Water Resources Research

RESEARCH ARTICLE
10.1029/2017WR022489

This article is a companion to Qin et al. (2018), https://doi.org/10.1029/2017WR022488.

Key Points:
- Robust Gauss-Newton algorithm achieves similar robustness to evolutionary optimizer SCE and offers efficiency gains of orders of magnitude
- Robust Gauss-Newton algorithm is generally more efficient than the Levenberg-Marquardt and dynamically dimensioned search algorithms
- A median of 5 and 1 RGN invocations are required to find global and tolerable optima with 95% confidence, a tighter range than its competitors

Correspondence to:
D. Kavetski,
dmitri.kavetski@adelaide.edu.au

Citation:
Qin, Y., Kavetski, D., & Kuczera, G. (2018). A robust Gauss-Newton algorithm for the optimization of hydrological models: Benchmarking against industry-standard algorithms. *Water Resources Research*, *54*, 9637–9654. https://doi.org/10.1029/2017WR022489

Received 27 DEC 2017
Accepted 3 JUN 2018
Accepted article online 17 AUG 2018
Published online 30 NOV 2018

A Robust Gauss-Newton Algorithm for the Optimization of Hydrological Models: Benchmarking Against Industry-Standard Algorithms

Youwei Qin[1,2], Dmitri Kavetski[3], and George Kuczera[2]

[1]State Key Laboratory of Hydrology-Water Resources and Hydraulic Engineering, Centre for Global Change and Water Cycle, Hohai University, Nanjing, China, [2]School of Engineering, The University of Newcastle, Callaghan, NSW, Australia, [3]School of Civil, Environmental and Mining Engineering, University of Adelaide, Adelaide, SA, Australia

Abstract Optimization of model parameters is a ubiquitous task in hydrological and environmental modeling. Currently, the environmental modeling community tends to favor evolutionary techniques over classical Newton-type methods, in the light of the geometrically problematic features of objective functions, such as multiple optima and general nonsmoothness. The companion paper (Qin et al., 2018, https://doi.org/10.1029/2017WR022488) introduced the robust Gauss-Newton (RGN) algorithm, an enhanced version of the standard Gauss-Newton algorithm that employs several heuristics to enhance its explorative abilities and perform robustly even for problematic objective functions. This paper focuses on benchmarking the RGN algorithm against three optimization algorithms generally accepted as "best practice" in the hydrological community, namely, the Levenberg-Marquardt algorithm, the shuffled complex evolution (SCE) search (with 2 and 10 complexes), and the dynamically dimensioned search (DDS). The empirical case studies include four conceptual hydrological models and three catchments. Empirical results indicate that, on average, RGN is 2–3 times more efficient than SCE (2 complexes) by achieving comparable robustness at a lower cost, 7–9 times more efficient than SCE (10 complexes) by trading off some speed to more than compensate for a somewhat lower robustness, 5–7 times more efficient than Levenberg-Marquardt by achieving higher robustness at a moderate additional cost, and 12–26 times more efficient than DDS in terms of *robustness-per-fixed-cost*. A detailed analysis of performance in terms of reliability and cost is provided. Overall, the RGN algorithm is an attractive option for the calibration of hydrological models, and we recommend further investigation of its benefits for broader types of optimization problems.

1. Introduction

Optimization of model parameters is a key step in environmental model calibration. In many hydrological (rainfall-runoff) models, parameters are not measurable and are hence estimated through calibration against observed data using objective functions to quantify the model goodness of fit. In hydrological modeling, the response of interest is typically streamflow, using objective functions such as the sum of squared errors (SSE), Nash-Sutcliffe efficiency (NSE), and related functions (Doherty, 2005; Hill et al., 2015; Tolson & Shoemaker, 2007). Probabilistic modeling, for example, using Bayesian techniques, also leads to optimization problems when searching for the maximum of the posterior distribution (e.g., Kuczera et al., 2016). In many common instances, posterior distributions and likelihood functions take the form of SSE functions, further highlighting the connection between optimization and probabilistic analysis.

The performance of an optimization algorithm—namely, its reliability in finding the best-fit parameter set at a low computational cost—depends on the properties of the objective function. It is well-known that for models that are strongly nonlinear with respect to their parameters, SSE-type objective functions depart from a well-behaved quadratic shape and exhibit irregular features, most notably multiple optima (Bates & Watts, 2007). In hydrological contexts, model multioptimality has been vividly illustrated by Duan et al. (1992), Kavetski and Clark (2010), and many others. Ideally, the optimization algorithm should find the best—global—optimum; however, in practice this is difficult to achieve and may depend on the initial solution estimates (i.e., on the starting points of the search). Another degeneracy frequently encountered in environmental models is nonsmooth dependence of model responses on model parameters, which translates into nonsmoothness of the objective function surface (Kavetski & Kuczera, 2007). Other problematic features include high parameter correlations, flat (insensitive) regions in the objective function, and so forth.

In response to these numerical challenges, a wide range of optimization algorithms have been developed in the hydrological and broader mathematical/engineering literature. In general, optimization methods can be classified into multiple groups: local versus global depending on their ability to find the global optimum, smooth versus nonsmooth depending on their assumptions about the smoothness of the objective function surface, deterministic versus stochastic depending on whether the search is randomized, and so forth. The distinction between local versus global methods is of particular relevance to this work:

- Local methods, in particular traditional gradient-based algorithms such as Newton-type methods, which assume the objective function is continuous (smooth) and near quadratic. Newton-type methods include Gauss-Newton-type algorithms tailored to SSE objective functions (e.g., the Levenberg-Marquardt [LM] algorithm), quasi-Newton algorithms applicable to general objective functions, and others (Nocedal & Wright, 2006). Newton-type methods tend to converge fast, but typically only to the optimum "nearest" to the initial search point, making them ill-suited for problems with macroscale multioptimality. Moreover, due to their reliance on derivatives, Newton-type methods often behave erratically in the presence of microscale roughness;
- Global methods, such as the shuffled complex evolution (SCE) algorithm (Duan et al., 1992), the dynamically dimensioned search (DDS) algorithm (Tolson & Shoemaker, 2007), AMALGAM (Vrugt et al., 2009), and others. These methods are designed to improve the chance of convergence to the global optimum from a broader range of initial points, and to be less susceptible to microscale roughness. However, global optimization tends to be computationally costly in terms of the number of objective function calls, and susceptible to the curse of dimensionality (Tolson & Shoemaker, 2007).

In hydrological modeling, current perceptions favor evolutionary optimization methods, on the basis of their general robustness and better global convergence properties. Evolutionary optimization methods are used in the studies of Duan et al. (1992), Tolson and Shoemaker (2007), Vrugt et al. (2009), Arsenault et al. (2014), and others. Nonetheless, Newton-type methods continue to be used in studies including Doherty (2005), Kavetski et al. (2007), Qin et al. (2017) and others, with their speed being of particular importance in computationally challenging problems (Hill et al., 2015).

Interest in revisiting Newton-type methods arises from the conclusions of several recent studies. First, Kavetski et al. (2018) have argued that the ultimate characteristic of interest in practical applications is not robustness on its own but rather efficiency defined as the *ability to find desired optima at a low cost*. They showed that in 8 of 12 modeling scenarios the LM algorithm was more efficient than the SCE algorithm because its much cheaper cost per invocation allowed it to match SCE's robustness through multiple invocations. Achieving global convergence through multiple invocations is the essence of the "multistart" approach (Kavetski et al., 2007; Skahill & Doherty, 2006; Tolson & Shoemaker, 2007) but in practice is necessarily constrained by the computational cost of a single algorithm invocation. Second, there are opportunities for improving the performance of Newton-type methods. Some of these avenues are model-intrusive, for example, smoothing the model and/or objective function (Kavetski & Kuczera, 2007) or, somewhat analogously, using smooth emulators (Koziel & Leifsson, 2013). Other avenues include using q-gradient techniques that seek to inform the Newton search direction by the objective function shape at the larger scale to avoid being misled by local irregularities (Gouvêa et al., 2016). Motivated by these ideas, Qin et al. (2018) introduced the robust Gauss-Newton (RGN) algorithm—a modification to the Gauss-Newton (GN) algorithm that dramatically improves its robustness characteristics. The aspiration of the RGN approach is to achieve robustness comparable to evolutionary methods at a fraction of the computational cost, hence yielding higher overall efficiency. Empirical tests confirm that the RGN algorithm vastly improves on the standard GN algorithm (Qin et al., 2018). The logical question taken up in this study is then—how does RGN compare against established "best-practice" evolutionary and Newton-type algorithms?

2. Aims

The aim of this paper is to compare the RGN algorithm against a selection of optimization methods generally considered as best practice in the hydrological community, over a range of representative hydrological models and catchments. More specifically, our aims are to

1. benchmark the RGN algorithm against SCE, LM, and DDS over four hydrological models and three catchments, using reliability, cost, and efficiency metrics of algorithm performance and

2. compare the convergence characteristics of RGN versus SCE, focusing on the rate of progress of the algorithms and on the convergence patterns.

The remainder of this paper is organized as follows. Section 3 reviews the theory of least squares model optimization and provides key background on the RGN algorithm and the benchmarking algorithms. Section 4 describes the case study setup and performance metrics. Section 5 reports the case study results, followed by section 6, which discusses and interprets the empirical findings using theoretical insights. Section 7 summarizes the main conclusions and outlines directions for further work.

3. Theory and Algorithms

3.1. Least Squares Calibration Problem

The classical form of the model calibration problem is given by the optimization of an objective function Φ with respect to the parameters $\boldsymbol{\theta} = \{\theta_k, k = 1, \cdots, N_{\boldsymbol{\theta}}\}$ of a hydrological model $\mathscr{H}(\boldsymbol{\theta}; \mathbf{x})$,

$$\boldsymbol{\theta}^{(\mathrm{opt})} = \arg\min_{\boldsymbol{\theta}} \Phi(\boldsymbol{\theta}), \qquad \boldsymbol{\theta}^{(\mathrm{L})} \le \boldsymbol{\theta} \le \boldsymbol{\theta}^{(\mathrm{H})} \tag{1}$$

where $\boldsymbol{\theta}^{(\mathrm{opt})}$ is the optimal parameter set, $\boldsymbol{\theta}^{(\mathrm{L})}$ and $\boldsymbol{\theta}^{(\mathrm{H})}$ are the lower and upper bounds, and $\mathbf{x}$ represents other model inputs (such as time series of precipitation, potential evapotranspiration, and others).

The objective function serves as a measure of mismatch between the model predictions of a quantity of interest (here streamflow time series $\mathbf{y}$ given inputs $\mathbf{x}$) and observed data $\tilde{\mathbf{y}}$. The model-data mismatch is typically quantified using functions derived from SSE criteria,

$$\Phi_{\mathrm{SSE}}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{r}(\boldsymbol{\theta})^T\mathbf{r}(\boldsymbol{\theta}) = \frac{1}{2}\sum r_t^2(\boldsymbol{\theta}) \tag{2}$$

$$\mathbf{r}(\boldsymbol{\theta}) := \mathbf{r}[\boldsymbol{\theta}; \tilde{\mathbf{y}}, \mathbf{x}] = \tilde{\mathbf{y}} - \mathscr{H}(\boldsymbol{\theta}; \mathbf{x}) \tag{3}$$

where $\tilde{\mathbf{y}} = \{\tilde{y}_t, t = 1, \cdots, N_{\tilde{\mathbf{y}}}\}$ is the vector of observed responses, $\mathbf{r}$ is the vector of model residuals, and T denotes the transpose operation. To avoid clutter, we omit the dependence of $\mathbf{r}(\boldsymbol{\theta})$ and $\Phi(\boldsymbol{\theta})$ on $\mathbf{x}$ and $\tilde{\mathbf{y}}$.

SSE-type objective functions are ubiquitous in hydrology and general modeling. In addition to their intuitive appeal, the SSE formulation can be derived from the general statistical technique of least squares inference (e.g., Bates & Watts, 2007; Kavetski, 2018). While in its simplest form the SSE function assumes the residuals follow a Gaussian distribution, the SSE formulation can be readily extended to allow for heteroscedasticity and skew of residual errors (e.g., Bates & Campbell, 2001; Kuczera, 1983; McInerney et al., 2017).

Alternative objective functions that fall outside the SSE family include, for example, metrics based on sums of absolute (rather than squared) errors (e.g., Zhao, 1992). The optimization of non-SSE objective functions is beyond the immediate scope of this study but is of interest from a more general perspective. We comment on the applicability of the contributions of this work to the optimization of general objective functions towards the end of the presentation.

3.2. Gauss-Newton and Related Methods

The Gauss-Newton algorithm and related techniques are based on the solution of the normal equations,

$$\mathbf{J}^T\mathbf{J}\,\boldsymbol{\delta} = -\mathbf{J}^T\mathbf{r} \tag{4}$$

where $\mathbf{J} = \partial\mathbf{r}/\partial\boldsymbol{\theta}$, defined as $\{J_{t,k} = \partial r_t/\partial\theta_k; t = 1, \cdots, N_{\tilde{\mathbf{y}}}; k = 1, \cdots, N_{\boldsymbol{\theta}}\}$, is the Jacobian matrix of the function $\mathbf{r}(\boldsymbol{\theta})$ defining the residual errors in equation (3). In most cases, the hydrological model equations do not lend themselves to analytical differentiation, and the model Jacobian is estimated using finite difference approximations.

Equation (4) represents a special case of the more general Newton equations, tailored to take advantage of the structure of the SSE function in equation (2) (Nocedal & Wright, 2006). It forms the basis for two important types of algorithms:

1. conventional Gauss-Newton algorithms, which use a linesearch along the Gauss-Newton direction $\boldsymbol{\delta}$ to safeguard the solution of equation (4), and
2. the LM algorithm, which uses trust region techniques to constrain and reorient the search direction within a region where the quadratic approximation assumed by equation (4) appears to be accurate.

The RGN is an enhanced version of the conventional Gauss-Newton algorithm and is outlined in section 3.3. The LM algorithm, used as one of the benchmarking algorithms, is described in section 3.4.1.

3.3. RGN Algorithm

The RGN algorithm is obtained by enhancing the standard GN algorithm with a set of three heuristic schemes to boost its exploratory capabilities and improve its behavior in the presence of multiple-optima, microscale noise and flat regions in the objective function. The next sections outline these heuristics.

3.3.1. Large Sampling Scale to Capture Large-Scale Features of Objective Function

The *large sampling scale* (LSS) heuristic allows the RGN algorithm to "see" as much of the search region as necessary to continue making progress and contributes the lion's share of robustness improvements achieved by the RGN algorithm (Qin et al., 2018). Recall that the region over which the Gauss-Newton approximation is constructed is controlled by the magnitude of the finite difference perturbations used when approximating the derivatives $\mathbf{J}$ in equation (4). These perturbations essentially "sample" the objective function; we refer to their magnitude as the *sampling scale*. Traditional GN algorithms use *small* sampling scales $\mathbf{h}$ in order to match the (local) analytical gradient as much as practical. In these applications, sampling scales, expressed as a fraction of current solution values, range from around 10^{-8} in Press et al. (2007) and the IMSL library (Rogue Wave Software, 2017), to around 10^{-2} (or slightly larger) in the PEST software tool (Doherty, 2005). The RGN algorithm takes an opposite perspective and makes the sampling scale as *large* as possible as long as progress is being made. Here progress is gauged by the success of the linesearch algorithm in locating a solution satisfying the sufficient decrease conditions. If a new point is found that satisfies the sufficient decrease condition, the sampling scale is increased for the next iteration; otherwise, it is decreased to redo the current iteration. Qin et al. (2018) established that artificial restrictions on the sampling scale tend to degrade RGN's global convergence; hence, we initialize the sampling scale to its largest value possible for a central difference scheme, namely, to half the width of the entire feasible solution region.

3.3.2. Best-Sampling Point Scheme to Take Advantage of Free Information

The *best-sampling point* (BSP) scheme keeps track of the best objective function value sampled during the difference approximation. If this point improves on the objective function value at the end of the linesearch, the RGN algorithm discards the linesearch results and adopts the BSP point as the starting point for its next iteration. The BSP scheme provides a free opportunity to accelerate progress using already-available information; empirical results suggest it is most beneficial for detecting new and better regions of attraction, hence improving robustness beyond what is achieved by the LSS scheme on its own (Qin et al., 2018).

3.3.3. Null-Space Jump Scheme to Escape Near-Flat Regions

The normal equations are typically solved using either the QR algorithm (e.g., Nocedal & Wright, 2006) or the singular value decomposition (SVD) algorithm (e.g., Doherty, 2005). The SVD algorithm is particularly robust —but also slower—because it explicitly constructs the set of singular values of the matrix $\mathbf{J}^T\mathbf{J}$. The set of singular values provides a mechanism for obtaining an approximate solution to near-singular systems, which in the context of optimization arise when the algorithm is in a "flat" region of the objective function. The truncation of singular values reduces the displacement in the null (insensitive) space and is frequently used to reduce noise in the solution of linear least squares problems (Press et al., 2007). The *null-space jump* (NSJ) scheme consists of tightening the truncation threshold, to make truncation less likely—and hence allow the algorithm to make large "noisy" jumps from the current location even in directions that are close to insensitive. Although somewhat contrary to standard SVD usage, which seeks to prevent noisy solutions (Doherty, 2005), the NSJ scheme was empirically found to help the RGN algorithm escape from "flat" regions (Qin et al., 2018). Note that the parameter bounds and linesearch sufficient decrease conditions act as safeguards to prevent this approach from producing meaningless results. The NSJ scheme tends to be activated only when optimizing hydrological models with highly correlated parameters (e.g., FUSE-536 with 14 parameters, see case study section), but in such modeling scenarios, its inclusion can improve optimization efficiency by as much as a factor of 4 (Qin et al., 2018).

3.3.4. Other Details

The heuristics described in sections 3.3.1–3.3.3 represent the core innovations behind the RGN algorithm. We refer to Qin et al. (2018) for a detailed description and empirical evaluation of the contributions of each individual heuristic. In addition, the RGN algorithm makes use of an active set method to handle the box constraints (parameter bounds), where the search direction is projected along the search boundaries. The termination criteria used in RGN are listed in Appendix A and closely follow the approach used in PEST (Doherty, 2005).

3.4. Benchmarking Algorithms

This section describes the industry-standard optimization methods used for benchmarking the RGN algorithm.

3.4.1. LM Algorithm

The LM algorithm is a common approach for least-squares calibration (Nocedal & Wright, 2006) and is widely used in environmental modeling applications. An industry-standard implementation of the LM algorithm is provided by the software package PEST, which offers many options and enhancements (Doherty, 2005). PEST is widely used in the calibration of environmental models, notably groundwater models (Doherty, 2003).

In this study the standard LM algorithm is used, with the model Jacobian estimated using central differences with sampling scales set to default PEST values (2% of current parameter values with a minimum of 0.01). Termination criteria listed in Appendix A are used (same as in the RGN algorithm).

Note that the trajectory repulsion approach (TRA; Skahill & Doherty, 2006), one of several PEST algorithmic options, is not used in this study. First, TRA is intended as an improvement of the multistart technique rather than of the GN algorithm. Second, the use of TRA could confound our convergence study because TRA seeks to reduce the probability of repeatedly finding the same optimum—whereas we specifically quantify robustness as repeated convergence to the desired optimum. The use of TRA within a multistart application of RGN will be investigated in a separate study.

3.4.2. SCE Algorithm

The SCE algorithm is a stochastic evolutionary optimization algorithm widely accepted as "best practice" in the hydrological community (Behrangi et al., 2008; Duan et al., 1992; Qin et al., 2017; Tolson & Shoemaker, 2007; Tolson & Shoemaker, 2008), due to its general robustness in locating the global optimum even for highly irregular objective functions.

The SCE algorithm is controlled by a number of settings, most notably the number of complexes. A large number of complexes ensures a thorough but more expensive exploration of the search space, whereas a low number of complexes sacrifices some robustness in return for reduced cost. The original recommendation was 2–25 depending on the number of parameters and degree of difficulty of the problem (Duan et al., 1993; Sorooshian et al., 1993). Subsequent work has suggested a lower number of complexes, as low as 2, to accelerate convergence especially in the early stages (Behrangi et al., 2008). To ensure robust conclusions, we consider two SCE configurations at the "book-ends" of its robustness-cost trade-off: SCE-nc10 uses 10 complexes, and SCE-nc2 uses 2 complexes.

Our SCE implementation is terminated if any of the following criteria are met: (a) scaled changes in the "best-so-far" objective function value, $|\Phi_{\text{best}}^{(i-1)} - \Phi_{\text{best}}^{(i)}| / \max\left(|\Phi_{\text{best}}^{(i)}|, \Phi^{(\text{scale})}\right)$, are below tolerance $\tau_\Phi = 10^{-5}$ over $N_{\Delta\Phi\approx 0} = 3$ consecutive iterations (complex shuffles), where $\Phi^{(\text{scale})}$ is the typical scale of the objective function (here $\Phi^{(\text{scale})} = 1.0$); or (b) total number of objective function evaluations, $N_\Phi^{(\max)}$, exceeds 10^6.

3.4.3. Dynamically Dimensioned Search

The DDS algorithm is another global optimizer widely used in hydrology (Arsenault et al., 2014; Tolson & Shoemaker, 2007; Yen et al., 2015), especially in the calibration of computationally expensive models. DDS employs a search strategy based on a random Gaussian walk over a gradually reducing subspace of the feasible search region.

Unlike the other optimization algorithms used in this study, DDS is not used with a convergence criterion and instead operates on a prespecified computational budget. In this work, the DDS budget is set to 800 objective function evaluations for the HYMOD, SIXPAR, and SIMHYD hydrological models, and to 2,500 evaluations for the FUSE model. These budgets are set empirically, in order to approximately match the cost of RGN

Table 1
Hydrological Models Used in the Case Study and Typical Characteristics[a] of Their Objective Function Surfaces

Model	Typical characteristics of objective function surfaces	References
HYMOD	Multiple optima; smooth surface with continuous derivatives; near-Gaussian shape in the vicinity of optima	Boyle (2001); Qin et al. (2017)
SIXPAR	Multiple optima; smooth surface with minor discontinuities in the derivatives; non-Gaussian shape even close to an optimum, including long curved ridges and localized insensitive regions	Gupta and Sorooshian (1985); Duan et al. (1992); Qin (2017)
SIMHYD	Multiple optima; smooth surface with minor discontinuities in the derivatives; non-Gaussian shape even close to an optimum, including long curved ridges. Some insensitive regions are localized, other insensitive regions may span large portions of the feasible space (i.e., some model parameters appear largely insensitive)	Chiew and Siriwardena (2005); Qin (2017); Qin et al. (2017)
FUSE-536[b] (FUSE)	Multiple optima; highly nonsmooth surface with sharply discontinuous derivatives and staircase structure; strongly non-Gaussian irregular shape even close to an optimum; higher-dimensional than the other models in this study, with some evidence of insensitive parameters	Kavetski and Clark (2010); Qin (2017); Qin et al. (2017)

[a]The objective function characteristics are related to the mathematical structure and constitutive functions within the model equations. The model behavior will depend, to some extent, on the forcing data in the catchment of application. [b]Here FUSE-536 is deliberately set to use the explicit Euler time stepping scheme in order to create a challenging objective function surface for the purposes of this study (Qin et al., 2018). The hydrological models HYMOD, SIXPAR, and SIMHYD also use explicit Euler or similar time stepping techniques for at least some of their fluxes.

optimization (see section 5.1) and thus enable a comparison of "robustness per given cost." The other DDS setting, its scaling perturbation expressed as a fraction of the search space, is set to 0.2 (Tolson & Shoemaker, 2007).

4. Empirical Study Material and Methods

Two experiments are used to benchmark the RGN algorithm. The selection of hydrological models and catchments is the same as in the companion paper (Qin et al., 2018) and is succinctly reviewed in section 4.1. Section 4.2 describes the design of each experiment, followed by section 4.3, which details the evaluation metrics and criteria.

4.1. Hydrological Models and Data

Four lumped daily-step conceptual hydrological models are used, namely, the five-parameter HYMOD model (Boyle, 2001), the six-parameter SIXPAR model (Duan et al., 1992), the seven-parameter SIMHYD model (Chiew & Siriwardena, 2005), and the 14-parameter FUSE-536 model (Clark & Kavetski, 2010). The characteristics of these models are summarized in Table 1. As noted in Qin et al. (2018), the four models are computationally fast, which allows extensive in-depth investigation of optimization algorithm performance, and present a range of parameter dimensionality (from 5 to 14 parameters) and geometric complexity of the objective function (from smooth to very rough).

The four hydrological models are applied in three Australian catchments, namely, Tambo River, Bass River, and Coopers Creek. These catchments have runoff coefficients of 0.11, 0.30 and 0.46, respectively; their hydroclimatology is summarized in Table 2 of the companion paper. The data used for model calibration include observed rainfall, estimated PET and observed daily runoff; temperature data are not used because the catchments do not experience snowfall. The calibration periods are listed in Table 2 of the companion paper.

4.2. Experimental Setup

4.2.1. Experiment 1: Benchmark RGN Against SCE, LM, and DDS

The purpose of this experiment is to benchmark the RGN algorithm against the best-practice optimization algorithms listed in section 3.4. All optimization algorithms under consideration are invoked within a multistart framework. The multistart framework represents a stochastic optimization strategy in its own right; in this study, it is employed to estimate average performance characteristics including reliability, cost, and efficiency ratios (section 4.3). For RGN, LM, and DDS, a single "start," that is, a single algorithm invocation, corresponds to launching the algorithm from a single initial point, whereas for SCE, it involves seeding an entire

initial population of complexes. To provide a consistent algorithm comparison, the *i*th invocation of all algorithms uses the same (randomly selected) initial search point; for the *i*th invocation of SCE, this is achieved replacing one of the search points in the initial population. For the optimization of the HYMOD, SIMHYD, and SIXPAR models, we use M = 10,000 invocations. For the optimization of FUSE we use M = 1,000 invocations because of the higher computational cost of this hydrological model.

4.2.2. Experiment 2: Analysis of Individual Search Trajectories

The purpose of this experiment is to investigate the convergence behavior of individual search trajectories of RGN and SCE. We consider the pattern and rate of optimization progress in individual algorithm invocations, and the variability across multiple invocations, in order to gain insights into algorithm robustness and cost characteristics. In addition, we check for cases (if any) of termination criteria artificially inflating computational costs. An example of undesirable behavior is when an algorithm makes quick initial progress and finds a "good" solution, but then spends many iterations refining it to meet unduly stringent termination criteria. This behavior is not only costly but also undesirable because it confounds the benchmarking of the algorithms and practical recommendations thereof.

The analysis is carried out for multiple invocations of the RGN and SCE algorithms. For each invocation, we plot the trace of the current "best" objective function value after each iteration versus the corresponding number of function calls. Note that the number of function calls tracked includes the cost of the initial point or, in the case of SCE, the initial population of points. By overlaying the traces corresponding to 100 invocations of each algorithm, we can inspect convergence patterns and detect occurrences (if any) of the undesirable behavior described above.

4.3. Performance Attributes and Metrics

Optimization algorithm performance is summarized by its robustness, computational cost, and efficiency. Here, robustness is defined as the ability to reliably find a desired optimum (global or tolerable, see below) irrespective of the initial search point and consistently across multiple modeling scenarios (optimization problems). Efficiency is defined as the ability to find the desired optimum at low computational cost (Kavetski et al., 2018; Qin et al., 2018).

4.3.1. Reliability and Cost

"Global reliability," or G-reliability, is used to quantify the ability of the algorithm to find the global optimum,

$$\mathscr{R}_G = \mathscr{R}(\boldsymbol{\Phi}_{\mathrm{NSE}}; \breve{\Phi}_{\mathrm{NSE}}, 1\%) \tag{5}$$

where $\mathscr{R}(\boldsymbol{\psi}; \breve{\psi}, \tau_\psi)$ is a general reliability metric, defined as the fraction of M algorithm invocations, with final solution estimates $\boldsymbol{\psi} = \{\psi_m; m = 1, \cdots, M\}$, that find a solution within a tolerance τ_ψ of the "best-known" hydrological model performance metric $\breve{\psi}$. Note that the best-known metric value refers to the best metric value obtained in a given modeling scenario (combination of catchment and hydrological model) using all optimization algorithms considered in this study. The hydrological model performance is gauged in terms of the NSE (Nash & Sutcliffe, 1970) rather than the SSE objective function itself.

"Tolerable reliability," or T-reliability, is used to quantify the ability of the algorithm to find a solution that is at worst "tolerable" or "acceptable" (Arsenault et al., 2014; Qin et al., 2018; Tolson & Shoemaker, 2007),

$$\mathscr{R}_T = \mathscr{R}(\boldsymbol{\Phi}_{\mathrm{NSE}}; \breve{\Phi}_{\mathrm{NSE}}, 10\%) \tag{6}$$

The computational cost is reported in terms of the average number of objective function calls across multiple algorithm invocations (irrespective of whether the particular invocation found an optimum of interest).

4.3.2. Efficiency Ratios

The number of algorithm invocations required to find a desired optimum with confidence level α is estimated as (Kavetski et al., 2018)

$$M_{X,\alpha} = \mathrm{roundUp}\left[\frac{\log(1-\alpha)}{\log(1-\mathscr{R}_X)}\right] \tag{7}$$

where the function roundUp[z] denotes the smallest integer greater than or equal to z, so that $M_{X,\alpha} \in \{1, 2, 3, \ldots\}$. Equation (7) is safeguarded by constraining $\mathscr{R}_X \in [0 + \epsilon_{\mathscr{R}}, 1 - \epsilon_{\mathscr{R}}]$ where $\epsilon_{\mathscr{R}} = 1/(M+1)$ is set according to

the number of invocations used in the performance assessment; values of $\mathscr{R}_X$ outside this range are truncated accordingly. The subscripts X are used to refer to quantities related to global and tolerable optima, $X = G$ and T, respectively.

We report the median value and interquartile range (IQR) of $M_{X,\alpha}$ for each algorithm. These characteristics are relevant to the modeler's specification of the number of invocations M to be carried out in a new application (Kavetski et al., 2018). An algorithm with low variability in $M_{X,\alpha}$ is said to have "high consistency": The modeler can be more certain in setting the value of M and achieving the confidence level α in finding the desired optimum.

An algorithm that has both high reliability and high consistency is referred to as *robust*—it is able to consistently find desired optima of many different problems (Kavetski et al., 2018). Note that robustness on its own is desirable but insufficient—it says nothing about the computational cost of employing the algorithm. Typically, there is a trade-off between robustness and cost, and these characteristics must be considered jointly when benchmarking optimization algorithms. This trade-off is quantified using the efficiency ratio metric (Kavetski et al., 2018) described next.

The efficiency ratio for any two optimization algorithms, say A and B, for a given confidence level α is

$$\kappa_{X,\alpha}^{(\mathrm{A}/\mathrm{B})} = \frac{C_{X,\alpha}^{(\mathrm{B})}}{C_{X,\alpha}^{(\mathrm{A})}} = \frac{M_{X,\alpha}^{(\mathrm{B})} \times N_\Phi^{(\mathrm{B})}}{M_{X,\alpha}^{(\mathrm{A})} \times N_\Phi^{(\mathrm{A})}} \tag{8}$$

where $C_{X,\alpha} = M_{X,\alpha} \times N_\Phi$ is the total cost (in terms of the total number of objective function calls) of finding the desired optimum with confidence level α, and N_Φ is the average number of function calls per invocation.

The quantity $\kappa_{X,\alpha}^{(\mathrm{A}/\mathrm{B})}$ represents the "efficiency ratio" of Algorithms A versus B: Algorithm A "typically" requires $\kappa_X^{(\mathrm{A}/\mathrm{B})}$ times fewer objective function calls than Algorithm B to achieve confidence level α.

In this study, we set $\alpha = 95\%$ and report $M_{G,95\%}$ and $M_{T,95\%}$ for every optimization algorithm. The G- and T-efficiency ratios, $\kappa_{G,95\%}^{(\mathrm{RGN}/\cdot)}$ and $\kappa_{T,95\%}^{(\mathrm{RGN}/\cdot)}$, respectively, are computed by comparing RGN to every benchmarking algorithm. To reduce clutter, we use the shorthand notation $M_X = M_{X,95\%}$ and $\kappa_X = \kappa_{X,95\%}$.

5. Results

5.1. Experiment 1: Comparison of RGN Against SCE, LM, and DDS

This section reports the results of benchmarking the RGN algorithm against the SCE, LM, and DDS algorithms. Figure 1 presents bar graphs of the G- and T-reliabilities and average costs of the algorithms for the 12 catchment/model scenarios. In what follows, we define *high reliability* as being >95%, an algorithm as being *competitive* if its reliability is within 10% of its competitor, and two algorithms as being *similarly-poor* if their reliabilities are below 20% and within 10% of each other. Figure 2 presents the estimated number of invocations required by each algorithm to find either the global or tolerable optimum with 95% confidence. Figure 3 reports the estimated G- and T-efficiency ratios of RGN relative to the benchmarking algorithms.

5.1.1. Comparison of RGN Against SCE-nc10

Figure 1 shows that, overall, RGN approaches the robustness of SCE-nc10, with occasionally worse G-reliability and similar T-reliability, but at a significantly lower cost. More specifically,

- G-reliability: RGN is competitive with SCE-nc10 in 3 of 12 scenarios, uncompetitive in 5 scenarios, better than SCE-nc10 in 1 scenario, and similarly-poor in 3 scenarios. RGN and SCE-nc10 both achieve a high reliability in 2 HYMOD and 1 SIMHYD scenarios. Out of the 5 scenarios where RGN falls short of SCE, it maintains a good reliability in 4 scenarios—for example, $\mathscr{R}_G^{(\mathrm{RGN})} = 83\%$ compares well to $\mathscr{R}_G^{(\mathrm{SCE\text{-}nc10})} \approx 100\%$ in SIXPAR Coopers. In the HYMOD Tambo scenario, RGN achieves $\mathscr{R}_G^{(\mathrm{RGN})} = 43\%$, whereas SCE-nc10 fails with $\mathscr{R}_G^{(\mathrm{SCE\text{-}nc10})} = 0.03\%$. Finally, RGN and SCE-nc10 both struggle to locate the global optimum in all 3 FUSE scenarios, with the best reliability being 12.8%.
- T-reliability: RGN and SCE-nc10 achieve a high reliability in 7 scenarios, including all the HYMOD and SIMHYD scenarios. In the remaining 5 scenarios, which comprise the FUSE and SIXPAR scenarios, SCE-nc10 maintains a high T-reliability, $\mathscr{R}_T^{(\mathrm{SCE\text{-}nc10})} \geq 99\%$, while RGN performs moderately to slightly worse, with $\mathscr{R}_T^{(\mathrm{RGN})}$ of 65–92%.

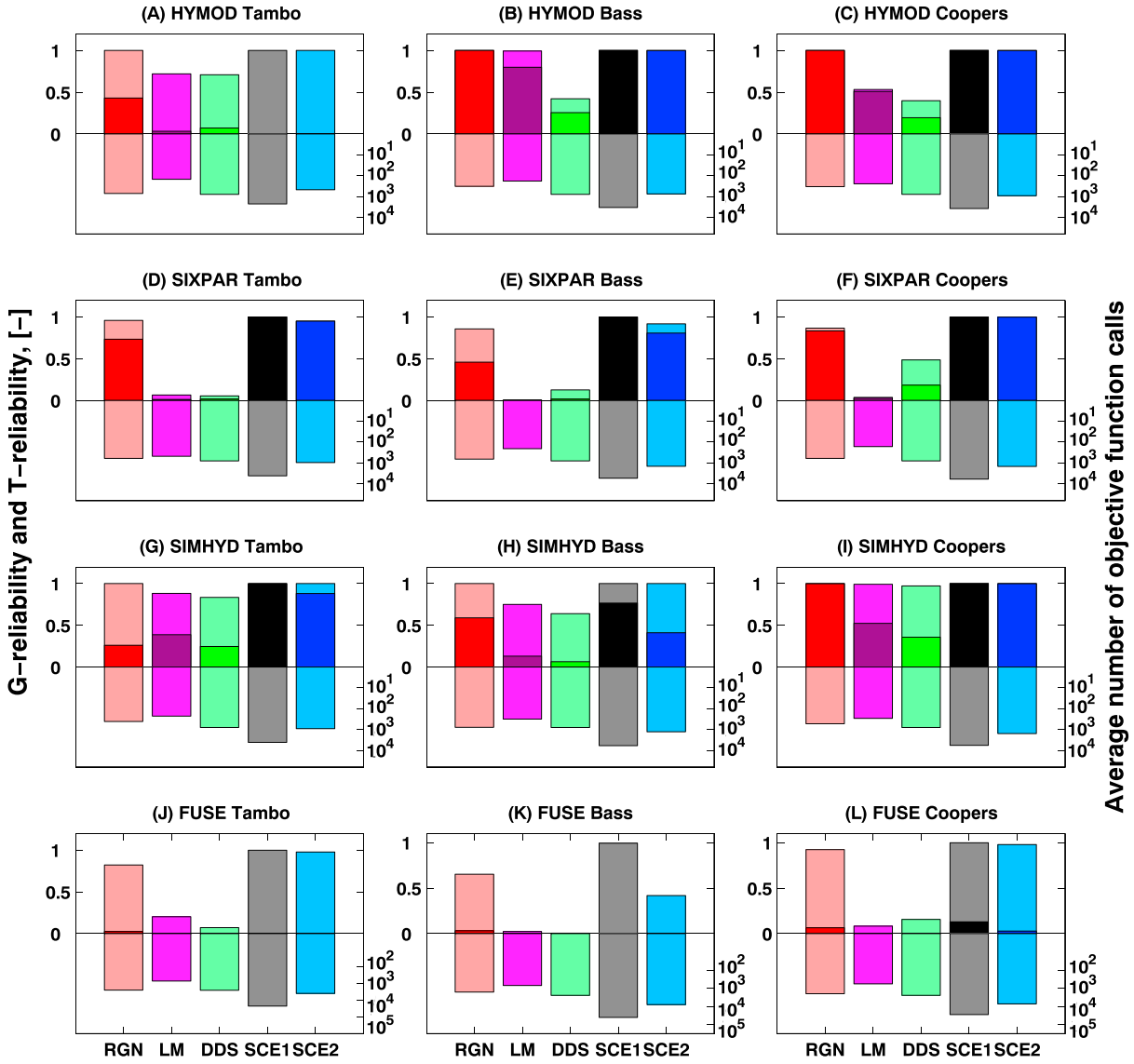

Figure 1. Reliability and cost of the RGN algorithm benchmarked against the LM, SCE-nc10 (labeled SCE1), SCE-nc2 (labeled SCE2), and DDS algorithms. G-reliability $\mathscr{R}_G$ (chance of finding a solution within 1% of the best-known NSE), T-reliability $\mathscr{R}_T$ (chance of finding a solution within 10% of the best-known NSE), and computational cost N_Φ (average number of objective function calls within a single algorithm invocation) are reported. The HYMOD, SIXPAR, and SIMHYD scenarios use 10,000 invocations, and the FUSE scenarios use 1,000 invocations. Algorithm abbreviations: DDS = dynamically dimensioned search; LM = Levenberg-Marquardt; NSE = Nash-Sutcliffe efficiency; RGN = robust Gauss-Newton; SCE = shuffled complex evolution.

- Computational cost: RGN invocations are substantially cheaper than SCE-nc10. In the HYMOD, SIXPAR, and SIMHYD scenarios, the number of objective function calls per invocation, N_Φ, required by RGN is in the range of 330–790, whereas for SCE-nc10, it is in the (approximate) range of 2,300–6,000. In the FUSE scenarios, $N_\Phi^{(\mathrm{RGN})} \approx 1{,}650 - 2{,}500$ and $N_\Phi^{(\mathrm{SCE\text{-}nc10})} \approx 22{,}000 - 40{,}500$. In relative terms, RGN is cheaper by factors of 3 to 11 in the HYMOD, SIXPAR, and SIMHYD scenarios, and by factors of 9 to 24 in the FUSE scenarios.

Figure 2 compares the estimated number of invocations M_X required by RGN and SCE-nc10 to find global or tolerable optima with 95% confidence:

- Global optimum: Over all scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 120 with a median of 4.5 and an IQR of 36.5, while $M_G^{(\mathrm{SCE\text{-}nc10})}$ ranges from 1 to 9,985 with a median of 1 and an IQR of 2,253 (due to SCE having low G-reliability in 4 scenarios). Excluding the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 10 with a median of 3 and an IQR of 4.5, and $M_G^{(\mathrm{SCE\text{-}nc10})}$ ranges from 1 to 9,985 with a median of 1 and an IQR of 1.0. The FUSE

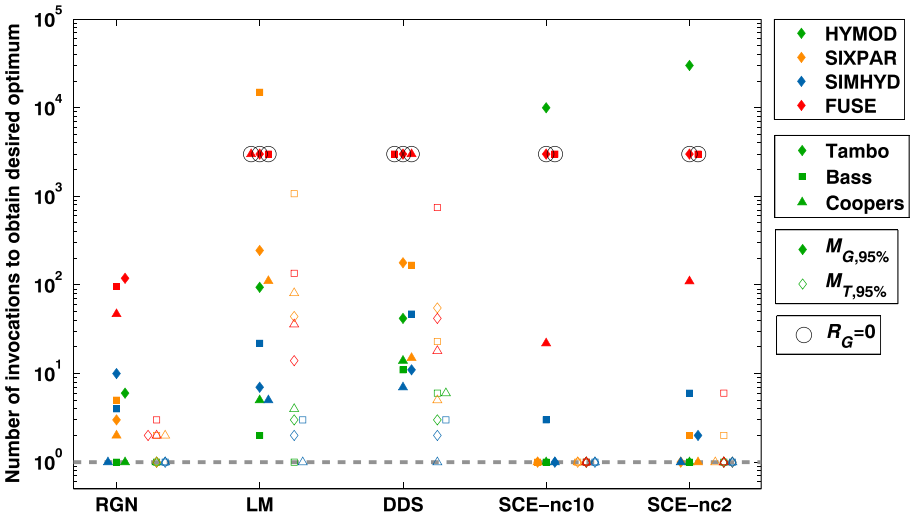

Figure 2. Number of invocations required by the optimization algorithms to find a desired (global or tolerable) optimum with 95% confidence, $M_{X,95\%}$. Results are distinguished for different hydrological models (symbol color) and catchments (symbol shape). Scenarios with zero G-reliability are flagged by black circles. The points have been jittered slightly to avoid overlaps. Algorithm abbreviations: DDS = dynamically dimensioned search; LM = Levenberg-Marquardt; RGN = robust Gauss-Newton; SCE = shuffled complex evolution.

scenarios tend to require appreciably more invocations: $M_G^{(\mathrm{RGN})}$ ranges from 47 to 120, and $M_G^{(\mathrm{SCE\text{-}nc10})}$ ranges from 22 to 3,000.

- Tolerable optimum: $M_T^{(\mathrm{RGN})}$ ranges from 1 to 3 with a median of 1 and an IQR of 1, while $M_T^{(\mathrm{SCE\text{-}nc10})} = 1$ in all 12 scenarios.

Figure 3 compares the efficiency of RGN relative to SCE-nc10:

- G-efficiency: RGN matches or outperforms SCE-nc10 in all 12 scenarios, with G-efficiency ratios $\kappa_G^{(\mathrm{RGN/SCE\text{-}nc10})}$ ranging from 1 to 5,300 with a median of 8.6.

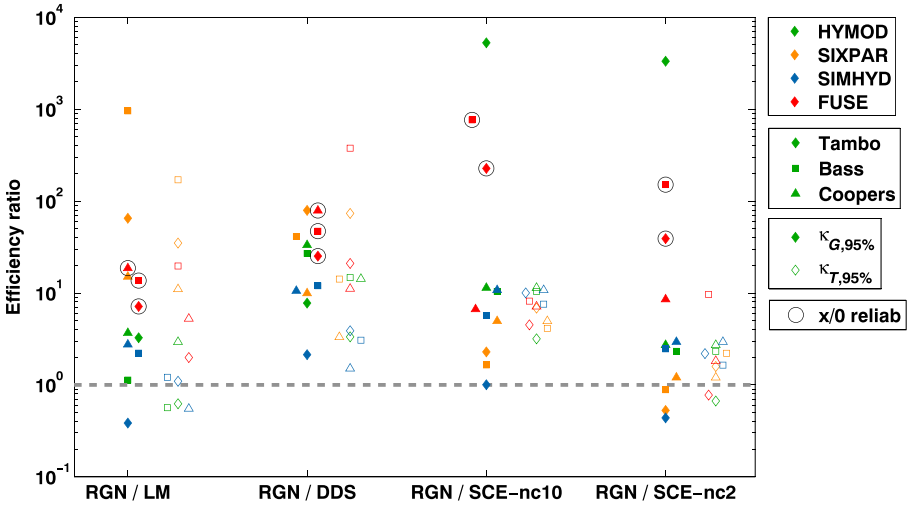

Figure 3. Efficiency ratios of the RGN algorithm with respect to the benchmarking algorithms, $\kappa_{X,95\%}^{(\mathrm{RGN}/\cdot)}$. G- and T-efficiency ratios are displayed (filled and empty symbols) for a confidence level of 95% in finding the desired optimum. Results are distinguished for different hydrological models (symbol color) and catchments (symbol shape). Scenarios where one or both optimization algorithms have zero reliability are flagged with black circles. The points have been jittered slightly to avoid overlaps. Algorithm abbreviations: DDS = dynamically dimensioned search; LM = Levenberg-Marquardt; RGN = robust Gauss-Newton; SCE = shuffled complex evolution.

- T-efficiency: RGN substantially outperforms SCE-nc10 in all 12 scenarios, with T-efficiency ratios $\kappa_T^{(\mathrm{RGN/SCE\text{-}nc10})}$ ranging from 3.2 to 11.4 with a median of 7.4.

5.1.2. Comparison of RGN against SCE-nc2

We now compare RGN against SCE-nc2, which was previously recommended as a more efficient SCE configuration (Section 3.4.2). Figure 1 shows that, overall, RGN has comparable, if not better, robustness than SCE-nc2 at a slightly lower cost. We also see that SCE-nc10 is more robust but costlier than SCE-nc2. More specifically,

- G-reliability: RGN is competitive with SCE-nc2 in achieving high G-reliability in 3 of 12 scenarios, uncompetitive in 4 scenarios, better than SCE-nc2 in 2 scenarios, and similarly-poor in all 3 FUSE scenarios. RGN and SCE-nc2 achieve a high $\mathscr{R}_G$ in 2 HYMOD and 1 SIMHYD scenarios. In 3 of the 4 scenarios where RGN is outperformed by SCE, RGN maintains good reliability, $\mathscr{R}_G^{(\mathrm{RGN})} \approx 50-84\%$. RGN is more G-reliable than SCE-nc2 in the HYMOD Tambo scenario (by over 40%) and in the SIMHYD Bass scenario (by 16%).
- T-reliability: RGN and SCE-nc2 achieve high T-reliability in 6 of 12 scenarios, namely, in all HYMOD and SIMHYD scenarios. In the remaining 6 scenarios, $\mathscr{R}_T^{(\mathrm{SCE\text{-}nc2})}$ ranges from 42% to 100%, while $\mathscr{R}_T^{(\mathrm{RGN})}$ ranges from 65% to 96%; SCE-nc2 is more T-reliable in 4 of these 6 scenarios.
- Computational cost: RGN invocations are cheaper than SCE-nc2 in all but 1 scenario (HYMOD Tambo). The cost difference is much lower than in the RGN versus SCE-nc10 comparison (section 5.1.1). In the HYMOD, SIXPAR, and SIMHYD scenarios, $N_\Phi^{(\mathrm{SCE\text{-}nc2})}$ ranges from (approximately) 490 to 1,600, whereas $N_\Phi^{(\mathrm{RGN})}$ of RGN ranges from 330 to 790. In the FUSE scenarios, $N_\Phi^{(\mathrm{SCE\text{-}nc2})} \approx 4{,}000-8{,}000$, whereas $N_\Phi^{(\mathrm{RGN})} \approx 1{,}650-2{,}500$. In relative terms, RGN is cheaper than SCE-nc2 by factors ranging from 0.7 to 3.0 in the HYMOD, SIXPAR, and SIMHYD scenarios, and by factors ranging from 1.6 to 5.0 in the FUSE scenarios.

Figure 2 compares the estimated number of invocations M_X required by RGN and SCE-nc2 to find global and tolerable optima with 95% confidence:

- Global optimum: Over all scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 120 with a median of 4.5 and an IQR of 36.5, while $M_G^{(\mathrm{SCE\text{-}nc2})}$ ranges from 1 to 30,000 with a median of 2 and an IQR of 2,275. Excluding the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 10 with a median value 3 and an IQR of 4.5, and $M_G^{(\mathrm{SCE\text{-}nc2})}$ ranges from 1 to 30,000 with a median 1 and an IQR of 3.0. In the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 47 to 120, and $M_G^{(\mathrm{SCE\text{-}nc2})}$ ranges from 110 to 3,000.
- Tolerable optimum: $M_T^{(\mathrm{RGN})}$ ranges from 1 to 3 with a median of 1 and an IQR of 1, while $M_T^{(\mathrm{SCE\text{-}nc2})}$ ranges from 1 to 6 with a median of 1 and IQR of 0, and in 10 of 12 scenarios $M_T^{(\mathrm{SCE\text{-}nc2})} = 1$.

Figure 3 compares the efficiency of RGN relative to SCE-nc2:

- G-efficiency: RGN outperforms SCE-nc2 in 9 of 12 scenarios, with G-efficiency ratios $\kappa_G^{(\mathrm{RGN/SCE\text{-}nc2})}$ ranging from 1.2 to 3,330 with a median of 2.95. The 3 exceptions are the SIMHYD Tambo, SIXPAR Tambo, and SIXPAR Bass scenarios, where the G-efficiency ratios are 0.44, 0.53 and 0.89, respectively.
- T-efficiency: RGN outperforms SCE-nc2 in 10 of 12 scenarios, with T-efficiency ratios $\kappa_T^{(\mathrm{RGN/SCE\text{-}nc2})}$ ranging from 1.2 to 9.7 with a median of 2.2. The 2 exceptions in this case are the FUSE Tambo and HYMOD Tambo scenarios, where the T-efficiency ratios are 0.8 and 0.7, respectively.

5.1.3. Comparison of RGN against LM

Figure 1 shows that RGN is overall significantly more robust but always costlier than LM. More specifically,

- G-reliability: RGN outperforms LM in 8 of 12 scenarios, is worse in 1 scenario, and marginally better but nonetheless similarly-poor in all 3 FUSE scenarios, where both RGN and LM struggle to locate the global optimum. In the 8 scenarios, RGN improvements over LM range between +20% in HYMOD Bass and over +80% in SIXPAR Coopers. In the FUSE scenarios, LM has zero reliability while RGN has reliabilities of 3–7%.
- T-reliability: RGN and LM are highly T-reliable in 2 of 12 scenarios. In the remaining 10 scenarios, RGN consistently outcompetes LM and maintains high T-reliability in 4 of those scenarios. In all 3 FUSE scenarios, the T-reliability of RGN ranges from 65% to 92% while for LM it ranges from 2% to 20%.
- Computational cost: LM invocations are consistently and significantly cheaper than RGN. In the HYMOD, SIXPAR, and SIMHYD scenarios, $N_\Phi^{(\mathrm{LM})}$ ranges from 150 to 480, while $N_\Phi^{(\mathrm{RGN})}$ ranges from 330 to 790. In the FUSE scenarios, $N_\Phi^{(\mathrm{LM})}$ ranges from 590 to 720, and $N_\Phi^{(\mathrm{RGN})}$ ranges from 1,650 to 2,500. In relative

terms, RGN is always costlier than LM, by factors ranging from 1.3 to 4.8 in the HYMOD, SIXPAR, and SIMHYD scenarios, and by factors ranging from 2.3 to 3.5 in the FUSE scenarios.

Figure 2 compares the estimated number of invocations M_X required by RGN and LM to find global and tolerable optima with 95% confidence:

- Global optimum: Over all scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 120 with a median of 4.5 and an IQR of 36.5, while $M_G^{(\mathrm{LM})}$ ranges from 1 to 15,000 with a median of 103 and an IQR of 2,993. Excluding the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 10 with a median value 3 and an IQR of 4.5, and $M_G^{(\mathrm{LM})}$ ranges from 1 to 15,000 with a median 22 and an IQR of 173. In the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 47 to 120, and $M_G^{(\mathrm{LM})}$ requires 3,000 invocations.
- Tolerable optimum: $M_T^{(\mathrm{RGN})}$ ranges from 1 to 3 with a median of 1 and an IQR of 1, while $M_T^{(\mathrm{LM})}$ ranges from 1 to 1,069 with a median of 9 and an IQR of 70.

Figure 3 compares the efficiency of RGN relative to LM:

- G-efficiency: RGN outperforms LM in 11 of 12 scenarios, with G-efficiency ratios $\kappa_G^{(\mathrm{RGN/LM})}$ ranging from 1.1 to 960 with a median of 7.2. The single exception is the SIMHYD Tambo scenario, where the G-efficiency ratio is 0.4.
- T-efficiency: RGN outperforms LM in 9 of 12 scenarios, with T-efficiency ratios $\kappa_T^{(\mathrm{RGN/LM})}$ ranging from 1.1 to 170 with a median of 5.3. The 3 exceptions are the HYMOD Bass, HYMOD Tambo, and SIMHYD Coopers scenarios, where the T-efficiency ratios are about 0.6.

5.1.4. Comparison of RGN against DDS

Figure 1 shows that RGN is always more robust than DDS using the budget selected in this study. More specifically,

- G-reliability: RGN outperforms DDS in all 12 scenarios. In the HYMOD, SIMHYD, and SIXPAR scenarios, RGN outperforms DDS substantially, with improvements in reliability ranging from 3% to 81%. In the FUSE scenarios, $\mathscr{R}_G^{(\mathrm{RGN})}$ is low and ranges from 2.5% to 6.3%, yet at least the global optimum is found; in contrast, $\mathscr{R}_G^{(\mathrm{DDS})} = 0$.
- T-reliability: RGN outperforms DDS in all scenarios. In the HYMOD, SIMHYD, and SIXPAR scenarios, RGN significantly outperforms DDS in all but one scenario (SIMHYD Coopers Creek), where both DDS and RGN have high T-reliability. In the FUSE scenarios, where $\mathscr{R}_T^{(\mathrm{DDS})}$ has low values in the range of $<1\%$ to 16%, $\mathscr{R}_T^{(\mathrm{RGN})}$ has clearly higher values in the range of 65% to 92%.
- Computational cost: DDS invocations use a predetermined budget as the termination criteria. In this work the DDS budget is assigned to approximately match the typical cost of RGN invocations, with 800 objective function calls for HYMOD, SIXPAR, and SIMHYD, and 2,500 calls for FUSE.

Figure 2 compares the estimated number of invocations M_X required by RGN and DDS to find global and tolerable optima with 95% confidence:

- Global optimum: Over all scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 120 with a median of 4.5 and an IQR of 36.5, while $M_G^{(\mathrm{DDS})}$ ranges from 7 to 3,000 with a median of 44.5 and an IQR of 2,280. Excluding the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 1 to 10 with a median value 3 and an IQR of 4.5, and $M_G^{(\mathrm{DDS})}$ ranges from 7 to 178 with a median of 15 and an IQR of 96. In the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ ranges from 47 to 120, and $M_G^{(\mathrm{DDS})}$ is as high as 3,000.
- Tolerable optimum: $M_T^{(\mathrm{RGN})}$ ranges from 1 to 3 with a median of 1 and an IQR of 1, while $M_T^{(\mathrm{DDS})}$ ranges from 1 to 748 with a median of 6 and an IQR of 34.

Figure 3 compares the efficiency of RGN relative to DDS:

- G-efficiency: RGN outperforms DSS in all scenarios, with G-efficiency ratios $\kappa_G^{(\mathrm{RGN/DDS})}$ ranging from 2.1 to 80 with a median of 26.
- T-efficiency: RGN outperforms DDS in all scenarios, with T-efficiency ratios $\kappa_T^{(\mathrm{RGN/DDS})}$ ranging from 1.5 to 380 with a median of 12.

5.2. Experiment 2: Analysis of search trajectories

Figure 4 presents trace plots of objective function value against the number of objective function calls after each iteration of the RGN and the two SCE configurations, for 100 invocations starting from different initial seeds. These plots provide visual information about the algorithms' patterns and rates of convergence, and whether termination criteria are unduly affecting the optimization cost. Five scenarios depicting a

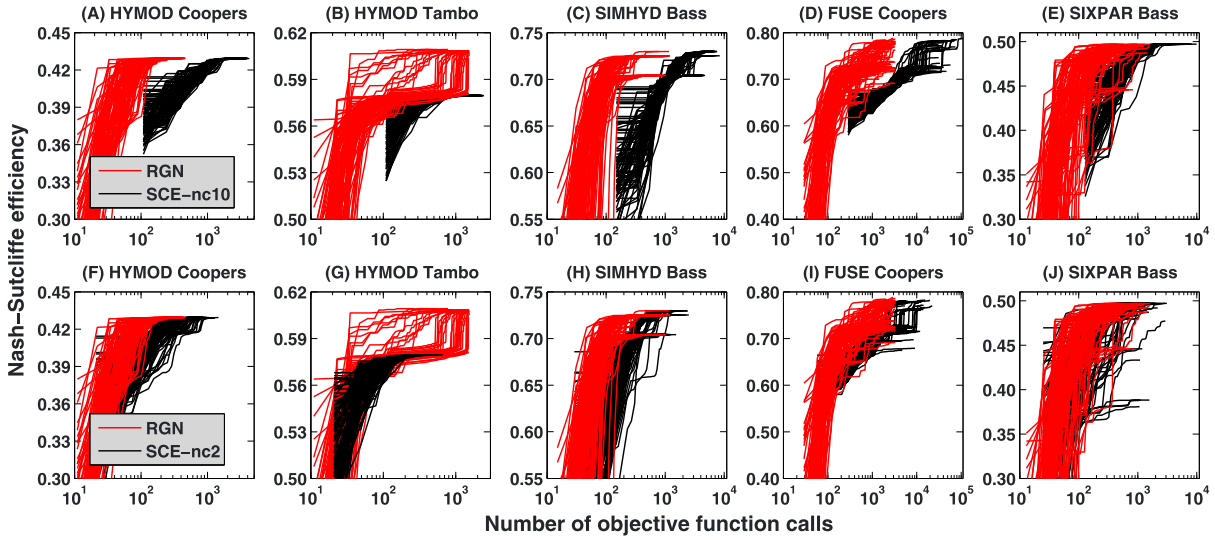

Figure 4. Traces of individual optimization trajectories of RGN and SCE (nc10 and nc2) algorithms, shown as the "best" objective function value found as optimization progresses plotted against the corresponding computational cost. Results for representative scenarios (models and catchments) are shown. Algorithm abbreviations: RGN = robust Gauss-Newton; SCE = shuffled complex evolution.

range of behavior are presented, namely HYMOD Coopers Creek, HYMOD Tambo River, SIMHYD Bass River, FUSE Coopers Creek, and SIXPAR Bass River. Note that the trajectories are plotted starting from the number of function calls needed to initialize the algorithms: while RGN requires a single initial point, SCE requires the initialization of its entire population of complexes, each comprising $(2 \times N_\theta) + 1$ individual solution estimates. For this reason, SCE-nc2 and SCE-nc10 trajectories start, respectively, at just below 60 and slightly above 110 function calls.

First, we consider the progress pattern throughout individual invocations. RGN and both SCE configurations exhibit similar types of behavior. Initially, individual invocations tend to converge rapidly towards a local optimum. From then on, three distinct convergence patterns are observed: (i) some invocations converge to the global optimum gradually, as seen in Panel A for all RGN and SCE invocations; (ii) some invocations jump sharply to better attractor regions; for example, in Panel B, many RGN invocations eventually jump from NSE of 0.58 to NSE of 0.61; and (iii) some invocations are trapped by local optima; this behavior is seen in Panel C where RGN and both SCE algorithms can get trapped by the same three optima, with NSE 0.73, 0.725, and 0.70.

Second, we compare the rate of progress of RGN and SCE, complementing the cost summaries in Figure 1. RGN clearly converges much faster than either SCE configuration. For example, in Panel A, most RGN invocations approach the global optimum of NSE 0.429 very rapidly, reaching an NSE of 0.426 after just 200 function calls and terminating after at most 500 function calls; in contrast many SCE-nc10 invocations reach an NSE of 0.426 only after as many as 2,000 function calls. RGN's speed advantage over SCE-nc2 is smaller but still very substantial, with the latter starting to converge after about 800 function calls. Qualitatively similar behavior is seen in Panels C/H and D/I, where RGN is much faster than SCE-nc10 and moderately faster than SCE-nc2.

Third, we consider the variability in the behavior of individual invocations. Across multiple invocations of a single algorithm, all three types of behavior listed above can occur, and a range of costs can be incurred, for example, as seen in Panel C for both the RGN and SCE-nc10 applications. That said, some scenarios do exhibit consistent algorithm performance—either consistently good or consistently poor. In some scenarios all optimization invocations converge without trouble (Panel A), whereas in Panels B and G all SCE-nc10 and SCE-nc2 invocations converge to a local optimum (whereas nearly 50% of the RGN invocations are successful). This shows of course that some objective functions are tougher to crack than others.

Finally, we consider the evidence of invocations with unduly long and nonproductive traces near termination. The overall behavior of the traces is as expected, displaying initially steep trajectories that gradually

level off. There is some evidence of plateaus before termination in several scenarios, for example, in Panels A, B, and E for SCE-nc10; in Panels G for SCE-nc2; and in Panel C for RGN. For example, in Panel E, SCE-nc10 spends over 5,000 function calls with little or no improvement in the objective function value before the algorithm is terminated, increasing costs from 3,000 to 8,000 function calls. Such expensive plateaus are not unexpected because SCE is set to require 3 complex reshuffles with no significant improvement before terminating, and such reshuffles are expensive when 10 complexes are used. However in some cases, notably for RGN and SCE-nc10 in Panels B and D, similar plateaus are eventually broken and better optima found. As a tight tolerance can see an algorithm invocation through such plateaus, relaxing the tolerance in an attempt to reduce costs is not a safe option, as it risks premature convergence and a substantial loss of reliability.

6. Discussion

We now discus and interpret the results of benchmarking the RGN algorithm. Our focus is on practical recommendations founded on consideration of trade-offs between reliability $\mathscr{R}_X$ versus invocation cost N_Φ, as well as variability in the estimated number of invocations M_X (section 4.3.2). The ideal practical algorithm will have high efficiency (low cost to find the desired optimum with high confidence) and high consistency (low variability in the number of required invocations). Armed with these perspectives, we now discuss the performance of RGN relative to each benchmarking algorithm (section 6.1), and summarize our final recommendations and ideas for further work (Section 6.2).

6.1. Interpretation of Performance Benchmarking

In comparison to SCE, RGN nearly matches its reliability—with RGN reliability often as high as $\mathscr{R}_T^{(\mathrm{RGN})} = 80 - 100\%$—but using 3–24 times fewer objective function calls than SCE-nc10 and 0.7–5 times fewer function calls than SCE-nc2 (section 5.1). This is an important finding because the SCE algorithm is generally considered amongst the most robust and widely used global optimization techniques used in hydrological model calibration (Tolson & Shoemaker, 2007). Yet RGN is sufficiently reliable and so much faster than either SCE configuration, that a fraction of its computational savings can be traded-off for virtually the same level of robustness as SCE, still leaving RGN with a savings surplus. This explains the substantial efficiency gains of RGN over both SCE configurations, with median G- and T-efficiency ratios of 8.6 and 7.4, respectively, over SCE-nc10, and 2.9 and 2.2, respectively, over SCE-nc2.

We note that SCE-nc10 exhibits the hallmarks of a robust (highly reliable and consistent) global optimizer, with $M_T^{(\text{SCE-nc10})} = 1$ in all 12 scenarios. This characterization is somewhat tempered by the fact that, while $M_G^{(\text{SCE-nc10})}$ has the ideal median of 1, its IQR is very high, 2,253, due to a failure in 4 scenarios. The high robustness of SCE-nc10 comes at a major computational cost per invocation, and RGN offers better efficiency by the virtue of being almost as reliable, but considerably cheaper. Remarkably, RGN has better consistency, with the IQR of $M_G^{(\mathrm{RGN})}$ as low as 37.

In comparison, SCE-nc2 is less consistent than SCE-nc10: $M_G^{(\text{SCE-nc2})}$ has a median 2 and IQR of 2,275, and $M_T^{(\text{SCE-nc2})}$ has a median 1 and IQR of 0. This performance suggests that SCE-nc2 is appreciably inferior to SCE-nc10 in terms of reliability and consistency, and cannot be considered to be in the same class of global performance. As such, RGN achieves similar robustness and consistency to SCE-nc2, and generally outperforms it by being somewhat faster per invocation.

Importantly, RGN appears to be more robust than SCE-nc10 and SCE-nc2 under extreme conditions. In particular, in the FUSE scenarios, $M_G^{(\mathrm{RGN})}$ is at most 120, whereas $M_G^{(\text{SCE-nc10})}$ and $M_G^{(\text{SCE-nc2})}$ reach as high as 3,000.

With reference to the mathematical structure of RGN and SCE, it follows that, when used with large-sampling scales, quadratic approximations require fewer samples to provide global search directions of comparable quality to the (shuffled) populations of simplexes. There are also instances (HYMOD and FUSE scenarios) where none or virtually none of the SCE-nc10 and SCE-nc2 invocations locate the global optimum, whereas RGN is moderately successful ($\mathscr{R}_G^{(\mathrm{RGN})} = 43\%$ in the HYMOD Tambo scenario and $\mathscr{R}_G^{(\mathrm{RGN})} = 2.5 - 6.3\%$ in the FUSE scenarios)—this highlights that no optimization algorithm can truly guarantee global convergence (Gill et al., 1981).

In comparison to LM, RGN achieves clearly better reliability, for example, in 3 SIXPAR scenarios we find $\mathscr{R}_G^{(\mathrm{LM})} \approx 0\%$ versus $\mathscr{R}_G^{(\mathrm{RGN})} \approx 50 - 80\%$—but at the expense of an appreciable increase in computational cost, by factors of 1.3–4.8 depending on the scenario (Section 5.1). G-efficiency ratios of RGN over LM range from 1.1 to 960 in 11 of 12 scenarios, and the T-efficiency ratios range from 1.1 to 170 in 9 scenarios. This efficiency gain implies that, in the majority of scenarios, the increased reliability of RGN more than compensates for its cost and tips the efficiency ratio in its favor. RGN is less G-efficient in 1 scenario ($\kappa_G^{(\mathrm{RGN/LM})} = 0.4$) and less T-efficient in 3 scenarios ($\kappa_T^{(\mathrm{RGN/LM})} = 0.6$).

Overall, LM is a typical local optimizer: computationally fast but suffering from both low reliability and consistency especially when applied to geometrically complex objective functions. The low reliability of LM manifests in low G- and T- reliability, and its low consistency manifests in high variability in M_X. RGN is substantially more G-consistent than LM: the IQR of $M_G^{(\mathrm{LM})}$ is very high, 2,993, whereas the IQR of $M_G^{(\mathrm{RGN})}$ is much lower, 37. RGN's advantage over LM is even stronger in terms of T-consistency: the IQR of $M_G^{(\mathrm{LM})}$ is 70 whereas the IQR of $M_T^{(\mathrm{RGN})}$ is just 1! The cost penalty of (a single invocation of) RGN compared to LM is appreciable but does not outstrip the improvement in reliability, resulting in RGN being overall more efficient. This finding confirms the gains in robustness identified in the companion paper (Qin et al., 2018) against the standard GN algorithm.

As the underlying approximations of the objective function by quadratic surfaces with Hessian $\mathbf{H} \approx \mathbf{J}^T\mathbf{J}$ are similar in RGN and LM, the improved reliability and consistency of RGN can be attributed to the use of large sampling scales. This interpretation is consistent with the companion paper, which established the major gains in robustness when the standard GN algorithm is augmented with the LSS, BSP and NSJ schemes. The increase in computational cost of RGN compared to LM is consistent with the discussion in the companion paper—enhanced exploration generally results in increased costs.

In comparison to DDS, RGN dominates in terms of achieving consistently higher reliabilities $\mathscr{R}_G$ and $\mathscr{R}_T$ for a comparable (by construction) cost-per-invocation N_Φ. The G-efficiency ratios of RGN over DDS are as high as 2.1–80, and T-efficiency ratios are even higher, 1.5–380. This finding indicates that, when applied with a large sampling scale, the quadratic Gauss-Newton approximation is substantially better at exploring the objective function surface than the near-random Gaussian Markov Chain search strategy underlying DDS. For example, at any scale, quadratic surfaces can stretch to adapt to highly correlated parameter surfaces—this is clearly impossible for a random search using an uncorrelated Gaussian distribution in a randomly selected subset of dimensions.

DDS is close to a random search and hence is liable to be inefficient when the objective function has highly structured features, such as parameter correlations. Overall, DDS appears non-robust, at least for the considered set of model calibration scenarios: it suffers from both low reliability and consistency, especially when applied to geometrically complex objective functions. The low reliability of DDS manifests in low G- and T- reliability, and the low consistency manifests in IQRs of $M_G^{(\mathrm{DDS})}$ and $M_T^{(\mathrm{DDS})}$ as high as 2,300 and 34 respectively. As such, RGN outperforms DDS by being both substantially more reliable and consistent for about the same computational budget.

Finally, note that our efficiency metric is based on an estimate of the total cost required to find desired optima (global and/or local) with a prescribed confidence level, and may be sensitive to the selected termination criteria. Experiment 2 suggests no evidence of termination criteria unduly affecting cost. That said, for slow models, it could be that budget-based measures of efficiency are more appropriate (Tolson & Shoemaker, 2007). But even in this respect RGN is hardly ceding any ground: it clearly converges faster than both SCE algorithms even during initial phases of the search—contradicting a common perception that Newton-type methods suffer from an initially slow convergence (Press et al., 2007).

6.2. Algorithm Recommendations and Further Work

In summary, the empirical results indicate that RGN offers appreciable efficiency gains over the current best-practice algorithms SCE, LM, and DDS included in the benchmarking. RGN is recommended in preference to both SCE-nc10 and SCE-nc2 to find the global optimum. In the case study scenarios, RGN required a median of 5 (and at most 120) invocations to achieve 95% confidence in finding the global optimum—a lower and

tighter range than its competitors. Importantly, RGN consistently incurred a lower total cost in terms of the total number of objective function calls. If a tolerable optimum is required, RGN can be used with a median of 1 (and most 3) invocation—similar to SCE-nc10 and SCE-nc2, but once again generally faster in terms of total number of function calls.

The ideal practical algorithm would have both high efficiency and consistency. None of the algorithms benchmarked in this work meet this ideal, due to a loss of reliability when faced with highly irregular objective function surfaces. RGN appears to come closest to this ideal, offering less variability in its performance than even the SCE algorithm. However, this conclusion is contingent on the hydrological models and catchments used in the empirical analysis. While the 12 modeling scenarios represent some of the spectrum of objective function pathologies encountered in conceptual rainfall-runoff modeling, broader assessment is recommended as the next step to confirm the generality of the findings. This broader assessment should also include the application of RGN to heavily parameterized computationally slow distributed hydrological and river basin models.

A practical limitation of the RGN algorithm—and any Gauss-Newton-type method including the LM method—is that they are designed for Least Squares objective functions. To tackle more general model calibration setups, the extension of RGN heuristics to other gradient-based algorithms, such as quasi-Newton methods, is of interest. The use of regularization approaches to improve the conditioning of the Jacobian matrices in the RGN algorithm is also of interest and is expected to help in the solution of high-dimensional and otherwise poorly-posed inverse problems (Doherty & Skahill, 2006; Tonkin & Doherty, 2005). Finally, RGN offers opportunities to provide better initial points for probabilistic uncertainty analysis using frequentist and Bayesian methods, as well as to provide estimates of parameter covariance using its (LSS) approximation of the Hessian matrix (Kavetski, 2018).

7. Conclusions

Parameter optimization is a key step in the application of hydrological models. The companion paper introduced a RGN algorithm for model optimization using least squares objective functions, which are widely used in practical applications. This paper focuses on benchmarking the RGN algorithm against optimization algorithms arguably accepted as best practice in hydrological modeling. Three benchmarking algorithms are used, namely the stochastic SCE and DDS algorithms, and the Gauss-Newton-type LM algorithm from the PEST package (Doherty, 2005). The benchmarking is carried out using four hydrological models with objective functions ranging from smooth to very rough, 3 catchments with a range of runoff ratios, yielding a total of 12 modeled scenarios.

The following empirical findings are obtained:

1. The RGN algorithm approaches the robustness of the SCE algorithm, while retaining the low cost of Newton-type algorithms. In terms of efficiency, which reflects the cost of finding the desired optimum with 95% confidence, RGN outperforms SCE with 10 complexes in all 12 scenarios, by factors with a median of 8.6 and 7.4 in terms of finding global and tolerable optima respectively. In comparison to SCE with 2 complexes, RGN is more efficient in 10 of 12 scenarios, by factors with a median of 3 and 2.2 in terms of finding global and tolerable optima respectively. However, neither RGN nor SCE are "guaranteed" to achieve global convergence, and their performance, in particular the chance to find the global optimum, deteriorates when working with nonsmooth models such as FUSE with explicit time stepping;
2. The RGN algorithm achieves clearly higher reliability than the LM algorithm at a moderate additional computational cost. In terms of efficiency in finding the global optimum, RGN outperforms LM in 11 of 12 scenarios by a median factor of 7.2, and in terms of efficiency in finding tolerable optima in 9 of 12 scenarios by a median factor of 5.3.
3. The RGN algorithm achieves clearly higher reliability than the DDS algorithm. RGN outperforms DDS in all 12 scenarios, with gains by a median factor of 26.1 in terms of finding global optima and gains by a median factor of 12.7 in terms of finding tolerable optima. Note that the comparison with DDS was constructed as "reliability-per-fixed-cost";
4. RGN has a low median number and comparatively low variability in the estimated number of invocations required to find global and tolerable optima with 95% confidence, making it attractive for practical applications. Based on performance over the 12 scenarios, a single RGN invocation is likely to be sufficient to

find a tolerable optimum; of the order of 120 invocations (but a median of just 4.5 invocations) might be needed to find the global optimum. In this respect, RGN is clearly preferable to LM and DDS. Although RGN performance is on average comparable to SCE-nc10 and SCE-nc2, RGN appears more robust when the objective function is highly irregular geometrically;

5. RGN makes quick progress during the early stages of optimization, with progress generally slowing down as the search nears the optimal regions. This pattern is qualitatively similar to the convergence patters of the SCE algorithm, which contradicts the notion that Newton-type methods are fast only once close to the optimum, and confirms the general robustness and global convergence properties of the RGN algorithm. In some instances, plateaus are observed at various stages of the RGN and SCE optimization trajectories, indicating the need to set the termination tolerance in a way that balances the risks of premature (false) convergence versus wasted effort at the end of the optimization.

The RGN algorithm achieves high robustness at a moderate cost, ultimately offering major efficiency benefits over the benchmarking algorithms used in this study. On the basis of our empirical findings, we recommend the RGN algorithm for practical optimization of hydrological models to least squares objective functions. Future work will explore the benefits of RGN in more general model calibration contexts, including the use of non-SSE objective functions, the optimization of computationally slow highly parameterized hydrological and river system models, the use of regularization approaches, and the use of optimization as a building block for probabilistic modeling and uncertainty analysis.

Appendix A: Termination criteria for RGN and PEST

In this work, the RGN and LM algorithms employ the termination criteria used in the PEST package and the broader optimization literature (Doherty, 2005; Duan et al., 1992; see also Qin et al., 2018). Iterations are terminated when any of the following criteria are met:

1. Function-based criterion: objective function value, $\Phi^{(i)} = \Phi(\boldsymbol{\theta}^{(i)})$, has not decreased over $N_{\Delta\Phi=0} = 4$ consecutive iterations;
2. Function-based criterion: (scaled) reductions in the objective function value, $(\Phi^{(i+1)} - \Phi^{(i)})/\max(|\Phi^{(i+1)}|, \Phi^{(\text{scale})})$, below a tolerance, $\tau_\Phi = 10^{-5}$, over $N_{\Delta\Phi\approx 0} = 5$ consecutive iterations;
3. Solution-based criterion: (scaled) changes in parameter values, $\max_{1\le k\le N_{\boldsymbol{\theta}}} |\theta_k^{(i+1)} - \theta_k^{(i)}| / \max(|\theta_k^{(i+1)}|, \theta_k^{(\text{scale})})$, below a tolerance, $\tau_{\boldsymbol{\theta}} = 10^{-5}$, over $N_{\Delta\boldsymbol{\theta}\approx\mathbf{0}} = 5$ consecutive iterations;
4. Iteration-based criterion: maximum number of iterations, $N_{\text{iter}}^{(\max)} = 10^2$, is exceeded;

The algorithmic settings above are based on the recommended values in the PEST manual (Doherty, 2005); they are seen as reasonable values, especially in terms of benchmarking against PEST.

Acknowledgments

We thank the Editor, the Associated Editor, and the three reviewers for their helpful comments and suggestions. We thank David McInerney for assistance provided during the earlier stages of this work. This study was partially supported by the Fundamental Research Funds for the Central Universities (2018B55414), the National Natural Science Foundation of China (41561134016 and 51879068), the China Scholarship Council, and the Australian Research Council (Linkage grant LP100200665). Hydrological data for the Tambo River, Bass River, and Coopers Creek catchments were obtained from Francis Chiew. The pseudocode for the RGN algorithm is listed in Appendices A and B of Qin et al. (2018); a Fortran-95 implementation is available from https://github.com/eachonly/Robust-Gauss-Newton-Algorithm.

References

Arsenault, R., Poulin, A., Côté, P., & Brissette, F. (2014). Comparison of stochastic optimization algorithms in hydrological model calibration. *Journal of Hydrologic Engineering*, *19*(7), 1374–1384. https://doi.org/10.1061/(ASCE)HE.1943-5584.0000938

Bates, B. C., & Campbell, E. P. (2001). A Markov chain Monte Carlo scheme for parameter estimation and inference in conceptual rainfall-runoff modeling. *Water Resources Research*, *37*(4), 937–947. https://doi.org/10.1029/2000WR900363

Bates, D. M., & Watts, D. G. (2007). *Nonlinear regression analysis and its applications*. New York: Wiley.

Behrangi, A., Khakbaz, B., Vrugt, J. A., Duan, Q., & Sorooshian, S. (2008). Comment on "Dynamically dimensioned search algorithm for computationally efficient watershed model calibration" by Bryan A. Tolson and Christine A. Shoemaker. *Water Resources Research*, *44*, W12603. https://doi.org/10.1029/2007WR006429

Boyle, D. P. (2001). Multicriteria calibration of hydrologic models, (PhD thesis). University of Arizona, Tucson.

Chiew, F. H. S., & Siriwardena, L. (2005). Estimation of SIMHYD parameter values for application in ungauged catchments. In A. Zerger & R. M. Argent (Eds.), *Modsim 2005: International Congress on Modelling and Simulation* (pp. 2883–2889). Melbourne, Australia.

Clark, M. P., & Kavetski, D. (2010). Ancient numerical daemons of conceptual hydrological modeling: 1. Fidelity and efficiency of time stepping schemes. *Water Resources Research*, *46*, W10510. https://doi.org/10.1029/2009WR008894

Doherty, J. (2003). Ground water model calibration using pilot points and regularization. *Ground Water*, *41*(2), 170–177. https://doi.org/10.1111/j.1745-6584.2003.tb02580.x

Doherty, J. (2005). *PEST: Model independent parameter estimation* (Fifth edition of user manual ed.). Brisbane, Australia: Watermark Numerical Computing.

Doherty, J., & Skahill, B. E. (2006). An advanced regularization methodology for use in watershed model calibration. *Journal of Hydrology*, *327*(3–4), 564–577. https://doi.org/10.1016/j.jhydrol.2005.11.058

Duan, Q., Gupta, V. K., & Sorooshian, S. (1993). Shuffled complex evolution approach for effective and efficient global minimization. *Journal of Optimization Theory and Applications*, *76*(3), 501–521. https://doi.org/10.1007/Bf00939380

Duan, Q., Sorooshian, S., & Gupta, V. (1992). Effective and efficient global optimization for conceptual rainfall-runoff models. *Water Resources Research*, *28*(4), 1015–1031. https://doi.org/10.1029/91WR02985

Gill, P. E., Murray, W., & Wright, M. H. (1981). *Practical optimization*. London; New York: Academic Press.

Gouvêa, É. J. C., Regis, R. G., Soterroni, A. C., Scarabello, M. C., & Ramos, F. M. (2016). Global optimization using q-gradients. *European Journal of Operational Research*, *251*(3), 727–738. https://doi.org/10.1016/j.ejor.2016.01.001

Gupta, V. K., & Sorooshian, S. (1985). The automatic calibration of conceptual catchment models using derivative-based optimization algorithms. *Water Resources Research*, *21*(4), 473–485. https://doi.org/10.1029/WR021i004p00473

Hill, M. C., Kavetski, D., Clark, M. P., Ye, M., Arabi, M., Lu, D., et al. (2015). Practical use of computationally frugal model analysis methods. *Groundwater*, *54*(2), 159–170. https://doi.org/10.1111/gwat.12330

Kavetski, D. (2018). Parameter estimation and predictive uncertainty quantification in hydrological modelling. In Q. Duan, F. Pappenberger, J. Thielen, A. Wood, H. L. Cloke, & J. C. Schaake (Eds.), *Handbook of hydrometeorological ensemble forecasting* (Chap. 25–1, pp. 1–42). Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-642-40457-3_25-1

Kavetski, D., & Clark, M. P. (2010). Ancient numerical daemons of conceptual hydrological modeling: 2. Impact of time stepping schemes on model analysis and prediction. *Water Resources Research*, *46*, W10511. https://doi.org/10.1029/2009WR008896

Kavetski, D., & Kuczera, G. (2007). Model smoothing strategies to remove microscale discontinuities and spurious secondary optima in objective functions in hydrological calibration. *Water Resources Research*, *43*, W03411. https://doi.org/10.1029/2006WR005195

Kavetski, D., Kuczera, G., Thyer, M., & Renard, B. (2007). Multistart Newton-type optimisation methods for the calibration of conceptual hydrological models. In L. Oxley & D. Kulasiri (Eds.), *Modsim 2007: International Congress on Modelling and Simulation* (pp. 2513–2519). Christchurch, New Zealand: Modelling and Simulation Society Australia and New Zealand.

Kavetski, D., Qin, Y., & Kuczera, G. (2018). The fast and the robust: Trade-offs between optimization reliability, cost and efficiency in the calibration of hydrological models. *Water Resources Research*, *54*. http://doi.org/10.1029/2017WR022051

Koziel, S., & Leifsson, L. (2013). *Surrogate-based modeling and optimization applications in engineering*. New York: Springer. https://doi.org/10.1007/978-1-4614-7551-4

Kuczera, G. (1983). Improved parameter inference in catchment models. 2. Combining different kinds of hydrologic data and testing their compatibility. *Water Resources Research*, *19*(5), 1163–1172. https://doi.org/10.1029/WR019i005p01163

Kuczera, G., Kavetski, D., Renard, B., & Thyer, M. (2016). Bayesian methods. In V. P. Singh (Ed.), *Handbook of applied hydrology* (Chap. 23, 2nd ed.). New York: McGraw-Hill Education.

McInerney, D., Thyer, M., Kavetski, D., Lerat, J., & Kuczera, G. (2017). Improving probabilistic prediction of daily streamflow by identifying Pareto optimal approaches for modeling heteroscedastic residual errors. *Water Resources Research*, *53*, 2199–2239. https://doi.org/10.1002/2016WR019168

Nash, J. E., & Sutcliffe, J. V. (1970). River flow forecasting through conceptual models part I—A discussion of principles. *Journal of Hydrology*, *10*(3), 282–290. https://doi.org/10.1016/0022-1694(70)90255-6

Nocedal, J., & Wright, S. J. (2006). *Numerical optimization*. New York: Springer.

Press, W., Teukolsky, S., Vetterling, W., & Flannery, B. (2007). *Numerical recipes: The art of scientific computing*. Cambridge, UK and New York: Cambridge University Press.

Qin, Y. (2017). A robust and efficient optimization algorithm for hydrological models, (PhD thesis). The University of Newcastle, Australia.

Qin, Y., Kavetski, D., & Kuczera, G. (2018). A robust Gauss-Newton algorithm for the optimization of hydrological models: From Gauss-Newton to robust Gauss-Newton. *Water Resources Research*, *54*. https://doi.org/10.1029/2017WR022488

Qin, Y., Kuczera, G., & Kavetski, D. (2017). Comparison of Newton-type and SCE optimisation algorithms for the calibration of conceptual hydrological models. *Australasian Journal of Water Resources*, *20*(2), 169–176. https://doi.org/10.1080/13241583.2017.1298180

Rogue Wave Software (2017). IMSL® Fortran numerical math library, release version 7.1, chapter 8 optimization, algorithm UNLSF. Retrieved from http://docs.roguewave.com/imsl/fortran/7.1/html/fnlmath/index.html

Skahill, B. E., & Doherty, J. (2006). Efficient accommodation of local minima in watershed model calibration. *Journal of Hydrology*, *329*(1–2), 122–139. https://doi.org/10.1016/j.jhydrol.2006.02.005

Sorooshian, S., Duan, Q., & Gupta, V. K. (1993). Calibration of rainfall-runoff models: Application of global optimization to the Sacramento soil moisture accounting model. *Water Resources Research*, *29*(4), 1185–1194. https://doi.org/10.1029/92WR02617

Tolson, B. A., & Shoemaker, C. A. (2007). Dynamically dimensioned search algorithm for computationally efficient watershed model calibration. *Water Resources Research*, *43*, W01413. https://doi.org/10.1029/2005WR004723

Tolson, B. A., & Shoemaker, C. A. (2008). Reply to comment on "Dynamically dimensioned search algorithm for computationally efficient watershed model calibration" by Ali Behrangi et al. *Water Resources Research*, *44*, W12604. https://doi.org/10.1029/2008WR006862

Tonkin, M. J., & Doherty, J. (2005). A hybrid regularized inversion methodology for highly parameterized environmental models. *Water Resources Research*, *41*, W10412. https://doi.org/10.1029/2005WR003995

Vrugt, J. A., Robinson, B. A., & Hyman, J. M. (2009). Self-adaptive multimethod search for global optimization in real-parameter spaces. *IEEE Transactions on Evolutionary Computation*, *13*(2), 243–259. https://doi.org/10.1109/TEVC.2008.924428

Yen, H., Jeong, J., Tseng, W.-H., Kim, M.-K., Records, R. M., & Arabi, M. (2015). Computational procedure for evaluating sampling techniques on watershed model calibration. *Journal of Hydrologic Engineering*, *20*(7), 04014080. https://doi.org/10.1061/(ASCE)HE.1943-5584.0001095

Zhao, R. (1992). The Xinanjiang model applied in China. *Journal of Hydrology*, *135*(1-4), 371–381. https://doi.org/10.1016/0022-1694(92)90096-E