Randomised Postiterations for Calibrated BayesCG

Niall Vyas — University of Southampton, United Kingdom
Disha Hegde — University of Southampton, United Kingdom
Jon Cockayne — University of Southampton, United Kingdom

Abstract

The Bayesian conjugate gradient method offers probabilistic solutions to linear systems but suffers from poor calibration, limiting its utility in uncertainty quantification tasks. Recent approaches leveraging postiterations to construct priors have improved computational properties but failed to correct calibration issues. In this work, we propose a novel randomised postiteration strategy that enhances the calibration of the BayesCG posterior while preserving its favourable convergence characteristics. We present theoretical guarantees for the improved calibration, supported by results on the distribution of posterior errors. Numerical experiments demonstrate the efficacy of the method in both synthetic and inverse problem settings, showing enhanced uncertainty quantification and better propagation of uncertainties through computational pipelines.

Proceedings of the 1st International Conference on Probabilistic Numerics (ProbNum) 2025, Sophia Antipolis, France. PMLR Volume 271. Copyright 2025 with the author(s)

1 Introduction

Probabilistic numerical methods provide a framework for solving numerical problems while quantifying uncertainty arising from finite computational resources. The Bayesian conjugate gradient method (BayesCG; see Cockayne et al. (2019)) is increasingly widely used as such a solver for linear systems of the form $Ax = b$, where the matrix A and vector b are known and the vector x is to be determined. The method constructs a Gaussian posterior over the solution space, reflecting uncertainty due to limited iterations. However, several major challenges impact the practicality of BayesCG: (i) there are compelling theoretical and numerical reasons to use a particular prior that results in an intractable posterior, and (ii) the posteriors produced are typically poorly calibrated. The *postiteration* method of Reid et al. (2023) addresses the first of these issues but not the second. In this work we propose a randomised version of this method for which we can provide calibration guarantees.

1.1 Challenges for BayesCG

In this section we describe two central challenges that this work will address.

Poor Calibration BayesCG is well-known to suffer from poor calibration as recently demonstrated in Hegde et al. (2025), due to nonlinearities in its conditioning procedure. Specifically, BayesCG operates by conditioning a Gaussian prior on information of the form $S(b)^\top Ax = S(b)^\top b$, where the search directions $S(b)$ are generated by the Lanczos process (Golub and Van Loan, 2013, Section 10.1). The information is therefore nonlinear in x, while the posterior is constructed by assuming that S is independent of x to apply *linear* conditioning, for a conjugate Gaussian posterior.

Inverse Prior There is a specific choice of prior for BayesCG that has favourable mathematical properties, but is intractable: the *inverse prior* $x \sim \mathcal{N}(x_0, A^{-1})$. Under this choice the BayesCG posterior mean is identical to that from CG, which can be computed without computing A^{-1}, and therefore converges at a rapid geometric rate in the number of iterations performed. However the posterior covariance still requires explicit computation of A^{-1}. In certain circumstances this requirement has been sidestepped using elegant marginalisation techniques (Wenger et al., 2022), but for more general applications this remains a challenge.

1.2 Postiterations

To address the challenges discussed in Section 1.1, recent work has proposed *empirical Bayesian* approaches that construct an x-dependent prior (Reid et al., 2020, 2023) in a way that is consistent with the inverse prior to retain the favourable mathematical properties of CG. This is typically constructed by performing several *postiterations* of CG, and using those iterations to construct the prior. The major downside of this work is that the posterior remains uncalibrated, which can have serious consequences in applications in which uncertainty is propagated through a computational pipeline, such as in an inverse problem, as we consider in this work. We therefore propose a straightforward modification of the postiteration procedure under which the posterior has more favourable calibration properties.

1.3 Contributions

This paper makes the following contributions to the literature.

1. We propose a new algorithm that exploits *randomised* postiterations to achieve a well-calibrated posterior while maintaining the favourable computational properties of Reid et al. (2020, 2023).

(Section 3)

2. We prove several theoretical results related to the calibration of this procedure. (Theorem 3.2 and Corollary 3.3)

3. We test the calibration properties of the posterior, demonstrating that it performs favourably compared to existing approaches, and also show that it has favourable performance in uncertainty-propagation applications. (Section 4)

1.4 Structure of the Paper

The rest of the paper proceeds as follows. In Section 2 we present the required background on BayesCG, calibration and postiteration methods. Section 3 introduces our randomised postiteration method. Section 4 presents numerical results related to calibration of the novel method and how it can be propagated through numerical pipelines. We conclude with some discussion in Section 5.

2 Background

In this section we will introduce the required background for the novel methodology introduced in Section 3. In Section 2.1 we present the salient details of BayesCG, Section 2.2 discusses statistical calibration, and Section 2.3 introduces the Krylov prior that our methodology modifies to achieve calibration.

2.1 BayesCG

BayesCG is an iterative probabilistic linear solver for problems of the form $Ax = b$, where $A \in \mathbb{R}^{d\times d}$ is an invertible matrix and $b \in \mathbb{R}^d$ is a vector, each given, while $x \in \mathbb{R}^d$ is an unknown vector to be determined. Most iterative solvers start from a user-specified *initial iterate* x_0 and iteratively construct a sequence (x_m), $x_m \in \mathbb{R}^d, m \in \mathbb{N}$, such that $x_m \to x$ as $m \to \infty$. BayesCG differs in that it takes a user-supplied Gaussian distribution as its initial iterate, $\mu_0 = \mathcal{N}(x_0, \Sigma_0)$, and returns a sequence (μ_m), $\mu_m = \mathcal{N}(x_m, \Sigma_m), m \in \mathbb{N}$, where $x_m \in \mathbb{R}^d$ and $\Sigma_m \in \mathbb{R}^{d\times d}_{\geq 0}$, the set of all positive semidefinite $d \times d$ matrices.

The means and covariances x_m and Σ_m are computed using Bayesian methods. We first define an information operator

$$\mathcal{I}_m(x) = S_m^\top A x$$

where $S_m \in \mathbb{R}^{d\times m}, S_m = [s_1, \dots, s_m]$ with $s_1, \dots, s_m \in \mathbb{R}^d$. The vectors $s_1, \dots, s_m$ are a set of linearly independent *search directions*. Linearity of the information operator results in a closed-form posterior distribution through the well-known Gaussian conditioning formula:

$$x_m = x_0 + \Sigma_0 A^\top S_m \Lambda_m^{-1} S_m^\top (b - Ax_0) \tag{1a}$$
$$\Sigma_m = \Sigma_0 - \Sigma_0 A^\top S_m \Lambda_m^{-1} S_m^\top A \Sigma_0 \tag{1b}$$

where $\Lambda_m = S_m^\top A \Sigma_0 A^\top S_m$.

BayesCG arises from a particular choice of search directions given in Cockayne et al. (2019).

Definition 2.1 (BayesCG Directions). *The* BayesCG directions *are given by* $s_1 = r_0$ *and*

$$s_m = r_{m-1} - (s_{m-1}^\top A\Sigma_0 A^\top r_{m-1}) \cdot \frac{s_{m-1}}{(s_{m-1}^\top A\Sigma_0 A^\top s_{m-1})},$$

where $r_m = b - Ax_m$.

These directions have the property of diagonalising Λ_m (see Cockayne et al., 2019, Proposition 7), resulting in an iterative expression for the posterior mean and covariance (Cockayne et al., 2019, Proposition 6).

In this paper we will restrict attention to the case of the inverse prior $\Sigma_0 = A^{-1}$, where A is assumed to be symmetric and positive definite. With this prior the posterior from Eq. (1) simplifies to

$$x_m = x_0 + S_m (S_m^\top A S_m)^{-1} S_m^\top (b - Ax_0) \tag{2a}$$
$$\Sigma_m = A^{-1} - S_m (S_m^\top A S_m)^{-1} S_m^\top, \tag{2b}$$

highlighting the dependence of the posterior covariance on A^{-1} mentioned in Section 1. The search directions from Definition 2.1 also simplify to the following definition.

Definition 2.2 (BayesCG directions under the inverse prior). *The* BayesCG directions *with* $\Sigma_0 = A^{-1}$ *are given by* $s_1 = r_0$ *and*

$$s_m = r_{m-1} - (s_{m-1}^\top A r_{m-1}) \cdot \frac{s_{m-1}}{(s_{m-1}^\top A s_{m-1})}$$

where $r_m = b - Ax_m$.

The search directions in Definition 2.2 coincide with the search directions used in the CG algorithm (Algorithm 1). Moreover, when these directions are used, the posterior mean in Eq. (2a) coincides with the CG iterate, and therefore enjoys all of the favourable properties of that iterate, such as its geometric rate of convergence[1]. This highlights the rationale behind the inverse prior. For other prior covariances analogous convergence results can be proven (see Cockayne et al., 2019, Proposition 10), but the rate is typically slower. Moreover, there is a growing literature on Gaussian process approximation (Wenger et al., 2022, 2024; Stankewitz and Szabo, 2024; Hegde et al., 2025; Pförtner et al., 2025) that *mandates* use of the A^{-1} prior.

The search directions in Definition 2.2 have several important theoretical properties. First recall that the Krylov space generated by a matrix A and a vector v is given by

$$K_m(A, v) := \operatorname{span}(v, Av, \dots, A^{m-1}v).$$

The *grade* of a Krylov space is the smallest M for which $K_{M+1}(B, v) = K_M(B, v)$. We will also assume throughout that all Krylov spaces involved in this work have full grade $M = d$, but note that Reid et al. (2020)

[1]This is a worst-case result; typical rates can be much faster.

Algorithm 1 Conjugate Gradient Method

```
procedure CG($A$, $b$, $x_0$, $\epsilon$)
    $r_0 \leftarrow b - Ax_0$
    $s_1 \leftarrow r_0$
    $k \leftarrow 1$
    while true do
        $\alpha_k \leftarrow r_{k-1}^\top r_{k-1}/(s_k^T A s_k)$
        $x_k \leftarrow x_{k-1} + \alpha_k s_k$
        $r_k \leftarrow r_{k-1} - \alpha_k A s_k$
        if $\|r_k\| < \epsilon \|b\|$ then
            break
        $\beta_k \leftarrow \frac{r_k^\top r_k}{r_{k-1}^\top r_{k-1}}$
        $k \leftarrow k + 1$
        $s_k \leftarrow r_{k-1} + \beta_{k-1} s_{k-1}$
    return $x_k$
```

carefully constructs BayesCG with $M < d$. Defining the m^{th} *residual* as $r_m = b - Ax_m$, the required properties of the directions are given in the following theorem, which follows from Golub and Van Loan (2013, Section 10.1 and Theorem 11.3.5).

Theorem 2.3. *The CG directions have the following properties:*

1. *s_i, s_j are A-orthogonal (i.e. $s_i^\top A s_j = 0$ if $i \neq j$).*
2. *$\operatorname{span}(s_1, \ldots, s_m) = K_m(A, r_0)$.*

Remark 2.4. *Note that a more general version of Theorem 2.3 holds for arbitrary Σ_0; see Li and Fang (2019).*

As a final point, we note that the posterior distribution from BayesCG has a singular posterior covariance and, significantly for the discussion in the next section, that its null space depends on the true solution x.

Theorem 2.5 (Propositions 1 and 4 in the rejoinder from Cockayne et al. (2019))**.** *The matrix Σ_m from Eq. (2b) has rank $d - m$ and its null space is given by $K_m(A, r_0)$.*

To mitigate the dependence of Eq. (2) on A^{-1}, a particular line of research advocates for employing the CG iterates to compute x_m and calculating Σ_m implicitly using some additional *postiterations*. This will be described in Section 2.3. First however we must discuss *calibration*.

2.2 Statistical Calibration

Formal calibration of probabilistic numerical methods has been increasingly studied in recent years, with Cockayne et al. (2022, 2021) introducing definitions of strong calibration that were applied to (non-Bayesian) probabilistic iterative methods, while Hegde et al. (2025) showed that calibration guarantees transfer from a probabilistic numerical method to a downstream application, in computation-aware GPs (Wenger et al., 2022). We argue that this is particularly important for Krylov-based probabilistic linear solvers, where previously mentioned miscalibration has been observed repeatedly.

The central challenge of applying calibration definitions in this setting is the support of the posterior. It is generally true that probabilistic numerical methods produce posteriors that are defined on a submanifold of the original space, and so any definition of calibratedness must hold in this setting. The original definition of strong calibration (Cockayne et al., 2022, Definition 8) is not suitable because of the required regularity assumptions (Cockayne et al., 2022, Definitions 2 and 7) which essentially require that the posterior has full support.

Cockayne et al. (2021); Hegde et al. (2025) circumvented this by exploiting the fact that the posteriors studied in those works, while not having *full* support, nevertheless all had the *same* support independent of x, allowing the more flexible definition from Cockayne et al. (2021, Definition 9); this essentially demands strong calibration on the common posterior support, and that the posterior mean matches the truth in the complement.

For Krylov-based posteriors we cannot apply a similar argument because, as highlighted in Theorem 2.5, the support of the posterior depends on the Krylov space explored, which in turn depends on the true solution x. This is randomised according to the prior in the definition of strong calibration. Applying the definition of weak calibration (Cockayne et al., 2022, Definition 13) yields similar issues. We therefore turn to a yet weaker calibration condition which has been used for Krylov solvers in the past (Cockayne et al., 2019; Reid et al., 2023).

Definition 2.6 (χ^2-Calibration)**.** *Consider a learning procedure on $\mathbb{R}^d$, $\mu_m(x) = \mathcal{N}(x_m, \Sigma_m)$, where Σ_m has rank $d - m$. Introduce the Z-statistic*

$$Z(x) = (x_m - x)^\top \Sigma_m^\dagger (x_m - x)$$

where $\Sigma_m^\dagger$ is the Moore-Penrose pseudoinverse of Σ_m. We say that the learning producedure is χ^2-calibrated for the prior μ_0 if it holds that $Z(X) \sim \chi^2_{d-m}$ when X is distributed according to μ_0.

While this is clearly weaker than the definitions of either strong or weak calibration, it solves the support problem through the projection onto $\mathbb{R}$ through Z. We will therefore use this condition throughout this paper. We defer the important problem of a more general definition of strong calibration suitable for Krylov probabilistic linear solvers to future work.

2.3 The Krylov Prior

To mitigate dependence of the posterior covariance on A^{-1} described in Section 2.1, one approach that has been proposed is the *Krylov prior* (Cockayne et al., 2019; Reid et al., 2020, 2023).

The Krylov prior can in the first instance be constructed by observing that if $\operatorname{span}\{v_1, \ldots, v_K\} \subseteq \mathbb{R}^d$, we can construct a Gaussian belief on $\mathbb{R}^d$ as

$$\begin{aligned} X &= x_0 + \sum_{i=1}^{K} v_i \phi_i^{\frac{1}{2}} \zeta_i \\ &= x_0 + V_K \Phi^{\frac{1}{2}} Z \end{aligned}$$

where

$$\begin{aligned} Z &= [\zeta_1, \dots, \zeta_K]^\top \sim \mathcal{N}(0, I) \\ \Phi &= \operatorname{diag}(\phi_1, \dots, \phi_K) \in \mathbb{R}^{K \times K} \\ V_K &= [v_1, \dots, v_K] \in \mathbb{R}^{d \times K}. \end{aligned}$$

The law of X is therefore $\mathcal{N}(x_0, V_K \Phi V_K^\top)$.

The idea behind the Krylov prior is to take $v_1, \dots, v_d$ to be A-orthonormal and such that $\operatorname{span}\{v_1, \dots, v_m\} = K_m(A, r_0)$. Going forward, to simplify notation we will take

$$V_m = S_m (S_m^\top A S_m)^{-\frac{1}{2}}$$

for each $m > 0$. We refer to V_m as the *normalised* search directions and S_m as the *unnormalised* search directions. Note that in implementations we will use unnormalised directions, which will be discussed in Section 3.1.1.

We then have the following theoretical results concerning the posterior:

Theorem 2.7 (Theorem 3.3 of Reid et al. (2020)). *Let $\Sigma_0 = V_d \Psi V_d^\top$ for some (as-yet unspecified) diagonal Ψ. The posterior mean and covariance conditioned on information of the form $S_m^\top A x = S_m^\top b =: y_m$, $m < d$, satisfies*

$$\begin{aligned} x_m &= x_0 + V_m V_m^\top r_0 \\ &= x_0 + V_m \boldsymbol{\psi}_m \\ \Sigma_m &= \bar{V}_m \bar{\Psi}_m \bar{V}_m^\top \end{aligned}$$

where $\bar{V}_m = [v_{m+1} \dots v_d]$, $\bar{\Psi}_m = \operatorname{diag}(\bar{\boldsymbol{\psi}}_m)$, $\boldsymbol{\psi}_m = [\psi_1, \dots, \psi_m]^\top$ and $\bar{\boldsymbol{\psi}}_m = [\psi_{m+1}, \dots, \psi_d]^\top$.

The next theorem proposes a choice of scalings ψ_i under which a particular measure of the size of the posterior covariance matches the error precisely.

Theorem 2.8 (Theorem 3.7 of Reid et al. (2020)). *Let*

$$\psi_i = \frac{r_{i-1}^\top r_{i-1}}{(s_i^\top A s_i)^{\frac{1}{2}}},$$

$i = 1, \dots, d$. Then $\operatorname{tr}(A \Sigma_m) = \|x - x_m\|_A^2$.

The rationale for the interest in $\operatorname{trace}(A\Sigma_m)$ is due to Reid et al. (2020, Lemma 3.5), which states that $\mathbb{E}\|X - x_m\|_A^2 = \operatorname{trace}(A\Sigma_m)$ when $X \sim N(x_m, \Sigma_m)$. Combining this with the result above, the intuition is that the expected A-norm distance between samples from X and its mean matches the CG error with this particular choice of weights.

Remark 2.9. *Note that Reid et al. (2020) represents these computations using $\phi_i = \psi_i^2$.*

Remark 2.10. *Under this choice of ψ_i the prior and posterior covariance are dependent on b (and thus x) and so this could be regarded as an empirical Bayesian approach. This does not preclude application of the calibration ideas defined in Section 2.2, since the learning procedure is only required to return a measure on $\mathbb{R}^d$; it is permissible for this measure to have a complex dependance on x.*

From Reid et al. (2020, Theorem SM3.2), we have that ψ_i can be equivalently calculated as

$$\psi_i = v_i^\top r_0.$$

It will also be helpful to establish the identity:

$$\frac{\psi_i}{(s_i^\top A s_i)^{\frac{1}{2}}} = \alpha_i \tag{3}$$

where α_i is as given in Algorithm 1.

While Theorem 2.8 shows that the width of the posterior covariance is "about right", the next theorem shows that with this choice of ϕ_i, the posterior is decidedly not calibrated.

Theorem 2.11. *With the choice from Theorem 2.8, the posterior is not χ^2-calibrated.*

Proof. From Reid et al. (2023, Theorem 4.13) $Z(x) = d - m$, independent of x. Therefore $Z(X)$ is a Dirac mass at $d - m$, and so the posterior is not χ^2-calibrated. $\square$

The proof given above highlights that the miscalibration is due to a *perfect cancellation* of the randomness induced by randomising x according to the prior, caused by the choice of scales in Theorem 2.8.

2.3.1 Practical Implementation

It must be mentioned that while the above is theoretically appealing, it is not practical. This is owing to the requirement that the full Krylov basis V_d be computed in order to define the posterior. The process of computing this basis is identical to that of running CG to convergence, which would make uncertainty quantification (UQ) irrelevant.

To address this, Reid et al. (2023) propose to instead run CG for t iterations, where $m < t \ll d$, and use the $t - m$ *postiterations* to construct the posterior. Due to the fast convergence of CG, this means that the most dominant subspace of $\operatorname{span}(\bar{V}_m)$ will still be captured with only a small number of postiterations. On the other hand, the resulting posterior will assign zero mass to the space $\operatorname{span}(\bar{V}_t)$, falsely implying that there is no error in that space, which could be problematic for some applications. Cockayne et al. (2019) proposed an approach to address this by computing an orthonormal basis of $\bar{V}_t$ using a QR decomposition, but this did not seem to work well empirically. We will explore the impact of this in Section 4, but will leave attempts to correct it for future work.

A further objection to this idea is that, if one has the budget to perform t iterations, why not use the better estimate of x provided by these iterations with no UQ? We argue that in some applications and downstream tasks, having a better calibrated probabilistic linear solver is more useful than a more accurate (but still high error) estimate, a conclusion supported by results in Poot et al. (2025). This idea is explored empirically in Section 4.2.

In the next section we propose our novel, randomised version of the Krylov prior that has more favourable calibration properties.

3 Randomised Postiterations

3.1 A Randomised Krylov Prior

First recall the following results from the proof of (Reid et al., 2023, Theorem 4.13):

$$\begin{aligned} x &= x_0 + V_d \boldsymbol{\psi}_d \\ x_m &= x_0 + V_m \boldsymbol{\psi}_m \\ \Sigma_m &= \bar{V}_m \bar{\Psi}_m \bar{V}_m^\top . \end{aligned}$$

An immediate consequence of this is that

$$x - x_m = \bar{V}_m \bar{\boldsymbol{\psi}}_m .$$

This leads us to consider *randomising* x_m to reintroduce the randomness that, as previously mentioned, is perfectly cancelled by the choice in Theorem 2.8.

Definition 3.1 (Randomised Postiterations)**.** *Let x_m be the posterior mean in Eq. (2a). The randomised posterior mean is given by $\tilde{x}_m = x_m + \bar{V}_m e_m$ where $\bar{V}_m$ is as given in Theorem 2.7 and*

$$e_m \sim \mathcal{N}(\bar{\boldsymbol{\psi}}_m, \bar{\Psi}_m),$$

where $\bar{\boldsymbol{\psi}}_m$ and $\bar{\Psi}_m$ are as given in Theorems 2.7 and 2.8.

Our first result below provides a (pointwise) guarantee that the randomised posterior follows a distribution that is intuitively correct. Notably this is stronger than required for Definition 2.6.

Theorem 3.2. *The randomised posterior from Definition 3.1 satisfies $L_m^\dagger(x - \tilde{x}_m) \sim \mathcal{N}(0, I_{d-m})$, where L_m is an arbitrary left square-root of Σ_m.*

Proof. First note that a left square-root of Σ_m is given by

$$L_m = \bar{V}_m \bar{\Psi}_m^{\frac{1}{2}} .$$

We can straightforwardly calculate the Moore-Penrose pseudoinverse of this as

$$L_m^\dagger = \bar{\Psi}_m^{-\frac{1}{2}} (\bar{V}_m^\top \bar{V}_m)^{-1} \bar{V}_m^\top .$$

(Reid et al., 2023, Lemma 4.12). We then have that

$$\begin{aligned} L_m^\dagger(x - \tilde{x}_m) &= L_m^\dagger(x - x_m) - L_m^\dagger \bar{V}_m e_m \\ &= \bar{\Psi}_m^{-\frac{1}{2}} (\bar{V}_m^\top \bar{V}_m)^{-1} \bar{V}_m^\top \bar{V}_m (\bar{\boldsymbol{\psi}}_m - e_m) \\ &= \bar{\Psi}_m^{-\frac{1}{2}} (\bar{\boldsymbol{\psi}}_m - e_m) \end{aligned}$$

which clearly has the distribution stated in the theorem. □

Theorem 3.2 has a notably different interpretation than other calibration results in the literature because the distribution is not induced by randomness in x but by randomness in $\tilde{x}_m$. Nevertheless we emphasise that because both $\bar{V}_m$ and $\bar{\Psi}_m$ are functions of x, the posterior depends highly nontrivially on the true solution which complicates calibration analysis. The next result establishes χ^2-calibratedness of the randomised posterior.

Corollary 3.3. *The randomised posterior from Definition 3.1 is χ^2-calibrated.*

Algorithm 2 The Randomised Postiteration procedure described in Section 3.

```
procedure CG_RPI($A$, $b$, $x_0$, $\epsilon_1$, $\epsilon_2$)
    $r_0 \leftarrow b - Ax_0$
    $s_1 \leftarrow r_0$
    $k \leftarrow 1$
    $\tilde{x}_k \leftarrow$ NOTHING
    $L_k \leftarrow [\,] \in \mathbb{R}^{d \times 0}$
    while true do
        $\alpha_k \leftarrow \frac{r_{k-1}^\top r_{k-1}}{s_k^\top A s_k}$
        $x_k \leftarrow x_{k-1} + \alpha_k s_k$
        $r_k \leftarrow r_{k-1} - \alpha_k A s_k$
        $\beta_k \leftarrow \frac{r_k^\top r_k}{r_{k-1}^\top r_{k-1}}$
        if $\neg$ISNOTHING($\tilde{x}_k$) then
            if $\|r_k\| > \epsilon_2 \|b\|$ then
                $z_k \sim \mathcal{N}(0, 1)$
                $\tilde{x}_k \leftarrow \tilde{x}_{k-1} + \alpha_k (z_k + 1) s_k$
                $L_k \leftarrow \begin{bmatrix} L_{k-1} & \alpha_k s_k \end{bmatrix}$
            else
                break
        else if $\|r_k\| \le \epsilon_1 \|b\|$ then
            $\tilde{x}_k \leftarrow x_k$
        $k \leftarrow k + 1$
        $s_k \leftarrow r_{k-1} + \beta_{k-1} s_{k-1}$
    return $\tilde{x}_k, L_k$
```

Proof. Since $\Sigma_m = L_m L_m^\top$, it follows that $\Sigma_m^\dagger = (L_m^\top)^\dagger L_m^\dagger$. We therefore have that

$$Z(x) = \|L_m^\dagger (x - \tilde{x}_m)\|_2^2 .$$

From Theorem 3.2, this is the squared norm of a $\mathcal{N}(0, I_{d-m})$ distributed random variable, which completes the proof. □

An interesting property of Theorem 3.2 and Corollary 3.3 is that both results are independent of the choice of $\bar{\Psi}_i$; because e_m has this as its covariance, the correct distribution will always be obtained. This reflects a general point of calibration results, that calibration is a property that is independent of other favourable properties of the posterior, such as rate of concentration, or indeed, whether the posterior concentrates at all (e.g. provided the covariance and the mean error diverge commensurately, the posterior can still be calibrated). To ensure that the posterior reflects the true error we recommend setting $\bar{\Psi}_i$ in the same way as in Theorem 2.8. Since these quantities are already computed during the postiterations, this does not change the cost of computation.

3.1.1 Practical Implementation

We now consider practical implementation of the procedure described in Theorem 3.2. Two main optimisations are discussed below, with the resulting algorithm given in Algorithm 2.

Recursion with Unnormalised Directions For the first m iterations the algorithm is identical to Algorithm 1, so we focus on the differences in the postiteration period where $k > m$. Note that $\tilde{x}_k$ can be computed iteratively by setting $\tilde{x}_m = x_m$, and $\tilde{x}_k = \tilde{x}_{k-1} + v_k e_k$, where $e_k \sim \mathcal{N}(-\psi_k, \psi_k^2)$. Next note that typically computations are performed using the directions s_k rather than v_k for stability. The randomised posterior mean can therefore be computed as

$$\begin{aligned}\tilde{x}_k &= \tilde{x}_{k-1} + \left(\frac{\psi_k}{(s_k^\top A s_k)^{\frac{1}{2}}} + \frac{\psi_k}{(s_k^\top A s_k)^{\frac{1}{2}}} z_k\right) s_k \\ &= \tilde{x}_{k-1} + \alpha_k (z_k + 1)\, s_k\end{aligned}$$

with $z_k \sim \mathcal{N}(0, 1)$.

Note that it is also straightforward to calculate the posterior covariance recursively. As in Cockayne et al. (2019) we opt to represent this using the low-rank factors L_k, where $L_m = [\,] \in \mathbb{R}^{d\times 0}$ and

$$L_k = \begin{bmatrix} L_{k-1} & \alpha_k s_k \end{bmatrix}$$

for $k > m$. Then we have that $\Sigma_k = L_k L_k^\top$.

Truncated Postiterations As was discussed in Section 2.3.1 it is not possible to realise $\bar{V}_m$ in practice. We propose to adopt the same approach discussed there of truncating the postiterations. That is, we perform t total iterations of CG, $m < t \ll d$, and use the $t - m$ postiterations to calibrate the posterior. To implement this truncation we introduce a second tolerance, $\epsilon_2 < \epsilon_1$ in Algorithm 2. The parameter ϵ_1 controls at which point standard CG iterations are terminated as with ϵ in Algorithm 1, while ϵ_2 controls how many postiterations are performed. That is, ϵ_1 defines m while ϵ_2 defines t. Standard CG is then recovered in the case where $\epsilon_1 = \epsilon_2$. To avoid computing too many further postiterations we suggest setting $\epsilon_2 = \delta\epsilon_1$ for some $0 < \delta < 1$ (e.g. $\delta = 0.1$). The impact of the choice of δ will be explored in Section 4.

4 Experiments

4.1 Simulation-Based Calibration

Since Corollary 3.3 provides fairly weak calibration guarantees, we begin by examining posterior calibratedness empirically. To do this we employ the simulation-based calibration (SBC) test of Talts et al. (2018), described in Algorithm 3. We use a Gaussian prior $\mu_0 = \mathcal{N}(0, A^{-1})$, and set $A = WDW^\top$, where W is an orthonormal matrix drawn from the Haar measure on such matrices, while D is diagonal with strictly positive entries drawn independently from Exp(1). All SBC experiments use a symmetric positive definite matrix $A \in \mathbb{R}^{d\times d}$ of dimension $d = 100$. This matrix is kept fixed across the simulations for individual runs of the SBC test (Figs. 1a to 1c) but redrawn for repeated runs (Fig. 1d). If calibrated, the histogram of implied posterior CDF evaluations should be distributed as $\mathcal{U}(0, 1)$.

4.1.1 Results

The results of the simulation-based calibration tests demonstrate that the posterior is indeed calibrated for sufficiently small choices of ϵ_2.

Algorithm 3 Simulation-Based Calibration. Φ denotes the CDF of the standard Gaussian distribution.

```
Require: Prior $\mu_0 = \mathcal{N}(u_0, A^{-1})$, number of simulations $N_{\text{SIM}}$, test vector $w$.
1 for $i = 1$ to $N_{\text{SIM}}$ do
2     $x^{(i)} \sim \mu_0$
3     $b^{(i)} \leftarrow Ax^{(i)}$
4     $x_m^{(i)}, L_m^{(i)} = \text{RPI}(A, b^{(i)}, x_0, \epsilon_1, \epsilon_2)$
5     $t_i = \Phi\left(\frac{w^\top (x_m^{(i)} - x^{(i)})}{(w^\top L_m^{(i)} (L_m^{(i)})^\top w)^{\frac{1}{2}}}\right)$
6 Plot histogram of $\{t_i\}_{i=1}^{N_{\text{SIM}}}$ and compare to $\mathcal{U}(0, 1)$.
```

Fig. 1a presents the results from 10,000 simulations of the procedure outlined in Algorithm 3, applied to the postiterations approach of Reid et al. (2023), with $\epsilon_2 = 10^{-5}$ and $\epsilon_1 = 10^{-1}$. As anticipated, the histogram exhibits a ∪-shaped pattern, indicating poor calibration with the computed posteriors being, on average, under-dispersed relative to the true posterior (Talts et al., 2018). In comparison, the results from the randomised version of the postiteration algorithm, with the same number of simulations and choice of the thresholds appear uniform in Fig. 1b, suggesting that the posterior is well-calibrated for this choice of thresholds.

However, the randomised algorithm does exhibit under-dispersed posteriors for larger choices of ϵ_2. This is seen in Fig. 1c where $\epsilon_2 = 10^{-2}$ and $\epsilon_1 = 10^{-1}$. We again observe the characteristic ∪-shaped histogram indicating generally over-confident posteriors but note that it is less extreme than the results from Fig. 1a, even with the relatively larger ϵ_2.

A quantitative comparison of the two algorithms is provided in Fig. 1d. The Kolmogorov-Smirnov (KS) test statistic is used to estimate the difference between the SBC results and the standard uniform distribution, for different choices of thresholds. Results were obtained by averaging the KS statistics from 1000 SBC runs, each using 1000 simulations. The median KS statistics along with their interquartile range are plotted. The grey line indicates the average KS statistic between samples from the standard uniform and the standard uniform itself. While the deterministic postiteration algorithm maintains the same level of poor calibration across the varying thresholds, the randomised algorithm's SBC results converges to being uniformly distributed and therefore calibrated.

Remark 4.1 (Calibration, for what measure?)**.** *While Algorithm 3 tests for calibration with-respect-to $\mathcal{N}(x_0, A^{-1})$, in fact Theorem 3.2 and Corollary 3.3 show that we should achieve calibration* independent *of the prior. This is because Algorithm 2 is independent of the prior. We have opted to use A^{-1} in Algorithm 3 only because this is the prior covariance under which BayesCG coincides with CG.*

4.2 Inverse Problem

We now consider incorporating the randomised postiteration procedure into an inverse problem. This has been considered several times in the literature, includ-

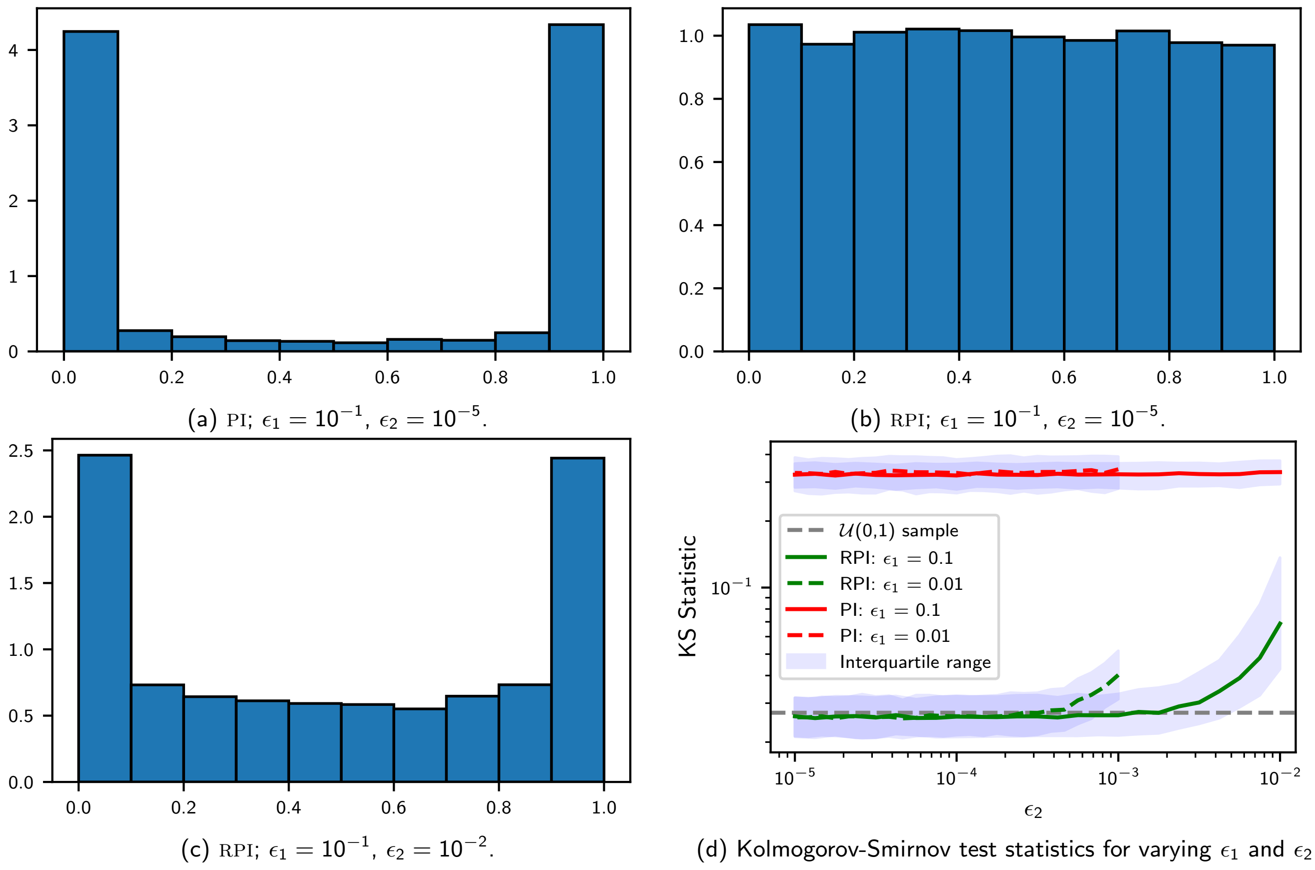

Figure 1: Simulation-based calibration test results. Figs. 1a to 1c give results for different values of ϵ_1, ϵ_2 while Fig. 1d gives KS test statistics as a function of ϵ_2, for multiple replicates of the experiment.

ing for linear solvers (see Cockayne et al., 2019). We consider a linear system arising from the discretisation of a partial differential equation describing steady-state flow through a porous medium, given as:

$$\begin{aligned} -\nabla \cdot (k(z;\theta)\nabla u(z;\theta)) &= f(z) & z &\in D \\ u(z;\theta) &= b(z) & z &\in \partial D_1 \\ \nabla u(z;\theta) \cdot n &= 0 & z &\in \partial D_2 \end{aligned}$$

where n is the outward pointing normal vector. For the domains we take

$$D = (0,1)^2 \tag{4a}$$

$$D_1 = \bigcup_{z_1 \in [0,1]} \{(z_1, 0), (z_1, 1)\} \tag{4b}$$

$$D_2 = \bigcup_{z_2 \in [0,1]} \{(0, z_2), (1, z_2)\}. \tag{4c}$$

In other words, D_1 is the top and bottom edges of the unit square and D_2 is the left and right edges. For the forcing we take $f(z) = 0$ and for the boundary term we take $b(z) = (1 - z_1)(1 - z_2) + z_1 z_2$.

The inverse porosity $k(z;\theta)$ is taken to be

$$k(z;\theta) = 1 + 1_{D_\theta}(z) \cdot \exp(\theta)$$

where $1_{D_\theta}(z) = 1$ if $z \in D_\theta$ and 0 elsewhere. We take $D_\theta = [0.25, 0.75]^2$. Thus this problem represents steady-state corner to corner flow through a domain with a low porosity central region that impedes the flow.

We set this up as a Bayesian inference problem to determine θ. To accomplish this we use the data-generating model

$$y_i = f(z_i) + \zeta_i, i = 1, \ldots, N.$$

We take $N = 4$ and $z_1, \ldots, z_4$ are taken to be the four corners of the region D_θ, while $\zeta_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$, $\sigma = 0.01$. Data was generated using the data-generating parameter $\theta^\dagger = 2$. Letting $\boldsymbol{z} = [z_1, \ldots, z_N]$ this implies a Gaussian likelihood

$$p(y \mid \theta) = \mathcal{N}(y; u(\boldsymbol{z};\theta), \sigma^2 I). \tag{5}$$

We endow θ with a Gaussian prior $\theta \sim \mathcal{N}(0, 1)$. Note however that the posterior $p(\theta \mid y)$ is non-Gaussian due to the nonlinear dependence of u on θ. We will therefore sample the posterior using Markov-chain Monte-Carlo (MCMC) methods.

To turn this into a linear system we discretise the PDE in Eq. (4) using the finite-element method, as implemented in the Julia package `Gridap` (Badia and Verdugo, 2020; Verdugo and Badia, 2022). We omit most of the details here, but the discretisation results in a (sparse) linear system $A_\theta x_\theta = b$ that must be solved for each θ to evaluate the likelihood, where b is independent of θ and x_θ has components that approximate the solution $u(z;\theta)$ at the nodal points on a fine mesh over the domain. The points in $\boldsymbol{z}$ are included in the nodal point set, and so the value of $u(\boldsymbol{z};\theta)$ may be related to x_θ as

$$u(\boldsymbol{z};\theta) \approx w x_\theta$$

for some $w \in \mathbb{R}^{N \times d}$, whose entries are either 0 or 1, to "pick out" the degrees of freedom in x_θ associated with $\boldsymbol{z}$. The likelihood in Eq. (5) is thus approximated as

$$p(y \mid \theta) \approx \hat{p}(y \mid \theta) := \mathcal{N}(y; w x_\theta, \sigma^2 I).$$

The grid size for this problem was such that the dimension of the systems involved is $d = 475$. Naturally this approach introduces discretisation error due to the use of an approximate PDE solver, but we will assume for this work that the system is discretised finely enough that the error is negligible.

We consider three approaches to solve the linear system:

EXACT: Explicit solution using the sparse Cholesky routine `CHOLMOD` (Chen et al., 2008) as is default in Julia for sparse Hermitian matrices.

CG: CG with fixed tolerance ϵ.

PI: Postiterations in the style of Reid et al. (2023), with tolerances $\epsilon_2 = \epsilon$ and $\epsilon_1 = 10\epsilon_2$.

RPI: Randomised postiterations with the same tolerances as PI.

The comparison between CG and RPI is intended to demonstrate that, in a constrained computing environment, it is sometimes better to use some of the computational budget to perform postiterations and calibrate the posterior, rather than spending all iterations on CG. When using postiterations we modify the likelihood from Eq. (5) to incorporate uncertainty from the probabilistic solver as has been described several times in the literature (e.g. Cockayne et al. (2019), which described this approach for linear systems). The resulting likelihood for RPI is given by

$$p_{\text{PN}}(y \mid \theta) = \mathcal{N}(y; w\tilde{x}_m(\theta), \sigma^2 I + w\Sigma_m(\theta)w^\top)$$

while for PI we replace $\tilde{x}_m$ with x_m. Thus, this likelihood incorporates the uncertainty due to postiterations by inflating the likelihood variance with the factor $w\Sigma_m(\theta)w^\top$.

For this simple inverse problem we applied the random walk Metropolis algorithm (Roberts and Rosenthal, 2004). We used Gaussian proposals $\theta_n \mid \theta_{n-1} \sim \mathcal{N}(\theta_{n-1}, \eta)$ with $\eta = 0.2$ for EXACT and CG, and 0.4 for PI and RPI (which have wider posteriors due to the variance inflation). We performed $N = 10,000$ iterations for each chain. In Fig. 2 kernel density estimates of the three posteriors are displayed with $\epsilon = 0.1$, to ensure that the error is still larger than the noise level. As can be seen, the MAP point of EXACT is close to $\theta^\dagger$, while the other methods show some bias: CG and PI positive and negative bias, while for RPI the bias is negative but much smaller. Notably, however, the posterior for RPI is significantly wider than for CG or PI, i.e. the posterior better represents the uncertainty due to computation, and places more mass around $\theta^\dagger$. Furthermore, both RPI and PI are also much closer to the prior, again reflecting increased scepticism about the data due to reduced computation.

5 Conclusion

In this paper we have introduced a randomised version of the postiteration algorithm from Reid et al. (2023) that has improved calibration properties. These properties have been examined empirically, and we have performed a more extensive computational examination. We have also shown how using some of a constrained computational budget for calibration can give better results in uncertainty propagation problems, such as inverse problems, than using all of the budget to compute a low-accuracy estimator.

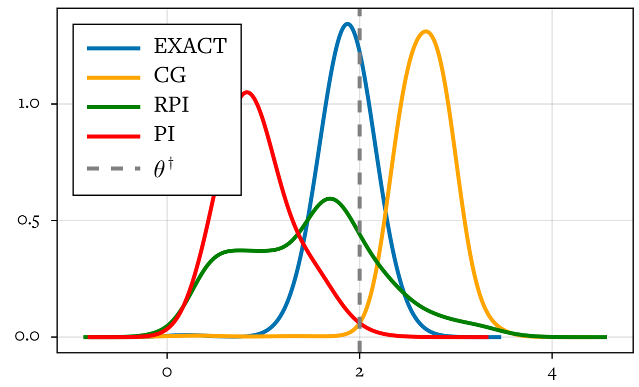

Figure 2: Posteriors for the inverse problem described in Section 4.2.

There are several avenues for future work. Firstly, the proposed procedure still suffers from the deficiency of Reid et al. (2023), that (when truncated) it assigns zero posterior mass to a subspace in which there is still error. It would be useful to examine whether techniques such as those described in Cockayne et al. (2019) can be extended to correct this, by assigning mass to this space without incurring the overhead of computing a basis for that space. Secondly, the calibration results we have derived are not as strong as we hoped. This is largely due to issues with the definition of strong calibration, which make it challenging to establish results for singular posteriors which arise in probabilistic linear solvers. A more flexible definition of strong calibration would be needed to address this, but this is beyond the scope of this paper.

A second line of work is in applications. Recent papers (Poot et al., 2024) have proposed strategies for calibrating the posterior in Bayesian finite element solvers that are similar to the postiteration ideas in Reid et al. (2023). It would be interesting to explore whether the randomisation strategies discussed in this paper can be applied to that infinite-dimensional setting to yield similar calibration improvements.

Acknowledgements

NV was supported by the UKRI AI Centre for Doctoral Training in AI for Sustainability [EP/Y030702/1]. JC was supported by EPSRC grant EP/FY001028/F1.

References

S. Badia and F. Verdugo. Gridap: An extensible finite element toolbox in Julia. *Journal of Open Source*

Software, 5(52):2520, 2020. doi: 10.21105/joss.02520. URL https://doi.org/10.21105/joss.02520.

Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software*, 35 (3):1–14, Oct. 2008. ISSN 1557-7295. doi: 10.1145/1391989.1391995. URL http://dx.doi.org/10.1145/1391989.1391995.

J. Cockayne, C. J. Oates, I. C. F. Ipsen, and M. Girolami. A Bayesian conjugate gradient method (with discussion). *Bayesian Anal.*, 14(3):937–1012, 2019. ISSN 1936-0975,1931-6690. doi: 10.1214/19-BA1145. URL https://doi.org/10.1214/19-BA1145. Includes 6 discussions and a rejoinder from the authors.

J. Cockayne, I. C. Ipsen, C. J. Oates, and T. W. Reid. Probabilistic iterative methods for linear systems. *Journal of Machine Learning Research*, 22 (232):1–34, 2021. URL http://jmlr.org/papers/v22/21-0031.html.

J. Cockayne, M. M. Graham, C. J. Oates, T. J. Sullivan, and O. Teymur. Testing whether a learning procedure is calibrated. *Journal of Machine Learning Research*, 23(203):1–36, 2022. URL http://jmlr.org/papers/v23/21-1065.html.

G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013. ISBN 978-1-4214-0794-4; 1-4214-0794-9; 978-1-4214-0859-0.

D. Hegde, M. Adil, and J. Cockayne. Calibrated computation-aware Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 2098–2106. PMLR, 2025. URL https://arxiv.org/abs/2410.08796.

X. Li and E. X. Fang. Invited discussion. Published as part of the discussion of Cockayne et al. (2019), 2019.

M. Pförtner, J. Wenger, J. Cockayne, and P. Hennig. Computation-aware Kalman filtering and smoothing. In *International Conference on Artificial Intelligence and Statistics*, pages 2071–2079. PMLR, 2025. doi: 10.48550/ARXIV.2405.08971. URL https://arxiv.org/abs/2405.08971.

A. Poot, P. Kerfriden, I. Rocha, and F. van der Meer. A bayesian approach to modeling finite element discretization error. *Statistics and Computing*, 34(5), Aug. 2024. ISSN 1573-1375. doi: 10.1007/s11222-024-10463-z. URL http://dx.doi.org/10.1007/s11222-024-10463-z.

A. Poot, I. Rocha, P. Kerfriden, and F. van der Meeer. The Bayesian finite element method in inverse problems: a critical comparison between probabilistic models for discretization error, 2025. URL https://arxiv.org/abs/2506.02815.

T. W. Reid, I. C. F. Ipsen, J. Cockayne, and C. J. Oates. BayesCG as an uncertainty aware version of CG, 2020. URL https://arxiv.org/abs/2008.03225.

T. W. Reid, I. C. F. Ipsen, J. Cockayne, and C. J. Oates. Statistical properties of BayesCG under the Krylov prior. *Numer. Math.*, 155(3-4):239–288, 2023. ISSN 0029-599X,0945-3245. doi: 10.1007/s00211-023-01375-7. URL https://doi.org/10.1007/s00211-023-01375-7.

G. O. Roberts and J. S. Rosenthal. General state space Markov chains and MCMC algorithms. *Probability Surveys*, 1(none), Jan. 2004. ISSN 1549-5787. doi: 10.1214/154957804100000024. URL http://dx.doi.org/10.1214/154957804100000024.

B. Stankewitz and B. Szabo. Contraction rates for conjugate gradient and Lanczos approximate posteriors in Gaussian process regression, 2024. URL https://arxiv.org/abs/2406.12678.

S. Talts, M. Betancourt, D. Simpson, A. Vehtari, and A. Gelman. Validating Bayesian inference algorithms with simulation-based calibration, 2018. URL https://arxiv.org/abs/1804.06788.

F. Verdugo and S. Badia. The software design of Gridap: A finite element package based on the Julia JIT compiler. *Computer Physics Communications*, 276:108341, July 2022. doi: 10.1016/j.cpc.2022.108341. URL https://doi.org/10.1016/j.cpc.2022.108341.

J. Wenger, G. Pleiss, M. Pförtner, P. Hennig, and J. P. Cunningham. Posterior and computational uncertainty in Gaussian processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. doi: 10.48550/arXiv.2205.15449. URL https://arxiv.org/abs/2205.15449.

J. Wenger, K. Wu, P. Hennig, J. R. Gardner, G. Pleiss, and J. P. Cunningham. Computation-aware Gaussian processes: Model selection and linear-time inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. doi: 10.48550/arXiv.2411.01036. URL https://arxiv.org/abs/2411.01036.