arXiv:2009.12132v1 [stat.CO] 25 Sep 2020

MULTILEVEL GIBBS SAMPLING FOR BAYESIAN REGRESSION*

JORIS TAVERNIER†, JAAK SIMM‡, ÁDÁM ARANY‡, KARL MEERBERGEN†, AND YVES MOREAU‡

Abstract. Bayesian regression remains a simple but effective tool based on Bayesian inference techniques. For large-scale applications, with complicated posterior distributions, Markov Chain Monte Carlo methods are applied. To improve the well-known computational burden of Markov Chain Monte Carlo approach for Bayesian regression, we developed a multilevel Gibbs sampler for Bayesian regression of linear mixed models. The level hierarchy of data matrices is created by clustering the features and/or samples of data matrices. Additionally, the use of correlated samples is investigated for variance reduction to improve the convergence of the Markov Chain. Testing on a diverse set of data sets, speed-up is achieved for almost all of them without significant loss in predictive performance.

Key words. Markov Chain Monte Carlo, Bayesian regression, linear mixed models, Multilevel methods

AMS subject classifications. 65C40, 62J05, 62F15

1. Introduction.

Linear regression remains a simple yet widely used technique in statistics and machine learning. Although more advanced techniques exist, linear models have the advantage that they offer a simple explanation of how the input variables influence the output variable(s). The simplest linear model is a linear combination between a weight vector $\mathbf{b} \in \mathbb{R}^{F\times 1}$ and a data matrix $X \in \mathbb{R}^{N\times F}$

$$\mathbf{y} = X\mathbf{b}$$

with $\mathbf{y} \in \mathbb{R}^{N\times 1}$ the target values, N the number of observations and F the number of features. This can be seen as statistical inference and the most common approaches are Bayesian inference and frequentist inference [2, 11, 15].

Bayesian inference will draw any conclusions about the parameters or observations in terms of probability distributions. One advantage of the Bayesian approach is that it allows uncertainty before (prior) and after (posterior) the observations X and $\mathbf{y}$ have been incorporated. Using prior probability distributions allows the inclusion of information about the model parameters. Another advantage of the Bayesian approach is the ability to make probabilistic predictions for new data using the posterior of the parameter $\mathbf{b}$.

For high dimensional spaces, using exact analytical techniques is not feasible or even impossible and the use of Monte Carlo (MC) methods is widely spread [2, 11, 29, 33]. MC still requires a distribution to draw samples from a probability distribution and Markov Chain Monte Carlo (MCMC) is a general method to sample from arbitrary distributions. MCMC algorithms have given researchers the opportunity to investigate Bayesian inference for complicated statistical models. The most common MCMC methods are Metropolis-Hastings [16, 26] and Gibbs sampling [10, 12] with Gibbs sampling being a special case of Metropolis-hastings.

Bayesian inference has been used for applications from genetics [6, 33], chemogenomics [31] and cell imaging [32]. The data matrices in these fields are often large-scale

*This work was supported Research Foundation - Flanders (FWO, No. G079016).

†Departement of Computer Science, KU Leuven, Joris.Tavernier@cs.kuleuven.be, https://people.cs.kuleuven.be/~joris.tavernier/, Karl.Meerbergen@cs.kuleuven.be.

‡Department of Electrical Engineering, ESAT - STADIUS, KU Leuven, jaak.simm@esat.kuleuven.be, adam.arany@esat.kuleuven.be, moreau@esat.kuleuven.be.

and sparse. High performance computing (HPC) techniques are required for these data sets. More specifically, research has been developed to speed-up Metropolis-Hastings, since not not all the generated samples are accepted in the Markov Chain. In Metropolis-Hastings, computing the next sample in the chain is often computationally intensive and this sample is not always accepted in the Markov chain. Several techniques have been developed in different fields to improve the acceptance rate of Metropolis-Hastings, for example see [4, 9, 20, 36]. In contrast, Gibbs sampling does not have an acceptance step and Simm et al. [31] describe a Gibbs sampler based on solving a linear system using Krylov subspace methods and thus preserving the data sparsity essential for HPC.

We draw inspiration from HPC techniques that have been developed for large-scale partial differential equations (PDE) using multiple levels. The finite difference method approximates solution of the PDE using a spatial discretization step resulting in a linear system. By varying this discretization step, several levels of different accuracies are created as shown in Figure 1.1a. More specifically, discretizing the spatial variables results in a grid of nodes and the solution of the PDE is then approximated in these nodes. A fine discretization leads to many nodes. On the other hand, a coarse discretization results in less grid points. Both the fine and the coarse grid provide approximate solutions of the PDE, where the fine grid finds a better approximation to the exact solution but is more expensive to compute due to the larger number of grid points. By varying the discretization, a hierarchy of grids is created with increasing number of grid points. This approach is called Multigrid and exploits this hierarchy to create an efficient iterative solver or preconditioner for the underlying linear system of the PDE [3, 37].

While Multigrid exploits the hierarchy of grids when solving the resulting linear system, multilevel Monte Carlo (MLMC) for PDEs with random coefficients however exploits the same hierarchy of grids in the sampling process. As detailed in [5] most of the sampling variance is present in the coarsest grid and MLMC will take most of the samples on the coarsest and thus cheaper levels. MLMC has also been successfully applied for stochastic differential equations in [13]. More recently, Robbe et al [28] have combined Multigrid with MLMC for PDEs with random coefficients. The succes of MLMC for classical MC methods has led to increased interest in multilevel Metropolis-Hastings algorithms [8].

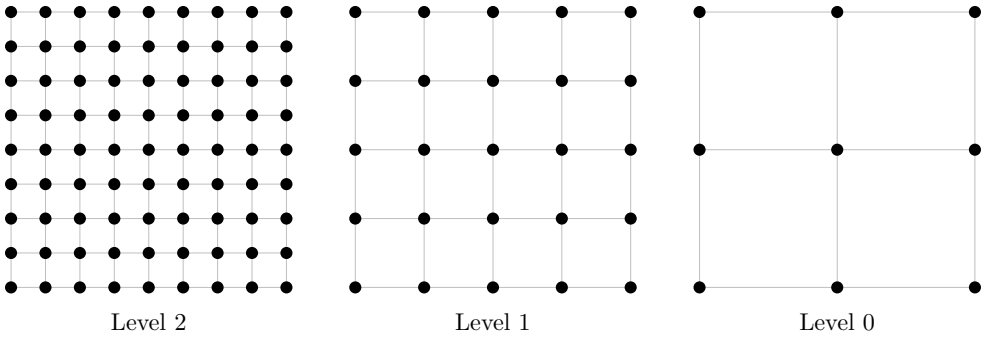
We will design a multilevel version of the Gibbs sampler for Bayesian regression by Simm et al. [31]. In contrast with classical MLMC which varies the discretization step, a hierarchy of data matrices is created using clustering algorithms for the rows and or columns of the data matrix X. Figure 1.1 shows an example for dense matrices with clustering for both rows and columns. Samples from these levels are then combined in one Markov Chain. Samples taken on the coarser levels are cheaper to compute, resulting in overall speed-up.

The outline of the paper is as follows. First, we will provide further background on MCMC and MLMC in Section 2. Next, the Gibbs sampler for Bayesian regression [31] is given and extended for linear mixed models in Section 3. This is followed by our multilevel Gibbs sampler in Section 4 and our multilevel Monte Carlo Gibbs sampler in Section 5. Finally results are given for a variety of data sets in Section 6 and a conclusion is given in Section 7.

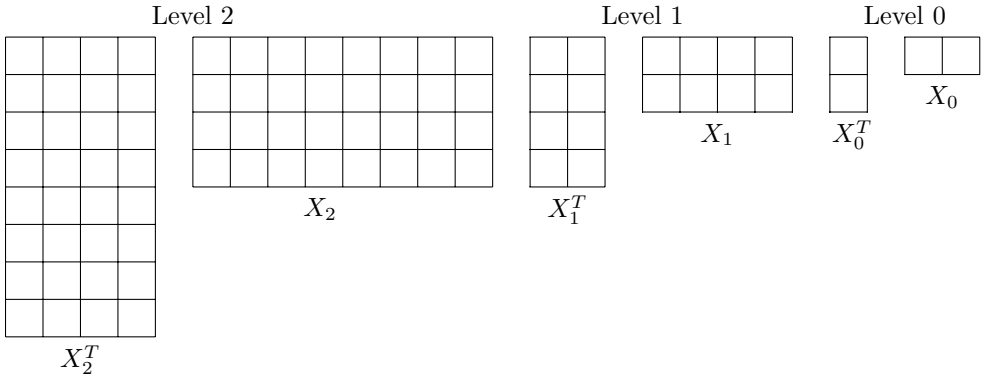
2. Preliminaries.

Linear models are defined by

$$\mathbf{y} = X\mathbf{b} + \mathbf{e} \tag{2.1}$$



(a) Hierarchy of Grids



(b) Hierarchy of dense data matrices for $X^T X$

Fig. 1.1: Examples of the level hierarchies in PDEs and data. Figure 1.1a shows the hierarchy of grids for the discretization of a PDE. Figure 1.1b shows the hierarchy of levels for $X^t X$ using a data matrix X and clustering both the rows and columns.

with $\mathbf{y} \in \mathbb{R}^{N\times 1}$ the N sample observations, $X \in \mathbb{R}^{N\times F}$ a collected feature matrix, F the number of features and $\mathbf{e} \in \mathbb{R}^{N\times 1}$ the unknown residuals. Linear regression aims to find a distribution or point estimate for the weight vector $\mathbf{b} \in \mathbb{R}^{F\times 1}$.

Bayesian inference for linear models estimates the parameter $\mathbf{b}$ using prior information and the Bayes rule. The Bayes rule is defined by

$$p(\mathbf{b}|X) = \frac{p(X|\mathbf{b})p(\mathbf{b})}{p(X)}$$

with $p(\mathbf{b}|X)$ the posterior, $p(X|\mathbf{b})$ the likelihood, $p(\mathbf{b})$ the prior and $p(X)$ the marginal likelihood. Using the posterior, probabilistic predictions can be made for new data X_{new} with the predictive distribution

$$p(X_{\text{new}}|X) = \int p(X_{\text{new}}|\mathbf{b})p(\mathbf{b}|X)d\mathbf{b}.$$

Deriving the exact posterior distribution is not always possible. However, the quantity of interest is often the expected value of the posterior $p(\mathbf{b}|X)$ or more gener-

ally of a function $f(\mathbf{b})$ under a target distribution $p(\mathbf{b})$ with $\mathbf{b}$ a discrete or continuous variable. The expected value is defined by

$$F = \mathbb{E}[f(\mathbf{b})] = \int f(\mathbf{b})p(\mathbf{b})d\mathbf{b}.$$

The Monte Carlo method (MC) approximates this expected value by drawing H random and uncorrelated samples $\mathbf{b}^{(h)}$ from $p(\mathbf{b})$ and then takes the average

$$\mathbb{E}[f(\mathbf{b})] \approx F_H^{MC} \tag{2.2}$$

$$= \frac{1}{H}\sum_{h=1}^{H} f(\mathbf{b}^{(h)}) \tag{2.3}$$

with F_H^{MC} as the MC estimate for the expected value using H samples.

Equation (2.3) estimates the expected value, but still requires to sample from the distribution $p(\mathbf{b})$. Sampling from a specific distribution $p(\mathbf{b})$ can be severely hard, particularly in high-dimensional spaces and $p(\mathbf{b})$ might even be unavailable. Markov Chain Monte Carlo (MCMC) is widely used to sample from these distributions and is a general method for drawing samples from arbitrary posterior distributions [2, 11]. The idea is to sample from proposal distributions and each sample is iteratively drawn from a distribution that approximates the target distribution better, with each sample used to improve and correct the proposal distribution. These samples are drawn sequentially, $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \ldots$, resulting in the Markov chain. Important is that each sample is thus drawn from a proposal distribution that increasingly better approximates the desired target distribution. A Markov Chain generally has a burn-in period where the first $H_{\text{burn_in}}$ samples are not taken into account as these are not yet drawn from a distribution that is close enough to the target distribution.

In order to create a Markov chain, a new sample $\mathbf{b}^{(h)}$ is proposed using a transition distribution $T^h(\mathbf{b}^{(h)}, \mathbf{b}^{(h-1)})$ with $\mathbf{b}^{(h-1)}$ the previous sample. In a Markov chain, a distribution is called stationary or invariant if each new sample leaves that distribution unchanged. The target distribution $p^*(\mathbf{b})$ should be invariant over the Markov chain and a sufficient condition is called detailed balance, defined as

$$p^*(\mathbf{b})T(\mathbf{b}, \mathbf{b}') = p^*(\mathbf{b}')T(\mathbf{b}', \mathbf{b})$$

with $\mathbf{b}$ and $\mathbf{b}'$ two samples. The final to be mentioned property is ergodicity, which says that for any start distribution, the target distribution is invariant and for $h \to \infty$ the distribution $p(\mathbf{b}^{(h)})$ converges to the target distribution $p^*(\mathbf{b})$. The invariant distribution is then called the equilibrium distribution.

The Metropolis-Hastings algorithm is widely used to generate samples in a Markov chain. For each sample $\mathbf{b}^{(h)}$, a follow-up sample $\mathbf{b}'$ is proposed from a jumping distribution $J_h(\mathbf{b}', \mathbf{b}^{(h)})$. The proposed sample is then accepted with probability

$$A(\mathbf{b}', \mathbf{b}^{(h)}) = \min\left(1, \frac{p(\mathbf{b}')J_h(\mathbf{b}^{(h)}, \mathbf{b}')}{p(\mathbf{b}^{(h)})J_h(\mathbf{b}', \mathbf{b}^{(h)})}\right).$$

If the sample is accepted, then $\mathbf{b}^{(h+1)} = \mathbf{b}'$ else $\mathbf{b}^{(h+1)} = \mathbf{b}^{(h)}$. Gibbs sampling can be considered as a special case of the more general Metropolis-Hastings algorithm [2, 11]. Suppose the parameter $\mathbf{b}$ consists of F subcomponents b_i with b_i the individual weight for linear regression. A Gibbs sampler will now sample each component b_i individually

assuming the other components $\mathbf{b}_{\setminus i}$ are given using $p(b_i|\mathbf{b}_{\setminus i})$. A Gibbs sampler has the advantage that each new sample is accepted.

As detailed in [8], the mean square error (MSE) of the MCMC estimator is given by

$$\epsilon(F_H^{MC}) = \mathbb{E}_{\mathbf{B}}\left[(F_H^{MC} - F)^2\right] \tag{2.4}$$

with $\mathbb{E}_{\mathbf{B}}$ the expected value with respect to the joint distribution of $\mathbf{B} = \{\mathbf{b}^{(h)}\}$. Equation (2.4) is typically rewritten as

$$\epsilon(F_H^{MC}) = \mathbb{V}_{\mathbf{B}}\left[F_H^{MC}\right] + \left(\mathbb{E}_{\mathbf{B}}\left[F_H^{MC}\right] - F\right)^2 \tag{2.5}$$

with $\mathbb{V}$ the variance. More details can be found in [8, 29]. It is not possible to express the MSE in terms of H due to the fact that the generated and consecutive samples in the Markov Chain are not independent. This is in contrast with classical Monte Carlo methods, where the MSE can be expressed in terms of the number of independent samples H, e.g. $\epsilon \sim \frac{1}{H}$.

Classical Monte Carlo methods are used in applications for which the numerical solution depends on a discretization step. For these applications, a hierarchy of levels can be created. This is, for example, true for stochastic differential equations [13] using a time step or PDEs using a spatial discretization step [5] as seen in Figure 1.1a. Multilevel Monte Carlo (MLMC) now combines decreasing discretization steps resulting in several levels $l = 0, \ldots, L$ with L the smallest discretization step and thus the finest level. By introducing this level hierarchy, the given notation is adjusted to incorporate the level parameter l. We denote with $f_l(\mathbf{b})$ the function of interest on level l and with $F_{l,H}^{MC}$ the MC estimate on level l for $\mathbb{E}[f_l(\mathbf{b})]$. Following the MLMC terminology, the telescoping sum uses this level hierarchy and exploits the linearity of the expectation operator

$$\mathbb{E}[f_L(\mathbf{b})] = \mathbb{E}[f_0(\mathbf{b})] + \sum_{l=1}^{L} \mathbb{E}[f_l(\mathbf{b}) - f_{l-1}(\mathbf{b})]. \tag{2.6}$$

Using Monte Carlo as unbiased estimator for the expectation combined with (2.6) results in

$$\mathbb{E}[f_L(\mathbf{b})] \approx \frac{1}{H_0}\sum_{h=1}^{H_0} f_0(\mathbf{b}^{(h)}) + \sum_{l=1}^{L}\frac{1}{H_l}\sum_{h=1}^{H_l}\left(f_l(\mathbf{b}^{(h)}) - f_{l-1}(\mathbf{b}^{(h)})\right)$$

with H_l the number of samples on level l. Denoting

$$Y_{l,H_l}^{MC} = \frac{1}{H_l}\sum_{h=1}^{H_l}\left(f_l(\mathbf{b}^{(h)}) - f_{l-1}(\mathbf{b}^{(h)})\right) \tag{2.7}$$

results in

$$\begin{aligned}\mathbb{E}[f_L(\mathbf{b})] &\approx F_{L,\{H_l\}}^{MLMC} \\ &= F_{0,H_0}^{MC} + \sum_{l=1}^{L} Y_{l,H_l}^{MC}\end{aligned}$$

with $F_{L,\{H_l\}}^{MLMC}$ the MLMC estimate for $\mathbb{E}[f_L(\mathbf{b})]$.

In short, there are two principles behind MLMC. The first principle is that taking samples on the coarser levels $l < L$ is computationally cheaper than taking samples on the finest level L. These cheaper samples however still approximate the same solution as the finest level with some loss in accuracy. The second principle can be seen as the use of a control variate. A control variate E for a random variable U is defined by the facts that taking samples from E is cheap and E is positively correlated with U. Exploiting the linearity of the expectation operator results in

$$\mathbb{E}[U] = \mathbb{E}[E] + \mathbb{E}[U - E]. \tag{2.8}$$

Since E is positively correlated with U, $\mathbb{V}[U - E] < \mathbb{V}[U]$ and an MC estimator for the right-hand side of (2.8) will require less samples taken from U than a MC estimator for $\mathbb{E}[U]$ in order to reach the same MSE. Note the resemblance of (2.8) and the telescoping sum (2.6). In order for the coarser level in (2.7) to act as a control variate, the samples $f_l(\mathbf{b}^{(h)})$ and $f_{l-1}(\mathbf{b}^{(h)})$ need to be positively correlated. It is thus important that when using MC for estimating the difference $\mathbb{E}[f_l(\mathbf{b}) - f_{l-1}(\mathbf{b})] \approx \frac{1}{H_l}\sum_{h=1}^{H_l}(f_l(\mathbf{b}^{(h)}) - f_{l-1}(\mathbf{b}^{(h)})$, that the H_l pairs $\{f_l(\mathbf{b}^{(h)}), f_{l-1}(\mathbf{b}^{(h)})\}$ are uncorrelated but within a pair the samples $f_l(\mathbf{b}^{(h)})$ and $f_{l-1}(\mathbf{b}^{(h)})$ are positively correlated. The coarser level $l-1$ can then be seen as a control variate for the finer level l. In conclusion, the coarser levels in MLMC are constructed such that the coarser levels contain most of the variance and MLMC will as a result take more samples on the coarser levels and thus cheaper levels, resulting in less computational work [5, 8, 13, 28].

Due to the success of MLMC in different fields, Multilevel Markov chain Monte Carlo has been investigated [8, 36]. By using a hierarchy of levels, the acceptance rate of Metropolis-Hastings can be improved. In [8] they created a hierarchy of chains to improve the acceptance rate and additionally reduce the variance for subsurface flow. Currently, there is no generally accepted method to design Multilevel Markov Chain Monte Carlo methods. One problem is that the generated samples in MCMC are not independent as required by MLMC. One technique is to only consider each k-th sample in the chain and is called thinning.

3. Noise Injection Gibbs Sampler for Bayesian regression. In ordinary linear regression [2, 11] the residuals are assumed uncorrelated and normally distributed in (2.1), e.g. $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \tau^{-1} I_N)$ with I_N the identity matrix of size $\mathbb{R}^{N\times N}$ and $\tau > 0$ the precision parameter. In the Bayesian approach the observations $\mathbf{y}$ are probabilistic

$$p(\mathbf{y}|X, \mathbf{b}) = \prod_{i=1}^{N} \mathcal{N}(y_i|\mathbf{x}_i\mathbf{b}, \tau^{-1})$$

with $\mathbf{x}_i \in \mathbb{R}^{1\times F}$ the features of sample i. The conditional posterior distribution of $\mathbf{b}$ given τ, X, β and $\mathbf{y}$ is then

$$p(\mathbf{b}|\tau, \mathbf{y}, X) \sim \mathcal{N}((X^T X + \beta I_F)^{-1} X^T \mathbf{y}, ((X^T X + \beta I_F)\tau)^{-1})$$

with β a regularization parameter. Computing the covariance matrix $((X^T X + \beta I_F)\tau)^{-1}$ is inconvenient and often computationally infeasible. Simm et al. [31] however describe a Gibbs sampler for Bayesian regression and using their noise injection trick, a sample of $\mathbf{b}$ can be taken by solving

$$\left(X^T X + \frac{\lambda_{\mathbf{b}}}{\tau} I_F\right)\mathbf{b} = X^T(\mathbf{y} + \mathbf{e}_1) + \frac{\mathbf{e}_2}{\tau}$$

with $\mathbf{e}_1 \sim \mathcal{N}(\mathbf{0}, \tau^{-1} I_N)$ and $\mathbf{e}_2 \sim \mathcal{N}(\mathbf{0}, \lambda_\mathbf{b} I_F)$. A zero mean and uncorrelated normal distribution is used as prior for $\mathbf{b}$:

$$p(\mathbf{b}, \lambda_\mathbf{b} | \alpha_\mathbf{b}, \beta_\mathbf{b}) \sim \mathcal{N}(\mathbf{b} | \mathbf{0}, \lambda_\mathbf{b}^{-1} I_F) \mathcal{G}(\lambda_\mathbf{b} | \alpha_\mathbf{b}, \beta_\mathbf{b})$$

with $\mathcal{G}(\lambda_\mathbf{b} | \alpha_\mathbf{b}, \beta_\mathbf{b})$ the gamma distribution used as conjugate prior for the precision parameter $\lambda_\mathbf{b} > 0$.

The noise injection trick can also be extended to the linear mixed model equations [17, 18]

$$\mathbf{y} = W\mathbf{v} + Z\mathbf{u} + \mathbf{e} \tag{3.1}$$

with $\mathbf{y} \in \mathbb{R}^{N \times 1}$ the observations and N the sample size. The vector $\mathbf{v} \in \mathbb{R}^{F \times 1}$ represents the F fixed effects and $\mathbf{u} \in \mathbb{R}^{S \times 1}$ are the S random effects. The matrices $W \in \mathbb{R}^{N \times F}$ and $Z \in \mathbb{R}^{N \times S}$ are incidence matrices coupling the effects to the observations [6]. We assume that the random effects and the residual error $\mathbf{e} \in \mathbb{R}^{N \times 1}$ are normally distributed: $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, G)$ and $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, R)$. This formulation can further be simplified by assuming that the random effects are a priori uncorrelated, e.g. $G = \lambda_\mathbf{u}^{-1} I_S$ with $\lambda_\mathbf{u} > 0$ the precision. As before, we assume that the residual errors ($\mathbf{e}$) are uncorrelated, resulting in $R = \tau^{-1} I_N$ with $\tau > 0$ the precision.

The linear mixed model (3.1) can now be rewritten as

$$\mathbf{y} = X\mathbf{b} + \mathbf{e}$$

with $X = [W \ Z]$ and $\mathbf{b} = [\mathbf{v}^T \ \mathbf{u}^T]^T$, resulting in

$$p(\mathbf{y} | X, \mathbf{b}) = \prod_{i=1}^{N} \mathcal{N}(y_i | \mathbf{x}_i \mathbf{b}, \tau^{-1}).$$

In the Bayesian setting [11, 33], the fixed effects are uncertain and assumed normally distributed $\mathbf{v} \sim \mathcal{N}(0 | \lambda_\mathbf{v}^{-1} I_F)$ with $\lambda_\mathbf{v} > 0$ the precision parameter. Using gamma distributions as priors for $\lambda_\mathbf{v}$ and $\lambda_\mathbf{u}$, the conditional probability is

$$p(\mathbf{b}, \Lambda | \mathbf{y}, X) = \mathcal{N}(\mathbf{b} | 0, \Lambda^{-1}) \mathcal{G}(\lambda_\mathbf{v} | \alpha_\mathbf{v}, \beta_\mathbf{v}) \mathcal{G}(\lambda_\mathbf{u} | \alpha_\mathbf{u}, \beta_\mathbf{u})$$

with

$$\Lambda = \begin{bmatrix} \lambda_\mathbf{v} & & & & & 0 \\ & \ddots & & & & \\ & & \lambda_\mathbf{v} & & & \\ & & & \lambda_\mathbf{u} & & \\ & & & & \ddots & \\ 0 & & & & & \lambda_\mathbf{u} \end{bmatrix}. \tag{3.2}$$

Using the noise injection trick [31], a sample of $\mathbf{b}$ can be taken by solving

$$\left(X^T X + \frac{\Lambda}{\tau} \right) \mathbf{b} = X^T (\mathbf{y} + \mathbf{e}_1) + \frac{\mathbf{e}_2}{\tau} \tag{3.3}$$

with $\mathbf{e}_1 = \mathcal{N}(\mathbf{0}, \tau^{-1} I)$ and $\mathbf{e}_2 = \mathcal{N}(\mathbf{0}, \Lambda)$. Samples for τ, $\lambda_\mathbf{v}$ and $\lambda_\mathbf{u}$ are taken from

respectively

$$p(\tau|\mathbf{b},\mathbf{y},X,\alpha_{\mathbf{e}},\beta_{\mathbf{e}}) = \mathcal{G}\left(\tau\middle|\alpha_{\mathbf{e}}+\frac{N}{2},\beta_{\mathbf{e}}+\sum_{i=1}^{N}\frac{(\mathbf{y}_i-X_i\mathbf{b})^2}{2}\right), \tag{3.4}$$

$$p(\lambda_{\mathbf{v}}|\mathbf{b},\mathbf{y},X,\alpha_{\mathbf{v}},\beta_{\mathbf{v}}) = \mathcal{G}\left(\lambda_{\mathbf{v}}\middle|\alpha_{\mathbf{v}}+\frac{F}{2},\beta_{\mathbf{v}}+\sum_{f=1}^{F}\frac{\mathbf{b}_f^2}{2}\right), \tag{3.5}$$

$$p(\lambda_{\mathbf{u}}|\mathbf{b},\mathbf{y},X,\alpha_{\mathbf{u}},\beta_{\mathbf{u}}) = \mathcal{G}\left(\lambda_{\mathbf{u}}\middle|\alpha_{\mathbf{u}}+\frac{S}{2},\beta_{\mathbf{u}}+\sum_{s=F+1}^{F+S}\frac{\mathbf{b}_s^2}{2}\right). \tag{3.6}$$

The noise injection Gibbs sampler for linear mixed models is given in Algorithm 3.1.

Algorithm 3.1 Noise injection for linear mixed models

Input:
- matrix X
- hyperparameters $\alpha_{\mathbf{e}}$, $\alpha_{\mathbf{v}}$, $\alpha_{\mathbf{u}}$, $\beta_{\mathbf{e}}$, $\beta_{\mathbf{v}}$ and $\beta_{\mathbf{u}}$

sample τ, $\lambda_{\mathbf{v}}$ and $\lambda_{\mathbf{u}}$ from priors
generate $\mathbf{e}_1 = \mathcal{N}(\mathbf{0},\tau^{-1}I)$ and $\mathbf{e}_2 = \mathcal{N}(\mathbf{0},\Lambda)$ using (3.2)
sample $\mathbf{b}^{(1)}$ by solving 3.3
for $h = 2,\ldots,H$ **do**
  sample τ, $\lambda_{\mathbf{v}}$ and $\lambda_{\mathbf{u}}$ from (3.4), (3.5) and (3.6)
  generate $\mathbf{e}_1 = \mathcal{N}(\mathbf{0},\tau^{-1}I)$ and $\mathbf{e}_2 = \mathcal{N}(\mathbf{0},\Lambda)$ using (3.2)
  sample $\mathbf{b}^{(h)}$ by solving (3.3)
end for
return $\mathbb{E}[\mathbf{y}] \approx \frac{1}{H}\sum_{h=1}^{H} X\mathbf{b}^{(h)}$

4. Hierarchical Markov Chain Monte-Carlo for Bayesian regression. We are interested in $\mathbb{E}[\mathbf{y}] = \mathbb{E}[X\boldsymbol{b}]$. Using the Gibbs sampler approach, we approximate

$$\mathbb{E}[\mathbf{y}] \approx \frac{1}{H}\sum_{h=1}^{H} X\mathbf{b}^{(h)}$$

with $\mathbf{b}^{(h)}$ the samples taken by solving (3.3). For large-scale data, solving thousands linear systems (3.3) can be computationally expensive even when iterative solvers are used. Therefore, we want to create a hierarchy of levels approximating the data set X resulting in data levels with increasing computational work and data information. We propose to use clustering algorithms to create this hierarchy of $L+1$ data matrices X_l for $l = 0\ldots L$ by clustering the features (columns) of $X = X_L$. This strategy has been applied to create a two-level preconditioner for Bayesian regression [34]. A coarser data matrix is created by defining the coarse features as

$$\mathbf{c}_{\mathcal{S}} = 1/\sqrt{|\mathcal{S}|}\sum_{i\in\mathcal{S}} X(:,i)$$

with $\mathbf{c}_{\mathcal{S}}$ the coarse feature representing the features of cluster $\mathcal{S}$ and $|S|$ the number of features in $\mathcal{S}$. When the features in $X_L = [X_1, X_2, \ldots, X_{F_C}]$ are ordered per cluster with F_C the number of clusters and X_i the data matrix with the features of cluster i,

we can define a coarser data matrix as $X_{L-1} = X_L P_L$ with

$$P_L^T = \begin{bmatrix} \overbrace{\frac{1}{\sqrt{n_1}} \cdots \frac{1}{\sqrt{n_1}}}^{n_1} & & & \overbrace{\quad}^{n_2} & & \cdots & \overbrace{\quad}^{n_{F_C}} \\ & & & \frac{1}{\sqrt{n_2}} \cdots \frac{1}{\sqrt{n_2}} & & & \\ & & & & \ddots & & \\ & & & & & \frac{1}{\sqrt{n_{F_C}}} \cdots \frac{1}{\sqrt{n_{F_C}}} \end{bmatrix}. \quad (4.1)$$

Note that this definition of P_L leads to

$$\begin{aligned} P_L^T (X_L^T X_L + \beta I) P_L &= P_L^T X_L^T X_L P_L + P_L^T \beta I P_L \\ &= X_{L-1}^T X_{L-1} + \beta \underbrace{P_L^T P_L}_{=I} \end{aligned}$$

as detailed in [34].

The linear system (3.3) involves $X^T X$ and this matrix-product is used in principle component analysis (PCA). PCA finds the eigenvectors with the largest eigenvalues of $X^T X$ and is a projection method that maximizes total data variance [2, 34]. By clustering the features, ideally the variance within a cluster is as small as possible. This means that the coarser levels can approximate the larger eigenvectors and contain most of the data variance. Note that applying PCA directly to create the coarser level is not feasible since it requires the eigenvalue decomposition of $X^T X$.

Using a hierarchy of data matrices, it is possible to define a Gibbs sampler on each level. One simple approach is to define a Markov Chain for each level. This would result in $L+1$ Markov Chains and thus $L+1$ burn-ins. It is, however, possible to interpolate a sample $\mathbf{b}_l$ on level l to level $l+1$ with $\mathbf{b}_{l+1} = P_{l+1}\mathbf{b}_l$ for $l = 0 \dots L-1$. This interpolated sample is then used to sample τ, $\lambda_{\mathbf{v}}$ and $\lambda_{\mathbf{u}}$ in equations (3.4), (3.5) and (3.6). Next, a sample $\mathbf{b}_{l+1}$ on level $l+1$ is drawn using equation (3.3). This provides us with samples for the parameter $\mathbf{b}_l$ for each level l. The quantity of interest is the expected value of the predicted observations $\mathbb{E}[\mathbf{y}]$. Since $X_l P_l = X_{l-1}$ and thus $X_{l-1}\mathbf{b}_{l-1} = X_l P_l \mathbf{b}_{l-1}$.

This means that $\mathbb{E}[\mathbf{y}]$ can be estimated using the fine level data matrix X_L and coarser solutions $\mathbf{b}_l$ with

$$\mathbb{E}[\mathbf{y}] \approx X_L \frac{\prod_{k=l}^{L-1} P_{k+1} \mathbf{b}_l}{H_l}$$

for $l = 0, \dots, L$. This is important since generally we are interested in the predictions of new and unseen data X_{test} with the model trained using different data X_{train}. Combining the samples from each level l, $\mathbb{E}[\mathbf{y}]$ is approximated by

$$\mathbb{E}[\mathbf{y}] \approx X_L \frac{\sum_{l=0}^{L} \left(\prod_{k=l}^{L-1} P_{k+1} \left(\sum_{h=1}^{H_l} \mathbf{b}_l^{(h)} \right) \right)}{\sum_{l=0}^{L} H_l}.$$

Multilevel Gibbs sampling using noise injection for linear mixed models is given in Algorithm 4.1.

Algorithm 4.1 Multilevel Noise injection Gibbs sampler for linear mixed models

Input:
- Hierarchy of matrices X_l for $l = 0, \dots, L$
- hyperparameters $\alpha_{\mathbf{e}}$, $\alpha_{\mathbf{v}}$, $\alpha_{\mathbf{u}}$, $\beta_{\mathbf{e}}$, $\beta_{\mathbf{v}}$ and $\beta_{\mathbf{u}}$

Take H_0 samples $\mathbf{b}_0^{(h)}$ on the **coarsest level** $l = 0$ using the Noise Injection Algorithm 3.1
$\boldsymbol{\Sigma_0} = X_L \prod_{k=0}^{L-1} P_{k+1} \left(\sum_{h=1}^{H_0} \mathbf{b}_0^{(h)} \right)$
Sample on the **remaining levels**
  for $l = 1, \dots, L$ **do**
    interpolate $\mathbf{b}_l = P_l \mathbf{b}_{l-1}^{(H_{l-1})}$
    for $h = 1, \dots, H_l$ **do**
      sample τ, $\lambda_{\mathbf{v}}$ and $\lambda_{\mathbf{u}}$ from (3.4), (3.5) and (3.6)
      generate $\mathbf{e}_1 = \mathcal{N}(\mathbf{0}, \tau^{-1} I)$ and $\mathbf{e}_2 = \mathcal{N}(\mathbf{0}, \Lambda)$ using (3.2)
      sample $\mathbf{b}^{(h)}$ by solving (3.3) using X_l
    end for
    $\boldsymbol{\Sigma_l} = X_L \prod_{k=l}^{L-1} P_{k+1} \left(\sum_{h=1}^{H_l} \mathbf{b}_l^{(h)} \right)$
  end for
return $\mathbb{E}[\mathbf{y}] \approx \frac{\sum_{l=0}^{L} \boldsymbol{\Sigma_l}}{\sum_{l=0}^{L} H_l}$

5. Multilevel Markov Chain Monte Carlo using correlated samples. MLMC has proven to speed-up Monte Carlo methods for different applications. As explained in Section **??**, positive correlation between samples on different levels results in variance reduction of the MC estimator. Before detailing the correlation for the Gibbs sampler, the telescoping sum (**??**) is first applied to linear models

$$\begin{aligned} \mathbb{E}[\mathbf{y}_L] &= \mathbb{E}[\mathbf{y}_0] + \sum_{l=1}^{L} \mathbb{E}[\mathbf{y}_l - \mathbf{y}_{l-1}] && (5.1) \\ &= \mathbb{E}[X_0 \mathbf{b}_0] + \sum_{l=1}^{L} \mathbb{E}[X_l \mathbf{b}_l - X_{l-1} \mathbf{b}_{l-1}] \\ &= X_0 \mathbb{E}[\mathbf{b}_0] + \sum_{l=1}^{L} X_l \mathbb{E}[\mathbf{b}_l - P_l \mathbf{b}_{l-1}] \\ &\approx X_0 \sum_{h=1}^{H_0} \mathbf{b}_0^{(h)} + \sum_{l=1}^{L} X_l \frac{1}{H_l} \sum_{h=1}^{H_l} \mathbf{b}_l^{(h)} - P_l \mathbf{b}_{l-1}^{(h)} && (5.2) \end{aligned}$$

and consists of the expected value on the coarsest level and L correcting terms. For the MLMC scheme to be profitable, the samples $\mathbf{b}_l^{(h)}$ and $P_l \mathbf{b}_{l-1}^{(h)}$ in (5.2) should be correlated. Two approaches to ensure correlation between the samples are considered. The first approach is simply to define $P_l^T \mathbf{b}_l = \mathbf{b}_{l-1}$ and this approach thus takes approximately the average within a cluster (up to a factor) of the fine solution as the coarse value which results in

$$\mathbb{E}[\mathbf{y}_l - \mathbf{y}_{l-1}] \approx X_L \prod_{k=l}^{L-1} P_{k+1} \left(\frac{1}{H_l} \sum_{h=1}^{H_l} \mathbf{b}_l^{(h)} - P_l P_l^T \mathbf{b}_l^{(h)} \right). \quad (5.3)$$

This is a simple projection of the finer solution to the coarser solution. It is, however, important that the finer samples $\mathbf{b}_l^{(h_l)}$ of the difference on level $l = i$ are drawn from

the same posterior as the coarser samples $\mathbf{b}_{l-1}^{(h_{l+1})}$ for the next level $l = i + 1$ for $i = 0, \ldots, L-1$. Simply aggregating the finer solution does not guarantee this.

The second approach solves (3.3) to draw sample $\mathbf{b}_{l-1}$ on the coarser level with exactly the same values $\tau, \lambda_{\mathbf{v}}$, $\lambda_{\mathbf{u}}$ and $\mathbf{e}_1$ used for drawing sample $\mathbf{b}^l$ on the finer level. Only the random vector $\mathbf{e}_2$ is projected to the coarser space $P_l^T \mathbf{e}_{2,l} = \mathbf{e}_{2,l-1}$. Empirical results will show that the second approach is computationally more expensive but obtains better results. The algorithm using the second approach for the correlated samples is outlined in Algorithm 5.1.

Algorithm 5.1 Multilevel Monte Carlo noise injection Gibbs sampler using linear systems solves

Input:
- Hierarchy of matrices X_l for $l = 0, \ldots, L$
- hyperparameters $\alpha_{\mathbf{e}}$, $\alpha_{\mathbf{v}}$, $\alpha_{\mathbf{u}}$, $\beta_{\mathbf{e}}$, $\beta_{\mathbf{v}}$ and $\beta_{\mathbf{u}}$

Take H_0 samples $\mathbf{b}_0^{(h)}$ on the **coarsest level** $l = 0$ using the Noise Injection Algorithm 3.1

$\mathbb{E}[\mathbf{y}_0] \approx X_L \prod_{k=0}^{L-1} P_{k+1} \left(\frac{1}{H_0} \sum_{h=1}^{H_0} \mathbf{b}_0^h \right)$

Sample on the **remaining levels**

for $l = 1, \ldots, L$ **do**

  interpolate $\mathbf{b}_l = P_l \mathbf{b}_{l-1}^{(H_{l-1})}$

  for $h = 1, \ldots, H_l$ **do**

    sample τ, $\lambda_{\mathbf{v}}$ and $\lambda_{\mathbf{u}}$ from (3.4), (3.5) and (3.6)

    generate $\mathbf{e}_1 = \mathcal{N}(\mathbf{0}, \tau^{-1} I)$ and $\mathbf{e}_2 = \mathcal{N}(\mathbf{0}, \Lambda)$ using (3.2)

    sample $\mathbf{b}_l^{(h)}$ by solving (3.3) using X_l

    restrict $\mathbf{e}_{2,l-1} = P_l^T \mathbf{e}_{2,l}$

    Using X_{l-1} and $\mathbf{e}_{2,l-1}$ sample $\mathbf{b}_{l-1}^{(h)}$ by solving 3.3 reusing τ, $\lambda_{\mathbf{v}}$, $\lambda_{\mathbf{u}}$ and $\mathbf{e}_1$

    store difference $\mathbf{d}_l^{(h)} = \mathbf{b}_l^{(h)} - P_l \mathbf{b}_{l-1}^{(h)}$

  end for

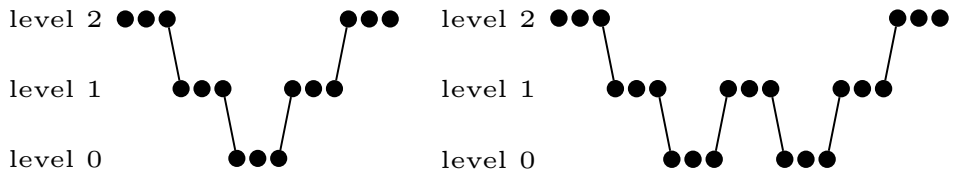
  $\mathbb{E}[\mathbf{y}_l - \mathbf{y}_{l-1}] \approx X_L \prod_{k=l}^{L-1} P_{k+1} \left(\frac{1}{H_l} \sum_{h=1}^{H_l} \mathbf{d}_l^{(h)} \right)$

end for

return $\mathbb{E}[\mathbf{y}] \approx \mathbb{E}[\mathbf{y}_0] + \sum_{l=1}^{L} \mathbb{E}[\mathbf{y}_l - \mathbf{y}_{l-1}]$

Algorithms 4.1 and 5.1 start sampling at the coarsest level and then refine until the finest level. This means that all the samples on each level are drawn consecutively. In contrast to multilevel Monte Carlo, the samples of a Markov Chain are not i.i.d. and consecutive samples are correlated. Instead of taking all the samples at once for each level, it is possible to take a few samples at one level, then move to another level and take a few samples before changing again to another level. Figure 5.1 shows examples of traversing the levels. Figure 5.1a details a V-cycle with three samples per level and figure 5.1b a W-cycle. Restricting the previous sample $\mathbf{b}_l$ from a finer level to a coarser level $\mathbf{b}_{l-1}$ is done using $\mathbf{b}_{l-1} = P_l^T \mathbf{b}_l$.

6. Experiments. We have investigated the described multilevel sampling schemes for Bayesian regression. Using real data X but simulated $\mathbf{b}$ and $\mathbf{y}$, the model was trained with $\mathbf{y}_{\text{train}} = X_{\text{train}} \mathbf{b} + \mathbf{e}$ and evaluated using $\mathbf{y}_{\text{test}} = X_{\text{test}} \mathbf{b}$ with X_{train} and X_{test} determined by 5-fold cross-validation. The characteristics of the different data matrices X are given in Table 6.1. Most are available online as indicated. Generating



(a) three level V-cycle (b) three level W-cycle

Fig. 5.1: Three level V-cycle and W-cycle with three samples after each change of level.

normal distributed $\mathbf{b} \sim \mathcal{N}(0, 10I_F)$ and $\mathbf{e} \sim \mathcal{N}(0, 10^3 I_N)$, $\mathbf{y}_{\text{train}}$ was simulated.

The application of the ChEMBl data is chemogenomics. They predict the activity of chemical compounds on certain proteins. More precisely, one of the indicators for drug-protein activity is the IC50 value. IC50 or the half maximal inhibitory concentration measures the substance concentration required to inhibit the activity of a protein by 50%. The bioactivity database ChEMBL version 19 [1] is publicly available. These chemical compounds are modeled by combinations of molecules. These chemical substructures are described by the extended-connectivity fingerprints (ECFP [30]) and were computed using rdkit [27] with 3 layers. More details can be found in [35].

The single-nucleotide polymorphism (SNP) data sets are used in genomic prediction for animals or plants [6]. This application uses a linear mixed model (3.1). The matrix Z was simulated using AlphaMPSim [19] and the random effects $\mathbf{u}$ represent the SNP marker effects. The number of SNP markers were set to 10 000, 50 000 and 100 000 resulting in three different data sets. Only the overall mean was modeled for the fixed effects resulting in $W = \mathbf{1}$. For the SNP data sets, the observations $\mathbf{y}$ were available and used.

Name	Source	value type	#rows	#columns	#nonzeros	fillin(%)
Trec10	[7]	double	106	478	8612	17.00
CNAE	[24]	double	1080	856	7233	0.78
micromass	[24]	double	360	1300	48 713	10.41
DrivFace	[24]	double	606	6400	3 878 400	100.00
arcene	[24]	double	100	10 000	540 941	54.09
SNP1	[19]	short int	10 000	10 000	57 641 064	57.64
tmc2007	SIAM 2007 Text Mining competition	double	21 519	30 438	2 283 179	0.35
nlpdata	[25]	double	31 572	34 023	2 277 757	0.21
rcv1_multi	[23]	double	15 564	47 236	1 028 284	0.14
SNP2	[19]	short int	10 000	50 000	282 212 533	56.44
news20	[21]	double	15 935	62 061	1 272 569	0.13
SNP3	[19]	short int	10 000	100 000	570 261 477	57.03
E2006	[22]	double	16 087	150 360	19 971 015	0.83
ChEMBL	[1]	binary	167 668	291 714	12 246 376	0.03

Table 6.1: The characteristics of the data matrices X used in the experiments.

Leader-follower clustering [14] was recursively used to cluster features, resulting in a hierarchy of data matrices. The performance of clustering algorithmns is often data dependent and Leader-follower clustering has the advantage that the algorithm

only passes trough the data once as detailed in [34]. Table 6.2 shows the number of levels and the corresponding feature sizes on the different levels for the used data sets. Leader-Follower clustering uses a threshold and passes trough the data and assigns data points to a cluster if the point is within the threshold. For the data sets with 10 000 are more features the coarse size was defined in the range $[3000, 8500]$ since computing the singular value decomposition is feasible in this range. The threshold can be selected by starting with a large threshold, for which leader-follower clustering is computed fast, and then increased until the coarse size is within the defined range.

Name	Number of levels	Feature sizes
Trec10	3	[181, 319, 478]
CNAE	3	[198, 354, 856]
micromass	3	[542, 754, 1300]
DrivFace	3	[2061, 3907, 6400]
arcene	3	[3341, 5933, 10000]
SNP1	3	[3582, 5178, 10000]
tmc2007	3	[5881, 18709, 30438]
nlpdata	3	[5258, 12095, 34023]
rcv1_multi	3	[6858, 18363, 47236]
SNP2	4	[7647, 15764, 28326, 50000]
news20	4	[5922, 17811, 31000, 62061]
SNP3	4	[8235, 21608, 44928, 100000]
E2006	4	[7647, 28326, 78091, 150360]
ChEMBL	6	[5664, 17019, 50382, 112612, 170223, 291714]

Table 6.2: The number of levels and average feature sizes for the different data sets used in the experiments.

Abbreviation	Description
Gibbs	Noise injection Gibbs sampler given in Algorithm 3.1
ML-G	Noise injection Multilevel (ML) Gibbs sampler (G) given in Algorithm 4.1
MLCSS-G	Noise injection Multilevel (ML) Gibbs sampler (G) with correlated samples (CS) using linear solves (S) given in Algorithm 5.1
MLCSP-G	Noise injection Multilevel (ML) Gibbs sampler (G) with correlated samples (CS) using projections (P) as detailed in (5.3)
MLMLP-G	Noise injection Multilevel (ML) Multilevel-preconditioned (MLP) Gibbs sampler (G) given in Algorithm 4.1
MLMLPCSS-G	Noise injection Multilevel (ML) Multilevel-preconditioned (MLP) Gibbs sampler (G) with correlated samples (CS) using linear solves (S) given in Algorithm 5.1
MLMLPCSP-G	Noise injection Multilevel (ML) Multilevel-preconditioned (MLP) Gibbs sampler (G) with correlated samples (CS) using projections (P) as detailed in (5.3)
W-cycle(30)	W-cycle level sample configuration with 30 samples at each level before changing to the next level.

Table 6.3: Abbreviations.

The level hierarchy created by clustering can be used as a two-level preconditioner to accelerate the convergence of CG [34]. This generally results in lower execution times for solving the linear system in the sampling process. This does, however,

requires the computation of the singular value decomposition on the coarsest level. We investigated both CG as solver on each level and CG accelerated by a two-level preconditioner with two iterations of CG as pre-smoothing for the multilevel sampling [34]. Table 6.3 shows the abbreviations of all the possible different samplers in the experiments.

Note that sampling distributions of τ, $\lambda\mathbf{v}$, $\lambda\mathbf{u}$ in equations (3.4), (3.5) and (3.6) are determined by the solution $\mathbf{b}$ found by solving the linear system. As a result, the sampling process is influenced by the underlying solver. Note that CG has a natural regularization role determined by the required tolerance or maximum number of iterations. Solving exactly on the coarse level can lead to different solutions for better or worse. Especially when using the correlated samples with system solves (MLCSS or MLMLPCSS) on the adjacent levels, the solvers on both levels should be of the same nature, e.g. iterative and thus not exact. If this not the case, the solutions on both levels can be too different, resulting in noise for the difference $\mathbf{b}_l^{(h)} - P_l\mathbf{b}_{l-1}^{(h)}]$ in (5.2). In the presented experiments, CG was always used on the coarsest level for fair comparison with all the other methods. Note that using an exact solver on the coarsest level for multilevel sampling without correlated samples is possible and would lead to even faster execution time with slightly different solutions.

Table 6.4 provides the average setup time, execution time, speed-up, pearson correlation ρ, root mean square error (RMSE) and mean absolute error (MAE) for the given data sets using the noise injection Gibbs sampler (Gibbs), the multilevel Gibbs sampler (ML-G), the multilevel preconditioned multilevel Gibbs sampler (MLMLP-G) and the multilevel preconditioned multilevel Gibbs sampler with correlated samples using systems solves (MLMLPCSS-G) on the test set. 2200 Samples were taken with 200 discarded as burn-in. Non-informative priors were used setting $\alpha_{\mathbf{e}} = 1.0$, $\beta_{\mathbf{e}} = 1.0$, $\alpha_{\mathbf{v}} = 1.0$, $\beta_{\mathbf{v}} = 10^{-3}$, $\alpha_{\mathbf{u}} = 1.0$. The burn-in samples of the multilevel samplers were taken at the coarsest level and the remaining samples were taken consecutively and equally distributed on the different levels starting at the coarsest level. The experiments were implemented in C++[1] and compiled with gcc 9.3.0 and OPENMP 4.0 with compile option -O3. The experiments were run on a machine with an Intel(R) Core(TM) i7-6560U (2.20GHz) processor with an L3 cache memory of 4096 KB and 16 GB of DRAM where 3 out of 4 cores were used. The reported values for the different experiments are the average of 5-fold cross-validation.

As can be seen from Table 6.4 taking all the samples on the fine level (Gibbs) often results in the slightly better predictions. Using the multilevel sampling techniques result in small loss of accuracy, but for almost all data sets within range of the standard deviation. The finest level does contain all of the information, so it is natural that this leads to the best predictions. The fine level does, however, contain more noise components which are less present in the coarser level and the multilevel approach can achieve better accuracy as is the case for the SNP3 data set. Note that for the SNP3 data set, there is a significant improvement using the multilevel sampling schemes.

[1] https://scm.cs.kuleuven.be/scm/git/multilevel_macau

		Gibbs	ML-G	MLMLP-G	MLMLPCSS-G
Trec10	setup (s)		1.70E-2	2.48E-2	2.40E-2
	exec. (s)	4.99	4.44	**2.92**	5.89
	speed-up	1.0	1.12	**1.71**	0.85
	ρ	**0.765** (0.106)	0.755 (0.107)	0.755 (0.107)	0.697 (0.159)
	RMSE	**6.44E2** (2.98E2)	6.55E2 (3.00E2)	6.55E2 (3.01E2)	7.24E2 (2.57E2)
	MAE	**4.06E2** (1.53E2)	4.19E2 (1.54E2)	4.19E2 (1.54E2)	4.68E2 (1.31E2)
CNAE	setup (s)		2.84E-2	1.07E-1	1.18E-1
	exec. (s)	8.96E-1	7.88E-1	5.22	8.87
	speed-up	1.0	**1.14**	0.17	0.10
	ρ	**0.647** (0.035)	0.644 (0.034)	0.644 (0.034)	0.643 (0.036)
	RMSE	**6.59** (4.63E-1)	6.60 (4.42E-1)	6.60 (4.42E-1)	6.62 (4.37E-1)
	MAE	4.97 (3.37E-1)	4.97 (3.50E-1)	4.97 (3.50E-1)	4.98 (3.54E-1)
micromass	setup (s)		9.59E-2	1.61E-1	1.75E-1
	exec. (s)	4.64E1	4.22E1	1.56E1	2.66E1
	speed-up	1.0	1.10	**2.97**	1.74
	ρ	0.984 (0.001)	0.984 (0.001)	0.984 (0.001)	0.984 (0.001)
	RMSE	**3.33E7** (3.93E5)	3.35E7 (4.16E5)	3.41E7 (5.49E5)	3.41E7 (5.41E5)
	MAE	**2.18E7** (3.16E5)	2.21E7 (3.34E5)	2.26E7 (3.80E5)	2.27E7 (3.80E5)
DrivFace	setup (s)		1.14E1	1.17E1	1.16E1
	exec. (s)	1.13E3	6.69E2	2.40E2	3.88E2
	speed-up	1.0	1.69	**4.71**	2.91
	ρ	0.974 (0.005)	0.975 (0.005)	0.975 (0.005)	0.974 (0.005)
	RMSE	9.41 (1.49)	**9.36** (1.54)	**9.36** (1.5)	9.40 (1.58)
	MAE	**6.99** (9.55E-1)	7.00 (1.00)	7.00 (1.0)	**6.99** (1.07)
SNP1	setup (s)		1.95E2	1.18E3	1.15E3
	exec. (s)	6.18E3	5.86E3	3.24E3	5.29E3
	speed-up	1.0	1.05	**1.91**	1.17
	ρ	0.958 (0.003)	0.958 (0.003)	0.958 (0.003)	0.958 (0.003)
	RMSE	1.47E-1 (4.16E-3)	1.47E-1 (4.20E-3)	1.47E-1 (4.19E-3)	1.47E-1 (3.84E-3)
	MAE	1.16E-1 (3.48E-3)	1.16E-1 (3.56E3)	1.16E-1 (3.55E-3)	1.16E-1 (3.26E-3)
tmc2007	setup (s)		8.00	1.50E3	1.49E3
	exec. (s)	2.43E3	1.30E3	4.95E2	7.59E2
	speed-up	1.0	1.87	**4.91**	3.20
	ρ	**0.988** (0.000)	0.984 (0.001)	0.982 (0.001)	0.985 (0.001)
	RMSE	**4.21E-1** (3.36E-3)	4.97E-1 (8.99E-3)	5.32E-1 (7.47E-3)	4.79E-2 (4.91E-3)
	MAE	**3.06E-1** (3.02E-3)	3.71E-1 (7.25E-3)	4.00E-1 (5.89E-3)	3.57E-1 (4.00E-3)

		Gibbs	ML-G	MLMLP-G	MLMLPCSS-G
nlpdata	setup (s)		7.09	1.91E3	1.93E3
	exec. (s)	3.23E2	2.71E2	1.98E2	3.35E2
	speed-up	1.0	1.19	**1.63**	0.96
	ρ	0.955 (0.014)	0.954 (0.015)	0.955 (0.015)	0.955 (0.014)
	RMSE	**7.09E1** (6.15)	7.24E1 (8.62)	7.20E1 (8.19)	7.12E1 (6.92)
	MAE	**3.59E1** (8.56E-1)	3.60E1 (9.16E-1)	3.60E1 (8.95E-1)	**3.59E1** (8.68E-1)
rcv_multi	setup (s)		5.51	3.25E3	3.24E3
	exec. (s)	1.11E3	5.60E2	1.39E3	1.98E3
	speed-up	1.0	**1.98**	0.80	0.56
	ρ	**0.893** (0.002)	0.883 (0.002)	0.882 (0.002)	0.838 (0.004)
	RMSE	**1.35** (1.59E-2)	1.41 (1.99E-2)	1.42 (1.95E-2)	1.64 (1.25E-2)
	MAE	**8.96E-1** (4.9E-3)	9.75E-1 (1.34E-2)	9.83E-1 (1.37E-2)	1.17 (1.09E-2)
SNP2	setup (s)		1.53E3	2.99E3	2.97E3
	exec. (s)	4.34E4	1.86E4	8.91E3	1.41E3
	speed-up	1.0	2.33	**4.87**	3.07
	ρ	0.959 (0.002)	**0.961** (0.002)	**0.961** (0.002)	0.960 (0.002)
	RMSE	1.57E-1 (2.00E-3)	**1.52E-1** (2.15E-3)	**1.52E-1** (2.16E-3)	1.54E-1 (2.22E-3)
	MAE	1.24E-1 (1.95E-3)	**1.20E-1** (2.00E-3)	**1.20E-1** (2.01E-3)	1.22E-1 (1.90E-3)
news20	setup (s)		9.83	1.65E3	1.65E3
	exec. (s)	7.22E3	3.59E3	4.22E3	6.75E3
	speed-up	1.0	**2.01**	1.71	1.07
	ρ	**0.912** (0.017)	0.899 (0.015)	0.899 (0.015)	0.892 (0.007)
	RMSE	**2.87E1** (8.39)	3.10E1 (1.05E1)	3.10E1 (1.05E1)	3.17E1 (9.14)
	MAE	**1.27E1** (5.50E-1)	1.33E1 (5.95E-1)	1.33E1 (5.95E-1)	1.43E1 (5.31E-1)
SNP3	setup (s)		1.61E4	2.20E4	2.14E4
	exec. (s)	1.38E5	3.85E4	1.78E4	2.78E4
	speed-up	1.0	3.58	**7.75**	4.96
	ρ	0.955 (0.002)	**0.962** (0.002)	**0.962** (0.002)	0.961 (0.002)
	RMSE	1.99E-1 (3.51E-3)	**1.79E-1** (5.14E-3)	**1.79E-1** (5.15E-3)	1.84E-1 (4.94E-3)
	MAE	1.57E-1 (3.55E-3)	**1.41E-1** (4.84E-3)	**1.41E-1** (4.84E-3)	1.45E-1 (4.88E-3)
E2006	setup (s)		2.02E2	2.76E3	2.75E3
	exec. (s)	7.76E3	4.12E3	1.77E3	2.85E3
	speed-up	1.0	1.88	**4.38**	2.72
	ρ	**0.962** (0.002)	0.959 (0.002)	0.959 (0.002)	0.961 (0.002)
	RMSE	**7.88E-2** (2.56E-3)	8.11E-2 (2.76E-3)	8.21E-2 (2.53E-3)	8.01E-2 (2.59E-3)
	MAE	**4.96E-2** (8.91E-3)	5.160E-2 (8.02E-4)	5.27E-2 (8.18E-4)	5.10E-2 (1.31E-3)
ChEMBL	setup (s)		4.24E2	4.63E3	4.74E3
	exec. (s)	1.72E3	1.99E3	1.35E3	2.44E3
	speed-up	1.0	0.86	**1.27**	0.70
	ρ	**0.772** (0.005)	0.766 (0.004)	0.766 (0.004)	0.768 (0.004)
	RMSE	**1.54E1** (1.14E-1)	1.57E1 (9.32E-2)	1.57E1 (9.58E-2)	1.56E1 (1.05E-1)
	MAE	**1.22E1** (8.05E-2)	1.23E1 (6.38E-2)	1.24E1 (6.53E-2)	1.23E1 (7.47E-2)

Table 6.4: Average setup time (setup), execution time (exec.), pearson correlation (ρ), root-mean-square error (RMSE) and mean absolute error (MAE) in scientific notation for the given data sets using Gibbs, ML-G, MLMLP-G and MLMLPCSS-G. The standard deviation for the accuracy measures is given within round brackets.

The speed-up achieved by the multilevel schemes varies for the different data sets. Part of the speed-up depends on the use of preconditioning. Using the preconditioner results in larger setup times due to the calculation of the SVD but generally results in faster execution times since the linear systems are solved more efficiently. For the ChEMBL data the multilevel sampling without preconditioning is slower than taking all the samples on the fine level due to the fact that the nonzeros are binary on the fine level and floating point numbers on the coarser levels. The multilevel sampling with correlated samples results in better accuracy for some data sets but not consistently. The execution time for the correlated samples with linear systems (MLMLPCSS-G) has increased since, for each sample, a linear system has to be solved on two adjacent levels.

Since taking samples at the coarser levels is cheaper, more samples can be taken at the coarse level in the same execution time as at the finest level. Table 6.5 shows the number of samples taken in different stages for 3 and 4 levels. Figure 6.1 shows the RMSE for four data sets in function of the execution time using the stages in Table 6.5.

Stage	$[H_0, H_1, H_2]$	$[H_0, H_1, H_2, H_3]$	H
1	[2, 2, 2]	[2, 2, 2, 2]	2
2	[8, 4, 2]	[16, 8, 4, 2]	4
3	[16, 8, 4]	[32, 16, 8, 4]	16
4	[32, 16, 8]	[64, 32, 16, 8]	64
5	[64, 32, 16]	[128, 64, 32, 16]	128
6	[128, 64, 32]	[256, 128, 64, 32]	256
7	[256, 128, 64]	[512, 256, 128, 64]	512
8	[512, 256, 128]	[1024, 512, 256, 128]	1024

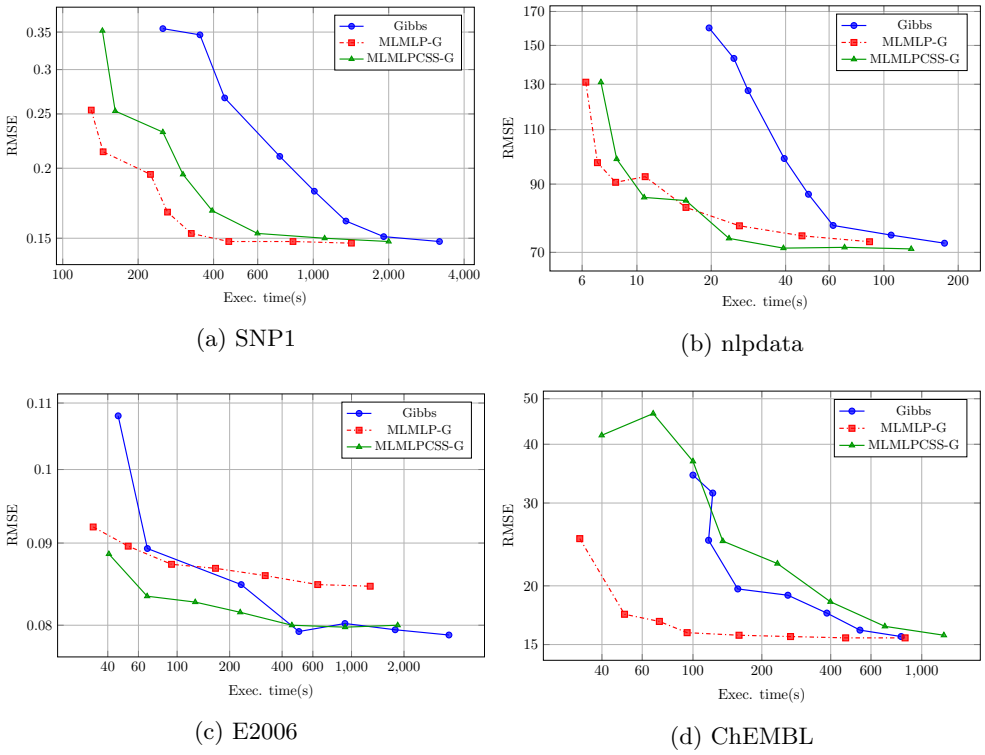
Table 6.5: The increasing number of samples H_l for the different levels

Figure 6.1 shows that the multilevel sampler (MLMLP-G) achieves better accuracy faster. Additionally the multilevel sampler converges faster than only sampling at the fine level (Gibbs) in Figures 6.1a, 6.1b and 6.1d. For the E2006 data set the convergence is fast for all samplers. For the ChEMBL data, only the 4 coarsest levels were used and no samples were taken at the two finest levels. The multilevel sampler with correlated samples (MLMLPCSS-G) results in slightly better accuracy for two data sets than without correlated samples but requires more execution time.

6.1. Samples configuration. For the results in Table 6.4, the samples are taken consecutively on each level. One can easily change levels while building the chain and Table 6.6 shows the number of samples taken at each level for different configurations using 3 and 6 levels.

Figure 6.2 now shows the RMSE for the different configurations in Table 6.6. As can be seen from Figure 6.2, more than 10 samples should be taken before changing to another level. It takes a few, generally less than 3, samples for the Markov Chain to adopt to the new level. This can easily be incorporated by using a burn-in after a level change or by means of thinning. If enough samples are taken before each level change, one can achieve the same accuracy as if the samples were taken consecutively and equally distributed over the levels with the remark that when using these configurations less samples are taken on the fine level.

As seen from Table 6.6 and Figure 6.2, it is possible to achieve high accuracy without taking much samples at the finest and thus expensive levels. For multilevel Markov chain Monte Carlo involving PDEs, one can determine the optimal number



(a) SNP1

(b) nlpdata

(c) E2006

(d) ChEMBL

Fig. 6.1: The average RMSE in function of the execution time in seconds for SNP1, nlpdata, E2006 and ChEMBL for Gibbs sampling, MLMLP-G and MLMLPCSS-G.

Config	3 levels	6 levels
V-cycle(3)	[498, 999, 501]	[201, 399, 399, 399, 399, 201]
V-cycle(10)	[500, 1000, 500]	[200, 400, 400, 400, 400, 200]
V-cycle(30)	[480, 990, 510]	[210, 390, 390, 390, 390, 210]
V-cycle(100)	[500, 1000, 500]	[200, 400, 400, 400, 400, 200]
W-cycle(3)	[666, 999, 333]	[522, 783, 387, 189, 90, 27]
W-cycle(10)	[670, 1000, 330]	[530, 790, 390, 190, 80, 20]
W-cycle(30)	[660, 990, 330]	[570, 840, 390, 150, 30, 0]
W-cycle(100)	[700, 1000, 300]	[600, 900, 400, 100, 0, 0]

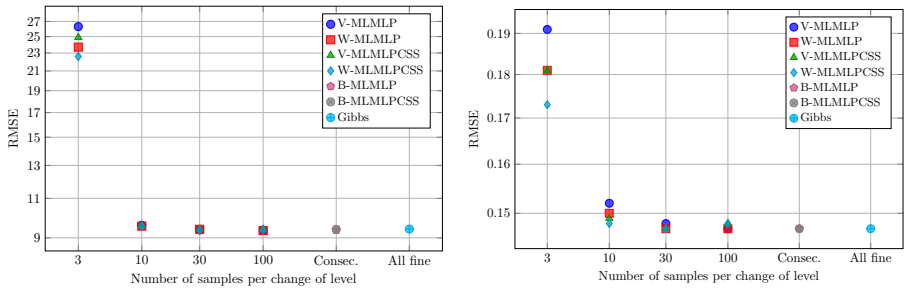
Table 6.6: The number of samples H_l given as $[H_0, \ldots, H_L]$ for the different configurations used in the experiments.

of samples on each level [8]. The number of samples on level l is given by

$$H_l = \frac{2}{\epsilon^2} \left(\sum_{l=0}^{L} \sqrt{s_l^2 C_l} \right) \sqrt{\frac{s_l^2}{C_l}} \tag{6.1}$$

with C_l the cost to take one sample at level l, s_l^2 the sample variance at level l and ϵ the required tolerance on the RMSE of the quantity of interest.

In contrast, we are interested in the predictions for new and unknown data. We actually do not know our quantity of interest. One could optimize the number of



(a) DrivFace (b) SNP1

(c) nlpdata (d) ChEMBL

Fig. 6.2: RMSE given in function of different samples configurations as given in Table 6.6 for DrivFace, SNP1, nlpdata and ChEMBL using MLMLP-G and MLMLPCSS-G. The reference values for taking H_l consecutively (consec.) and Gibbs sampling from Table 6.4 are additionally given.

samples based on the training set, but these are noisy observations. A closer look at equation (6.1) shows that the factor $\frac{2}{\epsilon^2}\left(\sum_{l=0}^{L}\sqrt{s_l^2 C_l}\right)$ is the same for all levels and the factor $\sqrt{\frac{s_l^2}{C_l}}$ is level specific. The latter can thus be used to define the ratio between the number of samples H_l at each level l. Using this factor we define the number of samples at level l as

$$H_l = \left\lfloor \frac{\sqrt{\frac{s_l^2}{C_l}}}{\left(\sum_{l=0}^{L}\sqrt{\frac{s_l^2}{C_l}}\right)} H \right\rfloor \tag{6.2}$$

with H the given total number samples over all levels and s_l^2 the sample variance of the average of the observations $\mathbf{y}$. The cost C_l to take one sample is defined as the number of nonzeros of X_l. Definition 6.2 is based on the fact that the variance of the difference $\mathbb{V}[\mathbf{y}_l - \mathbf{y}_{l-1}]$ is smaller than the variance $\mathbb{V}[\mathbf{y}_l]$ on level l due to positive correlation. As an alternative we considered a ratio purely based on the cost for taking

one sample for the multilevel sampler without correlated samples

$$H_l = \left\lfloor \frac{\frac{1}{C_l}}{\left(\sum_{l=0}^{L} \frac{1}{C_l}\right)} H \right\rfloor \tag{6.3}$$

with C_l and H as before.

		DrivFace	nlpdata	SNP2	news20	ChEMBL
Execution time(s)	Gibbs	1.13E3	3.23E2	4.34E4	7.22E3	1.72E3
	MLMLP	**2.40E2**	1.98E2	8.91E3	4.22E3	1.35E3
	MLMLPCSS	3.88E2	3.35E2	1.41E4	6.75E3	2.44E3
	Var-MLMLP	2.43E2	**1.97E2**	8.80E3	2.87E3	1.29E3
	Cost-MLMLP	2.49E2	2.09E2	**4.51E3**	8.89E3	1.30E3
	Var-MLMLPCSS	3.05E2	2.29E2	9.86E3	**1.87E3**	**1.00E3**
	Cost-MLMLPCSS	3.83E2	3.41E2	1.25E4	7.00E3	2.30E3
RMSE	Gibbs	9.41	7.09E1	1.57E-1	2.87E1	1.54E1
	MLMLP	9.36	7.20E1	**1.52E-1**	3.10E1	1.57E1
	MLMLPCSS	9.40	7.12E1	1.54E-1	3.17E1	**1.56E1**
	Var-MLMLP	9.42	7.78E1	1.53E-1	3.32E1	**1.56E1**
	Cost-MLMLP	**9.32**	7.19E1	**1.52E-1**	**3.06E1**	**1.56E1**
	Var-MLMLPCSS	9.49	7.25E1	1.55E-1	3.30E1	2.09E1
	Cost-MLMLPCSS	9.40	**7.08E1**	1.54E-1	3.32E1	**1.56E1**

Table 6.7: The average execution time (s)and RMSE using different data sets for multilevel sampling with and without correlated samples with the number of samples H_l based on the variance (Var) of the levels in equation (6.2) (Var) or the number of nonzeros of X_l (Cost) in equation (6.3). The values for Gibbs, MLMLP-G and MLMLPCSS with the samples equally distributed over levels from Table 6.4 are provided as reference. The best values for the multilevel sampling schemes are presented in bold.

Table 6.7 details the average execution time and the RMSE for the multilevel sampler with and without correlated samples with the number of samples H_l for each level l determined by (6.2) or (6.3). Using the variance of the average predictions $\mathbf{y}$ on level l combined with the correlated samples performs worse than using only the cost of taking one sample on level l. The sample variance was calculated using 50 samples on each level and then the number of total samples on each level was calculated using (6.2). Using the cost based distribution of number of samples H in equation (6.3) results in more samples at the coarser levels and generally without loss in accuracy with respect to uniformly distributing the number of samples H over the levels in MLMLP. This results in slightly faster execution time.

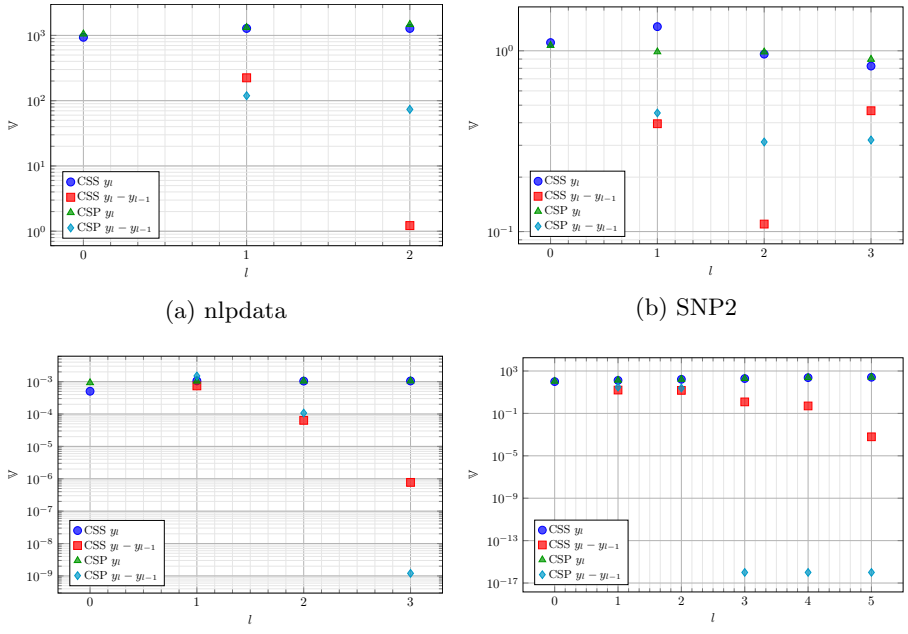
6.2. Correlated samples within the Gibbs sampler. As shown in previous results, using correlated samples can improve accuracy but does increase the execution time since each sample requires two linear solves on the adjacent levels. As mentioned in Section 5, there is a second way to define correlation between samples. The second approach simply projects the finer solution to the coarser level. This could lead to problems when sampling from different distributions for the same level depending on the quality of the clustering. If all features within one cluster were the same, this would not be a problem. In reality this is almost never true. Table 6.8 shows the execution time, pearson correlation and RMSE for correlated samples using two linear solves (CSS) and projecting the fine level solution to the coarser level (CSP).

For almost all data sets, the accuracy of CSP, although faster, is significantly less than CSS.

Data	Execution time(s)			Pearson Correlation			RMSE		
	Gibbs	CSS	CSP	Gibbs	CSS	CSP	Gibbs	CSS	CSP
DrivFace	1.13E3	3.88E2	2.65E2	0.974	0.974	0.974	9.41	9.40	9.50
nlpdata	3.23E2	3.35E2	2.07E2	0.955	0.955	0.955	7.09E1	7.12E1	7.11E1
SNP2	4.34E4	9.43E3	6.45E3	0.959	0.960	0.958	1.57E-1	1.54E-1	1.60E-1
news20	7.22E3	6.75E3	4.25E3	0.912	0.892	0.859	2.87E1	3.17E1	3.68E1
E2006	7.76E3	2.85E3	1.77E3	0.962	0.961	0.957	7.88E-2	8.01E-2	8.33E-2
ChEMBL	1.72E3	2.44E3	1.39E3	0.772	0.768	0.766	1.54E1	1.56E1	1.57E1

Table 6.8: The average execution time (s), pearson correlation and RMSE using different data sets for Gibbs sampling and multilevel sampling using correlated samples by linear solves (CSS) and Projections (CSP). Using CSP performs worse than CSS for almost all data sets.

Figure 6.3 shows the variances for each level l and variance of the difference of the correlated samples for observation $\mathbf{y}(0)$ for CSS and CSP. The figure shows that, as expected, the variance of the difference on level l and $l-1$ is smaller than the variance of level l. For the ChEMBL data, the finer levels for CSP do not contain additional information with respect to the coarser levels to aid in the prediction of $\mathbf{y}(0)$ resulting in zero variance.



(a) nlpdata

(b) SNP2

(c) E2006

(d) ChEMBL

Fig. 6.3: The variance $\mathbb{V}[\mathbf{y}_l]$ of level l for $l = 0, \ldots, L$ and the variance $\mathbb{V}[\mathbf{y}_l - \mathbf{y}_{l-1}]$ of the difference for $l = 1, \ldots, L$ on the training data of nlpdata, SNP2, E2006 and ChEMBL. The results are given using correlated samples with linear solves (CSS) and projections (CSP).

7. Conclusion. We have presented a multilevel Gibbs sampler for Bayesian regression for linear mixed models. Using clustering algorithms, a hierarchy of data matrices is created. This hierarchy allows to sample on different levels reducing execution time without significant loss in predictive performance for almost all data sets. Since the coarser levels contain most of the variance of the data, more samples can be taken at the coarser levels and the chain converges faster than taking all the samples at the finest level. Distributing the number of samples over the levels based on the cost to take one level sample was shown to provide fast and accurate results.

Furthermore, we investigated the use of correlated samples to reduce the variance of the Markov Chain. The use of correlated samples increased the execution time but did not consistently improve the accuracy of the predictions. The distribution of the number of samples across the levels based on the variance performed less optimal than distribution based on sampling cost alone. The multilevel sampler with correlated samples was often still faster than plain sampling on the fine level and by storing both the solutions of level l and the difference between the levels l and $l-1$, it is possible to get the predictions of both multilevel samplers as part of an ensemble.

8. Acknowledgement. This work was supported Research Foundation - Flanders (FWO, No. G079016N). We would like to thank David Vanavermaete for generating the SNP data sets.

REFERENCES

[1] A. P. Bento, A. Gaulton, A. Hersey, L. J. Bellis, J. Chambers, M. Davies, F. A. KrÃijger, Y. Light, L. Mak, S. McGlinchey, M. Nowotka, G. Papadatos, R. Santos, and J. P. Overington, *The ChEMBL bioactivity database: an update*, Nucleic Acids Research, 42 (2014), pp. D1083–D1090.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag New York, 2006.

[3] W. Briggs, V. Henson, and S. McCormick, *A Multigrid Tutorial, Second Edition*, Society for Industrial and Applied Mathematics, second ed., 2000, https://doi.org/10.1137/1.9780898719505, https://epubs.siam.org/doi/abs/10.1137/1.9780898719505, https://arxiv.org/abs/https://epubs.siam.org/doi/pdf/10.1137/1.9780898719505.

[4] J. A. Christen and C. Fox, *Markov Chain Monte Carlo using an approximation*, Journal of Computational and Graphical Statistics, 14 (2005), pp. 795–810, https://doi.org/10.1198/106186005X76983, https://doi.org/10.1198/106186005X76983.

[5] K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, Computing and Visualization in Science, 14 (2011), p. 3.

[6] A. D. Coninck, J. Fostier, S. Maenhout, and B. De Baets, *Dairry-blup: A high-performance computing approach to genomic prediction*, Genetics, 197 (2014), pp. 813–822, https://doi.org/10.1534/genetics.114.163683, https://www.genetics.org/content/197/3/813, https://arxiv.org/abs/https://www.genetics.org/content/197/3/813.full.pdf.

[7] T. A. Davis and Y. Hu, *The university of florida sparse matrix collection*, ACM Transactions on Mathematical Software (TOMS), 38 (2011), p. 1.

[8] T. Dodwell, C. Ketelsen, R. Scheichl, and A. Teckentrup, *A hierarchical multilevel Markov Chain Monte Carlo algorithm with applications to uncertainty quantification in subsurface flow*, SIAM/ASA Journal on Uncertainty Quantification, 3 (2015), pp. 1075–1108, https://doi.org/10.1137/130915005, https://doi.org/10.1137/130915005, https://arxiv.org/abs/https://doi.org/10.1137/130915005.

[9] Y. Efendiev, T. Hou, and W. Luo, *Preconditioning Markov Chain Monte Carlo simulations using coarse-scale models*, SIAM Journal on Scientific Computing, 28 (2006), pp. 776–803, https://doi.org/10.1137/050628568, https://doi.org/10.1137/050628568.

[10] A. E. Gelfand and A. F. M. Smith, *Sampling-based approaches to calculating marginal densities*, Journal of the American Statistical Association, 85 (1990), pp. 398–409, https://doi.org/10.1080/01621459.1990.10476213, https://amstat.tandfonline.com/doi/abs/10.1080/01621459.1990.10476213.

[11] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin,

Bayesian data analysis, Chapman and Hall/CRC, 2013.

[12] S. Geman and D. Geman, *Stochastic relaxation, gibbs distributions, and the bayesian restoration of images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-6 (1984), pp. 721–741, https://doi.org/10.1109/TPAMI.1984.4767596.

[13] M. B. Giles, *Multilevel Monte Carlo path simulation*, Operations Research, 56 (2008), pp. 607–617, https://doi.org/10.1287/opre.1070.0496, https://doi.org/10.1287/opre.1070.0496, https://arxiv.org/abs/https://doi.org/10.1287/opre.1070.0496.

[14] J. A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, Inc., New York, NY, USA, 99th ed., 1975.

[15] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag New York, 2009.

[16] W. K. Hastings, *Monte Carlo sampling methods using Markov Chains and their applications*, Biometrika, 57 (1970), pp. 97–109, http://www.jstor.org/stable/2334940.

[17] C. R. Henderson, *Selection index and expected genetic advance*, Statistical genetics and plant breeding, 982 (1963), pp. 141–163.

[18] C. R. Henderson, O. Kempthorne, S. R. Searle, and C. Von Krosigk, *The estimation of environmental and genetic trends from records subject to culling*, Biometrics, 15 (1959), pp. 192–218.

[19] J. M. Hickey, G. Gorjanc, S. Hearne, and B. E. Huang, *Alphampsim: flexible simulation of multi-parent crosses*, Bioinformatics, 30 (2014), pp. 2686–2688, https://doi.org/10.1093/bioinformatics/btu206, http://dx.doi.org/10.1093/bioinformatics/btu206, https://arxiv.org/abs//oup/backfile/content_public/journal/bioinformatics/30/18/10.1093_bioinformatics_btu206/2/btu206.pdf.

[20] V. H. Hoang, C. Schwab, and A. M. Stuart, *Complexity analysis of accelerated MCMC methods for bayesian inversion*, Inverse Problems, 29 (2013), p. 085010, https://doi.org/10.1088/0266-5611/29/8/085010, https://doi.org/10.1088%2F0266-5611%2F29%2F8%2F085010.

[21] S. S. Keerthi and D. DeCoste, *A modified finite newton method for fast solution of large scale linear svms*, Journal of Machine Learning Research, 6 (2005), pp. 341–361.

[22] S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. A. Smith, *Predicting risk from financial reports with regression*, in Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, Colorado, June 2009, Association for Computational Linguistics, pp. 272–280, https://www.aclweb.org/anthology/N09-1031.

[23] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, *Rcv1: A new benchmark collection for text categorization research*, J. Mach. Learn. Res., 5 (2004), pp. 361–397, http://dl.acm.org/citation.cfm?id=1005332.1005345.

[24] M. Lichman, *UCI machine learning repository*, 2013, http://archive.ics.uci.edu/ml.

[25] *MATLAB statistics and machine learning toolbox*, 2018. The MathWorks, Natick, MA, USA.

[26] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *Equation of state calculations by fast computing machines*, The journal of chemical physics, 21 (1953), pp. 1087–1092.

[27] *RDKit: Open-source cheminformatics*. http://www.rdkit.org. [Online; accessed 18-Januari-2017].

[28] P. Robbe, D. Nuyens, and S. Vandewalle, *Recycling samples in the Multigrid multilevel (quasi-)Monte Carlo method*, SIAM Journal on Scientific Computing, 41 (2019), pp. S37–S60, https://doi.org/10.1137/18M1194031.

[29] C. Robert and G. Casella, *Monte Carlo statistical methods*, Springer-Verlag New York, 2004, https://doi.org/10.1007/978-1-4757-4145-2.

[30] D. Rogers and M. Hahn, *Extended-connectivity fingerprints*, Journal of Chemical Information and Modeling, 50 (2010), pp. 742–754. PMID: 20426451.

[31] J. Simm, A. Arany, P. Zakeri, T. Haber, J. K. Wegner, V. Chupakhin, H. Ceulemans, and Y. Moreau, *Macau: Scalable bayesian factorization with high-dimensional side information using MCMC*, in 2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP), Sept 2017, pp. 1–6, https://doi.org/10.1109/MLSP.2017.8168143.

[32] J. Simm, G. Klambauer, A. Arany, M. Steijaert, J. K. Wegner, E. Gustin, V. Chupakhin, Y. T. Chong, J. Vialard, P. Buijnsters, I. Velter, A. Vapirev, S. Singh, A. E. Carpenter, R. Wuyts, S. Hochreiter, Y. Moreau, and H. Ceulemans, *Repurposing high-throughput image assays enables biological activity prediction for drug discovery*, Cell Chemical Biology, 25 (2018), pp. 611 – 618.e3, https://doi.org/https://doi.org/10.1016/j.chembiol.2018.01.015, http://www.sciencedirect.com/science/article/pii/S2451945618300370.

[33] D. Sorensen and D. Gianola, *Likelihood, Bayesian, and MCMC methods in quantitative genetics*, Springer Science+Business Media New York, 1 ed., 2002, https://doi.org/10.1007/b98952.

[34] J. Tavernier, J. Simm, K. Meerbergen, and Y. Moreau, *Multilevel preconditioning for ridge regression*, CoRR, abs/1806.05826 (2018), http://arxiv.org/abs/1806.05826, https://arxiv.org/abs/1806.05826.

[35] J. Tavernier, J. Simm, K. Meerbergen, J. K. Wegner, H. Ceulemans, and Y. Moreau, *Fast semi-supervised discriminant analysis for binary classification of large data sets*, Pattern Recognition, 91 (2019), pp. 86 – 99, https://doi.org/https://doi.org/10.1016/j.patcog.2019.02.015, http://www.sciencedirect.com/science/article/pii/S0031320319300792.

[36] H. Vandecasteele and G. Samaey, *A micro-macro Markov chain Monte Carlo method for molecular dynamics using reaction coordinate proposals I: direct reconstruction*, 2020, https://arxiv.org/abs/2002.09324.

[37] P. S. Vassilevski, *Multilevel block factorization preconditioners: Matrix-based analysis and algorithms for solving finite element equations*, Springer-Verlag New York, 2008, https://doi.org/10.1007/978-0-387-71564-3.