bibliography

[1] Jim Albert, editor. *Markov Chain Monte Carlo Methods*, pages 101–135. Springer New York, New York, NY, 2007.

[2] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR*, abs/1012.2599, 2010.

[3] D. L. Burkholder. On a class of stochastic approximation processes. *The Annals of Mathematical Statistics*, 27(4):1044–1059, 1956.

[4] Roman Garnett, Michael Osborne, and Stephen Roberts. Bayesian optimization for sensor set selection. pages 209–219, 01 2010.

[5] Leon J. Gleser. On the asymptotic theory of fixed-size sequential confidence bounds for linear regression parameters. *The Annals of Mathematical Statistics*, 36(2):463–467, 1965.

[6] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[7] Daniel Lizotte, Tao Wang, Michael Bowling, and Dale Schuurmans. Automatic gait optimization with gaussian process regression. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 944–949, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[8] Herbert Robbins and Sutton Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407, 09 1951.

[9] J. Stoer and R. Bulirsch. *Finding Zeros and Minimum Points by Iterative Methods*, pages 289–363. Springer New York, New York, NY, 2002.

[10] Jon Wakefield. *Bayesian Inference*, pages 85–151. Springer New York, New York, NY, 2013.

[11] Wasserman. Introduction. In *All of Nonparametric Statistics*, pages 1–11. Springer New York, New York, NY, 2006.

[12] Wasserman. Nonparametric regression. In *All of Nonparametric Statistics*, pages 61–123. Springer New York, New York, NY, 2006.

[13] Wasserman. Smoothing: General concepts. In *All of Nonparametric Statistics*, pages 43–60. Springer New York, New York, NY, 2006.