Two approaches of Probabilistic Linear Solvers

Georgy Guryev [1] **Samuel Lam** [1]

Abstract

Numerical linear solvers had been an active area of research. The current state of art is the classical Conjugate Gradient (CG) method. The method is still very inefficient in handling linear systems with very high condition number. Our paper studies two approaches to probabilistic linear solvers, in which CG is implemented followed by post-hoc analysis through the Bayesian arguments. With a careful choice of Bayesian model, the uncertainties of the posterior can reflect an actual error of the approximation.

1. Introduction

Solution to the system of equation

$$Ax = b, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is a large symmetric positive definite (spd) matrix and $b \in \mathbb{R}^n$ is a vector, has been a longstanding problem in computational linear algebra. It is the fundamental building block for quadratic optimisation, regression and numerical method differential equations. The classical algorithm of solving such system is the Conjugate Gradient (CG) method developed by the Hestenes and Stiefel in the early 1950's (Trefethen, 1997). It is known that the operation count for m iterations of CG is $\mathcal{O}(m^2n^2)$, and therefore if $m \ll n$ then CG is much faster than other (direct) methods, e.g. QR factorisation. Given that the condition number of A is κ, the number of iterations required for CG to converge is in order of $\mathcal{O}(\sqrt{\kappa})$. It is often the case that κ is large, e.g. the case of kernel matrices in Gaussian process, for which large number of iterations are still required to reach desired accuracy.

It motivates the application of probabilistic linear solvers. The idea is that the iterates of CG is the posterior mean/MAP of a certain Bayesian model. One would therefore hope that uncertainty of Bayesian model (e.g. its covariance) could act as a proxy of actual error of approximation (encode our uncertainty about the solution). If that's the case, one could run the CG with (much) fewer iterations while getting useful information about the actual solution itself (e.g with high probability that the actual solution is close to the current iterate). (Hennig et al., 2015; Cockayne et al., 2019)

Contribution Here we present two approaches of probabilistic linear solver: the (1) solution-based approach and the (2) matrix-based approach. The former puts prior on the actual solution x^*, while the latter put prior on the inverse of matrix A^{-1}. We compare the uncertainties of those Bayesian models against the actual error of the estimation. For a good probabilistic linear solver, we hope these quantities have similar order of magnitude.

1.1. Conjugate Gradient

Let us give a high-level review of the algorithm of conjugate gradient, as well as defining notations for the latter discussion. Recall that CG is one of the Krylov-subspace method, which iteratively obtain an estimate of actual solution x^* by elements in the low-dimensional Krylov-subspace $\mathcal{K}_m = \mathsf{Span}(b, Ab, ..., A^{m-1}b)$, call x_m. Define the A-conjugate inner product as $\langle x, y\rangle_A = x^\top Ay$ and A-conjugate norm as $\|x\|_A = \sqrt{\langle x, y\rangle_A}$. For CG, we hope that the quantity $\|x_m - x^*\|_A$ is minimised. If there exists a A-conjugate basis $\{s_1, ..., s_m\}$ of $\mathcal{K}_m$ (the *search directions*) normalised by the A-conjugate norm, then we have

$$x_m = x_0 + S_m(S_m^\top AS_m^\top)^{-1}S_m^\top(b - Ax_0) \tag{2}$$

where S_m is the matrix containing columns of the A-conjugate basis, and x_0 is the starting value of CG. This results in the following iterative formula

$$x_m = x_{m-1} + s_m s_m^\top r_{m-1} \tag{3}$$

where r_m is the residual $r_m = b - Ax_m$. Indeed, the search directions can be obtained by performing Gram-schmidt on the residuals r_m:

$$\tilde{s}_m = r_{m-1} - \sum_{i=1}^{m-1} \langle s_i, r_{m-1}\rangle_A \, s_i \tag{4}$$

where $\tilde{s}_m$ is the un-normalised search directions which could be normalised by dividing its A-conjugate norm. No-

[1]Department of Electrical, Engineering and Computer Science, Massachusetts Institute of Technology, Massachusetts, USA. Correspondence to: Georgy Guryev <georgy@mit.edu>.

tice that, mathematically, the inner products $\langle s_i, r_{m-1}\rangle_A$ should be zero for $i < m-1$. However in reality when the search directions has very small A-conjugate norm, the inner products are non-zero due to numerical error. If we ignore those inner product then the search directions obtained are no longer A-conjugate (known as the *loss of conjugacy*). To ensure stability of algorithm we perform full Gram-schmidt (Cockayne et al., 2019). In addition, we avoid normalisation of search directions and perform a modified version of Gram-schmidt. This, however, increase the number of operation counts by an order of $\mathcal{O}(m)$.

2. Probabilistic Linear Solvers

2.1. Solution-Based Approach

We begin by discussing the solution-based approach as proposed by Cockayne et al. in 2019 (Cockayne et al., 2019). Despite this is developed after the matrix-based approach, the computations involved are much simpler for the approach. As described, we put a normal prior on the actual solution:

$$\mathsf{x} \sim \mathsf{N}_n(x_0, \Sigma_0) \tag{5}$$

Let's say we use equation (3) to obtain solution estimates x_m using linearly independent search directions s_i, not necessarily A-conjugate, then we observe the quantities $y_m = S_m^\top A x$. Assuming that the matrix computation contains no error, it is therefore sensible to use dirac measure as likelihood on y

$$\mathsf{y}_m \,|\, \mathsf{x} = \delta(\mathsf{y} - S_m^\top A \mathsf{x}) \tag{6}$$

From this we can calculate the joint disribution of x and y_m, given as

$$\begin{bmatrix} \mathsf{x} \\ \mathsf{y}_m \end{bmatrix} \sim \mathsf{N}_{2n}\left(\begin{bmatrix} x_0 \\ S_m^\top A x_0 \end{bmatrix}, \begin{bmatrix} \Sigma_0 & \Sigma_0 A^\top S_m \\ \Sigma_0 A^\top S_m & S_m^\perp A x_0 \end{bmatrix} \right) \tag{7}$$

and therefore

$$\mathsf{x} \,|\, \mathsf{y}_m \sim \mathsf{N}_n(x'_m, \Sigma_m) \tag{8}$$

where

$$x'_m = x_0 + \Sigma_0 A^\top S_m \Lambda_m^{-1} S_m^\top r_0 \tag{9}$$

$$\Sigma_m = \Sigma_0 - \Sigma_0 A^\top S_m \Lambda_m^{-1} S_m^\top A \Sigma_0 \tag{10}$$

$$\Lambda_m = S_m^\top A \Sigma_0 A^\top S_m \tag{11}$$

In particular when Σ_0 is chosen to be A^{-1} and s_i the A-conjugate normalised basis of $\mathcal{K}_m$, then we recover the original CG given in equation (3). In fact, for the general case when s_i are Q-conjugate normalised basis of $\mathcal{K}'_m$, where $Q = A\Sigma_0 A^\top$ and $\mathcal{K}'_m$ being Krylov subspace $\mathsf{Span}(b, Qb, ..., Qb^{m-1})$, then we recover CG for the system

$$Q\tilde{x} = b, \quad \tilde{x} = (\Sigma_0 A^\top)^{-1} x \tag{12}$$

i.e. the original system $Ax = b$ right pre-conditioned with the pre-conditioner $P = (\Sigma_0 A^\top)^{-1}$. Therefore we can incorporate our knowledge of matrix A (e.g. sparsity) by choosing appropriate Σ_0 for preconditioning.

Choice of Preconditioners Some quick cases include:

- $\Sigma_0 = A^{-1}$, for which we recover the original CG. Notice we get no information about Σ_m, hence uncertainty, because computation of Σ_m involves direct inverse of A^{-1}.
- $\Sigma_0 = (A^\top A)^{-1}$ (the *natural prior*), for which Q is now identity and $\tilde{x} = A^{-1}x$. After one iteration the algorithm should immediate get the exact result. However, as one might expect, this is not practical since it involves obtaining inverting $A^\top A$ for uncertainty quantification and A. Nevertheless, if we replace A by P' with P' easy to be inverted and sufficiently closed to A, then there is hope that such preconditioning would speed up our calculation.
- $\Sigma_0 = I$, for which $Q = A^2$ and $\tilde{x} = A^{-1}x$. This has little significance in numerical analysis.

We will discuss this further in the case studies.

2.2. Matrix-Based Approach

We now turn to the matrix-based approach as proposed by Hennig in 2015. (Hennig, 2015; Wenger & Hennig, 2020) Instead of putting a prior on the solution x^*, we put a prior on the matrix $H := A^{-1}$. It starts with the general problem of having symmetric matrix B unknown and we want to infer B by a Bayesian approach. We put a prior [1] of

$$\mathsf{B} \sim \mathsf{N}_{n\times n}(B_0, W_0^{\mathsf{B}} \otimes_{\mathsf{sym}} W_0^{\mathsf{B}}) \tag{13}$$

where $\otimes_{\mathsf{sym}}$ is the symmetrized Kronecker product [2]. We then make observations $y_i = As_i$, where s_i are again linearly independent search directions). Letting Y_m be matrix with columns As_i, we have the delta-likelihood

$$\mathsf{Y}_m \,|\, \mathsf{B} \sim \delta(\mathsf{Y} - \mathsf{B}S_m) \tag{14}$$

[1] It is worth reminding that equation (12) means that if we stack the rows of B as a very long vector (i.e. vectorizing B), than the vector follows multivariate normal distribution with mean vector B_0 stacked in the same way as B, and covariance matrix being $W_0^{\mathsf{B}} \otimes_{\mathsf{sym}} W_0^{\mathsf{B}}$.

[2] $W_0^{\mathsf{B}} \otimes_{\mathsf{sym}} W_0^{\mathsf{B}}$ means the matrix $\Gamma(W_0^{\mathsf{B}} \otimes W_0^{\mathsf{B}})\Gamma^\top$, where Γ sends vectorized matrix B to the vectorization of matrix $(B + B^\top)/2$

From this we can obtain the posterior distribution

$$\mathsf{B} \,|\, \mathsf{Y}_m \sim \mathsf{N}_{n\times n}(B_m, W_m^{\mathsf{B}} \otimes_{\mathsf{sym}} W_m^{\mathsf{B}}) \tag{15}$$

where

$$B_m = B_0 + \Delta_0^{\mathsf{B}} U^\top + U(\Delta_0^{\mathsf{B}})^\top - US^\top \Delta_0^{\mathsf{B}} \tag{16}$$
$$W_m^{\mathsf{B}} = W_0^{\mathsf{B}}(I_n - SU_m^\top) \tag{17}$$
$$\Delta_0^{\mathsf{B}} = Y - B_0 S_m \tag{18}$$
$$U_m = W_0^{\mathsf{B}} S_m (S_m^\top W_0^{\mathsf{B}} S_m)^{-1} \tag{19}$$

Such problem can be applied to both matrices A and H. In fact, the formulae are almost equivalent, instead the role of S_m and Y_m exchanges. We can then obtain the posterior distribution matrix vector product Hx by the following formula:

$$(I_n \otimes x)\,\mathsf{H} \,|\, \mathsf{Y}_m \sim \mathsf{N}_n(H_m x, \Sigma_{m,x}) \tag{20}$$
$$\Sigma_{m,x} = \frac{1}{2}(W_m^{\mathsf{H}} x^\top W_m^{\mathsf{H}} x + (W_m^{\mathsf{H}} x)(x^\top W_m^{\mathsf{H}})) \tag{21}$$

In particular the posterior of x^* is $(I \otimes b)\mathsf{H}_m \,|\, \mathsf{S}$. The computations here are far more complicated than the previous method. Henning's et al. proposed the following: $A_0 = H_0^{-1} = \alpha I$ for $\alpha > 0$, and

$$W_0^{\mathsf{A}} = P^A_{S_m} + P^{I_n}_{S_m^\perp} \Phi_n P^{I_n}_{S_m^\perp}, \phi > 0 \tag{22}$$
$$W_0^{\mathsf{H}} = P^{H_0}_{Y_m} + P^{I_n}_{Y_m^\perp} \Psi_n P^{I_n}_{Y_m^\perp}, \psi > 0 \tag{23}$$

where P_S^B represents that projection matrix onto the subspace $\mathsf{Span}(S)$ with respect to B-conjugate norm and $P^B_{S^\perp}$ represents that projection matrix onto the orthogonal complement $\mathsf{Span}(S)^\perp$. Specifically,

$$P_S^B = BS(S^\top AS)^{-1}S^\top B, P^B_{S^\perp} = I_n - P_S^B \tag{24}$$

The fact that projection matrices are symmetric and idempotent simplifies a lot of computation. In particular, one can show that $A_m^{-1}Y = H_m Y$ for all m, A_m remains spd, and most importantly $H_m b$ resembles the CG. Therefore, this method is almost a solution-based approach with appropriate choice of covariance matrix.

It is important to note, in practice, the x_m are computed first from the classical CG to obtain the search directions s_i and y_i. We then use those to compute the matrices W_m^{H} for posterior covariance $\Sigma_{x,m}$. It quantifies the uncertainty, i.e. the error of approximation.

Choice of Φ and Ψ It remains for us to choose appropriate Φ and Ψ for correctly calibrating uncertainties. Assuming, for simplicity, that Φ and Ψ are diagonal matrices. They then represents the weighting of how much "unknown" information should be contributed to the uncertainty of predicting A and H respectively (which are the prior covariances of A and H respectively). Naturally, we would like use the smallest (largest) $n-m$ eigenvalues of A (H) for the diagonals of Φ (Ψ), but obtaining actual eigenvalues for both matrices A and H are much harder than inverting them.

Fortunately, the Rayleigh quotient

$$a(i) = \frac{s_i^\top A s_i}{s_i^\top s_i} \tag{25}$$

is a good proxy of eigenvalues. (Nocedal, 2006) [3] Indeed we get access to the $a(i)$ for i from 1 up to m. Hennig therefore suggested extrapolating $a(i)$ to estimate the remaining Rayleigh quotient. Assume we now obtain the estimates of the remaining Rayleigh quotients, or the $\hat{a}(i)$'s with $i > m$, then we set $\Phi = \phi I_n$, where ϕ is the empirical averages of the remaining estimated eigenvalues

$$\phi = \frac{1}{n-m}\sum_{i=m+1}^{n} \hat{a}(i) \tag{26}$$

This choice of Φ is motivated by the fact that we have no information about the unobserved subspace $\mathsf{Span}(S)^\perp$. (Hennig, 2015; Cockayne et al., 2019; Wenger & Hennig, 2020) With similar reason, we also set $\Psi = \psi I_n$ for some $\psi > 0$, and Hennig et. al. proposed the choice of $\psi = \phi^{-1}$.

The way we extrapolate depends on the structure of A, e.g. how does the eigenvalue decay. One can then fit a regression model, either least square or Gaussian process with appropriate basis function for extrapolation. One might worry about fitting of regression models significantly increases the amount of computation, but we believe this is not the case since m is much smaller than n. We will discuss the mechanism in greater detail in next section.

As a final remark, since we have no idea about the largest eigenvalues of A_m^{-1}, we directly assume $\Psi = \psi I_n$ with ϕ being defined above. In such case, ψ will definitely be an overestimate, but latter numerical experiment shows this value of ψ is good enough.

Preconditioning One should note that there is not a way as trivial as solution-based approach to carry out analysis on a linear system. (Cockayne et al., 2019) It is unclear how this would affect previous theoretical analysis.

3. Case Studies

We therefore study the performance of two approaches for two cases of A: (1) A being a kernel matrix in a Gaussian

[3] A precise statement can be seen in (Nocedal, 2006) pg. 113-115, in which they study the convergence of ordinary CG.

process, and (2) A being Laplace matrix used in solving Poisson equation. The structures of those matrices are very different, the former being dense and diagonally dominant, the latter being banded.

3.1. Gaussian Process

Recall that a Gaussian Process $f(x) \sim \mathsf{GP}(m(x), k(x, x'))$ is characterised by its mean function $m(x)$ and covariance kernel $k(x, x')$. (Rasmussen, 2006) One commonly used covariance kernel is the radial basis function (RBF) kernel, having the form

$$k(x, x') = \exp\left(-\frac{1}{2\eta^2}\|x - x'\|^2\right) \tag{27}$$

We attempt to fit a Gaussian process with zero mean and RBF kernel onto the Sarcos data set [4]. In particular, we use the 21 features of data set (containing positions, velocities and accelerations of robot arm) to predict the torque experienced by the first joint of robot arm. We select the first 1000 input-output pair for our test, and form the covariance matrix:

$$A = k(X, X) + \sigma^2 I \tag{28}$$

where $k(X, X)$ is the matrix representing the RBF covariance kernel evaluated at the data. We then try to solve $Ax = b$, where b is the torque of the first joint. We can then compute the posterior mean of torque $k(X, X)A^{-1}b$, as well as posterior covariance, even though this will not be presented in our result.

One would have hoped that the eigenvalues, hence also the Rayleigh quotients, should decay exponentially as a result of Weyl's inequality. (Wenger & Hennig, 2020) However, the following figure reveal that relationship between $\ln a(i)$ and i is not perfect linear as we might have expected.

To account such deviation, we propose two procedures. The easiest one is to perform a polynomial regression on the Rayleigh quotients, ignoring the largest one. Here we choose to discard the 20% largest Rayleigh quotient. A probabilistically more appealing approach is to fit a (much smaller-scaled) Gaussian process on the values of $a(i)$ with linear mean: [5]

$$\ln|a(i)| \,|\, \mathsf{A}, \mathsf{Y} \sim \mathsf{GP}(\theta_0 - i\theta_1, k(i, j)) \tag{29}$$

where $k(i, j)$ represents RBF kernel evaluated at the number of iterations i, j. We then obtain point estimates $\hat{a}(i) := \exp(\mathbb{E}(\ln|a(i)| \,|\, \mathsf{A}, \mathsf{Y}))$. The benefit is that it does not require manual selection of amount of data to be discarded. It captures the change of rate of decay as i increases, and the samples from the posterior is guaranteed to be smooth.

[4] Available from (Rasmussen, 2006)

[5] we could have also used polynomial mean, but further increasing flexibility may lead to overfitting.

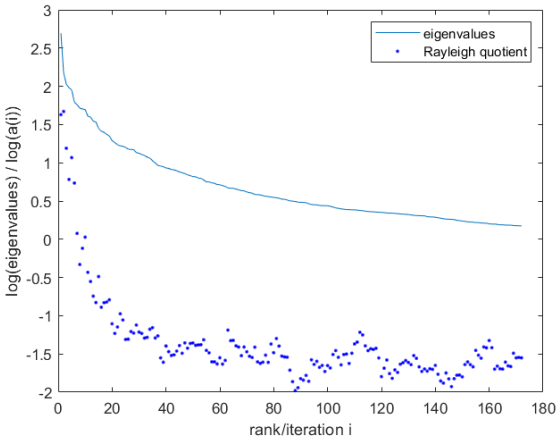

Figure 1. Plot of log of Rayleigh quotient as defined in (24) with s_i being the search direction from CG, as well as eigenvalues of matrix A sorted in descending order. Notice the change of decay rate at $i \approx 40 \approx 180/5$, and the clear underestimation of the true eigenvalues by the Rayleigh quotients.

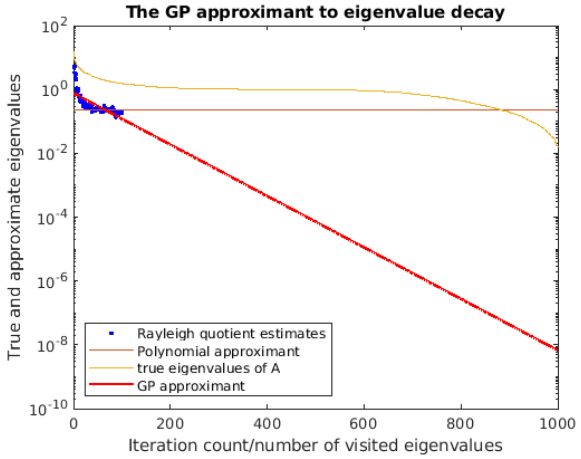

Figure 2. The extrapolation of the eigenvalue decay with the GP and polynomial function. GP fitted to the CG Rayleigh quotients after 100 iterations; the polynomial was fitted with the regularized linear regression.

We implement both ways for the matrix-based approach. It is confirmed in Figure 3 that CG does converge. The main problem is, whether the uncertainties given by the covariance matrices of posterior distribution succesfully quantify the error, i.e. 2-norm between x_m and x^*. We would hope those quantities have approximately same order.

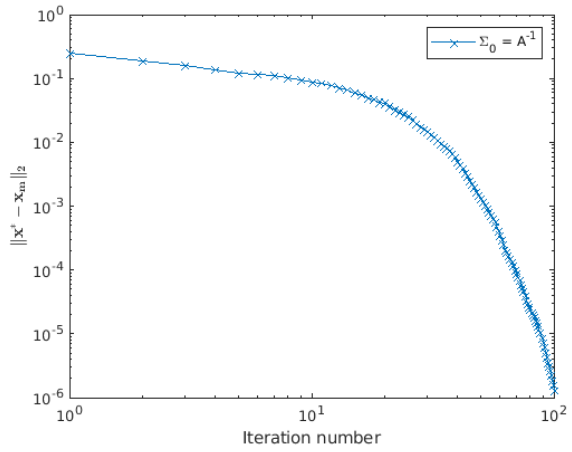

Figure 3. The convergence of the solution error in 2-norm $\|x_m - x^*\|_2$ for the covariance prior $\Sigma_0 = A^{-1}$.

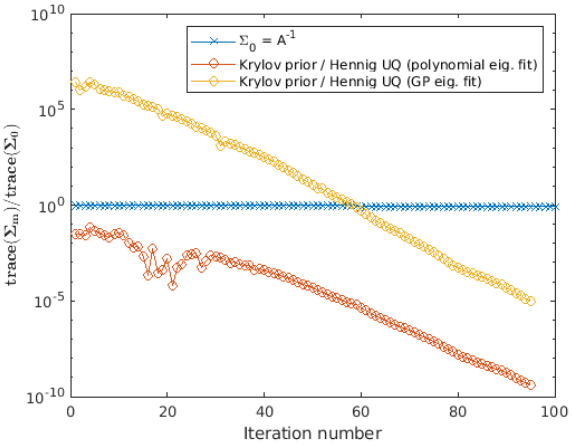

Figure 4. We plot the decay of uncertainties (log scale). The blue curve represents the trace of Σ_m divided by Σ_0 in the solution-based approach, while the purple and green curves represents the trace of $\Sigma_{m,x}$ divided by $\Sigma_{0,x}$ from the calibrated matrix-based approach, with ψ computed in two different ways.

Figure 4 shows the decay of trace of posterior covariance in log-scale. As we can see, the trace of posterior covariance for solution-based approach does not decay, so it is clear that such covariance would not give a good uncertainty quantification. We will explain why that is the case in the next subsection.

The quality of the quantification of error can be further studied by the quantity $\Sigma_x^{-1/2}(x_m - x^*)$, which should be approximately standard normal if the covariance gives good uncertainty quantification. [6] In particular, if the distribution of entries are too wide, then Σ_m underestimates the actual approximation error, and vice versa.

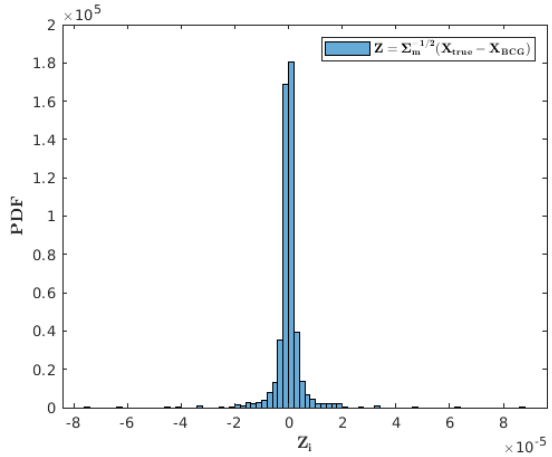

Figure 5. Predicting the solution after 100 iterations of CG on the Sarcos model. The plor investigates the standarized variable $z = \Sigma_x^{-1/2}(x^* - x_m)$. The posterior covariance matrix is calibrated following the solution-based approach. Notice the scale of the x-axis is of order 10^{-5}.

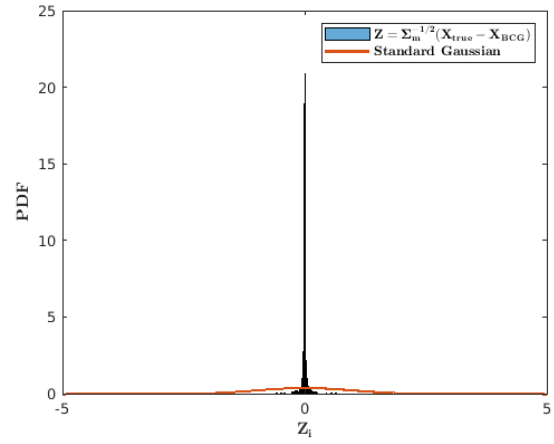

Figure 6. Predicting the solution after 100 iterations of CG on the Sarcos model. The plor investigates the standarized variable $z = \Sigma_m^{-1/2}(x^* - x_m)$. The posterior covariance matrix is calibrated following the matrix-based approach with ψ estimated by Gaussian process. Notice the histogram is wider compare to the one from solution-based approach, but still far from matching the standard normal.

[6] Here $\Sigma_m^{-1/2}$ is the matrix square root, obtained by first obtain eigenvalue decomposition $QDQ^\top$ then replace entries of D by its square root.

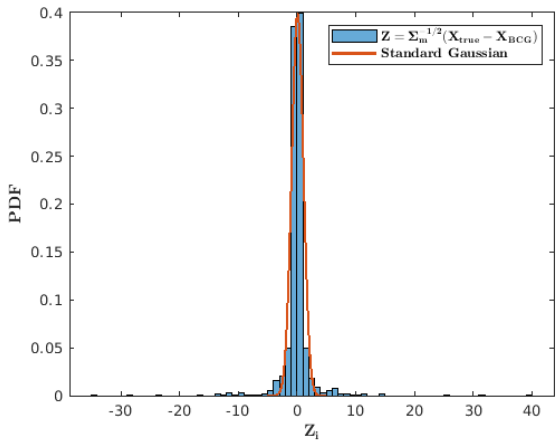

Figure 7. Predicting the solution after 100 iterations of CG on the Sarcos model. The plor investigates the standarized variable $z = \Sigma_x^{-1/2}(x^* - x_m)$. The posterior covariance matrix is calibrated following the matrix-based approach with ψ estimated using polynomial fit. The resulting standard variable is distributed according to the standard normal distribution what implies that the posterior covariance Σ_m is well-calibrated.

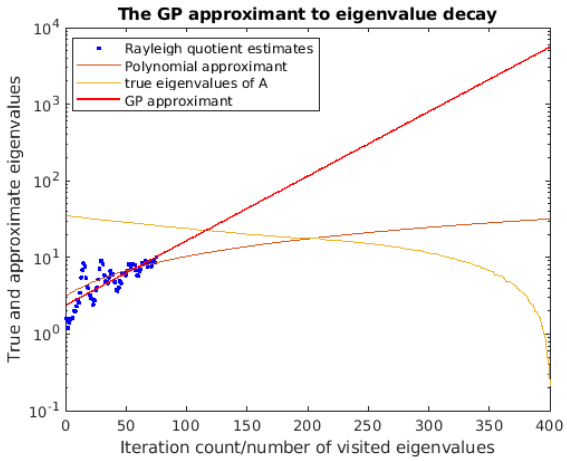

Figure 8. The GP and polynomial approximation of the eigenvalue.

As expected, uncertainty quantification from the solution-based approach had overestimated the actual approximation error, while uncertainty quantification from matrix-based approach is satisfactory if we perform polynomial regression to estimate ψ. It worth noticing that matrix-based approach had underestimated actual approximation error if we use Gaussian process to estimate ψ. This is perhaps because the mean function of GP decreases without bound, and we will see $\hat{a}(i)$ decreases to zero rapidly. Taking average over all $\hat{a}(i)$ for ϕ might significantly underestimate the actual error. Fitting GP may not be appropriate in this setting.

3.2. Poisson Equation

Let us also consider the performance of the two approaches in solving partial differential equations. Specifically, we consider the Poisson equation over a square region $\Omega = [0,1]^2$:

$$\begin{cases} -\Delta u = f(x,y) & (x,y) \in \Omega^\circ \\ u(x,y) = 0 & (x,y) \in \partial\Omega \end{cases} \tag{30}$$

where f is randomly in the selected in the sense of each values of $f(x,y)$ are iid $\mathsf{N}(0,1)$ selected. We discretise the system by replacing $-\Delta$ by Laplacian matrix A and letting b be the vector of values of $f(x,y)$. The problem is then reduced to solving $Ax = b$. (Larsson, 2003)

It worth noting that the Rayleigh quotient does not decay along the iteration. Instead, it grow linearly. Therefore to predict the ω one should perform linear GP regression with linear mean, as in Figure , i.e.

$$a(i) \,|\, \mathsf{A}, \mathsf{Y} \sim \mathsf{GP}(\theta_0 + i\theta_1, k(i,j)) \tag{31}$$

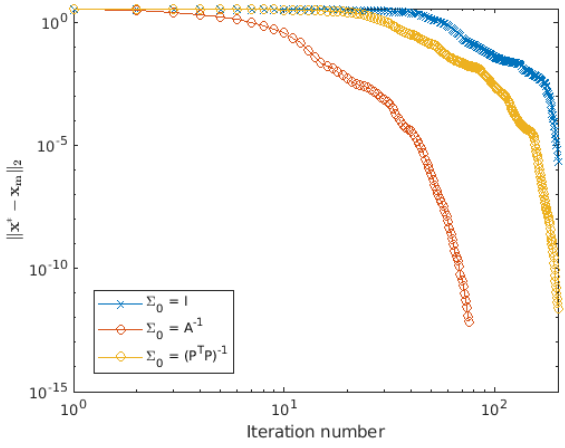

Figure 9. Once again, the convergence of 2-norm $\|x_m - x^*\|$ are plotted. Various choices of Σ_0 have been used. Notice the matrix-based approach returns the same x_m as classical CG, i.e. the x_m for the case when $\Sigma_0 = A^{-1}$.

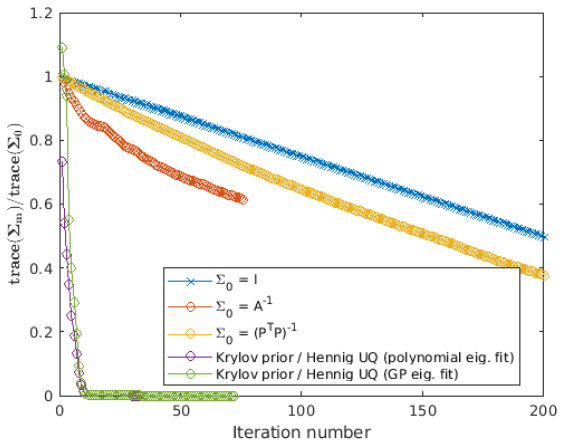

Figure 10. We plot the decay of uncertainties. The red, yellow and blue curves represent the trace of Σ_m divided by Σ_0 in the solution-based approach, while the purple and green curves represents the trace of $\Sigma_{m,x}$ divided by $\Sigma_{m,0}$ from the matrix based approach, with ψ computed in two different ways.

Effect of Preconditioner Here we run the solution-based approach for three choice of Σ_0: the trivial choice $\Sigma_0 = I$, the choice which recovers classical CG $\Sigma_0 = A^{-1}$, and the choice $\Sigma_0 = (P^\top P)^{-1}$, where P is the sparsified Cholesky decomposition formed by pulling respective diagonal entries from A. We haven't applied any conditioner for matrix-based approach in our experiment.

We once again notice exponential decay of the 2-norm $\|x_m - x^*\|$ and linear decrease of trace of covariance when the solution-based approach is used. This is confirmed theoretically in (Cockayne et al., 2019). As a result, the uncertainty from the solution-based approach is a clear over-estimation of actual error. On the other hand, the trace of covariance for matrix-based approach decreases rapidly, which is a sign of better caliberation of uncertainty.

It worth noting that the $\|x - x^*\|$ decays the quickest when we choose $\Sigma_0 = A^{-1}$, and slowest for $\Sigma_0 = I$. However, it is impractical to run solution-based method for the choice $\Sigma_0 = A^{-1}$ to obtain Σ_m. For practical implementation of solution-based method, a pre-conditioned prior must be used.

4. Conclusion

From the discussion, we see how classical numerical method can be interpreted in a Bayesian framework. One can therefore interpret the probabilistic linear solvers as classical numerical methods with post-hoc analysis. It is clear from the results that the uncertainty quantifications from the posterior covariance are very conservative and therefore not practical, especially for the case when solution-based approach with Σ_0 equals to A^{-1} or natural prior is used. Nevertheless, the matrix-based approach does produce promising uncertainty results.

It worth noting that further analysis is needed to talk about the speed of the uncertainty quantification. In particular, one should study the extra amount of computation needed to compute the matrix products. Nevertheless, we are sure that the probabilistic linear solvers deserve more attentions.

All codes are available at https://bitbucket.org/georgy_guryev/bayescg/src/master/.

References

Cockayne, J., Oates, C. J., Ipsen, I. C., and Girolami, M. A bayesian conjugate gradient method (with discussion). *Bayesian analysis*, 14(3), 2019. ISSN 1936-0975.

Hennig, P. Probabilistic interpretation of linear solvers. *SIAM journal on optimization*, 25(1):234–260, 2015. ISSN 1052-6234.

Hennig, P., Osborne, M. A., and Girolami, M. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society. A, Mathematical, physical, and engineering sciences*, 471(2179):20150142–20150142, 2015. ISSN 1364-5021.

Larsson, S. S. *Partial differential equations with numerical methods*. Texts in applied mathematics 43. Springer, Berlin, New York, 2003. ISBN 3540017720.

Nocedal, J. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, New York ;, 2nd ed. edition, 2006. ISBN 9780387303031.

Rasmussen, C. E. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. ISBN 9780262256834.

Trefethen, L. N. L. N. *Numerical linear algebra*. Society for Industrial and Applied Mathematics, Philadelphia, 1997. ISBN 9780898713619.

Wenger, J. and Hennig, P. Probabilistic linear solvers for machine learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6731–6742. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/4afd521d77158e02aed37e2274b90c9c-Paper.pdf.