Calibrated Computation-Aware Gaussian Processes

Disha Hegde
University of Southampton

Mohamed Adil

Jon Cockayne
University of Southampton

Abstract

Gaussian processes are notorious for scaling cubically with the size of the training set, preventing application to very large regression problems. Computation-aware Gaussian processes (CAGPs) tackle this scaling issue by exploiting probabilistic linear solvers to reduce complexity, widening the posterior with additional *computational* uncertainty due to reduced computation. However, the most commonly used CAGP framework results in (sometimes dramatically) conservative uncertainty quantification, making the posterior unrealistic in practice. In this work, we prove that if the utilised probabilistic linear solver is *calibrated*, in a rigorous statistical sense, then so too is the induced CAGP. We thus propose a new CAGP framework, CAGP-GS, based on using Gauss-Seidel iterations for the underlying probabilistic linear solver. CAGP-GS performs favourably compared to existing approaches when the test set is low-dimensional and few iterations are performed. We test the calibratedness on a synthetic problem, and compare the performance to existing approaches on a large-scale global temperature regression problem.

1 INTRODUCTION

Gaussian processes are a powerful and flexible tool for Bayesian nonparametric regression, allowing a user to fit a wide array of possibly nonlinear phenomena using simple computational routines. The most major challenge in scaling Gaussian process regression to high-dimensional datasets is its cubic scaling with the number of training points, arising from the need to invert

a Gramian matrix representing the prior covariance between data points (Rasmussen and Williams, 2005). To address this, a wide array of computational approximations have been proposed, including iterative solvers (e.g. Wenger et al. (2022a)), approximations to the kernel matrix (e.g. Ferrari-Trecate et al. (1998)), and inducing point methods (e.g. Titsias (2009)).

A particularly appealing approach recently proposed in Wenger et al. (2022b) is computation-aware GPs (CAGPs). In this framework one uses a probabilistic linear solver (PLS) to solve a linear system involving the Gramian matrix and then marginalises the uncertainty from the PLS. The result is an elegant cancellation that eliminates the need to invert the Gramian (see Section 2.2.2 for a more detailed explanation). The new posterior obtained is called "computation aware" because it is widened to represent additional *computational uncertainty* due to reduced computation in the PLS, compared to an exact (but computationally prohibitive) linear solve.

The most commonly used PLS for CAGPs is BayesCG (see Cockayne et al. (2019a))—we refer to this setting as CAGP-CG. This approach is favoured because of rapid mean convergence. One of the major challenges with CAGP-CG is that resulting CAGP is typically conservative—the posterior mean is much closer to the truth than the width of the posterior covariance suggests it should be. This is inherited from the widely observed poor *calibration* properties of BayesCG, described in several works including Cockayne et al. (2019a); Bartels et al. (2019); Wenger and Hennig (2020); Reid et al. (2023). We formally introduce calibration in Section 2.3. While some empirical Bayesian methods for mitigating this issue have been proposed (e.g., Wenger and Hennig (2020); Reid et al. (2022, 2023)), they are difficult to apply within CAGPs as the choice of prior is strictly constrained by the method.

1.1 Contributions

The contributions of this paper are as follows:

- We rigorously prove that, if a calibrated PLS is used, the resulting CAGP is calibrated (Proposi-

tion 4).

- We introduce a new class of CAGPs based on probabilistic stationary iterative methods (PSIMs) (as seen in Cockayne et al. (2021)).
- We explore efficient implementation of a new CAGP approach based on Gauss-Seidel iterations (CAGP-GS, Section 4.2). CAGP-GS scales favourably compared to CAGP-CG in regimes where only a small number of test points are required.
- We explore the empirical properties of CAGP-GS on a synthetic test problem, some benchmark datasets and a large-scale geospatial regression problem (Sections 5.1 to 5.3). In particular, we note that for small iteration numbers, CAGP-GS outperforms all known alternative CAGP frameworks in terms of mean convergence and uncertainty quantification.

1.2 Structure of the Paper

The rest of the paper proceeds as follows. In Section 2 we discuss the required background on CAGPs, PLSs and calibratedness. Section 3 discusses calibratedness in the context of CAGPs, while Section 4 presents a particular class of PLSs that are both calibrated and can be used in the CAGP framework. Section 5 presents simulations demonstrating the new methodology, and we conclude in Section 6. Proofs and additional results for experiments from Section 5 are included in the supplementary material.

2 BACKGROUND

2.1 GP Regression

Suppose $f \sim \mu_0 = \mathcal{GP}(m_0, k)$, where k is a positive definite kernel function. Consider Gaussian process (GP) inference under the observation model

$$y = f(X) + \zeta \tag{1}$$

where $X \subset D$ is a set of d distinct training points in the domain of f, while $\zeta \sim \mathcal{N}(0, \sigma^2 I)$. It may be helpful to think of D as a subset of $\mathbb{R}^n$ for some n, but this is not required. The predictive distribution conditional on this information at a set of d_{TEST} test points X' is given by

$$f(X') \mid y \sim \mu = \mathcal{GP}(\bar{m}, \bar{k}) \tag{2a}$$
$$\bar{m}(X') = m_0(X') + k(X', X)v^\star \tag{2b}$$
$$\bar{k}(X', X') = k(X', X') - k(X', X)G^{-1}k(X, X') \tag{2c}$$

where $G = K(X, X) + \sigma^2 I$, while $v^\star$ is the solution to the linear system

$$Gv^\star = b. \tag{3}$$

with $b = (y - m_0(X))$.

As has been widely noted (e.g., in Rasmussen and Williams (2005)), one of the major challenges of the GP regression is that the complexity of the computations above is $\mathcal{O}(d^3)$ owing to the inversion of the Gramian matrix G. Recent work (Wenger et al. (2022b)) proposes a novel framework to mitigate this cost called CAGPs, introduced next.

2.2 Computation-Aware GPs

CAGPs center on use of PLSs to solve the system Eq. (3). We will first outline the literature on PLSs before discussing CAGPs themselves in Section 2.2.2.

2.2.1 (Bayesian) PLS

At the most generic level, PLSs are probabilistic numerical methods (PNMs) (Hennig et al. (2022); Cockayne et al. (2019b)) for solving linear systems, that is, they are learning procedures that return a probability distribution intended to quantify error due to having expended reduced computational effort[1] to calculate $v^\star$. It is common for such learning procedures to depend on some "prior" belief about $v^\star$, expressed through the distribution η_0. We therefore use the notation $\eta : \mathcal{P}(\mathbb{R}^d) \times \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$ or $(\eta_0, b) \mapsto \eta(\eta_0, b)$, where $\mathcal{P}(\mathbb{R}^d)$ is the set of all probability measures on $\mathbb{R}^d$ for a PLS. We will limit attention to Gaussian learning procedures, i.e. those that accept Gaussian input $\eta_0 = \mathcal{N}(m_0, C_0)$ and return a Gaussian output $\eta(\eta_0, b) = \mathcal{N}(\bar{v}, \bar{C})$. As we see in Section 2.2.2, the particular prior $\eta_0 = \mathcal{N}(0, G^{-1})$ (referred to as the "inverse prior" in this paper) is intrinsic to CAGPs, and we will limit attention to this throughout the following sections. As will be described in Section 2.2.2, use of this prior is a fundamental requirement for CAGPs as currently understood.

Many PLSs have a Bayesian interpretation, i.e. they are based on conditioning η_0 on observations of the form $z_m = S_m^\top b$, where $S_m \in \mathbb{R}^{d \times m}$ is a matrix of *search directions* with linearly independent columns. Under such observations and the inverse prior, the posterior is

$$v \mid z_m = \eta \sim \mathcal{N}(\bar{v}_m, \bar{C}_m) \tag{4a}$$
$$\bar{v}_m = S_m(S_m^\top G S_m)^{-1}S_m^\top b \tag{4b}$$
$$\bar{C}_m = G^{-1} - D_m^{\text{BAYES}} \tag{4c}$$
$$\bar{D}_m^{\text{BAYES}} = S_m(S_m^\top G S_m)^{-1}S_m^\top. \tag{4d}$$

---

[1] e.g. compared to applying a direct method such as Cholesky factorisation followed by two triangular solves, which calculates $v^\star$ precisely in exact arithmetic, but have cubic complexity.

This posterior is not directly computable, as computing $\bar{C}_m$ requires computation of G^{-1}, which we assumed in Section 2.1 we did *not* want to compute. Nevertheless, as we will see in Section 2.2.2, this choice leads to some cancellations for CAGPs.

Choice of Search Directions Generic choices of S_m that have been examined in the literature include standard Euclidean basis vectors and random unit vectors (Cockayne et al., 2019a; Wenger et al., 2022b; Pförtner et al., 2024). However, these typically suffer from slow convergence of $v_m \to v^\star$ compared to state-of-the-art iterative methods, making them unattractive.

A particularly important choice of S_m are those based on the conjugate gradient method (CG)[2]. In this case the PLS is often referred to as BayesCG (Cockayne et al., 2019a). These directions are favoured because (i) they can be proven to converge at a geometric rate in m in the worst case (often faster in practice), and (ii) they are G-conjugate, meaning that $S_m^\top G S_m$ is diagonal. As a result the posterior reported in Eq. (4d) simplifies further. On the other hand, these directions result in poor calibration of the posterior, which will be discussed further in Section 2.3.

Computational Complexity Ignoring the cost of computing G^{-1}, the complexity of computing Eqs. (4b) and (4d) is $\mathcal{O}(m(m^2 + d^2))$ owing to the inversion of the $m \times m$ matrix $S_m^\top G S_m$ and the requirement to compute (dense) products of the form $G S_m$ which have complexity $\mathcal{O}(md^2)$. Thus, if $m \ll d$ the complexity compared to a direct method is significantly reduced. For BayesCG the diagonal matrix inversion is reduced to $\mathcal{O}(m)$, so that the complexity is only $\mathcal{O}(md^2)$.

2.2.2 The Marginalisation Trick

The central "trick" behind CAGPs is that, given any belief from a PLSs of the form

$$v \sim \eta = \mathcal{N}(\bar{v}, G^{-1} - D) \qquad (5)$$

we can construct a new belief over $f(X')$ through the marginalisation

$$\tilde{p}(f) = \int_{\mathbb{R}^d} p(f(X') \mid v)\eta(\mathrm{d}v) \qquad (6)$$

the law of which we denote $\tilde{\mu}$. Since all the involved distributions are Gaussian, Wenger et al. (2022b) de-

rived the modified posterior as

$$\tilde{\mu} = \mathcal{GP}(\tilde{m}, \tilde{k}) \qquad (7)$$

$$\tilde{m}(X') = m_0(X') + k(X', X)\bar{v} \qquad (8)$$

$$\tilde{k}(X', X') = k(X', X') - k(X', X)Dk(X, X'). \qquad (9)$$

Notably, the presence of G^{-1} in η is *crucial* as it leads to cancellation of the downdate involving G^{-1} from Eq. (2c). As a result we need only calculate the D term in the underlying PLS, negating any need to invert G. If D can be computed at significantly lower complexity than G^{-1}, as in Section 2.2.1, then the overall cost of GP inference is reduced. Moreover, Wenger et al. (2022b, Section 2) demonstrated that $\tilde{k}(X', X')$ is "wider" than $\bar{k}(X', X')$ and can be interpreted as providing additional uncertainty quantification for the reduced computation.

Algorithm 1 Computation Aware GP

1: **function** CAGP(X, y, X', σ^2)
2:      $b = y - m_0(X)$
3:      $V = k(X', X)$
4:      $G(v) = v \mapsto k(X, X)v + \sigma^2 v$
5:      $\tilde{v}, \tilde{D} = $ CAGP_PLS(G, b, V)
6:      $\hat{m} = m_0(X') + \tilde{v}$
7:      $\tilde{k} = k(X', X') - \tilde{D}$
8:      **return** $\tilde{m}, \tilde{k}$

Algorithm 1 gives an implementation of this as pseudocode. Note that it is assumed CAGP_PLS requires access to G only through its action on vectors v, rather than explicitly, allowing for a matrix-free implementation. Algorithm 1 is a slightly modified version of that presented in Wenger et al. (2022b); the quantities $\tilde{v}$ and $\tilde{D}$ returned by the routine CAGP_PLS are given by $\tilde{v} = V\bar{v} = k(X', X)\bar{v}$, and $\tilde{D} = VDV^\top = k(X', X)Dk(X, X')$, i.e. they are the image of η under the map $v \mapsto Vv$. This is more commensurate with the novel algorithms we will introduce in Sections 3 and 4.

2.3 Calibrated Learning Procedures

While BayesCG has several numerically appealing properties, as already mentioned it is *poorly calibrated*. In particular the uncertainty quantification provided is conservative, meaning that the posterior covariance $\bar{C}_m$ is typically much wider than the error $\bar{v}_m - v^\star$.

This is due to an incorrect application of Bayesian inference in constructing the posterior. It can be shown (Golub and Van Loan, 2013, Section 11.3.3) that (when the initial guess for the solution is zero) the CG directions form a basis of the Krylov subspace

$$K_m(G, b) = \mathrm{span}(b, Gb, G^2 b, \ldots, G^{m-1} b)$$
$$= \mathrm{span}(Gv^\star, G^2 v^\star, \ldots, G^m v^\star).$$

---

[2]Often these are instead obtained from the Lanczos algorithm, which provides directions that span the same space but have different orthogonality properties.

If we ignore the orthogonalisation of the search directions (which would not affect the posterior in a Bayesian inference problem) this results in information of the form $z_i = s_i^\top G v^\star = (v^\star)^\top G^{i+1} v^\star$, $i = 1, \ldots, m$, each of which is quadratic in $v^\star$ rather than linear. Linearity of the information z_i is intrinsic to the Gaussian conditioning argument that underpins PLSs, and ignoring this yields the poor calibration of BayesCG-based PLSs.

In this paper we will discuss how another class of PLSs can be used in the CAGP framework. These PLSs do not have a Bayesian interpretation, but can nevertheless be said to be *calibrated* in a formal sense. We will be interested in the question of whether, when the PLS is calibrated, the derived CAGP is also calibrated, so we present this rather generically. Cockayne et al. (2022) introduces the notion of *strong calibration*, a more intuitive description of which is given below; we refer the reader to the aforementioned paper for a formal introduction.

Briefly, a learning procedure $\mu : \mathcal{P}(\mathcal{U}) \times \mathcal{Y} \to \mathcal{P}(\mathcal{U})$ is said to be strongly calibrated with respect to a distribution μ_0 and a data-generating model $\textsc{dgm} : \mathcal{U} \to \mathcal{Y}$ if, under the following procedure:

1. $u^\star \sim \mu_0$

2. $y = \textsc{dgm}(u^\star)$

it holds that, on average over μ_0, $u^\star$ is a "plausible sample" from the posterior $\mu(\mu_0, y)$. The last statement can be made formal in several ways, but a particularly simple definition for the case of Gaussian learning procedures is given by Cockayne et al. (2021).

Definition 1 (Cockayne et al. (2021, Definition 6 and 9)). *Consider a fixed Gaussian prior $\mu_0 = \mathcal{N}(u_0, \Sigma_0)$, a data-generating model* $\textsc{dgm}$ *and a learning procedure $\mu(\mu_0, y) = \mathcal{N}(\bar{u}, \bar{\Sigma})$. Suppose that $\bar{\Sigma}$ is independent of y and potentially singular, with N and R matrices whose columns form (mutually) orthonormal bases of its null and row spaces respectively. Then μ is said to be* strongly calibrated *to $(\mu_0, \textsc{dgm})$ if:*

1. $(R^\top \bar{\Sigma} R)^{-\frac{1}{2}} R^\top (\bar{u} - u^\star) \sim \mathcal{N}(0, I)$.

2. $N^\top (\bar{u} - u^\star) = 0$

when $u^\star \sim \mu_0$.

Note that in Proposition 1 the randomisation of $u^\star$ induces randomness in $\bar{u}$ through dependence on y, making the statement rather nontrivial. The assumption that $\bar{\Sigma}$ is independent of y is important to ensure that the range and null spaces are consistent across draws from μ_0; this can be relaxed, but resulting definitions

are far less analytically tractable[3]. Lastly in the case that $\bar{\Sigma}$ is full rank, the second of the two conditions is redundant.

Proposition 1 provides a theoretical framework for validating strong calibration, but we are also interested in validating this numerically. To accomplish this, we will apply the simulation-based calibration tests of Talts et al. (2018); these are described in more detail in Section S2.

2.4 Probabilistic Stationary Iterative Methods

Another subclass of PLSs are PSIMs, introduced in Cockayne et al. (2021)[4]. PSIMs are based on an underlying *stationary iterative method* for solving the linear system (see e.g. Young (1971)), that is, methods that evolve an iterate v_m according to the map $v_m = P(v_{m-1})$, with some user-supplied initial guess v_0. Given such a method, the associated PSIM is obtained by pushing the prior η_0 through the map P^m defined by composing P with itself m times. We will restrict attention to affine P, i.e. $P(v) = Mv + g$, where $M \in \mathbb{R}^{d \times d}$ while $g \in \mathbb{R}^d$. Then, for Gaussian $\eta_0 = \mathcal{N}(0, G^{-1})$ the output of the PSIM is given by

$$\eta_m = \mathcal{N}(v_m, C_m) \tag{10a}$$

$$v_m = P^m(0) = \sum_{i=0}^{m-1} M^i g \tag{10b}$$

$$C_m = M^m G^{-1} (M^\top)^m. \tag{10c}$$

Since the output of a PSIM is not a Bayesian posterior, several important questions arise: (i) when does the posterior contract around the truth, and (ii) is the output calibrated in the sense of Proposition 1. For (i), Cockayne et al. (2021, Proposition 2) establishes that provided the underlying stationary iterative method converges to the true solution[5], the PSIM contracts around the true solution, and does so at the same rate as the error converges in any norm on $\mathbb{R}^d$. For (ii), Cockayne et al. (2021, Propositions 7 and 10) show that any PSIM based on appropriate affine P is strongly calibrated in the sense of Proposition 1, provided M is diagonalisable (over $\mathbb{C}$). This is a fairly mild restriction considering the density of diagonalisable matrices (Gorodentsev, 2017, Chapter 2).

---

[3]Note however that this precludes selecting hyperparameters using empirical Bayesian procedures.

[4]These were originally termed "probabilistic iterative methods", but we adopt different nomenclature to avoid confusion (since the methods described in Section 2.2.1 are also both probabilistic and iterative).

[5]This is not guaranteed; they converge to the truth only when the spectral radius of M is below 1 and if they are *completely consistent* (see Section 4.1).

In the next sections we will demonstrate how PSIMs can be adapted to work with CAGPs, and discuss transfer of calibratedness in this setting.

3 CALIBRATED COMPUTATION-AWARE GPs

In this section we prove that CAGPs are calibrated if a calibrated PLSs is used to solve Eq. (3). First we observe that the reparameterisation in Eq. (2c) can be formulated as Bayesian inference with an alternative observation model.

Proposition 2. *It holds that $f \mid y$ from Eq. (2c) is equal to the posterior $f \mid v$ from Bayesian inference under the observation model:*

$$v \mid f = G^{-1}b = G^{-1}(f(X) - m_0(X)) + \bar{\zeta}$$

where $\bar{\zeta} \sim \mathcal{N}(0, \sigma^2 G^{-2})$.

The next corollary establishes that under the observation model used in Proposition 2, the prior distribution adopted in CAGPs is correct in a subjective Bayesian sense.

Corollary 3. *The a-priori marginal distribution of v is $v \sim \mathcal{N}(0, G^{-1})$.*

This is an important result for calibration, since to be able to talk about calibrated posteriors for Eq. (3) we first need to know that if the prior on f is correct, the prior on v is also correct.

The next result is the central result of the paper, establishing calibratedness of CAGPs when a calibrated PLS is used.

Theorem 4. *Suppose that $f \sim \mathcal{GP}(m_0, k)$ and $y \mid f = f(X) + \zeta$, where k is a positive definite kernel. Further, suppose that the PLS in Eq. (5) is a learning procedure that is calibrated for $\mathcal{N}(0, G^{-1})$, and satisfies the following conditions:*

1. *D is independent of y.*

2. *$\mathrm{Cov}(f(X'), \bar{v}) = K(X', X)G^{-1}\mathrm{Cov}(y, \bar{v})$.*

Then the CAGP posterior is calibrated for the original GP prior and the original data generating model.

Regarding the conditions above, as mentioned previously, Condition 1 is satisfied by most PLSs, with the notable exception of those where some calibration procedure has been applied to choose the prior. Condition 2 seems technical, but is in fact satisfied under mild conditions.

Corollary 5. *Suppose that the assumptions of Proposition 4 are satisfied, and further that $\bar{v}$ is an affine map of y, i.e. $\bar{v} = My + g$. Then the CAGP posterior is calibrated for the original GP prior and original data generating model.*

Note that the requirements of Proposition 5 are satisfied by any Bayesian procedure, as well as for the probabilistic iterative methods we will introduce in the next section. An important exception is CAGP-CG since, as mentioned in Section 2.2.1, in this case the map is not affine owing to dependence of S_m on $v^\star$. Having established this result, we next proceed to show how PSIMs can be integrated with CAGPs to provide calibrated uncertainty.

4 PROBABILISTIC STATIONARY ITERATIVE METHODS WITH THE INVERSE PRIOR

To embed a probabilistic stationary iterative method in a CAGP we need to obtain an output measure as in Eq. (5). We now show that for all reasonable PSIMs, C_m from Eq. (10c) can be rearranged to have this structure.

4.1 Convergent Linear Stationary Iterative Methods

As mentioned in Section 2.4, convergence of stationary iterative methods is not guaranteed for all affine maps P. We therefore limit attention to *completely consistent* methods (Young, 1971, Section 3.2 and 3.5). These methods have the property that, if they converge, they are guaranteed to converge to the true solution, and therefore they are the widest class of reasonable stationary iterative methods to use for solving a linear system. For nonsingular G, any completely consistent iterative method can be written in the form $M = I - \tilde{M}G$, $g = \tilde{M}b$ where $\tilde{M}$ is a nonsingular matrix. The next proposition shows that the iterations of any PSIM based on a completely consistent iterative method can be written in a form commensurate with CAGPs.

Proposition 6. *Let η_m be the mth iterate of a probabilistic iterative method whose underlying stationary iterative method is completely consistent. Then $\eta_m \sim \mathcal{N}(\bar{v}_m, G^{-1} - D_m)$, where $\bar{v}_m$ is the mth iterate of the stationary iterative method with $v_0 = 0$, while $D_0 = 0$ and*

$$D_m = \tilde{M} + \tilde{M}^\top - \tilde{M}G\tilde{M}^\top + (I - \tilde{M}G)D_{m-1}(I - \tilde{M}G)^\top. \tag{11}$$

While it is useful to know that this holds for any completely consistent probabilistic iterative method, it is not clear that the above computations can be made to

be efficient. Moreover, we still need to ensure that the iterative method converges, as it will be highly problematic for embedding within GP regression otherwise. In the next section we consider a particular instance of a probabilistic iterative method which is provably convergent for any symmetric positive definite matrix.

4.2 Gauss-Seidel

The Gauss-Seidel method partitions $G = L + U$, where L is the lower-triangular part of G and U is the strict upper triangular part. We then take $M = -L^{-1}U$ and $g = L^{-1}b$. Golub and Van Loan (2013, Theorem 11.2.3) establishes that Gauss-Seidel converges to the true solution for any initial guess v_0 provided G is symmetric positive-definite, making it particularly attractive for CAGPs.

To apply Proposition 6 we must first identify $\tilde{M}$. Note that $U = G - L$, so that $M = -L^{-1}U = I - L^{-1}G$; thus $\tilde{M} = L^{-1}$. Also note that if D_G is the diagonal of G, since G is symmetric positive definite, we have that $U = (L - D_G)^\top$. The D_i from Proposition 6 can then be simplified:

$$L^{-1}GL^{-\top} = L^{-1}(L + L^\top - D_G)L^{-\top}$$
$$= L^{-1} + L^{-\top} - L^{-1}D_GL^{-\top}$$

so that $D_1 = L^{-1}D_GL^{-\top}$, and

$$D_i = L^{-1}D_GL^{-\top} + MD_{i-1}M^\top$$
$$= L^{-1}D_GL^{-\top} + L^{-1}UD_{i-1}U^\top L^{-\top}.$$

We therefore have the following non-recursive expression for the downdate:

$$D_m = \sum_{i=0}^{m-1}(L^{-1}U)^i L^{-1}D_GL^{-\top}(U^\top L^{-\top})^i. \quad (12)$$

We can also establish several important properties of this PLS, in the following proposition:

Proposition 7. *The covariance matrix $G^{-1} - D_m$ from probabilistic Gauss-Seidel has rank $d-1$ for all $m \geq 1$, and its null space is equal to $\text{span}(L^\top e_d)$, where e_d is the dth Euclidean basis vector.*

Considering Section 2 we can therefore implement CAGP_PLS with Gauss-Seidel as described in Algorithm 2.

4.2.1 Complexity

Since L^{-1} is lower triangular, the action of L^{-1} and $L^{-\top}$ can be computed using forward and back substitution, having complexity $\mathcal{O}(d^2)$, where d is the size of the training data. With for m iterations and d_{TEST}

Algorithm 2 CAGP-GS

1: **function** CAGP_PLS_GS(G, b, V, m)
2:     $Z_1 = L^{-\top}V^\top$
3:     $z = L^{-1}b$
4:     $v_1 = z$
5:     $\tilde{Z}_1 = D_G^{\frac{1}{2}}Z_1$
6:     **for** $i = 2$ to m **do**
7:         $v_i = z - L^{-1}Uv_{i-1}$
8:         $Z_i = L^{-\top}U^\top Z_{i-1}$
9:         $\tilde{Z}_i = D_G^{\frac{1}{2}}Z_i$
10:     **return** $\tilde{v} = Vv_i$, $\tilde{D} = \sum_{i=1}^m \tilde{Z}_i^\top \tilde{Z}_i$
---

testing points, the computational complexity of Algorithm 2 is thus $\mathcal{O}(mdd_{\text{TEST}}(d + d_{\text{TEST}}))$.

As mentioned in Section 2.2.1, the cost of CAGP-CG is $\mathcal{O}(md^2)$ to compute the posterior over v. Once this has been computed, computing the implied $\tilde{v}$ and $\tilde{D}$ costs, respectively $\mathcal{O}(dd_{\text{TEST}})$ and $\mathcal{O}(md_{\text{TEST}}(d + d_{\text{TEST}}))$, for an overall complexity of $\mathcal{O}(m(d^2 + dd_{\text{TEST}} + d_{\text{TEST}}^2))$.

In terms of memory, for CAGP-GS we need to store the matrix Z_i for complexity $\mathcal{O}(dd_{\text{TEST}})$; computed factors can be saved to disk and loaded later to compute $\tilde{D}$ (which is only $d_{\text{TEST}} \times d_{\text{TEST}}$). For CAGP-CG only $\mathcal{O}(d)$ memory is required at execution time. This ignores storage of G (required for both algorithms); however since each algorithm requires only the action of G (or L, U) on matrices / vectors, a matrix-free implementation is possible (though this is not explored in this paper). Also note that this highlights that to apply CAGP-CG on a new set of test points does not require rerunning the algorithm, while for CAGP-GS we need to do so.

Clearly CAGP-GS has a higher complexity than CAGP-CG, though under the assumption that $d \gg m, d_{\text{TEST}}$ the leading order of both algorithms is $\mathcal{O}(d^2)$, so in this setting the costs should still be comparable. In the next section we will consider the empirical performance of CAGP-GS compared to CAGP-CG.

5 EXPERIMENTS

A Python implementation of Algorithm 2 is available at https://github.com/hegdedisha/CalibratedCAGP.

5.1 Synthetic Problem

We first consider a synthetic test problem, so that we can test for calibratedness. We take $m_0 = 0$ and k to be a Matèrn $^3/_2$ covariance with amplitude set to 1, and will vary the length-scale. For the data-generating model we set $\sigma = 0.1$. We set our domain to be $D = $
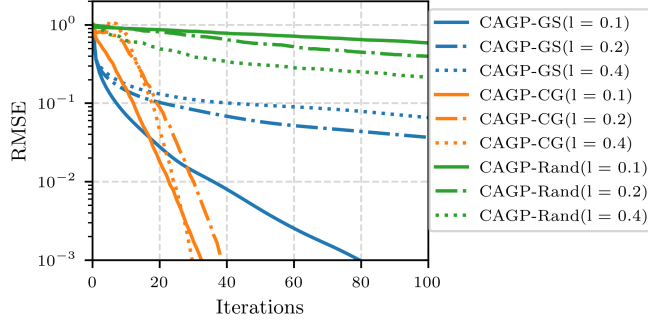
Figure 1: RMSE for the synthetic problem in Section 5.1.

Figure 2: NLL for the synthetic problem in Section 5.1.

$[0, 1]^2$ and generate our training points by sampling 400 points uniformly at random. Test points are a regular grid with spacing 0.05, i.e. generating 21×21 equally spaced points for a total of $d = 441$ points. The underlying true function is taken to be sample from the prior, so that the calibration guarantee from Proposition 5 can be tested.

Plots of root mean square error (RMSE) of the posterior mean as a function of m, with length scales set to $0.1, 0.2$ and 0.4, averaged over 50 runs, can be seen in Fig. 1. We compare CAGP-GS to both CAGP-CG and CAGP-Rand, a method that uses a Bayesian PLS with S_i having IID normal entries. This approach should be calibrated, but shows slow convergence typical of most Bayesian PLSs not using CG directions. Unexpectedly, the posterior mean for CAGP-GS initially converges *faster* than that from CAGP-CG for all length-scales, though ultimately it is overtaken by CG. However, in expensive problems where few iterations can be performed, this suggests that CAGP-GS should be preferred to CAGP-CG owing to its initially faster convergence with calibration guarantees. As a function of the length-scale, it appears that this behaviour is less pronounced for smaller values, in which cases the matrix will typically be better conditioned.

The negative log likelihood (NLL) plot for the same problem in Fig. 2 shows the added advantage of calibration for CAGP-GS. CAGP-GS outperforms CAGP-CG in NLL for more number of iterations than it does in RMSE, due to better calibrated posterior.

In Fig. 3 we highlight the uncertainty quantification properties of CAGP-GS compared to CAGP-CG and CAGP-Rand using the simulation-based calibration method described in Algorithm S1 for $N_{\text{SIM}} = 1000$ simulations, with length-scale now fixed to $\ell = 0.2$ and $m = 5$ iterations. We also report the result of a Kolmogorov-Smirnov test for uniformity. As implied by Proposition 4, both CAGP-GS and CAGP-Rand are calibrated (p-values 0.6689 and 0.6802), while
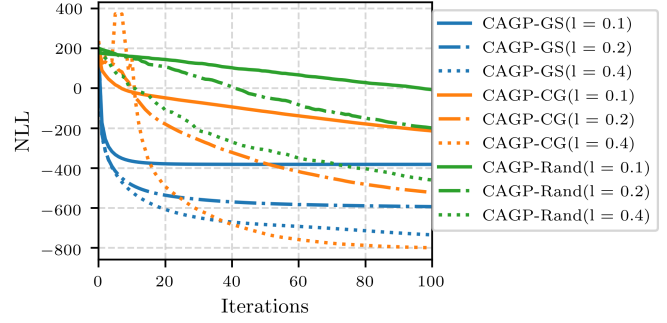
CAGP-CG shows the expected inverted U-shape characteristic of an overly conservative posterior and has a p-value of 0.0084.

5.2 UCI Benchmark Datasets

In this section we compare the performance of CAGP-GS and CAGP-CG on three datasets from the UCI Machine Learning Repository: Concrete Compressive Strength dataset (Yeh, 1998), Bike Sharing dataset (Fanaee-T, 2013) and 3D Road Network dataset (Kaul, 2013). These have varying number of explanatory variables and varying size of the dataset

As seen in the previous results, Fig. 4 shows that CAGP-GS outperforms CAGP-CG in RMSE for initial iterations. And due to better calibration, CAGP-GS also outperforms CAGP-CG in NLL for higher iteration numbers the last two datasets (bike sharing and 3D road network). Additional improvement in terms of NLL is not seen for the first dataset, and we speculate that this could be because of a small number of data points. Since we not "in-model" as in Section 5.1, the small data size could possibly lead to model mismatch and, thus, suboptimal performance in NLL. However, there is still a sizeable region where CAGP-GS outperforms CAGP-CG in NLL.

5.3 ERA5 Regression

In this section we solve a geospatial regression problem on the ERA5 global 2 metre temperature dataset (Hersbach et al., 2023). This is a reanalysis dataset with approximately 31 km resolution, and has temporal coverage from 1940 to present. For the purposes of this demonstration we limit our attention to a single timestamp on 1st January 2024 at 00.00. This results in a grid of a total of ≈ 1 million points. To obtain matrices that can be more easily represented in memory these points were downsampled as described in the results below.

To fix a prior we used a Matèrn $^3/_2$ covariance func-
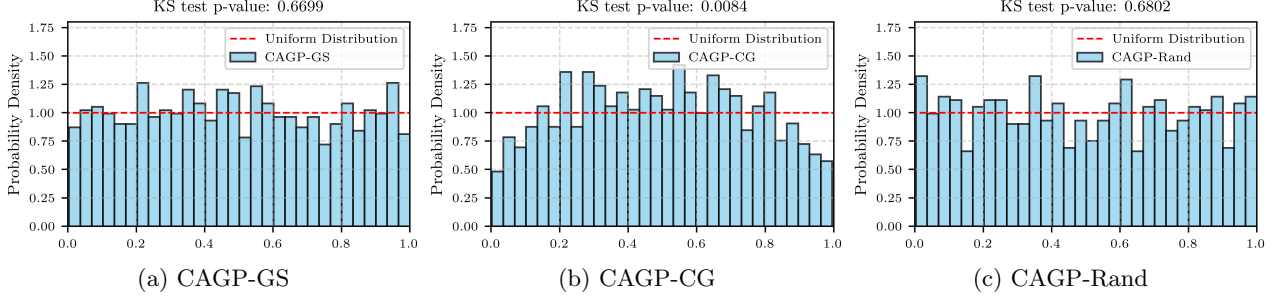
(a) CAGP-GS (b) CAGP-CG (c) CAGP-Rand

Figure 3: Simulation-based calibration results for the synthetic problem described in Section 5.1 after $m = 5$ iterations.
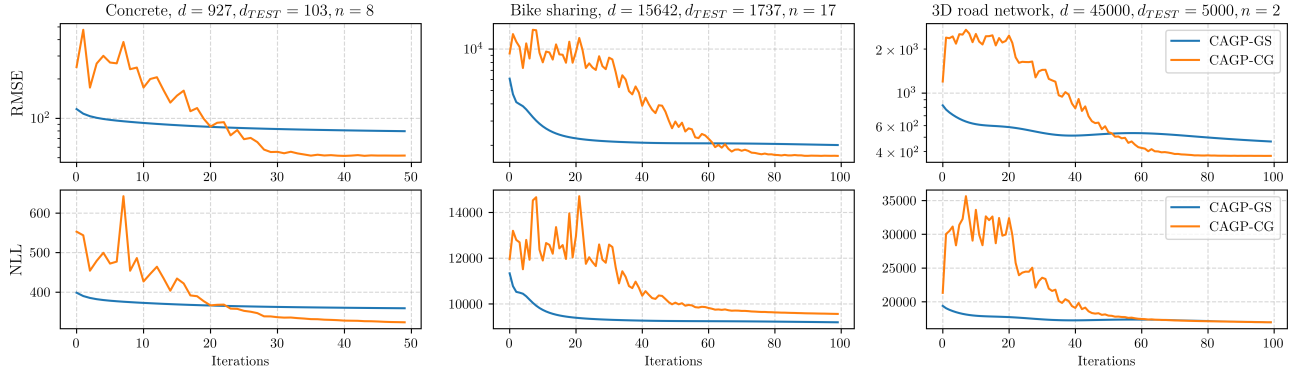

Figure 4: RMSE and NLL for different datasets in Section 5.2

tion, and optimised hyper-parameters by maximising marginal likelihood (see Rasmussen and Williams (2005, Sec. 2.2)) on a coarse uniform grid of 72×144 points on the globe. We use a constant prior mean fixed to the average of the data points.

We first examine qualitative convergence of the method by presenting plots of the posterior mean obtained from CAGP-GS and CAGP-CG for a uniformly spaced grid of $28,800$ training and $7,225$ test points in Fig. 5. Interestingly the CAGP-GS posterior means are considerably smoother than the CAGP-CG posterior means; this smoother convergence may be more desirable in low iteration number regimes, and is likely due to known smoothing properties of Gauss-Seidel (see e.g. Xu and Zikatanov (2017, Section 5.5)).

In Fig. S4, we compare the wall-time taken to compute the posterior mean and covariance using CAGP-GS and CAGP-CG. In each plot we vary one parameter, fixing the others to $m = 80, d = 10638, d_{\text{TEST}} = 25$. Results were computed on a high-performance computing service on a single node with 40 cores. The results are mostly as expected from Section 4.2.1. With increasing iterations, CAGP-GS and CAGP-CG scale similarly, with CAGP-GS performing slightly better for higher iterations. With increasing d CAGP-CG is initially faster, but CAGP-GS takes over in the later

part. For small d_{TEST} CAGP-GS performs better, but CAGP-CG improves as d_{TEST} increases and becomes comparable to d.

Finally, we test calibratedness for a uniformly spaced grid of $d = 64,700$ training points and $d_{\text{TEST}} = 100$ held out test points. Note that in this setting Proposition 4 does not provide a calibration guarantee, and in particular our theory says nothing about calibratedness on held-out data. Nevertheless, it is interesting to see whether some version of calibratedness is obtained in a practical setting. For this d, G requires around 30 GB of storage, so is around the largest that can practically be considered without matrix-free implementations. Histograms of $t_i = \Phi \left(\frac{\tilde{m}(x_i) - f(x_i)}{sd(x_i)} \right)$ for test points x_i are shown in Fig. S5. The Kolmogorov-Smirnov test for uniformity gives a p-value of 0.4707 for CAGP-GS and 5.0112×10^{-1} CAGP-CG. This shows that CAGP-GS is closer to calibrated, and CAGP-CG remains miscalibrated.

6 CONCLUSION

The theoretical and computational results presented above provide a clear motivation for considering PLSs other than BayesCG for CAGPs. Mean convergence is faster for small m as shown in Fig. 1, and the pos-
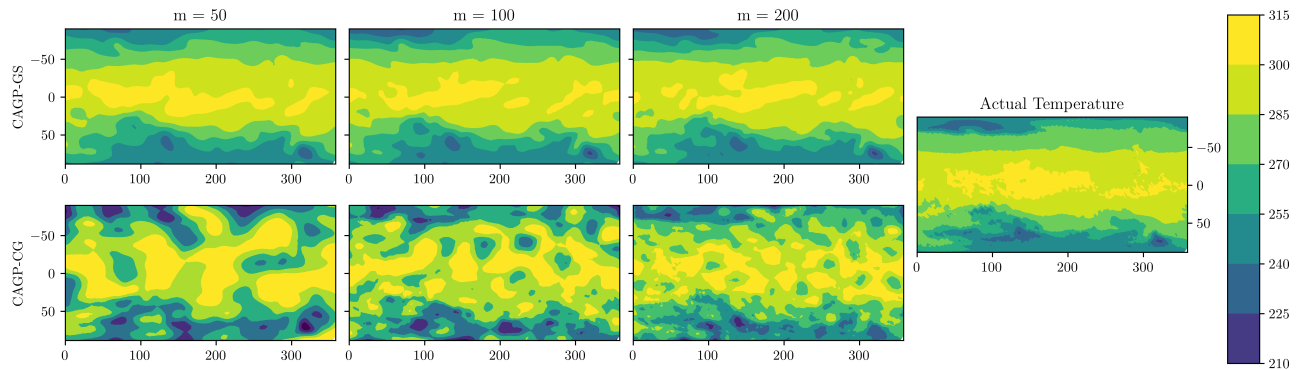
Figure 5: Sequences of posterior means for the large-scale regression from Section 5.3, as a function of m. The x and y axes represent longitudes and latitudes respectively, and the contours indicate the temperature in Kelvin.

terior mean plots in Fig. 5 show recoveries that, we would argue, retain more of the smoothness and structure of the prior than those for CAGP-CG. Further, with an appropriate GP calibration approach it appears we can obtain reasonably well-calibrated CAGPs even outside of the synthetic calibration guarantees of Proposition 4.

The principle downsides of CAGP-GS are (i) higher computational complexity and (ii) worse scaling for large m. On (i) we would argue that this approach should only be used in a small d_{TEST} regime which, in very large scale regression problems, is likely to be a limitation in any case. For (ii) we would argue that small m is the regime in which well-calibrated CAGPs are most attractive, since for larger m the added computational uncertainty is dominated by the mathematical uncertainty. Moreover, we feel that the calibration benefits are enough to justify these disadvantages.

There are several interesting future research directions. First, we have not explored matrix-free methods to scale to "big data" problems. Efficient, parallelisable matrix-free implementations of $L^{-\top}$ are challenging due to the inherently sequential nature of backward substitution, but would nevertheless allow further scaling. Second, we would like to explore further accelerations of calibrated CAGPs. One avenue that seems promising is to combine CAGP-GS and CAGP-CG; this would sacrifice calibratedness but, if GS is either used in the initial convergence period or interleaved with CG iterations to provide smoothing, this may result in superior mean convergence.

Acknowledgements

References

Bartels, S., Cockayne, J., Ipsen, I. C. F., and Hennig, P. (2019). Probabilistic linear solvers: a unifying view. *Statistics and Computing*, 29(6):1249–1263.

Cockayne, J., Graham, M. M., Oates, C. J., Sullivan, T. J., and Teymur, O. (2022). Testing whether a learning procedure is calibrated. *Journal of Machine Learning Research*, 23(203):1–36.

Cockayne, J., Ipsen, I. C., Oates, C. J., and Reid, T. W. (2021). Probabilistic iterative methods for linear systems. *Journal of Machine Learning Research*, 22(232):1–34.

Cockayne, J., Oates, C. J., Ipsen, I. C. F., and Girolami, M. (2019a). A Bayesian conjugate gradient method (with discussion). *Bayesian Anal.*, 14(3):937–1012. Includes 6 discussions and a rejoinder from the authors.

Cockayne, J., Oates, C. J., Sullivan, T. J., and Girolami, M. (2019b). Bayesian probabilistic numerical methods. *SIAM Review*, 61(3):756–789.

Fanaee-T, H. (2013). Bike sharing. UCI Machine Learning Repository. Licensed under CC BY 4.0.

Ferrari-Trecate, G., Williams, C., and Opper, M. (1998). Finite-dimensional approximation of gaussian processes. In *Advances in Neural Information Processing Systems*, volume 11.

Golub, G. H. and Van Loan, C. F. (2013). *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, 4 edition.

Gorodentsev, A. L. (2017). *Algebra II: Textbook for Students of Mathematics*. Springer International Publishing.

Hennig, P., Osborne, M. A., and Kersting, H. P. (2022). *Probabilistic Numerics: Computation as*

Machine Learning. Cambridge University Press, Cambridge.

Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., and Thépaut, J.-N. (2023). ERA5 hourly data on single levels from 1940 to present. Accessed on 24-09-2024. Licence permits free use for any lawful purpose; see https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels for more details.

Kaul, M. (2013). 3D Road Network (North Jutland, Denmark). UCI Machine Learning Repository. Licensed under CC BY 4.0.

Pförtner, M., Wenger, J., Cockayne, J., and Hennig, P. (2024). Computation-aware Kalman filtering and smoothing.

Rasmussen, C. E. and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning.* The MIT Press.

Reid, T. W., Ipsen, I. C. F., Cockayne, J., and Oates, C. J. (2022). BayesCG as an uncertainty aware version of CG.

Reid, T. W., Ipsen, I. C. F., Cockayne, J., and Oates, C. J. (2023). Statistical properties of BayesCG under the Krylov prior. *Numer. Math.*, 155(3-4):239–288.

Talts, S., Betancourt, M., Simpson, D., Vehtari, A., and Gelman, A. (2018). Validating bayesian inference algorithms with simulation-based calibration.

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In van Dyk, D. and Welling, M., editors, *Artificial intelligence and statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 567–574.

Wenger, J. and Hennig, P. (2020). Probabilistic linear solvers for machine learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 6731–6742.

Wenger, J., Pleiss, G., Hennig, P., Cunningham, J., and Gardner, J. (2022a). Preconditioning for scalable Gaussian process hyperparameter optimization. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23751–23780.

Wenger, J., Pleiss, G., Pförtner, M., Hennig, P., and Cunningham, J. P. (2022b). Posterior and computational uncertainty in Gaussian processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 10876–10890.

Xu, J. and Zikatanov, L. (2017). Algebraic multigrid methods. *Acta Numerica*, 26:591–721.

Yeh, I.-C. (1998). Concrete compressive strength. UCI Machine Learning Repository. Licensed under CC BY 4.0.

Young, D. M. (1971). *Iterative Solution of Large Linear Systems.* Academic Press.

Checklist

1. For all models and algorithms presented, check if you include:

   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]

   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]

   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No] Code will be made available for the camera-ready submission

2. For any theoretical claim, check if you include:

   (a) Statements of the full set of assumptions of all theoretical results. [Yes]

   (b) Complete proofs of all theoretical results. [Yes] All the proofs can be found in the Supplementary materials.

   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:

   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [No] Source of data is provided and code will be made available for the camera-ready submission

   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes] Existing data is used in Section 5.3 and creator is cited.

   (b) The license information of the assets, if applicable. [Yes]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]

   (d) Information about consent from data providers/curators. [Yes]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Calibrated Computation-Aware Gaussian Processes: Supplementary Materials

S1   Proofs of Theoretical Results

Proof of Proposition 2. This is demonstrated by direct calculation. We have that $f(X')$ and v are jointly Gaussian:

$$\begin{bmatrix} f(X') \\ v \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m_0(X') \\ 0 \end{bmatrix}, \begin{bmatrix} k(X', X') & k(X', X)G^{-1} \\ G^{-1}k(X, X') & G^{-1}k(X, X)G^{-1} + \sigma^2 G^{-2} \end{bmatrix} \right).$$

Clearly

$$G^{-1}k(X, X)G^{-1} + \sigma^2 G^{-2} = G^{-1}(k(X, X) + \sigma^2 I)G^{-1}$$
$$= G^{-1}$$

and so

$$\begin{bmatrix} f(X') \\ v \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m_0(X') \\ 0 \end{bmatrix}, \begin{bmatrix} k(X', X') & k(X', X)G^{-1} \\ G^{-1}k(X, X') & G^{-1} \end{bmatrix} \right).$$

Applying the Gaussian conditioning formula we obtain

$$\mathbb{E}[f(X') \mid v] = m_0(X') + k(X', X)G^{-1}Gv$$
$$= m_0(X') + k(X', X)v$$
$$\mathbb{V}[f(X') \mid v] = k(X', X') - k(X', X)G^{-1}GG^{-1}k(X, X')$$
$$= k(X', X') - k(X', X)G^{-1}k(X, X')$$

as required. □

Proof of Proposition 3. This can be verified by inspection of the joint distributions in the proof of Proposition 2. □

Proof of Proposition 4. To prove this we use the definition of calibratedness from Proposition 1. First let $\tilde{C} = \tilde{k}(X', X')$ and $\bar{C} = \bar{k}(X', X')$. We will similarly abbreviate $\tilde{m} = \tilde{m}(X')$, $\bar{m} = \bar{m}(X')$ and $f = f(X')$.

We first establish that $\tilde{C}$ is full rank; this is trivial since $\tilde{C} = \bar{C} + k(X', X)C_m k(X, X')$, where the latter term is positive semidefinite owing to positive semidefiniteness of C_m. Thus, $\tilde{C} \succeq \bar{C}$, and since $\bar{C}$ is positive definite, $\tilde{C}$ is full rank. For the purposes of checking calibration of the GP, we therefore do not need to worry about range and null spaces of $\tilde{C}$, so the quantity of interest is:

$$\tilde{C}^{-\frac{1}{2}}(\tilde{m} - f) = \tilde{C}^{\frac{1}{2}}\tilde{C}^{-1}(\tilde{m} - f) \tag{1}$$
$$= \tilde{C}^{\frac{1}{2}}\tilde{C}^{-1}(\bar{m} - f) + \tilde{C}^{\frac{1}{2}}\tilde{C}^{-1}k(X', X)(\bar{v} - v) \tag{2}$$

Preliminary Transformations. We start by applying the matrix inversion lemma to obtain a more useful expression for $\tilde{C}$. Since $C_m = G^{-1} - D_m$ is not assumed to be full rank we let R, N be bases of its row and null spaces of C_m respectively, and such that $U = \begin{bmatrix} R & N \end{bmatrix}$ is unitary. We then have that

$$
\begin{aligned}
\tilde{C} &= \bar{C} + k(X', X)C_m k(X, X') \\
&= \bar{C} + k(X', X)UU^\top C_m UU^\top k(X, X') \\
&= \bar{C} + k(X', X)RC_m^R R^\top k(X, X')
\end{aligned}
$$

where $C_m^R = R^\top C_m R$, since $C_m N v = 0$ by definition for all v. Applying the matrix inversion lemma we get

$$
\begin{aligned}
\tilde{C}^{-1} &= \bar{C}^{-1} - \bar{C}^{-1}k(X', X)\Sigma^R k(X, X')\bar{C}^{-1} \\
&= (I - P)\bar{C}^{-1}
\end{aligned}
$$

where

$$
\begin{aligned}
P &= \bar{C}^{-1}k(X', X)\Sigma^R k(X, X') \\
\Sigma^R &= R\Sigma^{-1}R^\top \\
\Sigma &= (C_m^R)^{-1} + R^\top k(X, X')\bar{C}^{-1}k(X', X)R
\end{aligned}
$$

We also have

$$
\begin{aligned}
\tilde{C}^{-1}k(X', X) &= \left[\bar{C}^{-1} - \bar{C}^{-1}k(X', X)\Sigma^R k(X, X')\bar{C}^{-1}\right] k(X', X)UU^\top \\
&= \left[\bar{C}^{-1} - \bar{C}^{-1}k(X', X)\Sigma^R k(X, X')\bar{C}^{-1}\right] k(X', X)(RR^\top + NN^\top) \\
\Sigma^R k(X, X')\bar{C}^{-1}k(X', X)RR^\top &= R\Sigma^{-1}R^\top k(X, X')\bar{C}^{-1}k(X', X)RR^\top \\
&= R\Sigma^{-1}R^\top k(X, X')\bar{C}^{-1}k(X', X)RR^\top \\
&= R\Sigma^{-1}(\Sigma - (C_m^R)^{-1})R^\top \\
&= R(I - \Sigma^{-1}(C_m^R)^{-1})R^\top
\end{aligned}
$$

So that

$$
\begin{aligned}
\tilde{C}^{-1}k(X', X)RR^\top &= \bar{C}^{-1}k(X', X)\left[RR^\top - RR^\top + R\Sigma^{-1}(C_m^R)^{-1}R^\top\right] \\
&= \bar{C}^{-1}k(X', X)R\Sigma^{-1}(C_m^R)^{-1}R^\top
\end{aligned}
\tag{3}
$$

Mean Computation. Next we proceed to apply these results to compute the mean and covariance of Eq. (1). Note that Gaussianity is guaranteed by the fact that Eq. (1) is an linear transformation of a difference of Gaussian random vectors, since the covariance is assumed to be independent of y.

Considering the first term in Eq. (2) we see that

$$
\tilde{C}^{-1}(\bar{m} - f) = (I - P)\bar{C}^{-1}(\bar{m} - f)
\tag{4}
$$

$$
\implies \mathbb{E}\left(\tilde{C}^{-1}(\bar{m} - f)\right) = (I - P)\bar{C}^{-\frac{1}{2}}\mathbb{E}\left(\bar{C}^{-\frac{1}{2}}(\bar{m} - f)\right)
$$

$$
= 0
$$

due to the fact that the conditional GP is Bayesian and thus calibrated for the prior, by (Cockayne et al., 2022, Example 1). For the second term we have that

$$
\begin{aligned}
\tilde{C}^{-1}k(X', X)(\bar{v} - v) &= \tilde{C}^{-1}k(X', X)(RR^\top + NN^\top)(\bar{v} - v) \\
&= \tilde{C}^{-1}k(X', X)RR^\top(\bar{v} - v)
\end{aligned}
$$

since, because the PLS is calibrated, $N^\top(\bar{v} - v) = 0$. Continuing, applying Eq. (3) we have

$$
\tilde{C}^{-1}k(X', X)RR^\top(\bar{v} - v) = \bar{C}^{-1}k(X', X)R\Sigma^{-1}(C_m^R)^{-1}R^\top(\bar{v} - v)
$$

$$
\mathbb{E}\left(\tilde{C}^{-1}k(X', X)(\bar{v} - v)\right) = \bar{C}^{-1}k(X', X)R\Sigma^{-1}(C_m^R)^{-\frac{1}{2}}\mathbb{E}\left((C_m^R)^{-\frac{1}{2}}R^\top(\bar{v} - v)\right)
$$

$$
= 0
$$

again due to calibratedness of the PLS.

Variance Computation. Next, for the variance, we have that

$$\mathbb{V}(\tilde{C}^{-\frac{1}{2}}(\tilde{m}-f))=\tilde{C}^{\frac{1}{2}}\left[\underbrace{\mathbb{V}\left(\tilde{C}^{-1}(\bar{m}-f)\right)}_{(1)}+\underbrace{\mathbb{V}\left(\tilde{C}^{-1}k(X',X)(\bar{v}-v)\right)}_{(2)}\right.$$

$$\left.-2\underbrace{\mathrm{Cov}\left(\tilde{C}^{-1}(\bar{m}-f),\tilde{C}^{-1}k(X',X)(\bar{v}-v)\right)}_{(3)}\right]\tilde{C}^{\frac{1}{2}}.$$

Starting with (1), from Eq. (4) we obtain

$$\mathbb{V}(\tilde{C}^{-1}(\bar{m}-f))=(I-P)\bar{C}^{-\frac{1}{2}}\underbrace{\mathbb{V}\left[\bar{C}^{-\frac{1}{2}}(\bar{m}-f)\right]}_{=I}\bar{C}^{-\frac{1}{2}}(I-P)^{\top}$$

$$=(I-P)\bar{C}^{-1}(I-P)^{\top}$$

where the fact that $\mathbb{V}\left[\bar{C}^{-\frac{1}{2}}(\bar{m}-f)\right]=I$ is due to calibratedness. Clearly

$$(I-P)\bar{C}^{-1}(I-P)^{\top}=\bar{C}^{-1}-P\bar{C}^{-1}-\bar{C}^{-1}P^{\top}+P\bar{C}^{-1}P^{\top}$$

and

$$P\bar{C}^{-1}=\bar{C}^{-1}P^{\top}=\bar{C}^{-1}k(X',X)\Sigma^{R}k(X,X')\bar{C}^{-1}.$$

Furthermore,

$$P\bar{C}^{-1}P^{\top}=\bar{C}^{-1}k(X',X)\Sigma^{R}k(X,X')\bar{C}^{-1}k(X',X)\Sigma^{R}k(X,X')\bar{C}^{-1}$$

$$=\bar{C}^{-1}k(X',X)R\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}k(X',X)R\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}$$

$$=\bar{C}^{-1}k(X',X)R\Sigma^{-1}\left[\Sigma-(C_{m}^{R})^{-1}\right]\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}$$

$$=\bar{C}^{-1}k(X',X)\Sigma^{R}k(X,X')\bar{C}^{-1}-\bar{C}^{-1}k(X',X)R\Sigma^{-1}(C_{m}^{R})^{-1}\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}$$

and so

$$\mathbb{V}(\tilde{C}^{-1}(m-f))=\bar{C}^{-1}-\bar{C}^{-1}k(X',X)\Sigma^{R}k(X,X')\bar{C}^{-1}-\bar{C}^{-1}k(X',X)R\Sigma^{-1}(C_{m}^{R})^{-1}\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}$$

$$=\tilde{C}^{-1}-\bar{C}^{-1}k(X',X)R\Sigma^{-1}(C_{m}^{R})^{-1}\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}. \tag{5}$$

Now for (2), again applying Eq. (3)

$$\mathbb{V}\left(\tilde{C}^{-1}k(X',X)(\bar{v}-v)\right)=\mathbb{V}\left(\tilde{C}^{-1}k(X',X)(RR^{\top}+NN^{\top})(\bar{v}-v)\right)$$

$$=\mathbb{V}\left(\tilde{C}^{-1}k(X',X)RR^{\top}(\bar{v}-v)\right)$$

since $N^{\top}(\bar{v}-v)=0$ due to calibratedness of the PLS. Further we have

$$\mathbb{V}\left(\tilde{C}^{-1}k(X',X)RR^{\top}(\bar{v}-v)\right)=\mathbb{V}\left(\bar{C}^{-1}k(X',X)R\Sigma^{-1}(C_{m}^{R})^{-1}R^{\top}(\bar{v}-v)\right)$$

$$=\bar{C}^{-1}k(X',X)R\Sigma^{-1}(C_{m}^{R})^{-\frac{1}{2}}\mathbb{V}\left((C_{m}^{R})^{-\frac{1}{2}}R^{\top}(\bar{v}-v)\right)(C_{m}^{R})^{-\frac{1}{2}}\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}$$

$$=\bar{C}^{-1}k(X',X)R\Sigma^{-1}(C_{m}^{R})^{-1}\Sigma^{-1}R^{\top}k(X,X')\bar{C}^{-1}$$

where the inner variance on the second line is I again due to calibratedness. This cancels with Eq. (5), yielding

$$\mathbb{V}(\tilde{C}^{-1}(m-f))+\mathbb{V}(\tilde{C}^{-1}k(X',X)(\bar{v}-v))=\bar{C}^{-1}-\bar{C}^{-1}k(X',X)\Sigma^{R}k(X,X')\bar{C}^{-1}$$

$$=\tilde{C}^{-1}$$

so that

$$\tilde{C}^{\frac{1}{2}} \left[\mathbb{V}(\tilde{C}^{-1}(m - f)) + \mathbb{V}(\tilde{C}^{-1}k(X', X)(\bar{v} - v)) \right] \tilde{C}^{\frac{1}{2}} = I \tag{6}$$

Finally, we examine the cross covariance term (3). Clearly

$$\operatorname{Cov}\left(\tilde{C}^{-1}(\bar{m} - f), \tilde{C}^{-1}k(X', X)(\bar{v} - v) \right) = \tilde{C}^{-1}\operatorname{Cov}\left(\bar{m} - f, \bar{v} - v \right) k(X, X')\tilde{C}^{-1}.$$

and due to bilinearity of the covariance we have

$$\operatorname{Cov}\left(\bar{m} - f, \bar{v} - v \right) = \operatorname{Cov}\left(\bar{m} - f, \bar{v} \right) - \operatorname{Cov}\left(\bar{m}, v \right) + \operatorname{Cov}\left(f, v \right).$$

The last two terms can be calculated directly. Since $\bar{m}(X') = m_0(X') + k(X', X)v$ and $v = G^{-1}y$, we get

$$\operatorname{Cov}\left(\bar{m}, v \right) = k(X', X)\operatorname{Cov}\left(v, v \right)$$
$$= k(X', X)G^{-1}$$
$$\operatorname{Cov}\left(f, v \right) = \operatorname{Cov}(f, y)G^{-1}$$
$$= k(X', X)G^{-1}$$

and so

$$\operatorname{Cov}\left(\bar{m} - f, \bar{v} - v \right) = \operatorname{Cov}\left(\bar{m} - f, \bar{v} \right) - G^{-1}k(X', X) + G^{-1}k(X', X)$$
$$= \operatorname{Cov}\left(\bar{m} - f, \bar{v} \right)$$

We can also simplify this result again using the expressions for $\bar{m}$ and f. Since $\bar{m}(X') = m_0(X') + K(X', X)G^{-1}y$ we have

$$\operatorname{Cov}\left(\bar{m} - f, \bar{v} \right) = k(X', X)G^{-1}\operatorname{Cov}(y, \bar{v}) - \operatorname{Cov}(f(X'), \bar{v})$$
$$= 0$$

by condition Item 2 from the theorem.

Putting this together we obtain that

$$\tilde{C}^{\frac{1}{2}} \left[\mathbb{V}(\tilde{C}^{-1}(\tilde{m} - f)) \right] \tilde{C}^{\frac{1}{2}} = I$$

as required. □

Proof of Proposition 5. Since $\bar{v} = My + g$,

$$\operatorname{Cov}(f(X'), \bar{v}) = \operatorname{Cov}(f(X'), y)M^\top$$
$$= k(X', X)M^\top$$

and

$$\operatorname{Cov}(y, v) = \operatorname{Cov}(y, My)$$
$$= GM^\top$$

so that

$$K(X', X)G^{-1}\operatorname{Cov}(y, \bar{v}) = K(X', X)M^\top,$$

completing the proof. □

Proof of Proposition 6. We first have that $C_0 = G^{-1}$. Therefore,

$$
\begin{aligned}
C_1 &= MG^{-1}M^\top \\
&= (I - \tilde{M}G)G^{-1}(I - \tilde{M}G)^\top \\
&= G^{-1} - \tilde{M} - \tilde{M}^\top + \tilde{M}G\tilde{M}^\top
\end{aligned}
$$

so that $C_1 = G^{-1} - D_1$ where $D_1 = \tilde{M} + \tilde{M}^\top - \tilde{M}G\tilde{M}^\top$.

Proceeding inductively, suppose that $C_{i-1} = G^{-1} - D_{i-1}$. Then applying the above,

$$
\begin{aligned}
C_i &= (I - \tilde{M}G)(G^{-1} - D_{i-1})(I - \tilde{M}G)^\top \\
&= G^{-1} - \tilde{M} - \tilde{M}^\top + \tilde{M}G\tilde{M}^\top \\
&\quad - (I - \tilde{M}G)D_{i-1}(I - \tilde{M}G)^\top \\
&= G^{-1} - D_i
\end{aligned}
$$

where

$$
D_i = \tilde{M} + \tilde{M}^\top - \tilde{M}G\tilde{M}^\top + (I - \tilde{M}G)D_{i-1}(I - \tilde{M}G)^\top. \tag{7}
$$

We therefore obtain the required structure. $\qquad\square$

Proof of Proposition 7. Note that since C_m is a Gramian matrix, $\mathrm{null}(C_m) = \mathrm{null}(G^{-\frac{1}{2}}(M^m)^\top) = \mathrm{null}((M^m)^\top)$, since $G^{-\frac{1}{2}}$ is positive definite. (This follows from the fact that if $C_m v = 0$ then $\|C_m^{\frac{1}{2}}v\|_2 = 0$ for any factor $C_m^{\frac{1}{2}}$.)

For Gauss-Seidel with $m = 1$ we have that $\mathrm{rank}(M^\top) = \mathrm{rank}(U^\top L^{-1})$. From (Ipsen, 2009, Fact 6.3), the range of $U^\top L^{-\top}$ is the same as the range of $U^\top$, which is easily seen to be $\mathrm{span}(e_2, \ldots, e_d)$ thanks to the strict lower triangular structure of $U^\top$. As a result the rank of $M^\top$ is $d - 1$. Using the rank-nullity theorem, we therefore have that the null space of $M^\top$ is 1-dimensional, and it is similarly easy to see that the null space must be $\mathrm{span}(L^\top e_d)$.

Proceeding to $m = 2$, we will identify the null space of $(M^\top)^2$. Consider $(M^\top)^2 v$ for arbitrary v. Clearly $(M^\top)^2 v = 0$ if for some α either:

1. $v = \alpha L^\top e_d$ (i.e. v lies in the null space of $M^\top$).

2. $M^\top v = \alpha L^\top e_d$ (i.e. $M^\top v$ lies in the null space of $M^\top$).

Considering the latter, if $M^\top v = L^\top e_d$ then Mv is equal to the last column of $L^\top$. However since the last column of $L^\top$ is dense, it does not lie in the range of $U^\top L^{-\top}$ (since the first component is nonzero). Hence, the null space of $(M^\top)^2$ is the same as the null space of $M^\top$, and its rank is $d - 1$ using the rank-nullity theorem. Iterating this argument shows that the null space of $(M^m)^\top$ is $\mathrm{span}(L^\top e_d)$. The statement about ranks again follows from the rank-nullity theorem. This completes the proof. $\qquad\square$

S2   Simulation-Based Calibration

In this section we outline the simulation-based calibration procedure introduced in Talts et al. (2018), which can be used to test for calibratedness numerically. The approach operates on similar principles to those described in Section 2.3, but pushes samples through a test functional to produce samples whose distribution can be more easily evaluated empirically. As a result, these tests are a necessary condition for strong calibration but not a sufficient one.

Since we operate in a Gaussian framework we will use a test statistic derived from projecting the distribution through a vector $w^\top$, as the required marginal distribution is then straightforward to derive. In this setting the simulation-based calibration test reduces to that described in Algorithm S1.

Typically, we will choose w to be a random unit vector. Algorithm S1 can then be used to check whether an arbitrary Gaussian learning procedure is calibrated and, in particular, to highlight miscalibration of CAGP-CG

Algorithm S1 Simulation-Based Calibration. Φ denotes the CDF of the standard Gaussian distribution.

Require: Prior $\mu_0 = \mathcal{N}(u_0, \Sigma_0)$, data-generating model DGM, learning procedure μ, number of simulations N_{SIM}, test vector w.

1: **for** $i = 1$ to N_{SIM} **do**
2: $\quad u_i^\star \sim \mu_0$
3: $\quad y_i = \text{DGM}(u_i^\star)$
4: $\quad \mu_i = \mu(\mu_0, y_i) = \mathcal{N}(\bar{u}_i, \bar{\Sigma}_i)$
5: $\quad t_i = \Phi\left(\frac{w^\top(\bar{u}_i - u_i^\star)}{(w^\top \Sigma_i w)^{\frac{1}{2}}} \right)$
6: Plot histogram of $\{t_i\}_{i=1}^N$ and compare to $\mathcal{U}(0,1)$.

(a) RMSE (b) NLL
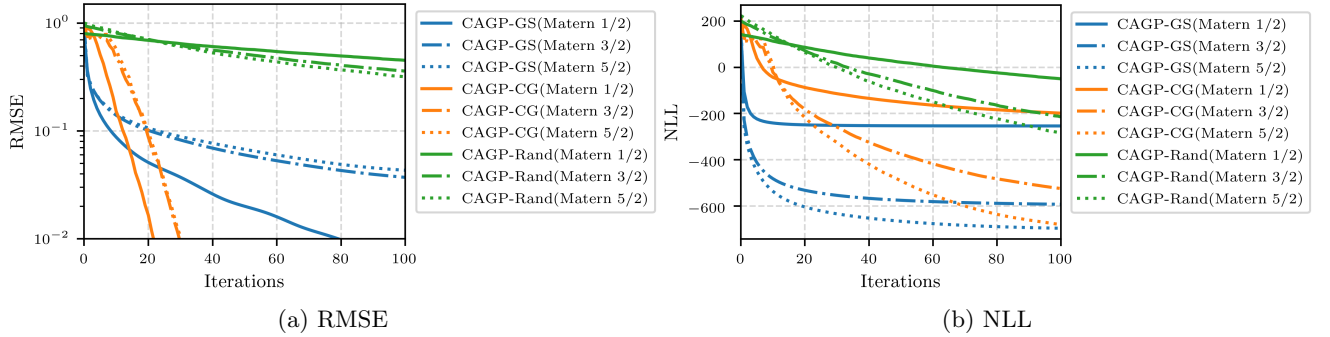
Figure S1: Convergence behaviour for the synthetic problem in Section 5.1.

in Section 5. In the event of miscalibration, the histogram may have a U-shape in the event of an overconfident posterior (i.e. the truth is typically in the tails of the learned distribution) or an inverted U-shape for a conservative posterior (i.e. the truth is typically near the modal point), though of course other shapes are possible.

S3 Further Simulation Results

S3.1 Synthetic Problem

Fig. S1 shows that the promising behaviour of computation-aware GP (CAGP)-GS root mean square error (RMSE) and negative log likelihood (NLL) extend to rougher and smoother processes. Here we plot RMSE and NLL for Matèrn $^3/_2$, $^3/_2$ and $^5/_2$ for a fixed length scale of $l = 0.2$ for the same setup in Section 5.1. CAGP-GS outperforms CAGP-CG in RMSE and NLL for initial few iterations as expected.

In Fig. S2 we show the effect of increasing number of iterations on calibration of CAGP-GS and CAGP-CG. We can see that while CAGP-GS remains calibrated (Fig. S2a), CAGP-CG becomes well-calibrated with increasing m. This is because in CAGP-CG we get linear convergence of the covariance but superlinear convergence of the mean, leading to poor calibration. Nevertheless, eventually the covariance "catches up" to the mean, and the posterior becomes calibrated for high iterations (Fig. S2c).

Another approach to reducing the cost of Gaussian process (GP) approximation is to use a subset of the training set to fit the regression model. In Fig. S3 we compare the performance of CAGP-GS and CAGP-CG with this subset GP approach, in terms of time taken to compute RMSE and NLL by increasing the number of iterations for CAGP-CG and CAGP-CG, and increasing size of a randomly chosen subset for subset GP. We use a setting where CAGP-GS is presumed to work well, with $d = 100$ and $d_{\text{TEST}} = 25$. We use the `numpy` implementation of Cholesky factorisation to compute the GP mean and posterior for the subset GP, and `jax` to jit compile our implementations of all three methods for speed. From Fig. S3a, we can see that CAGP-GS and CAGP-CG converge faster than subset GP, with CAGP-GS being the fastest for the first few iterations. With its better calibrated posterior, CAGP-GS also outperforms the other two approaches in NLL. However, it should be noted that this performance heavily depends on implementation, as well as the hardware used. For example, the `jax` Cholesky implementation is a parallelisable block algorithm, whereas our Gauss-Seidel implementation is harder
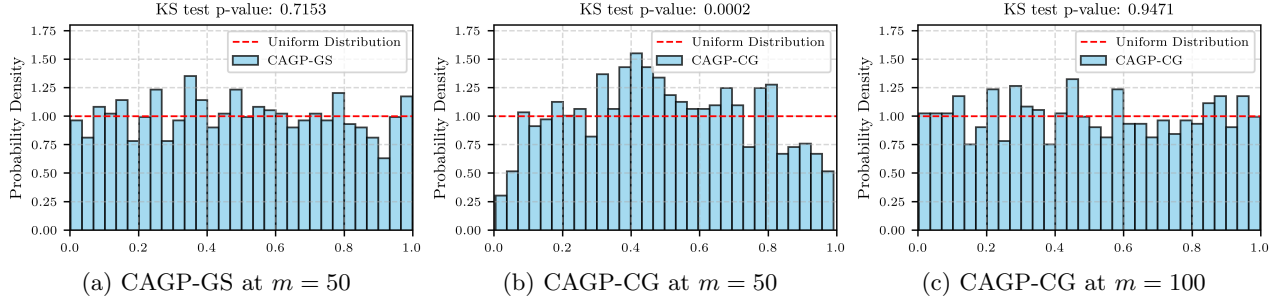
(a) CAGP-GS at $m = 50$ (b) CAGP-CG at $m = 50$ (c) CAGP-CG at $m = 100$

Figure S2: Simulation-based calibration results for the synthetic problem described in Section 5.1 for varying number of iterations

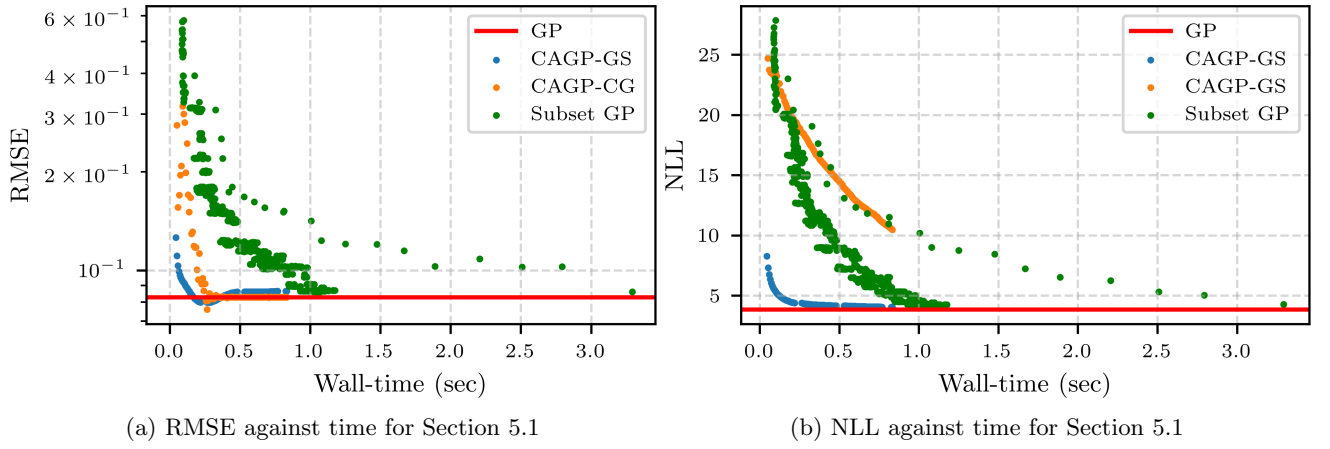(a) RMSE against time for Section 5.1   (b) NLL against time for Section 5.1

Figure S3: RMSE and NLL against time for Section 5.1

to parallelise as noted in Section 6. Nevertheless the results appeared to be similar across a range of hardware; the figures presented were from a machine equipped with an i5-1245U CPU with 10 cores, but similar performance was observed when using the University of Southampton's Iridris5 cluster on a single node with 40 tasks.

S3.2 ERA5 Regression Problem

Fig. S4 reports timings for the ERA5 regression problem from Section 5.3, while Fig. S5 shows results of the calibratedness experiment reported therein.

Fig. S6 show the RMSE and NLL for this regression problem. As in Section 5.1, CAGP-GS outperforms CAGP-CG for inital iterations in both RMSE and NLL. However, unlike the case of synthetic experiment, CAGP-GS outperforms CAGP-CG in NLL for almost the same number of iterations as RMSE, but the difference in NLL is smaller. We speculate that this is due to model mismatch, since we are outside the synthetic "in-model" setting

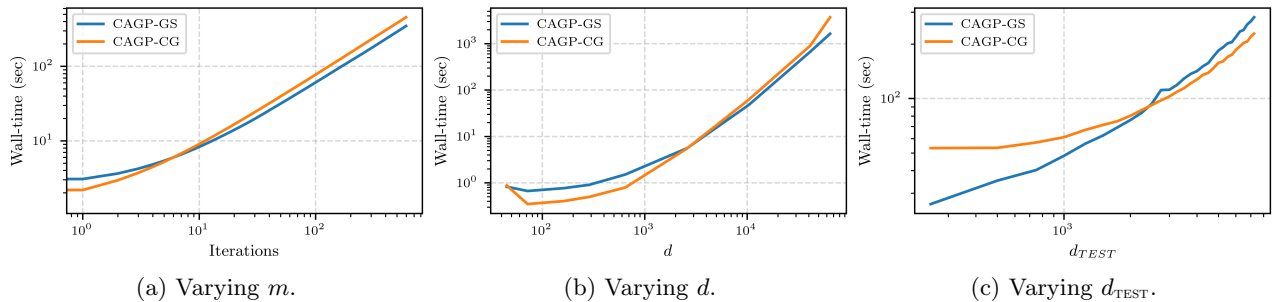(a) Varying m.   (b) Varying d.   (c) Varying d_{TEST}.

Figure S4: Timings for the large-scale regression problem from Section 5.3 as a variety of parameters are varied.
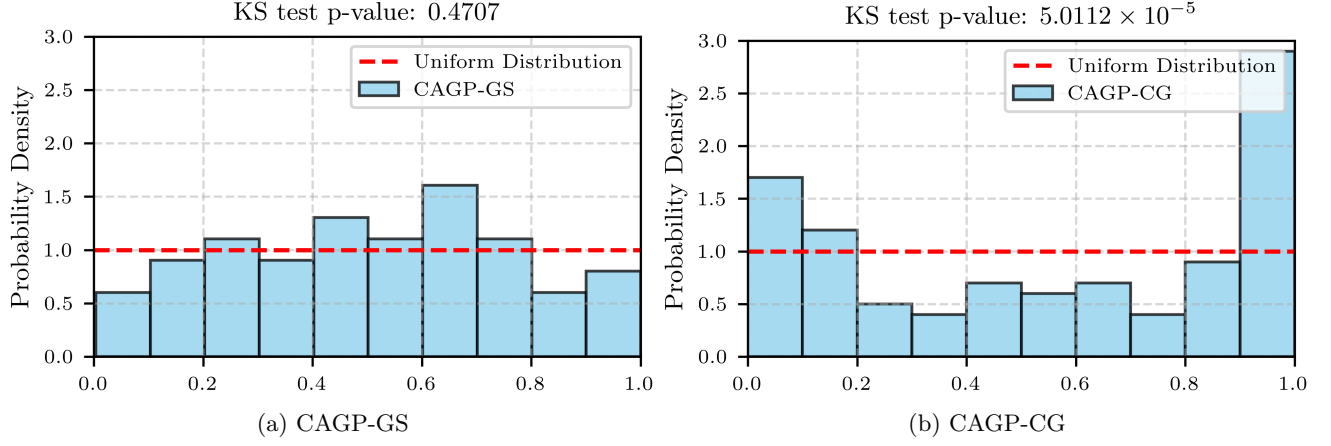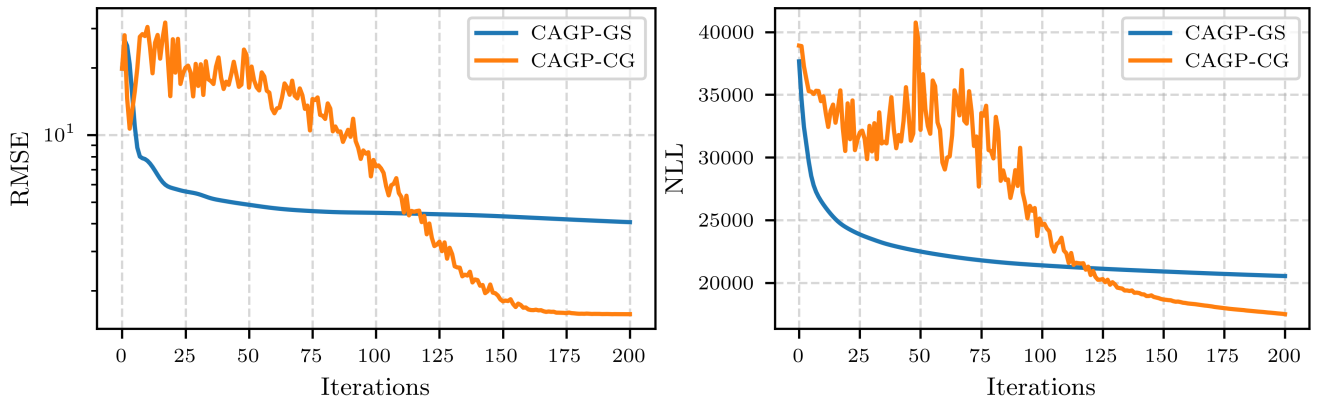
Figure S5: Calibration results for the large-scale regression problem from Section 5.3

(a) RMSE for the large-scale regression from Section 5.3

(b) NLL for the large-scale regression from Section 5.3

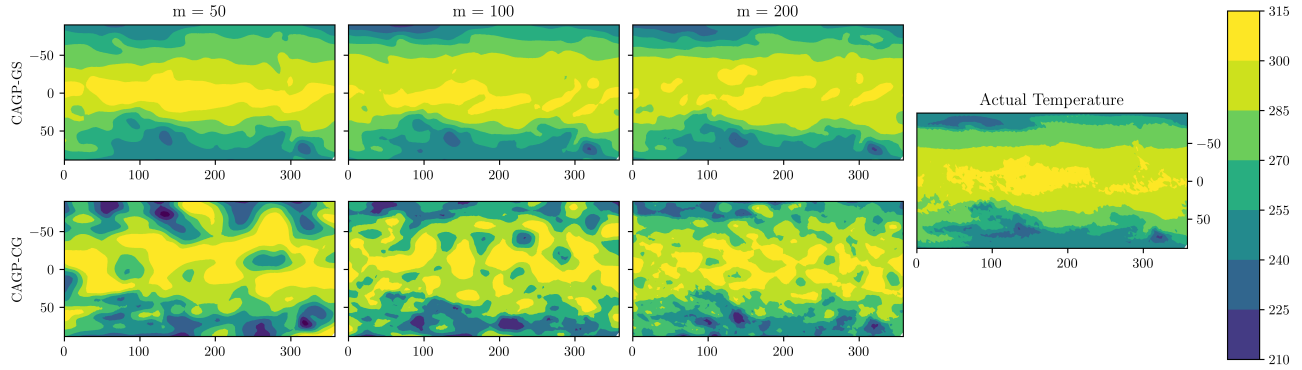Figure S6: RMSE and NLL for the large-scale regression from Section 5.3

Figure S7: Sequences of posterior means for the large-scale regression from Section 5.3, as a function of m for the detrended data. The x and y axes represent longitudes and latitudes respectively, and the contours indicate the temperature in Kelvin.
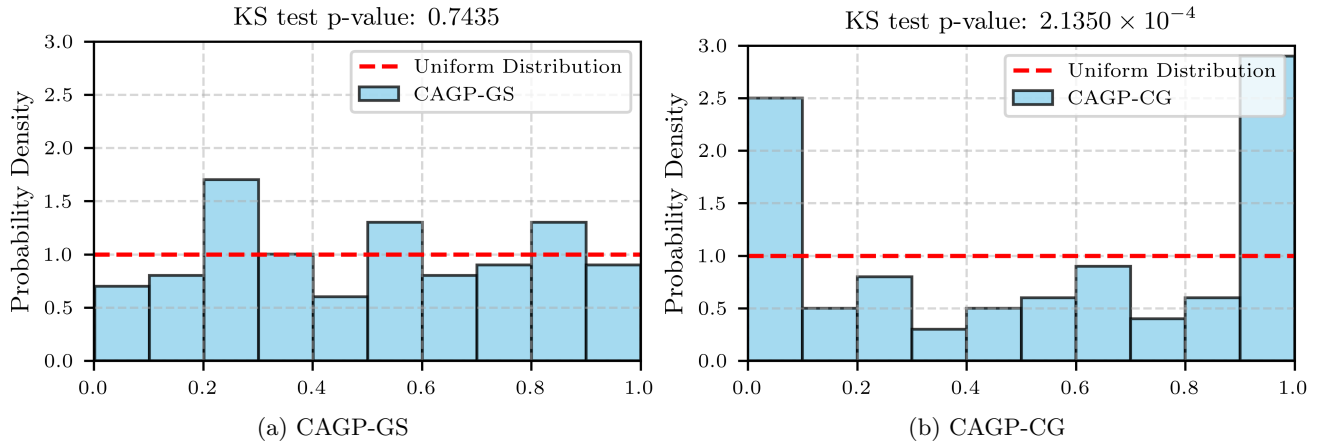

Figure S8: Calibration results for the large-scale regression problem from Section 5.3 for the detrended data.

here unlike the previous section.

We test if the conclusions from Section 5.3 is sensitive to trends in the data. As the global temperature is known to decrease as the latitude increases, using a detrended temperature data is a good way to examine this. We replace the constant prior mean in Section 5.3 with a fitted quadratic polynomial of latitude. Fig. S7 shows the posterior mean for the settings same as Fig. 5. As seen before, we see that the CAGP-GS posterior mean is smoother than the CAGP-CG posterior mean. Fig. S8 shows the results of calibratedness experiment (for the same settings as Fig. S5). Again, the p-values from the Kolmogorov-Smirnov test for uniformity show that CAGP-GS is well-calibrated, while CAGP-CG remains poorly calibrated. Detrending the data does not seem to affect the conclusions from before.

References

Cockayne, J., Graham, M. M., Oates, C. J., Sullivan, T. J., and Teymur, O. (2022). Testing whether a learning procedure is calibrated. *Journal of Machine Learning Research*, 23(203):1–36.

Ipsen, I. C. F. (2009). *Numerical matrix analysis.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA. Linear systems and least squares.

Talts, S., Betancourt, M., Simpson, D., Vehtari, A., and Gelman, A. (2018). Validating bayesian inference algorithms with simulation-based calibration.