Wageningen University

Hydrology and Environmental Hydraulics Group

Master Thesis

Earth and Environment

A comparison of Markov Chain Monte Carlo algorithms for parameter calibration in hydrology

Author:
Douwe Kamper

Supervisors:
Lieke Melsen, Syed Mustafa & Shota Gugushvili

May 19, 2025

Abstract

Calibration of model parameters is key for improving the reliability of model predictions and thereby informed decision making. Markov Chain Monte Carlo (MCMC) is a statistical method that has found widespread application in hydrological modelling for calibrating parameters and quantifying their uncertainty. Hundreds of MCMC algorithms have been designed, yet only a small selection is used by hydrologists, excluding some of the most popular algorithms in other fields of science, despite the absence of a comprehensive benchmark supporting this decision. In this study a contribution is made towards this benchmark, by comparing a single algorithm popular outside hydrology (Goodman and Weare's Affine Invariant sampler; AI) to two algorithms popular in hydrology (Ter Braak's DE and DE-SNK). Four simple steady-state synthetic groundwater flow models (developed using MODFLOW 6) were calibrated, with the number of calibrated parameters ranging from one to five. Performance diagnostics suggest that all three algorithms converge to the same posterior distribution for all models, but AI is distinctly less efficient than the others. DE-SNK overtakes DE for the model with the most calibrated parameters and is therefore recommended for groundwater modelling in general. In contrast, existing literature, based on vadose zone modelling, recommends AI based strategies below 10 calibrated parameters, implying the need to further study under which conditions which algorithm outperforms the others.

1 Introduction

Model predictions are at the heart of hydrological decision making, they for instance quantify the effect of groundwater abstraction on groundwater levels. An essential step to improve model predictions is calibration of the model parameters. Conventionally, calibration refers to the process of tuning model parameters to optimize the similarity between observed and simulated values, with the objective to find a unique set of optimal parameters, e.g. Refsgaard (1997), where the performance of a parameter set is evaluated with a goodness of fit diagnostic such as the Kling-Gupta Efficiency (Gupta et al., 2009). This is a challenging problem to solve, as the parameter space of hydrological models is characterized by many local optima (sometimes hundreds), which are not necessarily close to the global optimum (Duan et al., 1992). Historically, local calibration methods such as the widely used Levenberg-Marquardt algorithm (Beven and Binley, 1992) have been used for this purpose. Although computationally efficient, these local calibration methods do not attempt to explore all probable parts of parameter space. As a consequence, these local methods often converge to a local optimum, rather than the global optimum (Hendrickson et al., 1988; Blasone et al., 2007). Therefore, it is often recommended to restart optimization from different starting points multiple times, in the hope of arriving at a better optimum. However, if there are many local optima, this strategy is unlikely to succeed, while increasing the computational burden. Global search algorithms have been developed to avoid this problem, at the cost of generally being much slower. A prominent example is the Shuffled Complex Evolution (SCE-UA) method developed at the University of Arizona by Duan et al. (1992).

When making decisions based on model predictions, not only the prediction itself is important (global optimum), but also its uncertainty. For some purposes, such as flood forecasting, the uncertainty estimate may even be more important than the prediction itself. Although SCE-UA can reliably find the global optimum, it is inapplicable for parameter uncertainty assessment (Vrugt et al., 2003) due to its deterministic rather than probabilistic nature (Vrugt et al., 2006). Vrugt et al. (2003) addressed this by integrating SCE-UA into a probabilistic statistical method called Markov Chain Monte Carlo (MCMC), creating the Shuffled Complex Evolution Metropolis algorithm (SCEM-UA).

The advantage of MCMC with regard to parameter uncertainty estimation is that MCMC produces a probability distribution for each parameter, referred to as the posterior. Traditional MCMC algorithms such as Metropolis-Hastings achieve this as follows. Prior knowledge is formalized into a prior probability distribution for each parameter. Additionally, a likelihood function is selected, which quantifies the probability of the observed data given the (sampled) parameter value(s) of the hydrological model. With the prior and likelihood defined, sampling can start. For each parameter an initial value is determined from which step-wise exploration of parameter space starts. New locations are generated by a proposal distribution and accepted or rejected stochastically, based on an acceptance ratio, which is calculated with the prior and likelihood of the sampled parameter values. These subsequent dependent steps form a Markov chain. Sampling is conventionally split into two phases: during the burn-in phase the chain searches for the most probable areas of the parameter space and during the main sampling phase this area has been reached and is explored. Sampling is generally stopped when some arbitrary threshold of a selected convergence diagnostic is reached. The estimated posterior probability distribution is the distribution formed by by collecting all locations the Markov chain has sampled during the main sampling phase, where the probability equals the relative sampling density. The success of a MCMC algorithm largely depends on

Table 1: A comparison of the popularity of several MCMC algorithms between all fields of science and hydrology specifically. Popularity is quantified by counting how often the paper introducing the specific algorithms is cited. Three hydrological journals have been selected to indicate the popularity in hydrology: Journal of hydrology, Water Resources Research & Advances in Water Resources. These are the three journals where MCMC methods are most discussed, while specifically tailored to hydrology (Appendix A).

MCMC Method	citation in all journals	citations in hydrological journals	paper introducing algorithm
Metropolis-Hastings	7922	123	Hastings (1970)
Differential Evolution (DE)	565	32	Ter Braak (2006)
DE with snooker updater (DE-SNK)	338	53	Ter Braak and Vrugt (2008)
DREAM	777	196	Vrugt et al. (2009)
Multiple-Try DREAM$_{(zs)}$	360	93	Laloy and Vrugt (2012)
Affine Invariant sampler (AI)	1903	9	Goodman and Weare (2010)
Hamiltonian Monte Carlo (HMC)	2154	10	Duane et al. (1987)
No-U-Turn Sampler (NUTS)	1886	17	Hoffman et al. (2014)

whether its chain(s) get stuck in a subspace and on the speed by which they converge to this distribution.

Since the inception of MCMC with Random Walk Metropolis (Metropolis et al., 1953), hundreds of MCMC algorithms have been developed and introduced in scientific literature (Brooks et al., 2011). Two of these algorithms are very popular in hydrological literature (Table 1): Metropolis-Hastings and Differential Evolution Adaptive Metropolis (DREAM). Metropolis-Hastings was developed by Hastings (1970) and has an asymmetric proposal distribution and modified acceptance probability. Hastings (1970) solved the issue Random Walk Metropolis has with undersampling of parameter values near a parameter bound, which is caused by parameter bounds only being accessible from one side of the bound. The Metropolis-Hastings proposal distribution approaches a symmetric distribution far from parameter bounds and is symmetric in the absence of parameter bounds. It is still very popular due to its simplicity and versatility (Robert et al., 2016). DREAM was developed more recently (Vrugt et al., 2009) as a follow-up of SCEM-UA (Vrugt et al., 2003), with an ensemble method called Differential Evolution (DE) as its main building block (Ter Braak, 2006).

Ensemble methods use multiple chains, where each chain is updated using the (current) position of the other chains in the ensemble. Ter Braak and Vrugt (2008) introduced another ensemble algorithm (DE-SNK), which utilises a move called the snooker update in conjunction with the Differential Evolution move, to improve exploration of parameter space. Another powerful ensemble method is the Affine Invariant Ensemble sampler (AI) developed by Goodman and Weare (2010), which is very popular in other disciplines, but almost absent in hydrology (Table 1).

The choice of which MCMC method to use is not a straightforward one to make with the sheer amount of methods available and their respective advantages and disadvantages. Some work has been done on comparing MCMC methods within hydrology. Brunetti et al. (2023) compared DE, DE-SNK, AI and AI-SNK (AI extended with snooker updater) in several toy problems, a synthetic and one actual case study. They found that ensemble methods are suitable for vadose zone modelling, and recommend (as a guideline) Affine Invariant based samplers below 10 dimensions (i.e. fewer than 10 parameters to be calibrated), Differential Evolution based samplers between 10 and 20 dimensions and discourage both above 20 dimensions. This limited performance of most Ensemble samplers in high dimensions is also recognised by the authors of the Affine Invariant sampler python package emcee (Hogg and Foreman-Mackey, 2018). Conversely, an Affine Invariant MCMC Ensemble sampler with a particle filter showed good performance in a highly dimensional case study where it was used to infer the real rainfall from both rain and runoff observations (Bacci et al., 2023). Considering Differential Evolution based algorithms: Vrugt et al. (2009) argues that DREAM can efficiently handle problems involving high dimensionality and Multiple-Try DREAM(zs) (Laloy and Vrugt, 2012) has been specifically created for problems involving high dimensionality, further challenging the poor performance of Ensemble samplers in high dimensions.

A powerful algorithm called Hamiltonian Monte Carlo (Duane et al., 1987), which is widely considered the most effective approach when dealing with high dimensionality (Hogg and Foreman-Mackey, 2018), is almost absent from hydrological literature (Table 1). Recently, Bacci et al. (2023) compared the performance of Hamiltonian Monte Carlo to the Affine Invariant sampler with a particle filter in the highly dimensional case study described above. Ulzega and Albert (2023)

argue for the use of Hamiltonian Monte Carlo, while presenting the same case study.

The No-U-Turn Sampler created by Hoffman et al. (2014) is an extension of Hamiltonian Monte Carlo, providing auto-tuning of the two parameters and preventing the sampler to double back and retrace its steps (hence the name). However, it is rarely used in hydrological literature (Table 1), although it is much more user friendly than Hamiltonian Monte Carlo. In contrast, it is very popular in other fields of science. Additionally, the No-U-Turn Sampler is the main algorithm in the state-of-the-art probabilistic programming framework Stan (Carpenter et al., 2017). It was used in hydrology by Krapu and Borsuk (2022), who argue for the use of Hamiltonian Monte Carlo to perform inference for time-varying parameters (which results in a high dimensional problem) in hydrological models, supported by a real world case study, analysing 20 years of streamflow records from the Model Parameter Estimation Experiment (Duan et al., 2006).

A rigorous benchmark comparing MCMC algorithms remains absent in hydrology, yet this could aid more robust uncertainty estimation. The benchmarks that do exist either compare a limited number of relevant algorithms, such as Brunetti et al. (2023) comparing only Ensemble samplers, yet excluding DREAM. And/or they compare algorithms for only a specific case study (which is a specific dimensional problem), leaving the relative performance of the different algorithms as a function of dimensionality undetermined, e.g. Bacci et al. (2023).

Although it would be interesting to address this scientific gap, by comparing the most popular and powerful MCMC algorithms in hydrology with promising MCMC algorithms from other fields of science in a benchmark of varying dimensionality, this proved unfeasible for a MSc thesis. Therefore, the scope was scaled down to comparing an algorithm popular outside hydrology (Goodman and Weare's Affine Invariant sampler), to two algorithms popular in hydrology (Ter Braak's DE and DE-SNK, see Table 1). These three algorithms have the advantage, regarding feasibility, that they can all be implemented with the Python package emcee (Foreman-Mackey et al., 2013). This research is similar to Brunetti et al. (2023), but applied to groundwater modelling instead of inverse vadose zone modelling.

The performance of these samplers is evaluated with a calibration exercise, in which each sampler is given an identical calculation budget. The parameters of four steady-state synthetic groundwater flow models (MODFLOW6) are calibrated with a dimensionality ranging from 1 to 5. The importance of prior knowledge for model calibration is investigated by comparing the performance of the samplers using two different priors. Additionally, the number of observations per layer in the groundwater flow models is varied, to gain insight in the optimal number of observations.

In Chapter 2 a single step of the Metropolis algorithm is described in detail (selected for its simplicity), adapted from Johnson et al. (2022), to provide the (unfamiliar) reader with some insight in the machinery behind MCMC. In Chapter 3 the Methodology for comparing the different algorithms is described, followed by the results in Chapter 4. The results are discussed in Chapter 5, providing recommendations on which sampler to use at which dimensionality, together with attained insights on how to effectively implement MCMC for hydrogeological model calibration. Finally, Chapter 6 wraps everything up with the conclusions.

2 Theoretical Background

To understand MCMC it is essential to understand Bayes' rule (Equation 1). Bayes' rule yields the posterior density (probability density distribution of the parameters, θ, given the data), effectively inverting the likelihood, p(data|θ), through multiplication with the prior distribution, p(θ), and division by the marginal likelihood, p(data). The marginal likelihood is obtained by integrating the joint density, p(data, θ), across the range of θ. With each additional parameter, another dimension is added to the integral, quickly becoming unfeasible to solve analytically. This is where MCMC comes in. Since the posterior density is proportional to the likelihood multiplied by the prior (Equation 2), the shape of the posterior is known, just not its height. Conveniently only the shape of the posterior is required to sample from it.

$$p(\theta|data) = \frac{p(data|\theta)p(\theta)}{p(data)} \tag{1}$$

$$p(\theta|data) \propto p(data|\theta)p(\theta) \tag{2}$$

Consider a single parameter model with a Gaussian prior and likelihood. The selected prior for parameter θ has a mean of zero and standard deviation of one. The prior distribution is given by Equation 3 and its corresponding probability density function (pdf) is given by Equation 4. In this example the mean defined by parameter θ is unknown. However, the data is known to vary normally around this unknown mean with a standard deviation of 0.75. Additionally, a single observation of 6.25 has been made. With this information the likelihood and its pdf can be defined by Equation 5 and Equation 6 respectively.

$$\theta \sim N(0, 1^2) \tag{3}$$

$$p(\theta) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{\theta^2}{2}\right] \quad (4)$$

$$data|\theta \sim N(\theta, 0.75^2) \quad (5)$$

$$p(data|\theta) = \frac{1}{\sqrt{2\pi \cdot 0.75^2}} \exp\left[-\frac{(6.25-\theta)^2}{2 \cdot 0.75^2}\right] \quad (6)$$

Suppose, the current position of the chain is $\theta_t = 3$. In order to determine the next step, a proposal distribution needs to be selected. The Metropolis algorithm requires a symmetric proposal distribution. For the sake of simplicity a uniform proposal distribution is selected, but e.g. a Gaussian proposal distribution is also possible. The selected uniform distribution has a half width of $w = 1$ and is centred around the current position of the chain, $\theta_t = 3$, resulting in: Unif$(2, 4)$.

If all proposals were accepted, this would not result in convergence to regions of posterior density, but in completely random walk behaviour (often called the drunkard's walk). If instead only proposals with higher relative posterior density than the current position were accepted, this would increase the risk of the chain getting stuck in a local optimum, due to the limited reach of the proposal distribution. Metropolis et al. (1953) found the ideal balance for accepting/rejecting proposals to be determined by Equation 7:

$$\alpha = \min\{1, \frac{p(data|\theta_{t+1})p(\theta_{t+1})}{p(data|\theta_t)p(\theta_t)}\} \quad (7)$$

Where α is the acceptance probability, θ_t is the current position and θ_{t+1} is the proposal for the next position. Note that both the numerator and denominator of Equation 7 are proportional to the posterior density at that location (Equation 2). The equation dictates that proposals with a higher posterior density relative to the current position are always accepted, while proposals with lower posterior density are only accepted stochastically.

Suppose, Unif(2,4) randomly generates the proposal $\theta_{t+1} = 2.933$. The acceptance probability of this proposal can be calculated by filling θ_t and θ_{t+1} into Equation 7 (requiring substitution of Equation 4 and Equation 6 into Equation 7). The acceptance probability for this proposal is $\alpha \approx (5.4 \cdot 10^{-3} \cdot 3.0 \cdot 10^{-5})/(4.4 \cdot 10^{-3} \cdot 4.5 \cdot 10^{-5}) \approx 0.83$. In order to determine whether the proposal is accepted a random draw (r) is made from Unif(0,1). Say, the value drawn for r equals 0.72, then the proposal will be accepted (as $\alpha > r$) and the chain moves towards $\theta = 2.933$.

Now that the chain has moved to a new location, the process is repeated. For the next step, a new proposal will be generated from a proposal distribution centred around $\theta = 2.933$ and accepted stochastically by applying Equation 7.

3 Methodology

In Section 3.1 the MCMC samplers used in this study are explained in detail. The diagnostics used to evaluate the performance of these samplers are described in Section 3.2. Finally, in Section 3.3 the synthetic case studies designed for this study are presented, including the prior and likelihood choices. Everything is implemented in the Python programming language.

3.1 MCMC samplers used

Although the simplicity of the Metropolis algorithm is appealing, it is relatively inefficient at exploring the posterior. The algorithms compared in this study attempt to increase this efficiency by utilising different methods to generate proposals for a Markov chain. Opposed to the Metropolis algorithm, these Ensemble Samplers have dependent chains; proposals are generated based on the location of other chains in the ensemble.

The first MCMC algorithm used in this study is Differential Evolution (DE). Ter Braak (2006) combined the genetic algorithm Differential Evolution (Storn and Price, 1997) with MCMC (Figure 1). The Differential Evolution algorithm is known in hydrology for it being the main building block of DREAM, the most widely used MCMC method in hydrology.

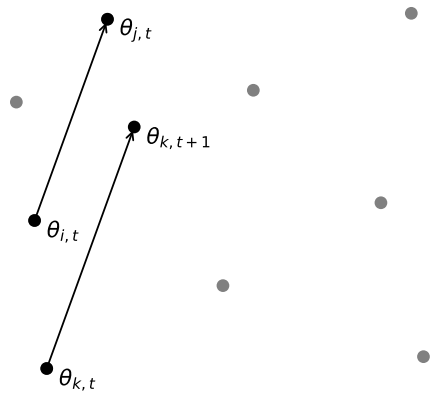


Fig. 1: Differential Evolution move for parameter θ of chain k. The grey dots represent the chains not participating. A proposal is generated by randomly selecting two other chains (i and j) and superimposing the difference vector of these two chains to the current chain. The difference vector is multiplied by a user-defined scalar (here 1.2), resulting in the proposal $\theta_{k,t+1}$.

The second MCMC algorithm used in this study is Differential Evolution extended with a snooker updater (DE-SNK). Ter Braak and Vrugt (2008) introduced a move called a snooker update to MCMC (Figure 2).

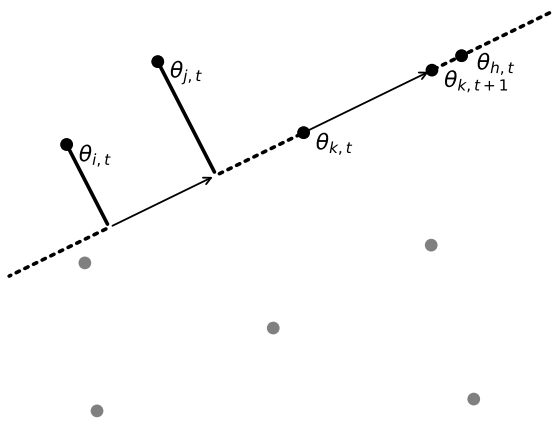


Fig. 2: Snooker update move for chain k. A proposal is generated by randomly selecting another chain (h) for determining the direction and two chains (i and j) for determining the distance. The distance is calculated by projecting i and j orthogonally on the line connecting k to h and subsequently multiplying with a user-defined scalar (here 1.2), resulting in the proposal $\theta_{k,t+1}$.

Ter Braak and Vrugt (2008) found that combining Differential Evolution with snooker updates in a 90 to 10 percent ratio leads to the best performance, which is also the setup in this study. They also introduced an algorithm similar to DE-SNK, which requires the use of very few chains (typically 3), by exploiting information from their past by generating jumps from differences of pairs of past states. Although powerful, sampling from past states was not included as it is not supported by the software used in this study.

The third and final MCMC algorithm used in this study is the Ensemble Sampler by Goodman and Weare (AI), that utilises the Stretch move (Figure 3). Goodman and Weare (2010) introduced several affine-invariant MCMC algorithms; hence the name. Here, affine-invariant refers to the algorithm performing well regardless of how skewed the posterior distribution is, e.g. due to strongly correlated parameters. The introduced algorithms utilise different moves, of which the Stretch Move is considered the most powerful (Goodman and Weare, 2010).

All three samplers are implemented with emcee version 3.1.6 (Foreman-Mackey et al., 2013). Different moves were selected using the *moves* keyword for the (class) *EnsembleSampler*. Since the three samplers only vary in the moves used, selecting different moves results in selecting a different sampler. Optionally, a weighted mixture of moves can be used, as required for DE-SNK. During sampling, at each step, a move is randomly selected from the mixture. For all moves, the default scaling parameters of the emcee package are used, which are consistent with the developers recommendations of said algorithms (Ter Braak, 2006; Ter Braak and Vrugt, 2008; Goodman and Weare, 2010).

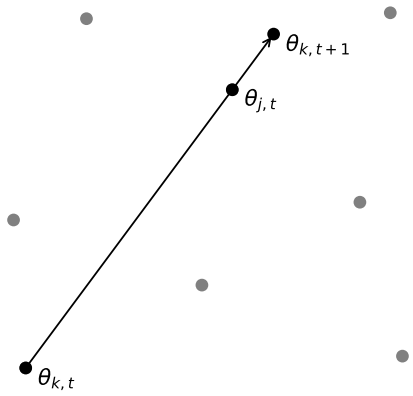


Fig. 3: A stretch move for parameter θ of chain k. The grey dots represent the chains not participating in this move. A proposal is generated by randomly selecting another chain (j) and moving in this direction, with the magnitude determined by a user determined scalar (here 1.2), resulting in the proposal $\theta_{k,t+1}$.

3.2 Performance diagnostics

The performance of the different samplers is compared using several diagnostics explained in this section. Chain convergence is evaluated with the Gelman-Rubin diagnostic ($\hat{R}$). Sampling efficiency is assessed with the acceptance rate and the Effective Sample Size (ESS). The posteriors are evaluated on their median, kernel density estimate (KDE) plots and pairs plots.

3.2.1 Gelman-Rubin diagnostic

Gelman and Rubin (1992) designed a widely used metric to determine whether one's chains have converged, typically referred to as the Gelman-Rubin diagnostic ($\hat{R}$). This is a ratio between the total variance and the within chain variance (where the total variance is due to both the within and between chain variance). As $\hat{R} \to 1$, the chains approach a stationary distribution (Lambert, 2018). In practice, when $\hat{R}$ becomes smaller than an arbitrary cutoff value, one's chains are considered to have converged, for which Gelman and Rubin (1992) suggested using $\hat{R} \leq 1.1$.

Gelman et al. (2013) made a small modification to $\hat{R}$, called split-$\hat{R}$, which also compares the first half of each chain to the second half, with the objective to detect lack of convergence within each chain.

While very popular, several serious issues have been identified with $\hat{R}$ and split-$\hat{R}$, such as failing to correctly diagnose convergence when the chain has a heavy tail or

when the variance varies across the chains. To address these problems, Vehtari et al. (2021) proposed an alternative rank-based version of $\hat{R}$. In addition Vehtari et al. (2021) recommends a much more conservative convergence criterion of $\hat{R} \leq 1.01$.

The rank-based $\hat{R}$ from (Vehtari et al., 2021) is implemented with the function *rhat* from the *ArviZ* package, version 0.19.0 (Kumar et al., 2019) and will from now on simply be referred to as $\hat{R}$.

3.2.2 Effective Sample Size

MCMC is a dependent sampling method, resulting in correlation between serial steps of a Markov chain. As a consequence, the effective number of independent samples, called the Effective Sample Size (ESS), is smaller than the total number of steps (Equation 8).

$$\widehat{ESS} = \frac{N}{\hat{\tau}_f} \tag{8}$$

Where N represents the total number of steps of the Markov chain and $\hat{\tau}_f$ represents an estimate of the integrated autocorrelation time, which quantifies how many intermediate steps are required for steps to be considered independent.

For a longer integrated autocorrelation time, more samples must be generated to produce a representative sample of the target density. For a Markov chain τ_f can theoretically be estimated with Equation 9:

$$\hat{\tau}_f(N) = 1 + 2\sum_{\tau=1}^{N} \hat{\rho}_f(\tau) \tag{9}$$

where $\hat{\tau}_f(N)$ represents an estimate of the integrated autocorrelation time, N represents the number of steps of the Markov chain and $\hat{\rho}_f(\tau)$ represents an estimate of the normalised autocorrelation function. Here, the autocorrelation function is normalised by dividing it by the autocovariance at lag 0.

At larger lags, $\hat{\rho}_f(\tau)$ starts to contain more noise than signal, summing up to N will therefore lead to a noisy estimate of $\hat{\tau}_f$. Instead, Sokal (1997) recommends estimating τ_f, for some $M \ll N$, with Equation 10:

$$\hat{\tau}_f(M) = 1 + 2\sum_{\tau=1}^{M} \hat{\rho}_f(\tau) \tag{10}$$

By introducing M, the function is evaluated for a smaller number of lags, decreasing the variance at the cost of a small increase in bias. Sokal (1997) recommends choosing the smallest value of M, where $M \geq C\,\hat{\tau}_f(M)$ and $C \sim 5$. For an extensive discussion on computing the integrated autocorrelation time the reader is referred to Sokal (1997), or to Foreman-Mackey (2022) for a summary of the most important points. Here, $\hat{\tau}_f(M)$ was computed with the function *autocorr.integrated_time* from the *emcee* package, version 3.1.6 (Foreman-Mackey et al., 2013).

3.2.3 Other diagnostics

Generally, groundwater flow models have a constant value for each parameter. On the other hand, MCMC produces an estimated posterior distribution for each parameter. This introduces the challenge of choosing the most suitable value from this distribution to use in the groundwater flow model. Possible options include the distribution mean, median, or mode. The mean has the drawback that it is strongly influenced by extreme values. This is problematic because, the the posterior distributions of the parameters in this study span multiple orders of magnitude. Selecting the mode of the distribution may be an intuitive value to use in the groundwater flow model. However, problems arise when the distribution is multimodal (the tallest mode, may e.g. have little width, making it challenging to select a mode). Since the median does not have the drawbacks of the mean and mode, it has been selected as the point estimator to use for parameters in MODFLOW 6.

The fraction of proposed steps that are accepted is called the acceptance rate, giving an indication of sampling efficiency. If the acceptance rate approaches 1, most proposals are accepted, resulting in inefficient random-walk behaviour. An acceptance rate close to 0 on the other hand, results in inefficiency due to the chain being mostly stuck in the same place.

Gelman et al. (1997) found that optimal acceptance rates for the Metropolis algorithm scale with dimensionality (number of calibrated parameters), starting at about 0.44 for 1 dimension and decreasing to about 0.23 for highly dimensional problems. The optimal acceptance rate decreases with dimensionality because, for higher dimensional problems, the 'volume' containing posterior density becomes increasingly small relative to the remainder of parameter space (MacKay, 2003). Although the optimal acceptance rates found by Gelman et al. (1997) are for the Metropolis algorithm, the results are widely used as a guideline for other samplers, including by Ter Braak (2006) for DE. More recently Schmon and Gagnon (2022) found optimal Metropolis acceptance rates for specific dimensional problems. By adjusting the tuning parameters of a sampler (e.g. the width of the proposal distribution for Metropolis), different acceptance rates can be achieved. However, for Ensemble Samplers, default tuning parameters are typically used (also in this study, see Section 3.1). Therefore, considering sampling efficiency, it is useful to examine whether default tuning parameters for the samplers used in this study result in (close to) optimal acceptance rates.

Kernel density estimate (KDE) plots represent the posterior samples using a continuous probability density curve, providing a clear visualization of the posterior distributions of each parameter. KDE plots were used for identifying multimodality and for comparing the posteriors produced by different samplers. KDE plots were implemented using the function *kdeplot* from the *Seaborn* package, version 0.13.2 (Waskom, 2021).

A pairs plot is a matrix of graphs that visualizes the relationship between parameters by pairwise comparison. It combines both histogram and scatter plots, providing an overview of estimated posterior distributions and parameter covariance. Pairs plots were implemented (with a KDE style) using the function *pairplot* from the *Seaborn* package.

3.3 Synthetic scenarios

The MCMC samplers (DE, DE-SNK and AI) are compared in a calibration exercise. Four simple steady-state synthetic groundwater flow models (designed using MODFLOW 6) were calibrated, which are described in Section 3.3.1. The performance of the samplers is evaluated in different scenarios, in which not only different models are used, but also different priors and number of observations, as described in Section 3.3.2.

3.3.1 Groundwater flow models

The four synthetic groundwater flow models designed for this study are shown in Figure 4. These models were developed using MODFLOW 6, version 6.5.0 (Langevin et al., 2017). To create, run and post-process these models, FloPy version 3.8.1 was used (Hughes et al., 2023). The parameters *inner_maximum* and *outer_maximum* from FloPy's Iterative Model Solution package were both increased from their default values (25 and 50, respectively) to 1000, allowing more iterations for the solver. This solved MODFLOW simulation errors encountered during testing runs (Appendix E.3).

The hydraulic conductivities presented in Figure 4 represent the 'true' values for each model. Running each model with these 'true' hydraulic conductivities results in the 'true' hydraulic heads for every model cell. With the well located in the phreatic layer in every model, one would expect flow paths to be towards the well for all layers. This is indeed the case as can be seen by the deeper layers having increasingly large hydraulic heads (Figure 5).

To simulate the effects of measurement errors of hydraulic head measurements performed in the field, noise was added to the 'true' hydraulic heads of every model cell. Rau et al. (2019) compared eight different electric dip meters and found that the random measurement error varied by several centimeters depending on the operator and the water depth. Furthermore, it was

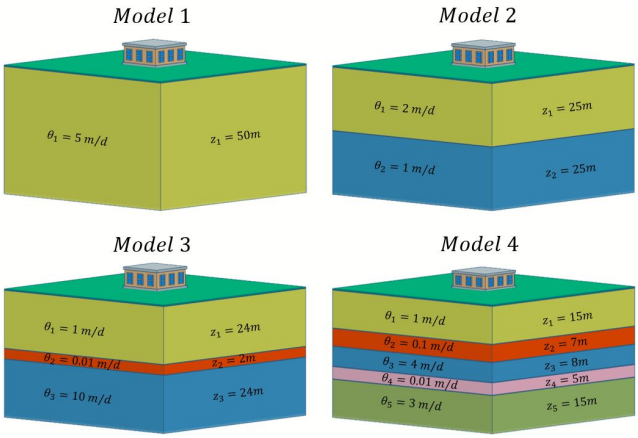


Fig. 4: Schematic of the four steady-state synthetic groundwater flow models used in this study. Layer thickness (z_n) and the respective isotropic hydraulic conductivities (θ_n) are shown inside each layer (n). The models have a length and width of 1000 meters, consisting of 25 rows and columns with a length of 40 meters. The total depth of each model equals 50 meters. At the centre of each model is an abstraction well (indicated by the building) with an extraction rate of 500 m^3/day. The well screens extend over the full depth of the phreatic aquifer of the respective model, resulting in e.g. a 50 meter long well screen for Model 1, versus a 15 meter long well screen for Model 4. At the model edges, there are constant head boundaries of 10 meters.

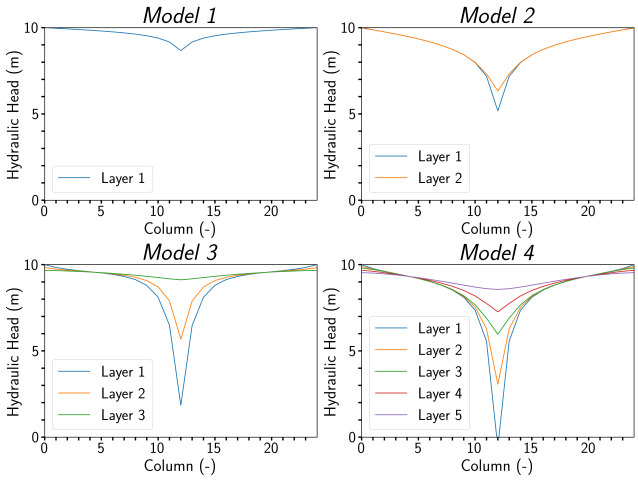


Fig. 5: Side view of the true hydraulic heads (m) in all models. The side view shows row 12 out of 25 rows total, indexing from 0. The x axis shows the columns of the different models (also 25 total). Note that layer 1 always refers to the uppermost layer, with deeper layers having sequentially greater numbers.

shown by Rau et al. (2019) that the magnitude of systematic errors tends to be much larger than that

Table 2: Priors for the hydraulic conductivities in the designed groundwater flow models. The presented prior distributions (truncated normal) are transformed for use by the MCMC samplers[1]. For each prior, a formula is presented to transform the respective parameter as used in MCMC (θ_{mc}), to a hydraulic conductivity in m/d (θ_{mod}). For the narrow prior, $\mathcal{O}$ is introduced, which represents log10() of the 'true' hydraulic conductivity (m/d), subsequently rounded down to an integer (in python: math.floor(math.log10(x))).

Name	Sediment	Prior distribution	Transformation (m/d)	bounds (m/d)
Wide	clean sand	$\mathcal{TN}(0, 0.5^2, -1, 1)$	$\theta_{\text{mod}} = 10^{2\theta_{\text{mc}}+1}$	$[10^{-1}, 10^{3}]$
Wide	silty sand	$\mathcal{TN}(0, 0.5^2, -1, 1)$	$\theta_{\text{mod}} = 10^{2\theta_{\text{mc}}}$	$[10^{-2}, 10^{2}]$
Wide	silt, loess	$\mathcal{TN}(0, 0.5^2, -1, 1)$	$\theta_{\text{mod}} = 10^{2\theta_{\text{mc}}-1}$	$[10^{-3}, 10^{1}]$
Narrow	-	$\mathcal{TN}(0, 0.125^2, -1, 1)$	$\theta_{\text{mod}} = 10^{2\theta_{\text{mc}}+\mathcal{O}}$	$[10^{-2+\mathcal{O}}, 10^{2+\mathcal{O}}]$

of random errors. Systematic errors can have a constant bias (e.g. poorly calibrated equipment) or a non-constant bias (e.g. salinity can result in a linearly increasing bias with depth due to density differences compared to fresh water). For simplicity, all errors were lumped into a Gaussian distribution with an inflated variance based on these findings. Measurement noise was added $\epsilon \sim N(0, 0.1)$, with the mean and standard deviation in meters.

After adding noise to all hydraulic heads, observation locations were selected stochastically. This was achieved by randomly generating numbers between 2 and 24 using the function *randint* from Python's built-in *random* module, with each number specifying the row or column of a cell in the model. Cells on the boundary of the model, with corresponding row and column integers of 1 and 25, are excluded from being selected as measurement locations, due to the boundary conditions at the edges of the models. If by chance the same x,y coordinates are selected more than once, resampling occurs. If the model has multiple layers, selected observations are always on top of each other, similar to how one measurement well contains multiple piezometers in different layers.

To provide insight into the importance of the amount of available data, three scenarios per model were created. The scenarios differ in the number of observations per layer: 1, 3 or 5. For a more complex model this results in a greater number of total observations than a simpler model, but in the same amount of observations per layer (i.e. per parameter).

3.3.2 MCMC setup

Six different prior-likelihood combinations (scenarios) were created for each of the four synthetic groundwater flow models shown in Figure 4. These scenarios differ in: prior knowledge (on hydraulic conductivities) and available data (number of hydraulic head measurements for computing the likelihood).

The priors are similar to what may be used in a real case study, by using representative values of the respective sediments/geological materials. For each layer of each model two different priors were created, from now on referred to as the narrow and wide prior.

The wide priors are based on lithology (supposedly) found in the respective layers during e.g. piezometer construction. The prior bounds for hydraulic conductivities are inspired by the ranges for clean sand, silty sand, and silt presented in Woessner and Poeter (2020). The 'true' hydraulic conductivity value of each layer was designed to fall within the prior bounds of the lithology found within each layer (see Table 2 for the priors and Appendix C2 for the lithology of each layer).

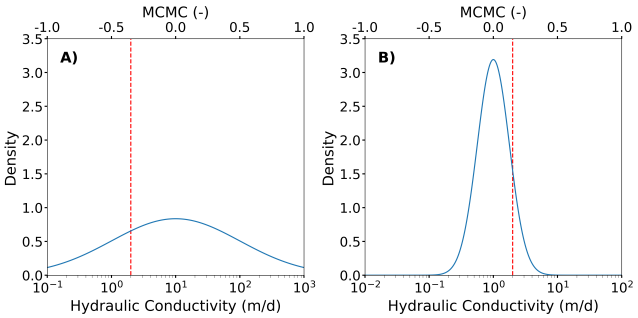


Fig. 6: Prior distributions for the hydraulic conductivity of layer 1 in Model 2 (contains clean sand), with **A)** showing the wide prior and **B)** the narrow prior. The top x-axes indicate the parameter values used by the MCMC algorithm, while the bottom x-axes show the corresponding transformed hydraulic conductivity values (m/d) as used by MODFLOW 6. The priors are truncated at the plot boundaries. The dotted red line indicates the true hydraulic conductivity.

[1]The applied parameter transformations differ from statistical convention. Conventionally, the transformation for the likelihood function would simply take the exponent of the parameter: $\theta_{mod} = 10^{\theta_{mc}}$. The prior for e.g. clean sand would then be $\mathcal{TN}(1, 1, -1, 3)$ and the prior for silty sand would be $\mathcal{TN}(0, 1, -2, 2)$. However, the result of both methods on inference is the same.

For the narrow prior, it is assumed that a much more precise estimate of the hydraulic conductivity of each layer is available (e.g. from pumping tests). The narrow priors have a standard deviation four times smaller than that of the wide priors. Additionally, the centre of the narrow prior distribution depends on the 'true' hydraulic conductivity (Table 2). Note that the centre of the narrow prior is within an order of magnitude distance from the 'true' hydraulic conductivity (Table 2). An example of the narrow and wide prior distributions is shown in Figure 6.

With hydraulic conductivities spanning many orders of magnitude, numerical stability of the MCMC samplers was a concern. Parameter transformation was applied to address this issue. The parameters were transformed at each step of the Markov chain, to a hydraulic conductivity in meters per day (Table 2). Subsequently, the respective model was run in MODFLOW 6 with the transformed parameters. For each selected observation location, the difference was calculated between the hydraulic heads produced by this model run and the noisy observations (δh). For each observation, δh was inserted into the log probability density function (logpdf) of the error. The log-likelihood was computed as the sum of the logpdf of every observation. The selected likelihood distribution is $\mathcal{N}(0, 0.1)$, which is identical to the distribution that was used to generate the measurement noise. Priors and likelihood were implemented with the functions *truncnorm*, *norm* and *logpdf* from the *SciPy.stats* package, version 1.11.4 (Virtanen et al., 2020).

Chain initialisation was performed by taking a random sample from the respective prior distribution with the *rvs* function from the *SciPy.stats* package. This was programmed with a constant seed, to ensure that identical datasets were used for chain initialisation of different samplers and scenarios.

For each of the six scenarios (prior-likelihood combinations), five ensembles were run per sampler (AI, DE and DE-SNK), each consisting of 10 chains with a length of 2000 steps. This procedure was carried out for all (four) designed synthetic groundwater flow models. Trace plots of testing runs showed good mixing of the Markov chains within 1000 steps. Therefore, burn-in was set at 1000 steps, leaving 1000 steps per chain for estimation of the posterior distribution.

4 Results

This chapter presents the results of calibrating Model 1-4 with the MCMC samplers: AI, DE and DE-SNK, for the scenarios described above. The efficiency of the samplers is evaluated using acceptance rates and the estimated Effective Sample Size ($\widehat{ESS}$). Subsequently, chain convergence is assessed with the Gelman-Rubin diagnostic ($\hat{R}$). Next, samplers are compared based on how closely their posterior medians approximate the true hydraulic conductivities. Finally, the shape of the posterior distributions and the relationship between different parameters is investigated with Kernel Density Estimate plots (KDE) and pairs plots, respectively.

The acceptance rate of proposed steps varies strongly between the different samplers, with the Affine-Invariant sampler (AI) reporting the highest acceptance rates (Table 3). The observed acceptance rate for all samplers decreases when the number of calibrated parameters increases, considering that dimensionalities for Model 1 up to Model 4 are 1,2,3 and 5, respectively. The optimal acceptance rate according to theory is 0.44 for 1 dimension, decreasing to 0.28 for 5 dimensions and asymptotically approaching 0.23 for even higher dimensions (Schmon and Gagnon, 2022; Ter Braak, 2006). For DE-based strategies, most acceptance rates closely align with these optimal values. Except for too low values when using the wide prior for Model 3 and 4. In contrast, AI reports too high acceptance rates for most scenarios. Regarding prior choice, using the narrow prior resulted on average in more optimal acceptance rates for DE and DE-SNK, but not for AI. The variability in acceptance rates between chains of a specific sampler for a specific scenario is relatively small, indicated by the compactness of the boxplots, suggesting consistency between different ensembles (Table 3).

The estimated Effective Sample Size ($\widehat{ESS}$) decreases with increasing model number (Table 4). This is what was expected based on the acceptance rate also decreasing with increasing Model number (Table 3). Although AI has the highest acceptance rate, its $\widehat{ESS}$ is actually the lowest for almost all scenarios. Thus, the autocorrelation of Markov chains must be relatively high for AI, implying that the AI 'Stretch' move produces relatively poor proposals. There is no clear

Table 3: Acceptance rates for each scenario are presented with boxplots. Each boxplot was constructed using acceptance rates of the last 1000 steps of 50 chains (5 ensembles of 10 chains). Optimal Metropolis acceptance rates, based on dimensionality, are indicated with dashed red lines: 0.44 for Model 1, 0.35 for Model 2, 0.31 for Model 3 and 0.28 for Model 4 (Schmon and Gagnon, 2022). Although invisible, all boxplots in the same column share the same x-axis and thus can be compared. Additionally, average row acceptance rates are provided in the rightmost column and average column acceptance rates are provided in the bottom row.

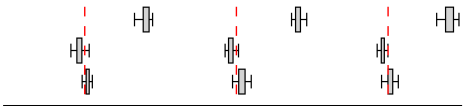
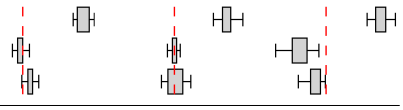
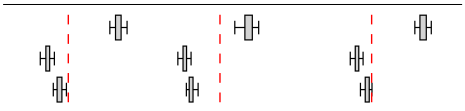
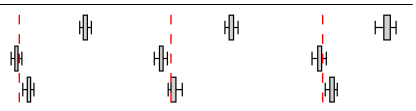
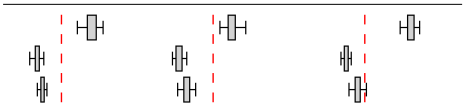
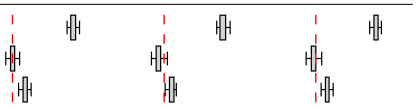
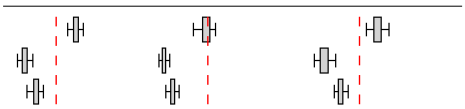
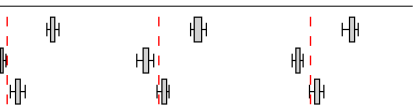
Model	Sampler	Wide prior			Narrow prior			Average
		1 obs	3 obs	5 obs	1 obs	3 obs	5 obs	
1	AI							**0.73**
	DE							**0.40**
	DE-SNK							**0.44**
2	AI							**0.64**
	DE							**0.28**
	DE-SNK							**0.33**
3	AI							**0.56**
	DE							**0.24**
	DE-SNK							**0.30**
4	AI							**0.44**
	DE							**0.16**
	DE-SNK							**0.25**
Average		**0.37**	**0.30**	**0.38**	**0.46**	**0.43**	**0.43**	-

Table 4: The Estimated Effective Sample Size ($\widehat{ESS}$) for each scenario. The presented values were obtained by first calculating the mean $\widehat{ESS}$ for the last 1000 steps of each chain. Then, the sum was taken for all 10 chains of all 5 ensembles (50 chains total and 50,000 samples). For interpretability, row and column averages are also provided.

Model	Sampler	Wide prior			Narrow prior			Average
		1 obs	3 obs	5 obs	1 obs	3 obs	5 obs	
1	AI	2462	2291	2422	3341	2878	3036	**2738**
	DE	9377	5755	9241	7743	10708	5165	**7998**
	DE-SNK	8534	4637	8902	7460	9193	5212	**7323**
2	AI	2867	1686	3009	4252	3506	4210	**1567**
	DE	3498	1801	4147	6241	4459	6508	**4442**
	DE-SNK	3840	1906	4045	6168	4278	5953	**4365**
3	AI	1065	903	1276	1673	1612	1851	**1397**
	DE	2323	1629	2494	4787	4131	4393	**3293**
	DE-SNK	2585	1620	2890	4383	3706	4292	**3246**
4	AI	807	670	805	1202	1094	1075	**942**
	DE	1333	851	1026	2719	2045	2063	**1673**
	DE-SNK	1514	921	1462	2748	2369	2326	**1890**
Average		**3228**	**1985**	**3345**	**4207**	**4015**	**3657**	-

winner for $\widehat{ESS}$, with DE and DE-SNK reporting similar values for most models, except for Model 4, for which DE-SNK consistently outperforms DE. Using more observations for likelihood evaluations appears to have a negative effect on the $\widehat{ESS}$, although not distinctly (column totals Table 4). Using a more informative prior does appear to have a positive effect on the $\widehat{ESS}$ (Table 4), consistent with more optimal acceptance rates for the narrow prior (Table 3).

Table 5: The number of ensembles with all Gelman-Rubin diagnostic values ($\hat{R}$) below the threshold of 1.05. First, $\hat{R}$ was calculated for all parameters using the last 1000 steps of each chain per ensemble. Then, the ensembles with all $\hat{R} < 1.05$ were counted for each scenario. For interpretability, row and column averages are also provided.

Model	Move	Wide prior			Narrow prior			Average
		1 obs	3 obs	5 obs	1 obs	3 obs	5 obs	
1	AI	5	5	4	2	1	1	**3.00**
	DE	5	5	5	5	5	0	**4.17**
	DE-SNK	5	5	5	5	2	0	**3.67**
2	AI	0	0	1	4	2	2	**1.50**
	DE	5	4	5	5	5	5	**4.83**
	DE-SNK	5	3	5	5	5	5	**4.67**
3	AI	0	0	0	2	0	3	**0.83**
	DE	4	1	4	5	5	5	**4.00**
	DE-SNK 3	5	0	4	5	5	5	**4.00**
4	AI	0	0	0	0	0	0	**0.00**
	DE	0	0	0	5	1	4	**1.67**
	DE-SNK	1	0	0	5	5	4	**2.50**
Average		**2.92**	**1.92**	**2.75**	**4.00**	**3.00**	**2.83**	**-**

Too few ensembles passed the recommended $\hat{R}$ threshold of 1.01 by Vehtari et al. (2021) for any meaningful comparison. Therefore, the threshold in Table 5 was relaxed to 1.05, which is still relatively strict compared to the original recommendation of 1.10 by Gelman and Rubin (1992). AI reports the fewest ensembles with all $\hat{R}$ below the threshold of 1.05 for each model, suggesting that AI convergences relatively poorly (Table 5). This aligns with AI reporting the lowest $\widehat{ESS}$ values, as chains tend to have a longer integrated autocorrelation time (τ_f) during burn-in (Hogg and Foreman-Mackey, 2018). Using the narrow prior results in more ensembles with all $\hat{R}$ below the threshold for Model 2,3 and 4, indicating improved convergence. This trend is consistent with $\widehat{ESS}$ being larger for Model 2,3 and 4, when using the narrow prior (Table 4). For models with more parameters (higher model number), $\hat{R}$ indicates worse convergence for all samplers (Table 5).

The median parameter values of all steps of all chains of all (5) ensembles for a scenario (from now on referred to as simply the median), are extremely similar for the different samplers (Table 6), even though AI showed the worst acceptance rate, $\widehat{ESS}$ and $\hat{R}$. The median is a much better estimate of the true parameter values ($\theta_{n,\,true}$) when using the narrow prior, opposed to using the wide prior. This difference is especially striking for parameters with a low $\theta_{n,\,true}$. Increasing the number of observations for evaluating the likelihood clearly improves the estimate of $\theta_{n,\,true}$, which is again most striking for parameters with a low $\theta_{n,\,true}$. Interestingly, using more observations was shown to decrease convergence, yet the estimate of $\theta_{n,\,true}$ improves. This suggests that using more observations leads to a better estimate of the true posterior at the cost of a longer burn-in time.

The influence of the prior on the shape of the posterior appears strong, with the wide prior resulting in a much broader posterior (Figure 7). These Kernel Density Estimate (KDE) plots also reveal a second mode for θ_1 of Model 1, though it contains comparatively little density. Interestingly, the second optimum is (visually) absent for the scenario, where the wide prior is used in combination with three observations for evaluating the likelihood. Figure 7 only shows results for DE-SNK, but plots for AI and DE reveal very similar posteriors (appendix figures D1 and D4), which is consistent with Table 6 showing little variation of the Median from different samplers.

Pairs plots were created for Model 2-4 to identify the posterior distributions and parameter covariances. For conciseness, pairs plots are limited to the scenario with the wide prior and 1 observation per layer. Additionally, only pairs plots for DE-SNK are shown, as different samplers produced similar pairs plots (Appendix D.2).

For Model 2, both parameters show a strongly skewed unimodal distribution (Figure 8), with the posterior of the upper layer, p(θ_1|data), being much narrower than p(θ_2|data). The joint distribution reveals that θ_1 is largely insensitive to θ_2, if $\theta_2 \lessapprox 0$.

Table 6: Median parameter values of all (5) ensembles, with θ_n representing the hydraulic conductivity (m/d) of layer n. The true hydraulic conductivities are given between parentheses in column two (Parameter). Each coloured cell represents a scenario (for which 5 ensembles were run), described by the row and column labels. The colour of each cell is an indication of how much the cell value differs from the true hydraulic conductivity of the respective layer (absolute log difference), with dark green representing no difference (best) and red representing large differences (worst). The absolute log difference was used to determine cell colours, because different parameters are of different orders of magnitude.

Model	Parameter	Move	Wide prior			Narrow prior		
			1 obs	3 obs	5 obs	1 obs	3 obs	5 obs
1	θ_1 (5.0)	AI	13	4.1	19	2.2	2.7	3.1
		DE	12	4.1	20	2.3	2.8	3.0
		DE-SNK	12	4.0	19	2.2	2.7	3.1
2	θ_1 (2.0)	AI	1.9	1.9	2.9	1.3	1.4	1.9
		DE	2.0	2.0	2.9	1.4	1.4	1.9
		DE-SNK	2.0	1.9	2.9	1.3	1.4	1.9
	θ_2 (1.0)	AI	0.92	1.0	0.46	1.3	1.4	1.4
		DE	0.83	0.86	0.54	1.3	1.4	1.4
		DE-SNK	0.79	0.96	0.57	1.3	1.4	1.4
3	θ_1 (1.0)	AI	0.90	0.89	1.3	1.1	0.89	1.2
		DE	0.83	0.84	1.2	1.1	0.90	1.2
		DE-SNK	0.79	0.77	1.3	1.1	0.91	1.2
	θ_2 (0.010)	AI	0.048	0.089	0.0087	0.010	0.010	0.0095
		DE	0.047	0.090	0.0088	0.010	0.010	0.0093
		DE-SNK	0.053	0.092	0.0083	0.010	0.010	0.0095
	θ_3 (10)	AI	6.1	3.1	16	11	8.6	10
		DE	6.9	3.2	15	10	8.7	10
		DE-SNK	6.0	3.4	15	10	8.7	10
4	θ_1 (1.0)	AI	1.9	1.2	0.95	2.1	2.3	1.6
		DE	2.0	1.2	0.90	2.1	2.3	1.6
		DE-SNK	2.0	1.1	0.96	2.1	2.3	1.7
	θ_2 (0.10)	AI	0.020	0.095	0.067	0.11	0.11	0.072
		DE	0.021	0.075	0.067	0.11	0.11	0.070
		DE-SNK	0.019	0.063	0.064	0.11	0.11	0.069
	θ_3 (4.0)	AI	6.6	2.9	3.1	1.3	1.3	1.8
		DE	5.7	3.2	3.8	1.3	1.3	1.8
		DE-SNK	5.9	4.0	3.2	1.3	1.3	1.8
	θ_4 (0.010)	AI	0.068	0.078	0.019	0.010	0.012	0.011
		DE	0.049	0.099	0.017	0.010	0.012	0.010
		DE-SNK	0.059	0.11	0.018	0.010	0.012	0.010
	θ_5 (3.0)	AI	7.8	2.4	5.5	1.2	1.2	1.8
		DE	9.0	2.7	4.8	1.2	1.1	1.7
		DE-SNK	11	3.0	5.3	1.2	1.1	1.7

The parameters of Model 3 show more symmetric behaviour, with p(θ_1|data) again being the most concentrated (Figure 9). Another interesting feature of p(θ_1|data) is that it is approximately uniform (though it does appear to have a mode on the right side). In reality, the distribution may look even more uniform with sharper edges, due to KDE having the (negative) effect of smoothing vertical edges, giving the appearance of tails (Analytica Docs, 2024).

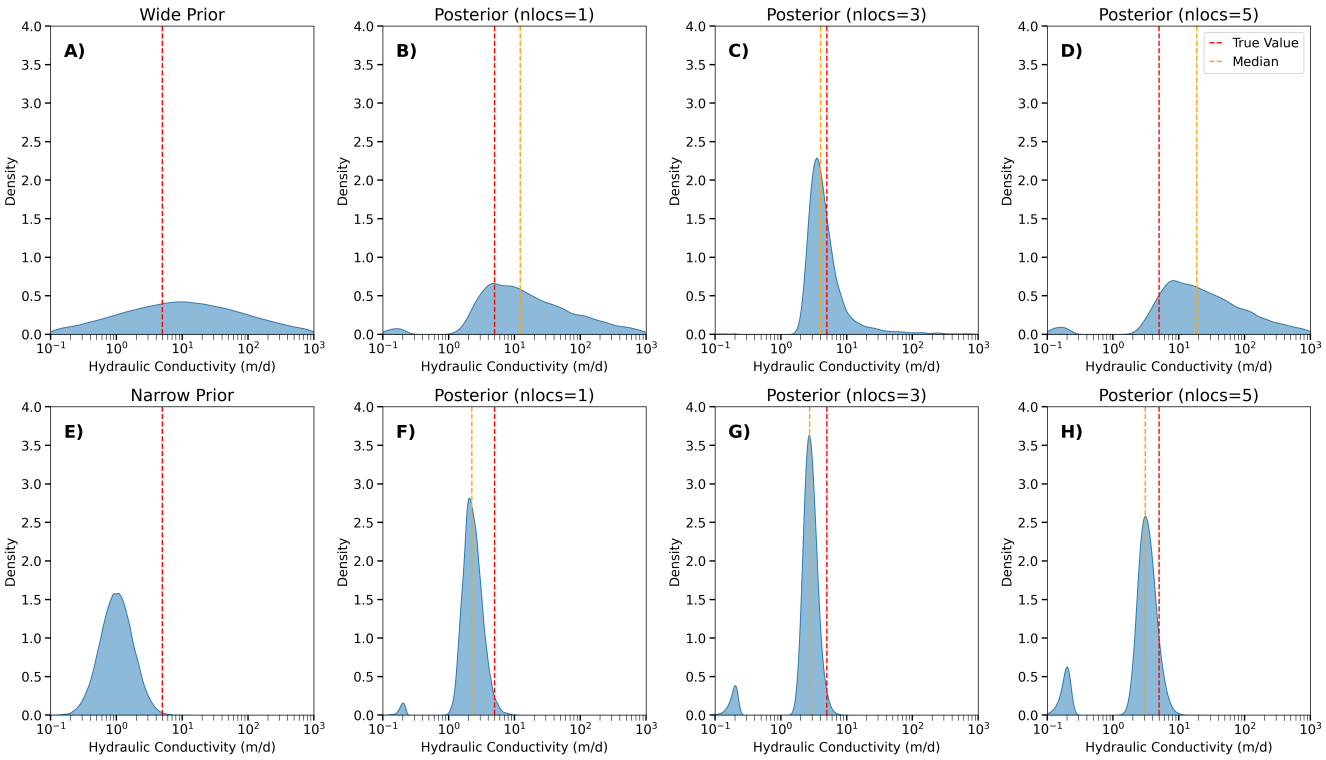


Fig. 7: Kernel Density Estimate plots for the only parameter of Model 1 (θ_1), which was calibrated with DE-SNK. **A)** shows the wide prior. **B), C)** and **D)** show the posteriors after calibrating θ_1 with: the wide prior in combination with 1,3 or 5 observations for evaluating the likelihood, respectively. Similarly, **E)** shows the narrow prior. **F), G)** and **H)** show the posteriors after calibrating θ_1 with: the narrow prior in combination with 1,3 or 5 observations for evaluating the likelihood. In each plot the dashed red line indicates the true value for θ_1. Similarly, the posterior median is indicated by a dashed yellow line.

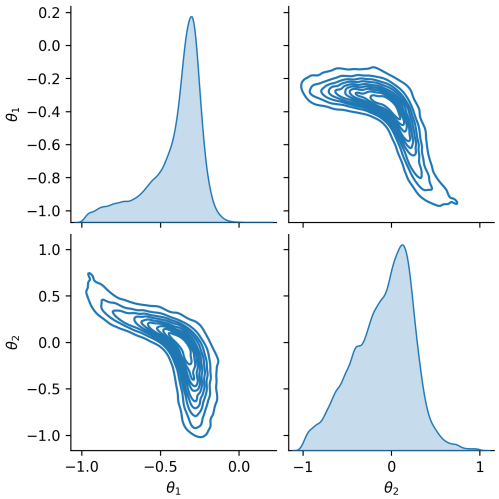


Fig. 8: Pairs plots (with a KDE style) for all calibrated parameters of Model 2. The presented data is from 5 ensembles (last 1000 steps of each chain), run with DE-SNK, a wide prior and 1 observation for evaluating the likelihood. The parameters as presented here are untransformed (used for MCMC).

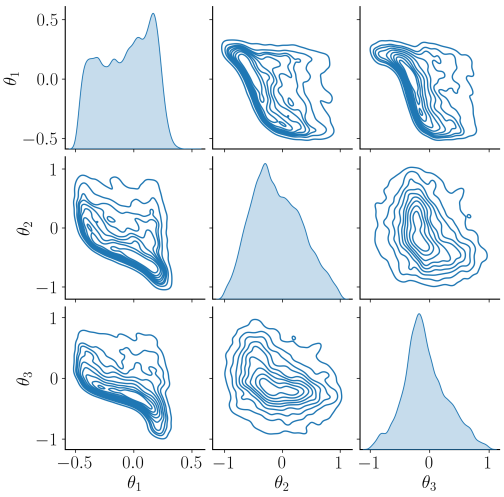


Fig. 9: Pairs plots (with a KDE style) for all calibrated parameters of Model 3. The presented data is from 5 ensembles (last 1000 steps of each chain), run with DE-SNK, a wide prior and 1 observation for evaluating the likelihood. The parameters as presented here are untransformed (used for MCMC).

Joint distributions indicate an inversely proportional relationship between θ_1 and the other two parameters (Figure 9). The approximately spherical joint distribution of θ_2 and θ_3 suggests that these parameters are insensitive to one another (Gelman et al., 2013).

For Model 4, p(θ_1|data) is the most concentrated posterior (Figure 10), similar to Models 2 and 3. All posteriors are unimodal, with p(θ_1|data) and p(θ_2|data) being skewed, while the other posteriors appear symmetric. Joint distributions reveal that the parameters of the deeper layers (θ_3, θ_4, θ_5) are insensitive to one another, with their posteriors increasingly resembling the prior distribution, as the parameter number increases (see the narrow prior in Figure 7).

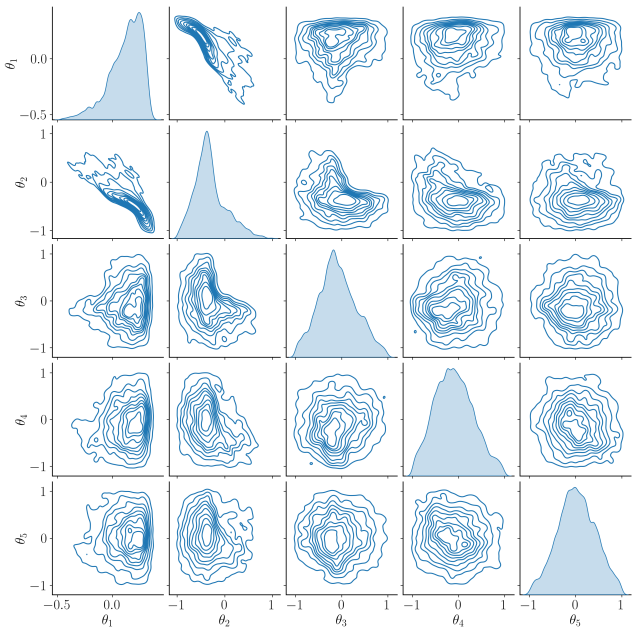


Fig. 10: Pairs plots (with a kernel density style) for all calibrated parameters of Model 4. The presented data is from 5 ensembles (last 1000 steps of each chain), run with DE-SNK, a wide prior and 1 observation for evaluating the likelihood. The parameters as presented here are untransformed (used for MCMC).

5 Discussion

All three samplers managed to converge to similar posterior distributions for the different models and prior-likelihood combinations, but differences in efficiency were observed. The AI sampler was consistently outperformed by DE and DE-SNK, reporting too high acceptance rates, relatively low $\widehat{ESS}$ and higher $\hat{R}$, making AI overall less efficient. The differences reported between DE and DE-SNK are much smaller, with DE appearing slightly better for Model 1 (d=1), but DE-SNK showing superior performance diagnostics for Model 4 (d=5). With increasing dimensionality, this difference is expected to increase further in favour of DE-SNK. Ter Braak and Vrugt (2008) showed that DE-SNK outperforms DE for different student distributions with dimensionality ranging from 10 to 100. Furthermore, DE-SNK is a component of the Multiple-Try DREAM$_{(zs)}$ algorithm, which has been specifically designed to solve high-dimensional posteriors in hydrology (Laloy and Vrugt, 2012). For hydrogeological modelling, these results suggest that DE-SNK is the better choice, as models are typically highly parameterized.

There was some concern that the second mode in the posteriors of Model 1 was caused by convergence issues, such as chains getting stuck, or the burn-in phase being too short. However, inspection of trace plots revealed that different chains move back and forth between these modes. For example, ensemble 5 from DE with the narrow prior (Figure D6) and ensemble 1 from DE with the wide prior (Figure D5).

Though there are exceptions, where different chains appear stuck in a different mode, not mixing, e.g. ensemble 2 from AI with the narrow prior (Figure D3). As a consequence $\hat{R}$ for this ensemble is very high (1.44), but also the ensembles containing chains that do move back and forth between modes tend to have a relatively high $\hat{R}$ (e.g. 1.27, Figure D6), indicating that a chain length of 1000 steps post-burn in was too short to achieve adequate mixing.

Results show that prior choice has a strong impact on the selected performance diagnostics. With the more informative prior, on average, resulting in a more optimal acceptance rate, higher $\widehat{ESS}$, lower $\hat{R}$, and posterior medians closer to the 'true' hydraulic conductivities. In contrast, the convention in hydrogeology is to use uniform prior distributions (Laloy and Vrugt, 2012; Keating et al., 2010). Although uniform priors are often chosen for their perceived lack of influence, this assumption is incorrect. In reality, they are weakly informative, assigning equal probability mass to both implausible and plausible values (Gelman and Yao, 2020). Given these findings, it is recommended that hydrogeologists carefully reconsider their choice of priors.

Also in this study improvements can be made on prior design. The standard deviation of the narrow prior was too small considering the uncertainty of the parameter estimate. The transformed standard deviation of the parameter equals 0.25 orders of magnitude (Table 2). In contrast, true values approaching 1.0 orders of magnitude away from the centre of the prior distribution are possible, though unlikely, given that $>$ 99.99%

of probability density mass is within 4σ. This poses certain risks considering that the posterior predictive distribution can be strongly affected by the prior when there is not much observed data and substantial prior mass is concentrated around infeasible values (Gelman, 2006). A better choice may have been to increase the narrow prior standard deviation from 0.125 to say 0.25 (so still half of the wide prior). This would decrease the amount of prior density concentrated around unfeasible values, when the estimate is relatively far off from the true value.

Using more observations for evaluating the likelihood had a generally negative effect on $\widehat{ESS}$ and $\hat{R}$, but a positive effect on the median as an estimate for the true hydraulic conductivity. These results suggest that using more observations leads to a better estimate of the true posterior at the cost of a longer burn-in time. However, the quality of observations also played an important role. An example of this is the calibration of Model 1 with the wide prior. Here, 5 observations produced the worst median, consistent for all samplers (vs 1 and 3 observations). This implies that there is much to gain from improving measurement quality, considering that the stochastically added noise here is based on random and systematic errors for hydraulic head measurements encountered in the field (Rau et al., 2019). Concerningly, Rau et al. (2019) point out that some measurement techniques have not seen performance improvement in decades and that the pursuit of measurement error reduction in hydrogeology is lacking compared to other fields of science.

Pairs plots revealed correlations between hydraulic conductivities of the shallowest layers, regardless of the model, suggesting equifinality. For Model 4 hydraulic conductivities of the deeper layers show less sensitivity to the other layers, with roughly spherical joint distributions (Gelman et al., 2013). The presence of equifinality can be argued based on the two shallowest layers in Model 2,3, and 4 showing (nonlinear) negative correlations. Physically, this implies that as the hydraulic conductivity of the shallowest layer decreases, the hydraulic conductivity of the layer below it increases, to ensure the total inflow of the well located in the shallowest layer remains the same.

The posteriors of the parameters of the shallower layers are concentrated on a smaller part of parameter space, than the parameters of the deeper layers. Providing further evidence that the models are most sensitive to the shallower layers. However, even the posteriors of the shallower layers span approximately two orders of magnitude (see for Table 2 for conversion), indicating large uncertainty of parameter estimates.

As a consequence, the hydraulic heads predicted by the models may contain substantial uncertainty. In follow-up research, it could be interesting to perform Posterior Predictive Checks (PPC). These are a set of tools that compare predictions from the calibrated model to the observed data, by taking draws from the joint posterior distribution (Gelman et al., 2013).

Testing runs with untransformed parameters showed poor convergence ($\hat{R}$) and very low acceptance fractions for Model 4 (see Appendix E.2 for a comprehensive description of these testing runs and their results). One of the main causes for this inefficiency is the wide range of possible hydraulic conductivity values, with e.g. clean sand reporting hydraulic conductivities ranging four orders of magnitude (Woessner and Poeter, 2020). As a consequence, the Ensemble moves used in this study, which are based on interpolation, become inefficient at proposing steps. Additionally, hydraulic conductivity cannot be negative, but it can be very close to 0 for aquitards such as clay. Without parameter transformation these small nonzero values are very hard to reach, as the sampler will often propose values that extend into the negative range, which are physically meaningless and must be rejected. As a consequence, the values close to zero will be undersampled. The applied log-transformation (Table 2) solved these issues. By working in logarithmic space, the algorithm can propose steps that are more appropriate for a parameter that varies over multiple orders of magnitude. This allows the ensemble moves to remain effective, ensuring that proposals remain well-scaled relative to parameter space (SAS Institute Inc., 2020).

Although recommended, it is unclear whether hydrogeologists apply parameter transformation in their studies, often it is not reported (Vrugt et al., 2009; Laloy and Vrugt, 2012). Furthermore, parameter transformation may be confused with a change of variables: "A transformation samples a parameter, then transforms it, whereas a change of variables transforms a parameter, then samples it". Applying parameter transformation is recommended, because a change of variables requires a Jacobian adjustment (Stan Development Team, 2025).

Our results are contrary to the results of Brunetti et al. (2023), who recommend the use of AI-based strategies below 10 dimensions and DE-based strategies above that threshold. The one caveat being that Brunetti et al. (2023) performed vadose zone modelling experiments (Hydrus-1D), versus modelling groundwater flow.

Diving into the research performed by Brunetti et al. (2023) reveals that they recommend AI-based strategies for lower dimensions based on results from a toy distribution and of a synthetic vadose modelling scenario. The toy distribution is a 10-dimensional Rosenbrock distribution, which is banana shaped, as frequently encountered in vadose zone modelling Brunetti

et al. (2023). In the synthetic scenario the different samplers are used for calibrating 7 soil hydraulic parameters in a Hydrus-1D model, using synthetic observations from a weighable lysimeter.

Regarding the 10-dimensional Rosenbrock distribution, Brunetti et al. (2023) point out that results show poor convergence for DE and DE-SNK, when using 11 chains, based on the Earth's Movers Distance, which quantifies the cost required to transform the sampled posterior distribution to the true posterior distribution (Rosenbrock). However, this is not how these samplers should be used; Ter Braak recommends always using atleast twice as many chains per ensemble, as there are dimensions. Indeed, when using 20 chains or more, these convergence problems disappear (Brunetti et al., 2023).

For the synthetic scenario, results in Brunetti et al. (2023) show very low acceptance rates for DE and DE-SNK, opposed to nearly optimal acceptance rates for AI. As a consequence, the integrated autocorrelation time ($\hat{\tau}_f$) is hundreds of steps long for the shown ensembles, which consist of 50, 100 and 200 chains, respectively.

Brunetti et al. (2023) suggest that these low acceptance rates for DE and DE-SNK could be improved by dynamically adapting tuning parameters (the user-defined scalar parameters in Figure 1 and Figure 2). However, they do not specify whether parameter transformation was applied. If parameter transformation was not applied, this could explain the low acceptance rates.

6 Conclusions

Some MCMC algorithms that are popular in other fields of science are not only rarely used in hydrology, but their performance is also underexplored. Therefore, in this study, an MCMC algorithm most prominently used in astrophysics, AI, is compared to two MCMC algorithms commonly used in hydrology: DE and DE-SNK.

The performance of these algorithms was evaluated with a calibration exercise, in which each algorithm was given an identical calculation budget. The parameters of four steady-state synthetic groundwater flow models (MODFLOW 6), with a dimensionality ranging from 1 to 5, were calibrated. The importance of prior knowledge for model calibration was investigated by comparing the performance of the samplers for two different Gaussian priors. Additionally, the number of observations per layer in the groundwater flow models was varied (1, 3 or 5), to gain insight in the optimal number of observations.

All three samplers converged to similar posterior distributions for the different models, but AI was distinctly the least efficient. Performance of DE and DE-SNK was similar, but DE-SNK is recommended, because it performed better when the dimensionality was 5 and because this difference is expected to increase in favour of DE-SNK based on existing literature.

The prior choice and the number of observations per layer had little effect on the performance of the samplers relative to each other. However, for all samplers, performance diagnostics indicated improvements when using the more informative prior. Furthermore, more observations per layer appear to result in a better estimate of the true posterior at the cost of a longer burn-in.

Other findings include the importance of parameter transformation for the numerical stability of MCMC samplers. Although this is well known in the statistical community, its application is rarely mentioned in hydrological MCMC literature, suggesting that parameter transformation is rarely applied. The absence of parameter transformation could also explain conflicting results with existing vadose zone modelling literature on whether AI or DE based strategies are superior in low dimensional problems. However, parameter transformation not being mentioned, does not guarantee that it was not applied. Therefore, it is recommended to further explore this in follow-up research.

Even though, in this study, AI was consistently outperformed by DE and DE-SNK. It remains unknown how other algorithms popular outside of hydrology, such as the No-U-Turn sampler, perform at hydrological model calibration, representing an important topic for future research.

Code and Data Availability

The Python code used in this study and the synthetic data that was generated with this code are available on GitHub at https://github.com/DouweKamper/thesis.

References

Analytica Docs: Kernel Density Smoothing. Accessed: 2024-11-06 (2024). https://docs.analytica.com/index.php/Kernel_Density_Smoothing

Beven, K., Binley, A.: The future of distributed models: model calibration and uncertainty prediction. Hydrological processes **6**(3), 279–298 (1992)

Brooks, S., Gelman, A., Jones, G., Meng, X.-L.: Handbook of Markov Chain Monte Carlo. CRC Press, Florida (2011)

Blasone, R.-S., Madsen, H., Rosbjerg, D.: Parameter estimation in distributed hydrological modelling: comparison of global and local optimisation techniques. Hydrology Research **38**(4-5), 451–476 (2007)

Bacci, M., Sukys, J., Reichert, P., Ulzega, S., Albert, C.: A comparison of numerical approaches for statistical inference with stochastic models. Stochastic

Environmental Research and Risk Assessment, 1–21 (2023)

Brunetti, G., Šimunek, J., Wöhling, T., Stumpp, C.: An in-depth analysis of markov-chain monte carlo ensemble samplers for inverse vadose zone modeling. Journal of Hydrology, 129822 (2023)

Carpenter, B., Gelman, A., Hoffman, M.D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M.A., Guo, J., Li, P., Riddell, A.: Stan: A probabilistic programming language. Journal of statistical software **76** (2017)

Duane, S., Kennedy, A.D., Pendleton, B.J., Roweth, D.: Hybrid monte carlo. Physics letters B **195**(2), 216–222 (1987)

Duan, Q., Schaake, J., Andréassian, V., Franks, S., Goteti, G., Gupta, H., Gusev, Y., Habets, F., Hall, A., Hay, L., *et al.*: Model parameter estimation experiment (mopex): An overview of science strategy and major results from the second and third workshops. Journal of Hydrology **320**(1-2), 3–17 (2006)

Duan, Q., Sorooshian, S., Gupta, V.: Effective and efficient global optimization for conceptual rainfall-runoff models. Water resources research **28**(4), 1015–1031 (1992)

Foreman-Mackey, D.: Autocorrelation Tutorial. Accessed: 2024-08-01 (2022). https://emcee.readthedocs.io/en/stable/tutorials/autocorr/

Foreman-Mackey, D., Hogg, D.W., Lang, D., Goodman, J.: emcee: the mcmc hammer. Publications of the Astronomical Society of the Pacific **125**(925), 306 (2013) https://doi.org/10.1086/670067

Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., Rubin, D.B.: Bayesian Data Analysis, 3rd edn., pp. 281–286. Chapman and Hall/CRC, Boca Raton, FL (2013). With errors fixed as of 15 February 2021

Gelman, A.: Prior distributions for variance parameters in hierarchical models (comment on article by browne and draper). Bayesian Analysis (2006)

Gelman, A., Gilks, W.R., Roberts, G.O.: Weak convergence and optimal scaling of random walk metropolis algorithms. The annals of applied probability **7**(1), 110–120 (1997)

Gupta, H.V., Kling, H., Yilmaz, K.K., Martinez, G.F.: Decomposition of the mean squared error and nse performance criteria: Implications for improving hydrological modelling. Journal of hydrology **377**(1-2), 80–91 (2009)

Gelman, A., Rubin, D.B.: Inference from iterative simulation using multiple sequences. Statistical science **7**(4), 457–472 (1992)

Goodman, J., Weare, J.: Ensemble samplers with affine invariance. Communications in applied mathematics and computational science **5**(1), 65–80 (2010)

Gelman, A., Yao, Y.: Holes in bayesian statistics. Journal of Physics G: Nuclear and Particle Physics **48**(1), 014002 (2020)

Hastings, W.K.: Monte carlo sampling methods using markov chains and their applications (1970)

Hogg, D.W., Foreman-Mackey, D.: Data analysis recipes: Using markov chain monte carlo. The Astrophysical Journal Supplement Series **236**(1), 11 (2018)

Hoffman, M.D., Gelman, A., *et al.*: The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. J. Mach. Learn. Res. **15**(1), 1593–1623 (2014)

Hughes, J.D., Langevin, C.D., Paulinski, S.R., Larsen, J.D., Brakenhoff, D.: FloPy Workflows for Creating Structured and Unstructured MODFLOW Models. Groundwater (2023) https://doi.org/10.1111/gwat.13327

Hendrickson, J., Sorooshian, S., Brazil, L.E.: Comparison of newton-type and direct search algorithms for calibration of conceptual rainfall-runoff models. Water Resources Research **24**(5), 691–700 (1988)

Hydrogeologic, W.: IMS Package Translation Settings - Visual MODFLOW Flex 10.0. Accessed: 2024-09-18 (2024). https://www.waterloohydrogeologic.com/help/vmod-flex/index.html?vm_mf6_ims_translationsettings.htm

Johnson, A.A., Ott, M.Q., Dogucu, M.: Bayes Rules!: An Introduction to Applied Bayesian Modeling. CRC Press, Florida (2022)

Krapu, C., Borsuk, M.: A differentiable hydrology approach for modeling with time-varying parameters. Water Resources Research **58**(9), 2021–031377 (2022)

Kumar, R., Carroll, C., Hartikainen, A., Osvaldo, M.: ArviZ a unified library for exploratory analysis of Bayesian models in Python. Journal of Open Source Software (2019) https://doi.org/10.21105/joss.01143

Keating, E.H., Doherty, J., Vrugt, J.A., Kang, Q.: Optimization and uncertainty assessment of strongly nonlinear groundwater models with high parameter dimensionality. Water Resources Research **46**(10) (2010)

Lambert, B.: A student's guide to bayesian statistics. A Student's Guide to Bayesian Statistics, 1–520 (2018)

Langevin, C., Hughes, J., Banta, E., Provost, A., Niswonger, R., Panday, S.: Modflow 6 modular hydrologic model: Us geological survey software. US

Geological Survey (2017) https://doi.org/10.5066/P9FL1JCC

Laloy, E., Vrugt, J.A.: High-dimensional posterior exploration of hydrologic models using multiple-try dream (zs) and high-performance computing. Water Resources Research **48**(1) (2012)

MacKay, D.J.: Information Theory, Inference and Learning Algorithms. Cambridge university press, Cambridge, UK (2003)

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. The journal of chemical physics **21**(6), 1087–1092 (1953)

Robert, C.P., Casella, G., Robert, C.P., Casella, G.: The metropolis—hastings algorithm. Monte Carlo statistical methods, 267–320 (2016)

Refsgaard, J.C.: Parameterisation, calibration and validation of distributed hydrological models. Journal of hydrology **198**(1-4), 69–97 (1997)

Rau, G.C., Post, V.E., Shanafield, M., Krekeler, T., Banks, E.W., Blum, P.: Error in hydraulic head and gradient time-series measurements: a quantitative appraisal. Hydrology and Earth System Sciences **23**(9), 3603–3629 (2019)

SAS Institute Inc.: SAS/STAT® 15.2 User's Guide, 15.2 edn. (2020). Chap. The MCMC Procedure. Example 79.20 Using a Transformation to Improve Mixing. Last updated: October 28, 2020. https://documentation.sas.com/doc/en/statug/15.2/statug_mcmc_examples29.htm

Schmon, S.M., Gagnon, P.: Optimal scaling of random walk metropolis algorithms using bayesian large-sample asymptotics. Statistics and Computing **32**(2), 28 (2022)

Sokal, A.: Monte carlo methods in statistical mechanics: foundations and new algorithms. In: Functional Integration: Basics and Applications, pp. 131–192. Springer, Boston, MA (1997)

Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization **11**, 341–359 (1997)

Stan Development Team: Reparameterization and Changes of Variables. Accessed: 17 Feb. 2025. Permalink: https://github.com/stan-dev/docs/blob/ad347b8755a4e3eda21d2403794d1010c9ac16eb/src/stan-users-guide/reparameterization.qmd (2025). https://mc-stan.org/docs/stan-users-guide/reparameterization.html#changes-of-variables

Ter Braak, C.J.: A markov chain monte carlo version of the genetic algorithm differential evolution: easy bayesian computing for real parameter spaces. Statistics and Computing **16**, 239–249 (2006)

Ter Braak, C.J., Vrugt, J.A.: Differential evolution markov chain with snooker updater and fewer chains. Statistics and Computing **18**, 435–446 (2008)

Ulzega, S., Albert, C.: Bayesian parameter inference in hydrological modelling using a hamiltonian monte carlo approach with a stochastic rain model. Hydrology and Earth System Sciences **27**(15), 2935–2950 (2023)

Vrugt, J.A., Gupta, H.V., Bouten, W., Sorooshian, S.: A shuffled complex evolution metropolis algorithm for optimization and uncertainty assessment of hydrologic model parameters. Water resources research **39**(8) (2003)

Vrugt, J.A., Gupta, H.V., Dekker, S.C., Sorooshian, S., Wagener, T., Bouten, W.: Application of stochastic parameter optimization to the sacramento soil moisture accounting model. Journal of Hydrology **325**(1-4), 288–307 (2006)

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods **17**, 261–272 (2020) https://doi.org/10.1038/s41592-019-0686-2

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., Bürkner, P.-C.: (rank-normalization, folding, and localization: An improved $\hat{R}$ for assessing convergence of mcmc (with discussion). Bayesian analysis **16**(2), 667–718 (2021) https://doi.org/10.1214/20-BA1221

Vrugt, J.A., Ter Braak, C., Diks, C., Robinson, B.A., Hyman, J.M., Higdon, D.: Accelerating markov chain monte carlo simulation by differential evolution with self-adaptive randomized subspace sampling. International journal of nonlinear sciences and numerical simulation **10**(3), 273–290 (2009)

Waskom, M.L.: seaborn: statistical data visualization. Journal of Open Source Software **6**(60), 3021 (2021) https://doi.org/10.21105/joss.03021

Woessner, W.W., Poeter, E.P.: Hydrogeologic Properties of Earth Materials and Principles of Groundwater Flow. Groundwater Project, Waterloo, Ontario, Canada (2020). Chap. 5.3 Hydraulic Conductivity Values for Earth Materials

Appendix A Search strategy in Scopus

This section explains how the data for Table 1 was acquired. For convenience a copy of Table Table 1 is given below (Table A1). In order to identify which hydrological journals discuss MCMC most frequently, the following search query was carried out (28-04-2024) in Scopus:

> **Query:** TITLE-ABS-KEY (mcmc OR "markov chain monte carlo") AND TITLE-ABS-KEY (hydrolog* OR hydrogeolog* OR "subsurface flow*" OR groundwater)

This query resulted in 542 publications (in journals specifically) up to and including the year 2023. The journals returning most result from this query were:

- 87 from: Water Resources Research
- 70 from: Journal of Hydrology
- 26 from: Stochastic Environmental Research and Risk Assessment
- 21 from: Advances in Water Resources
- 12 from: Journal of Hydrologic engineering

Stochastic Environmental Research and Risk Assessment was omitted, as it is not specifically tailored to hydrology. Therefore, the three selected journals were: Water Resources Research, Journal of Hydrology & Advances in Water Resources. It was decided to not include more journals, because that would make it more time consuming to perform queries for the different MCMC algorithms with strongly diminishing returns regarding the number of papers mentioning MCMC.

For the algorithms presented in Table 1 the number of citations (by journals up to and including the year 2023) of the paper introducing the specific algorithm were looked up in Scopus, resulting in the 2$^{\text{nd}}$ column. To give an indication of the popularity in hydrology, the search was narrowed down to the 3 selected hydrological journals (presented in 3$^{\text{rd}}$ column).

Table A1: A comparison of the popularity of several MCMC algorithms between all fields of science and hydrology specifically. Popularity is quantified by counting how often the paper introducing the specific algorithms is cited. Three hydrological journals have been selected to indicate the popularity in hydrology: Journal of hydrology, Water Resources Research & Advances in Water Resources. These are the three journals where MCMC methods are most discussed, while specifically tailored to hydrology (Appendix A).

MCMC Method	citation in all journals	citations in hydrological journals	paper introducing algorithm
Metropolis-Hastings	7922	123	Hastings (1970)
Differential Evolution (DE)	565	32	Ter Braak (2006)
DE with snooker updater (DE-SNK)	338	53	Ter Braak and Vrugt (2008)
DREAM	777	196	Vrugt et al. (2009)
Multiple-Try DREAM$_{(zs)}$	360	93	Laloy and Vrugt (2012)
Affine Invariant sampler (AI)	1903	9	Goodman and Weare (2010)
Hamiltonian Monte Carlo (HMC)	2154	10	Duane et al. (1987)
No-U-Turn Sampler (NUTS)	1886	17	Hoffman et al. (2014)

Appendix B Generative AI

Generative artificial intelligence (ChatGPT) was used to assist with writing: its suggestions were used to improve the flow of text and to correct spelling and grammar mistakes. In addition, ChatGPT was used to help with formatting, structuring, and debugging the LaTeXdocument. Finally, ChatGPT was used to provide assistance with programming in Python. It was mainly used as a debugging tool, but also to suggest ideas on how to approach specific challenges.

Appendix C Priors

Table C2: Lithology of all layers in Model 1-4, where lithology is limited to three sediment classes: clean sand, silty sand and silt/loess. The true hydraulic conductivities of each layer are given between parentheses in meters per day. Layer 1 always refers to the uppermost layer, with deeper layers having sequentially greater numbers.

Model	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5
1	clean sand (5.0)				
2	clean sand (2.0)	silty sand (1.0)			
3	silty sand (1.0)	silt, loess (0.01)	clean sand (10.0)		
4	silty sand (1.0)	silt, loess (0.1)	clean sand (4.0)	silt, loess (0.01)	clean sand (3.0)

Appendix D Additional Results

D.1 Kernel Density Estimate plots and Trace Plots

Shown are the Kernel Density Estimate plots of Model 1 for each sampler. In addition, trace plots are shown for the scenario with 5 observations, because here the second mode is relatively pronounced. Trace plots show all 10 chains of the selected ensembles. For each sampler one ensemble was selected for the wide prior and one for the narrow prior, based on their $\hat{R}$, with the ensembles with the highest $\hat{R}$ being selected. These KDE plots and trace plots for AI, DE and DE-SNK are given in Sections D.1.1, D.1.2 and D.1.3, respectively. Finally, pairs plots for AI and DE are given in Section D.2.

D.1.1 AI

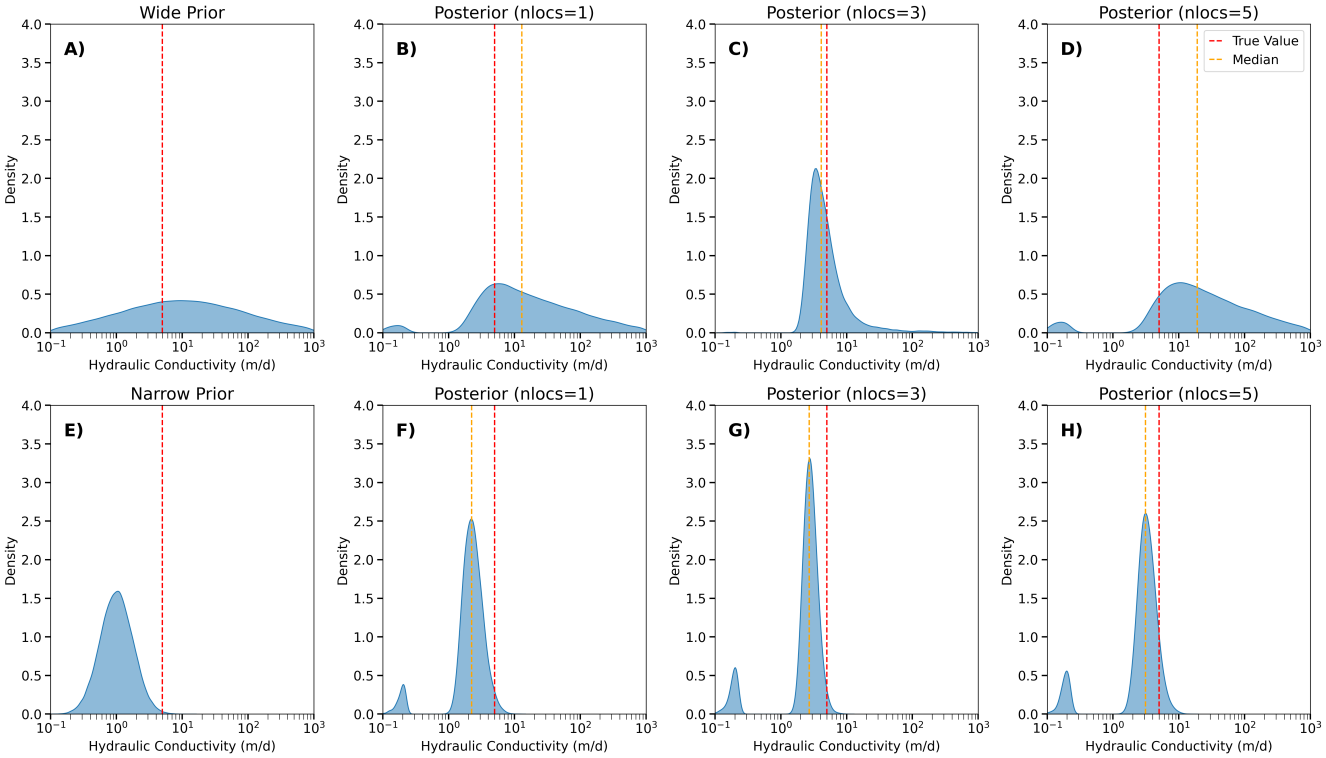


Fig. D1: Kernel Density Estimate plots for the only parameter of Model 1 (θ_1), which was calibrated with AI. **A)** shows the uninformative prior. **B), C)** and **D)** show the posteriors after calibrating θ_1 with: the uninformative prior in combination with 1,3 or 5 observations for evaluating the likelihood. Similarly, **E)** shows the informative prior. **F), G)** and **H)** show the posteriors after calibrating θ_1 with: the informative prior in combination with 1,3 or 5 observations for evaluating the likelihood. In each plot the dashed red line indicates the true value for θ_1. Similarly, the posterior median is indicated by a dashed yellow line.

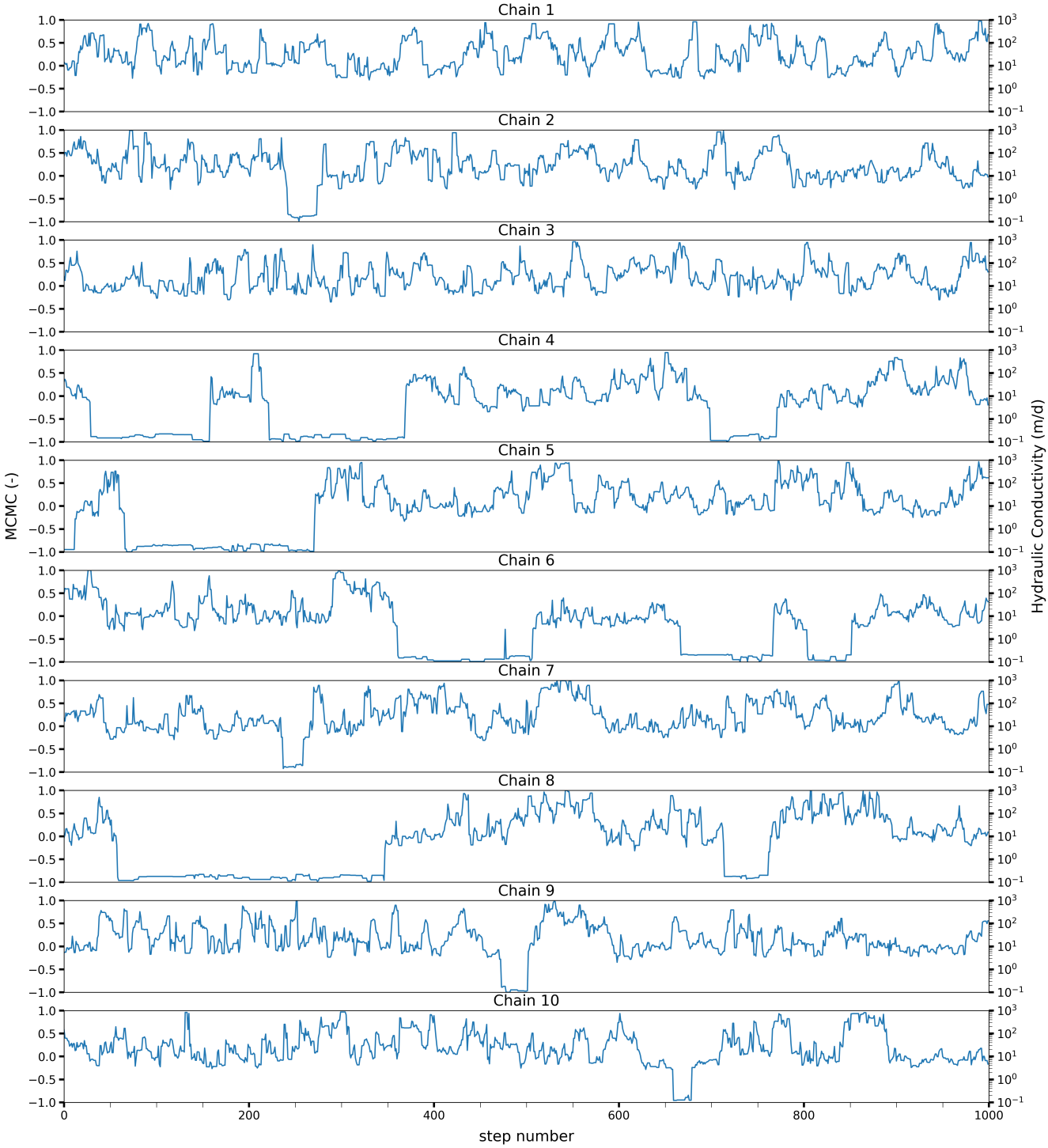


Fig. D2: Trace plots for ensemble 5 of AI, from calibrating θ_1 in Model 1 with the wide prior and 5 observations. All post burn-in steps are shown for all 10 chains, with the step number given on the x-axis. On the left y-axis the parameter values as used by the MCMC algorithm are shown. And on the right y-axis the transformed parameter values, used by MODFLOW 6, for calculation of the likelihood. For this ensemble $\hat{R} = 1.11$.

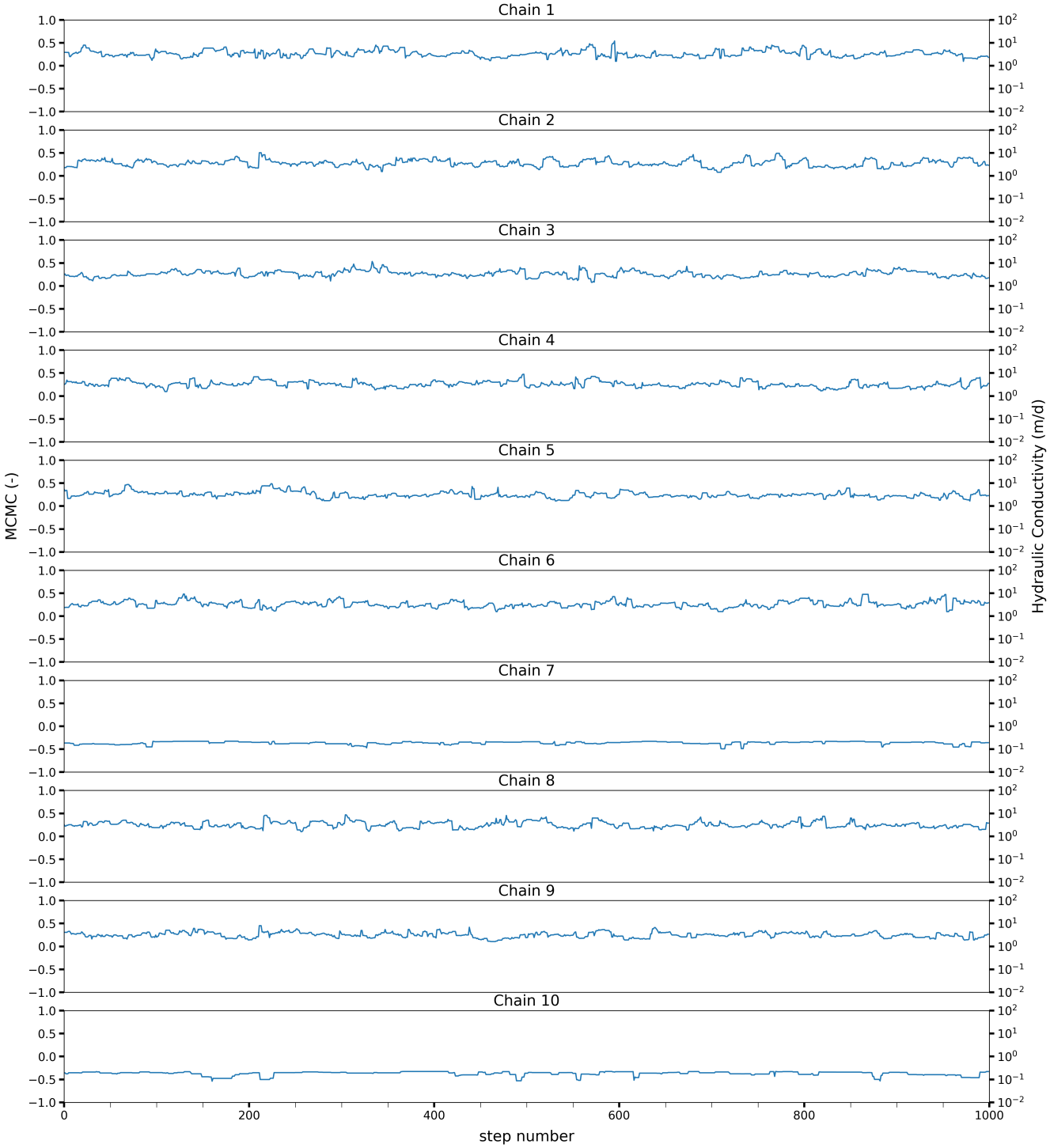


Fig. D3: Trace plots for ensemble 2 of AI from calibrating θ_1 in Model 1 with the narrow prior and 5 observations. All post burn-in steps are shown for all 10 chains, with the step number given on the x-axis. On the left y-axis the parameter values as used by the MCMC algorithm are shown. And on the right y-axis the transformed parameter values, used by MODFLOW 6, for calculation of the likelihood. For this ensemble $\hat{R} = 1.44$.

D.1.2 DE

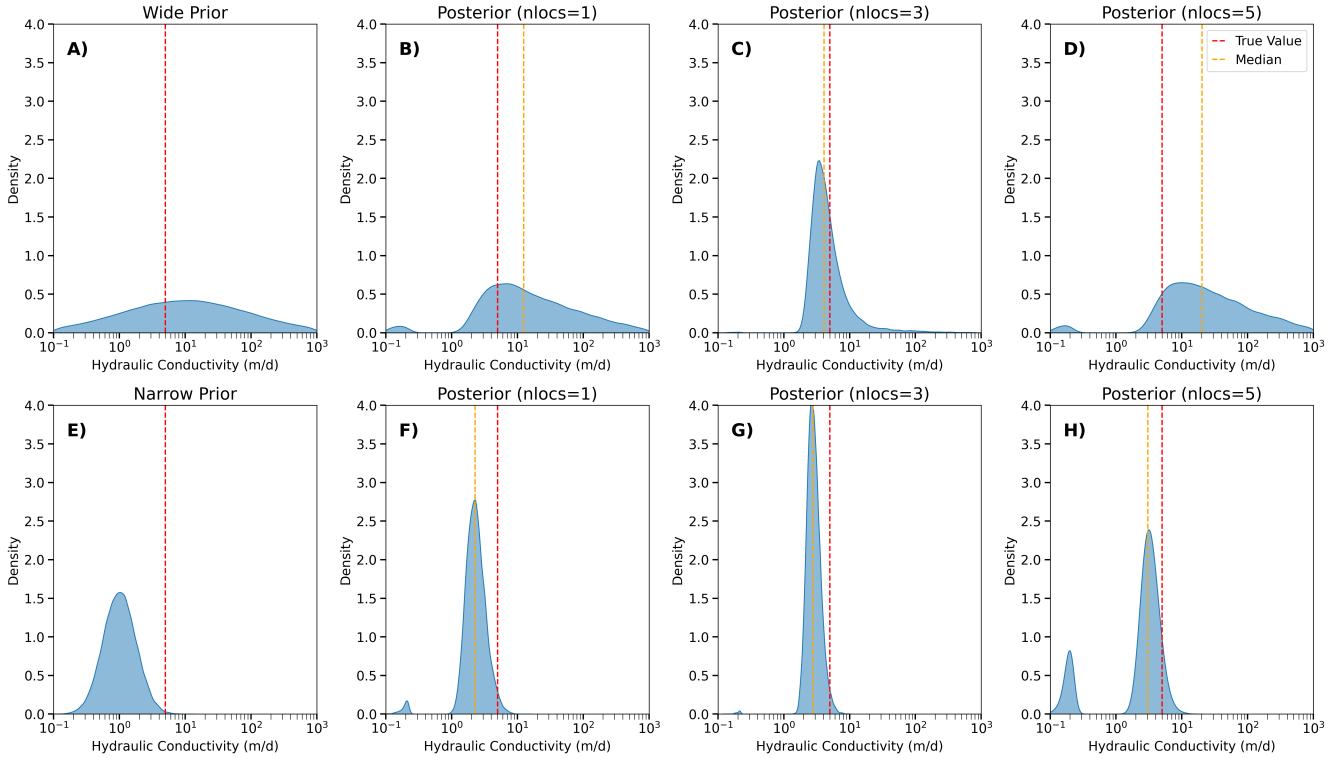


Fig. D4: Kernel Density Estimate plots for the only parameter of Model 1 (θ_1), which was calibrated with DE. **A)** shows the uninformative prior. **B), C)** and **D)** show the posteriors after calibrating θ_1 with: the uninformative prior in combination with 1,3 or 5 observations for evaluating the likelihood. Similarly, **E)** shows the informative prior. **F), G)** and **H)** show the posteriors after calibrating θ_1 with: the informative prior in combination with 1,3 or 5 observations for evaluating the likelihood. In each plot the dashed red line indicates the true value for θ_1). Similarly, the posterior median is indicated by a dashed yellow line.

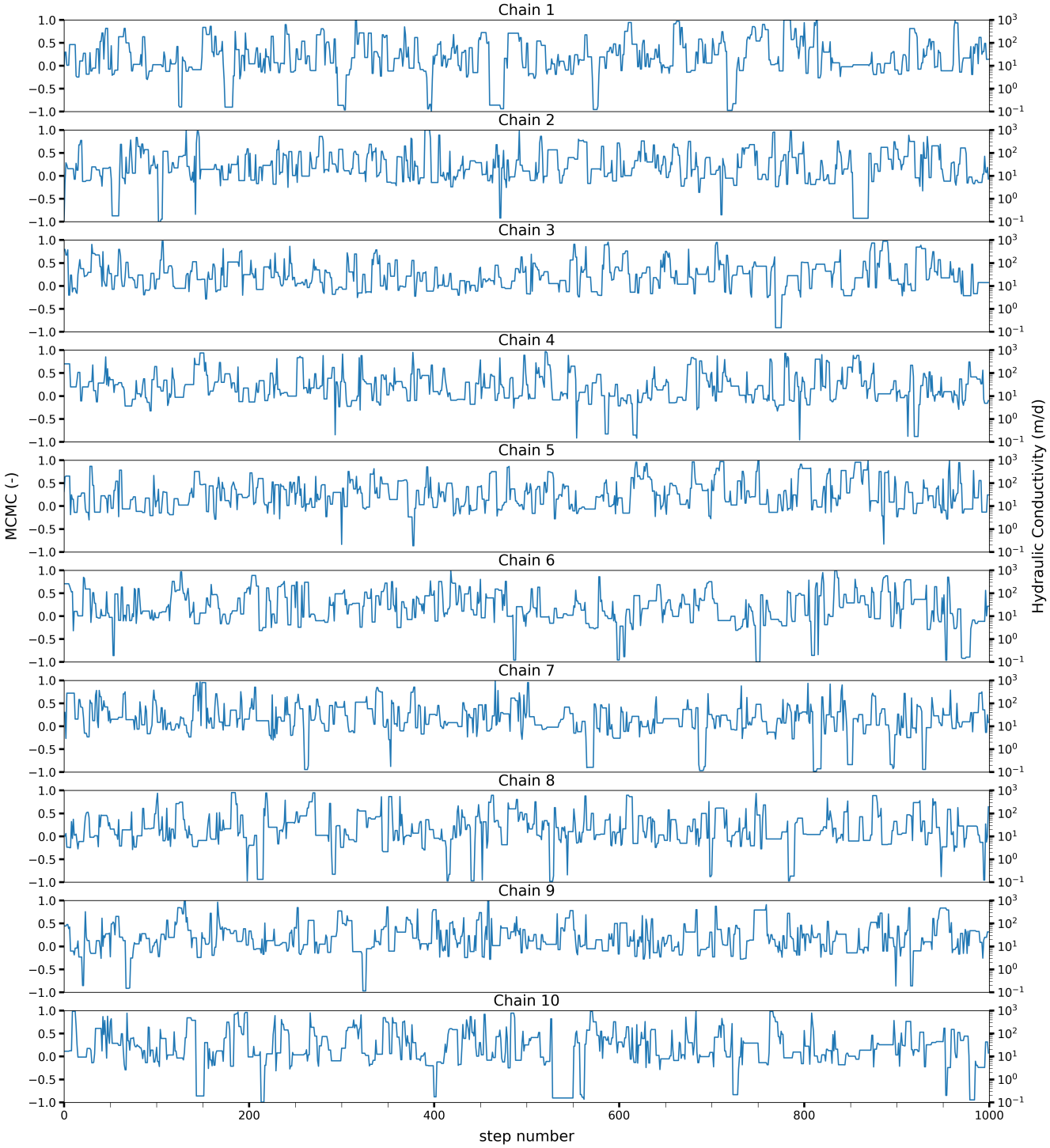


Fig. D5: Trace plots for ensemble 1 of DE from calibrating θ_1 in Model 1 with the wide prior and 5 observations. All post burn-in steps are shown for all 10 chains, with the step number given on the x-axis. On the left y-axis the parameter values as used by the MCMC algorithm are shown. And on the right y-axis the transformed parameter values, used by MODFLOW 6, for calculation of the likelihood. For this ensemble $\hat{R} = 1.01$.

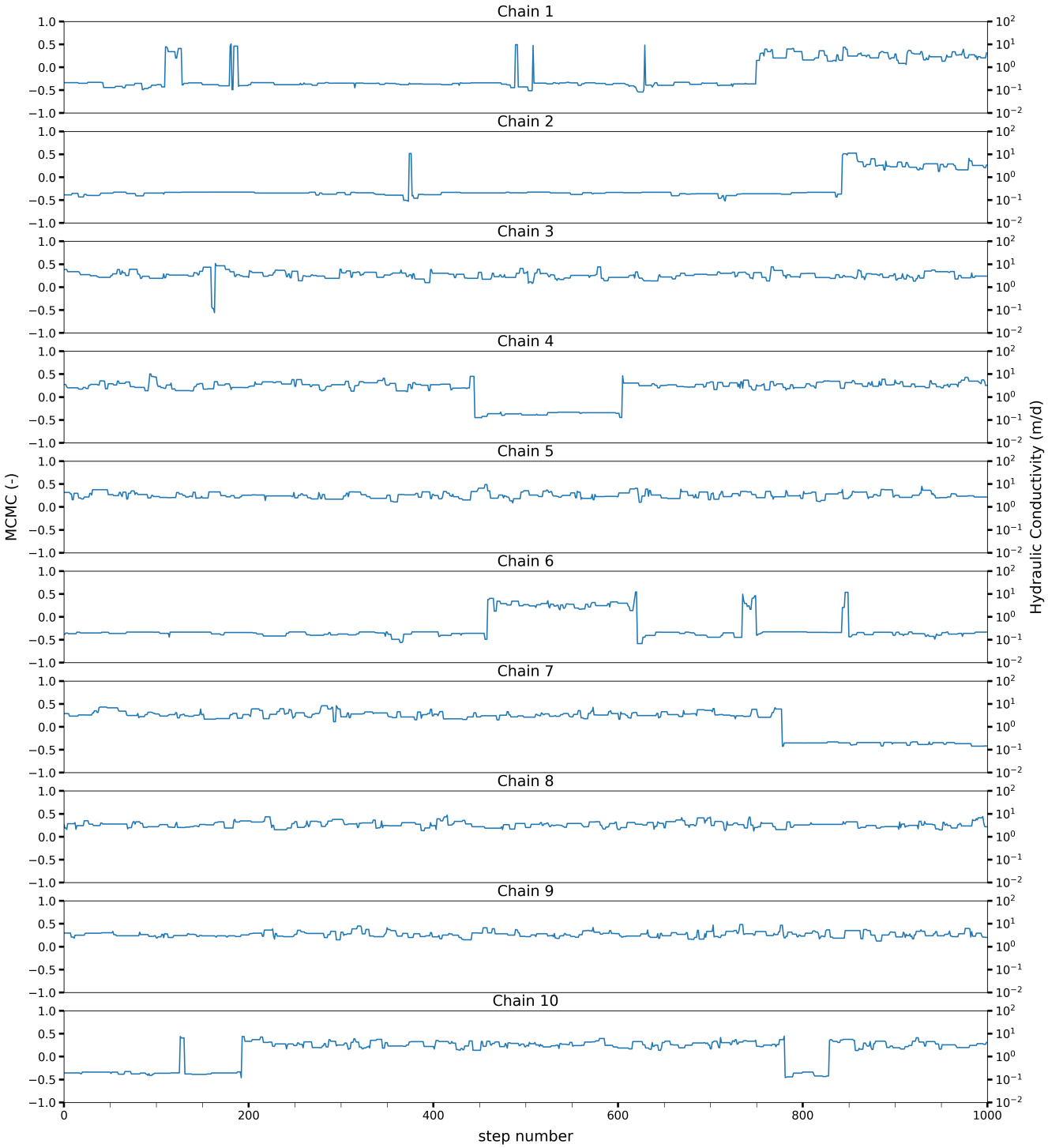


Fig. D6: Trace plots for ensemble 5 of DE from calibrating θ_1 in Model 1 with the narrow prior and 5 observations. All post burn-in steps are shown for all 10 chains, with the step number given on the x-axis. On the left y-axis the parameter values as used by the MCMC algorithm are shown. And on the right y-axis the transformed parameter values, used by MODFLOW 6, for calculation of the likelihood. For this ensemble $\hat{R} = 1.27$.

D.1.3 DE-SNK

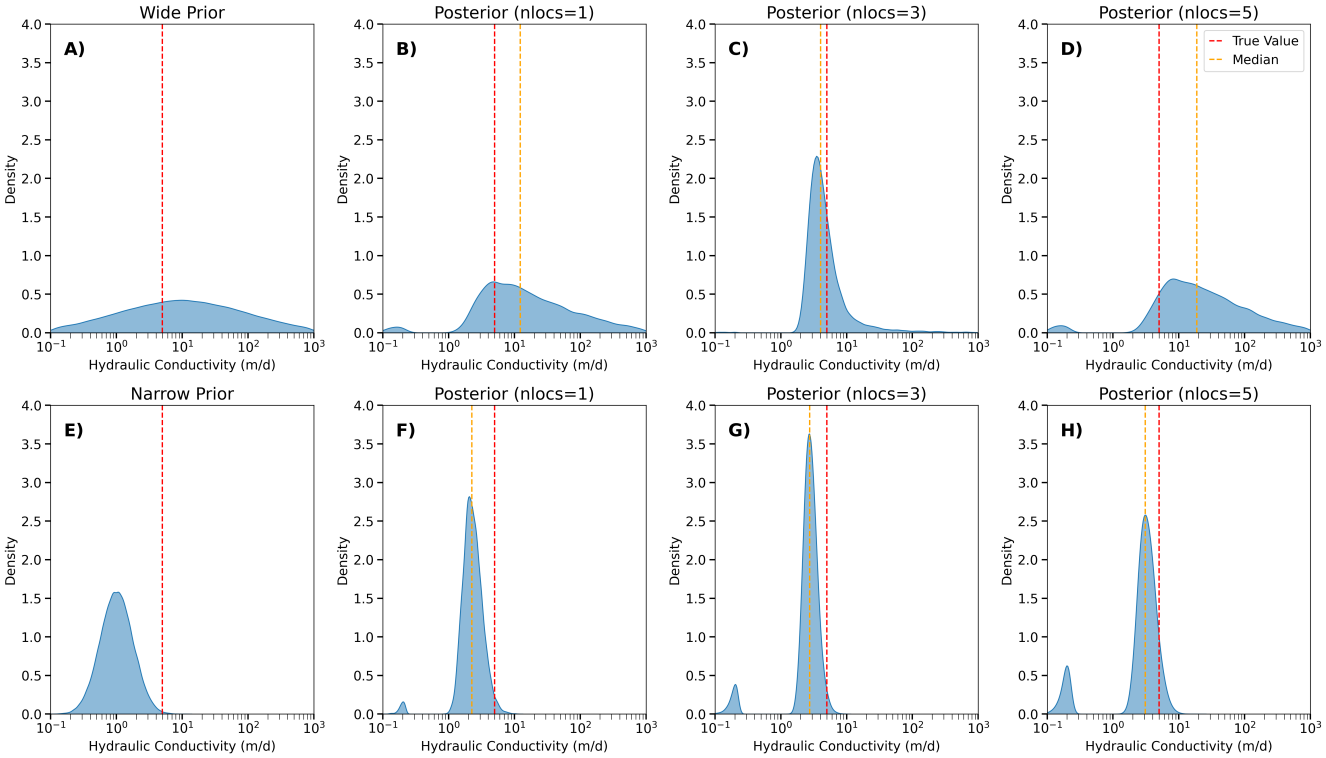


Fig. D7: Kernel Density Estimate plots for the only parameter of Model 1 (θ_1), which was calibrated with DE-SNK. **A)** shows the wide prior. **B), C)** and **D)** show the posteriors after calibrating θ_1 with: the wide prior in combination with 1,3 or 5 observations for evaluating the likelihood, respectively. Similarly, **E)** shows the narrow prior. **F), G)** and **H)** show the posteriors after calibrating θ_1 with: the narrow prior in combination with 1,3 or 5 observations for evaluating the likelihood. In each plot the dashed red line indicates the true value for θ_1. Similarly, the posterior median is indicated by a dashed yellow line.

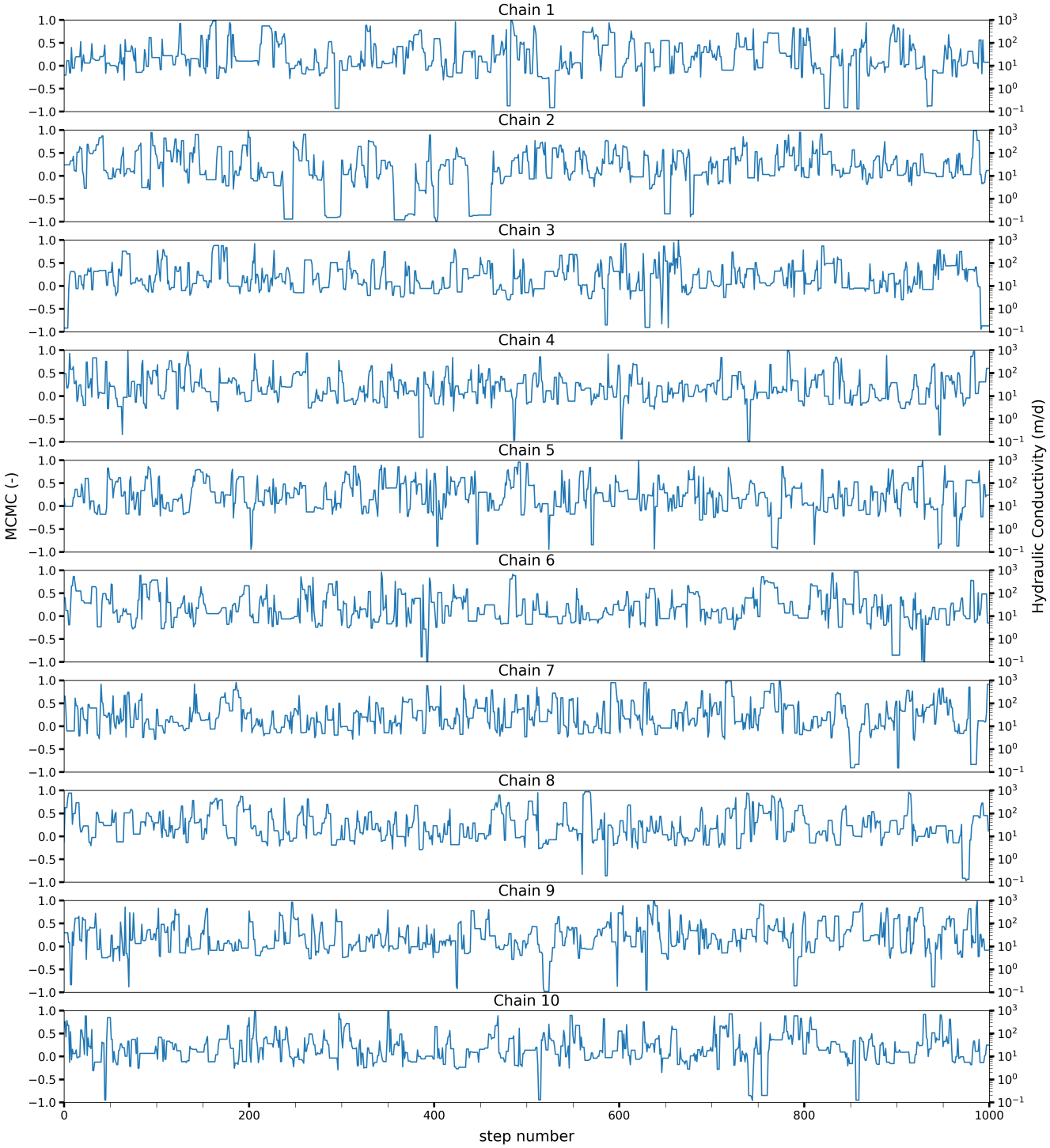


Fig. D8: Trace plots for ensemble 4 of DE-SNK from calibrating θ_1 in Model 1 with the wide prior and 5 observations. All post burn-in steps are shown for all 10 chains, with the step number given on the x-axis. On the left y-axis the parameter values as used by the MCMC algorithm are shown. And on the right y-axis the transformed parameter values, used by MODFLOW 6, for calculation of the likelihood. For this ensemble $\hat{R} = 1.01$.

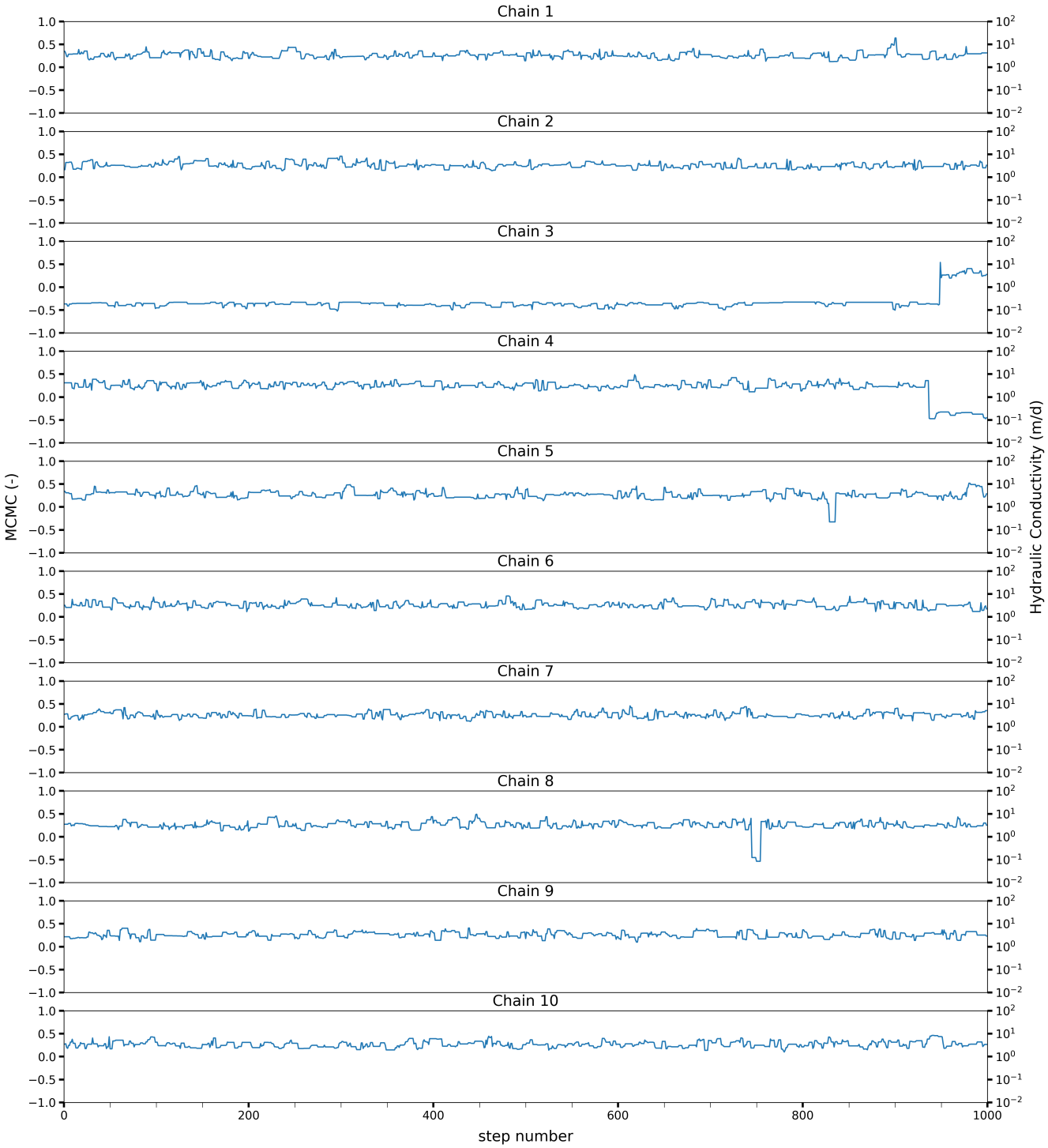


Fig. D9: Trace plots for ensemble 3 of DE-SNK from calibrating θ_1 in Model 1 with the narrow prior and 5 observations. All post burn-in steps are shown for all 10 chains, with the step number given on the x-axis. On the left y-axis the parameter values as used by the MCMC algorithm are shown. And on the right y-axis the transformed parameter values, used by MODFLOW 6, for calculation of the likelihood. For this ensemble $\hat{R} = 1.21$.

D.2 Pairs plots

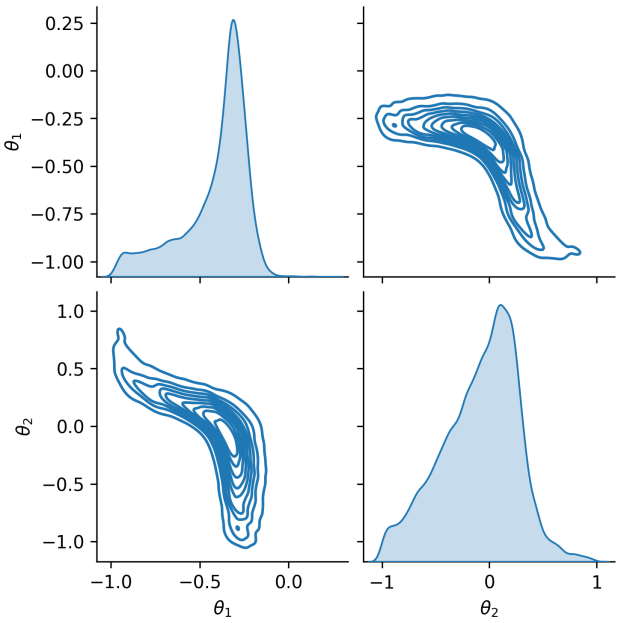


Fig. D10: Pair plots (with a kernel density style) for all calibrated parameters of Model 2. The presented data is from 5 ensembles (last 1000 steps of each chain), run with DE, an uninformative prior and 1 observation for evaluating the likelihood. The parameters as presented here are transformed for MCMC.

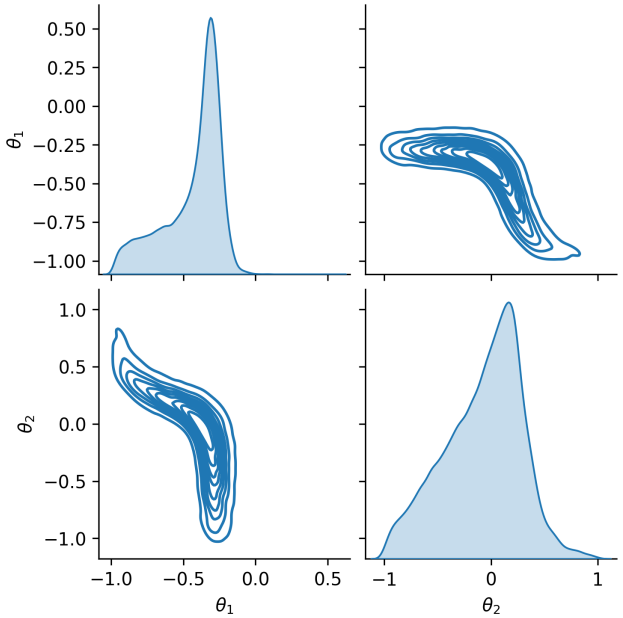


Fig. D11: Pairs plots (with a kernel density style) for all calibrated parameters of Model 2. The presented data is from 5 ensembles (last 1000 steps of each chain), run with AI, an uninformative prior and 1 observation for evaluating the likelihood. The parameters as presented here are transformed for MCMC.

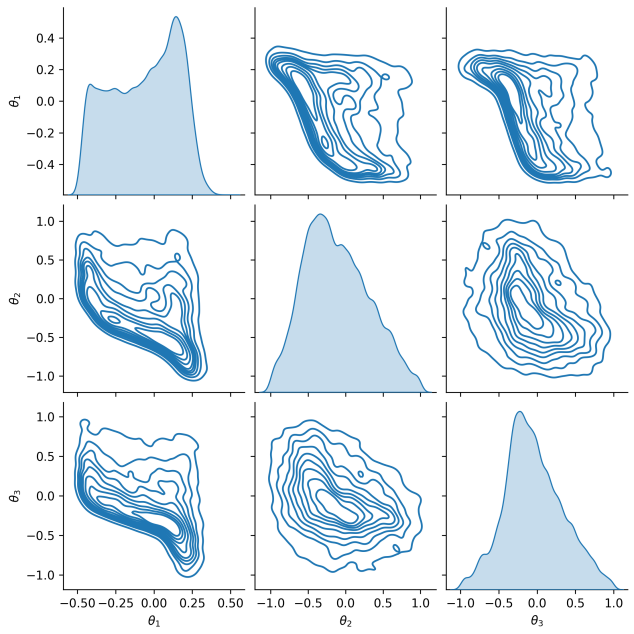


Fig. D12: Pairs plots (with a kernel density style) for all calibrated parameters of Model 3. The presented data is from 5 ensembles (last 1000 steps of each chain), run with DE, an uninformative prior and 1 observation for evaluating the likelihood. The parameters as presented here are transformed for MCMC.

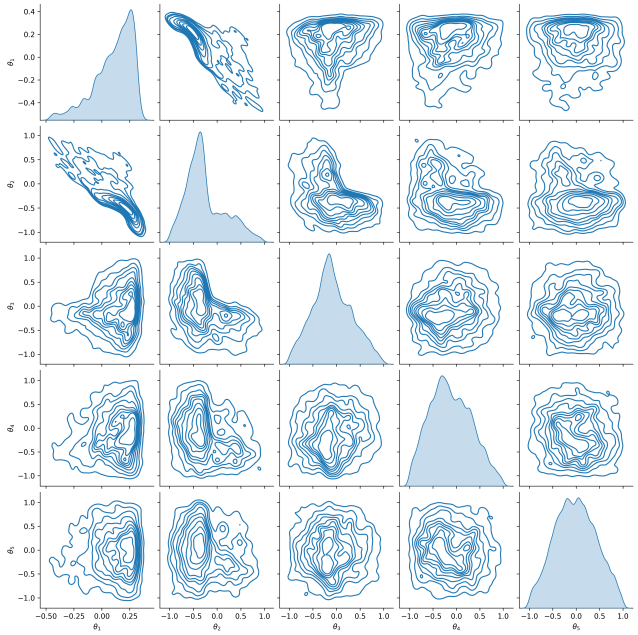


Fig. D14: Pairs plots (with a kernel density style) for all calibrated parameters of Model 4. The presented data is from 5 ensembles (last 1000 steps of each chain), run with DE, an uninformative prior and 1 observation for evaluating the likelihood. The parameters as presented here are transformed for MCMC.

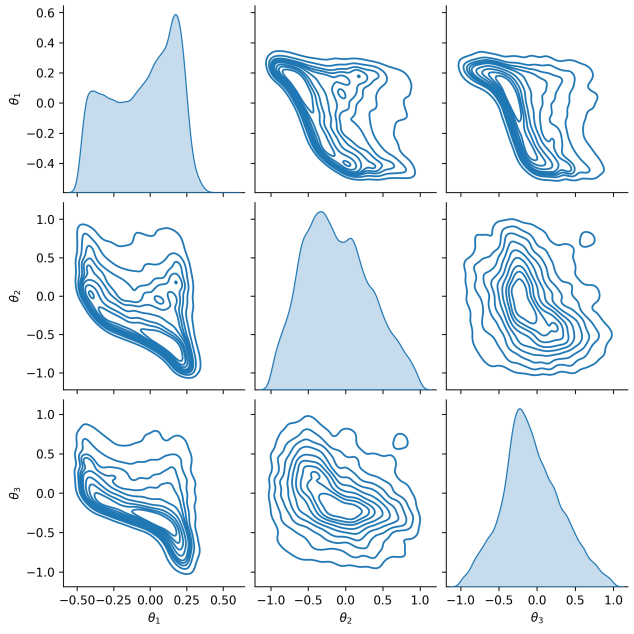


Fig. D13: Pairs plots (with a kernel density style) for all calibrated parameters of Model 3. The presented data is from 5 ensembles (last 1000 steps of each chain), run with AI, an uninformative prior and 1 observation for evaluating the likelihood. The parameters as presented here are transformed for MCMC.

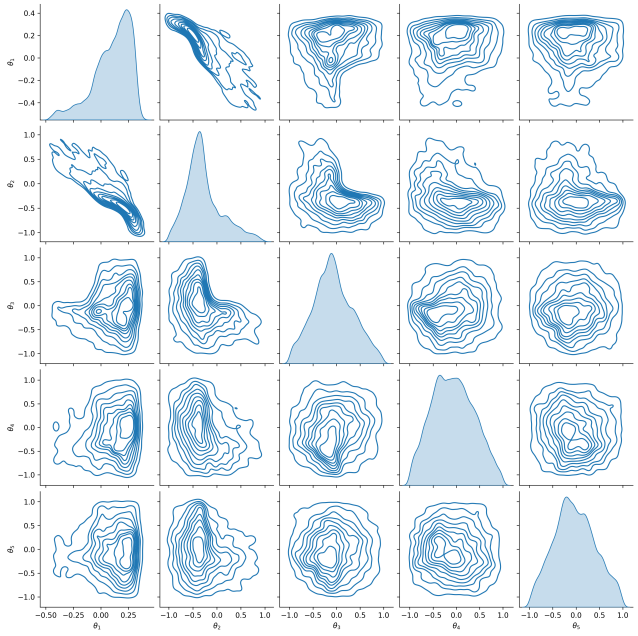


Fig. D15: Pairs plots (with a kernel density style) for all calibrated parameters of Model 4. The presented data is from 5 ensembles (last 1000 steps of each chain), run with AI, an uninformative prior and 1 observation for evaluating the likelihood. The parameters as presented here are transformed for MCMC.

Appendix E Trial runs

Poor chain convergence was identified during trial runs, as shown in Section E.2. This is has been attributed to the absence of parameter transformation, as convergence improved drastically after implementing it during testing. The Methodology of these trial runs is explained in Section E.1. The trial runs were similar to the final runs (same models and samplers), but there were also some notable differences. Furthermore, MODFLOW raised an *AssertionError* for some parameter sets during these trial runs. How these errors have been resolved is explained in Section E.3.

E.1 Methodology

Three samplers (DE, DE-SNK and AI) were used to calibrate four models (Figure E16). Each model was calibrated with 3 ensembles (per sampler) consisting of 10 chains of 1500 steps each. All chains were run for 500 steps burn-in and an additional 1000 steps of main-sampling. The different samplers were initialised at the same position for fair comparison, with starting positions selected stochastically from Unif(0.0001,20). Priors for all parameters were set to N(10,10) and measurement uncertainty, which determines the likelihood function, was set to N(0,0.2). No parameter transformation was applied, so both for sampling and likelihood evaluation the same parameter values for the hydraulic conductivity (m/d) were used.

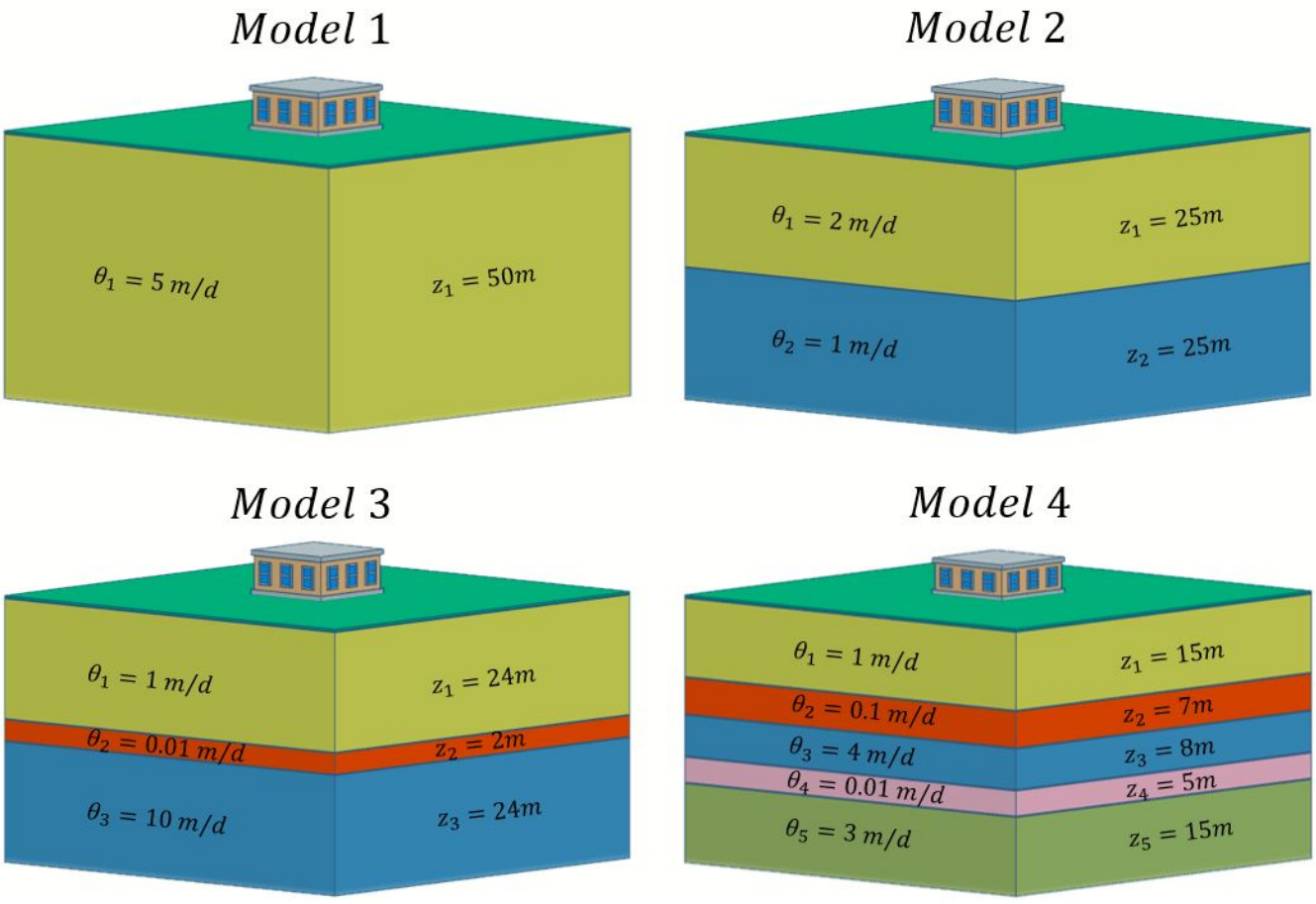


Fig. E16: Schematic of the four steady-state synthetic groundwater flow models used in this study. Layer thickness and the respective isotropic hydraulic conductivities are shown inside each layer. The models have a length and width of 1000 meters, consisting of 25 rows and columns, with a length of 40 meters respectively. The total depth of each model equals 50 meters. At the centre of each model is an abstraction well (indicated by the building), extending over the full depth of the phreatic aquifer, with an extraction rate of 500 m^3/day. At the edges of each model, there are constant head boundaries of 10 meters.

E.2 Trace plots

Trace plots are presented for model four to investigate whether the different moves explore the parameter space as intended. Model four was selected, because it has the highest number of parameters to be calibrated (5). The ensemble and subsequent chain selection was arbitrary.

The first 600 steps of the AI sampler of a specific chain show little variation in parameter values (Figure E17), suggesting high autocorrelation and low acceptance fraction for proposals. This problem appears to be even worse for the DE sampler (Figure E18) and for DE-SNK (Figure E19), with the chains moving only roughly every hundred steps. Another problem is that parameter 2 in Figure E17 appears to never move.

The acceptance fraction for these specific chains has been calculated to verify whether the chains indeed exhibit the infrequent movement suggested by the trace plots, or if the moves are so small that they are not detectable in the trace plots. The acceptance fraction for different parameter within specific chains does not vary (Table E3), meaning that parameter 2 in Figure E17 does move as intended, but due to the scale it is not visible. All acceptance fractions are low, as was expected based on the trace plots, with DE scoring the worst with roughly 1 in 100 proposals being accepted. These acceptance fractions are very low compared to the ideal acceptance fractions, which, as a rule of thumb, should be between 0.2 and 0.5 (Foreman-Mackey et al., 2013).

Table E3: Compares the acceptance fraction of proposed steps by different samplers (AI, DE, DE-SNK) for calibrating Model four. The presented acceptance fractions for each parameter (θ) are from the 2nd chain of the 3rd ensemble of each sampler.

	θ_1	θ_2	θ_3	θ_4	θ_5
AI	0.059	0.059	0.059	0.059	0.059
DE	0.011	0.011	0.011	0.011	0.011
DE-SNK	0.020	0.020	0.020	0.020	0.020

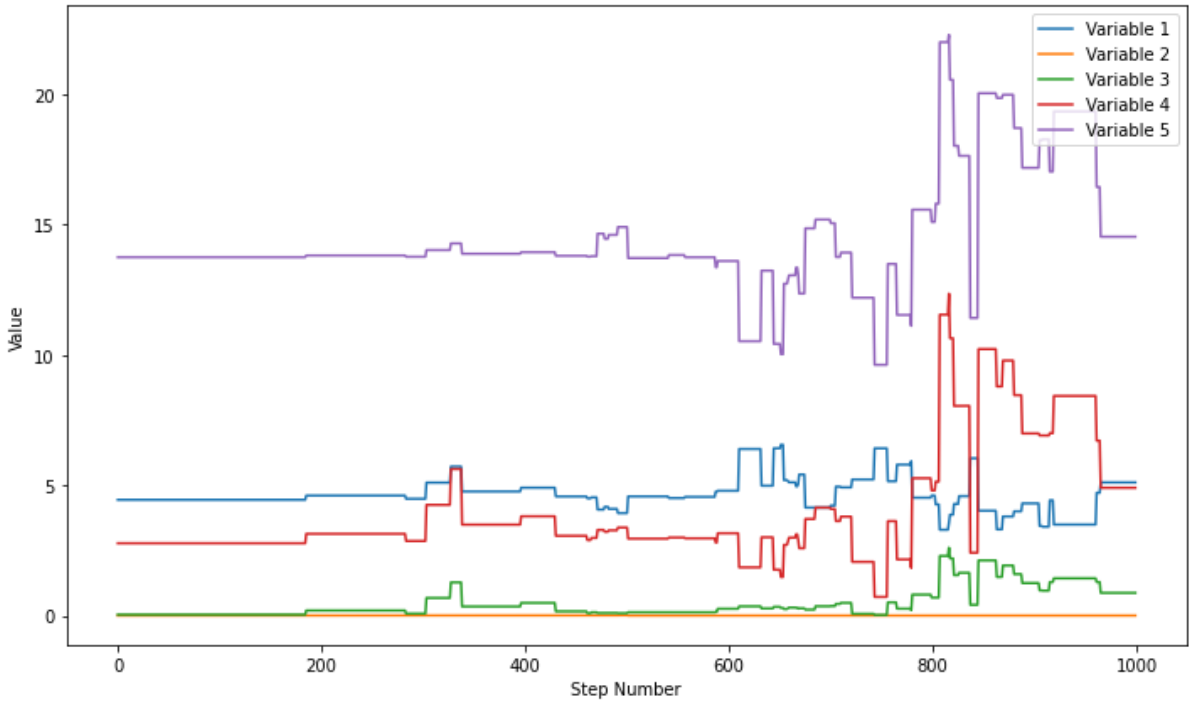


Fig. E17: A trace plot of the Stretch move for calibrating model four. The presented chain is the 2nd chain of the 3rd ensemble. Only the 1000 steps of the main sampling are presented, the 500 burn-in steps were discarded.

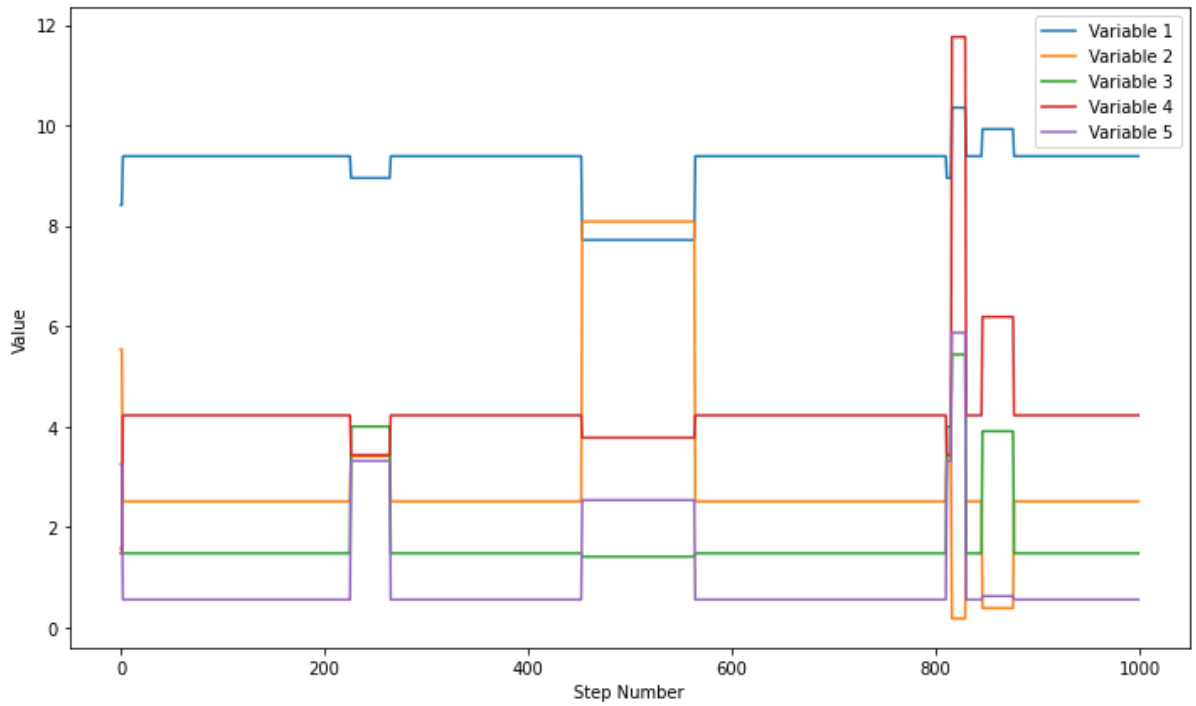


Fig. E18: A trace plot of the Differential Evolution move for calibrating model four. The presented chain is the 2nd chain of the 3rd ensemble. Only the 1000 steps of the main sampling are presented, the 500 burn-in steps were discarded.

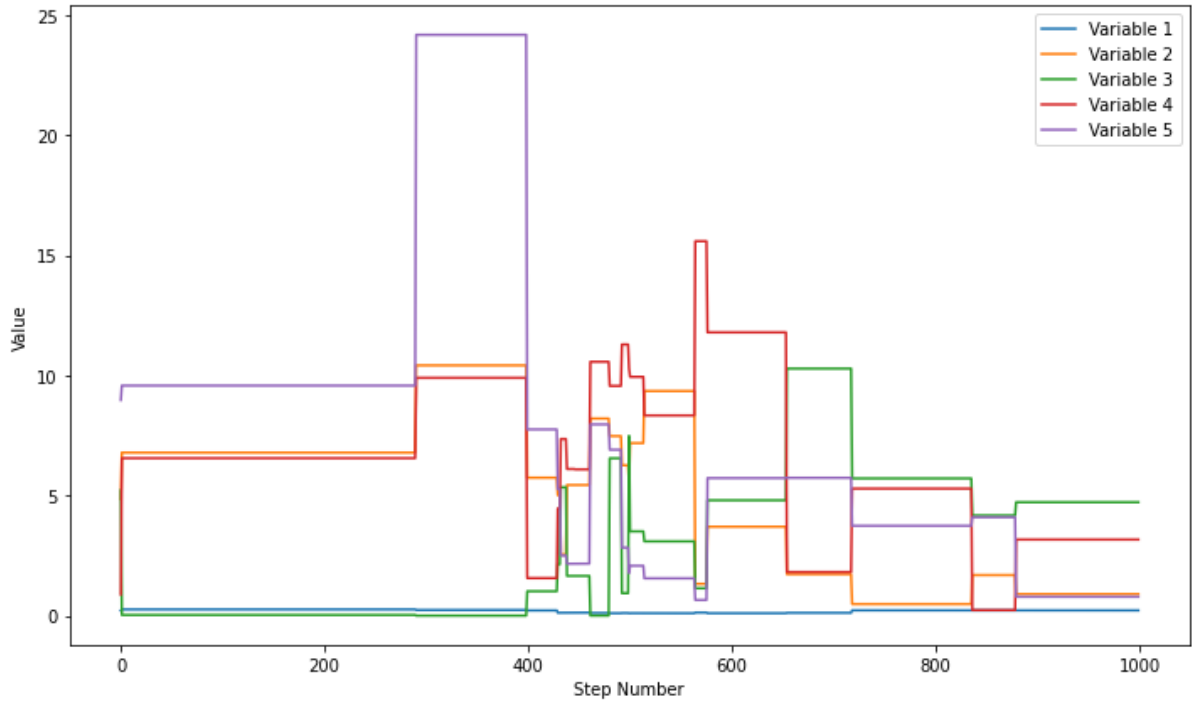


Fig. E19: A trace plot of a combination of Differential Evolution (frequency 0.8) & snooker update (frequency 0.2) moves for calibrating model four. The presented chain is the 2nd chain of the 3rd ensemble. Only the 1000 steps of the main sampling are presented, the 500 burn-in steps were discarded.

E.3 MODFLOW simulation errors

MODFLOW raised an *AssertionError* for some parameter sets, during trial runs. In total 185 AssertionErrors were encountered in approximately 135,000 simulations (3 samplers * 3 ensembles * 10 chains * 1500 steps). AssertionErrors were encounterd from all models, with the majority from Model 4.

These errors can be resolved in different ways. By default the selected solver is the Simple solver, where Simple indicates that default solver input values will be defined that work well for nearly linear models. This option is generally suitable for models that do not include nonlinear stress packages and models that are either confined or consist of a single unconfined layer that is thick enough to contain the water table within a single layer (Hydrogeologic, 2024). Changing the solver complexity to Moderate or Complex will resolve most errors. However, the Simple solver is most appropriate for the models designed for this thesis, as they have little complexity.

Another possible solution is to make convergence criteria of the selected solver less strict (note that every solver has a linear and non-linear version, from which one is selected automatically). This can be accomplished by changing the values of Inner_dvclose and Outer_dvclose. Where, Outer_dvclose defines the dependent-variable (for example, head or concentration) change criterion for convergence of the outer (nonlinear) iterations, in units of the dependent-variable (for example, length for head or mass per length cubed for concentrations). When the maximum absolute value of the dependent-variable change at all nodes during an iteration is less than or equal to OUTER_DVCLOSE, iteration stops (Hydrogeologic, 2024). Inner_dvclose is similar to Outer_dvclose, but used by the linear solver instead. While increasing Inner_dvclose and Outer_dvclose does eventually resolve all errors, it requires increasing their values by three orders of magnitude (Table E4). This may result in premature convergence to very different hydraulic head values in specific cells, compared to a model run with stricter convergence criteria, and was therefore deemed a poor solution.

Table E4: Number of errors remaining after adjusting Outer_dvclose and Inner_dvclose, in MODFLOW simulations. Where, the errors refer to the MODFLOW simulation errors encountered when generating the results. And Outer_dvclose and Inner_dvclose are parameters set in MODFLOW6's iterative model solution (IMS) package, which is used to solve flow and/or transport simulations. Outer refers to the non-linear solver and Inner to the linear solver.

Outer_dvclose (m)	Inner_dvclose (m)	Number of Errors Remaining
0.001 (default)	0.001 (default)	185
0.01	0.01	75
0.1	0.1	18
1.0	1.0	0

Finally, the maximum allowed number of iterations was increased, allowing the numerical solver more computation time until convergence. Increasing the maximum iterations for the non-linear solver from 25 to 100 and for the linear solver from 50 to 100, decreased the number of errors from 185 to 1 (Table E4). Further increasing both parameters to allow 1000 iterations each, removes the remaining error. Increasing the maximum allowed number of iterations to 1000 for IMS has little to no noticeable influence on total run time, considering that there were only 185 errors in 135 thousand model calls. Therefore, this is the solution that was implemented to generate the main results.

Table E5: Number of errors remaining after adjusting the parameters: Outer_maximum and Inner_maximum, in MODFLOW simulations. Where the errors refer to the MODFLOW simulation errors encountered when generating the results. And Outer_maximum and Inner_maximum are parameters set in MODFLOW6's iterative model solution (IMS) package, which is used to solve flow and/or transport simulations. Outer refers to the non-linear solver and Inner to the linear solver.

Outer_maximum (iterations)	Inner_maximum (iterations)	Number of Errors Remaining
25 (default)	50 (default)	185
100	100	1
1000	1000	0