Maximum Likelihood Estimation Using Bayesian Monte Carlo Methods

Master's Thesis

LUND UNIVERSITY

Faculty of Science

Danial Ali Akbari

Supervisor: Umberto Picchini

Spring 2015

Abstract

The objective of this thesis is to give a general account of the MCMC estimation approach dubbed *data cloning*, specifically performing maximum likelihood estimation via Bayesian Monte Carlo methods. An account of the procedure will be given, and it will applied to four different maximum likelihood estimation problems: simple linear regression, multiple linear regression, a stochastic dynamical model (Gompertz), and a state space model. In each case, different aspects of the method will be performed, and a comparison with the true or a approximative measure of the MLE will be done. In the final example, a comparison with the bootstrap particle filter is conducted. The data cloning approach was found to have several advantages over the SMC methods, some of these are simple implementation, fewer numerical issues and less complicated choice of proposal function. Most importantly, it avoids numerical optimization of a function. Other benefits of the data cloning procedure is that the convergence of the estimates to the true MLE as the number of clones increases, is invariant to the choice of the prior distribution. Furthermore, the approximative normality of the estimates, provides a convenient way of producing confidence intervals. The data cloning method is also accompanied by several diagnostic tools which are mentioned in the study.

Key Words: *Data cloning*, *maximum likelihood*, *Bayesian estimation*, *bootstrap particle filter*.

Acknowledgments

Writing a thesis seems always like an impossibility at the beginning. Often, however, this feeling fades as one receives support from those around. This thesis, like any other, would not have been possible without the support of many others. I would, however, especially thank my supervisor, Umberto Picchini who always countered my frustration with warm support, positive attitude and the provision of relevant feedback and knowledge. For that, I am eternally grateful to him and hope to be able to live up to that example.

List of Abbreviations

CIC	Chain Increasing Clones
ESS	Effective Sample Size
MCMC	Markov Chain Monte Carlo
MIC	Mean Increasing Clones
MLE	Maximum Likelihood Estimate
QQ	Quantile Quantile
SIS	Sequential Importance Sampling
SISR	Sequential Importance Sampling with Resampling
SMC	Sequential Monte Carlo

Contents

Chapter 1

Introduction

The objective of this thesis is to give a general account of the estimation approach dubbed *data cloning*, specifically performing maximum likelihood estimation via Bayesian Monte Carlo methods. In this section, firstly a brief account of MCMC methods, specifically the Metropolis-Hastings algorithm will be presented. Then, the data cloning method will be introduced and some aspects of it discussed.

1.1 The Metropolis-Hastings Algorithm

Assume that $\boldsymbol{\varphi} = (\varphi_1, ..., \varphi_d)$ is a stochastic variable with dimension d and probability density function $f(\boldsymbol{\varphi})$. Let also $\boldsymbol{\varphi}_i, i = 1, ..., n$ be a finite i.i.d. sample from f of size n, where $\boldsymbol{\varphi}_i = (\varphi_{1,i}, ..., \varphi_{d,i})$. If $h(\boldsymbol{\varphi})$ is a properly defined measurable function, then by the Law of Large Numbers it follows that:

$$\frac{1}{n}\sum_{i=1}^{n} h(\boldsymbol{\varphi}_i) \overset{\substack{a.s. \\ n \to \infty}}{\longrightarrow} \mathbb{E}_f(h(\boldsymbol{\varphi}))$$

Hence, if the calculation of $\mathbb{E}_f(h(\boldsymbol{\varphi}))$ is analytically cumbersome, one could simulate $\boldsymbol{\varphi}_i \sim f(\boldsymbol{\varphi}), i = 1, ..., n$, calculate $h(\boldsymbol{\varphi}_i)$ and evaluate the mean of the sample. This mean is an estimate of the expected value above, provided that n is chosen large enough.

Often, however, the target distribution f is either unknown or difficult to sample from. MCMC schemes provide an ailment for this problem, by introducing the concept of the *instrumental distribution* or *proposal function*. The instrumental distribution, say $q(\boldsymbol{\varphi})$, proposes values which are then accepted based on some probability that reflects how likely it is that they are from the target distribution f. The Metropolis-Hastings algorithm is a specific and widely used version of such MCMC schemes.

In most of the Metropolis-Hastings algorithms, the proposed value at each iterative step of the MCMC chain, depends on the last accepted value. In other words, the Metropolis-Hastings algorithm allow for so called "local updates" (Cosma and Evers, 2010, pp. 53-56). A Bayesian version of the Metropolis-Hastings algorithm is mentioned in Algorithm (1). In the Bayesian version, a prior $\pi(\boldsymbol{\varphi})$ is also introduced, which enables initialization of the MCMC Metropolis-Hastings scheme. In a non-Bayesian, setting the starting value $\boldsymbol{\varphi}_0$ is assumed to be fixed or given. Hastings (1970) proves that this algorithm defines a Marokov chain that has as stationary distribution the desired target posterior f.

Algorithm 1 Metropolis-Hastings Algorithm in a Bayesian setting.

Let $q(\boldsymbol{\varphi})$ be a proposal function, $\pi(\boldsymbol{\varphi})$ is a suitable prior and $h(\boldsymbol{\varphi})$ a measurable function.

1. Initialization: Generate $\boldsymbol{\varphi}_0 \sim \pi(\boldsymbol{\varphi})$.
2. Simulation: Repeat the following steps for $i \leq n$, where n is large enough.
   2.1 Propose $\boldsymbol{\varphi}^{\#} \sim q(\boldsymbol{\varphi}|\boldsymbol{\varphi}_{i-1})$.
   2.2 Compute
   $$\alpha = \min\{1, \frac{f(\boldsymbol{\varphi}^{\#})q(\boldsymbol{\varphi}_{i-1}|\boldsymbol{\varphi}^{\#})\pi(\boldsymbol{\varphi}^{\#})}{f(\boldsymbol{\varphi}_{i-1})q(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}_{i-1})\pi(\boldsymbol{\varphi}_{i-1})}\} \tag{1.1}$$
   2.3 Draw $\omega \sim \mathbb{U}[0,1]$. If $\omega < \alpha$, set $\boldsymbol{\varphi}_i = \boldsymbol{\varphi}^{\#}$. Otherwise, set $\boldsymbol{\varphi}_i = \boldsymbol{\varphi}_{i-1}$.
3. Estimation: Calculate $\frac{1}{n}\sum_{i=1}^{n} h(\boldsymbol{\varphi}_i)$.

1.1.1 Random Walk Proposals

There is a wide range of proposal functions that are frequently utilized. One such proposal scheme, which is used in this study, is the random walk proposal scheme. In such a scheme, the proposal function q is chosen to be a conditionally symmetric distribution. In other words,

$$q(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^{*}) = q(\boldsymbol{\varphi}^{*}|\boldsymbol{\varphi}^{\#})$$

Then the acceptance rate given in (1.1) is reduced to the following:

$$\alpha = \min\{1, \frac{f(\boldsymbol{\varphi}^{\#})\pi(\boldsymbol{\varphi}^{\#})}{f(\boldsymbol{\varphi}_{i-1})\pi(\boldsymbol{\varphi}_{i-1})}\}.$$

One such proposal scheme is the *normal random walk*, which is used in this study. In such a scheme, step (2.1) in Algorithm (1), is given by:

$$\boldsymbol{\varphi}^{\#} = \boldsymbol{\varphi}_{i-1} + \boldsymbol{\varepsilon}_i, \boldsymbol{\varepsilon}_i \sim N(\mathbf{0}_d, C_i)$$

where $\boldsymbol{\varepsilon}_i$ are the d-dimensional multivariate normal distribution with zero mean and variance-covariance matrix C_i.

The variance-covariance matrix of the normal random walk proposal C_i, could be chosen as fixed. However, usually allowing the algorithm to adjust the covariance of the proposals, based on the history of the iterations, improves the acceptance rate, prompting it to approach the theoretically optimal acceptance rate of about 23% for random walk metropolis algorithms (Roberts et al., 1997, p. 113).[1] In this study, the following Metropolis update scheme was chosen for the covariance matrix of the proposals(Haario, Saksman & Tamminen, 2001, p. 225):

$$C_i = \begin{cases} C_0, & i \leq i_0, \\ s_d(cov(\boldsymbol{\varphi}_0, ..., \boldsymbol{\varphi}_{i-1}) + \epsilon I_d) & i > i_0. \end{cases} \tag{1.2}$$

[1] Provided that certain stability conditions hold.

where C_0 is the initial guess with regards to a proper covariance for the random walk proposals, i_0 is the threshold for the update to commence, I_d is the d-dimensional identity matrix, $\epsilon > 0$ a control parameter avoiding C_i become singular or near singular, and s_d is the update parameter which by Haario et al. (2001, p. 226) is suggested to be put as $s_d = \frac{2.4^2}{d}$. Higher values of i_0 indicates higher trust in the initial guess C_0. Another approach is to update could be done every n_0^{th} MCMC iteration instead of performing the update every iteration after certain threshold:

$$C_i = \begin{cases} C_0, & i \leq n_0, \\ s_d(cov(\boldsymbol{\varphi}_0, ..., \boldsymbol{\varphi}_{i-1}) + \epsilon I_d) & \mod (i, n_0) = 0. \end{cases} \tag{1.3}$$

1.2 A Brief Account of Data Cloning

The idea of the data cloning method is simple (Lele, Dennis, and Lutscher, 2007, p. 553). Let the observed data be $y = (y_1, ..., y_n)$, where n is the number of observed data. After constructing a full Bayesian model, with specified priors for the unknown parameters instead of using the likelihood of the observed data, y, one uses the likelihood function that corresponds to the vector containing K clones of the data, i.e.

$$y^{(K)} = \underbrace{(y, ..., y)}_{K \text{ times}}. \tag{1.4}$$

The clones are assumed be attained independently of each other, and K is assumed to be large enough. In other words, one performs Bayesian estimation as if one had $y^{(K)}$ as the obtained observed data set, where each observed data point, was attained independently of the others. The posterior, or the *pseudo-posterior* rather, is then calculated through the familiar use of the MCMC scheme, such as Metropolis-Hastings. The process is mentioned in Algorithm (1.2). The pseudo-posterior is then used to perform the desired estimations for which, one would originally have used the regular posterior in a Bayesian setting.

Algorithm 2 A general account of the Data Cloning Approach.

Let the observed data be $y = (y_1, ..., y_n)$, where n is the number of observed data.

1. Clone the data K times to attain $y^{(K)}$ in (1.4). Treat $y^{(K)}$ as if its elements have been attained independently of each other.

2. Target the pseudo-posterior through the use of an MCMC scheme upon the K-times cloned data. The pseudo-posterior is given by the following:

$$p(\boldsymbol{\varphi}|y^{(K)}) \propto p(y^{(K)}|\boldsymbol{\varphi})\pi(\boldsymbol{\varphi}),$$

   where $\boldsymbol{\varphi}$ are the unknown parameters for which π is a prior.

In this study, the interest lies in attaining an approximation of the MLE and its confidence. Consider the following setting. Let the observed data, Y, be assumed to

have the following hierarchical structure:

$$\begin{aligned} &Y \sim g(y|X;\boldsymbol{\theta}) \\ &X \sim f(x|\boldsymbol{\vartheta}) \\ &Z = (Z_1, ..., Z_n) \text{ where } Z \text{ is either } y, x, Y \text{ or } X \\ &\boldsymbol{\theta} = (\theta_1, ..., \theta_{d_y}) \\ &\boldsymbol{\vartheta} = (\vartheta_1, ..., \vartheta_{d_x}) \\ &\boldsymbol{\varphi} = (\boldsymbol{\theta}, \boldsymbol{\vartheta}) \end{aligned} \tag{1.5}$$

Here X denotes the input variables that are stochastic with conditional density f, $\boldsymbol{\theta}$ are the unknown parameters of the observed state's density function (with dimension d_θ), $\boldsymbol{\vartheta}$ are the unknown parameters of the input (hidden) state function (with dimension d_ϑ), n the number of observations and $\boldsymbol{\varphi}$ the total collection of unknown parameters, with dimension $d = d_x + d_y$.

In this setting, the goal is to approximate the MLE of $\boldsymbol{\varphi}$ given the observed data y, dubbed $\boldsymbol{\varphi}_{MLE}$. In the ordinary Bayesian setup, one attempts to evaluate the MLE through maximizing the likelihood function:

$$L(\boldsymbol{\varphi}; y) = p(y|\boldsymbol{\varphi}) = \int g(y|X;\boldsymbol{\theta}) f(X|\boldsymbol{\vartheta}) dX \tag{1.6}$$

The model in (1.5) may lack a stochastic input X, which reduces the structure further to the following:

$$\begin{aligned} &Y \sim g(y|\boldsymbol{\varphi}) \\ &Z = (Z_1, ..., Z_n) \text{ where } Z \text{ is either } Y \text{ or } y \\ &\boldsymbol{\varphi} = (\varphi_1, ..., \varphi_d) \end{aligned} \tag{1.7}$$

In such a case, the likelihood function is simply the conditional distribution:

$$L(\boldsymbol{\varphi}; y) = p(y|\boldsymbol{\varphi}) = g(y|\boldsymbol{\varphi}). \tag{1.8}$$

In the data-cloning method, the observed data y is substituted for the the K-times cloned observed data set $y^{(k)}$. In both settings mentioned, i.e. with hidden state in (1.5) and fixed effect in (1.7), one can prove that the likelihood function of the K-times cloned data is equal to the likelihood function of the original data to the power of K.[2] In other words,

$$L(\boldsymbol{\varphi}; y^{(K)}) = \left(L(\boldsymbol{\varphi}; y)\right)^K. \tag{1.9}$$

Furthermore, Lele (2007, et al., pp. 562-3) that the pseudo-posterior $\pi(\boldsymbol{\varphi}|Y^{(K)})$ is degenerate at the true MLE. The proof of this property is mentioned in the sections below.

Model with Fixed Effects

Consider the model in (1.7), and let $\boldsymbol{\Phi}$ be the parameter space of $\boldsymbol{\varphi}$. Furthermore, let $\pi(\varphi)$ be a prior distribution with support on the whole of the parameter space $\boldsymbol{\Phi}$. Then the posterior distribution is given by the following:

$$\pi^{(1)}(\boldsymbol{\varphi}|y) = \frac{g(y|\boldsymbol{\varphi})\pi(\boldsymbol{\varphi})}{\int g(y|\boldsymbol{\varphi})\pi(\boldsymbol{\varphi}) d\boldsymbol{\varphi}}$$

[2]The proof of this for the fixed effect case, is a direct result of the assumption that the copies are acquired independently of each other. The proof of this property for the model with latent stochastic states is not very cumbersome either, and is mentioned in section (4.1.1) for state space models.

Using the new posterior as prior again, one obtains the following,

$$\pi^{(2)}(\boldsymbol{\varphi}|y) = \frac{g(y|\boldsymbol{\varphi})\pi^{(1)}(\boldsymbol{\varphi})}{\int g(y|\boldsymbol{\varphi})\pi^{(1)}(\boldsymbol{\varphi})d\boldsymbol{\varphi}} = \frac{[g(y|\boldsymbol{\varphi})]^2\pi(\boldsymbol{\varphi})}{\int [g(y|\boldsymbol{\varphi})]^2\pi(\boldsymbol{\varphi})d\boldsymbol{\varphi}}$$

Then, by induction, one arrives at the following result,

$$\pi^{(K)}(\boldsymbol{\varphi}|y) = \frac{[g(y|\boldsymbol{\varphi})]^K\pi(\boldsymbol{\varphi})}{\int [g(y|\boldsymbol{\varphi})]^K\pi(\boldsymbol{\varphi})d\boldsymbol{\varphi}} \tag{1.10}$$

Moreover, it is clear that $\pi^{(K)}(\boldsymbol{\varphi}|y)$ is equal to the pseudo-posterior distribution resulting from employing the cloned data $y^{(K)}$ within the likelihood function. In other words,

$$\pi(\boldsymbol{\varphi}|y^{(K)}) = \pi^{(K)}(\boldsymbol{\varphi}|y)$$

Now let $\boldsymbol{\varphi}_{MLE}$ be the MLE of the unknown parameter given the data y. Then, by definition of the MLE, it follows that

$$g(y|\boldsymbol{\varphi}_{MLE}) > g(y|\boldsymbol{\varphi}). \tag{1.11}$$

Since the prior π has the entire parameter space as its support, then by (1.10) and (1.11) it follows that,

$$\lim_{K\to\infty} \frac{\pi^{(K)}(\boldsymbol{\varphi}|y)}{\pi^{(K)}(\boldsymbol{\varphi}_{MLE}|y)} = \frac{[g(y|\boldsymbol{\varphi})]^K\pi(\boldsymbol{\varphi})}{[g(y|\boldsymbol{\varphi}_{MLE})]^K\pi(\boldsymbol{\varphi}_{MLE})} = \kappa$$

$$\kappa = \begin{cases} 0, & \boldsymbol{\varphi} \neq \boldsymbol{\varphi}_{MLE} \\ 1, & \boldsymbol{\varphi} = \boldsymbol{\varphi}_{MLE} \end{cases}$$

Hence, as the number of clones (K) increases, the pseudo-posterior converges to a degenerate distribution with its probability density concentrated at the MLE.

Model with Latent Variables

The proof of the asymptotic degeneracy of the pseudo-posterior in the case where the model structure contains latent states, is very similar to the proof in the fixed effect case. Consider the model structure in (1.5). As before let $\pi(\varphi)$ be a prior distribution with support on the whole of the parameter space $\boldsymbol{\Phi}$. Then the posterior is given by,

$$\pi^{(1)}(\boldsymbol{\varphi}|y) = \frac{[\int g(y|X,\boldsymbol{\theta})f(X|\boldsymbol{\vartheta})dX]\pi(\boldsymbol{\varphi})}{C^{(1)}(y)},$$

$$C^{(1)}(y) = \int\int g(y|X,\boldsymbol{\theta})f(X|\boldsymbol{\vartheta})\pi(\boldsymbol{\varphi})dXd\boldsymbol{\varphi}.$$

Observe that the expression in the brackets in the numerator is the likelihood function:

$$L(\boldsymbol{\varphi};y) = \int g(y|X,\boldsymbol{\theta})f(X|\boldsymbol{\vartheta})dX$$

Again, utilizing this posterior as a prior, one obtains the following posterior distribution,

$$\pi^{(2)}(\boldsymbol{\varphi}|y) = \frac{L(\boldsymbol{\varphi};y)\pi^{(1)}(\boldsymbol{\varphi})}{C^{(2)}(y)} = \frac{[L(\boldsymbol{\varphi};y)]^2\pi(\boldsymbol{\varphi})}{C^{(2)}(y)},$$

where $C^{(2)}(y)$ is given by,

$$\begin{aligned} C^{(2)}(y) &= \int\int g(y|X,\boldsymbol{\theta}) f(X|\boldsymbol{\vartheta}) \pi^{(1)}(\boldsymbol{\varphi}) dX d\boldsymbol{\varphi} \\ &= \frac{\int [\int g(y|X,\boldsymbol{\theta}) f(X|\boldsymbol{\vartheta}) dX]^2 \pi(\boldsymbol{\varphi}) d\boldsymbol{\varphi}}{C^{(1)}(y)}. \end{aligned}$$

Hence, by induction the following is obtained:

$$\pi^{(K)}(\boldsymbol{\varphi}|y) = \frac{[L(\boldsymbol{\varphi};y)]^K \pi(\boldsymbol{\varphi})}{C^{(K)}(y)}, \tag{1.12}$$

$$C^{(K)}(y) = \frac{\int [\int g(y|X,\boldsymbol{\theta}) f(X|\boldsymbol{\vartheta}) dX]^K \pi(\boldsymbol{\varphi}) d\boldsymbol{\varphi}}{C^{(K-1)}(y)}. \tag{1.13}$$

As before, it is clear that $\pi^{(K)}(\boldsymbol{\varphi}|y)$ is equal to the pseudo-posterior distribution resulting from employing the cloned data $y^{(K)}$ within the likelihood function. In other words,

$$\pi(\boldsymbol{\varphi}|y^{(K)}) = \pi^{(K)}(\boldsymbol{\varphi}|y)$$

Similarly, assume that $\boldsymbol{\varphi}_{MLE}$ is the MLE of the unknown parameter given the data y. Then, by definition of the MLE, it follows that

$$L(\boldsymbol{\varphi}_{MLE};y) > L(\boldsymbol{\varphi};y). \tag{1.14}$$

Since the prior π has the entire parameter space as its support, then by (1.12) and (1.14) it follows that,

$$\begin{aligned} \lim_{K\to\infty} \frac{\pi^{(K)}(\boldsymbol{\varphi}|y)}{\pi^{(K)}(\boldsymbol{\varphi}_{MLE}|y)} &= \frac{[L(\boldsymbol{\varphi};y)]^K \pi(\boldsymbol{\varphi})}{[L(\boldsymbol{\varphi}_{MLE};y)]^K \pi(\boldsymbol{\varphi}_{MLE})} = \kappa \\ \kappa &= \begin{cases} 0, & \boldsymbol{\varphi} \neq \boldsymbol{\varphi}_{MLE} \\ 1, & \boldsymbol{\varphi} = \boldsymbol{\varphi}_{MLE} \end{cases} \end{aligned}$$

Hence, as the number of clones (K) increases, the pseudo-posterior converges to a degenerate distribution with its probability density concentrated at the MLE.

1.2.1 Asymptotic Properties

We have already mentioned one asymptotic property of the pseudo-posterior retrieved through data cloning, namely that it is asymptotically degenerate at the MLE. This result also indicates that the convergence to the true MLE is invariant to the prior.

Lele et. al. (2010) further proves the asymptotic normality for each fixed K large enough and given certain regularity conditions:

$$\bar{\boldsymbol{\varphi}}_{(K)} \underbrace{\sim}_{\text{approximately}} N(\boldsymbol{\varphi}_{MLE}, \frac{1}{K} I^{-1}(\boldsymbol{\varphi}_{MLE})), \tag{1.15}$$

where $\bar{\boldsymbol{\varphi}}_{(K)}$ is the sample mean of the draws given the K-times cloned data. Hence, the mean of the accepted parameters targeting $\boldsymbol{\varphi}$, is an approximation of the true MLE and, likewise, the variance-covariance matrix of those simulated parameters, is an approximation of the variance-covariance matrix of the true MLE divided by the number of clones K.

1.2.2 Diagnostics

Lele et al. (2010), suggests two asymptotic measures for correct convergence of the estimate of the MLE via the data cloning procedure: the ω- and the $\tilde{r}^2$-statistics mentioned in equations (1.17) and (1.18). They are derived through the following reasoning (Withanage, 2013, p. 43). As the number of clones K increased, the Mahalanobis distance between realizations of the vector of the parameters being estimated ($\boldsymbol{\varphi}$) and its expected value has a χ^2-distribution with d degrees of freedom:

$$(\boldsymbol{\varphi} - \mathbb{E}(\boldsymbol{\varphi}))^T \text{cov}^{-1}(\boldsymbol{\varphi}|y^{(K)})(\boldsymbol{\varphi} - \mathbb{E}(\boldsymbol{\varphi})) \sim \chi^2_d \tag{1.16}$$

where $y^{(K)}$ is the K-times cloned observation and d the dimension of $\boldsymbol{\varphi}$. Hence, a χ^2-quantile-quantile plot could be used to determine whether the estimates are converging correctly or not. This would be done by replacing the expected value $\mathbb{E}(\boldsymbol{\varphi})$ by its estimate given the simulations, that is $\bar{\boldsymbol{\varphi}} = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{\varphi}_i$ (M is the number of MCMC simulations), which also is the approximation of the MLE through the data cloning method.

A modified approach is, however, to create other statistics based on this relationship between the actual quantiles (those attained through the simulations) and the theoretical ones (corresponding to χ^2_d). The following ones were suggested by Lele et al. (2010):

$$\omega = \frac{1}{M}\sum_{i=1}^{M}(\hat{Q}_{(i)} - Q_i)^2 \tag{1.17}$$

$$\tilde{r}^2 = 1 - \text{corr}^2(\hat{\boldsymbol{Q}}, \boldsymbol{Q}) \tag{1.18}$$

where $\hat{Q}_{(1)} \leq ... \leq \hat{Q}_{(M)}$ are the ordered estimated quantiles

$$\hat{Q}_i = (\boldsymbol{\varphi}_i - \bar{\boldsymbol{\varphi}})^T \text{cov}^{-1}(\boldsymbol{\varphi}|y^{(K)})(\boldsymbol{\varphi}_i - \bar{\boldsymbol{\varphi}}), \boldsymbol{\varphi} = (\boldsymbol{\varphi}_1, ..., \boldsymbol{\varphi}_M),$$

$\hat{\boldsymbol{Q}} = (\hat{Q}_{(1)}, ..., \hat{Q}_{(M)})$, $Q_i = \chi^2_{d,(i-0.5)/M}$ the theoretical quantiles (the $(i-0.5)/M^{th}$ quantiles) from the corresponding χ^2_d-distribution, and $\boldsymbol{Q} = (Q_1, ..., Q_M)$. When the data cloning procedure is done properly, both ω and $\tilde{r}^2$ should be close to zero, or better yet, converge to zero with increasing number of clones K.

Furthermore, the convergence of the standardized largest eigenvalue to zero is also suggested as a measure of estimability. The standardized largest eigenvalue of the covariance matrix of the simulations with number of clones K - i.e. λ^S_K - is the largest eigenvalue of the covariance matrix of the simulations with K number of clones (λ_K) divided by the the largest eigenvalue of the covariance matrix of the simulations with one clone (λ_1), that is $\lambda^S_K = \frac{\lambda_K}{\lambda_1}$. As proven by Lele et al. (2010), if $\boldsymbol{\varphi}$ is estimable, this value (λ^S_K) should decrease to zero at the rate of $1/K$. Hence, a comparison of the λ^S_K and $1/K$, can inform the researcher of the estimability of the parameter in question.

Furthermore, the assessment of normality of the pseudo-posterior is carried out by the use of quantile-quantile plots of the chain as suggested by Jacquier et al. (2007) for maximum likelihood estimations via MCMC simulations.

Chapter 2

Static Models

2.1 Applying the Theory

In the case of static models, there is only one level of hierarchy in the statistical model, i.e. that of the observational process $\boldsymbol{Y} = (Y_1, ..., Y_n)$, where n is the number of the observations. This is due to the fact that the quantities, $\boldsymbol{x} = (x_1, ..., x_n)$, affecting the observations are assumed to be fixed and known. Hence, the model could be summarized in the following way:

$$\boldsymbol{Y} \sim f(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{\varphi}) \tag{2.1}$$

where f is a joint probability density function and $\boldsymbol{\varphi} = (\varphi_1, ..., \varphi_d)$, is a vector of unknown parameters affecting the observations.

A Data-cloning MCMC algorithm for static models is mentioned in algorithm (3). Observe that all the calculations could be done using logarithmic transform. In fact, such transformation will avoid the problem of the quotas being of very different magnitudes and the subsequent computational problems that arises. Hence, operating with such transformations is recommended.

Furthermore, using a symmetric random walk proposal scheme, for the proposal function $v(\boldsymbol{\varphi})$, would entail that $v(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^{*}) = v(\boldsymbol{\varphi}^{*}|\boldsymbol{\varphi}^{\#})$. Hence, for such a scheme, acceptance probability α is reduced to $\alpha = \min(1, \frac{q^{\#}}{q^{*}})$. This is the scheme used in this section.

Two different methods of increasing the number of clones K was experimented with. In one method, one increases the number of clones every T number of algorithmic iterations.[1] Hence, when moving from one block of iteration with K_1 number of clones, to another with K_2 number of clones ($K_2 > K_1$), one remains in the last accepted proposal, until a new proposal is accepted. For reference purposes, this method is dubbed Chain Increasing Clones (CIC).

In another method, one increases the number of clones every T number of MCMC iterations as well. However, instead of using the last accepted proposal, one uses the estimate of the MLE of $\boldsymbol{\varphi}$, i.e. the mean of the states for the previous number of clones K (excluding a certain burn-in period if necessary). Let's dub this method

[1] One should of course correct for the increase in power along the iterations, so as to avoid comparing the pseudo-posterior for states with different number of clones.

Mean Increasing Clones (MIC). Both methods seem to function well and have no clear advantages to one another.

Algorithm 3 A Data-cloning MCMC algorithm for static models

1. Initialization: Fix starting value $\boldsymbol{\varphi}^*$ or generate it from its prior $\pi(\boldsymbol{\varphi})$. Set numbers of copies equal to K. Set $\boldsymbol{\varphi_1} = \boldsymbol{\varphi}^*$.
2. Calculate the pseudo-posterior at this state: $q^* = (f(y; \boldsymbol{x}, \boldsymbol{\varphi}^*))^K \cdot \pi(\boldsymbol{\varphi}^*)$.
3. Propose a new state from the MCMC-algorithm's proposal function: $\boldsymbol{\varphi}^{\#} \sim v(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^*)$.
4. Calculate the pseudo-posterior at the newly proposed state ($q^{\#} = (f(y; \boldsymbol{x}, \boldsymbol{\varphi}^{\#}))^K \cdot \pi(\boldsymbol{\varphi}^{\#})$) and compare its value with the last accepted state i.e. q^*. In other words, calculate the acceptance probability $\alpha = \min(1, \frac{q^{\#} \cdot v(\boldsymbol{\varphi}^*|\boldsymbol{\varphi}^{\#})}{q^* \cdot v(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^*)})$.
5. Generate a uniform random variable $u \sim U(0,1)$. If $u < \alpha$, set $\boldsymbol{\varphi_{j+1}} = \boldsymbol{\varphi}^{\#}$, $\boldsymbol{\varphi}^* = \boldsymbol{\varphi}^{\#}$ and $q^* = q^{\#}$. Otherwise set $\boldsymbol{\varphi_{j+1}} = \boldsymbol{\varphi_j}$.
6. Repeat 3-5 for $j \leq M$ number of simulations, i.e. repeat until the chain for $\boldsymbol{\varphi}$ is deemed to have converged.

2.2 Multiple Linear Regression

In the case of multiple linear regression, the model scheme above (in (2.1)) will amount to:

$$Y_i = \beta_0 + \beta_1 x_{1,i} + ... + \beta_q x_{q,i} + \eta_i, i = 1, ..., n,$$

where η_i is the noise process, for instance, $\eta_i \sim N(0, \sigma^2)$ iid. In such a framework, $\boldsymbol{\varphi} = (\beta_0, ..., \beta_q, \sigma^2)$.

It is worth to take a look at an example and some of its practical aspects. For a model, the following was chosen:

$$Y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \eta_i, \eta_i \sim N(0, \sigma^2) \tag{2.2}$$

where x_i as 101 equidistant points between 0 and 1, i.e. $(0, 0.01, 0.02, ..., 1)$, and $(\beta_0, ..., \beta_3, \sigma^2) = (0.01, 10, -20, 30, 0.09)$. To improve the acceptance rate over time a Metropolis adaptive scheme was employed. Updating the variance-covariance matrix of the random walk proposal was done through the update mechanism in (1.3) with updates done every 100^{th} MCMC iteration. Furthermore, the priors were all uniform in the following manner: $\beta_0 \sim U(-1, 1), \sigma^2 \sim U(0, 10)$ and $\beta_q \sim U(-50, 50), q = 1, 2, 3$. A realization of the simulations through the CIC method is illustrated in Figure (2.1).[2] The simulations converge to the MLE and the deviation from MLE decreases as the number of clones is increased.

[2]The corresponding plots for the MIC-method (Figure (7.2)) are very similar and can be viewed in section (7.1.1) in the Appendix.

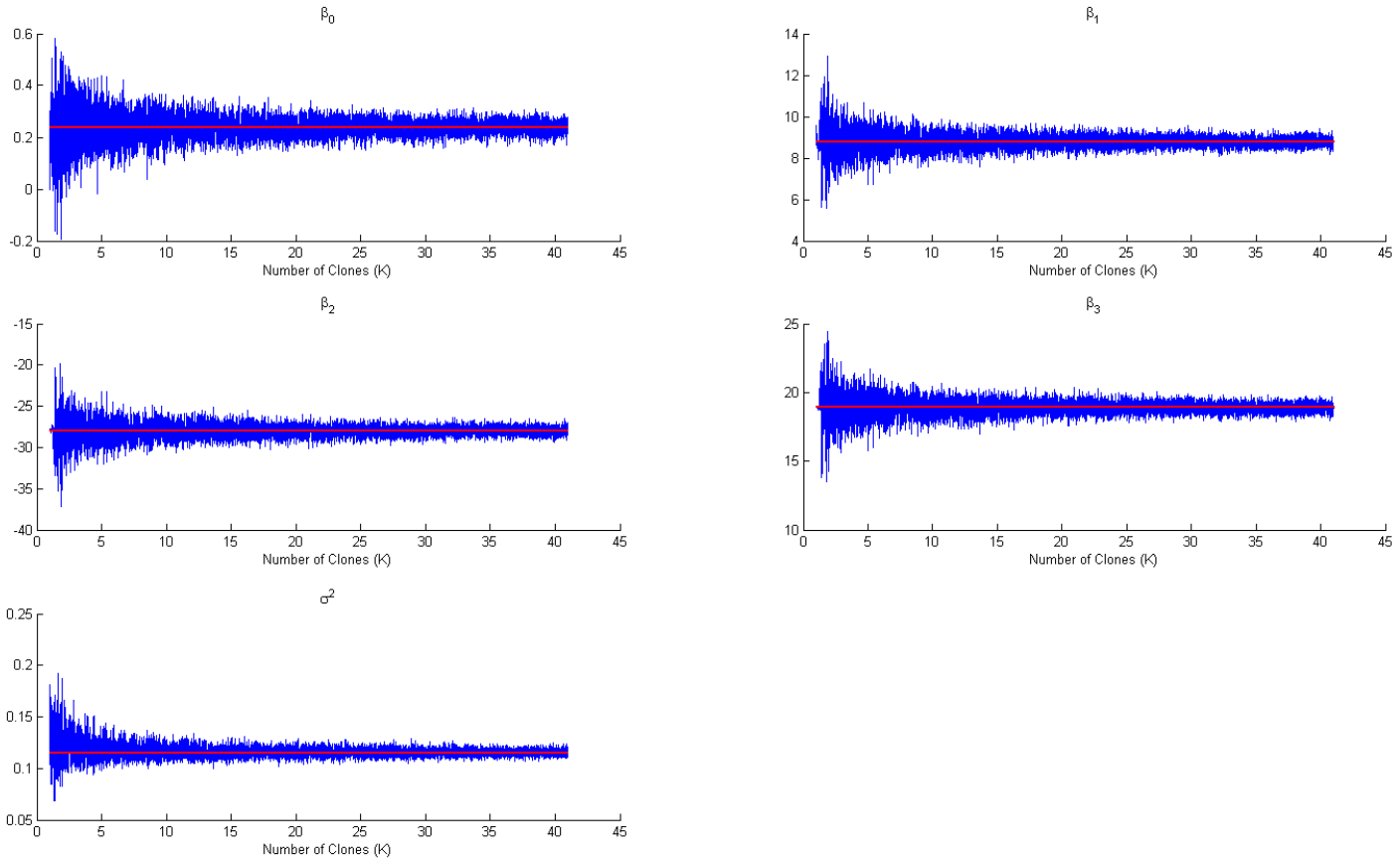

Figure 2.1: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 algorithmic iterations, applying the CIC method to the multiple linear regression model in (2.2). The red line is the true MLE.

In Figure (2.2) the increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation, starting at $K = 1$. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. This procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result.

Both methods, CIC and MIC, have an acceptance rate in the chain, which is about 28%, and hence is close to the theoretically desirable value of 23% (see Figure (2.2c)). They both perform fairly well with respect to convergence to the actual MLE and its variance covariance matrix (see Figure (2.2a,b)). However, the precision of the estimate of the variance-covariance matrix, does not seem to improve visibly after $K = 5$.

The ω- and the $\tilde{r}^2$-statistics (Figure (2.2)d,e) converge to zero in the beginning, but remain only close to zero afterwards. This is not problematic from a theoretical stand point, since, in theory, the ω- and the $\tilde{r}^2$ do not necessarily converge to zero, but should merely be close to zero.

Moreover, the convergence of the standardized largest eigenvalues of the covariance

	β_0	β_1	β_2	β_3	σ^2
True Val.	-0.01	10	-30	20	0.09
MLE	0.2373 (0.1303)	8.7999 (1.1339)	-28.0060 (2.6422)	18.9064 (1.7364)	0.1153 (0.0162)
CIC					
K=1	0.2485 (0.1290)	8.7322 (1.1051)	-27.8987 (2.5774)	18.8548 (1.6990)	0.1248 (0.0181)
K=5	0.2331 (0.1256)	8.8406 (1.0989)	-28.0956 (2.5436)	18.9612 (1.6597)	0.1165 (0.0167)
K=10	0.2385 (0.1289)	8.7749 (1.0883)	-27.9442 (2.5248)	18.8674 (1.6622)	0.1162 (0.0172)
K=40	0.2365 (0.1371)	8.8012 (1.1817)	-28.0061 (2.7319)	18.9058 (1.7860)	0.1154 (0.0166)
MIC					
K=1	0.2476 (0.1377)	8.7115 (1.1951)	-27.8172 (2.7725)	18.7869 (1.8171)	0.1256 (0.0191)
K=5	0.2335 (0.1360)	8.8196 (1.1818)	-28.0459 (2.7705)	18.9306 (1.8278)	0.1177 (0.0168)
K=10	0.2362 (0.1352)	8.8065 (1.1818)	-28.0192 (2.7569)	18.9143 (1.8099)	0.1166 (0.0164)
K=40	0.2373 (0.1331)	8.8062 (1.1277)	-28.0264 (2.6441)	18.9222 (1.7506)	0.1155 (0.0159)

Table 2.1: Summary of the estimates of the multiple regression model: the true values, the MLE and the estimates via the MIC and CIC methods of data cloning. The standard errors are mentioned in the parenthesis. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. The Metropolis update of the covariance of the random walk proposals was performed every 100^{th} MCMC iteration. The initial value (guess) of $\boldsymbol{\varphi}$ was the true values of the parameters added with some minor zero-mean normal noise.

matrices (λ_K^S) to zero is equal to the theoretical rate of $1/K$ (Figure 2.2f). The convergence to zero correctly illustrates estimability of the unknown parameters. A summary of the estimates for some different number of copies is mentioned in table (2.1) for both the CIC and the MIC method.

Similar results are found when we increase the number of clones sporadically (Figure (2.7)). The increase was done sporadically. The number of clones were $1, 2, 5, 10, 20$ and 40. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. As before, this procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result. No visible disadvantages are observed compared to increasing the number of clones continuously rather than sporadically, indicating that one should perform the estimation through the latter procedure, saving computational time.

One can further check the normality of the pseudo-posterior using QQ-plots. The QQ-plots of pseudo-posteriors of the estimates of β_0 and that of β_3, through the CIC-method, are illustrated in Figures (2.4) and (2.5) respectively.[3] The distribution of the pseudo-posteriors appear to be fairly normal.

[3]For the QQ-plots of the other parameters and those through the MIC-method see the Appendix (section (7.1.2)).

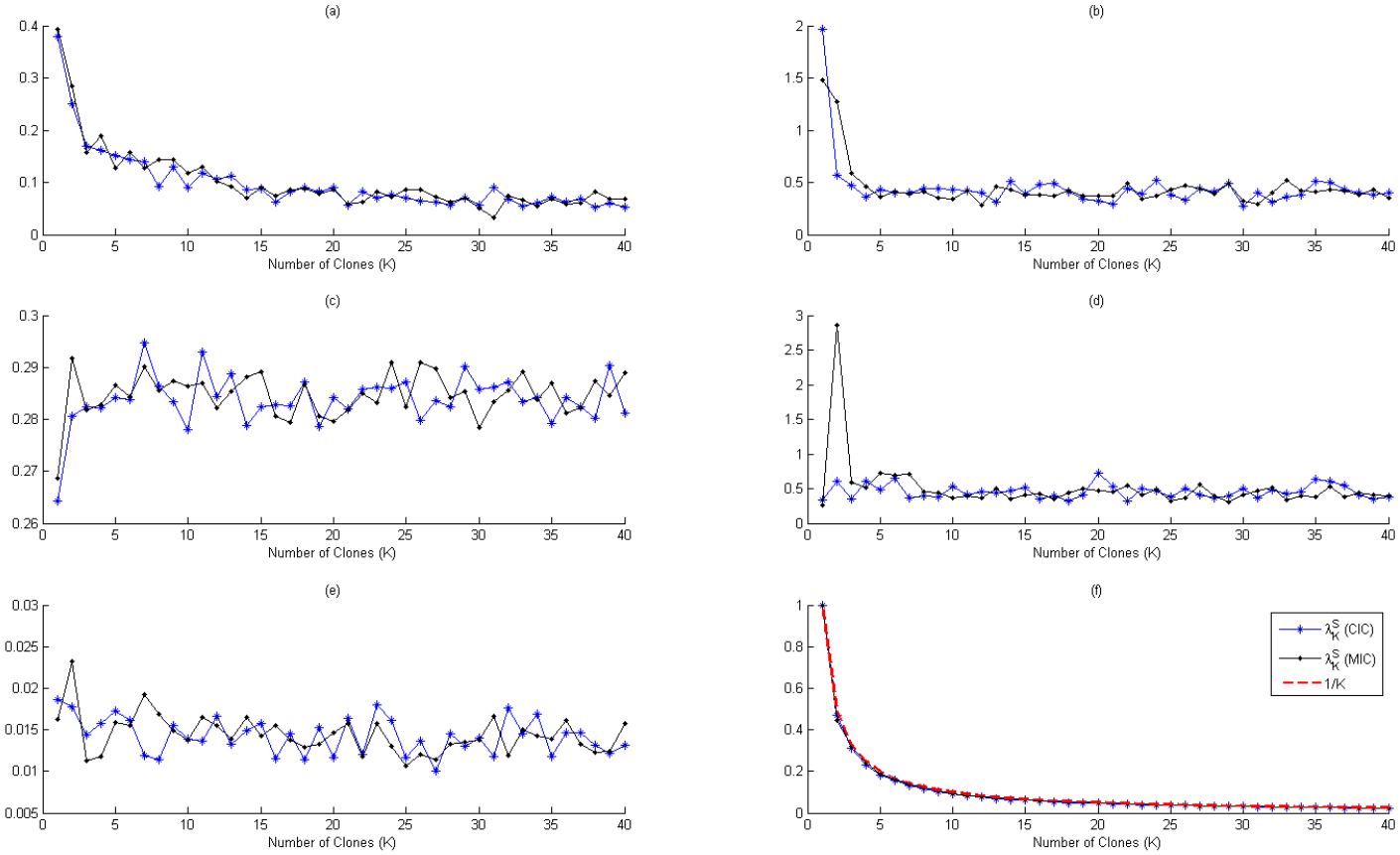

Figure 2.2: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. This procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE (the 2-norm of the difference), (b) Precision of the estimates of the variance covariance matrix of the MLE (the 2-norm of the difference) (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

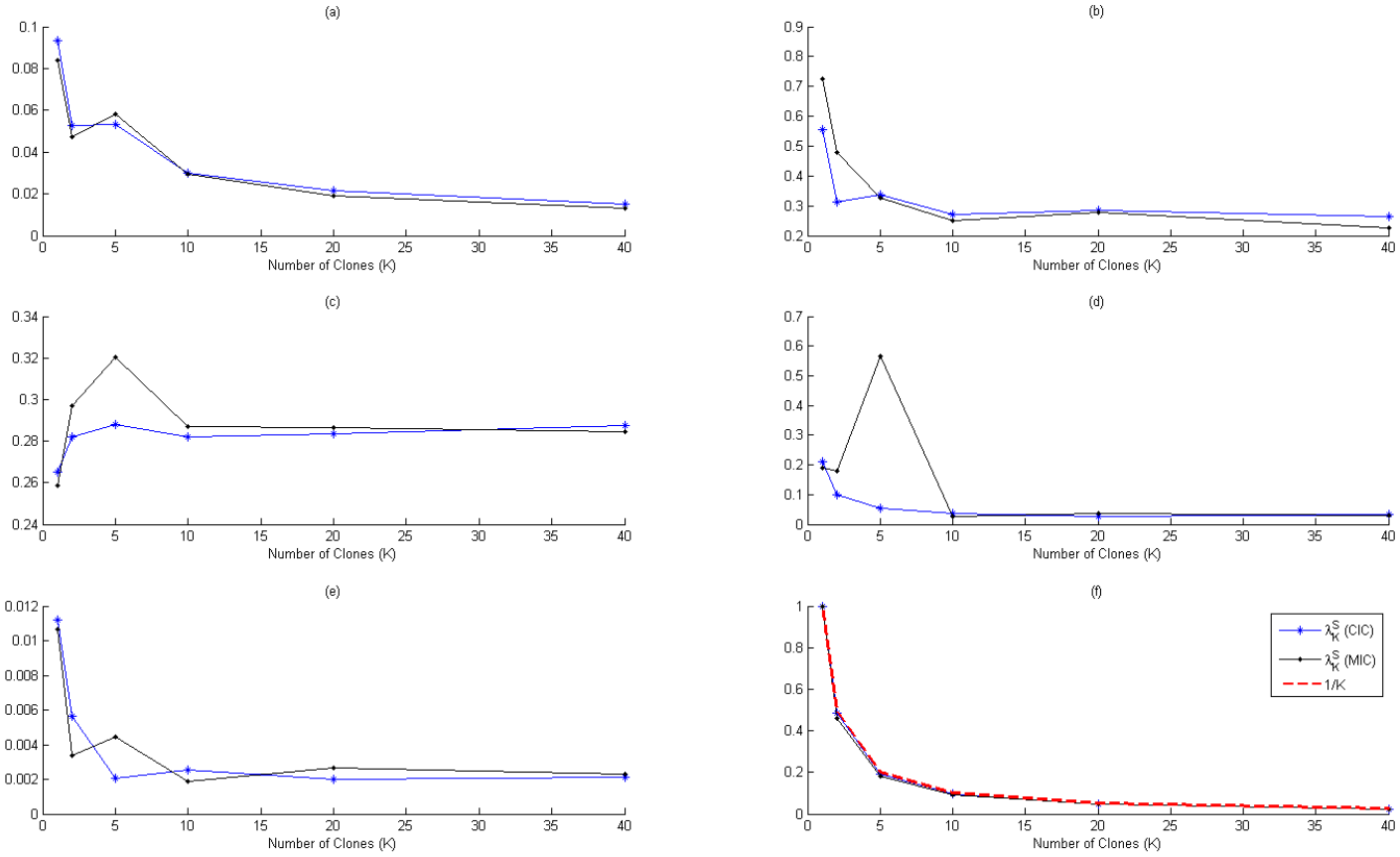

Figure 2.3: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase was done sporadically. The number of clones were $1, 2, 5, 10, 20$ and 40. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. This procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE (the 2-norm of the difference), (b) Precision of the estimates of the variance covariance matrix of the MLE (the 2-norm of the difference) (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

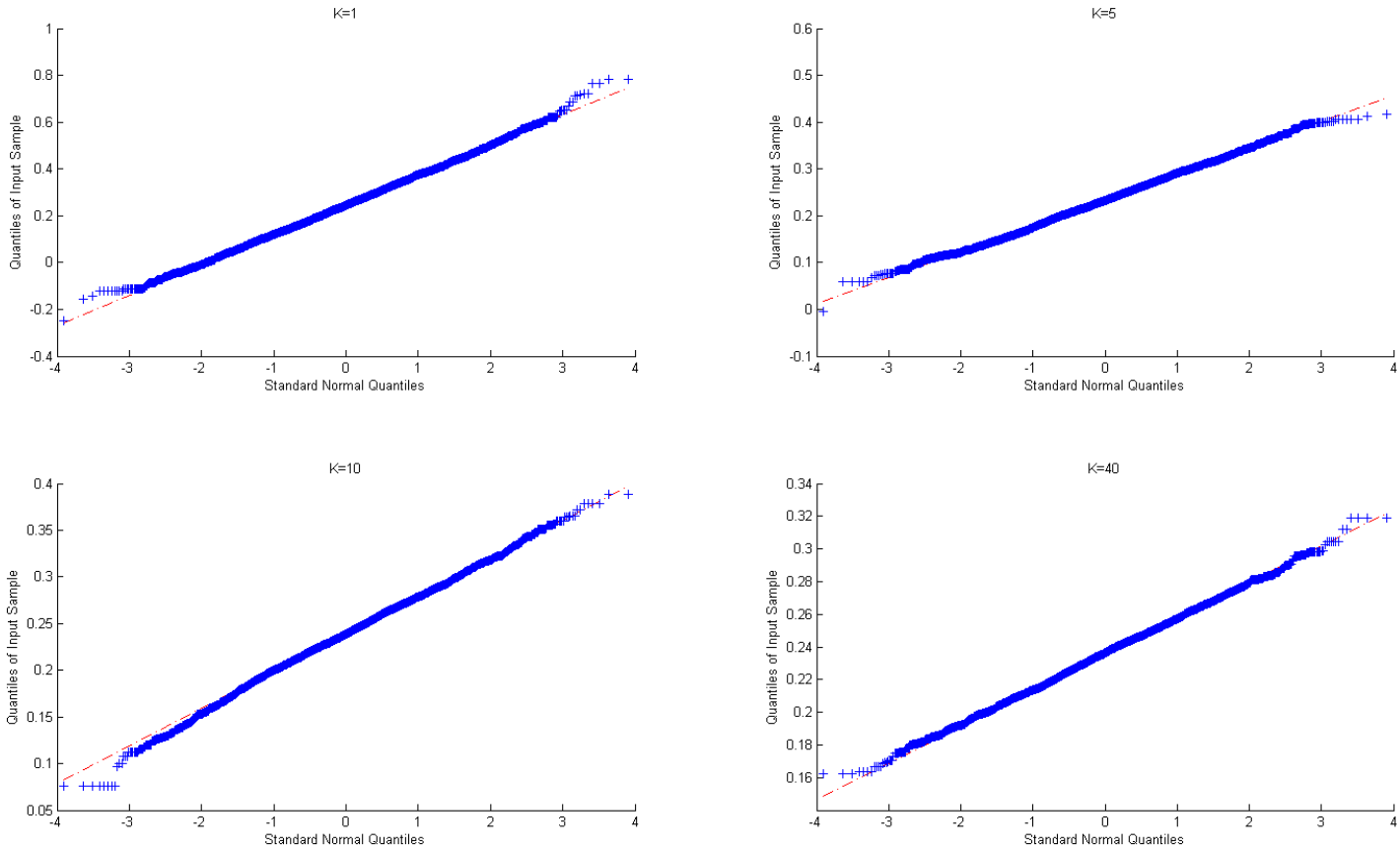

Figure 2.4: Quantile-quantile plots for the simulations of β_0 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

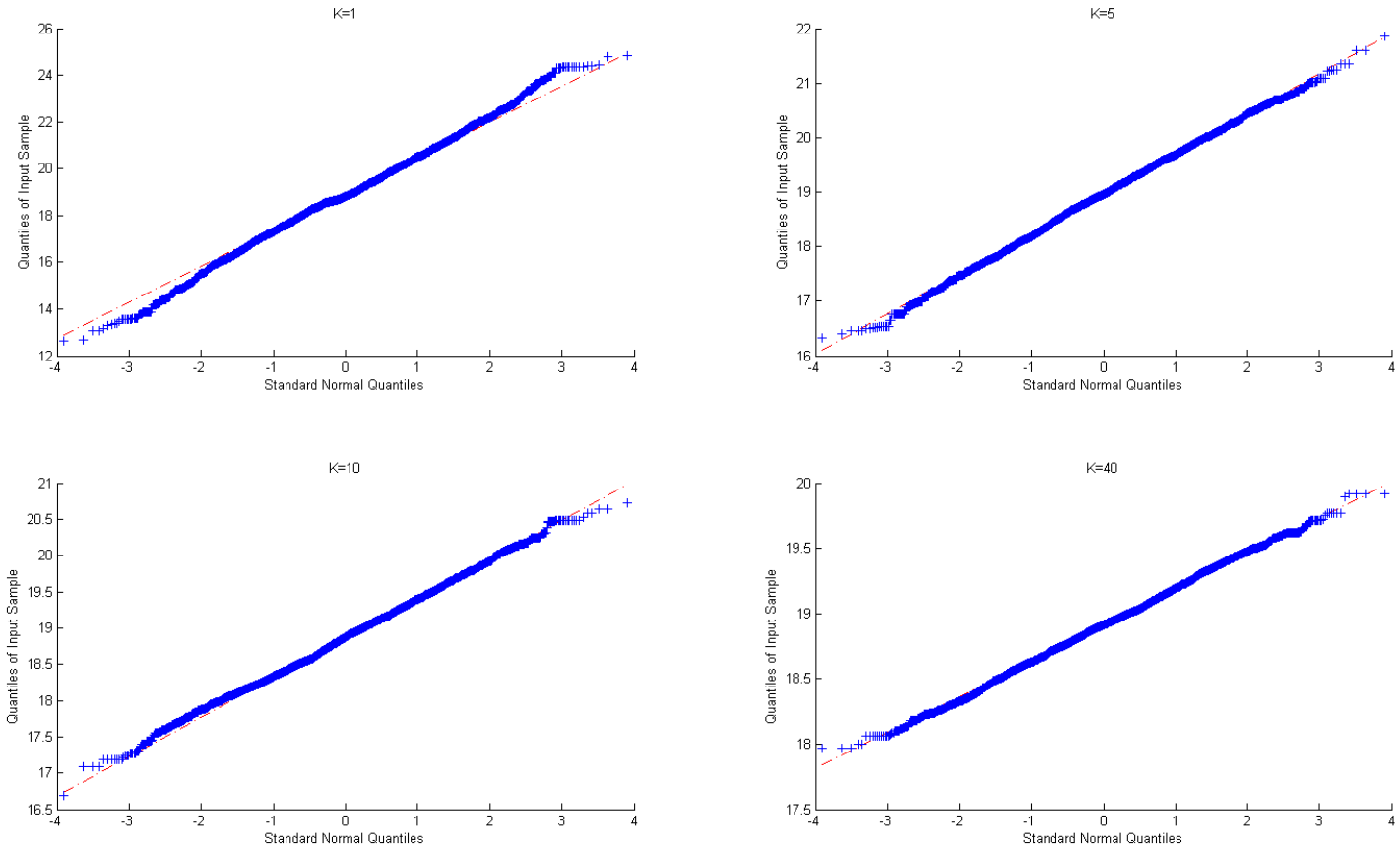

Figure 2.5: Quantile-quantile plots for the simulations of β_3 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

2.2.1 Simple Linear Regression

When performing the estimations for a simple linear regression example, some discrepancies from the results in the multiple linear regression case was observed. The example and the results are therefore briefly mentioned here.

In the case of simple linear regression, the model scheme above (in (2.1)) will amount to:

$$Y_i = \beta_0 + \beta_1 x_i + \eta_i, i = 1, ..., n, \tag{2.3}$$

where η_i is the noise process. For instance, η_i might be independent identically distributed random variables, normally distributed withe the mean zero and variance σ^2, i.e. $\eta_i \sim N(0, \sigma^2)$. In such a framework, $\boldsymbol{\varphi} = (\beta_0, \beta_1, \sigma^2)$ where $\boldsymbol{\varphi}$, from (2.1), is all the unknown parameters affecting the observations.

Here, it is worth to take a look at an example and some of its practical aspects. For a model, this author chose $x = (1, ..., 20)$ and $\boldsymbol{\varphi} = (10, 2, 4)$. To improve the acceptance rate over time, a Metropolis adaptive scheme was employed. Updating the covariance matrix of the random walk proposal was done through the update mechanism in (1.3) with updates done every 100^{th} algorithmic iterations. The priors that were chosen were all uniform in the following manner: $\beta_0 \sim U(-100, 100)$, $\beta_1 \sim U(-20, 20)$ and $\sigma^2 \sim U(0, 900)$. They are all indeed fairly wide and non-informative.

Figure (2.6) illustrates certain features of continuously increasing the number of clones through both the CIC and the MIC methods. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation, starting at $K = 1$. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. This procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result.

The precision of the estimates of the MLE and that of the covariance matrix of the MLE seems to increase as the number of clones increase (a,b), and the convergence to the MLE is at a faster rate than the multiple linear regression case. The acceptance rates seems also to stabilize around 31%. This satisfactory since according to Roberts et al. (1997, p. 113), the theoretically optimal acceptance rate for random walk metropolis algorithms (under certain stability conditions) is about 23%. Moreover, the ω- and $\tilde{r}^2$-statistics are close to zero and even converge to that value as the number of clones K is increased (d,e).

Furthermore, the standardized largest eigenvalue of the variance covariance matrix (λ_K^S) seems to converge to zero, indicating estimability of the MLE of $\boldsymbol{\varphi} = (\beta_0, \beta_1, \sigma^2)$, which is correct (f). However the rate of convergence is faster than $1/K$. The convergence is, nevertheless, correct as the diagnostic statistics are close to zero. The faster rate of convergence may be due to the additional information provided by each block of simulation when increasing the number of clones, rather than running each simulation separately for each number of copies. The multiple linear regression case does not manifest the same advantageous behavior. Hence, the simplicity of the model may also be a factor in the manifestation of the observed behavior. A summary of the estimates for some different number of copies is mentioned in table (2.2) for both the CIC and the MIC method.

	β_0	β_1	σ^2
True Val.	10	2	4
MLE	8.6194 (1.1298)	2.0636 (0.0943)	5.9157 (1.8707)
CIC			
K=1	8.6587 (1.3802)	2.0607 (0.1160)	8.4918 (3.4653)
K=5	8.6273 (1.1700)	2.0621 (0.0984)	6.3005 (2.1249)
K=10	8.6311 (1.1035)	2.0628 (0.0929)	6.0901 (1.8783)
K=40	8.6123 (1.1160)	2.0641 (0.0951)	5.9716 (1.9278)
MIC			
K=1	8.6506 (1.3473)	2.0626 (0.1160)	8.4165 (3.2683)
K=5	8.6115 (1.1765)	2.0634 (0.0965)	6.2805 (2.0590)
K=10	8.6267 (1.1261)	2.0636 (0.0946)	6.1292 (2.0266)
K=40	8.6174 (1.1533)	2.0636 (0.0949)	5.9689 (1.8847)

Table 2.2: Summary of the estimates of the simple linear regression model: the true values, the MLE and the estimates via the MIC and CIC methods of data cloning (DC) for some different number of clones. The standard errors are mentioned in the parenthesis. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. The Metropolis update of the covariance of the random walk proposals was performed every 100^{th} algorithmic iterations. The initial value (guess) of $\boldsymbol{\varphi}$ was the true values of the parameters added with some minor zero-mean normal noise.

There are some differences between the behavior of the estimates of the parameters of simple and multiple linear regression models. It has already been mentioned that the convergence to the MLE, in the simple linear regression model is at a faster rate than the multiple linear regression case. These are the behavior of the standardized largest eigenvalues and that of the ω- and the $\tilde{r}^2$-statistics. While in the simple linear regression case, the ω- and the $\tilde{r}^2$-statistics converge to zero (Figure (2.6)d,e), in the multiple linear regression case (Figure (2.2)d,e) they converge to zero in the beginning, but remain only close to zero afterwards. This is not problematic from a theoretical stand point, since, in theory, the ω- and the $\tilde{r}^2$-statistics do not necessarily converge to zero, but should merely be close to zero. In other words, the performance of the algorithm for the simple linear regression case is too well, and for the multiple linear regression case in line with theory.

The same is true with respect to the convergence of the standardized largest eigenvalues of the covariance matrices (λ_K^S). In the simple linear regression case (Figure (2.6f), the convergence to zero is faster than the theoretical rate of $1/K$. In the multiple linear regression case, however, the rate of convergence is visibly equal to that value (Figure 2.2f). The convergence to zero in both cases correctly illustrates estimability of the unknown parameters.

In short, the simplicity of the model, in the simple linear regression case, together with the additional information provided by each block of simulation when increasing the number of clones (rather than running each simulation separately for each number of copies) may be the factors in the manifestation of the observed behavior.

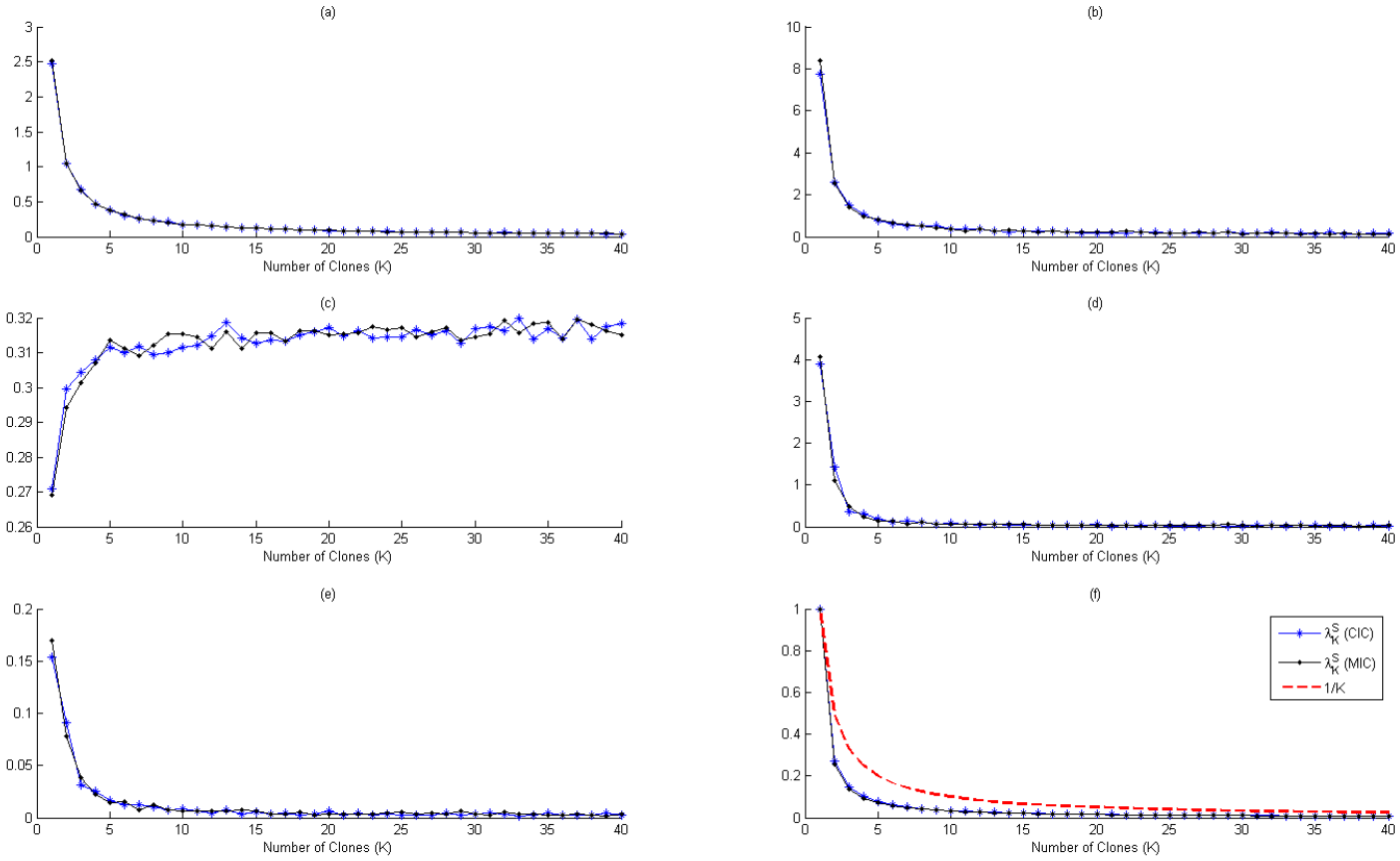

Figure 2.6: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. This procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE (the 2-norm of the difference), (b) Precision of the estimates of the variance covariance matrix of the MLE (the 2-norm of the difference) (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

Similar results are found when one increases the number of clones sporadically (Figure (2.7)). The number of clones were $1, 2, 5, 10, 20$ and 40. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. As before, this procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result. No visible disadvantages are observed to increasing the number of clones continuously rather and sporadically, indicating that one should perform the estimation through the latter one, saving computational time.

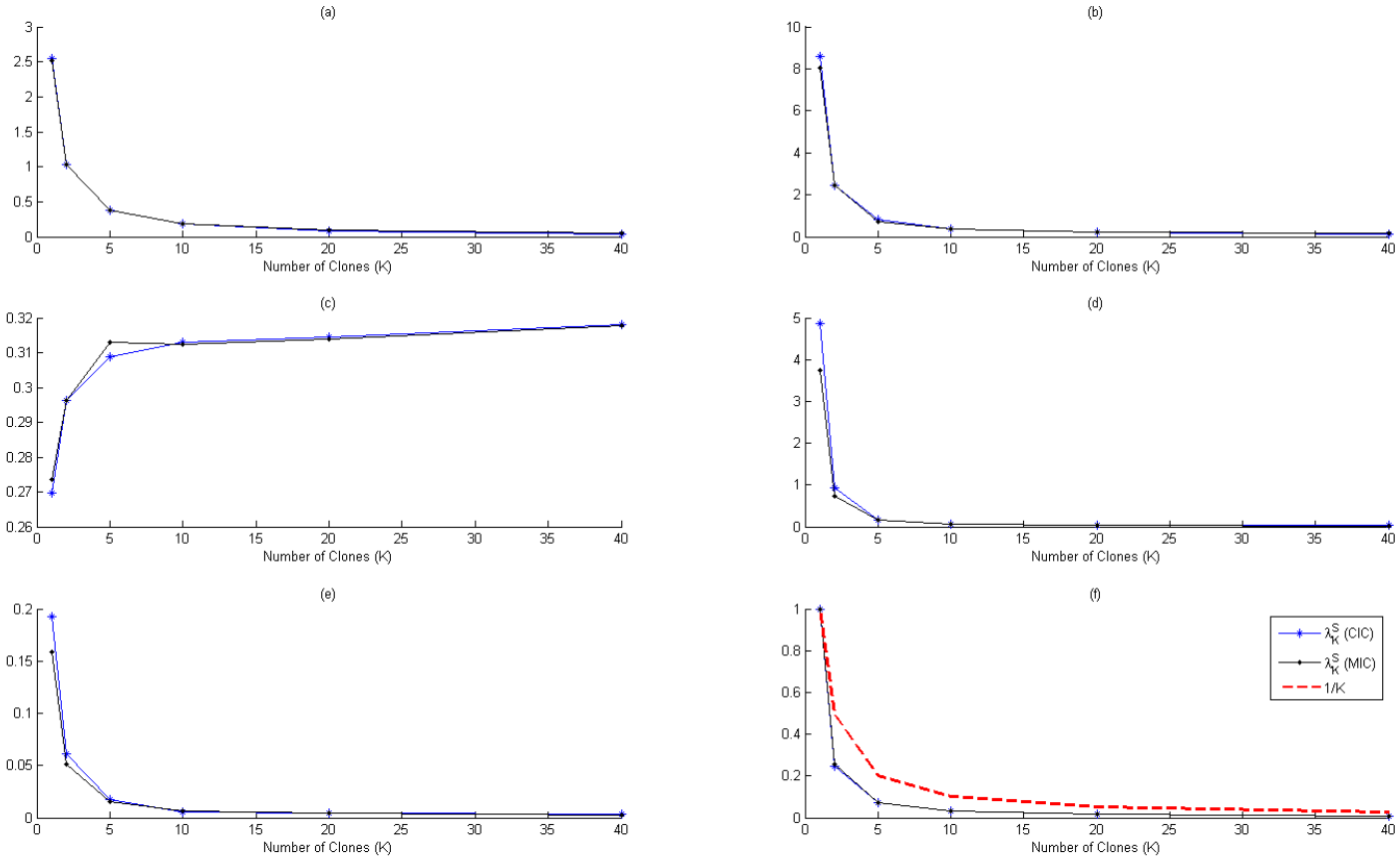

Figure 2.7: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase was done sporadically. The number of clones were $1, 2, 5, 10, 20$ and 40. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. This procedure was conducted 20 times and the average of the estimates and their corresponding statistics were calculated in order to present a smoother result. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE (the 2-norm of the difference), (b) Precision of the estimates of the variance covariance matrix of the MLE (the 2-norm of the difference) (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

2.3 Some Practical Issues

Some practical matters are worth mentioning. The choice of the initial value given to the MCMC data cloning procedure affects how many iterations is needed (as burn-in) before the process converges to the neighborhood of the true MLE. The choice of priors has a similar affect. Informative priors speeds up the convergence of the MCMC chain. As mentioned in section (1.2), however, the convergence itself is invariant to the choice of priors. This state of invariance was tested, by choosing the priors differently and getting similar results.

Another crucial issue is the tuning of the random walk proposals. The initial guess

of the covariance matrix, C_0 in (1.2) and (1.3), speeds up the process and yields more suitable acceptance rates. The choice could be done by letting the MCMC chain run for a while and adapt, and then restart the process, with the adapted covariance matrix C_t. Furthermore, one should make use of the whole information in the covariance updates C_t and not just the variances on the diagonal. Otherwise, the proposals tend to deteriorate in quality and the acceptance rate diminishes drastically.

Chapter 3

Stochastic Dynamical Models without Observational Noise

Applying the data cloning method to stochastic dynamical models, without observational noise, is very similar to the case of static models. The only difference is indeed how the likelihood function is calculated, which depends on the model. In other words, Algorithm (3) applies in this case as well. In the notation used in this chapter, the Y in Algorithm (3) corresponds to the variable Z in this section. Hence the model in (2.1) is replaced by:

$$Z \sim f(z; x, \boldsymbol{\varphi}).$$

The example that was chosen here was the Gompertz model which has been used in agronomy , e.g. for modeling the growth of chickens (Ditlevsen and Samson, 2013, pp. 29-35):

$$x(t) = Ae^{-Be^{-Ct}}, \tag{3.1}$$

which depends on the parameters A, B and C (all positive), and verifies the ordinary differential equation (ODE) below:

$$x'(t) = BCe^{-Ct}x(t), x(0) = Ae^{-B} \tag{3.2}$$

From (3.1), one can see that $x(t) \to A$ as $t \to \infty$. Hence, A can be interpreted as the equilibrium weight of the chicken. The interpretation of the parameters B and C, on the other hand, is not as straightforward. The parameter B, could be seen as a parameter determining what ratio of the equilibrium weight the chicken has at birth, since $x(0) = Ae^{-B}$. The parameter C, on the other hand, determines at what rate the weight of the chicken approaches the equilibrium value A. The larger C is, the faster the weight of the chicken approaches the value A.

Accommodating for heteroscedasticity, the stochastic differential equation (SDE) derived from the equation (3.2) will then be the following:

$$dX_t = BCe^{-Ct}X_t dt + \sigma X_t dW_t, X_0 = Ae^{-B} \tag{3.3}$$

where W_t is a Wiener process, and σX_t is the diffusion coefficient ($\sigma > 0$). The equation (3.3) is an Itô process that belongs to the family of Geometric Brownian motions with time inhomogeneous drift, and hence has an explicit solution. By setting $Z_t = \ln(X_t)$ and applying Itô's formula, the conditional distribution of $Z_{t+h}|(Z_s)_{s\leq t}, h > 0$ is given by:

$$Z_{t+h}|(Z_s)_{s\leq t} \sim N(Z_t - Be^{-Ct}(e^{-Ch} - 1) - \frac{1}{2}\sigma^2 h, \sigma^2 h), Z_0 = \ln(A) - B,$$

which in discretized form corresponds to the following:

$$Z_{t_j}|Z_{t_{j-1}} \sim N(Z_{t_j} - Be^{-Ct_{j-1}}(e^{-C\Delta_j} - 1) - \frac{1}{2}\sigma^2\Delta_j, \sigma^2\Delta_j), Z_0 = \ln(A) - B, \quad (3.4)$$

where $\Delta_j = (t_j - t_{j-1})$. The parameters to be estimated are $\boldsymbol{\varphi} = (\ln(A), B, C, \sigma^2)$.[1]

Since the model has a Markovian structure to equation (3.4), the likelihood function is then calculated by the following:

$$f(\boldsymbol{Z}|\boldsymbol{t}, \boldsymbol{\varphi}) = \prod_{j=1}^{n} f(Z_{t_j}|Z_{t_{j-1}}) \quad (3.5)$$

where $\boldsymbol{Z} = (Z_{t_0}, ..., Z_{t_n})$, $\boldsymbol{t} = (t_0, ..., t_n)$, n the number of data and $f(Z_{t_j}|Z_{t_{j-1}}, j = 1, ..., n$ is given by equation (3.5).

Since the likelihood function does not depend on $\ln(A)$ in the current format we assign a distribution to the first observation as well, i.e.

$$Z_{t_0} \sim N(\ln(A) - B, \sigma_0^2) \quad (3.6)$$

where σ_0^2 is a meta-parameter and assumed to be known. The practical interpretation of the variance of the first observation σ_0^2 could be seen as the variance due to the different characteristics of the parents of the chicken. Hence, the final and complete likelihood function would be:

$$f(\boldsymbol{Z}|\boldsymbol{t}, \boldsymbol{\varphi}) = f(Z_{t_0}) \prod_{j=1}^{n} f(Z_{t_j}|Z_{t_{j-1}}) \quad (3.7)$$

where $f(Z_{t_0})$ is derived from equation (3.6) and the rest as in equation (3.5).

For our example we chose the following time points and parameters

$$\begin{aligned} \boldsymbol{t} &= (0, 4, 6, 8, 12, 16, 20, 24, 28, 32, 36, 40), \\ \boldsymbol{\varphi} &= (\ln(A), B, C, \sigma) = (8, 5, 0.2, 0.01), \\ \sigma_0 &= 0.02. \end{aligned}$$

The time is expressed in weeks. In other words, $\exp(Z_t)$ is the weight of the chicken at week t. The observations in $\boldsymbol{Z}$ were then simulated according to equations (3.6) and (3.4). One such simulation is shown in Figure (3.1).

In order to find the MLE, the negative logarithm of the likelihood function in (3.7), was minimized through the use of the MATLAB function `fminunc`, which performs unconstrained nonlinear optimization. The algorithm used was the default *trust-region* algorithm. This function also provides a Hessian at the estimated point, which is approximated using finite differences. The inverse of the Hessian then was taken to attain the varaince-covariance matrix of the true MLE.

[1] One could, of course, choose $\boldsymbol{\varphi} = (A, B, C, \sigma^2)$, but the alternative chosen above provides better scaling for the subsequent comparisons. Hence, the choice was merely a practical one, and not theoretically motivated.

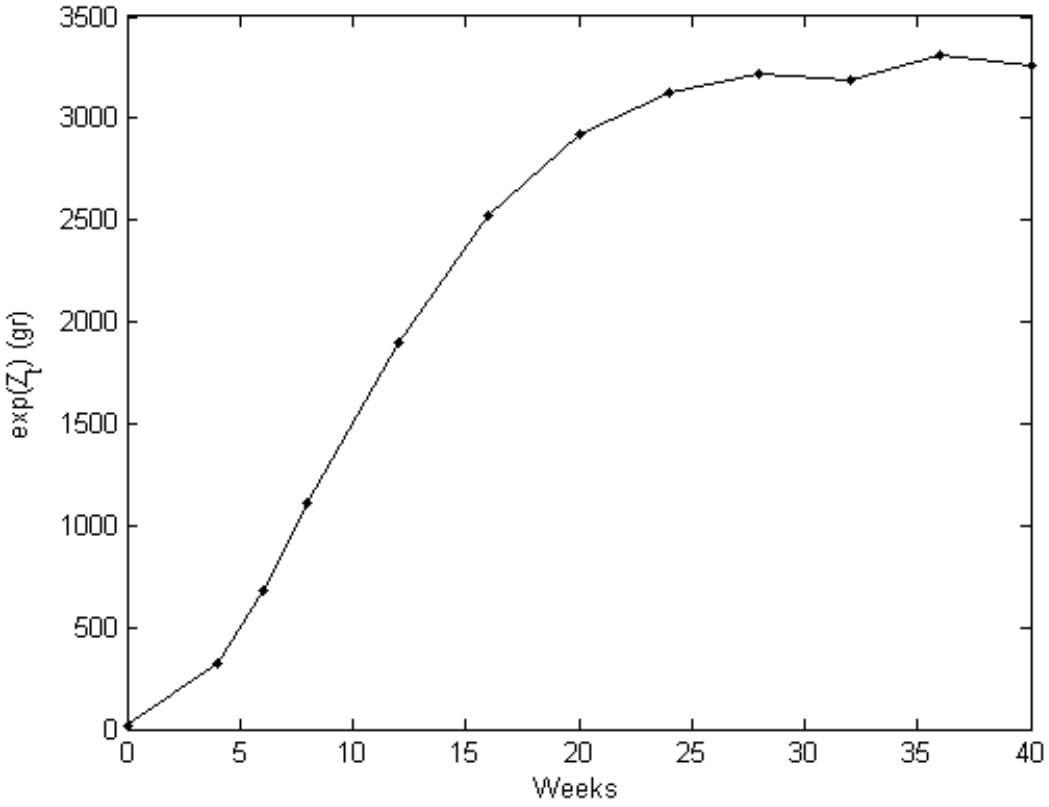

Figure 3.1: A simulation of a stochastic dynamic model based on the Gompertz model.

As in the static model case, when performing the data-cloning procedure one needs to choose prior distributions for the unknown parameters. Here the following were chosen,

$$\ln(A) \sim U(\ln(500), \ln(5000))$$
$$B, C, \sigma \sim U(0, 10)$$

which are fairly uninformative. The number of iterations for each K number of clones was ten thousand regular plus ten thousand burn-in. The covariance of the proposal function was updated every 100^{th} step. The starting values in the simulations are the true values of the parameters, not to be confused with the true MLE. Using the MCMC method via data cloning the following simulations were attained (Figures 3.2 and 7.18). The simulations via both the MIC and the CIC methods prove to converge to the true MLE, and approach it as the number of copies is increased.

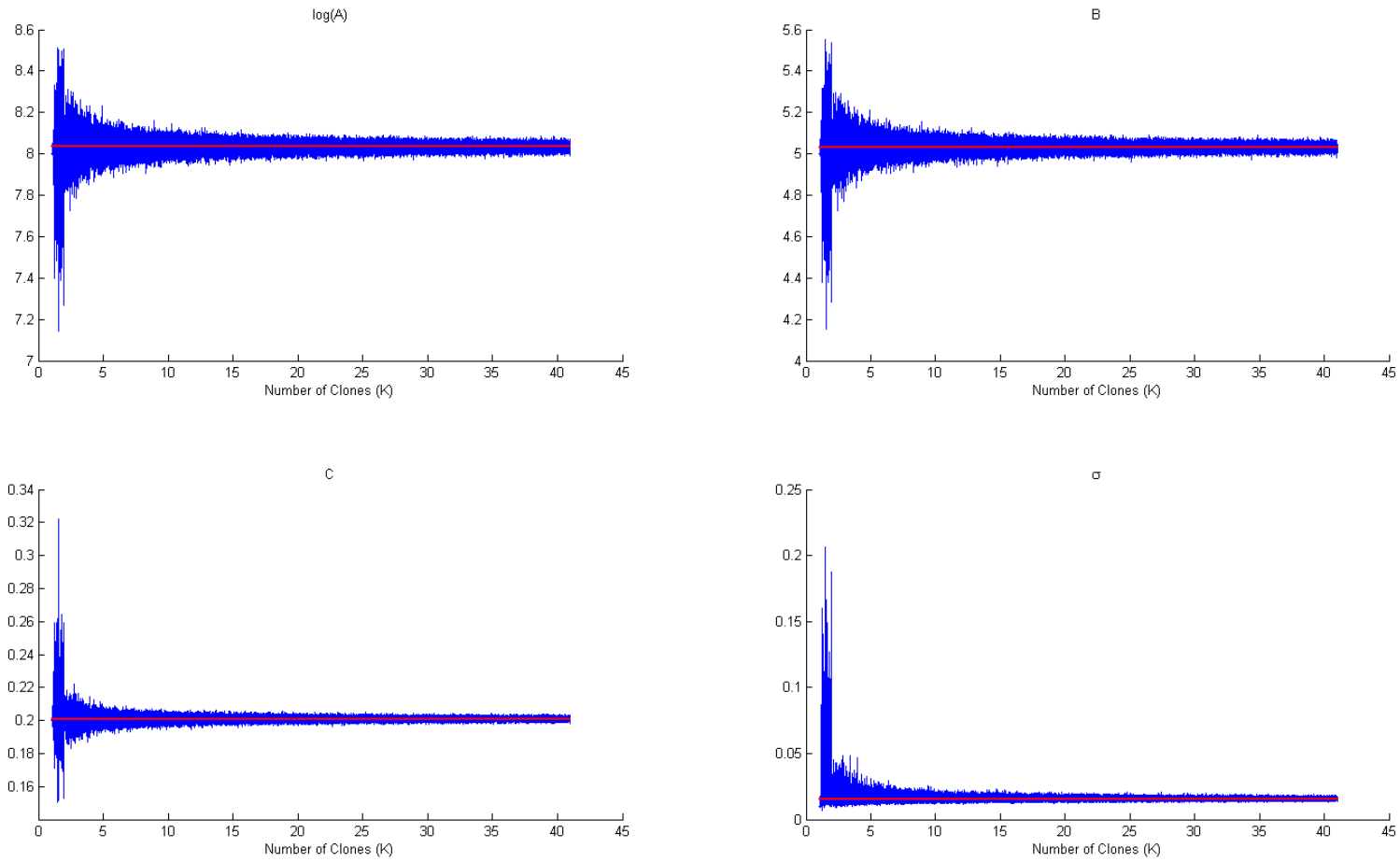

Figure 3.2: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 iterations, applying the CIC method to the Gompertz model. The red line is the true MLE. The starting values in the simulation are the true values of the parameters, not to be confused with the true MLE.

Some of the properties of the CIC and MIC method applied to the Gompertz model are summarized in Figure (3.3). We see that the estimates and their respective variance covariance matrices converge to each other as the number of clones (K) increase (Figures a,b). Furthermore, both stabilize with respect to the acceptance rate (Figure c). Moreover, the ω- and $\tilde{r}^2$-statistics are close to zero and even converge to that value as the number of clones K is increased (d,e). In addition, the largest standardized eigenvalue (λ_K^S) approaches zero as the number of clones increase. However, the rate of convergence is faster than $1/K$. This may be due to the additional information provided by continuously increasing the number of clones, rather than running each simulation separately for each number of copies. The convergence is, nevertheless, correct as the diagnostic statistics are close to zero. The rate of convergence of λ_K^S towards zero seems to be improved by a factor 4 in this way. Increasing the number of clones sporadically does not seem to have any significant effect on the convergence of the chain either (Figure (7.19)). Hence, in order to reduce computational time, the

latter should be preferred.

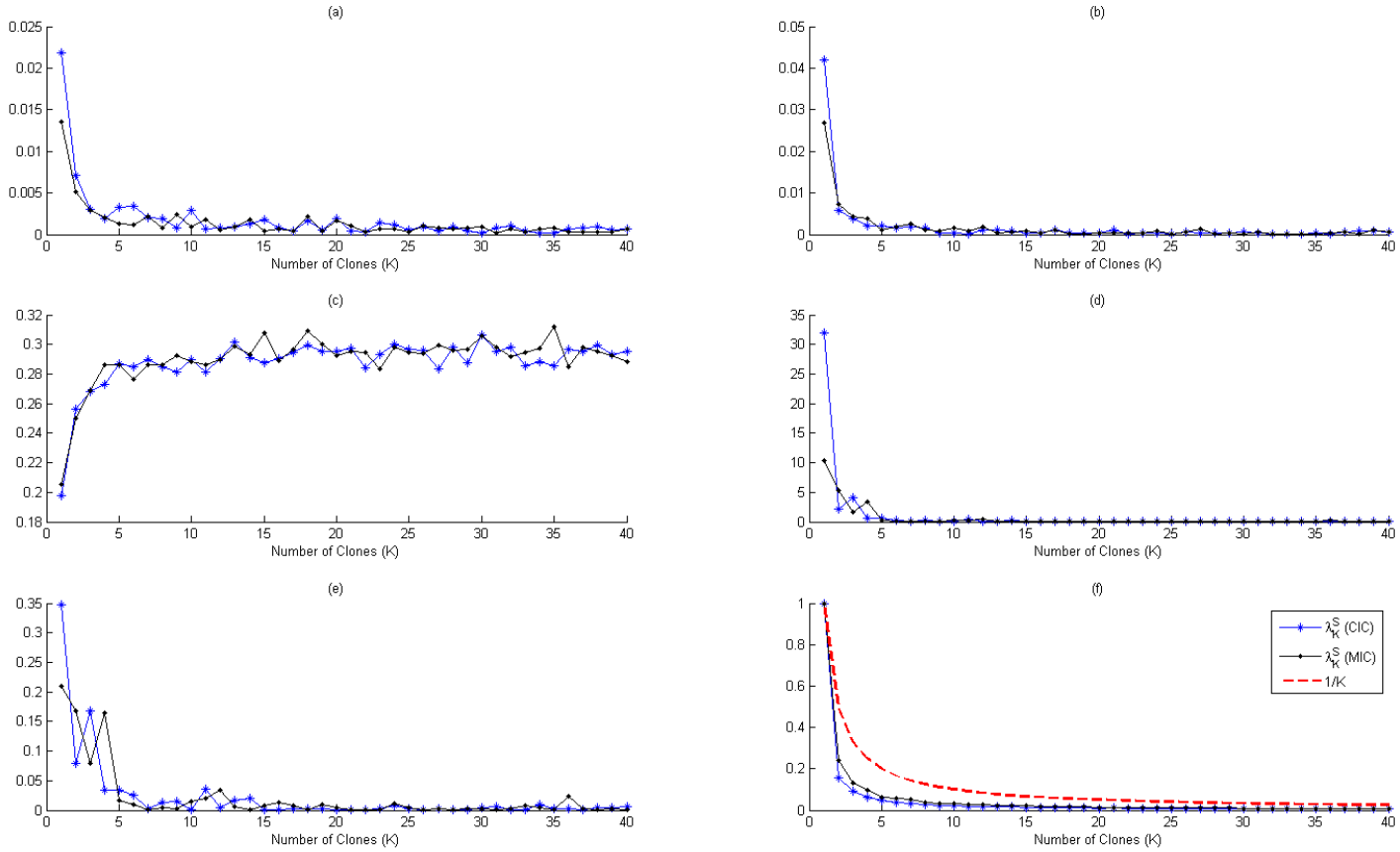

Figure 3.3: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE (the 2-norm of the difference), (b) Precision of the estimates of the variance covariance matrix of the MLE (the 2-norm of the difference) (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

As seen in Figure (3.4), increasing the number clones drastically improves the normality of the pseudo-posterior for the B parameter. This is true for all the other parameters as well.[2] As seen in the case of data cloning performed on static models, the CIC and MIC methds do not seem to have any advantages over the other. A summary of the estimates are mentioned in table (3.1). Observe that, the practical issues mentioned in section (2.3) is also applicable for Bayesian estimation of the MLE via data cloning in the case of the type of models of this chapter, i.e. stochastic dynamical models without observational noise.

[2]See section (7.2).

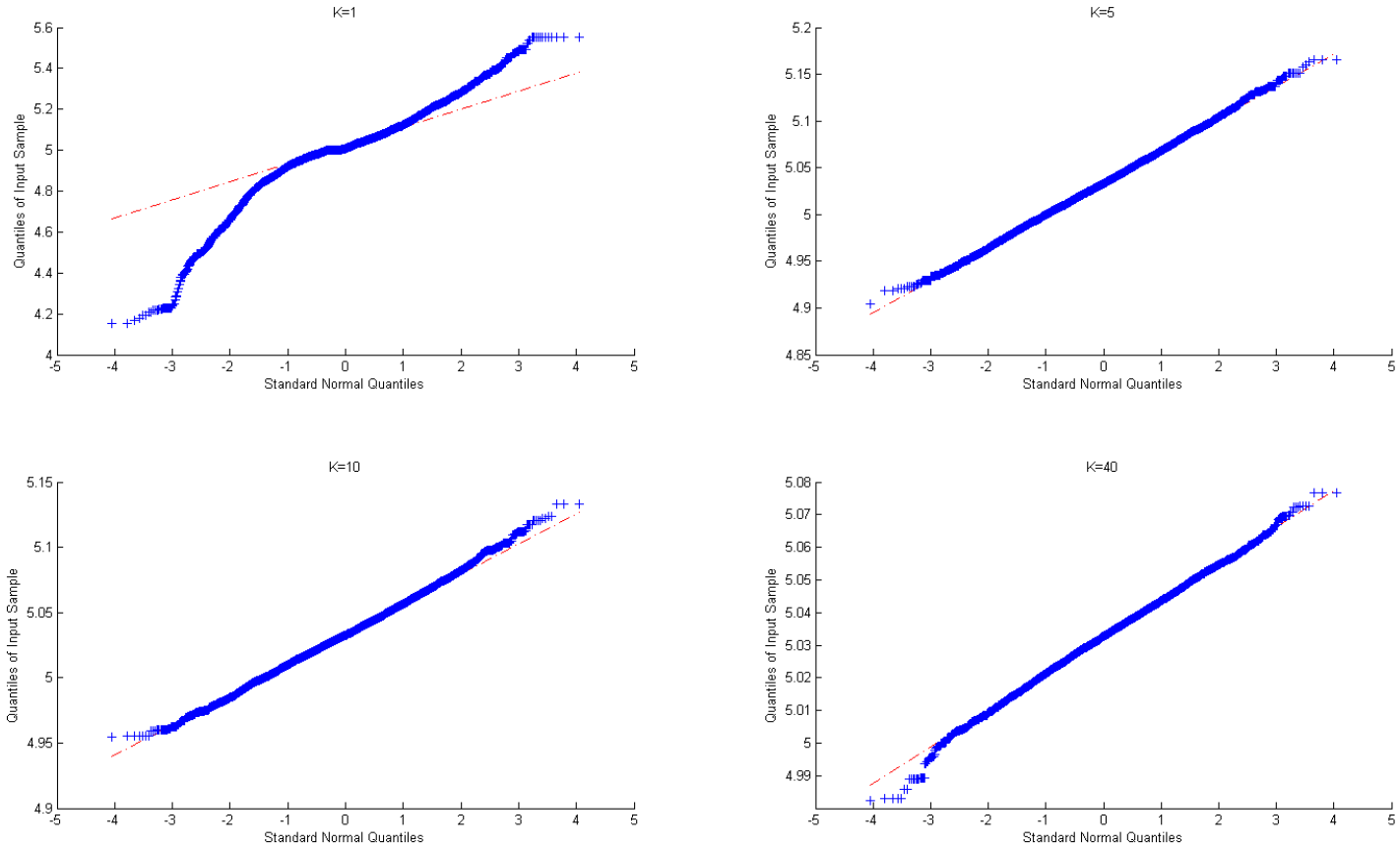

Figure 3.4: Quantile-quantile plots for the simulations of B using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

	$\ln(A)$	B	C	σ
True Val.	8	5	0.2	0.01
MLE	8.0336 (0.0746)	5.0323 (0.0719)	0.2010 (0.4543·1e-2)	0.0155 (0.4575·1e-2)
CIC				
K=1	8.0235 (0.1627)	5.0223 (0.1615)	0.2024 (1.2415·1e-2)	0.0321 (2.0671·1e-2)
K=5	8.0353 (0.0810)	5.0346 (0.0788)	0.2009 (0.4990·1e-2)	0.0170 (0.5518·1e-2)
K=10	8.0356 (0.0759)	5.0342 (0.0729)	0.2009 (0.4591·1e-2)	0.0162 (0.4757·1e-2)
K=40	8.0342 (0.0763)	5.0326 (0.0738)	0.2009 (0.4582·1e-2)	0.0157 (0.4557·1e-2)
MIC				
K=1	8.0375 (0.1380)	5.0354 (0.1361)	0.2010 (0.9031·1e-2)	0.0281 (1.4492·1e-2)
K=5	8.0337 (0.0784)	5.0318 (0.0754)	0.2010 (0.4750·1e-2)	0.0167 (0.5217·1e-2)
K=10	8.0331 (0.0794)	5.0317 (0.0777)	0.2009 (0.4926·1e-2)	0.0161 (0.5186·1e-2)
K=40	8.0341 (0.0767)	5.0327 (0.0740)	0.2009 (0.4646·1e-2)	0.0157 (0.4635·1e-2)

Table 3.1: Summary of the estimates of the Gompertz model: the true values, the MLE and the estimates via the MIC and CIC methods of data cloning (DC). The standard errors are mentioned in the parenthesis.

Chapter 4

State Space Models

4.1 Theoretical Background

In state space models (or hidden Markov models), there are at least two levels of hierarchy:

$$\begin{cases} \text{Hidden state:} & \boldsymbol{X}_t \sim f(x_t|X_{t-1};\boldsymbol{\vartheta}) \\ \text{Observed state:} & \boldsymbol{Y}_t \sim g(y_t|X_t;\boldsymbol{\theta}) \end{cases} \tag{4.1}$$

where f the distribution for the hidden state with the parameters $\boldsymbol{\vartheta} = (\vartheta_1, ..., \vartheta_{d_x})$, and g the distribution for the observed state with the parameters $\boldsymbol{\theta} = (\theta_1, ..., \theta_{d_y})$. Hence the total parameter space $\boldsymbol{\varphi} = (\boldsymbol{\vartheta}, \boldsymbol{\theta})$ has the dimension $d = d_x + d_y$.

4.1.1 Data Cloning for State Space Models

As performed in the previous section,when performing Bayesian inference via data-cloning, we start from a prior regarding the parameters, say $\pi(\boldsymbol{\varphi})$, in order to sample from the pseudo-posterior $\pi(\boldsymbol{\varphi}|y_{1:t}^{(K)})$, where

$$y_{1:t}^{(K)} = \underbrace{(y_{1:t}, ..., y_{1:t})}_{K \text{ times}}$$

where, $y_{1:t} = (y_1, ..., y_t)$. When $K = 1$, classical Bayesian inference is performed. The samples from the pseudo-posterior is then used to conduct Bayesian inference on the posterior with increased numerical accuracy, where the asymptotic properties of said inference are discussed in section (1.2). Ultimately, data-cloning is a Bayesian approach to approximated maximum likelihood estimation, that converges to the MLE asymptotically. Hence, as in earlier sections, we are concerned with the likelihood function $L(\boldsymbol{\varphi}; y_{1:t}) = p(y_{1:t}|\boldsymbol{\varphi})$.

For the the state space model in (4.1), the likelihood function is the following:

$$L(\boldsymbol{\varphi}; y_{1:t}) = p(y_{1:t}|\boldsymbol{\varphi}) = p(y_1) \prod_{i=2}^{t} p(y_i|y_{1:i-1};\boldsymbol{\varphi}) \tag{4.2}$$

$$= \int g(y_{1:t}|x_{1:t};\boldsymbol{\varphi}) p(x_{1:t}|\boldsymbol{\varphi}) dx_{1:t} \tag{4.3}$$

$$= \int \prod_{i=1}^{t} g(y_i|x_i,\boldsymbol{\theta}) \prod_{i=1}^{t} f(x_i|x_{i-1},\boldsymbol{\vartheta}) dx_{1:t} \tag{4.4}$$

where $x_i, i = 1, .., t$ are the hidden states,[1] and appropriately $x_{1:t} = (x_1, ..., x_t)$ and $dx_{1:t} = (dx_1, ..., dx_t)$. The second equality exploits Bayes' theorem, and the fourth equality exploits the conditional independence between the observed states and the Markovian structure of the hidden states.

When performing data-cloning, as mentioned before, we assume that K independent copies of the observed states are gathered $(y_{1:t}^{(K)})$. Then, $y_{1:t}^{(K)}$ is assumed to be the actual available observed data set. The independence of the copied observed states, in turn, assumes that K independent realizations of the hidden process $\{X_{1:t}\}$ have produced K identical observed states $y_{1:t}$. Let these states be $x_{1:t}^{(k)}, k = 1, ..., K$. Then the likelihood function for the cloned data set would be the following:

$$L(\boldsymbol{\varphi}; y_{1:t}^{(K)}) = p(y_{1:t}^{(K)}|\boldsymbol{\varphi}) = \int ... \int \prod_{k=1}^{K} \left[g(y_{1:t}|x_{1:t}^{(k)}; \boldsymbol{\varphi}) p(x_{1:t}^{(k)}|\boldsymbol{\varphi}) \right] dx_{1:t}^{(1)} ... dx_{1:t}^{(K)} \tag{4.5}$$

$$= \prod_{k=1}^{K} \left[\int g(y_{1:t}|x_{1:t}^{(k)}; \boldsymbol{\varphi}) p(x_{1:t}^{(k)}|\boldsymbol{\varphi}) dx_{1:t}^{(k)} \right] = \left(L(\boldsymbol{\varphi}; y_{1:t})\right)^K, \tag{4.6}$$

where the last equality follows from (4.2) and (4.3), and

$$g(y_{1:t}|x_{1:t}^{(k)}; \boldsymbol{\varphi}) = \prod_{i=1}^{t} g(y_i|x_i^{(k)}; \boldsymbol{\theta}),$$

$$p(x_{1:t}^{(k)}|\boldsymbol{\varphi}) = \prod_{i=1}^{t} f(x_i^{(k)}|x_{i-1}^{(k)}, \boldsymbol{\vartheta}),$$

as indicated by the arguments motivating the equality in (4.4). By (4.5) and (4.6), the likelihood function of the K-copied observations $(y_{1:t}^{(K)})$, is the likelihood of the actual observed state $(y_{1:t})$ to the power of the number of copies K. Therefore, the argument that maximizes $L(\boldsymbol{\varphi}; y_{1:t}^{(K)})$ is the same as the one which maximizes $L(\boldsymbol{\varphi}; y_{1:t})$, indicating that the data-cloning approach will target the same estimate. The fact that this target is the MLE, is proven by Lele et al. (2010). Indeed the asymptotic degeneracy of the pseudo-posterior at the true MLE was proven in section (1.2).

A procedure for parameter estimation via data-cloning is mentioned in Algorithm (4). A convenient choice of the proposal function for the hidden states is of course:

$$v(x_{1:t}|\boldsymbol{\varphi}) = p(x_{1:t}|\boldsymbol{\varphi}).$$

This choice reduces the acceptance rate to the following:

$$\alpha = \min(1, \frac{\tilde{q}^{\#}}{\tilde{q}^{*}} \times \frac{u(\boldsymbol{\varphi}^{*}|\boldsymbol{\varphi}^{\#})}{u(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^{*})}),$$

where the $\tilde{q}^{\#}$ and $\tilde{q}^{*}$ are as defined in (4.7) and (4.8). The acceptance rate, however, could be simplified further. Using random walk proposals (for instance normal random walks), the acceptance rate becomes the following:

$$\alpha = \min(1, \frac{\tilde{q}^{\#}}{\tilde{q}^{*}}),$$

since $u(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^{*}) = u(\boldsymbol{\varphi}^{*}|\boldsymbol{\varphi}^{\#})$ due to the known property of random walks.

[1] Observe that x_0 is assumed to be fixed.

Algorithm 4 A General Data Cloning MCMC process for Parameter Estimation in State Space Models using Metropolis Hastings Algorithm.

Assume a model format as in (4.1).

1. Initialization: Fix starting value $\boldsymbol{\varphi}^*$ or generate it from its prior $\pi(\boldsymbol{\varphi})$. Set numbers of copies equal to K. Set $\boldsymbol{\varphi}_1 = \boldsymbol{\varphi}^*$.

2. Generate K independent trajectories of the hidden state from a hidden state proposal function: $x_{1:t}^{*(k)} \sim v(x_{1:t}^{(k)}|\boldsymbol{\varphi}^*), k = 1, ..., K$.

3. Calculate the pseudo-posterior at this state:

$$q^* = \underbrace{\pi(\boldsymbol{\varphi}^*) \prod_{k=1}^{K} g(y_{1:t}|x_{1:t}^{*(k)}; \boldsymbol{\varphi}^*)}_{=:\tilde{q}^*} p(x_{1:t}^{*(k)}|\boldsymbol{\varphi}^*). \tag{4.7}$$

4. Propose a new state from the parameter proposal function: $\boldsymbol{\varphi}^{\#} \sim u(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^*)$. Generate K independent trajectories of the hidden state from a hidden state proposal function: $x_{1:t}^{\#(k)} \sim v(x_{1:t}^{(k)}|\boldsymbol{\varphi}^{\#}), k = 1, ..., K$. Calculate the pseudo-posterior at this state:

$$q^{\#} = \underbrace{\pi(\boldsymbol{\varphi}^{\#}) \prod_{k=1}^{K} g(y_{1:t}|x_{1:t}^{\#(k)}; \boldsymbol{\varphi}^{\#})}_{=:\tilde{q}^{\#}} p(x_{1:t}^{\#(k)}|\boldsymbol{\varphi}^{\#}). \tag{4.8}$$

5. Compare its value with the last accepted state i.e. q^*. In other words, calculate the acceptance probability

$$\alpha = \min[1, \underbrace{\frac{q^{\#}}{q^*}}_{\text{ratio of posteriors}} \times \underbrace{\frac{u(\boldsymbol{\varphi}^*|\boldsymbol{\varphi}^{\#}) \prod_{k=1}^{K} v(x_{1:t}^{*(k)}|\boldsymbol{\varphi}^*)}{u(\boldsymbol{\varphi}^{\#}|\boldsymbol{\varphi}^*) \prod_{k=1}^{K} v(x_{1:t}^{\#(k)}|\boldsymbol{\varphi}^{\#})}}_{\text{ratio of proposals}}]. \tag{4.9}$$

6. Generate a uniform random variable $\omega \sim U(0, 1)$. If $\omega < \alpha$, set $\boldsymbol{\varphi}_{j+1} = \boldsymbol{\varphi}^{\#}$, $\boldsymbol{\varphi}^* = \boldsymbol{\varphi}^{\#}$ and $q^* = q^{\#}$. Otherwise set $\boldsymbol{\varphi}_{j+1} = \boldsymbol{\varphi}_j$.

7. Repeat 3-6 for $j \leq N$ number of simulations, i.e. repeat until the chain for $\boldsymbol{\varphi}$ is deemed to have converged, where N is a large enough integer.

8. Set $\hat{\boldsymbol{\varphi}} = \frac{1}{N} \sum_{j=1}^{N} \boldsymbol{\varphi}_j$.

The value $\hat{\boldsymbol{\varphi}}$ is an approximation of the MLE.

4.1.2 Approximative Maximum Likelihood Estimation

Estimating the true MLE in state space models are not usually feasible. Therefore, an approximation of the MLE is needed. In order to estimate the correct MLE, we'll simulate different realizations of the hidden state $\boldsymbol{X}_t^{(r)}, r = 1, ..., R$, and use a special version of Sequential Importance Sampling with Resampling (SISR) called the *bootstrap particle filter* to find the best possible hidden states corresponding to the observed values. This method is a specific form of Sequential Monte-Carlo (SMC) methods. Below, is an introduction to the concept.

Importance Sampling

When performing MCMC-simulations, often the situation arises where the distribution which one wants to sample from, i.e. the *target distribution*, say, is either too complex and/or analytically ambiguous. In such a situation, one can use a so called *instrumental distribution* which is easy to sample from. Mathematically, the generalized idea of importance sampling could be expressed in terms of calculating an expectation (Cosma & Evers, 2010, pp. 35-37). Let X be a random variable with the target distribution $p(x)$ with support S, $q(x)$ an instrumental distribution and $h(x)$ a measurable function. Then,

$$\mathbb{E}_p(h(X)) = \int_S p(x)h(x)dx = \int_S q(x) \underbrace{\frac{p(x)}{q(x)}}_{=:w(x)} h(x)dx = \int_S q(x)w(x)h(x)dx = \mathbb{E}_q(w(X)\cdot h(X)), \quad (4.10)$$

if $q(x) > 0$ for (almost) all with $p(x) \cdot h(x) \neq 0$, when $x \in S$.

Hence, given a sample $X_1, ..., X_M \sim q$ and provided that $\mathbb{E}_q |w(X)h(X)|$ exists, then by the Law of Large Numbers:

$$\frac{1}{M}\sum_{k=1}^{M} w(X_k)h(X_k) \overset{\substack{a.s. \\ M\to\infty}}{\longrightarrow} \mathbb{E}_q(w(X)\cdot h(X))$$

which then by (4.10) yields

$$\frac{1}{M}\sum_{k=1}^{M} w(X_k)h(X_k) \overset{\substack{a.s. \\ M\to\infty}}{\longrightarrow} \mathbb{E}_p(h(X)).$$

Hence, in practice, $\hat{\mu} = \frac{1}{M}\sum_{k=1}^{M} w(X_k)h(X_k)$ could be used as an estimator of the the expectation $\mathbb{E}_p(h(X))$.

The weights $w(X_k), k = 1, ..., M$ do not necessarily sum up to M, and therefore, sometimes the normalized weights are used: $\tilde{w}(X_k) = \frac{w(X_k)}{\sum_{k=1}^{M} w(X_k)}$. Then the following two estimators act as candidates:

$$\begin{aligned} \hat{\mu} &= \tfrac{1}{M}\textstyle\sum_{k=1}^{M} w(X_k)h(X_k) \\ \tilde{\mu} &= \textstyle\sum_{k=1}^{M} \tilde{w}(X_k)h(X_k) \end{aligned} \quad (4.11)$$

The first estimator, $\hat{\mu}$, is unbiased, while the second one, $\tilde{\mu}$, is only asymptotically so (*ibid.*).

Not any distribution q could be instrumentally useful when estimating the estimation through any particular target distribution p. The first condition that needs to be

fulfilled, as already mentioned, is $\text{supp}(p \cdot h) \subset \text{supp}(q)$. Furthermore, finite variance of the estimator is also desirable. Both of the set of conditions below individually guarantee finite variance (*ibid.*):

- $p(x) < M \cdot q(x)$ and $Var_p(h(X)) < +\infty$
- S is compact, p is bounded above on S, and q is bounded below on S

The algorithm for importance sampling is mentioned in Algorithm (5).

Algorithm 5 Importance Sampling

Choose q such that $\text{supp}(p \cdot h) \subset \text{supp}(q)$:

1: **for** $k = 1, ..., M$ **do**
2: Generate $X_k \sim q$.
3: Set $w(X_k) = \frac{p(X_k)}{q(X_k)}$.
4: Return either $\tilde{\mu}$ or $\hat{\mu}$ from (4.11).

Sequential Importance Sampling

When dealing with dynamic models it is efficient and desirable to employ the sequential relation of the probabilities when using the importance sampling algorithm. Hence, the instrumental distribution also must be chosen such that it accommodates sequential procedure.

This could be expressed more formally in the following manner (Liu and Chen, 1998). Let the target distribution be $p_t(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_t)$, where $\mathbf{x}_t = (x_1, ..., x_t)$ are the random realizations from the target distribution, and $\mathbf{z}_t$ is the dynamical elements available that provide information of the behavior of the target distribution. For instance when dealing with state space models, the observed states are the dynamical elements providing this crucial information. Similarly, let $q_t(\mathbf{x}_t)$ be the instrumental distribution defined in a corresponding manner. The sequential pattern which is desired for the instrumental distribution, is expressed by the following factorization (Cosma and Evers, 2010, p. 81):

$$q_{t+1}(\mathbf{x}_{t+1}) = q_1(x_1) \prod_{k=2}^{t+1} q_k(x_k|\mathbf{x}_{k-1}) = q_t(\mathbf{x}_t) q_{t+1}(x_{t+1}|\mathbf{x}_t) \tag{4.12}$$

The sequential importance sampling (SIS) procedure is mentioned in Algorithm (6). The core idea is that given that the particles $\{\mathbf{x}_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^M$ act as fair approximations of the target distribution at time t, i.e. $p_t(\mathbf{x}_t)$, then through a suitable instrumental function $q_{t+1}(x_{t+1}|\mathbf{x}_t)$, the expanded weighted sample $\{\mathbf{x}_{t+1}^{(i)}, \tilde{w}_{t+1}^{(i)}\}_{i=1}^M$ would provide the same access and information with respect to the target distribution at time $t+1$, i.e. $p_{t+1}(\mathbf{x}_{t+1})$. This core idea is motivated by the following line of argument. At time $t > 1$:

$$w_t^{(i)} = \frac{p_t(\mathbf{x}_t^{(i)})}{q_t(\mathbf{x}_t^{(i)})}$$

Hence, for the following step,

$$w_{t+1}^{(i)} = \frac{p_{t+1}(\mathbf{x}_{t+1}^{(i)})}{q_{t+1}(\mathbf{x}_{t+1}^{(i)})} = \frac{p_{t+1}(\mathbf{x}_{t+1}^{(i)})}{q_t(\mathbf{x}_t^{(i)})q_{t+1}(x_{t+1}^{(i)}|\mathbf{x}_t^{(i)})} = w_t^{(i)} \underbrace{\frac{p_{t+1}(\mathbf{x}_{t+1}^{(i)})}{p_t(\mathbf{x}_t^{(i)})q_{t+1}(x_{t+1}^{(i)}|\mathbf{x}_t^{(i)})}}_{=:u_{t+1}^{(i)}} \tag{4.13}$$

where the second step follows from equation (4.12), and $u_t^{(i)}$ is called the *incremental weight*. Since $\{\mathbf{x}_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^M$ act as fair approximations of $p_t(\mathbf{x}_t)$, they could be used to make estimations of the target distribution, which otherwise would have been impossible. For instance, let h be a fixed measurable function, such that $\text{supp}(p \cdot h) \subset \text{supp}(q)$, then, under suitable regularity conditions, by the Law of Large Numbers,

$$\sum_{i=1}^{M} \tilde{w}_t^{(i)} h(X_t^{(i)}) \to \mathbb{E}_{p_t}(h(X)) \text{ as } M \to +\infty. \tag{4.14}$$

Algorithm 6 Sequential Importance Sampling

Let p be the target distribution and q the instrumental one.
For $i = 1, ..., M$ and $t = 1, ..., T-1$:

1. Set $t = 1$.

   1.1 Propagation: Sample $x_1^{(i)} \sim q_1(x_1)$.

   1.2 Updating: Set $w_1^{(i)} = p_1(x_1^{(i)})/q_1(x_1^{(i)})$. Normalize the the weights: $\tilde{w}_1^{(i)} = \frac{w_1^{(i)}}{\sum_{j=1}^M w_1^{(j)}}$.

   Set $t = 2$.

2. At time $t > 1$,

   2.1 Propagation: Draw sample $x_t^{(i)} \sim q_t(x_t|\mathbf{x}_{t-1}^{(i)}), i = 1, ..., M$.

   2.2 Updating: Set $\mathbf{x}_t^{(i)} = (\mathbf{x}_{t-1}^{(i)}, x_t^{(i)})$. Set $u_t^{(i)} = \frac{p_t(\mathbf{x}_t^{(i)})}{p_{t-1}(\mathbf{x}_{t-1}^{(i)})q_t(x_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}$ (the incremental weight) and let $w_t^{(i)} = u_t^{(i)} w_{t-1}^{(i)}$. Normalize the the weights: $\tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^M w_t^{(j)}}$.

   Set t to $t + 1$.

The particles $\{\mathbf{x}_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^M$ act as approximations of the target distribution at time t: $p_t(\mathbf{x}_t)$.

Doucet et al. (2000, p. 199) prove that the SIS-algorithm suffers from the phenomenon of *weight degeneracy*, which indicates that in the long run, certain few weighted particles accommodate most of the probability mass, while most other particles have (normalized) weights close to zero. Formally, this translates into the variance of the weights increasing over time (Cosma and Evers, 2010, p. 82). Hence, it is imperative to find a suitable instrumental distribution that minimizes said variance. The

optimal instrumental distribution, according to Doucet et al. (2000, p. 199), is,

$$q_{t+1}(x_{t+1}^{(i)}|\mathbf{x}_t^{(i)}) = p_{t+1}(x_{t+1}^{(i)}|\mathbf{x}_t^{(i)}) \tag{4.15}$$

which usually is difficult to sample from. When the instrumental distribution is chosen as in (4.15), then the weights simplify to $p_{t+1}(x_{t+1}^{(i)})/p_t(x_{t+1}^{(i)})$. This method is called the *bootstrap filter*, and fundamentally boils down to using $p_t(x_t)$ to predict x_{t+1} (Cosma and Evers, 2010, pp. 82).

Sequential Importance Sampling with Resampling

In the previous section, the problem of weight degeneracy of the SIS algorithm was mentioned. Furthermore, it was mentioned that choosing an optimal or semi-optimal instrumental distribution function, would alleviate said problem. A second approach is to perform *resampling* after each propagation and updating step. In such a manner, the sample is *rejuvenated* at each step.

The outline of the idea is as follows. Assume that through sequential importance sampling, the weighted sample is available $\{x_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^M \sim p_t(x_t)$, where the weights are normalized. Then the empirical density is given by the following:

$$\hat{p}_t(x_t) = \sum_{i=1}^{M} \tilde{w}_t^{(i)} \delta_{x_t^{(i)}}(x_t) \tag{4.16}$$

where $\delta_{x_t^{(i)}}(x_t)$ is the Dirac delta function with the density centered at $x_t^{(i)}$. Let h be a fixed measurable function. Then, given suitable regularity conditions, by (4.14), it follows

$$\mathbb{E}_{\hat{p}_t}(h(X)) \to \mathbb{E}_{p_t}(h(X)) \text{ as } M \to +\infty. \tag{4.17}$$

where $\mathbb{E}_{\hat{p}_t}(h(X)) = \sum_{i=1}^M \tilde{w}_t^{(i)} h(X_t^{(i)})$.

Instead of using the particles $\{x_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^M$, however, one could use a sample from them instead. In other words, take a sample of size $M' \leq M$ from said particles with replacement: $\tilde{x}_t^{(j)} = x_t^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_t^{(i)}$. The new particles, have now equal weights: $\{\tilde{x}_t^{(j)}, 1/M'\}_{j=1}^M$. Then again, by the Law of Large Numbers,

$$\frac{1}{M'} \sum_{j=1}^{M'} h(\tilde{X}_t^{(j)}) \to \sum_{i=1}^{M} \tilde{w}_t^{(i)} h(X_t^{(i)}) = \mathbb{E}_{\hat{p}_t}(h(X)) \text{ as } M' \to +\infty. \tag{4.18}$$

Hence, by (4.17) and (4.18),

$$\frac{1}{M'} \sum_{j=1}^{M'} h(\tilde{X}_t^{(j)}) \to \mathbb{E}_{p_t}(h(X)) \text{ as } M, M' \to +\infty. \tag{4.19}$$

In the SIS algorithm, the resampling step is applied to the whole trajectory $\mathbf{x}_t^{(i)} = x_{1:t}^{(i)}$, rather than the last particle. The extended algorithm is referred to as Sequential Importance Sampling with Resampling (SISR). The procedure is mentioned in Algorithm (7). Notice that due to the fact that resampling adds extra randomness to the sample, estimation is to be conducted *before* resampling.

Algorithm 7 Sequential Importance Sampling with Resampling

Let p be the target distribution and q the instrumental one.
For $i = 1, ..., M$ and $t = 1, ..., T-1$:

1. Set $t = 1$.

   1.1 Propagation: Sample $x_1^{(i)} \sim q_1(x_1)$.

   1.2 Updating: Set $w_1^{(i)} = p_1(x_1^{(i)})/q_1(x_1^{(i)})$. Normalize the the weights: $\tilde{w}_1^{(i)} = \frac{w_1^{(i)}}{\sum_{j=1}^{M} w_1^{(j)}}$.

   1.3 Resampling: Sample $\tilde{x}_1^{(j)} = x_1^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_1^{(i)}$. Set $w_1^{(i)} = 1/M'$.

   Set $M = M'$ and $t = 2$.

2. At time $t > 1$,

   2.1 Propagation: Draw sample $x_t^{(i)} \sim q_t(x_t|\mathbf{x}_{t-1}^{(i)}), i = 1, ..., M$.

   2.2 Updating: Set $\mathbf{x}_t^{(i)} = (\mathbf{x}_{t-1}^{(i)}, x_t^{(i)})$. Set $u_t^{(i)} = \frac{p_t(\mathbf{x}_t^{(i)})}{p_{t-1}(\mathbf{x}_{t-1}^{(i)}) q_t(x_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}$ (the incremental weight) and let $w_t^{(i)} = u_t^{(i)} w_{t-1}^{(i)}$. Normalize the the weights: $\tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{M} w_t^{(j)}}$.

   2.3 Resampling: Sample $\tilde{\mathbf{x}}_t^{(j)} = \mathbf{x}_t^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_t^{(i)}$. Set $w_t^{(i)} = 1/M'$.

   Set $M = M'$ and t to $t + 1$.

The particles $\{\mathbf{x}_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^{M}$ act as approximations of the target distribution at time t: $p_t(\mathbf{x}_t)$. In other words, estimation is done *before* resampling.

Since the resampled trajectories become dependent of each other, the variance of the estimate is increased. Therefore, there is a trade-off between the *resampling frequency* and the *variance of the estimator*. This notion, begs the question what the *effective sample size* (ESS) is. According to Kong et al. (1994, pp. 283-4), the effective sample size is given by:

$$\text{ESS} = \frac{M}{1 + \text{Var}_{q_{t+1}(x_{t+1})}(\tilde{w}_{t+1})} \tag{4.20}$$

Since $\mathbb{E}_{q_{t+1}(x_{t+1})}(\tilde{w}_{t+1}) = 1$ (i.e. the weights are normalized), (4.20) can be simplified, in the following manner (Cosma and Evers, 2010, pp. 86-7).,

$$\frac{M}{1 + \text{Var}_{q_{t+1}(x_{t+1})}(\tilde{w}_{t+1})} = \frac{M}{1 + \mathbb{E}_{q_{t+1}(x_{t+1})}[\tilde{w}_{t+1})^2] - (\mathbb{E}_{q_{t+1}(x_{t+1})}[\tilde{w}_{t+1}])^2} = \frac{M}{\mathbb{E}_{q_{t+1}(x_{t+1})}[(\tilde{w}_{t+1})^2]} \tag{4.21}$$

Since $\text{Var}_{q_{t+1}(x_{t+1})}(\tilde{w}_{t+1})$ and, consequently $\mathbb{E}_{q_{t+1}(x_{t+1})}[(\tilde{w}_{t+1})^2]$, are not often analytically attainable, an approximation is needed. One such approximation is the following:

$$\mathbb{E}_{q_{t+1}(x_{t+1})}[(\tilde{w}_{t+1})^2] \approx \frac{\mathbb{E}_{q_{t+1}(x_{t+1})}[(w_{t+1})^2]}{(\mathbb{E}_{q_{t+1}(x_{t+1})}[\sum_{i=1}^{M} w_{t+1}^{(i)}])^2} \approx \frac{M^{-1} \sum_{i=1}^{M} (w_{t+1}^{(i)})^2}{M^{-2} (\sum_{i=1}^{M} w_{t+1}^{(i)})^2} = M \sum_{i=1}^{M} (\tilde{w}_{t+1}^{(i)})^2$$

Then, by (4.20) and (4.21), the effective sample size could be approximated by,

$$\tilde{\text{ESS}} = \frac{1}{\sum_{i=1}^{M} (\tilde{w}_{t+1}^{(i)})^2}. \tag{4.22}$$

Furthermore, from (4.20), one notices that:

$$\text{ESS} \to M \text{ as } \text{Var}_{q_t(x_t)}(\tilde{w}_t) \to 0.$$

Hence, a value of $\tilde{\text{ESS}}$ close to M indicates that the sample contains almost as much information as an i.i.d. sample of size M from the target distribution $p_t(x_t)$. In practice a threshold, th_M, lower than M (often $M/2$) is chosen. When $\tilde{\text{ESS}} < th_M$, resampling is performed (*ibid.*).

Bootstrap Particle Filter

Wilkinson (2012, p. 294) describe the bootstrap particle filter as a simple SISR-technique for estimating the hidden states in a state space model conditional on the observations and the parameters. Consider the state-space model given in (4.1) and assume that the parameters $\boldsymbol{\varphi} = (\boldsymbol{\vartheta}, \boldsymbol{\theta})$ are known. Similar to the case of employing the bootstrap filter through SIS, when using the bootstrap particle filter, the instrumental function is

$$q_t(x_{1:t}^{(i)}) = p(x_t | x_{1:t-1}^{(i)}, y_{1:t-1}; \boldsymbol{\varphi}) = f(x_t | x_{t-1}^{(i)}; \boldsymbol{\vartheta}). \tag{4.23}$$

Then, the incremental weight $u_t^{(i)}$, defined in (4.13) is given by the following:

$$u_t^{(i)} = \frac{p(x_{1:t}^{(i)} | y_{1:t}; \boldsymbol{\varphi})}{p(x_{1:t-1}^{(i)} | y_{1:t-1}; \boldsymbol{\varphi}) f(x_t^{(i)} | x_{t-1}^{(i)}; \boldsymbol{\vartheta})} \tag{4.24}$$

The numertor, $p(x_{1:t}^{(i)} | y_{1:t}; \boldsymbol{\varphi})$, can however, be simplified. Given the hidden Markovian relation described in (4.1), the joint distribution of the hidden and observed state trajectories has the following recursive relation (Cosma and Evers, 2010, pp. 74-75),

$$\begin{aligned} p(x_{1:t}, y_{1:t} | \boldsymbol{\varphi}) &= p(x_1) g(y_1 | x_1; \boldsymbol{\theta}) \prod_{k=1}^{t} p(x_k, y_k | x_{1:k-1}, y_{1:k-1}) \\ &= p(x_1) g(y_1 | x_1; \boldsymbol{\theta}) \prod_{k=1}^{t} f(x_k | x_{k-1}; \boldsymbol{\vartheta}) g(y_k | x_k; \boldsymbol{\theta}) \\ &= p(x_{1:t-1}, y_{1:t-1} | \boldsymbol{\varphi}) f(x_t | x_{t-1}; \boldsymbol{\vartheta}) g(y_t | x_t; \boldsymbol{\theta}) \end{aligned}$$

Hence, by Bayes' theorem, the conditional density of interest satisfies the following,

$$p(x_{1:t} | y_{1:t}; \boldsymbol{\varphi}) \propto p(x_{1:t-1} | y_{1:t-1}; \boldsymbol{\varphi}) f(x_t | x_{t-1}; \boldsymbol{\vartheta}) g(y_t | x_t; \boldsymbol{\theta}) \tag{4.25}$$

Then, by (4.24), the the incremental weight of the bootstrap particle filter has the following property,

$$u_t^{(i)} \propto \frac{p(x_{1:t-1}^{(i)} | y_{1:t-1}; \boldsymbol{\varphi}) f(x_t^{(i)} | x_{t-1}^{(i)}; \boldsymbol{\vartheta}) g(y_t | x_t^{(i)}; \boldsymbol{\theta})}{p(x_{1:t-1}^{(i)} | y_{1:t-1}; \boldsymbol{\varphi}) f(x_t^{(i)} | x_{t-1}^{(i)}; \boldsymbol{\vartheta})} = g(y_t | x_t^{(i)}; \boldsymbol{\theta}) \tag{4.26}$$

Therefore, the incremental weight, in practice, could be used as $u_t^{(i)} = g(y_t|x_t^{(i)};\boldsymbol{\theta})$, which is very convenient.

The bootstrap particle filter, employing the effective sample size approach, is summarized by Algorithm (8). One may notice the addition of an initialization step compared to Algorithm (7). This step is necessary, since the distribution of the hidden state is only conditionally known.

Algorithm 8 Bootstrap Particle Filter

Let $th_M < M$.
For $i = 1, ..., M$ and while $t \leq T$,

1. Initialization: Set $t = 0$.

   1.1 Propagation: Sample $x_0^{(i)} \sim \pi(x_0)$ from some prior or choose $x_0^{(i)}$ fixed.

   1.2 Updating: Set $\tilde{w}_0^{(i)} = 1/M$. Set t=1.

2. At time $t = 1$.

   2.1 Propagation: Sample $x_1^{(i)} \sim f(x_1|x_0^{(i)};\boldsymbol{\vartheta})$.

   2.2 Updating: Set $w_1^{(i)} = g(y_1|x_1^{(i)};\boldsymbol{\theta})$. Normalize the the weights: $\tilde{w}_1^{(i)} = \frac{w_1^{(i)}}{\sum_{j=1}^M w_1^{(j)}}$.

   2.3 Resampling: Set $\tilde{\text{ESS}} = \frac{1}{\sum_{i=1}^M (\tilde{w}_1^{(i)})^2}$. If $\tilde{\text{ESS}} < th_M$, then sample $\tilde{x}_1^{(j)} = x_1^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_1^{(i)}$. Set $w_1^{(i)} = 1/M'$.

   Set $M = M'$ and $t = 2$.

3. At time $t > 1$,

   3.1 Propagation: Sample $x_t^{(i)} \sim f(x_t|x_{1:t-1}^{(i)};\boldsymbol{\vartheta})$.

   3.2 Updating: Set $x_{1:t}^{(i)} = (x_{1:t-1}^{(i)}, x_t^{(i)})$. Set $u_t^{(i)} = g(y_t|x_t^{(i)};\boldsymbol{\theta})$ (the incremental weight) and let $w_t^{(i)} = u_t^{(i)} w_{t-1}^{(i)}$. Normalize the the weights: $\tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^M w_t^{(j)}}$.

   3.3 Resampling: Set $\tilde{\text{ESS}} = \frac{1}{\sum_{i=1}^M (\tilde{w}_t^{(i)})^2}$. If $\tilde{\text{ESS}} < th_M$, then sample $\tilde{x}_{1:t}^{(j)} = x_{1:t}^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_t^{(i)}$. Set $w_t^{(i)} = 1/M'$.

   Set $M = M'$ and t to $t + 1$.

Parameter Estimation

The calculations leading up to Algorithm (8) were conducted under the assumption that the parameters $\boldsymbol{\varphi} = (\boldsymbol{\vartheta}, \boldsymbol{\theta})$ in the model (4.1), are known. If, however, they are not known, then maximum likelihood inference is concerned with maximizing the likelihood function $p(y_{1:t}|\boldsymbol{\varphi})$. The likelihood function of the state space model is given

by,

$$\underbrace{p(y_{1:t}|\boldsymbol{\varphi})}_{=:L(\boldsymbol{\varphi})} = \underbrace{p(y_1)}_{=:L_1} \prod_{t=2}^{T} \underbrace{p(y_t|y_{1:t-1};\boldsymbol{\varphi})}_{=:L_t(\boldsymbol{\varphi})} \tag{4.27}$$

where,

$$p(y_t|y_{1:t-1};\boldsymbol{\varphi}) = \int p(y_t|x_t, y_{1:t-1};\boldsymbol{\varphi}) p(x_t|y_{1:t-1};\boldsymbol{\varphi}) dx_t.$$

Due to the conditional independence of the data (y_t) given the latent states (x_t), $p(y_t|x_t, y_{1:t-1};\boldsymbol{\varphi}) = p(y_t|x_t;\boldsymbol{\varphi}) = g(y_t|x_t;\boldsymbol{\varphi})$, and the integral reduces to:

$$p(y_t|y_{1:t-1};\boldsymbol{\varphi}) = \int g(y_t|x_t;\boldsymbol{\varphi}) p(x_t|y_{1:t-1};\boldsymbol{\varphi}) dx_t.$$

In an ideal scenario, on would sample from $x_t^{(i)} \sim p(x_t|y_{1:t-1};\boldsymbol{\varphi}), i = 1, ..., M$, and perform the following approximation:

$$p(y_t|y_{1:t-1};\boldsymbol{\varphi}) \approx \hat{p}(y_t|y_{1:t-1};\boldsymbol{\varphi}) =: \frac{1}{M} \sum_{i=1}^{M} g(y_t|x_t^{(i)};\boldsymbol{\varphi}).$$

where,

$$\hat{p}(y_t|y_{1:t-1};\boldsymbol{\varphi}) \overset{\substack{a.s. \\ M \to \infty}}{\longrightarrow} p(y_t|y_{1:t-1};\boldsymbol{\varphi}),$$

by the Law of Large Numbers.

The distribution $p(x_t|y_{1:t-1};\boldsymbol{\varphi})$, however, is not usually accessible, so an instrumental distribution needs to be chosen. As mentioned in the previous section, for the bootstrap particle filter, the instrumental distribution is $f(x_t|x_{t-1};\boldsymbol{\vartheta})$, which, by (4.26) results in the following incremental weights:

$$u_t^{(i)} = g(y_t|x_t^{(i)};\boldsymbol{\theta}), i = 1, ..., M. \tag{4.28}$$

So, the incremental weights themselves, are the target of estimation. Hence, the approximation of the MLE, $\hat{\boldsymbol{\varphi}}_{MLE}$, would be achieved by:

$$\hat{\boldsymbol{\varphi}}_{MLE} = \underset{\boldsymbol{\varphi}}{\operatorname{argmax}}\ \hat{L}(\boldsymbol{\varphi})$$

$$\hat{L}(\boldsymbol{\varphi}) = \hat{L}_1(\boldsymbol{\varphi}) \textstyle\prod_{t=2}^{T} \hat{L}_t(\boldsymbol{\varphi}) \tag{4.29}$$

$$\hat{L}_t(\boldsymbol{\varphi}) = \tfrac{1}{M} \textstyle\sum_{i=1}^{M} u_t^{(i)}, t = 1, ..., T$$

where $u_t^{(i)}$ is given by (4.28) with $x_t^{(i)} \sim f(x_t|x_{t-1}^{(i)};\boldsymbol{\vartheta}), i = 1, .., M$. Observe that $\hat{L}, \hat{L}_t, t = 1, ..., T$ are approximations of the corresponding likelihoods defined in (4.27). The result is the procedure mentioned in Algorithm (9).

Resampling Techniques

Performing resampling could be summarized in selecting $M' \leq M$ number of particles $\tilde{x}^{(j)}$ with replacement among an original sample $\{x^{(i)}\}_{i=1}^{M}$, where $\mathbb{P}(\tilde{x}^{(j)} = x^{(i)}) = \tilde{w}_j$,

Algorithm 9 Parameter Estimation through the Bootstrap Particle Filter

Let $th_M < M$.
For $i = 1, ..., M$ and while $t < T$,

1. Initialization: Set $t = 0$.

   1.1 Propagation: Sample $x_0^{(i)} \sim \tilde{\pi}(x_0)$ from some prior or choose $x_0^{(i)}$ fixed.

   1.2 Updating: Set $\tilde{w}_0^{(i)} = 1/M$. Set t=1.

2. At time $t = 1$.

   2.1 Propagation: Sample $x_1^{(i)} \sim f(x_1|x_0^{(i)}; \boldsymbol{\vartheta}^*)$.

   2.2 Updating: Set $w_1^{(i)} = g(y_1|x_1^{(i)}; \boldsymbol{\theta}^*)$. Normalize the the weights: $\tilde{w}_1^{(i)} = \frac{w_1^{(i)}}{\sum_{j=1}^M w_1^{(j)}}$.

   2.3 Estimation: Set $\hat{L}_1 = \frac{1}{M} \sum_{i=1}^M w_1^{(i)}$.

   2.4 Resampling: Set $\tilde{\text{ESS}} = \frac{1}{\sum_{i=1}^M (\tilde{w}_1^{(i)})^2}$. If $\tilde{\text{ESS}} < th_M$, then sample $\tilde{x}_1^{(j)} = x_1^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_1^{(i)}$. $w_1^{(i)} = 1/M'$

   Set $M = M'$ and $t = 2$.

3. At time $t > 1$,

   3.1 Propagation: Sample $x_t^{(i)} \sim f(x_t|x_{1:t-1}^{(i)}; \boldsymbol{\vartheta}^*)$.

   3.2 Updating: Set $x_{1:t}^{(i)} = (x_{1:t-1}^{(i)}, x_t^{(i)})$. Set $u_t^{(i)} = g(y_t|x_t^{(i)}; \boldsymbol{\theta}^*)$ (the incremental weight) and let $w_t^{(i)} = u_t^{(i)} w_{t-1}^{(i)}$. Normalize the the weights: $\tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^M w_t^{(j)}}$.

   3.3 Estimation: Set $\hat{L}_t = \frac{1}{M} \sum_{i=1}^M u_t^{(i)}$.

   3.4 Resampling: Set $\tilde{\text{ESS}} = \frac{1}{\sum_{i=1}^M (\tilde{w}_t^{(i)})^2}$. If $\tilde{\text{ESS}} < th_M$, then sample $\tilde{x}_{1:t}^{(j)} = x_{1:t}^{(i)}, j = 1, ..., M'$ with probability $\tilde{w}_t^{(i)}$. Set $w_t^{(i)} = 1/M'$.

   Set $M = M'$ and t to $t + 1$.

Set $\hat{L} = \prod_{t=1}^T \hat{L}_t$. Then, $\hat{\boldsymbol{\varphi}}_{MLE} = \underset{\boldsymbol{\varphi}}{\text{argmax}}\ \hat{L}$.

and $\sum_{j=1}^{M} \tilde{w}_j = 1$. For simplicity, we choose $M' < M$. The line of thought in cases when $M' \neq M$ is very similar.

Choosing a new particle could be seen as an event within the framework of probability theory. It could be thought of letting the random variable of the index of the selected particle, I_j, being equal to the index of one of the particles in the sample $\{x^{(i)}\}_{i=1}^{M}$. In other words, $I_j = i$. Hol (2004, p. 13) summarizes an event generation algorithm in the following manner:

Algorithm 10 Event generation.

The generation of an event I_j with $\mathbb{P}(I_j = i) = \tilde{w}_j$ is obtained by

1. Simulate $u_j \sim \mathbb{U}[0, 1)$,
2. Assign $I_j = i$, if $S_{i-1} < u_j \leq S_i$, where $S_i = \sum_{s=1}^{i} \tilde{w}_s$, and $1 \leq i, j \leq M$.

The resampling methods used in this study are *stratified* and *systematic resampling*. They are both based on Algorithm (10) and are very similar. According to Hol (2004, pp. 18-19), an enhanced version of stratified resampling in one dimension could be seen as partitioning the index space, $[0, 1)$ into M disjoint strata of equal size $[\frac{j}{M}, \frac{j+1}{M})$ for $j = 0, ..., M - 1$ and then choosing one particle within each interval. Hence, the resampled particles are sampled independently of each other. The returned sequence would then be a random permutation of the stratified samples. In other words,

$$I_j = i_j \text{ where } i_j = \frac{\pi_{j-1} + u_j}{M}, u_j \sim \mathbb{U}[0, 1), j = 0, ..., M - 1$$

where π is a uniform random permutation of $0, ..., M - 1$.

In systematic resampling, however, the samples are no longer independent, and are positioned in the same manner in each strata relative to each other:

$$I_j = i_j \text{ where } i_j = \frac{\pi_{j-1} + u}{M}, u \sim \mathbb{U}[0, 1), j = 0, ..., M - 1$$

The lack of independence sometimes may lead to higher variance of the resampled index random variable compared to the stratified resampling method. However, systematic resampling produces samples that are more uniform and are to have minimal discrepancy (*ibid*). Both are suitable resampling techniques in one dimension.

Log Sum of Exponentials

When summing up the normalized weights $\tilde{w}_j, j = 1, ..., M$, the magnitude of the weights may differ, sometimes significantly, which numerically will result in the program (here MATLAB) evaluating the insignificant ones as zero due to software finite numerical representation capabilities, compared to the larger weights. Blevins (2008) argues that such numerical phenomenon may often result in underflow and overflow in the representation of the particles when resampling is carried out. In order to avoid this problem the *log sum exponential* approach is used.

Hence, the sum of the weights needs to be adjusted in a manner. Let $v = (a, b)$, then for any c, we can write:

$$\exp(a) + \exp(b) = (\exp(a - c) + \exp(b - c)) \exp(c)$$

which when taking the logarithm of both sides becomes:

$$\log(\exp(a) + \exp(b)) = \log(\exp(a - c) + \exp(b - c)) + c$$

Choosing c properly will reduce the incidence of overflow or underflow.

Now let $v = (v_1, ..., v_M)$, where $v_j = \log(\tilde{w}_j), j = 1, ..., M$. Furthermore, let $\|v\| = (|v_1|, ..., |v_M|)$. The version of the log sum exponential approach employed in this study distinguishes between two situations. In one case, the vector v contains negative values with very high absolute values, i.e. $\max_{v_j}\|v\| > \max_{v_j} v$. In other words, the some of the weights are really small compared to the larger weights. This increases the probability of underflow occurring. In this case, c is chosen as: $c = \min_{v_j} v$.

In the other case, it holds that $\max_{v_j}\|v\| = \max_{v_j} v$, which indicates that there are very large weights. In this case there is a possibility of overflow occurring. Hence, c is chosen as the following, $c = \max_{v_j} v$. The process boils down to what is presented in Algorithm (11). The method however, does not guarantee the absence of overflow and underflow, but merely reduces their frequency of occurrence.

Algorithm 11 Log Sum Exponential Approach.

Let $v = (v_1, ..., v_M)$, where $v_j = \log(\tilde{w}_j), j = 1, ..., M$. Furthermore, let $\|v\| = (|v_1|, ..., |v_M|)$. Furthermore, let $S = \sum_{j=1}^{M} \tilde{w}_j$.

1. If $\max_{v_j}\|v\| > \max_{v_j} v$, set $c = \min_{v_j} v$. Otherwise, set $c = \max_{v_j} v$.
2. Calculate $\hat{\log(S)} = \log\left(\sum_{j=1}^{M} \exp(v_j - c)\right) + c$,

where $\hat{\log(S)}$ is numerically closer to the actual value of $\log(S)$.

4.2 A Practical Example

As a practical example the process $\boldsymbol{Z}$ in chapter (3) was chosen as the hidden process and some Gaussian noise was added to arrive at the observed states $Y_{t_0:t_j}$. In other words,

$$Y_{t_j} = Z_{t_j} + \varepsilon_j, \tag{4.30}$$

where $\varepsilon_j \sim N(0, \gamma^2)$ i.i.d. and Z_{t_j} is as defined in equation (3.5). The parameter space is then increased by one dimension:

$$\boldsymbol{\varphi} = (\ln(A), B, C, \sigma, \gamma) \tag{4.31}$$

where A, B, C are the parameters of the now hidden Gompertz model in (3.1), σ is the diffusion parameter in (3.3), and γ is the standard deviation of the observational noise.

As in the example with dynamical models in the previous chapter, a distribution is assigned to the first latent state. Otherwise the likelihood function does not depend on $\ln(A)$. The choice of this distribution is identical to the one in (3.6), where σ_0^2 is again a meta-parameter and assumed to be known. The practical interpretation of this parameter is the same as before.

For our example we chose the following time points and parameters:[2]

$$\begin{aligned} \boldsymbol{t} &= (0, 2, 3, 3.5, 7, 11, 20, 24, 25, 26, 30, 33, 34, 40), \\ \boldsymbol{\varphi} &= (\ln(A), B, C, \sigma, \gamma) = (8, 5, 0.2, 0.01, 0.15), \\ \sigma_0 &= 0.02. \end{aligned}$$

The time is expressed in weeks. In other words, $\exp(Z_t)$ is the *true* weight of the chicken at week t, and $\exp(Y_t)$ is the *observed* weight of the same chicken at the same time point. The observations in $\boldsymbol{Z}$ were then simulated according to equations (3.6) and (3.4), while $\boldsymbol{Z}$ is simulated in accordance with (4.30). One such simulation is shown in Figure (4.1), where both the latent and observed processes are mentioned.

In order to find the MLE, the negative logarithm of the likelihood function in (4.29), attained through the bootstrap particle filter mentioned in Algorithm (9), was minimized. Resampling was performed via both stratefied and systematic methods. The settings used for the bootstrap particle filter were the following:

$$\begin{cases} M' = M = 1000 \\ th_M = M/2 \\ x_0^{(i)} = \ln(A) - B \\ x_{t_1}^{(i)} | x_0^{(i)} \sim N(x_0^{(i)}, \sigma_0^2) \end{cases} \tag{4.32}$$

where $\sigma_0 = 0.02$ is the same meta-parameter in the initialization of the process and $i = 1, .., M$.

The minimization could have been done through the use the MATLAB function `fmincon`, which performs constrained nonlinear optimization. However, the Hessian provided by this function was unstable. Instead the MATLAB function `fminsearch` was utilized, which finds minimum of an unconstrained multivariable function using derivative-free methods, another form of unconstrained nonlinear optimization.

This function, however, does not provide an inbuilt Hessian. The variance-covariance matrix was instead attained indirectly though the following scheme (Seber & Wild, 2003, pp. 42-48):

$$\operatorname{cov}(\boldsymbol{\varphi}_{MLE} | y) = s^2 (J'J)^{-1}$$

where J is the Jacobian of at the MLE, and s^2 is the unbiased estimate of the variance of the errors, i.e. the difference between the observations and the predicted values. The Jacobian was estimated using the function `jacobianest` provided by D'Errico (2014). It is worth mentioning that even after performing the estimation of the variance-covariance matrix of the MLE in this manner, the stability of said matrix was not completely optimal. Higher values of standard error of the estimates obtained via the bootstrap particle filter, compared to the the ones obtained through data cloning, could also be explained partially by the fact that resampled trajectories become dependent of each other and therefore the variance of the estimates increase.

[2]It is worth commenting why the choice of the time points in this section, differs from those used in the dynamic models example without observational noise. As the rest of the study will mention, using the former time-points the parameter σ, was declining towards zero through the simulations. The intuition was, that this may very well be due to the constant time lapses between the time points in the dynamic models example. Hence, the time lapses was set in such a manner, so that the intervals between them were not constant. The results seemed, however, to be invariant to the choice of the sample intervals, as will be mentioned later on.

As before, when performing the data-cloning procedure one needs to choose prior distributions for the unknown parameters. Here, the following were chosen,

$$\ln(A) \sim \mathbb{U}(\ln(500), \ln(5000))$$
$$B, C, \sigma, \gamma \sim \mathbb{U}(0, 10)$$

which are fairly uninformative. The number of iterations for each K number of clones was ten thousand regular plus ten thousand burn-in. The covariance of the proposal function was updated every 100^{th} step. The starting values in the simulations are the true values of the parameters, not to be confused with the true MLE.

A problem that arose when dealing with the state space model in this chapter, was that the acceptance rate, at first, dramatically decreased to about 5%. To relive this issue, the stability parameter s_d in the covariance update scheme mentioned in (1.2) and (1.3), was chosen as 0.4 instead of the theoretically motivated value of $2.4^2/d$, where d is the dimension of the space of the unknown parameters.

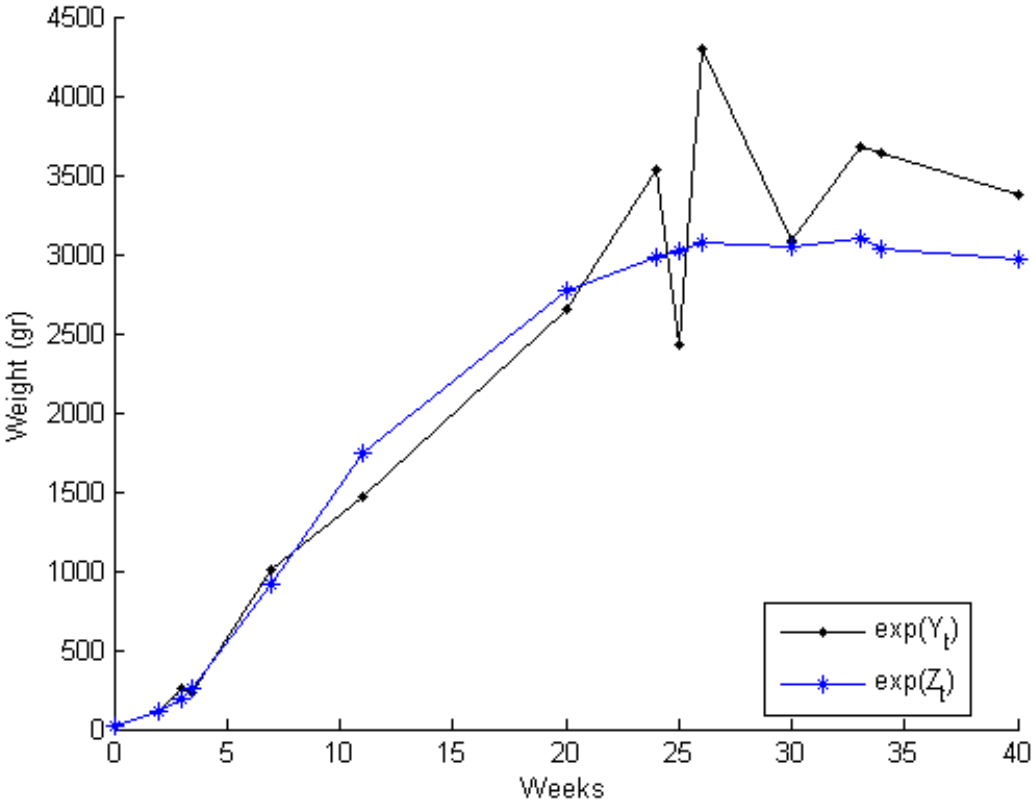

Figure 4.1: A simulation of a hidden stochastic dynamic model based on the Gompertz model ($\exp(Z_t)$) together with the observed state ($\exp(Y_t)$).

Decreasing s_d makes the the Metropolis update scheme susceptible to producing near singular covariances. Here, the MIC method showed a significant advantage over the CIC one, rarely falling into that trap. For the CIC method, this instability was much more frequently observed, stable cases being almost an exception.

Using the MCMC method via data cloning the following simulations were attained (Figures 4.2 and 7.27). The simulations via both the MIC and the CIC methods prove

to converge closely to the approximations of the MLE via the bootstrap particle filter, and approach it as the number of copies is increased. This is true in both cases of the resampling technique being stratified and systematic.

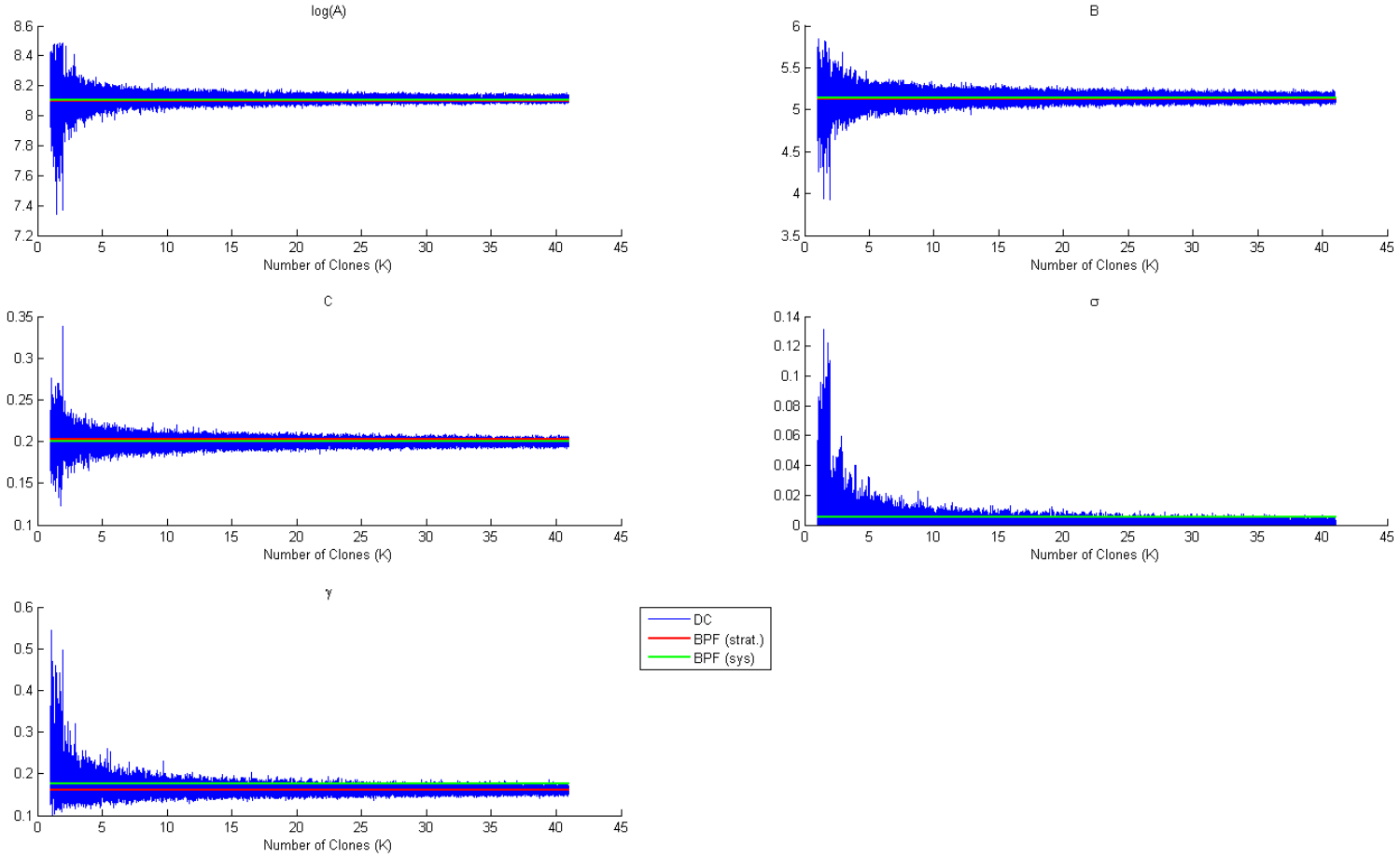

Figure 4.2: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 iterations, applying the MIC method to the state space model. The red and green lines are the aprroximative MLEs obtained via SMC via SISR, where the resampling techniques are stratified and systematic respectively. The starting values in the simulation are the true values of parameter, not to be confused with the true MLE.

Figure (4.3) shows the estimates of the parameters through the MIC method against the number of clones, together with the approximations of the MLE through the bootstrap particle filter. The values are close and both approximations of the MLE through the bootstrap method are within the interval of one standard error of the estimates.

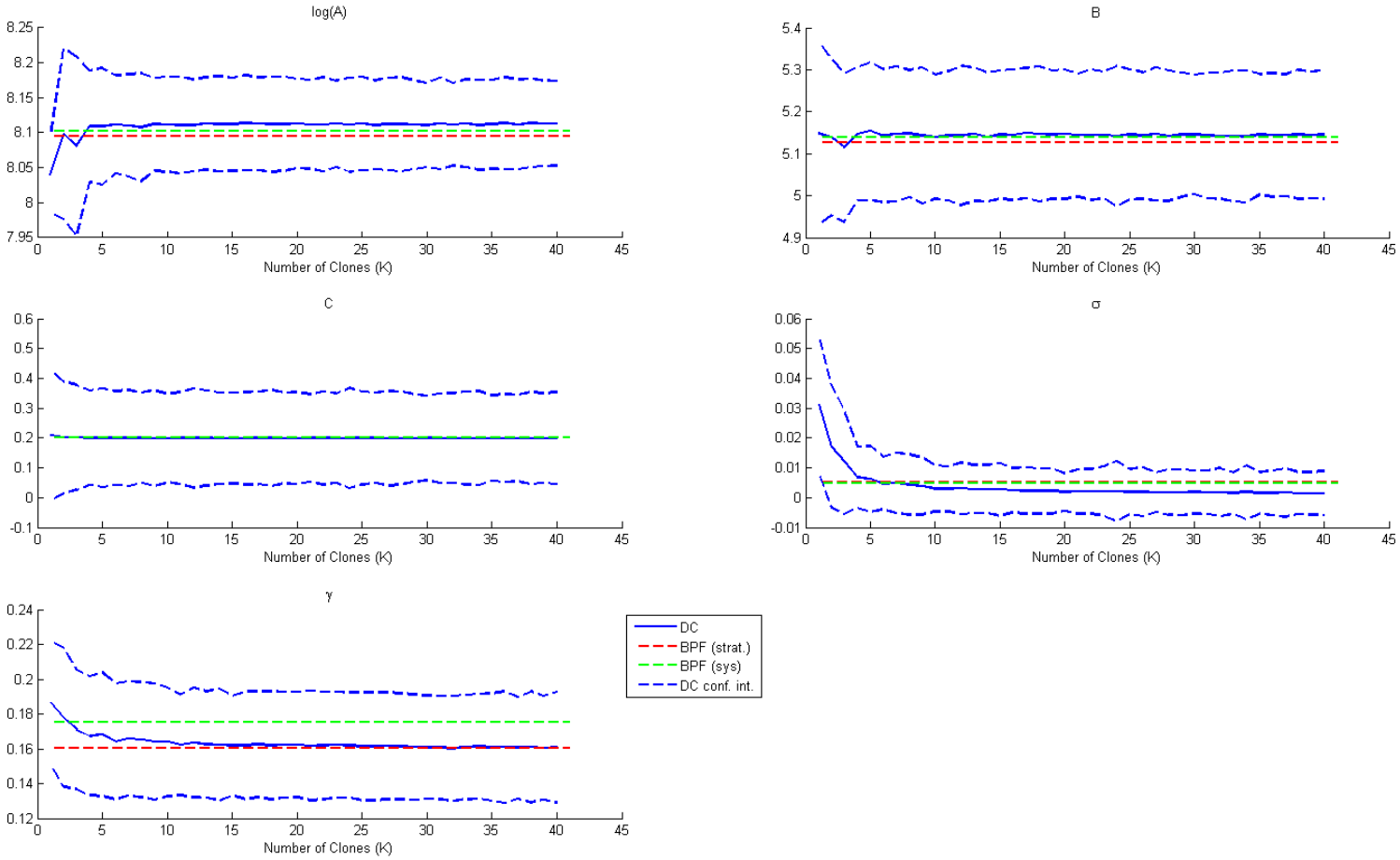

Figure 4.3: Estimates of the the MLE via the data cloning approach with increasing number of copies (by one, starting at one) every 10^4 iterations, applying the MIC method to the Gompertz model. The red and the green dashed lines are the approximative MLE via, respectively, the stratified and systematic resampling techniques of the bootstrap particle filter. The dashed blue lines are the data cloning estimate plus and minus one standard error (about 70% of confidence under the assumption of normality).

One of the estimates however, proves to be a bit peculiar. The estimate of σ seems to approach zero. This is also observed in Figure (4.2). It seems as if the data cloning method strives to put this value to zero. This cannot be a result of the time intervals of the simulation being constant due to the choice of the time points of the simulation. It may very well be that the low number of observations (14 in total) are too few to make a good estimation of this value.[3]

[3] To confirm this further, a new simulation was done where the parameter σ, was replaced with the parameter $\ln(\sigma)$, and prior distribution $\ln(\sigma) \sim \mathbb{U}[-10^6, 10]$. During those simulations, $\ln(\sigma)$ as approaches the minimum value of the prior, the simulation suddenly deteriorates. This due to the fact that, the exponential transformations of $\ln(\sigma)$ is necessary to attain the standard deviation for use in calculation of the posterior. However, small enough values of $\ln(\sigma)$, will numerically yield zero

	$\ln(A)$	B	C	σ	γ
True Val.	8	5	0.2	0.01	0.15
App. MLE					
Strat.	8.0955 (0.0846)	5.1264 (0.1900)	0.2029 (1.3331·1e-2)	0.5075·1e-2 (1.5789·1e-2)	0.1603 (7.2880·1e-2)
Sys.	8.1020 (0.1740)	5.1386 (0.3368)	0.2005 (3.7918·1e-2)	0.4879·1e-2 (2.5139·1e-2)	0.1750 (16.4241·1e-2)
CIC					
$K = 1$	8.0108 (0.2016)	5.0591 (0.2490)	0.2072 (2.0915·1e-2)	3.5305·1e-2 (2.1042·1e-2)	0.1844 (3.4294·1e-2)
$K = 5$	8.1111 (0.0751)	5.1466 (0.1611)	0.1999 (1.5151·1e-2)	0.6070·1e-2 (1.1142·1e-2)	0.1668 (3.1389·1e-2)
$K = 10$	8.1123 (0.0678)	5.1421 (0.1548)	0.1994 (1.4408·1e-2)	0.3393·1e-2 (0.8095·1e-2)	0.1634 (3.2647·1e-2)
$K = 40$	8.1123 (0.0662)	5.1446 (0.1437)	0.1994 (1.3247·1e-2)	0.1585·1e-2 (0.7031·1e-2)	0.1608 (3.0224·1e-2)
MIC					
$K = 1$	8.0406 (0.0550)	5.1477 (0.2177)	0.2091 (1.3002·1e-2)	3.1297·1e-2 (2.2905·1e-2)	0.1864 (3.5659·1e-2)
$K = 5$	8.1090 (0.0836)	5.1540 (0.1653)	0.2002 (1.7005·1e-2)	0.6223·1e-2 (1.1102·1e-2)	0.1683 (3.5479·1e-2)
$K = 10$	8.1117 (0.0678)	5.1399 (0.1479)	0.1991 (1.3769·1e-2)	0.3052·1e-2 (0.7670·1e-2)	0.1639 (3.1280·1e-2)
$K = 40$	8.1133 (0.0603)	5.1456 (0.1540)	0.1992 (1.3159·1e-2)	0.1488·1e-2 (0.7412·1e-2)	0.1609 (3.1682·1e-2)

Table 4.1: Summary of the estimates of the state space model: the true values; the approximative MLE through the bootstrap particle filter with the conditions mentioned in (4.32) (both via stratified and systematic resampling); and the estimates via the MIC and CIC methods of data cloning (DC). The standard errors are mentioned in the parenthesis.

Some of the properties of the CIC and MIC method applied to the state space model in question are summarized in Figure (4.4). We see that the estimates via the data cloning method converge to the approximations of the MLE via the bootstrap particle filter as the number of clones (K) increase (Figures a,b). Furthermore, both stabilize with respect to the acceptance rate (Figure c). The acceptance rates indeed starts at a very low value (around 10%) but stabilize around the theoretically motivated values of 23%.

Moreover, the ω- and $\tilde{r}^2$-statistics are close to zero and even converge to that value as the number of clones K is increased (d,e). With respect to these statistics, the MIC method proves to b less volatile. It has already been mentioned that the MIC method is more stable when decreasing the values of the stability parameter (s_d) in the covariance update scheme. These to issues may be related.

In addition, the largest standardized eigenvalue (λ_K^S) approaches zero as the number of clones increase. However, the rate of convergence is faster than $1/K$. This may, as mentioned before, be due to the additional information provided by continuously increasing the number of clones, rather than running each simulation separately for each number of copies. The convergence is, nevertheless, correct as the diagnostic statistics are close to zero. Increasing the number of clones sporadically does not seem to have any significant effect on the convergence of the chain either (Figure (7.28)). Hence, in order to reduce computational time, the latter should be preferred.

Table (4.1) provides a summary of the approximations and estimates of the MLE. The estimates provided through the bootstrap particle filter are generally close to those calculated through the data cloning approach. In the standard errors, however,

when the exponential is taken. This renders $\ln(\sigma)$ irrelevant after it has become small enough.

Another interesting observation was that, in such a setting, the standardized largest eigenvalue (λ_K^S) of the covariance matrix, does not converge to zero. This result would, according to Lele (2010) be an indication of the non-estimability of the parameters. However, this is not the case here, as this is merely a numerical problem. Therefore, the study of the convergence pattern of the standardized largest eigenvalue (λ_K^S) as a test of estimability of the parameters should be performed with caution, excluding the possibility of numerical issues.

there is a discrepant tendency. This is especially notable in the standard errors of the σ and γ parameters. This may very well be due to the difficulty of estimating σ with few observations, that is also manifested in the bootstrap method.

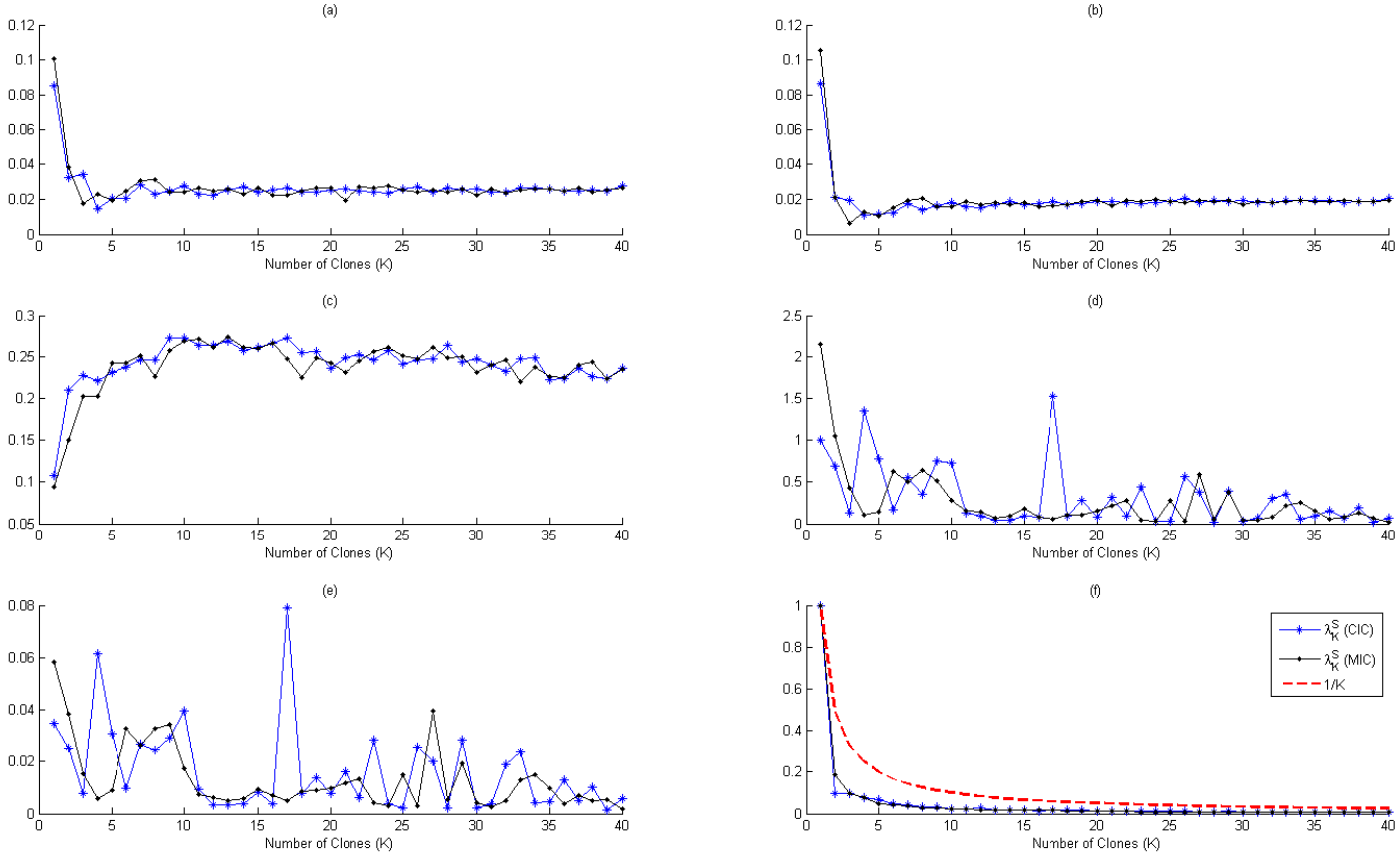

Figure 4.4: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE via data-cloning compared with the approximative MLE obtained through the bootstrap particle filter with stratified resampling (the 2-norm of the difference), (b) the same as (a) but with systematic resampling (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

As seen in Figure (4.5), increasing the number clones improves the normality of the pseudo-posterior for the γ parameter. This is true for all the other parameters except for σ (Figure (7.32)).[4] The quantile tail is heavier around zero for that parameter which strengthens the previous line of argument that the data cloning method tends to put that parameter close to zero. Hence, the confidence interval for σ presented in Figure (4.3) should be viewed with caution.

[4]For the QQ-plots of the other parameters, see section (7.3).

Observe that, the practical issues mentioned in section (2.3) is also applicable for Bayesian estimation of the MLE via data cloning in the case of the type of models of this chapter, i.e. hidden Markov models.

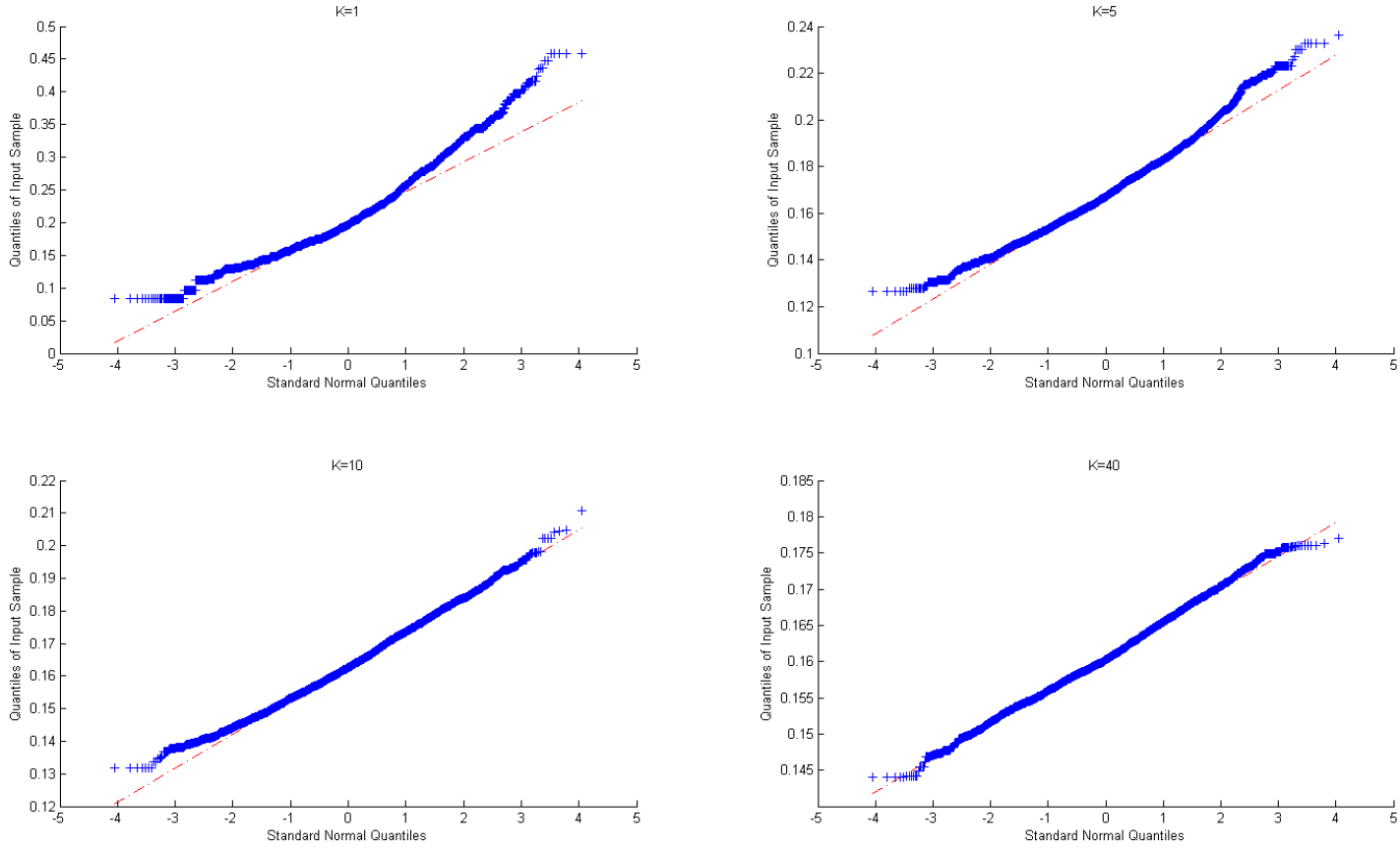

Figure 4.5: Quantile-quantile plots for the simulations of γ using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

Chapter 5

Concluding Remarks and Future Studies

As a student of Monte Carlo methods, when this author was introduced to the subject, the initial impression was one of surprise. Too many topics of the subject matter resembled the strive to put out continuously erupting fires.

While the MCMC methods were theoretically attractive, the use of blind simulations was not always successful even with the use of informative priors in a Bayesian setting.[1] That fire was then put out by using smarter simulations for instance through the use of SMC methods. However, the estimation power and rate of convergence achieved through the SMC methods, is too often too dependent on the choice of the instrumental distribution. Hence, importance sampling was introduced.

Nevertheless, importance sampling suffered from its own numerical problems, namely the weight degeneracy problem and the curse of dimensionality. Too often the weights, being too small in comparison to some other particles in the chain, were numerically equated with zero, rendering few particles carrying the whole weight of the chain. To relieve this issue, different approaches were introduced such as antithetic sampling and resampling techniques, which in turn had issues of their own. For instance, there were a wide range of resampling techniques to be chosen, no one universally successful.

The data cloning approach takes the researcher back to the initial fire, the low success rate of blind simulations in MCMC methods, and provides a simple tool to combat that problem, namely copying the data. The method is surprisingly successful, and improves the accuracy of the estimates when approximating the MLE.

The data cloning procedure have clear advantages over ordinary maximum likelihood estimation. It avoids analytic and numerical evaluation of high-dimensional integrals. Moreover, it avoids numerical optimization of a function, and numerical computation of the curvature of the likelihood function. Of course not all of the tuning issues related to MCMC methods can be avoided and the convergence rate should be monitored.

The data cloning approach has also several advantages over the SMC methods. First and foremost, it is significantly simpler to implement. Furthermore, it comparatively raises fewer numerical issues to deal with. Moreover, data-cloning, as any MCMC approach, is done through blind simulations, which means that the choice of the proposal function becomes simpler. Most importantly, it avoids numerical opti-

[1] Blind simulations refer to simulations that do not depend on the observed data, as in the MCMC methods.

mization of a function.

The data cloning procedure has other benefits in its own rights. Firstly, the convergence to the true MLE as the number of clones increases, is invariant to the choice of the prior distribution. This was mentioned in section (1.2), where it was proven that the pseudo-posterior of the K times cloned data was degenerate at the true MLE. Furthermore the approximative normality of the estimates, provides a convenient way of producing confidence intervals.

The data cloning method is also accompanied by several diagnostic tools, namely the ω, $\tilde{r}^2$ and the standardized largest eigenvalues of the variance-covariance matrix λ_K^S. The first two ones are to be close to, or better yet converge to, zero if the estimation process is successful. The standardized largest eigenvalues of the variance-covariance matrix λ_K^S is also supposed to converge to zero if the estimation process is successful. Failure to do so could be a symptom of the non-estimability of the parameters. The conclusion of non-estimability should, however, be drawn with caution, as failure to converge could be a result of numerical transformations.[2]

The data-cloning method, however, has some disadvantages. A prominent one appears when dealing with very large data sets, especially when dealing with state space models. In such cases, the simulation of the latent process (in step 3 in Algorithm (4)) may be very inaccurate and too far away from the true latent state. The reason for this phenomenon is the broad spectrum of possible trajectories. In such cases SMC methods are more suitable, since the proposals depend on the observations, and are not obtained blindly as in the data cloning method.[3]

The two data cloning procedures MIC and CIC, seem to function largely similarly. However, as the complexity of the problem was increased in the state space model (see section (4.2)), the MIC method proved to be more stable. This could be explained by the fact that, in the MIC method as the mean is taken after each block of iterations for each fixed number of clones K, the chain is drawn to a point with higher likelihood. In the CIC method, however, the chain may be stuck in the tail producing singular covariance updates in update of the variance-covariance matrix of the random walk proposal mechanism.

For future studies, one may want to study how the data cloning approach handles even more complex scenarios. Furthermore, combining the data cloning approach with other estimations techniques would be of particular interest. Moreover, a rigorous assessment of the theoretical grounds of the data cloning method is of utmost importance.

All in all, the data cloning approach seems to have a lot of potential. It increases the numerical accuracy of the estimates of the MLE significantly, while adding almost no further complexity to the simple MCMC process in a Bayesian setting. Considering further its relative simplicity to the SMC methods and its tuning issues, it could help relieve many researchers of considerable burdens when performing statistical estimations.

[2]See footnote (3) in section (4.2).

[3]One should make a distinction, however, between two different kinds of large data sets: *long* data sets and *rich* data sets. Long data sets contain a large number of data points corresponding to a large interval over a certain dimension, say time. Rich data sets, on the other hand, contain also a large number of data points, but over a short interval. The data-cloning method is very suitable for rich data sets, but performs poorly for long ones. SMC methods are more suitable for long data sets.

Chapter 6

References

Blevins, Jason (2008), "Calculating the Log Sum of Exponentials," [Online] Available from: http://jblevins.org/log/log-sum-exp. [Accessed: 1^{st} June 2015].

Cosma, Iona A. & Evers, Ludger (2010) "Markov Chains and Monte Carlo Methods," [Online] AIMS. Available from: http://users.aims.ac.za/ ioana/notes.pdf/. [Accessed: 12^{th} May 2015].

D'Errico, J. (2014). "Adaptive Robust Numerical Differentiation," [Online] MathWorks. Available from: http://www.mathworks.com/matlabcentral/fileexchange/13490-adaptive-robust-numerical-differentiation. [Accessed: 4^{th} June 2015].

Ditlevsen, S. and Samson, A. (2013) "Introduction to Stochastic Models in Biology." In Bachar M., Batzel J. J. and Ditlevsen, S. (eds.). *Stochastic Mathematical Models with Applications to Neuronal Modeling*. Berlin Heidelberg, Springer-Verlag.

Doucet, Arnaud; Godsill, Simon; & Andriue, Christophe (2000) "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, pp. 197-208.

Haario, Heikki; Saksman, Eero, & Tamminen Johanna (2001), "An adaptive Metropolis algorithm", *Bernoulli*, vol. 7, no. 2, pp. 223-242.

Hastings, W. K. (1970) "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, vol. 57, no. 1., pp. 97-109.

Hol, Jeroen D. (2004) "Resampling in particle filters" [Online] Linköping: Department of Electrical Engineering, Linköping University. Available from: http://www.control.isy.liu.se/student/exjobb/xfiles/0283.pdf. [Accessed: 26^{th} May 2015].

Jacquier, Eric; Johannes, Michael, & Polson, Nicholas (2007) "MCMC maximum likelihood for latent state models," *Journal of Econometrics*, vol. 137, pp. 615-640.

Lele, Subhash R.; Dennis, Brian, & Lutscher, Fritjof (2007), "Data cloning: easy maximum likelihood estimation for complex ecological models using Bayesian Markov chain Monte Carlo methods," *Ecology Letters*, vol. 10, pp. 551-563.

Lele, Subhash R.; Nadeem, Khurram, & Schmuland, Byron (2010), "Estimability and Likelihood Inference for Generalized Linear Mixed Models Using Data Cloning," *Journal of the American Statistical Association*, vol. 105, no. 492, December, pp. 1617-1625.

Liu, Jun S.; & Chen, Rong (1998, "Srquential Monte Carlo Methods for Dynamic Systems," *Journal of the American Statistical Association,* vol. 93, no. 443, pp. 1032-1044.

Kong, Augustine; Liu, Jun S., & Wong, Hung Wing (1994) "Sequential Imputations and Bayesian Missing Data Problems," *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278-288.

Roberts, G. O.; Gelman, A., & Gilks, W. R. (1997), "Weak Convergence and optimal Scaling of Random Walk Metropolis Algorithms", *The Annals of Applied Probability*, vol. 7, no. 1, pp. 110-120.

Seber, G. A. F.; & Wild, C. J. (2003), *Nonlinear Regression*. Wiley series in probability and statistics. New Jersey, John Wiley & Sons Inc.

Wilkinson D. J. (2012), *Stochastic Modeling for Systems Biology*. 2^{nd} Ed. Boca Raton, CRC Press.

Withange, Norishan N. K. P. (2013), "Methods and Applications in the Analysis of Correlated Non-Gaussian Data," [Online] University of Calgary, Alberta. Available from: http://theses.ucalgary.ca/bitstream/11023/848/2/ucalgary_2013_withanage_niroshan.pdf/. [Accessed: 5^{th} May 2015]

Chapter 7

Appendix - Omitted Figures

7.1 Static models

7.1.1 Simple Linear Regression

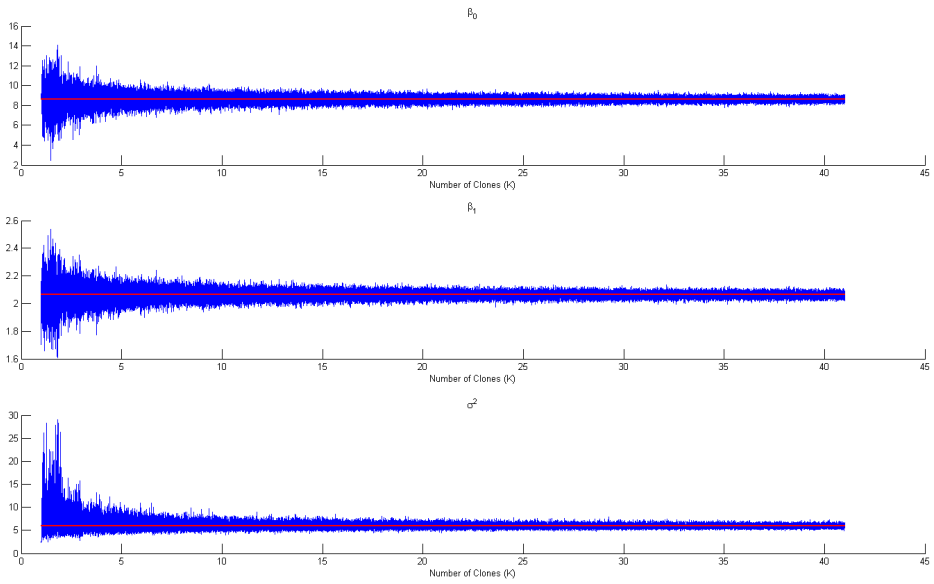

Figure 7.1: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 iterations, applying the CIC method to the simple linear regression model in (2.3). The red line is the true MLE.

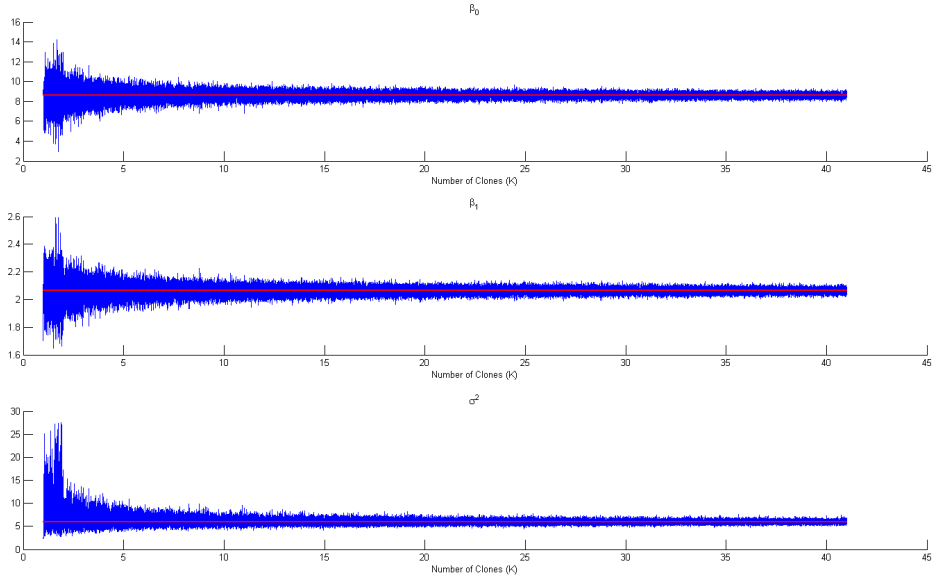

Figure 7.2: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 iterations (excluding a thousand iterations of burn-in), applying the MIC method to the simple linear regression model in (2.3). The red line is the true MLE.

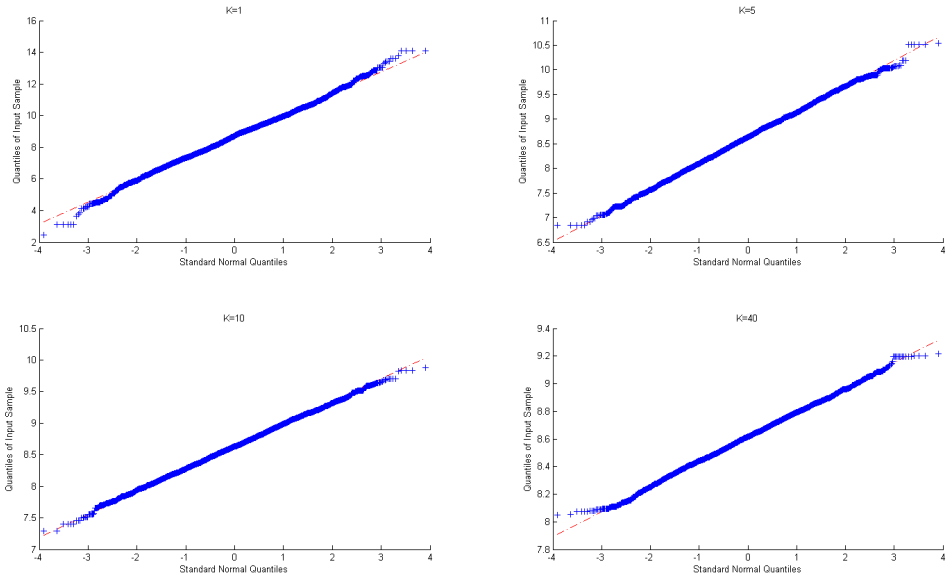

Figure 7.3: Quantile-quantile plots for the simulations of β_0 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

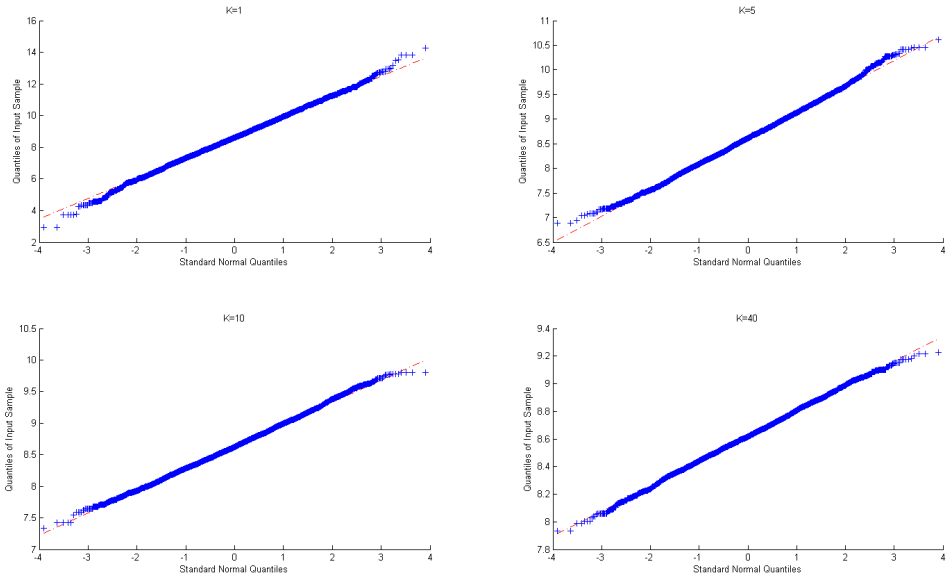

Figure 7.4: Quantile-quantile plots for the simulations of β_0 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

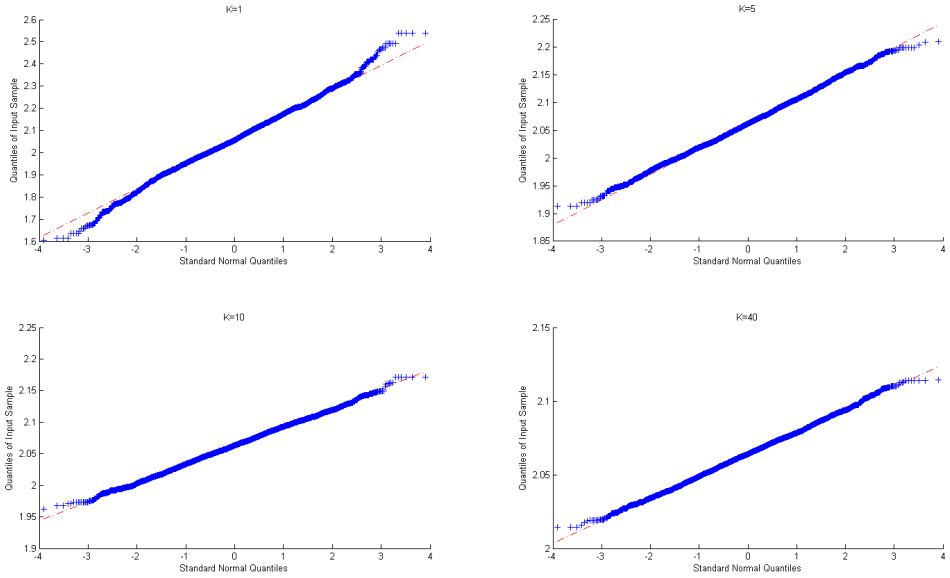

Figure 7.7: Quantile-quantile plots for the simulations of β_1 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

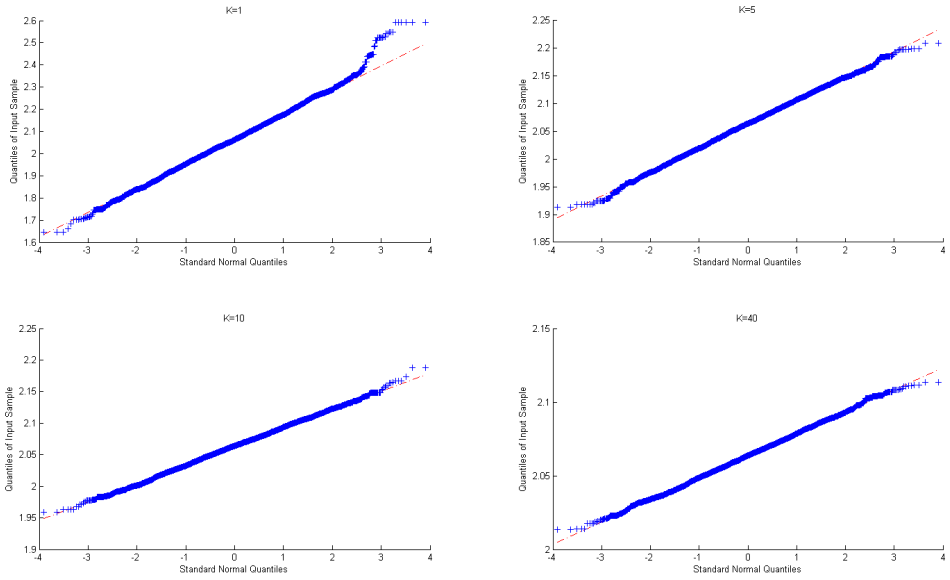

Figure 7.5: Quantile-quantile plots for the simulations of β_1 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

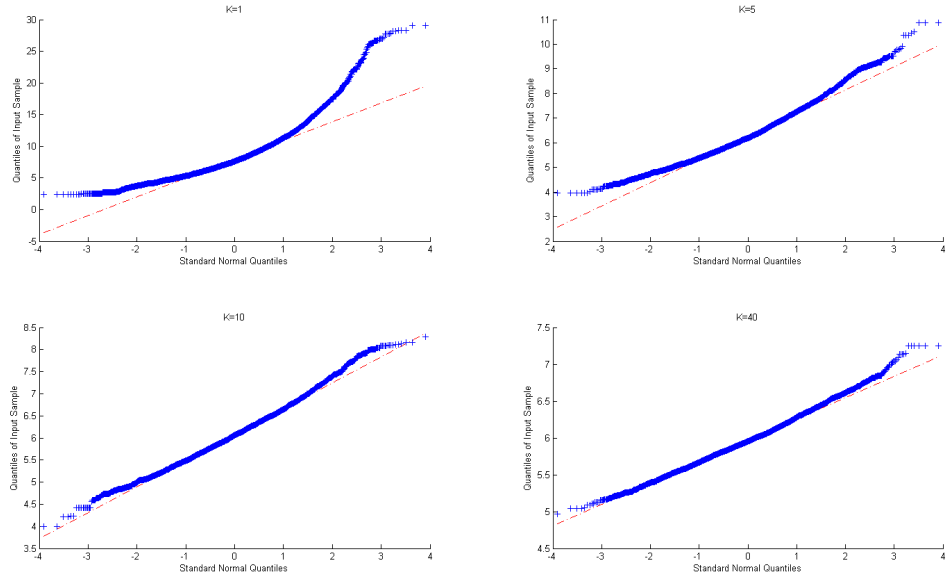

Figure 7.8: Quantile-quantile plots for the simulations of σ^2 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

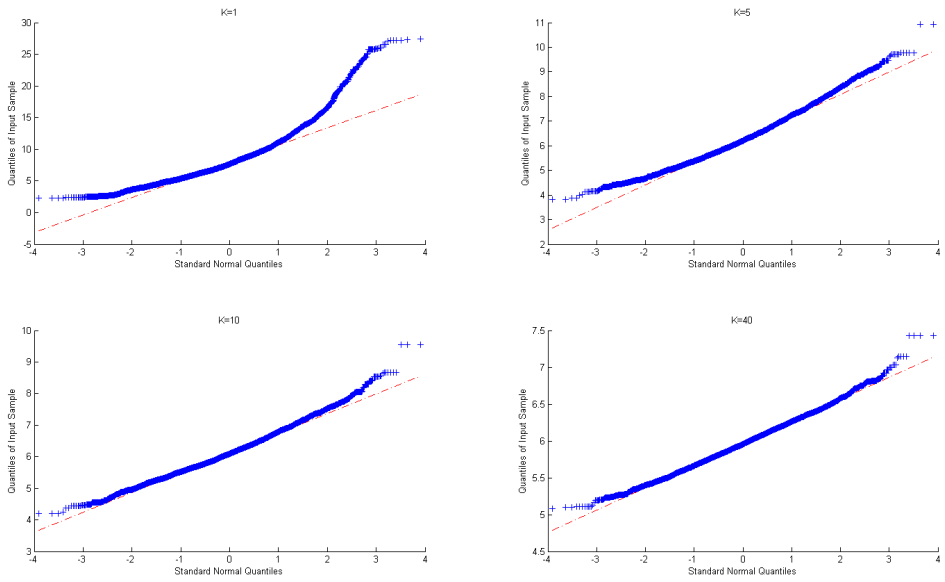

Figure 7.6: Quantile-quantile plots for the simulations of σ^2 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

7.1.2 Multiple Linear Regression

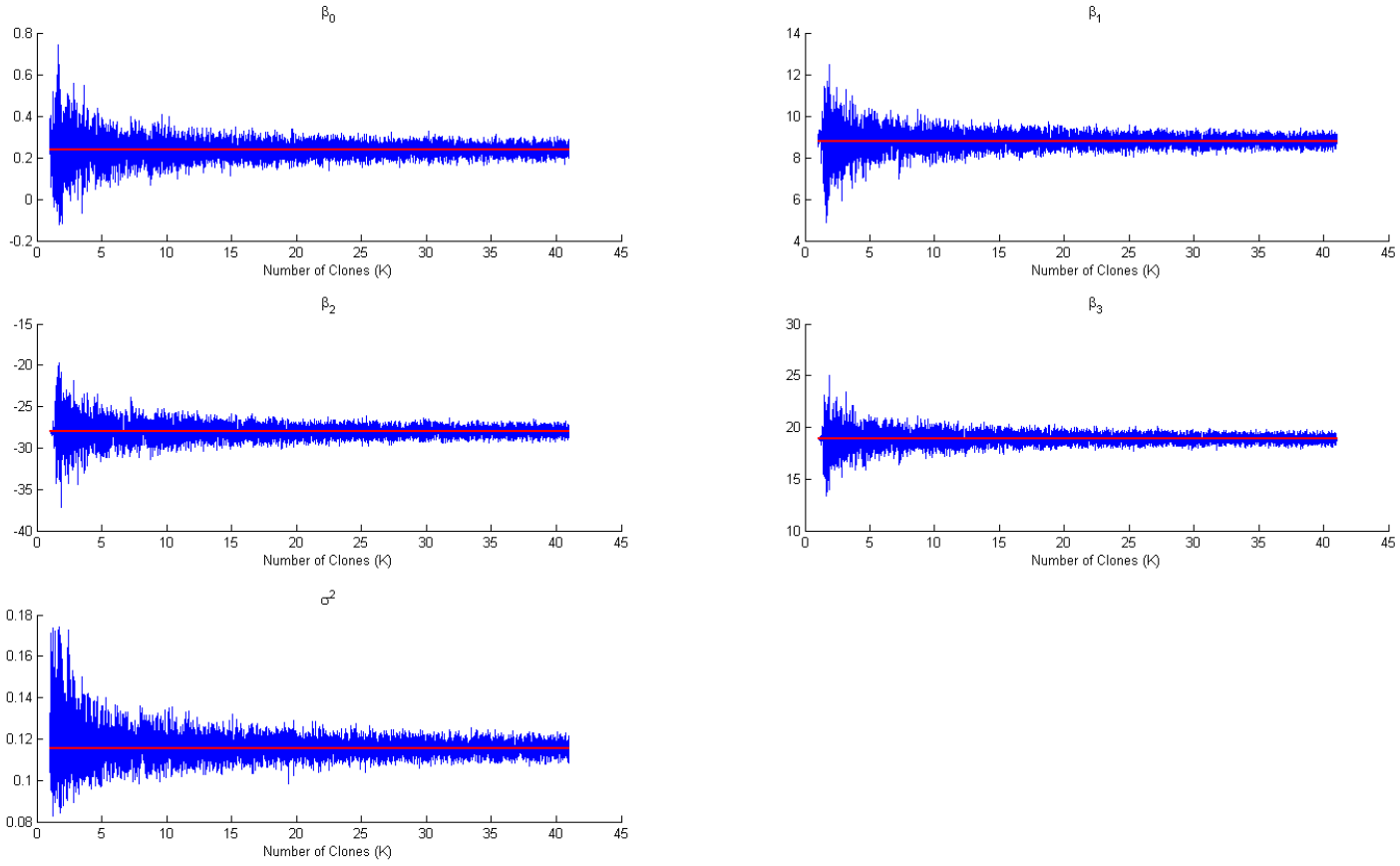

Figure 7.9: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 iterations (excluding a thousand iterations of burn-in), applying the MIC method to the multiple linear regression model in (2.2). The red line is the true MLE.

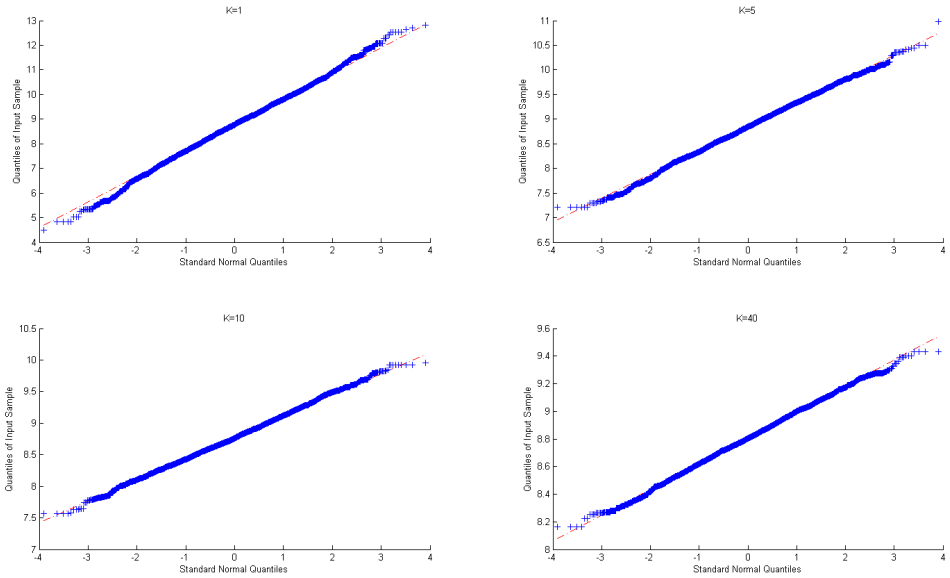

Figure 7.10: Quantile-quantile plots for the simulations of β_1 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

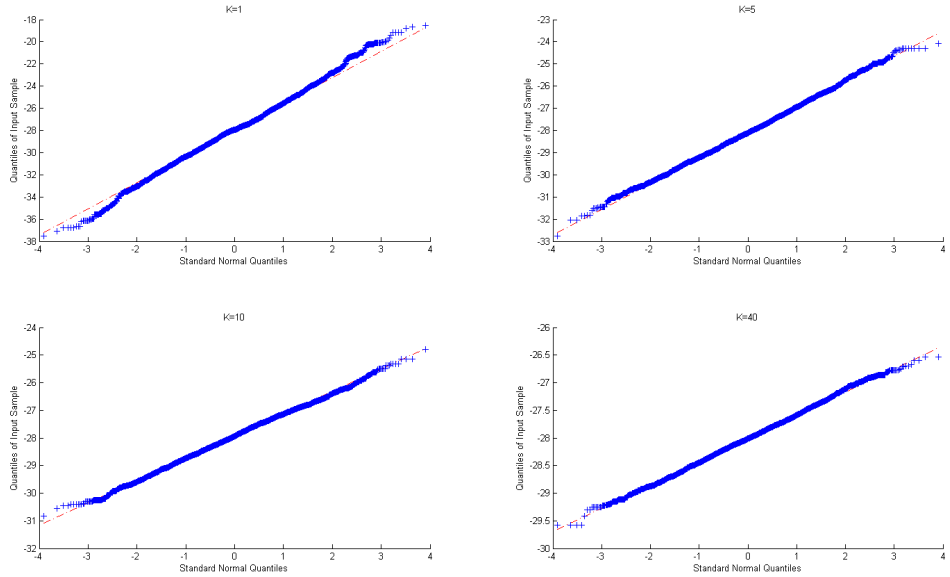

Figure 7.11: Quantile-quantile plots for the simulations of β_2 using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

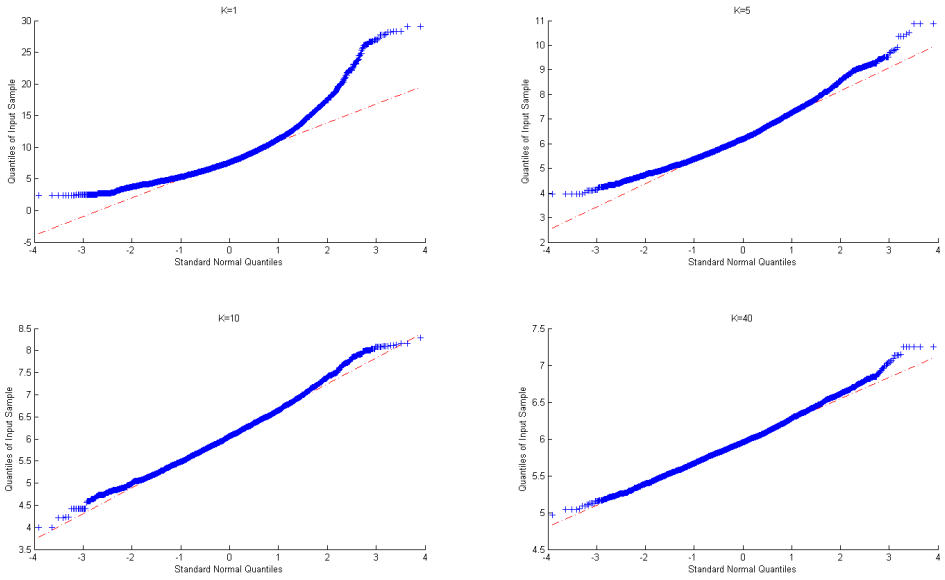

Figure 7.12: Quantile-quantile plots for the simulations of σ^2 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

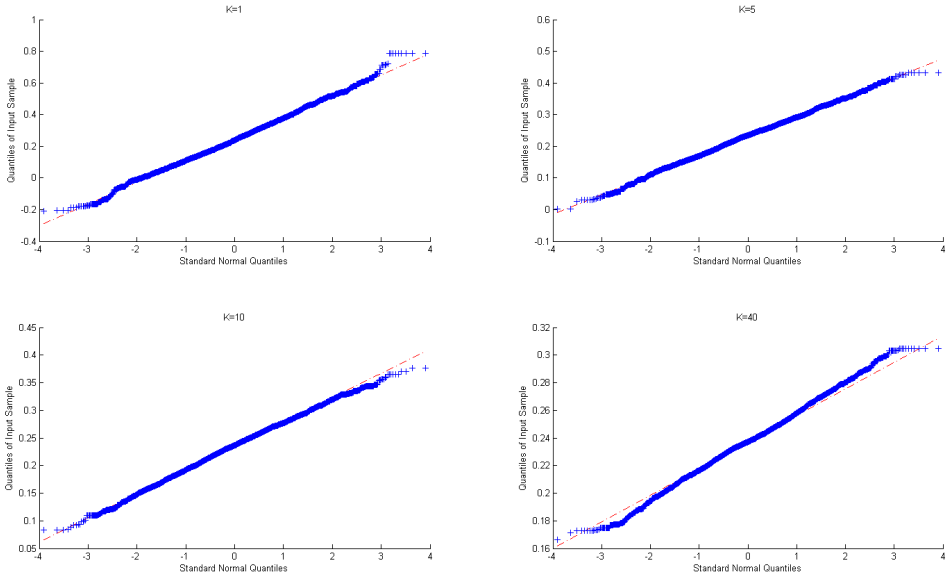

Figure 7.13: Quantile-quantile plots for the simulations of β_0 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

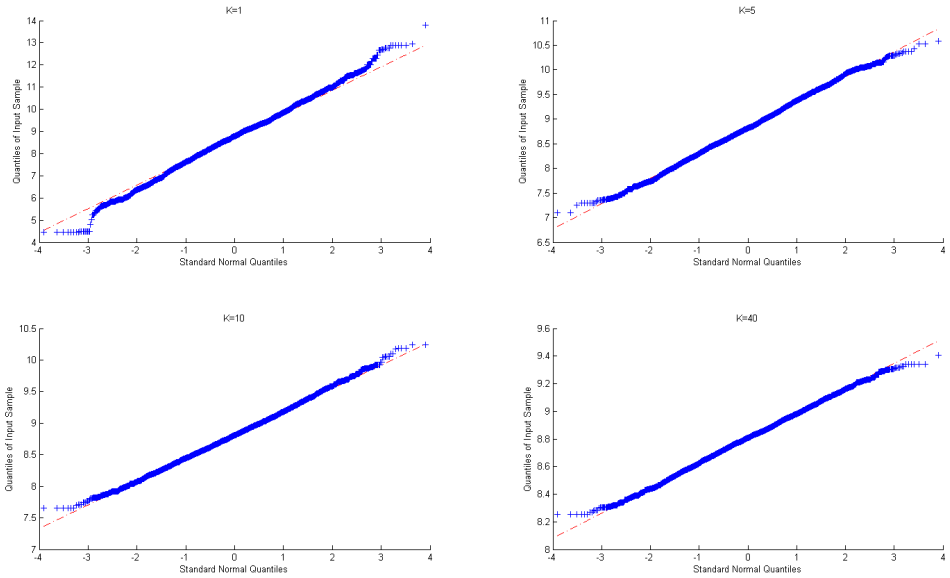

Figure 7.14: Quantile-quantile plots for the simulations of β_1 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

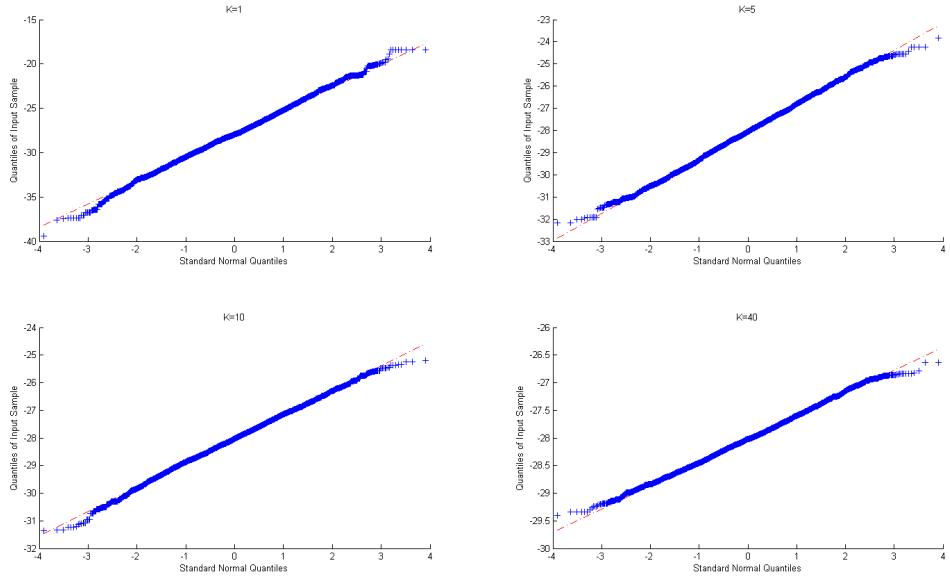

Figure 7.15: Quantile-quantile plots for the simulations of β_2 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

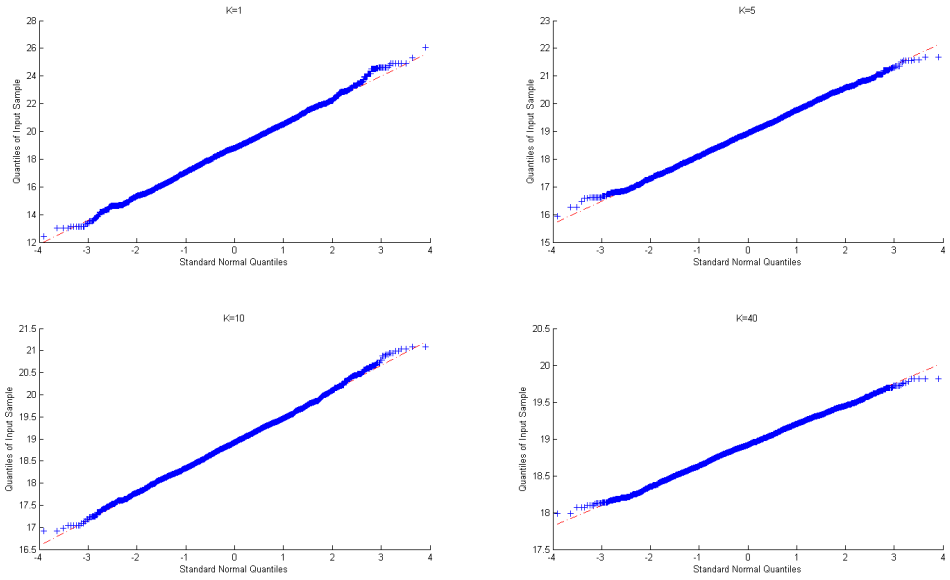

Figure 7.16: Quantile-quantile plots for the simulations of β_3 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

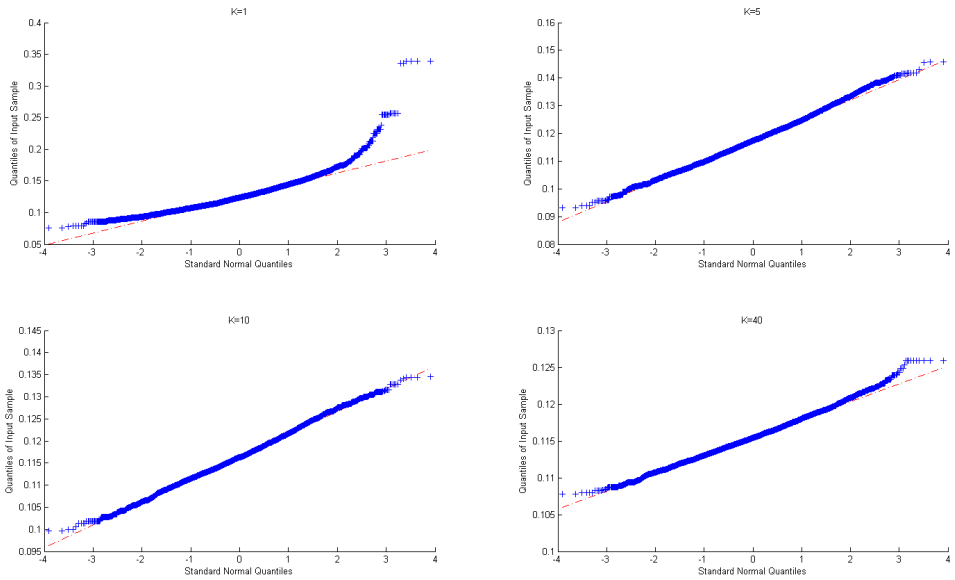

Figure 7.17: Quantile-quantile plots for the simulations of σ^2 using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in.

7.2 Stochastic Dynamical Models without Observational Noise

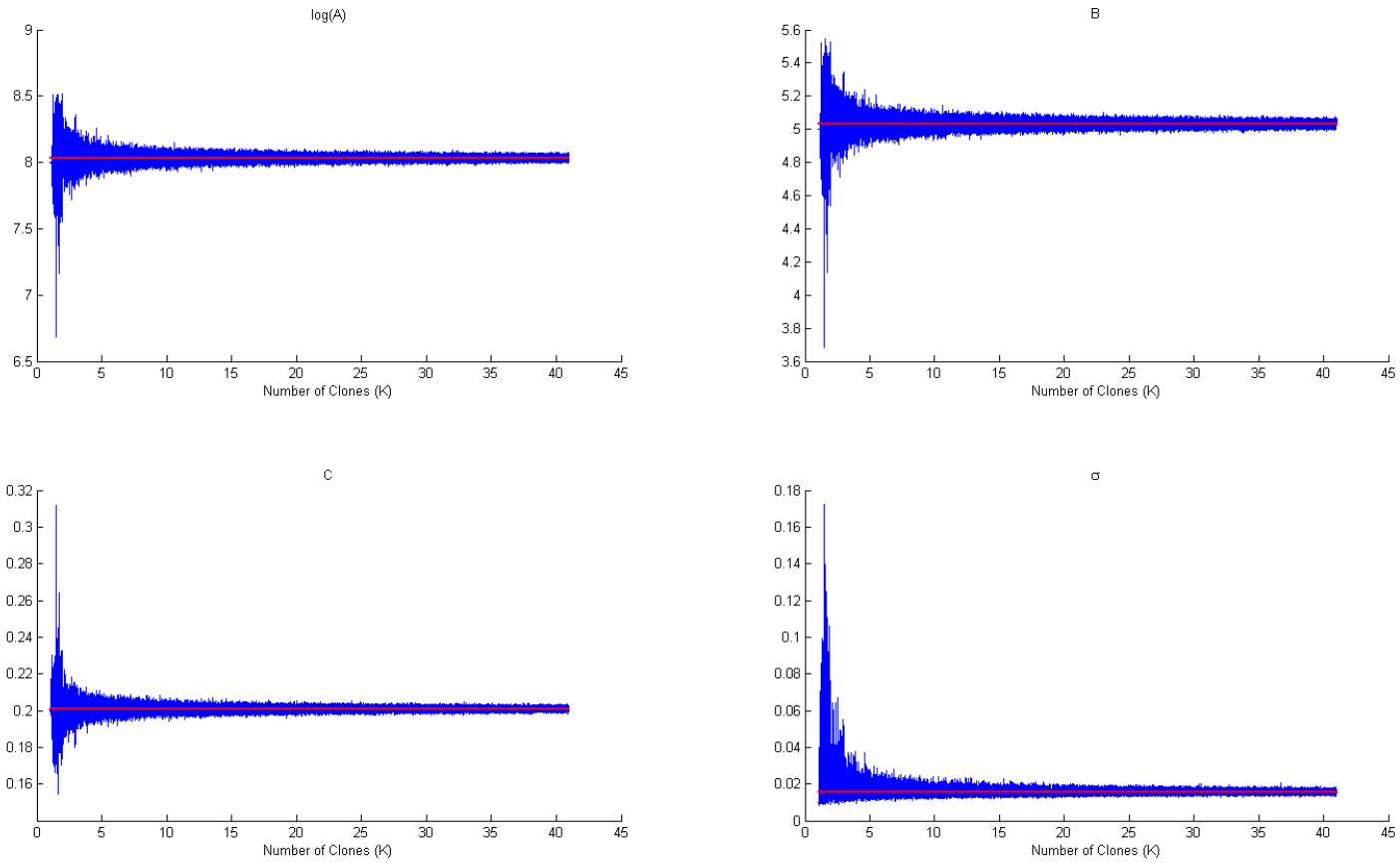

Figure 7.18: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 steps, applying the MIC method to the Gompertz model. The red line is the true MLE. The starting values in the simulation are the true values of parameter, not to be confused with the true MLE.

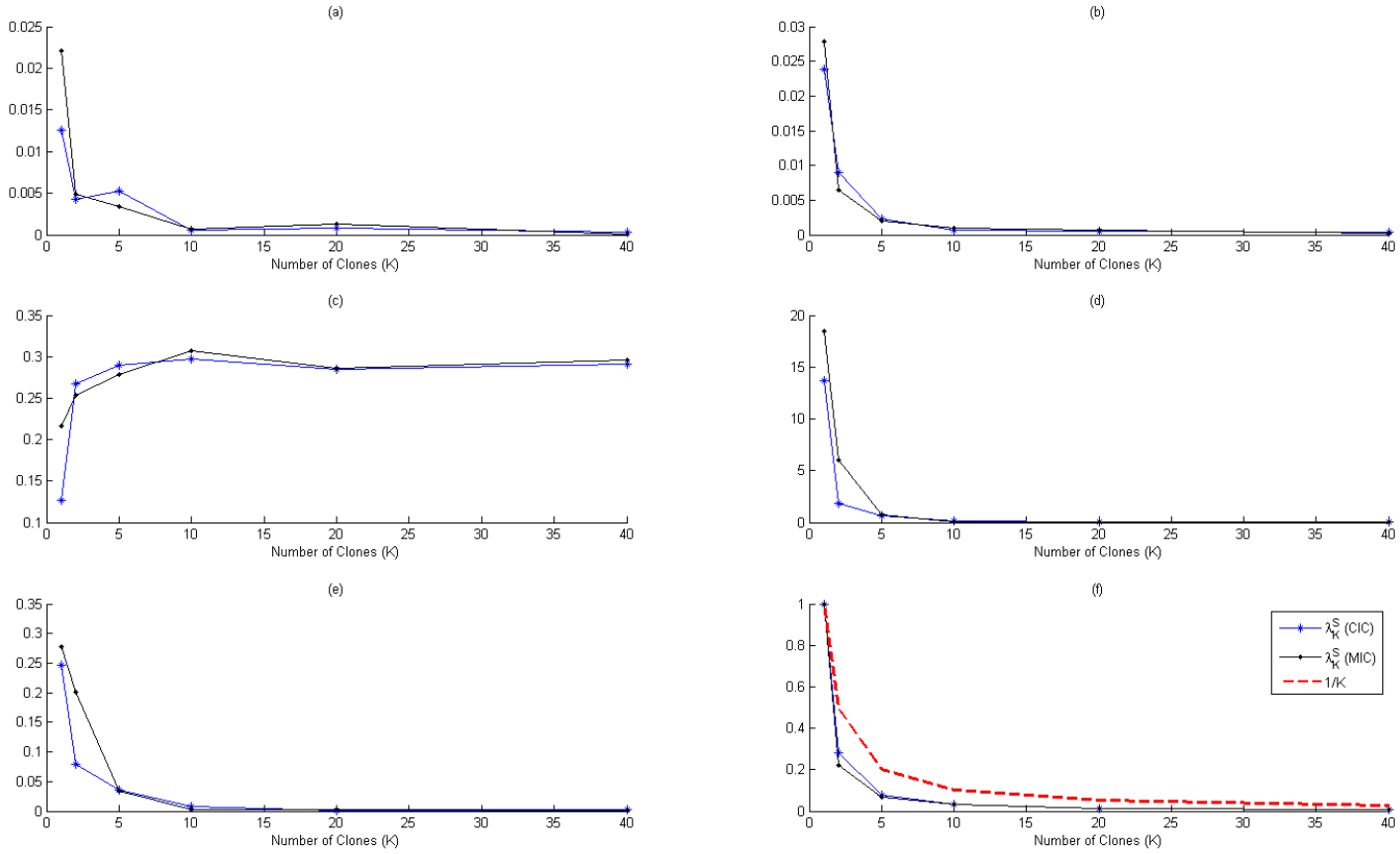

Figure 7.19: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase was done sporadically. The number of clones were $1, 2, 5, 10, 20$ and 40. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE (the 2-norm of the difference), (b) Precision of the estimates of the variance covariance matrix of the MLE (the 2-norm of the difference) (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

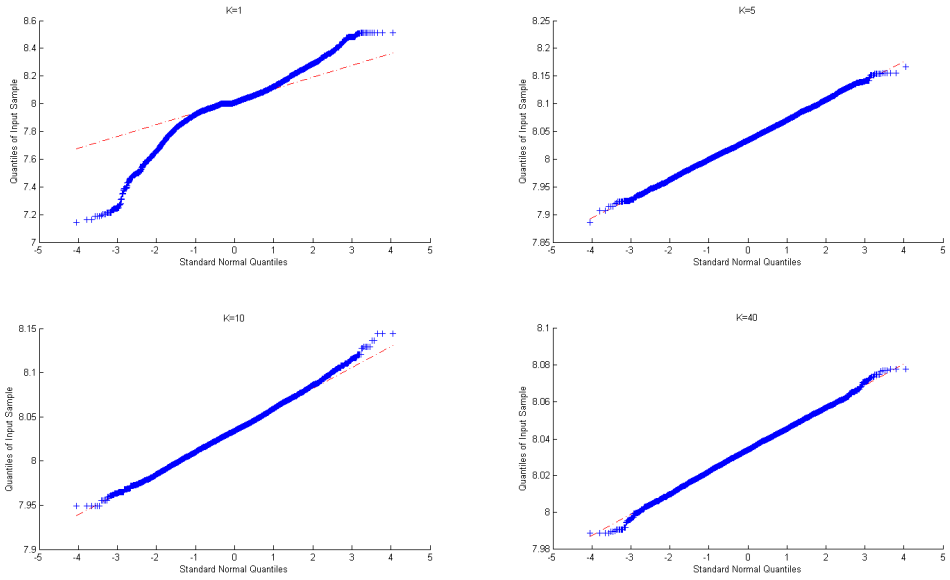

Figure 7.20: Quantile-quantile plots for the simulations of $\ln(A)$ using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

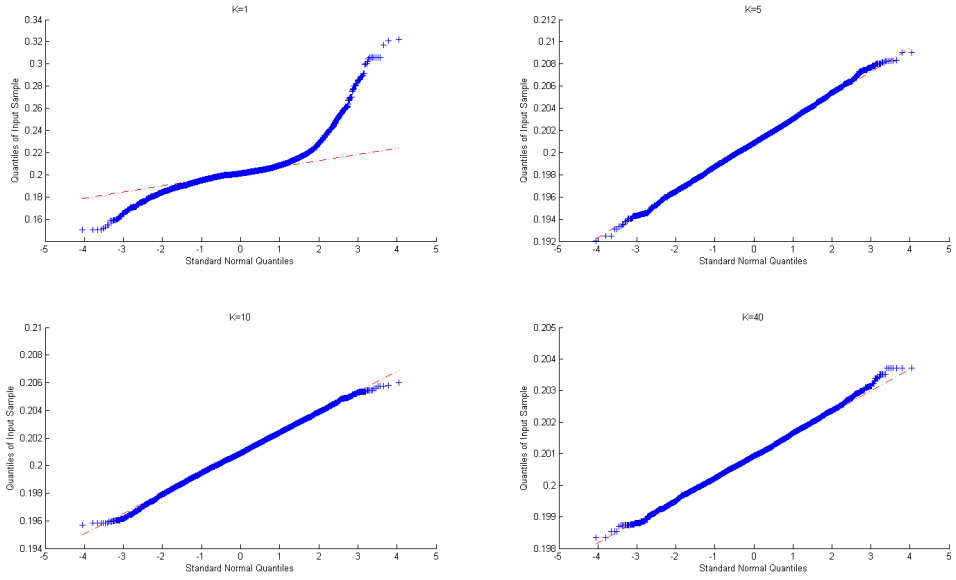

Figure 7.21: Quantile-quantile plots for the simulations of C using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

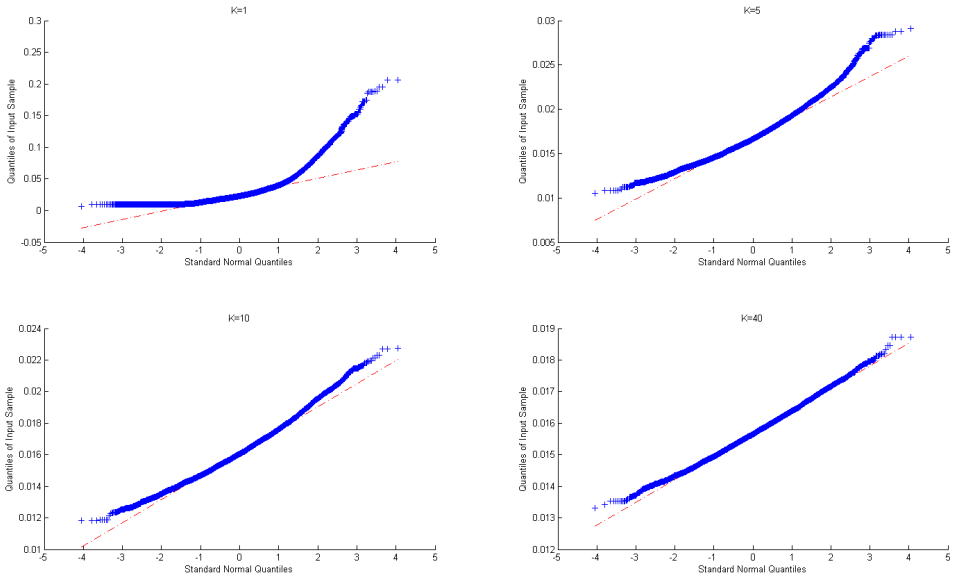

Figure 7.22: Quantile-quantile plots for the simulations of σ using the CIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

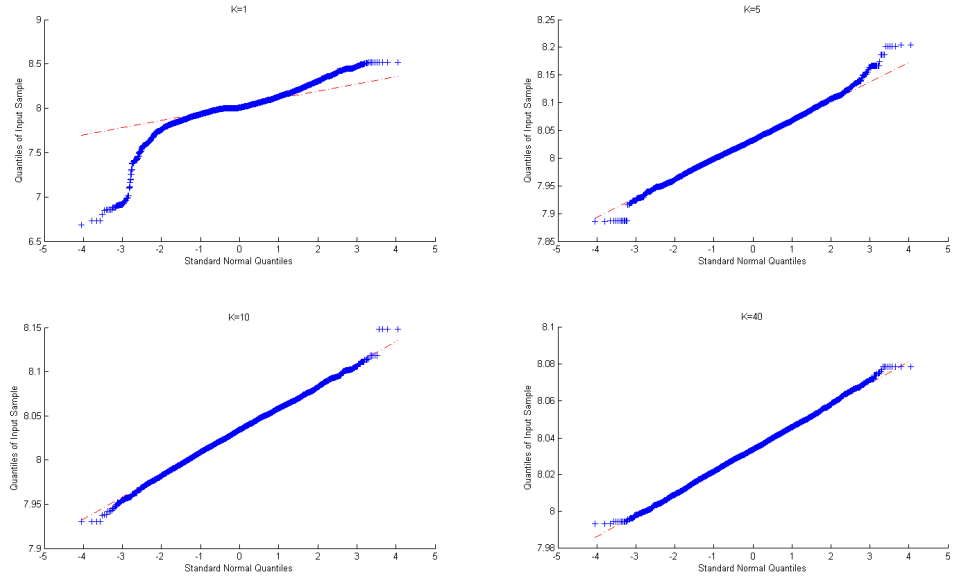

Figure 7.23: Quantile-quantile plots for the simulations of $\ln(A)$ using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

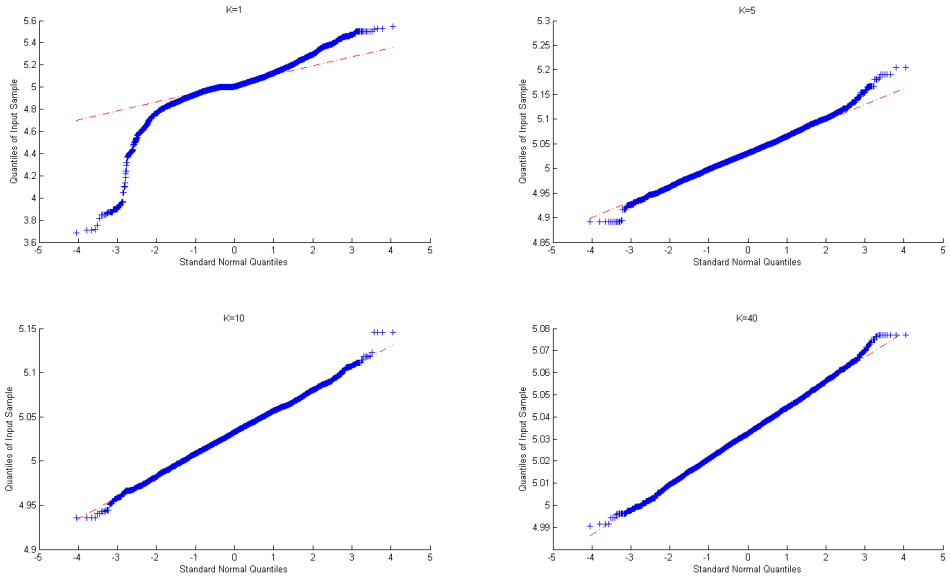

Figure 7.24: Quantile-quantile plots for the simulations of B using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

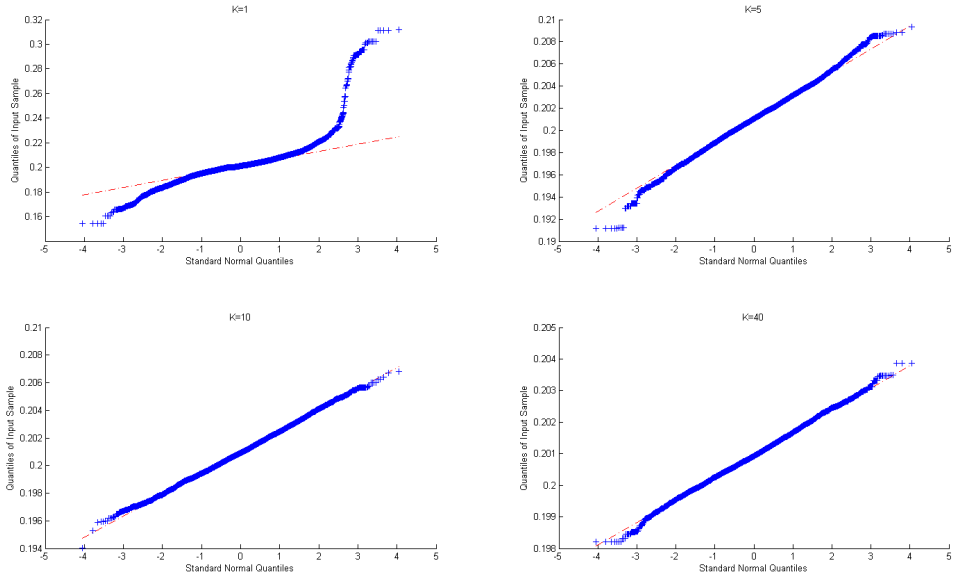

Figure 7.25: Quantile-quantile plots for the simulations of C using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

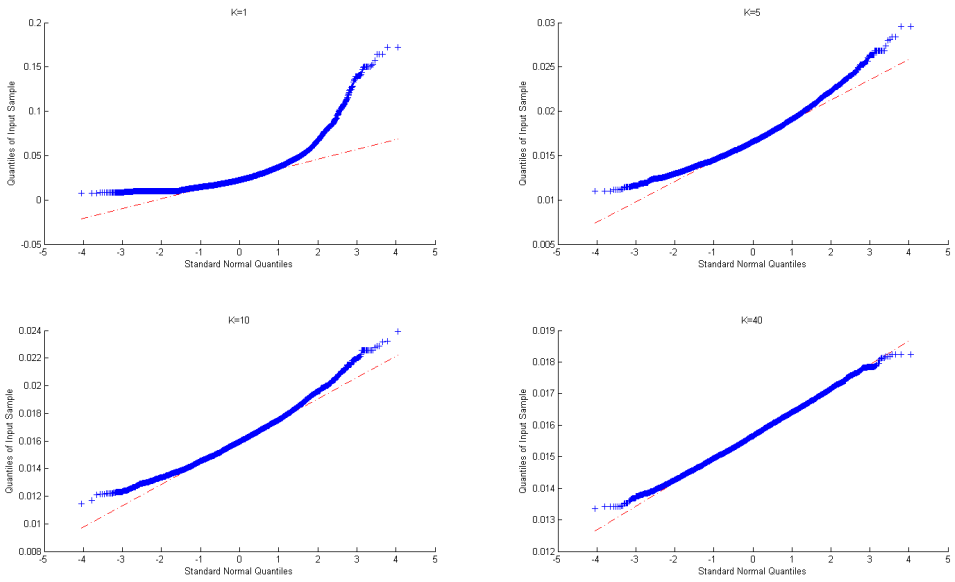

Figure 7.26: Quantile-quantile plots for the simulations of σ using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

7.3 State Space Models

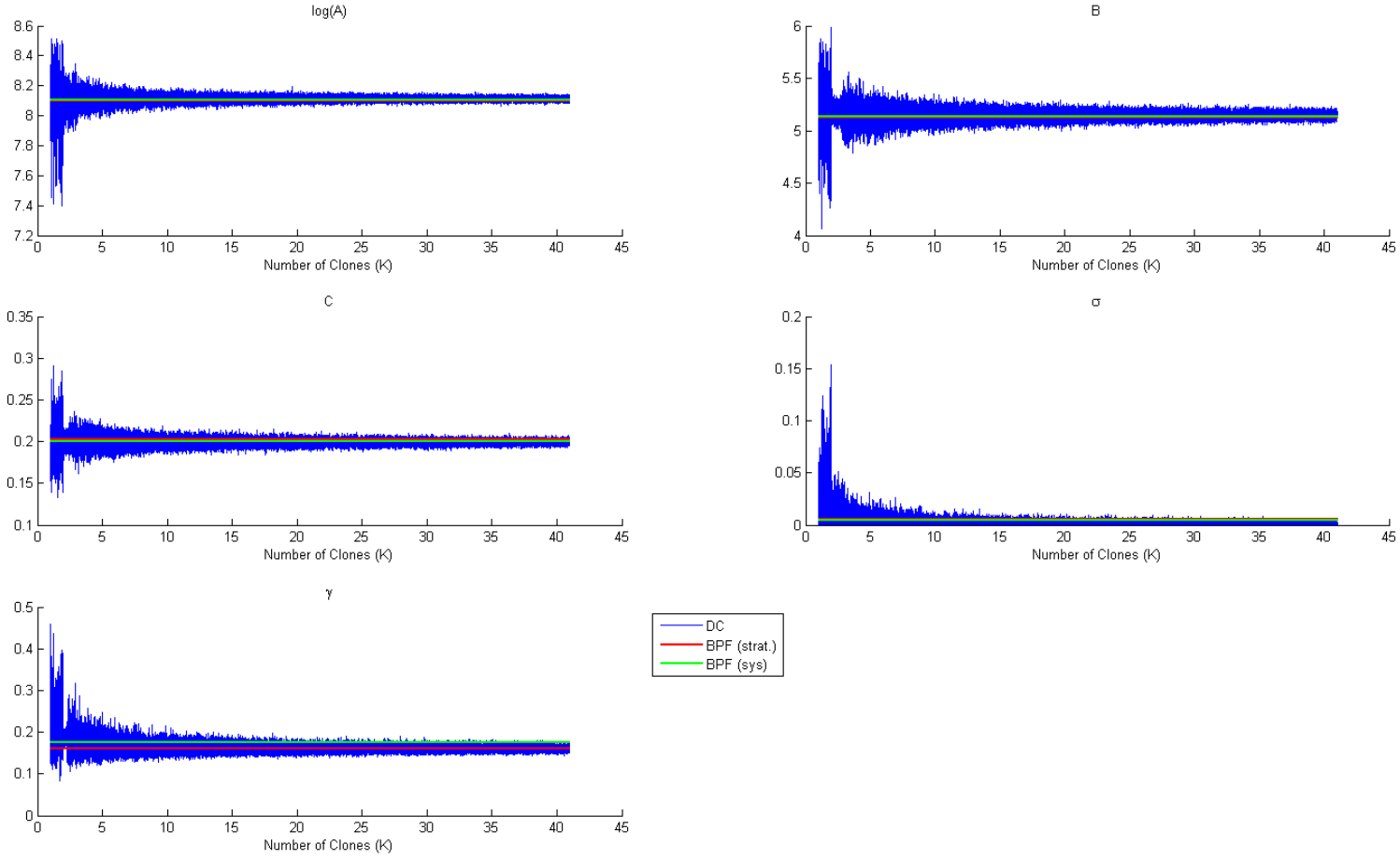

Figure 7.27: MCMC chain of simulations with increasing number of copies (by one, starting at one) every 10^4 iterations, applying the CIC method to the Gompertz model. The red line is the true MLE. The starting values in the simulation are the true values of parameter, not to be confused with the true MLE.

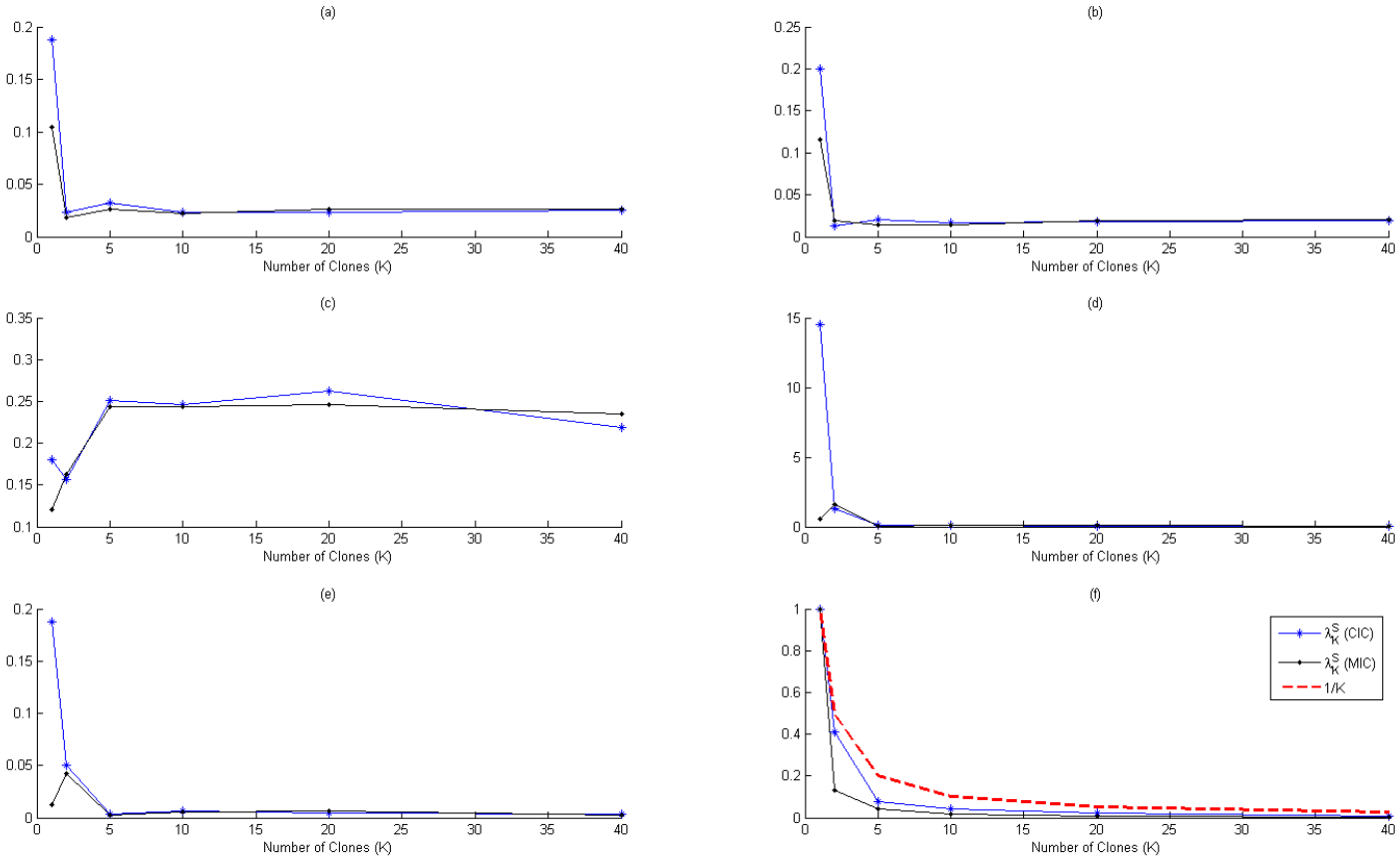

Figure 7.28: Some aspects of the estimates of the MLE via the MIC and CIC methods of data cloning. The increase was done sporadically. The number of clones were $1, 2, 5, 10, 20$ and 40. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus a thousand for burn-in. All is expressed as a function of number of clone (K): (a) Precision of the estimates of the MLE via data-cloning compared with the approximative MLE obtained through the bootstrap particle filter with stratified resampling (the 2-norm of the difference), (b) the same as (a) but with systematic resampling (c) the acceptance rate, (d) the ω-statistic (e) the $\tilde{r}^2$-statistic (f) the standardized largest eigenvalue (λ_K^S) of the covariance matrix.

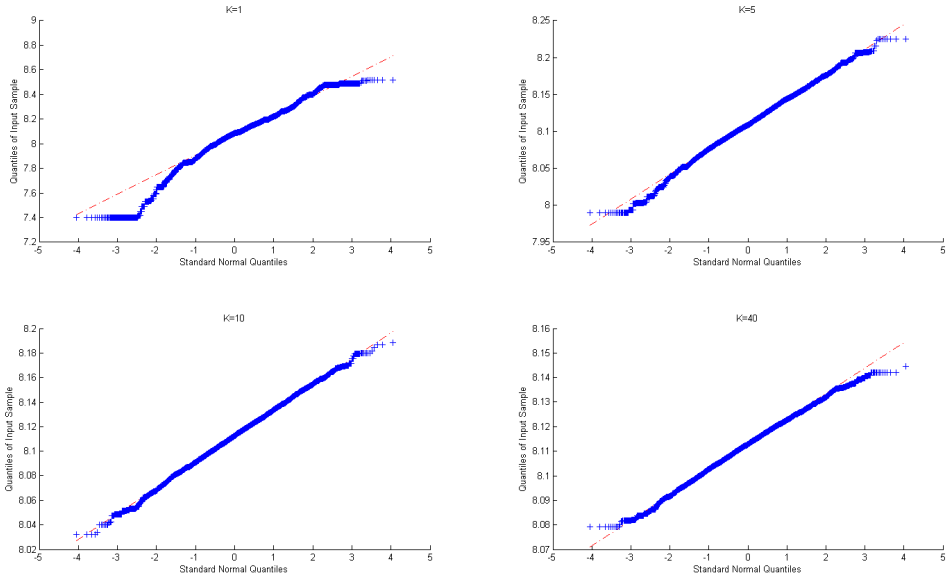

Figure 7.29: Quantile-quantile plots for the simulations of $\ln(A)$ using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

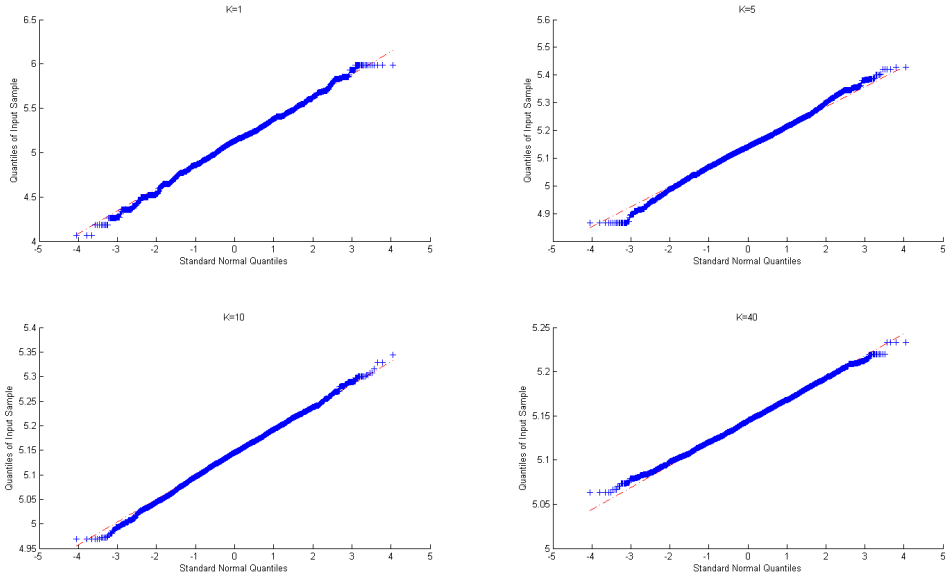

Figure 7.30: Quantile-quantile plots for the simulations of B using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

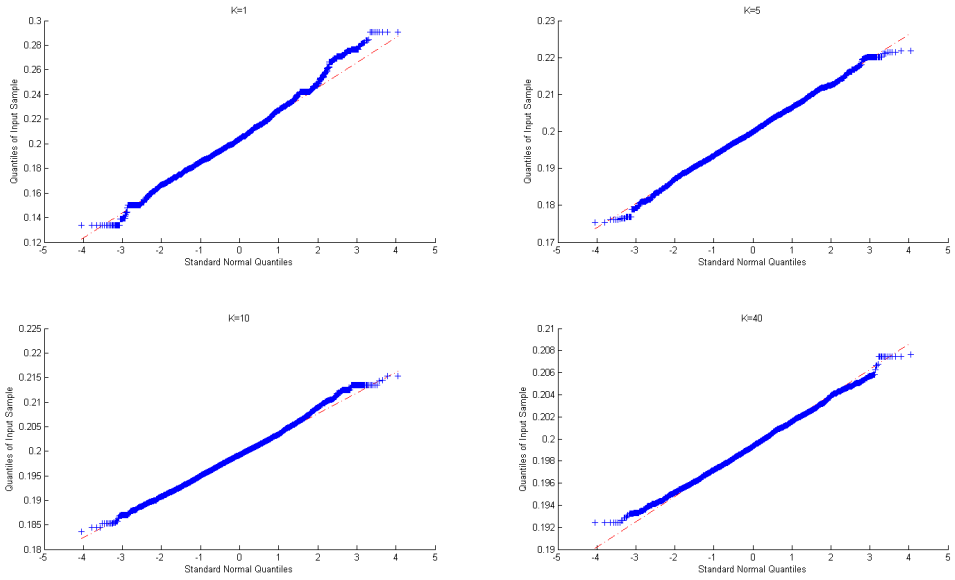

Figure 7.31: Quantile-quantile plots for the simulations of C using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.

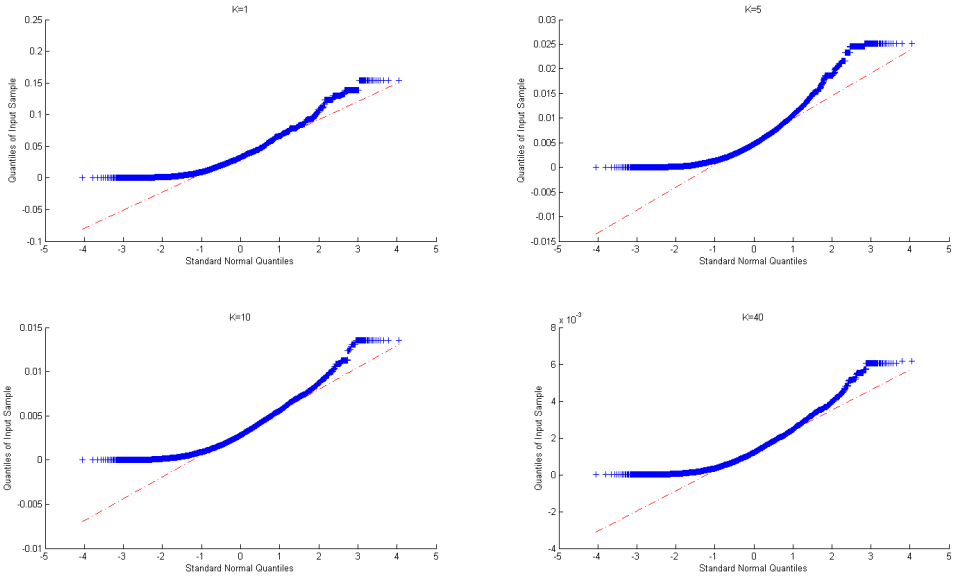

Figure 7.32: Quantile-quantile plots for the simulations of σ using the MIC-method, and for different number of clones (K). The increase of the number of clones has been conducted continuously, i.e. by increasing the number of clones K by one after each block of simulation. For each block, the number of Monte Carlo simulations conducted has been ten thousand regular simulations plus ten thousand for burn-in.